An Efficient Watermarking Method for Latent Diffusion Models via Low-Rank Adaptation

Dongdong Lin, Yue Li, Benedetta Tondi, *Member, IEEE*, Bin Li, *Senior Member, IEEE*, Mauro Barni *Fellow, IEEE*

Abstract—The rapid proliferation of deep neural networks (DNNs) is driving a surge in model watermarking technologies, as the trained deep models themselves serve as intellectual properties. The core of existing model watermarking techniques involves modifying or tuning the models' weights. However, with the emergence of increasingly complex models, ensuring the efficiency of watermarking process is essential to manage the growing computational demands. Prioritizing efficiency not only optimizes resource utilization, making the watermarking process more applicable, but also minimizes potential impacts on model performance. In this letter, we propose an efficient watermarking method for latent diffusion models (LDMs) which is based on Low-Rank Adaptation (LoRA). We specifically choose to add trainable low-rank matrices to the existing weight matrices of the models to embed watermark, while keeping the original weights frozen. Moreover, we also propose a dynamic loss weight tuning algorithm to balance the generative task with the watermark embedding task, ensuring that the model can be watermarked with a limited impact on the quality of the generated images. Experimental results show that the proposed method ensures fast watermark embedding and maintains a very low bit error rate of the watermark, a high-quality of the generated image, and a zero false negative rate (FNR) for verification.

Index Terms—Model Watermarking, Latent Diffusion Model, Low-Rank Adaptation, Efficient Watermarking, Dynamic Loss Weight Tuning.

I. INTRODUCTION

N recent years, text-to-image models have sprinted ahead in the techniques of image generation. Among these, Latent Diffusion Models (LDMs) have emerged as frontrunners. Methods like Stable Diffusion [1]–[3] and Diffusion Transformer [4], [5] have shown their dominance in generating high-quality images with impressive capabilities, leading to commercial applications such as LiblibAI and SeaArt. These models typically require extensive data and substantial computational resources for training, which has shifted the focus towards preventing unauthorized use and safeguarding intellectual property rights [6]–[13].

A direct approach to achieve the goal of model identification and ownership verification is through *model watermarking*, which involves embedding an identity message (watermark)

Dongdong Lin and Bin Li are with the Guangdong Provincial Key Laboratory of Intelligent Information Processing and the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China (e-mail: dongdonglin8@gmail.com; libin@szu.edu.cn). *Corresponding author: Bin Li*.

Yue Li is with Huaqiao University, and also with Xiamen Key Laboratory of Data Security and Blockchain Technology, Xiamen 361021, China (e-mail: liyue_0119@hqu.edu.cn)

Benedetta Tondi and Mauro Barni are with the Department of Information Engineering and Mathematics of the University of Siena, Italy (e-mail:benedetta.tondi@unisi.it; barni@diism.unisi.it).

into the model itself. In this letter, we focus on invisible watermarking. Without loss of generality, in the following, we broadly categorize the existing methods for LDM watermarking into three main categories.

Full Parameter Embedding: The watermark is embedded by updating all the model's parameters. The methods utilize specially crafted datasets (see WatermarkDM [14]), or design triggers to embed watermark during training (see Concept Watermarking [15], DiffStega [16], Safe-SD [17], Latent Watermark [18], and DIAGNOSIS [19]), or through a pre-trained watermark encoder-decoder to update the model's parameters (see Stable Signature [20] and ProMark [21]).

Additional Parameter Embedding: The watermark is embedded through additional trainable parameters, which can be added to the original model's parameters in parallel (see AquaLoRA [22]), or inserting extra layers between the original layers of the model (see WMAdapter [23] and LaWa [24]).

Latent Space Embedding: This category includes methods where the model's parameters remain unchanged. Instead, a watermark encoder-decoder is trained and plugged into the model to modify the latent features in LDMs, thereby influencing the output to embed the watermark (see RoSteALS [25] and PiGW [26]).

