MaxMark: High-Capacity Diffusion-Native Watermarking via Robust and Invertible Latent Embedding

Supplementary Material

This appendix provides supplementary materials of MaxMark to support the main findings presented in the paper. It includes additional implementation details, application to other models, algorithm, supplementary observations and extended discussions that complement the main text. These materials aim to enhance the transparency, reproducibility, and comprehensiveness of the research.

6. Implementation Details

6.1. Attacks

We adopted a unified attack configuration for both our method and the baseline approaches to assess the robustness of our proposed method. Specifically, we employ seven types of attacks, and the impact of these attacks on the images is illustrated in the Fig.6.

(a) original (b) blur (c) brightness (d) noise

(e) contrast (f) hue (g) JPEG (h) resize

Figure 6. **The specific impact of the attacks on the images.** (a) Original image; (b) Gaussian blur (radius = 5); (c) brightness adjustment (factor = 3); (d) Gaussian noise ($\mu = 0, \sigma = 0.05$); (e) contrast adjustment (factor = 1); (f) hue adjustment (factor = 0.25); (g) JPEG compression (quality = 25); (h) resize to 25% of the original resolution.

The robustness is evaluated by computing the average bit recover accuracy under the applied attacks. Results for each attack are shown in Fig.7, demonstrating our method has more robustness across different payload sizes. This effect is achieved through the combined use of our robust watermark embedding strategy and the distribution transformation module.

6.2. Bit-Error Distribution across Latent Positions

Here we examine the relationship between the probability of bit flipping in the latent space during the forward and inverse diffusion processes and the spatial position of the bit. This forms the theoretical foundation for the automated hyperparameter optimization in our approach. Specifically, we compare the sign bit at each position in the original latent space and the inversion one. If the sign flips after recovery, the position is considered fragile. We select 100 different prompts, generating 100 images for each. For each position, we calculate the average bit error probability. The visualized results, as shown in the Fig.8, demonstrate that the bit-error probability is uniform across each position. This supports our subsequent calculation of the overall watermark error rate. Specifically, since the error probability p_{error} for individual bits can be empirically determined, we can calculate the theoretical overall error rate based on the actual bit utilization.

7. Application to Other Models and Datasets

7.1. Comparison on Other Models

Here we fistly show the generation performance of MaxMark on SD V1.4 and V2.1, as Table 11 and Table 12. It is evident that our method consistently generates high-quality images across different payload sizes.

Subsequently, we show the bit accuracy comparison with baseline methods in Table 9 and 10. Here we only compare MaxMark with state-of-the-art latent-based methods, Gaussian Shading and PRC watermark. With SD V1.4, our method outperform the state-of-the-art latent-based watermarking when capacity size larger than 1024 bits. With SD V2.1, when the capacity size is larger than 4096 bits, ours achieves highest bit accuracy. These results indicate that our proposed high-capacity watermarking method for LDMs is generalizable across popular diffusion models.

Table 9. **Bit accuracy comparison across different message capacities on SD V1.4**. PRC means PRC Watermark while G.S. represents Gaussian Shading.

Methods	Bit Accuracy(%) ↑					
	256	1024	4096	8192	12288	16384
PRC	100.0	99.4	54.1	50.3	48.6	<u>48.3</u>
G.S.	99.9	99.8	93.2	84.4	-	49.9
Ours	99.7	**99.8**	**97.6**	**96.4**	**95.4**	**95.4**

7.2. Visual Comparison on Various Datasets

In this section, we show the visual comparison of ours with other baselines on different datasets, based on SD V1.5. In

