Watermarking Image Autoregressive Models

Michel Meintz [*1] **Jan Dubiński** [*2] **Franziska Boenisch** [1] **Adam Dziedzic** [1]

Abstract

Image generative models have become increasingly popular, but training them requires large datasets that are costly to collect and curate. To circumvent these costs, some parties may exploit existing models by using the generated images as training data for their own models. In general, watermarking is a valuable tool for detecting unauthorized use of generated images. However, when these images are used to train a new model, watermarking can only enable detection if the watermark persists through training and remains identifiable in the outputs of the newly trained model—a property known as radioactivity. In this work, we are the first to propose a radioactive watermarking method tailored for Image Autoregressive Models (IARs)—drawing inspiration from techniques in large language models (LLMs), which share IARs' autoregressive paradigm. Our extensive experimental evaluation highlights our method's effectiveness in preserving radioactivity within IARs, enabling robust provenance tracking, and preventing unauthorized use of their generated images.

1. Introduction

Generative models, particularly in the vision domain (Rombach et al., 2022; Ho et al., 2020; Song & Ermon, 2020; Tian et al., 2024; Yu et al., 2024), have gained significant attention for their ability to generate high-quality, realistic images. However, training these models demands large and diverse datasets, which are often costly and time-consuming to collect and curate (Schuhmann et al., 2022; Oquab et al., 2023; Andrews et al., 2023). In light of these high costs, some parties attempt to bypass the data collection process by using images generated by existing models as training data for their own new models (Shumailov et al., 2024). This practice not only undermines the original data sources but also raises critical concerns regarding unauthorized use of generated outputs (Wang et al., 2024).

To address these concerns, watermarking has emerged as a valuable tool to help model owners detect and track the misuse of their generated images (Fernandez et al., 2023; Wen et al., 2023; Gunn et al., 2024; Yang et al., 2024). By embedding distinctive markers within the generated content, watermarking allows model owners to trace the use and provenance of their generated data. However, while watermarking can effectively detect direct misuse of images, its effectiveness is limited in scenarios where the images are not directly published but instead used as training data for new models. In such cases, traditional watermarking methods can only detect unauthorized use if the watermark persists through the training process and remains detectable in the outputs of the newly trained model—a property known as *radioactivity* (Sablayrolles et al., 2020).

We turn our attention to image autoregressive models (IARs) (Tian et al., 2024; Yu et al., 2024), which have recently achieved new SOTA results in image generation quality and efficiency. To date, no watermarking schemes have been proposed for these models. Motivated by the success of watermarking for LLMs (Kirchenbauer et al., 2023), which share the autoregressive framework with IARs, we introduce **WIAR**, the first watermarking scheme designed specifically for IARs with radioactivity in mind. WIAR embeds and detects watermarks in IARs based on their token distribution. Our extensive experimental evaluation demonstrates that WIAR preserves radioactivity, facilitating robust detection of unauthorized use of generated images in IARs. In summary, we make the following contributions:

- We propose the first watermarking scheme for IARs, ensuring radioactivity by transferring to new models.
- Our WIAR method requires no additional training, embeds at inference time into pre-trained models, and preserves high image quality.
- We conduct a comprehensive empirical evaluation of WIAR, showing its radioactivity across SOTA autoregressive models.

[1]CISPA Helmholtz Center for Information Security [2]Warsaw University of Technology, NASK - National Research Institute. Correspondence to: Michel Meintz <michel.meintz@cispa.de>, Jan Dubiński <jan.dubinski.dokt@pw.edu.pl>, Franziska Boenisch <boenisch@cispa.de>, Adam Dziedzic <adam.dziedzic@cispa.de>.

Published at Data in Generative Models Workshop: The Bad, the Ugly, and the Greats (DIG-BUGS) at ICML 2025, Vancouver, Canada.

2. Background

We consider image autoregressive models, and watermarking techqniues based on the autoregressive framework. Additional related work can be found in Appendix A.

2.1. Image Autoregressive Models (IARs)

Following the standard next-token-prediction paradigm, for a given set of already predicted tokens $x_{i-1}, x_{i-2}, ..., x_2, x_1$, IARs autoregressively predict the next token x_i. The autoregressive model p_θ is trained to maximize the probability $p_\theta(x_i|x_{i-1}, x_{i-2}, ..., x_2, x_1)$. Early IARs applied row-by-row raster-scan, z-curve, or spiral orders (Chen et al., 2020; Oord et al., 2016; Yu et al., 2022; Esser et al., 2021). Recent autoregressive approaches for image generation propose improved self-supervised learning objectives. Visual autoregressive models (VARs) use the next scale (or resolution) prediction as the pretext task (Tian et al., 2024) while RARs (Yu et al., 2024) randomly permute the tokens and then perform the standard next token prediction, given the token positional information. Overall, the shift in the design paradigm of autoregressive models for images, such as VARs and RARs, allowed them to outperform DMs in the image generation quality and efficiency.