As models increase in size, Full Parameter Embedding for LDM watermarking becomes prohibitively expensive due to the vast number of parameters. For example, the latest SDXL model [2] has 2.6 billion parameters in its UNet, requiring days of training and substantial memory. Latent Space Embedding needs a meticulously designed watermark encoder-decoder, and modify the latent feature will change the content of the generative images, which may not be commonly applicable. Additional Parameter Embedding is a more efficient approach, as it reduces the computational burden and ensures the watermarking process is fast and efficient, which is also adopted in our method. Additional issue is, in methods such as Stable Signature [20], the watermark embedding process may degrade the model's generative performance. In practice, we hope the watermark can be embedded with limited impact on the generative task. However, using fixed hyper-parameters to manage the watermark loss and generative loss may lead to an imbalance between generative task and watermark task. To sum up, we raise two research questions (RQs):

RQ1: How to ensure the efficiency of the watermark embedding process?

RQ2: How to balance the performance of the generative task and the watermark embedding task?

On one hand, to answer RQ1, we propose an Efficient

Watermark in latent diffusion model via Low-Rank Adaptation (EW-LoRA). For a well-trained LDM, the watermark embedding is supported with the technique of parameterefficient fine-tuning (PEFT), where additional parameters are added for optimization (i.e., LoRA [27]). Different from WMAdapter [23] and LaWa [24], the additional parameters can be merged into the original parameters by simple addition without extra storage burdens. AquaLoRA [22] introduces LoRA to the denoising module in LDM with a large rank, where the parameters of LoRA are relatively large. It chooses to add LoRA to attention layers, which directly impacts the generative images. Unlike AquaLoRA, we choose to add LoRA to specific network layers in the VAE decoder, and a smaller rank is acceptable, significantly reducing the computational load. On the other hand, we propose a dynamic loss weight tuning algorithm for training such that the watermark is embedded with limited impacts on the generative images under an acceptable quality, to answer RQ2.

II. METHODOLOGY

A. Problem Formulation

Threat Model: We assume the model is accessible via an API, and we consider that the generated images may suffer from unauthorized use, such as post-processing attacks. Malicious users may also try to remove or overwrite the watermark to evade detection or claim authorship [28].

Protector's Goal: The protector's goal is to protect the intellectual property of the model by addressing the two research questions outlined in the Introduction. Specifically, we aim to enable ownership inference from the model's output in cases of unauthorized use following the model's release.

B. Efficient Watermarking Method for LDM

This letter focuses on the model watermarking methods that involves parameter updates. The watermarked model is optimized, and has new parameters $\tilde{\theta}$, which can be seen as adding extra parameters $\Delta\theta$ to its original parameters θ :

$$\tilde{\theta} = \theta + \Delta\theta,\tag{1}$$

and it has the same format as LoRA [27]. Inspired by this, we use low-rank matrices B and A to replace the extra parameters $\Delta\theta$. Furthermore, the number of parameters can be reduced by choosing proper layers in the LDM, since layers near the output typically handle more abstract and high-level features. Applying LoRA closer to the output layers directly affects these high-level features, making task-specific adjustments more effective.

As illustrated in Figure 1, inside an LDM, the VAE decoder D is used to decode the denoised latent feature \tilde{z}_0 produced by the model, to generate the output image \tilde{x} . In our method, we choose the VAE decoder D as the target to introduce extra parameters, as previous works [20], [23] did. Specifically, we choose the parameters in the upsampler convolutional layers of the VAE decoder to embed the watermark, since they are close to the output layer and they increase the number of the

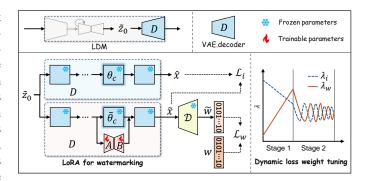


Fig. 1. The working flowchart of the proposed method. In an LDM, the parameters in VAE decoder D are chosen for introducing LoRA. The LoRA parameters A and B are trained with a pre-trained watermark decoder \mathcal{D} . During training, a dynamic loss weight tuning algorithm (DLWT) is used to balance the generative task and the watermark embedding task.