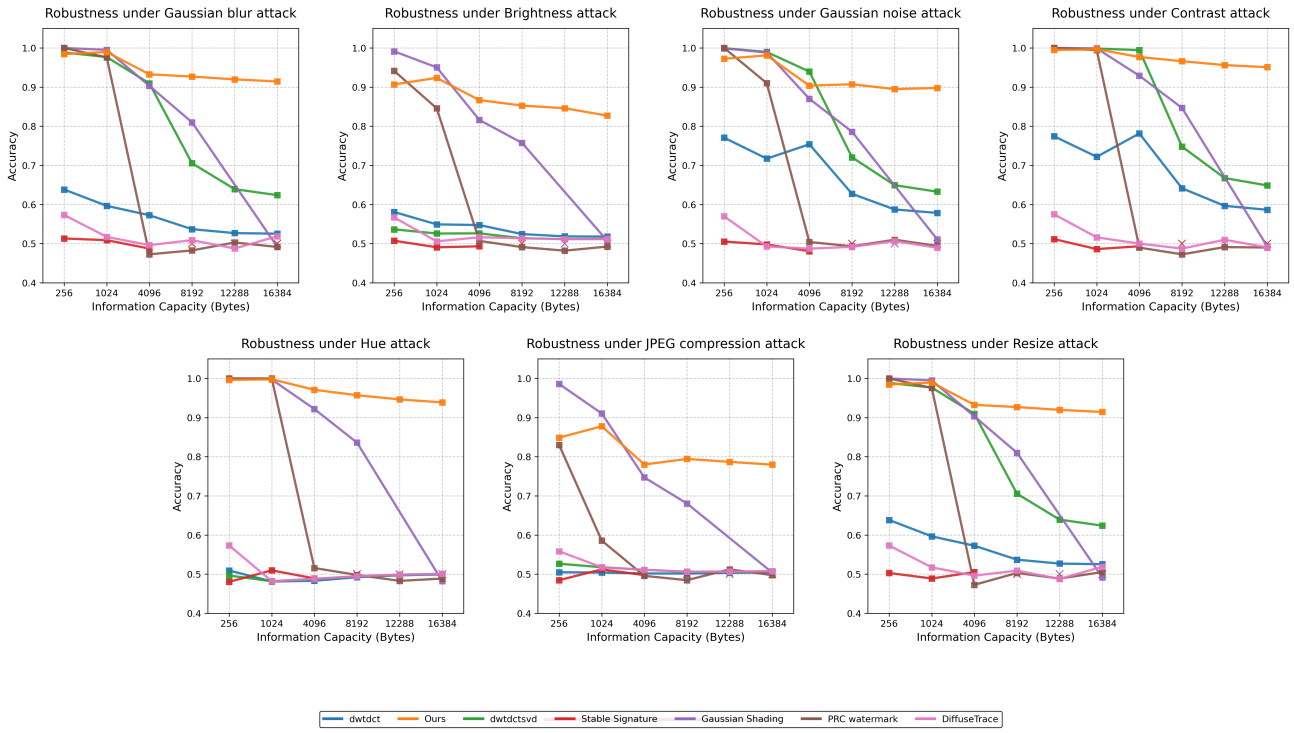


Figure 7. **Robustness comparison under seven different attacks.** The proposed method (Ours) maintains high accuracy across all attack types on high payload sizes.

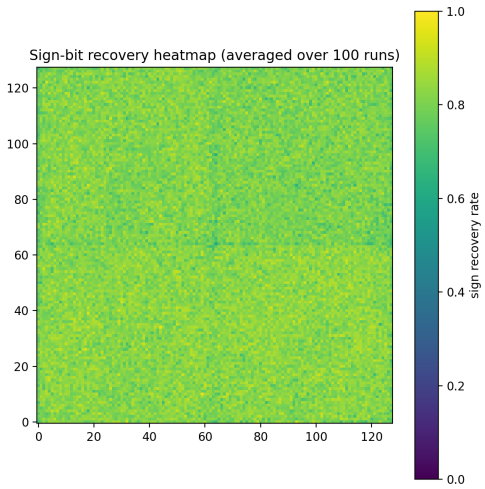


Figure 8. **The bit-error probability across entire latent.**

Table 13, we use the citations of MS-COCO-2017 training set as text prompts to generate images, MaxMark maintains a high degree of quality consistency under large-capacity settings. While the MS-COCO-2017 dataset is mainly towards realistic images, we also evaluate on the Stable-Diffusion-Prompts dataset, which features a more artistic generation style. The results are shown in Table 14. Results show that MaxMark preserves the original capabilities of the model.

Table 10. **Bit accuracy comparison across different message capacities on SD V2.1**. PRC means PRC Watermark while G.S. represents Gaussian Shading.

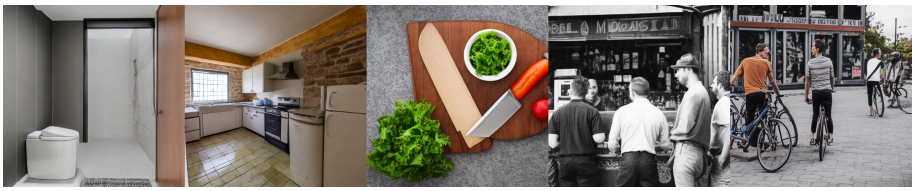
Methods	**Bit Accuracy(%) ↑**					
	256	1024	4096	8192	12288	16384
PRC	100.0	100.0	57.1	52.7	49.1	<u>49.6</u>
G.S.	99.9	96.3	81.2	72.3	-	49.9
Ours	97.1	97.8	**93.3**	**93.4**	**92.9**	**93.4**

8. ECC-Hyperparameters Search Algorithm

In this section, we introduce the details of proposed automatic ECC hyperparameters searching algorithm, aiming to maximize the payload size and extraction accuracy.