2.2. Watermarking Autoregressive Models.

Recently, the main efforts in the area of watermarks for autoregressive models were directed towards designing watermarks for LLMs (Kirchenbauer et al., 2023; Christ et al., 2024). We focus on the watermarking method proposed by Kirchenbauer et al. (2023) due to its simplicity, practicality, and performance. Their technique first randomly divides the set of all possible tokens into a $green$ list and a red list, followed by *softly* nudging the generation towards sampling the tokens from the $green$ list. The detection of the watermark does not require the access to the LLM and is based on how many times the tokens from the $green$ list instead of the red list are used in a given text. Making use of the concepts from the LLM watermarking, we propose the first watermarking scheme for IARs.

3. Watermarking Method for IARs

IARs, like LLMs, follow an autoregressive generation paradigm. This suggests that watermarking in IARs should also be based on their generated tokens. However, fundamental differences between IARs and LLMs make direct adaptations of LLM watermarking techniques infeasible. The key distinctions lie in the structure of their tokens: LLMs operate on text tokens with a *well-defined linear order*, whereas IARs generate image tokens that are spatially arranged without naturally forming a one-dimensional (1D) ordered sequence. Therefore, designing a watermarking scheme for IARs requires first defining a meaningful ordering of tokens. We do so by following the respective inherent generation properties of different IAR types.

Next, LLM tokens are *discrete* and correspond to *discrete* words or subwords, which are directly output without any modifications. The mapping between tokens and text is one-to-one. In contrast, IARs generate tokens that represent *continuous* signals, which are subsequently decoded into images. However, reversing this process by encoding the images and then tokenizing them can result in a different set of tokens as we analyze more in detail in Figure 1. This discrepancy arises from the imperfect image encoding and decoding. Consequently, this introduces a major challenge for watermarking in IARs, where the watermark detection on the level of tokens is inherently more difficult than in LLMs and requires a robust approach. To address this, we identify token sequences with the highest overlap and incorporate them into the watermark detection process. In the following sections, we detail our proposed WIAR watermarking method, including the notation, watermark encoding, and decoding.

Notation. We denote by u_i the autoregressive unit generated in each step. For example, for RAR, this would be a single token id x_j, whereas for VAR, this would be all the tokens output for a given resolution, *i.e.,* $(x_1, \ldots, x_{t_i})$, where t_i is the number of tokens for the resolution i.

3.1. Watermark Embedding

Next, we present our WIAR approach for embedding a watermark into IARs. Intuitively, WIAR relies on red and $green$ lists, following (Kirchenbauer et al., 2023) and we nudge the model to generate tokens from the $green$ list (and not from the red list) during inference. We present the details in Algorithm 1, where we color-code the lines for watermarking in blue and outline the surrounding general image generation steps in a unified manner across different types of IARs. The concrete instantiation of the generation process can be adjusted based on the concrete IAR type. For example, for VARs, the number of tokens per resolution i in lines 6, 10, and 14 varies. In contrast, for RARs, only one token is generated at each step. These differences do not affect the general watermarking procedure.

Embedding a Watermark. The watermarking starts with hashing the integer representations of the tokens u_{i-1} from the previous $(i-1)$ resolution. We use the obtained hash value to seed a PRG (Pseudo-Random Generator). Using the obtained random seed (rand), we randomly partition the vocabulary V (codebook) into a green list $green$ and a red list red. The dynamic creation of the lists based on previous token representations, rather than relying on static lists, enables higher-quality generations. The parameter $\gamma \in (0, 1)$ is the scaling factor for the size of the $green$ list. Thus, the size of the lists are $\gamma|V|$ for the $green$ and $(1-\gamma)|V|$

for the red one. The size of the $green$ list, γ, represents a trade-off between watermark strength and output quality, with smaller values of γ resulting in a stronger watermark at the expense of generation quality. Next, we generate the new tokens for the resolution i. For the individual logits $l_j^{(k)}$ (where k is index in the logit vector l_j) corresponding to tokens from the $green$ list ($k \in G$), we add the bias term δ. We do not add the bias term to the individual logits corresponding to tokens from the red list ($k \in R$). The softmax function is defined by Equation (2). Intuitively, we softly nudge the selection of each new token x_j to be from the $green$ list based on the biased probability vector p_j.