features, which can be easy to introduce watermark features. The generated image can be denoted as

$$\tilde{x} = D(\tilde{z}_0; \tilde{\theta}_c) = D(\tilde{z}_0; \theta_c + \Delta \theta_c), \tag{2}$$

here D is the VAE decoder, \tilde{z}_0 is the denoised latent feature. θ_c and $\tilde{\theta}_c$ are the parameters of the chosen layers in the clean D and the corresponding watermarked one. During training, θ_c are frozen and only $\Delta\theta_c$ are trainable, and the extra parameters $\Delta\theta_c$ are approximated by two low-rank matrices A and B:

$$\Delta\theta_c = \alpha B A,\tag{3}$$

then the total loss for training LDM watermarking is

$$\mathcal{L} = \lambda_i \mathcal{L}_i(\tilde{x}, \hat{x}) + \lambda_w \mathcal{L}_w(\tilde{w}, w), \tag{4}$$

where \mathcal{L}_i is the generative loss, \hat{x} is the image generated from a clean VAE decoder $\hat{x} = D(\tilde{z}_0, \theta_c)$. \mathcal{L}_w is the watermark loss, $\tilde{w} = \mathcal{D}(\tilde{x})$ is the extracted watermark and w is the pre-defined watermark, where \mathcal{D} is a pre-trained watermark decoder. λ_i and λ_w are the loss weights to manage the balance. For generative loss \mathcal{L}_i , we choose Watson-VGG perceptual loss [29], [30] as used in [20], and the watermark loss is the binary cross entropy (BCE) loss:

$$\mathcal{L}_{w} = \mathbb{E}\left[\frac{1}{n}\sum_{j=1}^{n}\tilde{w}^{(j)} + (1 - w^{(j)})\log(1 - \tilde{w}^{(j)})\right], \quad (5)$$

where n is the length of the watermark.

In our method, we follow the same training strategy as Stable Signature [20] did. The watermark decoder \mathcal{D} is trained by introducing a noise layer, and is fixed to jointly finetune the LDM for watermarking.

C. Dynamic Loss Weight Tuning

In our method, we further propose a dynamic loss weight tuning algorithm (DLWT) to address the challenge of the balance for watermarking task and generative task. As shown in Algorithm 1, in each training iteration, DLWT adjusts the loss weights λ_i and λ_w based on the current PSNR (Peak Signal-to-Noise Ratio) and ACC (bit-wise accuracy). By defining a PSNR threshold ρ and a target ACC μ , DLWT handles four cases with the current PSNR ρ_t and ACC μ_t :

Algorithm 1 Dynamic Loss Weight Tuning

Require: PSNR ρ_t , ACC μ_t ; λ_i and λ_w ; PSNR threshold ρ ; target ACC μ ; A damping factor γ ; A step size η

- 1: $\Delta_{\rho} \leftarrow \rho \rho_t, \ \Delta_{\mu} \leftarrow \mu \mu_t$
- 2: if $\rho_t > \rho$ and $\mu_t < \mu$ then
- $\lambda_w \leftarrow \lambda_w + \gamma$
- 4: else if $\rho_t < \rho$ and $\mu_t \ge \mu$ then
- $\lambda_i \leftarrow \lambda_i + \gamma$

6: else if
$$\rho_t < \rho$$
 and $\mu_t < \mu$ then
7: $\lambda_i \leftarrow \lambda_i + \gamma \cdot \left(\frac{\Delta_{\rho}}{\rho}\right), \ \lambda_w \leftarrow \lambda_w + \gamma \cdot \left(\frac{\Delta_{\mu}}{\mu}\right)$

- 8: else
- $\lambda_i \leftarrow \max(\lambda_i \eta \gamma, 0), \ \lambda_w \leftarrow \max(\lambda_w \eta \gamma, 0)$
- 10: end if
- 11: **return** λ_i , λ_w
 - If $\rho_t > \rho$ and $\mu_t < \mu$, increase the watermark loss weight λ_w to strengthen the influence of the watermark.
 - If $\rho_t < \rho$ and $\mu_t \ge \mu$, increase the image loss weight λ_i to improve image quality.
 - If $\rho_t < \rho$ and $\mu_t < \mu$, tunes both the image loss weight λ_i and the watermark loss weight λ_w simultaneously based on the changes in PSNR and ACC.
 - If none of the above conditions are met, gradually decrease the loss weights λ_i and λ_w .

Figure 1 illustrates the general trend of how our algorithm manages the loss. The algorithm operates in two stages. Initially, we set $\lambda_i = 1$ and $\lambda_w = 0$ to ensure optimal generative quality. At Stage 1, the algorithm decrease λ_i as the PSNR exceeds a predefined threshold and increases λ_w as the ACC falls below the target. At Stage 2, the algorithm adjusts the loss weights—decreasing them if they exceed target values and increasing them if they fall below.

