Defense against Model Extraction Attack by Bayesian Active Watermarking

Zhenyi Wang [1] **Yihan Wu** [1] **Heng Huang** [1]

Abstract

Model extraction is to obtain a cloned model that replicates the functionality of a black-box victim model solely through query-based access. Present defense strategies exhibit shortcomings, manifesting as: (1) computational or memory inefficiencies during deployment; or (2) dependence on expensive defensive training methods that mandate the re-training of the victim model; or (3) watermarking-based methods only *passively* detect model theft without actively preventing model extraction. To address these limitations, we introduce an innovative Bayesian *active* watermarking technique to fine-tune the victim model and learn the watermark posterior distribution conditioned on input data. The fine-tuning process aims to maximize the log-likelihood on watermarked in-distribution training data for preserving model utility while simultaneously maximizing the change of model's outputs on watermarked out-of-distribution data, thereby achieving effective defense. During deployment, a watermark is randomly sampled from the estimated watermark posterior. This watermark is then added to the input query, and the victim model returns the prediction based on the watermarked input query to users. This proactive defense approach requires only slight fine-tuning of the victim model without the need of full re-training and demonstrates high efficiency in terms of memory and computation during deployment. Rigorous theoretical analysis and comprehensive experimental results demonstrate the efficacy of our proposed method.

1. Introduction

Training deep learning models as commercial products demands significant computational resources and expensive labeled data, thus these deep learning models are considered as a form of intellectual property (IP) (Sun et al., 2023; Hong et al., 2024). Model extraction or stealing is to replicate the functionality of the pre-trained commercial black-box model (victim model) solely through query access. Attackers may utilize these cloned models for financial gain or other purposes. Consequently, there is a pressing need for defense against model extraction. However, safeguarding against model extraction to preserve model IP is challenging, due to the requirements for efficiency, effectiveness, and the preservation of model utility.

Despite the plethora of defense methods developed to safeguard against model extraction, the existing approaches are riddled with several limitations. On one hand, current *active* defense methods exhibit either computation inefficiency during deployment due to the many times of backpropagation during test time (Orekondy et al., 2020; Mazeika et al., 2022) or memory inefficiency due to ensemble of multiple models (Kariyappa et al., 2021b), hindering their practical viability. Alternatively, they may demand costly defensive training with distributionally robust optimization (Wang et al., 2023), necessitating alterations to the original training procedure for re-training. This, in turn, leads to a substantial increase in additional training costs. On the other hand, existing watermark-based methods (Jia et al., 2021a), while offering a form of defense, are *passive* in nature. These methods can only furnish evidence of model theft after the fact but lack the proactive capability to prevent the extraction of the model from occurring in the first place.

To tackle the aforementioned challenges, we propose a novel probabilistic Bayesian active watermarking technique designed to fine-tune the pre-trained victim model to prevent model extraction. We choose a probabilistic approach for its multiple benefits: (1) Adding a probabilistic watermark during deployment increases uncertainty for attackers, making it harder to determine the model's parameters. (2) Introducing inconsistent or incorrect gradient information complicates model extraction for attackers. (3) The method enhances resilience against various adaptive model extraction techniques by introducing significant randomness, increasing the difficulty of reverse engineering. Specifically, in the fine-tuning phase, our primary goal is to make minimal adjustments to the victim model while obtaining a watermark posterior based on observed data. We maximize the

[1]Department of Computer Science, University of Maryland, College Park, USA. Correspondence to: Zhenyi Wang <zwang169@umd.edu>, Heng Huang <heng@umd.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

log-likelihood on watermarked in-distribution (ID) training data to sustain model utility. Simultaneously, we aim to maximize the KL-divergence between the model outputs on watermarked/non-watermarked out-of-distribution (OOD) data, thereby significantly altering the model output to enhance defense against model extraction, illustrated in Figure 1. However, determining the exact watermark posterior is a formidable task due to the intractable nature of analytically computing the watermark posterior probability distribution. Additionally, as the watermark posterior distribution calculation is datapoint-specific, i.e., $q(\boldsymbol{w}|\boldsymbol{x})$, computing the approximate distribution for each datapoint proves computationally intensive. To overcome this challenge, we propose an efficient amortized variational inference algorithm to reduce computation costs, utilizing Wasserstein gradient flow (Ambrosio et al., 2008) in probability measure space. In the deployment phase, with every query to the victim model, a watermark is randomly sampled from the posterior distribution and included in the query as input to the victim model. This eliminates the necessity for test-time backpropagation, ensemble methods, or re-training the model from scratch.