Generating the Watermarked Image. After the token selection, the image generation follows the standard IAR procedure, *i.e.,* we query the vocabulary of tokens V to obtain their representations z_i per each token index x_j, interpolate to the full resolution (h_K, w_K), and project with ϕ_i to the embedding space of image encoding. The embeddings are aggregated across all the resolutions. The final embedding is decoded to the watermarked image im and returned.

3.2. Watermark Detection

Our watermark detection then relies on taking a suspect image, *i.e.,* an image where we want to detect whether a watermark was embedded. This suspect image is then encoded into tokens, and we check whether these tokens stem (mainly) from the $green$ list (and not from the red list). Intuitively, images that consists (mainly) of tokens from the $green$ list are marked as watermarks. To perform detection, we do not require access to the IAR model.

Detecting the Watermark. Algorithm 2 presents our WIAR watermark detection algorithm where we again color-code the lines responsible for the watermark detection in blue. Algorithm 2 follows the standard encoding from IARs to obtain u_i, which is the integer representation of tokens for the current resolution i. We use a counter C, that acts as an accumulator and denotes the number of times that tokens from the $green$ list G are selected to represent the input image im. Note that the more times the selected tokens come from the $green$ list G, the higher the probability that a watermark was embedded in the image im. Next, in a similar vein to watermark embedding from Algorithm 1, we obtain a random seed from the previous autoregressive unit u_{i-1}. The intuition is that if we pick the same initial seed during encoding and decoding with the same pseudo random generator, and if the encoded previous autoregressive unit in the encoded image is the same as during the generation process, we will obtain the same seed and be able to divide the vocabulary into the same $green$ and red lists. The remaining steps follow standard IAR encoding. Eventually, to detect the watermark, we analyze whether its tokens stem from the $green$ or red lists. We note that the autoregressive model f_θ is not used for the watermark detection. Given the input image im, we only require the access to the image encoder $\mathcal{E}$, the quantizer $\mathcal{Q}$, and the pseudorandom generator PRG.

Robust Statistical Testing. A naive way to detect the watermark would rely on counting occurrences of $green$ vs. red list tokens. However, this approach might not be reliable due to the inherent problem that tokens generated by an IAR during image generation might not entirely match the tokens resulting from encoding the *exact same* generated image afterwards. This effect is caused by imperfections in the encoding and decoding from continuous image tokens to discrete token ids. We quantify this effect in Figure 1 and find that the inherent token mismatch is substantial, making watermark detection in IARs inherently more challenging than for LLMs where each token id has an exact match with a discrete textual token—making encoding the same sequence *lossless*.

However, we find that we can overcome the problem in IARs by relying on robust statistical testing, following the approach by (Kirchenbauer et al., 2023), used in LLMs to prevent paraphrasing or synonym replacement attacks. Therefore, we make our Algorithm 2 return the output from the statistical test instead. The null hypothesis is that $\mathcal{H}_0$: *The sequence of tokens is generated with no knowledge about the* red *and* $green$ *lists*. Since the red and $green$ lists are selected at random, a non-watermarked image is expected to consists of γT $green$ and $(1-\gamma)T$ red tokens, where $T = \sum_{i=1}^{K} t_i$ is the total number of tokens for the image im. The watermarked image should consists of a significantly more $green$ tokens that a non-watermarked image.

More formally, the color ($green$ or red) of the next token is a random variable X that follows the Bernoulli distribution. For the non-watermarked image, the expectation (mean) is $E[X] = \gamma$ and the Variance $Var[X] = \gamma(1-\gamma)$. The color for all the tokens of image im can be defined as another random variable Y that follows the Binomial distribution with $E[Y] = \gamma T$ and the Variance $Var[X] = \gamma(1-\gamma)T$. The probability P that a non-watermarked image would consists of only the tokens from the $green$ list is γ^T, which is extremely small. For example, for an image with the 256×256 resolution, we obtain $T = 680$ tokens and this probability P would be as low as γ^{680}. This enables us to detect the watermark by rejecting the null hypothesis $\mathcal{H}_0$. We follow (Kirchenbauer et al., 2023) and compute the z-statistics:

$$z = (|s|_G - \gamma T)/\sqrt{T\gamma(1-\gamma)}, \tag{1}$$

where $|s|_G$ is the number of the green tokens in the encoded image. We detect the watermark when the $\mathcal{H}_0$ is rejected for $z > \tau$, where τ is some pre-defined threshold.