Specifically, for a user-defined binary message s of length l, and a target reliability level ε, we first estimate the bit-flip probability p_{error} based on empirical statistics ob-

Table 11. **Generated examples under different payload sizes with SD V1-4.**

Capacity	Generated Images
256	
1024	
4096	
8192	
12288	
16384	

tained from diffusion inversions. Based on this, we can estimate the symbol error rate and block-level decoding failure probability using Eq.5 and Eq.6. We then introduce a search algorithm to identify the optimal RS code configuration (n, k) that satisfies the reliability constraint ε while maximizing redundancy efficiency, detailed at Algorithm 2.

$$p_{sym} = 1 - (1 - p_{error})^m \tag{5}$$

$$P_{fail} = \sum_{i=t_{rs}+1}^{k_{rs}} \binom{k_{rs}}{i} p_{sym}^i (1 - p_{sym})^{k_{rs}-i} \tag{6}$$

The algorithm iterates over all possible block sizes B to evaluate candidate RS configurations. For each block size, it computes the total number of blocks $N_{\text{blocks}} = \lceil l/B \rceil$ and the corresponding symbols per block $k_{rs} = \lceil B/m \rceil$. Valid RS codeword lengths n_{rs} are then derived from GetRSCandidates(k_{rs}, m), ensuring compatibility with the finite field GF(2^m), detailed at Algorithm 1.

Table 12. **Generated examples under different capacity payload sizes with SD V2-1.**

Capacity(bits)	Generated Images
256	
1024	
4096	
8192	
12288	
16384	

For each candidate n_{rs}, the algorithm calculates the number of check symbols $r_{\text{sym}} = n_{rs} - k_{rs}$, the maximum correctable errors $t_{rs} = \lfloor r_{\text{sym}}/2 \rfloor$, the symbol error probability $p_{\text{sym}} = 1 - (1 - p_{\text{error}})^m$, and the block-level decoding failure rate P_{fail}. The total message failure probability $P_{\text{total}} = 1 - (1 - P_{\text{fail}})^{N_{\text{blocks}}}$ is compared against ε to filter valid configurations. Among these, the configuration with the lowest redundancy cost—defined as cost $= r_{\text{bits}}/B$, where $r_{\text{bits}} = r_{\text{sym}} \times m$.

9. Supplementary Observations and Results

9.1. Time Cost and Accuracy Analysis

To evaluate applicability of each method across capacity payload sizes, we plot bit accuracy and time cost as functions of capacity, the results are shown in Fig.9. We tested 10,000 images at different payload sizes for each method. Since training overhead must be considered in deployment scenarios, the time cost is calculated based on both training and inference expenses, and costs are averaged per image.

Algorithm 1 Generate RS Code Length Candidates

Input: k_{rs}, m
Output: set of candidate RS code lengths

$q \leftarrow 2^m$
$D \leftarrow \{d \in \mathbb{N} \mid (q-1) \bmod d = 0\}$
candidates $\leftarrow \{\, n \in D \mid n \geq k_{rs}\}$
if candidates $= \varnothing$ **then**
    return $\{k_{rs}\}$
else
    return candidates
end if

Algorithm 2 Automatic ECC Hyper-parameters Search

Input: l (secret length), p_{error} (bit error probability), m (symbol bits), ε (max allowable error)
Output: $(n_{rs}, k_{rs}, r_{sym}, t_{rs}, B)$ (optimal RS parameters and block size)