D. Watermark Verification

For watermark verification, two tasks are required: i) Watermark Validation and ii) Reliability Test. In Watermark Validation, ACC is calculated by comparing extracted watermark \tilde{w} with the predefined watermark w. To assess reliability, hypothesis testing [20], [31] is applied. Based on a threshold for a given false positive rate (FPR), the verifier computes the corresponding false negative rate (FNR). We use FNR@tFPR as a performance metric for *Reliability Test*, with t = 0.5% in this letter.

III. EXPERIMENTS

A. Watermark Embedding and Comparison with Existing Methods (RQ1)

In our experiments, we measure the number of trainable parameters (Params), bit-wise accuracy (ACC), PSNR for image quality in size of $256 \times 256 \times 3$, the achievement time when reached 99% validation accuracy (AT-99), and FNR@0.5%FPR for Reliability Test. We also validate EW-LoRA's feasibility on different LoRA techniques, including PiSSA [32] and LoHA [33]. PiSSA initializes matrices A and B with the principal components of θ to accelerate training.

TABLE I PERFORMANCE COMPARISON OF DIFFERENT LDM WATERMARKING METHODS

Method	Target	Params	ACC	PSNR	AT-99	N.5P
AquaLoRA [22]	UNet	135.6595	96.35	-	665 [†]	0
S.S. [20]	VAE.dec.	49.4902	100	29.39	16.56	0
WMA. [23]	VAE.dec.	1.3304	99.87	32.67	5.27	0
LaWa [24]	VAE.dec.	11.2295	99.22	29.28	6239	0
Ours (LoRA)	VAE.dec.	0.0768	100	33.86	1.08	0
Ours (PiSSA)	VAE.dec.	0.0768	100	31.75	0.97	0
Ours (LoHA)	VAE.dec.	1.5360	99.39	32.09	3.30	0

This is the achievement time when bit-wise accuracy reaches its highest instead of 99%, since it cannot reach to 99%.



Fig. 2. Comparison between images generated by original Stable Diffusion and watermarked Stable Diffusion: the left six images are generates without text prompts, and the right six are generated with a text prompt.

LoHA re-parameterizes θ by the Hadamard product for higher learning capacity beyond rank constraints. We train all the models with Wikiart dataset [34], with 15,000 train images and 1,000 test images. We also test the models on Pokemon dataset [35] with 1,000 images and text prompts. We set the number of the watermark bits n = 48.

The comparison results are shown in Table I. The target module represents the layers' position where the methods apply LoRA or Adapter. Table I shows that EW-LoRA outperforms other methods in Params, ACC, AT-99, and FNR@0.5%FPR, returning the fewest training parameters. It is worth noted that EW-LoRA generates higher-quality images than Stable Signature and AquaLoRA while achieving quick, accurate watermark extraction with minimal FNR. The high PSNR (also can be seen in Figure 2) confirms that watermark embedding minimally affects generative quality.

B. Dynamic Loss Weight Tuning (RQ2)

We have studied the impact of the watermark embedding on the generative task. For simplicity, in the experiments, Stable Signature [20] is used for comparison since it uses the same loss function as our method. We set the target PSNR $\rho = 30$ and ACC threshold $\mu = 95\%$. The trend of the loss weight change during training is shown in Figure 3. It can be seen that, in Stage 1, both EW-LoRA and Stable Signature decrease the weight λ_i and optimize to embed watermark. When the step comes to Stage 2, EW-LoRA can quickly reach the target image quality and watermark extraction accuracy, therefore it decreases both the weights of λ_i and λ_w . In contrast, Stable Signature keeps increasing weights to achieve the target image quality and watermark accuracy, making its convergence speed significantly slower than EW-LoRA.

