

PlugMark: A Plug-in Zero-Watermarking Framework for Diffusion Models

Pengzhen Chen, Yanwei Liu, Xiaoyan Gu, et al.

Institute of Information Engineering, Chinese Academy of Sciences
 University of Chinese Academy of Sciences
 Columbia University
 Tsinghua University
 University of Science and Technology of China

ICCV 2025

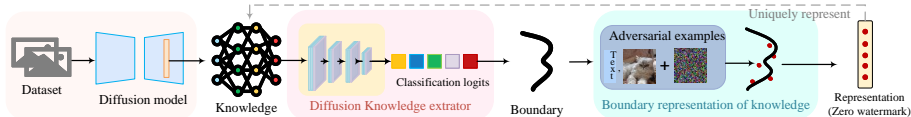
- 1 Introduction & Motivation
- 2 PlugMark: The Proposed Framework
- 3 Methodology in Detail
- 4 Experiments & Results
- 5 Conclusion

The Rise of Diffusion Models The Need for IP Protection

- Diffusion models (like Stable Diffusion, DALL·E2) have revolutionized image synthesis.
- They require massive resources to train, making them valuable intellectual property (IP).
- Unauthorized use, plagiarism, and model leakage are serious concerns.

Problem with Existing Methods

- Most methods embed watermarks by altering the model.
- This **compromises image quality** (fidelity).
- They are **vulnerable to attacks** like fine-tuning, especially in open-source models.



Our Goal: A robust IP protection method that does not alter the original model (**zero-watermarking**) and works in a 'white-box' scenario.

The Core Idea Behind PlugMark

Our approach is based on two key observations:

Observation 1: Classifiers and Decision Boundaries

A deep learning classifier can be uniquely characterized by its decision boundaries.

Observation 2: Knowledge in Diffusion Models

The unique knowledge of a diffusion model is encapsulated within its UNet, which contains rich semantic information.

The Core Idea Behind PlugMark

Our approach is based on two key observations:

Observation 1: Classifiers and Decision Boundaries

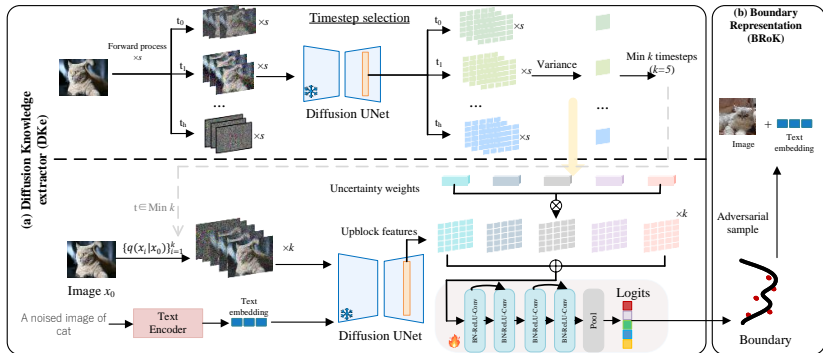
A deep learning classifier can be uniquely characterized by its decision boundaries.

Observation 2: Knowledge in Diffusion Models

The unique knowledge of a diffusion model is encapsulated within its UNet, which contains rich semantic information.

Main Idea:

Convert the unique knowledge of a diffusion model into a unique set of decision boundaries, and use those boundaries as a fingerprint, or **zero-watermark**.



(a) Diffusion Knowledge Extractor (DKe):

- A plug-in module that "reads" the knowledge from the UNet.
- Outputs a classification result without changing the diffusion model.

(b) Boundary Representation of Knowledge (BRoK):

- Generates sample pairs (Image, Text) that lie on the decision boundary of the DKe.
- These pairs form the unique, zero-distortion watermark.

Goal: Extract stable knowledge from the UNet's multi-timestep process.

Timestep Selection Mechanism

Simply using one timestep is insufficient; using all is too costly. We need to find the most *stable* timesteps.