// Iterate over all possible block sizes (B = m, 2m, ..., l)
for $B = m, 2m, \ldots, l$ **do** ▷ Each block contains B bits
    $N_{\text{blocks}} \leftarrow \lceil l/B \rceil$ ▷ Total blocks
    $k_{rs} \leftarrow \lceil B/m \rceil$ ▷ Symbols per block
    // Get valid RS code length candidates
    $n_{rs} \in \text{GetRSCandidates}(k_{rs}, m)$
    for each candidate n_{rs} **do**
        $r_{sym} \leftarrow n_{rs} - k_{rs}$
        $t_{rs} \leftarrow \lfloor r_{sym}/2 \rfloor$
        $p_{\text{sym}} \leftarrow 1 - (1 - p_{\text{error}})^m$
        $P_{\text{fail}} \leftarrow \sum_{i=t_{rs}+1}^{k_{rs}} \binom{k_{rs}}{i} p_{\text{sym}}^i (1 - p_{\text{sym}})^{k_{rs}-i}$
        $P_{\text{total}} \leftarrow 1 - (1 - P_{\text{fail}})^{N_{\text{blocks}}}$
        if $P_{\text{total}} \leq \varepsilon$ **then** ▷ Check reliability constraint
            $r_{\text{bits}} \leftarrow r_{sym} \times m$
            cost $\leftarrow r_{\text{bits}}/B$
            if best $=$ None **or** cost $<$ best.cost **then** ▷ Select configuration with minimal redundancy
                best $\leftarrow (n_{rs}, k_{rs}, r_{sym}, t_{rs}, B)$
            end if
        end if
    end for
end for
return best ▷ Return optimal parameters

We begin with a 256-bit payload and record the training time as well as the inference time on 10,000 images. The cost is calculated as the average per-image value. This process is scaled from a 256-bit payload up to 16,384 bits.

Several baselines become impractical under specific capacity settings: DiffuseTrace requires retraining for each payload size, and Stable Signature additionally depends on content-specific retraining, i.e., each training can only embed a fixed watermark message, making them infeasible beyond 4096 bits due to prohibitive training costs. PRC

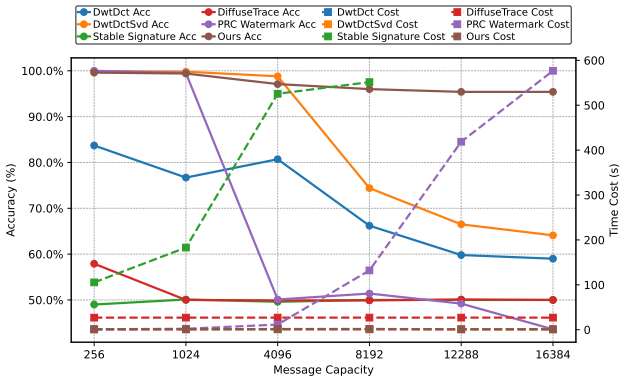


Figure 9. **Bit accuracy and time cost across various capacity sizes**. Cost indicates how many times the original model's inference cost is increased. Our method exhibits robust applicability, exhibiting low computational overhead across various payload sizes, which ensures its deployability. Additionally, it maintains a high recovery rate, thereby ensuring the reliability of the approach.

Watermark exhibits a sharp increase in decoding cost after 1024 bits. Moreover, The PRC watermarking scheme necessarily employs PRC codes to construct the embedded watermark message. The resulting PRC codeword contains not only the original message but also additional redundancy, which is essential for the encoding and decoding processes to function correctly. Gaussian Shading restricts supported message lengths to divisors of 16,384, limiting its applicability under arbitrary payload sizes. As shown in the figure, our method maintains strong performance in terms of both accuracy and time cost across various payload sizes.

9.2. Robustness of Embedding Positions

We investigate the robustness of information recovery for each bit of the fp32. When latent noise is represented using 32-bit floating-point numbers, we embed the watermark across different bits and evaluate the watermark message recovery rate. It is important to note that, different from the ablation study, the Maxmark framework is not incorporated in this analysis, i.e., we employ the original pipeline of LDMs. The results, as shown in the figure, indicate that sign and higher-order bits exhibit higher recovery rate. Embedding watermark information in these positions allows for better preservation of watermark during the forward and inverse diffusion processes. However embedding watermark into high-order bits can cause obvious perturbations to the distribution of latent, leading to degradation of image quality. This finding supports our proposed sign-based watermark embedding strategy and also motivates us to design the distribution transformation module.

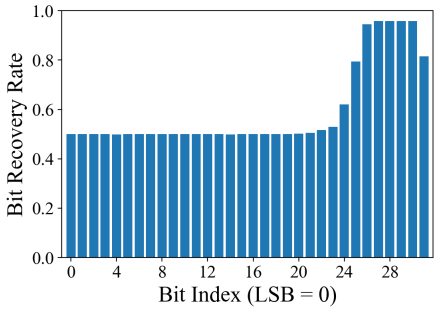


Figure 10. **Per-bit recovery accuracy of the latent noise.** Recovery rate for each of the 32 bits in the `fp32` latent noise, measured after generating images and reconstructing the latent noise via DDIM inversion.