4. Empirical Evaluation

We present our setup, main results, and ablations regarding performance of our WIAR watermark. Additional results can be found in Appendix C.

Experimental Setup. For IARs, we consider VAR and RAR as two representative architectures, using the code and models provided in their respective code repositories. We run the IAR experiments using ImageNet, with $\delta = 2, \gamma = 0.25$ for RAR and $\delta = 6, \gamma = 0.25$ for VAR. Additional information is given in Appendix B.

Metrics. We follow Kirchenbauer et al. (2023) and report **TPR@FPR=1%** for WIAR, which measures the true positive rate (TPR) at a 1% false positive rate (FPR) when detecting watermarked images. In this case, TPR = 1% corresponds to random guessing, whereas a significantly higher TPR ($>> 1\%$) indicates a robust watermark.

In the last step of the WIAR method, we run the statistical z-test (see Equation (1)) to detect the presence of the watermark when $z > \tau$. To ensure low probability (3×10^{-5}) of false-positives, *i.e.,* rejecting the null hypothesis $\mathcal{H}_0$ when the image is not watermarked, we set a high threshold $\tau = 4$.

Our Watermark for IARs is Radioactive. We observe that our new watermarks for IARs successfully transfer from M_1 to the subsequently trained M_2 IARs. In the extreme scenario, see Table 2, where we fine-tune M_2 on a single image, the watermark exhibits even *perfect* radioactivity (100% transfer). In the practical scenario, when fine-tuning IARs on 40k images, in Table 1, we observethat the TPR at 1% FPR is around 30-40%, *i.e.,* still significantly higher than random guessing. Thereby, our WIAR watermark remains the only watermark that consistently transfers between models of high-generation quality.

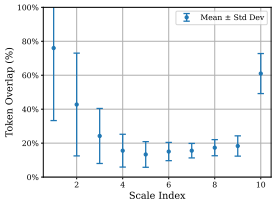
Table 1: **WIAR is radioactive.** We report TPR@FPR=1% (*i.e.,* 1% corresponds to random guessing).

Type of M_1	Type of M_2	Output of M_1	Output of M_2
VAR-*d*16	VAR-*d*16	94.0	36.1
VAR-*d*20	VAR-*d*20	94.7	36.5
VAR-*d*24	VAR-*d*24	97.1	38.1
VAR-*d*30	VAR-*d*30	95.5	37.2
RAR-B	RAR-B	92	33
RAR-L	RAR-L	90	32
RAR-XL	RAR-XL	94	36
RAR-XXL	RAR-XXL	90	29

Table 2: **Extensive fine-tuning on a single watermarked sample.** We use the metrics from Table 1.

Type of M_1	Type of M_2	Output of M_1	Output of M_2
VAR-*d*16	VAR-*d*16	100	100
VAR-*d*20	VAR-*d*20	100	100
VAR-*d*24	VAR-*d*24	100	100
VAR-*d*30	VAR-*d*30	100	100
RAR-B	RAR-B	100	100
RAR-L	RAR-L	100	100
RAR-XL	RAR-XL	100	100
RAR-XXL	RAR-XXL	100	100

Watermarking vs. Generation Quality. We analyze the impact of our WIAR on the quality of generated images



(a) VAR.

Type of RAR	Avg. Token Overlap
RAR-B	$65.45\% \pm 9.04\%$
RAR-L	$67.62\% \pm 7.21\%$
RAR-XL	$65.75\% \pm 6.38\%$
RAR-XXL	$65.68\% \pm 8.02\%$

(b) RAR.

Figure 1: **Token overlap between generated vs encoded images for VAR and RAR.**

(measured in FID score (Heusel et al., 2017)) in Table 3. The results show that our WIAR only marginally affects image quality, highlighting that our method is able to provide robust provenance tracking without sacrificing performance.