TABLE II
PERFORMANCE COMPARISON OF IMAGE POST-PROCESSING ATTACKS

Method	Crop	Rotation	JPEG	Bright.	Contr.	Sharp.	Resize	Overlay.	Comb.
S.S. [20]	0.89	0.41	0.56	0.94	0.96	0.98	0.60	0.93	0.64
WMA. [23]	0.89	0.53	0.76	0.97	0.98	0.99	0.73	0.97	0.85
LaWa [24]	0.89	0.50	0.52	0.96	0.96	0.97	0.67	0.97	0.50
EW-LoRA	0.89	0.55	0.81	0.95	0.97	0.98	0.78	0.97	0.85

TABLE III PERFORMANCE COMPARISON OF DIFFERENT WATERMARK OVERWRITING ATTACKS

Attacks	S.S. [20]	WMA. [23]	LaWa [24]	EW-LoRA
StegaStamp [36]	89.21/30.91	93.96/32.92	99.98/34.33	89.09/29.54
SSL-WM. [37]	99.56/39.98	99.99/39.98	99.99/40.34	98.93/40.09

Data format: ACC/PSNR.

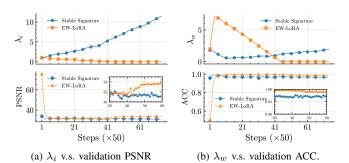


Fig. 3. Trend of loss weights, PSNR and ACC across steps for Stable Signature and EW-LoRA.

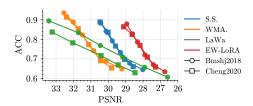


Fig. 4. Performance comparison of different watermark removal attacks.

C. Attacks

In practice, the attacker can alter the generated images to evade detection. We evaluate the robustness of the watermarking methods against various attacks, including image post-processing, watermark overwriting and removal.

Image Post-processing: In Table II, we report the results for evaluating the robustness of different methods against attacks including image center crop (aspect ratio=0.1), rotation (angle=25), JPEG compression (QF=50), brightness, contrast and sharpness enhancement (all the factors are set as 1.5), overlay text, and combination attacks (combined by JPEG (80), brightness (1.5) and center crop (0.5)). We can see that EW-LoRA outperforms other methods in most cases, showing its robustness against image post-processing attacks.

Watermark Overwriting & Removal: We simulate watermark overwriting attacks using two images watermarking methods: StegaStamp [36] and SSL-watermarking [37], and watermark

TABLE IV
ABLATION STUDY ON EMBEDDING LAYERS

n	Target module	r, α	Params/M	ACC/%	PSNR/dB	AT-99/min
48	to_q, to_k, to_v, to_out.0	4,4	0.0164	63.83	-	-
	to_q, to_k, to_v, to_out.0	40,40	0.1638	61.52	-	-
	vae.decoder.upsampler.conv	4,4	0.0051	91.04	65.48	_
	vae.decoder.upsampler.conv	40,40	0.0512	100	33.86	1.08
	vae.decoder.upsampler.conv	60,60	0.0768	100	32.45	1.67
	vae.decoder	-	49.4902	100	26.67	15.36
100	to_q, to_k, to_v, to_out.0	40,40	0.1638	61.34	-	-
	vae.decoder.upsampler.conv	40,40	0.0768	100	33.12	1.56
	vae.decoder	-	49.4902	100	26.32	16.45
150	to_q, to_k, to_v, to_out.0	40,40	0.1638	62.33	-	-
	vae.decoder.upsampler.conv	40,40	0.0768	100	32.13	2.42
	vae.decoder	-	49.4902	100	25.89	19.78

removal attacks with two autoencoders Bmshj2018 [38] and Cheng2020 [39] for image compression. Then we calculate the ACC evaluate on the attacked images and PSNR between the generated images from the watermarked model and their attacked images. From Table III we can see that, although EW-LoRA achieves a bit lower ACC and PSNR than the other methods, it is still robust against overwriting attacks. For watermark removal attacks, EW-LoRA performs better robustness than the other three, retuning a higher ACC under the same level of image compression (the same PSNR).

D. Ablation Study

We study the impact of embedding layers on watermarking performance by embedding into different LDM layers: the attention layer, the upsampler convolutional layer in the VAE decoder, and the full VAE decoder (Stable Signature). Moreover, varying LoRA settings (rank r and alpha α) are also evaluated in the experiments (See Table IV). It can be seen that the upsampler convolutional layer in VAE decoder is a better choice than attention layers for watermark embedding, with a better performance achieved when using r=40 and $\alpha=40$. While increasing the r and α will not improve the watermarking performance, and even degrade the quality of the generated images. Additionally, watermarking performance remains stable as the number of watermark bits n increases.