- 1 Uniformly sample h timesteps from the diffusion process.
- 2 For each timestep t , get features $f_{t,i}$ over s forward passes.
- 3 Calculate the variance of the features: $\sigma_t^2 = \frac{1}{s} \sum_{i=1}^s (f_{t,i} - \mu_t)^2$.
- 4 **Select the Min- k timesteps** with the lowest variance (i.e., highest certainty/stability).

Goal: Extract stable knowledge from the UNet's multi-timestep process.

Timestep Selection Mechanism

Simply using one timestep is insufficient; using all is too costly. We need to find the most *stable* timesteps.

- 1 Uniformly sample h timesteps from the diffusion process.
- 2 For each timestep t , get features $f_{t,i}$ over s forward passes.
- 3 Calculate the variance of the features: $\sigma_t^2 = \frac{1}{s} \sum_{i=1}^s (f_{t,i} - \mu_t)^2$.
- 4 **Select the Min- k timesteps** with the lowest variance (i.e., highest certainty/stability).

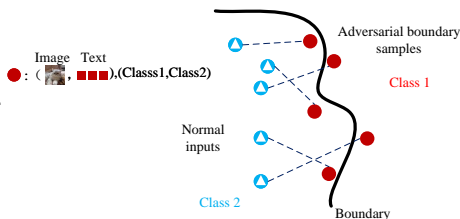
Feature Fusion & Classification:

- The features from these Min- k timesteps are combined using a weighted average (weights are inverse variance).
- This final feature vector is fed into a ResNet-like network to produce a classification result.
- **Crucially, the original diffusion model remains frozen**

Part 2: Boundary Representation of Knowledge (BRoK)

Goal: Generate the zero-watermark by finding points on the DKe's decision boundaries.

- The watermark is a set of sample pairs:
 $\{(\text{Image}_i, \text{Text}_i), (\text{Class}_1, \text{Class}_2)\}$.
- We use an optimization process to generate these pairs.



- A point is on the boundary between class i and j if its classification logits l_i and l_j are nearly equal and are the highest among all classes.

Optimization Loss Function: $L = \lambda_s L_{sim} + \lambda_m L_{mar}$

- **Similarity Loss (L_{sim}):** Pushes logits to be close.
 $\max(0, (l_i - l_j)^2 - T_s)$
- **Max Margin Loss (L_{mar}):** Ensures logits for classes i and j are greater than all other classes by a margin m .
 $\text{ReLU}(l_{\text{other}} - l_i + m) + \text{ReLU}(l_{\text{other}} - l_j + m)$

- 1 The model owner has the pre-trained **DKe** and the generated **BRoKs** (the watermark).

- 1 The model owner has the pre-trained **DKe** and the generated **BROKs** (the watermark).
- 2 To check a suspect model, the owner plugs it into their pre-trained DKe.

- 1 The model owner has the pre-trained **DKe** and the generated **BROKs** (the watermark).
- 2 To check a suspect model, the owner plugs it into their pre-trained DKe.
- 3 The owner inputs the (Image, Text) pairs from the BROKs into the combined system.

- 1 The model owner has the pre-trained **DKe** and the generated **BRoKs** (the watermark).
- 2 To check a suspect model, the owner plugs it into their pre-trained DKe.
- 3 The owner inputs the (Image, Text) pairs from the BRoKs into the combined system.
- 4 They check if the Top-2 classification outputs from the DKe match the (Class1, Class2) pair stored in the BRoK.