In contrast to existing defense mechanisms, our method boasts several key advantages. Firstly, the probabilistic and dynamic nature of our defense strategy, which varies each time, further complicates the attacker's ability to discern the specific added defenses, substantially enhancing the overall resilience of our approach. Secondly, in contrast to state-of-the-art (SOTA) defensive training approach (Wang et al., 2023) which requires full re-training, our method accomplishes the same objective with only a minimal need for fine-tuning the victim model, thereby eliminating the necessity for re-training. Thirdly, compared to (Orekondy et al., 2020; Mazeika et al., 2022; Kariyappa et al., 2021b), our method exhibits superior memory and computation efficiency during deployment without the necessity for backpropagation and ensemble techniques. Lastly, our watermark-based method is characterized as an *active* defense, effectively thwarting attempts at model extraction. In contrast, conventional watermarking defense methods are *passive*, limited to verifying model theft after the fact.

We systematically conduct extensive experiments across various model extraction settings and datasets to protect the victim model, which is trained using either supervised learning or self-supervised learning. The outcomes reveal that, in contrast to the SOTA defensive training method (Wang et al., 2023), our approach necessitates only minimal fine-tuning of the victim model, resulting in a noteworthy reduction in re-training costs by 87%. Additionally, it achieves $17\times \sim 172\times$ speed up compared to (Orekondy et al., 2020; Mazeika et al., 2022) during inference. Furthermore, our approach surpasses other SOTA defense methods by up to 12% across various query budgets. Meanwhile, we conduct theoretical analysis to provide the performance guarantee for our proposed method.

Our contributions can be summarized in four key aspects:

- We introduce a novel Bayesian active watermarking framework for model extraction defense, requiring only minimal fine-tuning of the pre-trained victim model.
- We propose an efficient amortized variational inference algorithm designed for calculating the watermark posterior distribution.
- Rigorous theoretical analysis is derived to guarantee the effectiveness of our proposed method.
- Comprehensive experiments conducted on various model extraction settings and datasets demonstrate the SOTA performance of our proposed method.

2. Related Work

2.1. Model Extraction Attack

According to the output information provided by the victim model, model extraction attacks (Lowd & Meek, 2005; Tramèr et al., 2016; Wang & Gong, 2018; Oh et al., 2018; Orekondy et al., 2019; Jagielski et al., 2020; Li et al., 2023; Pal et al., 2020; Juuti et al., 2019; Wang, 2021; Kariyappa et al., 2021a; Truong et al., 2021; Sanyal et al., 2022) can be classified into two settings: *soft-label* and *hard-label*. In the *soft-label* setting, the victim model offers softmax probability outputs over the prediction classes to users. In contrast, in the *hard-label* setting, the victim model only provides the top-1 class prediction to users.

According to the query data used by attacker, model extraction techniques can be classified into two categories: data-based and data-free model extraction. *(1) Data-based Model Extraction (DBME):* DBME focuses on extracting the victim model using real dataset (Papernot et al., 2017; Orekondy et al., 2019; Pal et al., 2020). *(2) Data-free Model Extraction (DFME):* DFME, on the other hand, aims to extract the victim model using synthetic data exclusively (Wang, 2021; Kariyappa et al., 2021a; Truong et al., 2021; Sanyal et al., 2022; Hu et al., 2023). These approaches reduce the dataset requirement for stealing the victim model.

2.2. Model Extraction Defense

We categorize existing defense methods into **active** and **passive** defense approaches.

Active Defense against model extraction seeks to *prevent the extraction process*. Current such defense strategies can be classified into three classes: (1) *Output-perturbation-based methods*: P-poison (Orekondy et al., 2020), Adaptive Misinformation (Kariyappa & Qureshi, 2020), GRAD (Mazeika et al., 2022) and ModelGuard (Tang et al., 2024) involve either computation-intensive optimization or more