Table 3: **Impact of watermarking on image generation quality.**

Model	# of Parameters	Clean FID↓	Watermark FID↓
VAR-*d*16	310M	6.50	7.41
VAR-*d*20	600M	5.77	6.63
VAR-*d*24	1.0B	5.25	6.21
VAR-*d*30	2.0B	4.96	5.92
RAR-B	216M	5.02	6.16
RAR-L	461M	4.73	5.78
RAR-XL	955M	4.66	5.45
RAR-XXL	1499M	4.50	5.21

Token Mismatch in Decoded (Generated) vs. Encoded Images. We identify an inherent challenge in IAR watermarking arising from imperfect image encoding and decoding. Specifically, a model generates tokens T, which are then decoded into a generated image. When *the exact same image* is subsequently encoded and tokenized, it produces a different set of tokens, T', which do not fully match the original tokens T, *i.e.,* $T' \neq T$. This differs from LLMs, where the tokens decoded to the generated *text* match perfectly the tokens obtained from the tokenized exact same *text*. The discrepancy decreases the watermark detection rate of WIAR, as it relies on the correct tokens to determine whether they belong to the $green$ or red list. We quantify the token overlap in Figure 1a for VAR and in Figure 1b for RAR, respectively. For VAR, the initial and final scales (resolutions) exhibit the highest token overlap. This is likely because the initial resolutions capture broad structural features, while at the final resolution, the image is already refined, resulting in fewer token changes. For RAR, which only generates one token at a time, we observe that the average token overlap is similar among all model sizes, namely around 65%.

5. Conclusion

We introduced WIAR, the first watermarking method for IARs, designed to ensure radioactivity and enable provenance tracking even when generated images are used for training new IAR models. Through extensive experiments, we demonstrated the effectiveness of our approach in preserving watermark radioactivity, providing a robust solution for detecting unauthorized use of generated images.

References

Andrews, J., Zhao, D., Thong, W., Modas, A., Papakyriakopoulos, O., and Xiang, A. Ethical considerations for responsible data curation. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=Qf8uzIT1OK.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., and Sutskever, I. Generative pretraining from pixels. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1691–1703. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20s.html.

Christ, M. and Gunn, S. Pseudorandom error-correcting codes. *arXiv preprint arXiv:2402.09370*, 2024. URL https://arxiv.org/abs/2402.09370.

Christ, M., Gunn, S., and Zamir, O. Undetectable watermarks for language models. In Agrawal, S. and Roth, A. (eds.), *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pp. 1125–1139. PMLR, 30 Jun–03 Jul 2024. URL https://proceedings.mlr.press/v247/christ24a.html.

Esser, P., Rombach, R., and Ommer, B. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Fernandez, P., Couairon, G., Jégou, H., Douze, M., and Furon, T. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22466–22477, October 2023.

Gunn, S., Zhao, X., and Song, D. An undetectable watermark for generative image models. *arXiv preprint arXiv:2410.07369*, 2024.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., and Goldstein, T. A watermark for large language models. In *International Conference on Machine Learning*, pp. 17061–17084. PMLR, 2023.

Liu, Y., Song, Y., Ci, H., Zhang, Y., Wang, H., Shou, M. Z., and Bu, Y. Image watermarks are removable using controllable regeneration from clean noise. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mDKxlfraAn.

Oord, A. v. d., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. Conditional image generation with pixelcnn decoders. NIPS'16, pp. 4797–4805, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Sablayrolles, A., Douze, M., Schmid, C., and Jégou, H. Radioactive data: tracing through training. In *International Conference on Machine Learning*, pp. 8326–8335. PMLR, 2020.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35: 25278–25294, 2022.

Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., and Gal, Y. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, 2024.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems*,

2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92c3b916311a5517d9290576e3ea37ad-Paper.pdf.

Tian, K., Jiang, Y., Yuan, Z., PENG, B., and Wang, L. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=gojL67CfS8.

Wang, Z., Chen, C., Lyu, L., Metaxas, D. N., and Ma, S. DIAGNOSIS: Detecting unauthorized data usages in text-to-image diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=f8S3aLm0Vp.

Wen, Y., Kirchenbauer, J., Geiping, J., and Goldstein, T. Tree-rings watermarks: Invisible fingerprints for diffusion images. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 58047–58063. Curran Associates, Inc., 2023.

Yang, Z., Du, M., Jin, Z., and Hu, X. Gaussian shading: A new perspective for watermarking diffusion models. *arXiv preprint arXiv:2401.05678*, 2024.

Yu, J., Li, X., Koh, J. Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., and Wu, Y. Vector-quantized image modeling with improved VQGAN. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=pfNyExj7z2.

Yu, Q., He, J., Deng, X., Shen, X., and Chen, L.-C. Randomized autoregressive visual generation. 2024. URL https://arxiv.org/abs/2411.00776.

Zhao, Y., Pang, T., Du, C., Yang, X., Cheung, N.-M., and Lin, M. A recipe for watermarking diffusion models, 2023.