IV. CONCLUSION

In this letter, we propose EW-LoRA, an efficient watermarking method for LDMs that embeds the watermark into the VAE decoder using LoRA with minimal trained parameters. We introduce a dynamic loss weight tuning algorithm to balance the generative task and watermark embedding. EW-LoRA performs best when applied to layers near the output and effectively embeds watermarks using dynamic loss tuning. Future work can explore extending the method to other generative models or optimizing the embedding position.

REFERENCES

 R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun. 2022, pp. 10674–10685.

- [2] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis," in *Proceedings of International Conference on Learning Representations*, Oct. 2023.
- [3] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, D. Podell, T. Dockhorn, Z. English, and R. Rombach, "Scaling Rectified Flow Transformers for High-Resolution Image Synthesis," in *Proceedings of International Conference on Machine Learning*, Jun. 2024.
- [4] W. Peebles and S. Xie, "Scalable Diffusion Models with Transformers," in *Proceedings of IEEE/CVF International Conference on Computer Vision*, Oct. 2023, pp. 4172–4182.
- [5] K. Crowson, S. A. Baumann, A. Birch, T. M. Abraham, D. Z. Kaplan, and E. Shippole, "Scalable High-Resolution Pixel-Space Image Synthesis with Hourglass Diffusion Transformers," in *Proceedings of International Conference on Machine Learning*, Jun. 2024.
- [6] P. Fernandez, A. Level, and T. Furon, "What Lies Ahead for Generative AI Watermarking," Tech. Rep. [Online]. Available: https://blog.genlaw.org/pdfs/genlaw_icml2024/27.pdf
- [7] L. Li, W. Zhang, and M. Barni, "Universal BlackMarks: Key-Image-Free Blackbox Multi-Bit Watermarking of Deep Neural Networks," *IEEE Signal Processing Letters*, vol. 30, pp. 36–40, 2023.
- [8] P. Zhu, T. Takahashi, and H. Kataoka, "Watermark-embedded Adversarial Examples for Copyright Protection against Diffusion Models," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 24420–24430.
- [9] Y. Liu, Z. Li, M. Backes, Y. Shen, and Y. Zhang, "Watermarking Diffusion Model," arXiv:2305.12502, May 2023.
- [10] T. Šarčević, A. Karlowicz, R. Mayer, R. Baeza-Yates, and A. Rauber, "U Can't Gen This? A Survey of Intellectual Property Protection Methods for Data in Generative AI," arXiv:2406.15386, Apr. 2024.
- [11] S. Peng, Y. Chen, C. Wang, and X. Jia, "Protecting the Intellectual Property of Diffusion Models by the Watermark Diffusion Process," arXiv:2306.03436, Jun. 2023.
- [12] M. Xue, Y. Zhang, J. Wang, and W. Liu, "Intellectual Property Protection for Deep Learning Models: Taxonomy, Methods, Attacks, and Evaluations," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 908–923, 2022.
- [13] J. Zhang, D. Chen, J. Liao, W. Zhang, H. Feng, G. Hua, and N. Yu, "Deep Model Intellectual Property Protection via Deep Watermarking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4005–4020, 2022.
- [14] Y. Zhao, T. Pang, C. Du, X. Yang, N.-M. Cheung, and M. Lin, "A Recipe for Watermarking Diffusion Models," arXiv:2303.10137, Mar. 2023.
- [15] W. Feng, J. He, J. Zhang, T. Zhang, W. Zhou, W. Zhang, and N. Yu, "Catch You Everything Everywhere: Guarding Textual Inversion via Concept Watermarking," arXiv:2309.05940, Sep. 2023.
- [16] Y. Yang, Z. Liu, J. Jia, Z. Gao, Y. Li, W. Sun, X. Liu, and G. Zhai, "DiffStega: Towards Universal Training-Free Coverless Image Steganography with Diffusion Models," in *Proceedings of International Joint Conference on Artificial Intelligence*, vol. 2, Aug. 2024, pp. 1579–1587.
- [17] Z. Ma, G. Jia, B. Qi, and B. Zhou, "Safe-SD: Safe and Traceable Stable Diffusion with Text Prompt Trigger for Invisible Generative Watermarking," in *Proceedings of ACM Multimedia*, Jul. 2024.
- [18] Z. Meng, B. Peng, and J. Dong, "Latent Watermark: Inject and Detect Watermarks in Latent Diffusion Space," arXiv:2404.00230, Mar. 2024.
- [19] Z. Wang, C. Chen, L. Lyu, D. N. Metaxas, and S. Ma, "DIAGNO-SIS: Detecting Unauthorized Data Usages in Text-to-image Diffusion Models," in *Proceedings of International Conference on Learning Rep*resentations, Apr. 2024.
- [20] P. Fernandez, G. Couairon, H. Jégou, M. Douze, and T. Furon, "The Stable Signature: Rooting Watermarks in Latent Diffusion Models," in *Proceedings of IEEE/CVF International Conference on Computer Vision*, 2023, pp. 22466–22477.
- [21] V. Asnani, J. Collomosse, T. Bui, X. Liu, and S. Agarwal, "Pro-Mark: Proactive Diffusion Watermarking for Causal Attribution," in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 10802–10811.
- [22] W. Feng, W. Zhou, J. He, J. Zhang, T. Wei, G. Li, T. Zhang, W. Zhang, and N. Yu, "AquaLoRA: Toward White-box Protection for Customized Stable Diffusion Models via Watermark LoRA," in *Proceedings of International Conference on Machine Learning*, Jul. 2024, pp. 13423–13444
- [23] H. Ci, Y. Song, P. Yang, J. Xie, and M. Z. Shou, "WMAdapter: Adding WaterMark Control to Latent Diffusion Models," arXiv:2406.08337, Jun. 2024