- 1 The model owner has the pre-trained **DKe** and the generated **BRoKs** (the watermark).
- 2 To check a suspect model, the owner plugs it into their pre-trained DKe.
- 3 The owner inputs the (Image, Text) pairs from the BRoKs into the combined system.
- 4 They check if the Top-2 classification outputs from the DKe match the (Class1, Class2) pair stored in the BRoK.
- 5 The matching accuracy is calculated:

$$r = \frac{1}{\text{num}} \sum_{i=1}^{\text{num}} \frac{\text{number of matched pairs in BRoK}_i}{\text{total number of pairs in BRoK}_i}$$

- **Base Model:** Stable-Diffusion-v2
- **Datasets for DKe training:** CIFAR-10, CIFAR-100, MNIST
- **Attack Scenarios (Post-processing):**
 - ▶ **Fine-tuning:** Using DreamBooth for 100, 1k, 20k, and 200k steps.
 - ▶ **Weight Pruning:** Removing 10%, 20%, and 30% of the model parameters.
- **Negative Models:** Re-initialized or unrelated models to test for false positives.
- **Metrics:**
 - ▶ **Fidelity:** Fréchet Inception Distance (FID) ↓
 - ▶ **Robustness:** Matching Accuracy (%) ↑ and True Positive Rate (TPR) ↑ at very low False Positive Rate (FPR).

Results: Robustness Against Attacks

Table: Matching accuracy (%) on post-processed models (DKe trained on CIFAR-10).

Model	Fine-tuning (steps)				Pruning (ρ)			Unrelated
	100	1k	20k	200k	0.1	0.2	0.3	Negative
Accuracy	99.18	98.83	72.55	69.82	98.00	86.67	81.55	10.00

Key Findings

- PlugMark maintains **very high accuracy** even after extensive fine-tuning and significant pruning.
- The accuracy on unrelated (negative) models is extremely low, demonstrating **high uniqueness** and a low false positive rate.

Table: Comparison of PlugMark with other watermarking methods.

Method	Zero-WM?	FID ↓	Origin Acc. ↑	Finetune (1k) Acc. ↑	Prune (0.1) Acc. ↑
None (No WM)	-	24.25	-	-	-
Stable Signature	✗	24.77	98.30	51.02	48.33
AquaLoRA	✗	24.88	95.79	50.13	50.00
WatermarkDM	✗	25.92	High	Low	Very Low
PlugMark (Ours)	✓	24.25	100.00	98.83	98.00

- **Fidelity:** PlugMark causes **zero image quality degradation** (same FID as the original model) because it doesn't modify the model. Other methods increase FID.

Table: Comparison of PlugMark with other watermarking methods.

Method	Zero-WM?	FID ↓	Origin Acc. ↑	Finetune (1k) Acc. ↑	Prune (0.1) Acc. ↑
None (No WM)	-	24.25	-	-	-
Stable Signature	✗	24.77	98.30	51.02	48.33
AquaLoRA	✗	24.88	95.79	50.13	50.00
WatermarkDM	✗	25.92	High	Low	Very Low
PlugMark (Ours)	✓	24.25	100.00	98.83	98.00

- **Fidelity:** PlugMark causes **zero image quality degradation** (same FID as the original model) because it doesn't modify the model. Other methods increase FID.
- **Robustness:** Baseline methods fail catastrophically after fine-tuning or pruning. PlugMark remains **extremely robust**.

We tested the importance of each component.

Table: Matching accuracy (%) with different components removed.

Components	Prune (0.1)	Finetune (1k)
Full Model (image+text+time+uncertainty)	98.00	98.83
- uncertainty selection	94.90	94.82
- multi-timestep	90.00	85.18
- text in BRoK	61.90	66.09
- text in DKe	42.75	40.67

Conclusions:

- The multimodal (**image+text**) nature is critical.
- Using **multiple, stable timesteps** is essential for robustness.
- The uncertainty-based **timestep selection mechanism** provides a significant boost.

The Problem

Existing IP protection for diffusion models degrades image quality and is not robust to common attacks like fine-tuning.

Our Solution: PlugMark

- A novel **plug-in, zero-watermarking** framework.
- Achieves IP protection **without compromising model fidelity**.
- Introduces the idea of using a **boundary representation of knowledge (BRoK)** as a unique and robust watermark.

Key Takeaway

Extensive experiments show PlugMark is highly effective, adaptive, and robust against fine-tuning and pruning, offering a superior solution for protecting valuable diffusion models.