A. Additional Related Work

A.1. Image Generative Models

Diffusion Models (DMs) (Ho et al., 2020; Song & Ermon, 2020) are trained by progressively adding noise to the data and then learning to reverse this process. The forward diffusion process adds Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ to a clean image x, yielding a noisy sample $x_t \leftarrow \sqrt{\alpha_t}x + \sqrt{1-\alpha_t}\epsilon$, where $t \in [0, T]$ is the diffusion timestep, and $\alpha_t \in [0, 1]$ is a decaying parameter with $\alpha_0 = 1$ and $\alpha_T = 0$. The model f_θ is trained to predict the noise ϵ by minimizing: $\mathcal{L}(x, t, \epsilon; f_\theta) = \|\epsilon - f_\theta(x_t, t)\|_2^2$. In conditional settings, f_θ is guided by an additional input y, such as a class label (Ho et al., 2020) or a text embedding from a pretrained encoder like CLIP (Radford et al., 2021).

Latent diffusion models (LDMs) (Rombach et al., 2022) improve DMs by conducting the diffusion process in the latent space instead of in the pixel space, which significantly reduces computational complexity, making training scalable and inference more efficient. For the LDMs, the encoder $\mathcal{E}$ transforms the input x to the latent representation $z = \mathcal{E}(x)$ and the diffusion loss is formulated as $\mathcal{L}(z, t, \epsilon; f_\theta) = \|\epsilon - f_\theta(z_t, t)\|_2^2$.

Elucidated diffusion models (EDMs) (Karras et al., 2022), in contrast, operate in the pixel space (directly on x instead of on its latent representation z).

A.2. Image Autoregressive Models (IARs)

RARs (Randomized Autoregressive Models) (Yu et al., 2024) randomly order the tokens and then train the model to correctly predict the next token in the sequence. Each token is assigned position in the sequence, so the pretext task in RARs facilitates the multidirectional representation since each token can be predicted from any set of previously given other tokens. This reflects the nature of images where a given part of the image might be influenced by any not necessarily neighboring parts. The randomization of token order in RARs is slowly reduced (annealed) from fully random order to perfect raster-scan order.

A.3. Watermarking Diffusion Models.

SOTA watermarking techniques for DMs are learning-based (Zhao et al., 2023; Fernandez et al., 2023; Wen et al., 2023; Gunn et al., 2024). These watermarking methods have three key components: a watermark (w), an encoder (E) and a decoder (D). The encoder receives an input image X and the watermark w as input and is trained to embed the watermark into the image, while not reducing the image quality. Simultaneously, the decoder is trained to reconstruct the watermark given the watermarked image. This can be formulated as: $w = D(E(X, w))$ and $X \simeq X_w = E(X, w)$. Watermarking methods for LDMs, can target different points of the diffusion process, such as the initial noise prediction (Wen et al., 2023; Gunn et al., 2024) or the decoding process (Fernandez et al., 2023).

Recipe Method. Zhao et al. (2023) proposed a method that utilizes a pretrained encoder-decoder structure to embed a predefined binary signature into the training data of an EDM. The EDM is then trained on this watermarked dataset, causing it to reproduce the watermark in its generated images. A watermark decoder is subsequently used to extract and verify the binary signature from these outputs. We refer to this approach to as the *Recipe* method.

Stable Signature. Fernandez et al. (2023) introduced a watermarking technique for LDMs (Rombach et al., 2022), leveraging the model's structure. The method fine-tunes the latent decoder to embed an invisible binary signature in every generated image. This signature can later be recovered using a pretrained watermark extractor.

*Tree-Ring.*Wen et al. (2023) proposed embedding a structured pattern into the *initial noise vector* used in DM sampling. By structuring these patterns in the Fourier space, the method achieves a high level of robustness against common image transformations, such as cropping, dilation, and rotation. Unlike the *Stable Signature* approach, which modifies the decoder, *Tree-Ring* influences the sampling process itself, allowing for watermark detection by inverting the diffusion process and analyzing the retrieved noise vector.

PRC Watermarking. Gunn et al. (2024) introduced *P*seudo *R*andom *C*ode (PRC) watermarking, which utilizes the random generation of an *initial noise vector*. This noise vector is sampled based on a pseudorandom error-correcting code (Christ & Gunn, 2024), enabling watermark detection. Given an image, the original noise vector can be reconstructed, allowing the identification of the specific random code used to generate the image.