- [24] A. Rezaei, M. Akbari, S. R. Alvar, A. Fatemi, and Y. Zhang, "LaWa: Using Latent Space for In-Generation Image Watermarking," in Proceedings of European Conference on Computer Vision, Aug. 2024.
- [25] T. Bui, S. Agarwal, N. Yu, and J. Collomosse, "RoSteALS: Robust Steganography Using Autoencoder Latent Space," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 933–942.
- [26] R. Ma, M. Guo, L. Yuming, H. Zhang, C. Ma, Y. Li, X. Xie, and S. Zhang, "PiGW: A Plug-in Generative Watermarking Framework," arXiv:2403.12053, Jan. 2024.
- [27] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," in Proceedings of International Conference on Learning Representations, Oct. 2021.
- [28] X. Zhao, K. Zhang, Z. Su, S. Vasan, I. Grishchenko, C. Kruegel, G. Vigna, Y.-X. Wang, and L. Li, "Invisible image watermarks are provably removable using generative AI," in *Advances in Neural Information Processing Systems*, 2024.
- [29] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.
- [30] S. Czolbe, O. Krause, I. Cox, and C. Igel, "A Loss Function for Generative Neural Networks Based on Watson's Perceptual Model," in Proceedings of Neural Information Processing Systems, vol. 33, 2020, pp. 2051–2061.
- [31] D. Lin, B. Tondi, B. Li, and M. Barni, "A CycleGAN Watermarking Method for Ownership Verification," *IEEE Transactions on Dependable* and Secure Computing, pp. 1–15, 2024.
- [32] F. Meng, Z. Wang, and M. Zhang, "PiSSA: Principal Singular Values and Singular Vectors Adaptation of Large Language Models," arXiv:2404.02948, May 2024.
- [33] N. Hyeon-Woo, M. Ye-Bin, and T.-H. Oh, "FedPara: Low-rank Hadamard Product for Communication-Efficient Federated Learning," in Proceedings of International Conference on Learning Representations, Oct. 2021.
- [34] "WikiArt.org Visual Art Encyclopedia," https://www.wikiart.org/.
- [35] "Jugg1024/pokemon-gpt4o-captions Datasets at Hugging Face," https://huggingface.co/datasets/jugg1024/pokemon-gpt4o-captions.
- [36] M. Tancik, B. Mildenhall, and R. Ng, "StegaStamp: Invisible hyperlinks in physical photographs," in *Proceedings of the IEEE/CVF Conference* on Computer Vision and Pattern Recognition, 2020, pp. 2117–2126.
- [37] P. Fernandez, A. Sablayrolles, T. Furon, H. Jégou, and M. Douze, "Watermarking Images in Self-Supervised Latent Spaces," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2022, pp. 3054–3058.
- [38] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," arXiv.org, Feb. 2018.
- [39] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7939–7948.