9.3. Performance of Random ECC Settings

In the paper, we conduct ablation studies to validate the effectiveness of our proposed automatic ECC hyperparameter searching algorithm. Specifically, for each capacity setting, we randomly sample five sets of hyperparameters and report the average accuracy. The details are shown in Table 15. Here, the symbol size m is fixed as $m = 8$, n represents codeword length, k denotes message length, $r = n - k$ as the number of redundancy check symbols, t represents error correction capability, B denotes the block size. These results demonstrating that our ECC hyperparameters search method outperforms the random setting strategy, fully leverages the high-capacity character of MaxMark.

Table 13. **Visual comparison between Gaussian Shading, PRC Watermark and our method on MS-COCO-2017 training prompts**. In this experiment, we embed 16384 bits in ours, while 64 bits in Gaussian Shading and PRC Watermark, demonstrating that MaxMark maintains high visual quality of the generated images even under large capacity settings.

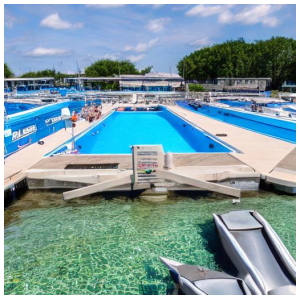
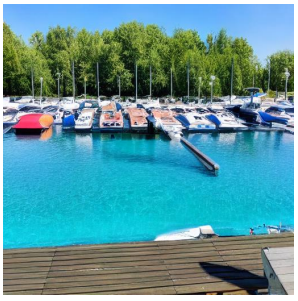
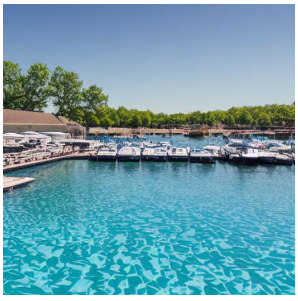
Prompt	Gaussian Shading	PRC Watermark	Ours
A white plate topped with meat, vegetables and fruit.			
There is a two level tour bus in the street.			
A wooden bench on the ground next to some trees.			
A pool next to a dock lined with lots of boats.			
a corner of a James Smith & Sons store with various umbrellas lined up in the window.			

Table 14. **Visual comparison between Gaussian Shading, PRC Watermark and our method on Stable-Diffusion-Prompts dataset**. In this experiment, we also embed 16384 bits watermark in ours, while 64 bits in Gaussian Shading and PRC Watermark, demonstrating that MaxMark maintains high visual quality under large capacity settings, while not impacting model performance.

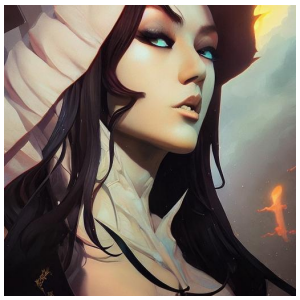
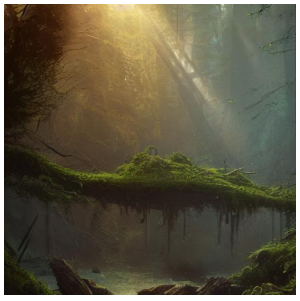
Prompt	Gaussian Shading	PRC Watermark	Ours
occult leader by artgerm, tooth wu, dan mumford, beeple, wlop, rossdraws, james jean, marc simonetti.			
portrait of a feminine boy with curly shoulder length dirty blond hair, wearing a white t shirt and black work apron.			
baroque oil painting of anime key visual environment concept art of anime rail canon artillery firing over castle walls, smoke debris.			
rusty warship dreadnought shipwreck in a lush forest, volumetric lighting, god rays, , global illumination, puddles of water, sci-fi.			
perfectly detailed goddess princess of white roses!! blessed by nature with ever - increasing physical mental perfection.			

Table 15. **Bit accuracy (%) comparison between random ECC configurations and our automatic optimal hyper-parameters searching**. Our proposed algorithm has higher accuracy for fully leverage of the high payload capacity.

Capacity	**Hyperparameters**					**Avg**	**Ours**
	n	k	r	t	B		
256	51/17/15/5/3	30/14/11/3/1	21/3/4/2/2	10/1/2/1/1	240/112/88/24/8	95.54	**99.69**
1024	255/85/85/51/17	85/76/72/28/9	170/9/13/23/8	85/4/6/11/4	680/608/576/224/72	96.28	**99.79**
4096	255/255/51/51/17	240/221/46/18/10	15/34/5/33/7	7/17/2/16/3	1920/1768/368/144/80	95.71	**97.76**
8192	255/85/51/17/17	200/62/22/16/8	55/23/29/1/9	27/11/14/0/4	1600/496/176/128/64	95.10	**96.27**