A.4. Watermarking Properties

In text-to-image models, watermarks are designed to be imperceptible to the human eye yet detectable by a specialized detection algorithms. Image watermarks must possess the following essential properties: (1) *Undetectability* ensures that the watermark remains imperceptible to unauthorized parties, (2) *Unforgeability* guarantees that an adversary cannot reproduce the same watermark, (3) *Tamper-evidence* allows for the detection of any modifications to the watermark, and (4) *Robustness* ensures that the watermark remains detectable even under adversarial attacks.

B. Experimental Details

The RAR and VAR M_2 models are trained on 40000 generated watermarked ImageNet images for 10 epochs, with batch size = 4, learning rate = $1e-4$ and a 256×256 resolution. The other model specific hyperparameters are taken from the original training setup as specified in (Yu et al., 2024) and (Tian et al., 2024). For a more consistent training we utilize only major-row scan in RAR for the whole fine-tuning process. For RAR we generate images with a δ of 2.0 and for VAR with a δ of 6.0. For both architectures we utilize a γ of 0.25.

C. Additional Experiments

Table 4: **Extensive fine-tuning on a single watermarked sample.** We extend our analysis from Table 2. We use the metrics from Table 1.

Type of M_1	Type of M_2	Output of M_1	Output of M_2
VAR-*d*16	VAR-*d*30	100	100
VAR-*d*20	VAR-*d*16	100	100
VAR-*d*24	VAR-*d*16	100	100
VAR-*d*30	VAR-*d*16	100	95.2
RAR-B	RAR-XXL	100	100
RAR-L	RAR-B	100	100
RAR-XL	RAR-B	100	100
RAR-XXL	RAR-B	100	93.5

Attacks. We perform a wide range of attacks, namely: 1) **Noise** : Adds Gaussian noise with a std of 0.1 to the image, 2) **Kernel** : Noise attack + application of a Gaussian blur with a kernel size of 7, 3) **Color**: Color jitter with a random hue of 0.3, saturation scaling of 3.0 and contrast of 3.0, 4) **Grey** : Transforming the image to greyscale, 5) **JPEG** : 25% JPEG compression, 6) **SD-VAE** : Reconstruction attack based on encoding-decoding with the Stable Diffusion 2.1 VAE, 7) **CtrlRegen+** : Controlled regeneration from noise in the latent space following (Liu et al., 2025).

The CtrlRegen removal attack removes SOTA watermarks such as Tree-Ring (Wen et al., 2023) and Stable Signature (Fernandez et al., 2023), but WIAR is resilient to it and many other attacks (mean of 5 runs,1k images) with TPR>>1%@FPR=1%.

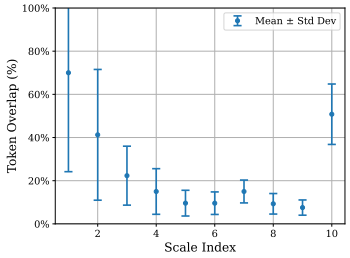
Table 5: **We analyze the robustness of WIAR under different attacks.**

Model↓ / Attacks→	None	Noise	Kernel	Color	Grey	JPEG	SD-VAE	CtrlRegen+
RAR-B	96.50	16.58	14.04	12.73	38.20	81.57	85.33	16.20
RAR-L	95.70	15.33	12.59	12.28	32.00	80.32	83.74	10.00
RAR-XL	96.60	18.95	16.67	13.95	36.70	85.28	87.58	17.80
RAR-XXL	93.80	14.44	12.86	11.52	25.60	78.05	79.81	4.40
VAR 16	99.30	45.32	42.80	56.20	88.10	88.20	54.00	4.70
VAR 20	98.30	50.20	47.00	59.80	91.00	90.50	62.40	8.50
VAR 24	99.30	51.10	49.10	60.30	90.50	90.40	64.40	8.40
VAR 30	99.00	49.20	45.80	57.10	87.60	87.80	60.20	6.90

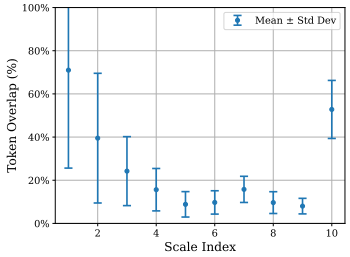
Watermarking vs. Generation Speed. We report the wall-clock time (in seconds) for an image generation on NVIDIA A40 GPU without and with WIAR, which adds a relatively small overhead, similarly to LLM watermarking.

Table 6: **Impact of watermarking on image generation speed.**

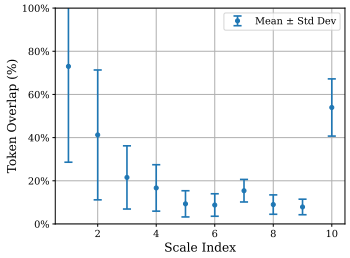
Setup	RAR-B	RAR-L	RAR-XL	RAR-XXL	VAR 16	VAR 20	VAR 24	VAR 30
original	4.043	4.129	5.430	6.623	0.399	0.428	0.462	0.531
WIAR	6.781	6.889	8.201	9.411	0.566	0.587	0.605	0.641



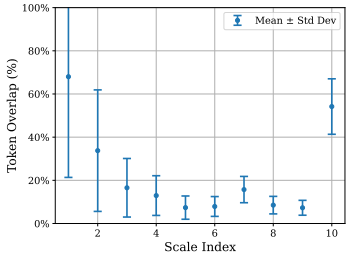
(a) Token overlap for the VAR model with depth 16.



(b) Token overlap for the VAR model with depth 20.



(c) Token overlap for the VAR model with depth 24.



(d) Token overlap for the VAR model with depth 30.

Figure 2: **Comparison of the token overlap at the different scales (resolutions) and for different depths (16, 20, 24, and 30) of the VAR model.**

D. Algorithms

The notation of Algorithm 1 and Algorithm 2 follows Tian et al. (2024) with the following operations:

- Interpolate: Up- or down- scaling to the respective resolution.
- Partition: Split the vocabulary into a green and red list given a random number and size of the green list.
- Bias: Bias all logits that are part of the green list with the delta value, *i.e.,* adding it to the logits of all green list tokens.
- Lookup: Lookup the tokens in the codebook V and return the quantized values.

The softmax in line 12 is defined as:

$$p_j = \begin{cases} \frac{\exp(l_j^{(k)}+\delta)}{\sum_{i\in R}\exp(l_j^{(i)})+\sum_{i\in G}\exp(l_j^{(i)}+\delta)}, & k \in G \\ \frac{\exp(l_j^{(k)})}{\sum_{i\in R}\exp(l_j^{(i)})+\sum_{i\in G}\exp(l_j^{(i)}+\delta)}, & k \in R. \end{cases} \tag{2}$$

Algorithm 1 Image Generation for IARs with WIAR: Watermark Embedding

Inputs: autoregressive model f_θ, vocabulary V, decoder $\mathcal{D}$
Hyperparameters: steps K, resolutions $(h_i, w_i)_{i=1}^K$, $t_i = h_i \cdot w_i$, scaling ratio γ, bias constant δ
$e \leftarrow$ init()
$u_0 \leftarrow$ {initial seed}
for $i = 1$ to K **do**
  $(l_1, \ldots, l_{t_i}) \leftarrow f_\theta(\text{interpolate}(e, h_i, w_i))$
  seed $\leftarrow$ hash(u_{i-1})
  rand $\leftarrow PRG$(seed)
  Green, Red $\leftarrow$ Partition(V, rand, γ)
  for $j = 1$ to t_i **do**
    $l_j \leftarrow$ Bias(Green, l_j, δ)
    $p_j \leftarrow$ Softmax(l_j)
    $x_j \leftarrow$ Sample(p_j)
  end for
  $u_i \leftarrow (x_1, \ldots, x_{t_i})$
  $z_i \leftarrow$ lookup(V, u_i)
  $z_i \leftarrow$ interpolate(z_i, h_K, w_K)
  $e \leftarrow e + \phi_i(z_i)$
end for
$im \leftarrow \mathcal{D}(e)$
Return: watermarked image im

Algorithm 2 Image Encoding for IARs with WIAR Watermark Detection

Inputs: image im, encoder $\mathcal{E}$, quantizer $\mathcal{Q}$ to vocab V
Hyperparameters: steps K, resolutions $(h_i, w_i)_{i=1}^K$, ratio γ
$e \leftarrow \mathcal{E}(im)$
$u_0 \leftarrow$ {initial seed}
$C \leftarrow 0$
for $i = 1$ to K **do**
  $u_i \leftarrow \mathcal{Q}(\text{interpolate}(e, h_i, w_i))$
  seed $\leftarrow$ hash(u_{i-1})
  rand $\leftarrow PRG$(seed)
  Green, Red $\leftarrow$ Partition(V, rand, γ)
  $C \leftarrow$ Count(u_i, Green)
  $z_i \leftarrow$ lookup(V, u_i)
  $z_i \leftarrow$ interpolate(z_i, h_K, w_K)
  $e \leftarrow e - \phi_k(z_k)$
end for
Return: StatisticalTest$(\mathcal{H}_0(C))$