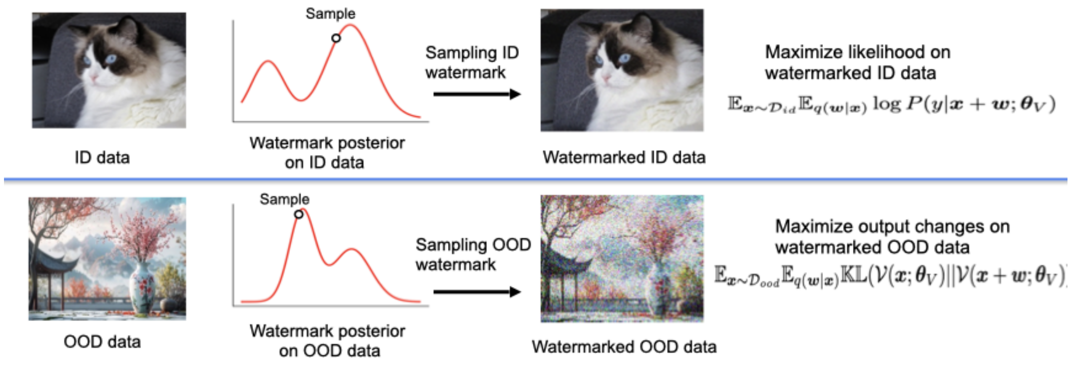
ID data
Sample
Watermark posterior on ID data
Sampling ID watermark
Watermarked ID data
Maximize likelihood on watermarked ID data
$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{id}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}\log P(y|\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V)$
OOD data
Sample
Watermark posterior on OOD data
Sampling OOD watermark
Watermarked OOD data
Maximize output changes on watermarked OOD data
$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V)$

Figure 1. The fine-tuning objective for in-distribution (ID) data is to learn a watermark posterior that maximizes the likelihood on watermarked ID data, thereby maintaining model utility. Conversely, the fine-tuning objective for out-of-distribution (OOD) data is to learn a watermark posterior that maximizes the change in outputs on watermarked OOD data, achieving effective defense.

memory consumption during testing. (2) *Ensemble-based methods* (Kariyappa et al., 2021b) trains a varied collection of models with a discontinuous decision boundary to increase resistance against extraction. However, this approach is memory inefficient due to the storage requirements for multiple pre-trained models. (3) *Defensive training*: While MeCo (Wang et al., 2023) provides memory and computation efficiency by employing defensive training with distributionally robust optimization (Rahimian & Mehrotra, 2019), it does require re-training, which results in a significant additional training cost. In contrast, our method ensures computational and memory efficiency and avoids re-training the victim model from scratch, leading to a substantial reduction in training cost.

Passive Defense can only detect or verify model theft but could not prevent model extraction from happening. Existing methods can be categorized into two classes: (1) *Detection-based methods*: (Juuti et al., 2019; Pal et al., 2021) aim to differentiate between attack queries and benign ones. However, they rely on assuming a known prior distribution of attack query data, leaving them vulnerable to adaptive changes in the attacker's query distribution. Attacker can always change their query data distribution and easily evade these detection, rendering these defenses ineffective. (2) *Verification-based methods*: watermark-based (Adi et al., 2018) methods (Jia et al., 2021a; Szyller et al., 2021), proof-of-learning (Jia et al., 2021b), and dataset inference (Maini et al., 2021) can only verify model theft but could not prevent it. Unlike conventional *passive* watermark methods that merely indicate whether a model has been stolen, our innovative approach is *active*, going beyond detection to actively thwart attempts at model theft.

Our approach falls under the category of active defense, demonstrating the ability to significantly diminish the accuracy of clone models extracted by attackers. In contrast, passive defense methods lack this capability.

3. Preliminary

3.1. Model Extraction Attack

In the presence of a black-box victim model $\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)$, parameterized by $\boldsymbol{\theta}_V$, the attacker leverages an unlabeled dataset $\{\boldsymbol{x}_i\}_{i=1}^{i=N}$ to query the victim model, acquiring corresponding outputs $\boldsymbol{y}_i = \mathcal{V}(\boldsymbol{x}_i;\boldsymbol{\theta}_V)$. Subsequently, the attacker utilizes this labeled dataset, $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{i=N}$, to train a clone model $\mathcal{C}(\boldsymbol{x};\boldsymbol{\theta}_C)$, which is parameterized by $\boldsymbol{\theta}_C$. The attacker's objective is to make the clone model's performance close to that of the victim model. In the *hard-label* setting, the victim model provides only the top-1 class label, whereas in the *soft-label* setting, it outputs class probabilities. Due to space constraints, we include the details of existing model extraction methods in Appendix E.

3.2. Model Extraction Defense

The objective of model extraction defense is to minimize the accuracy of the cloned model while preserving its usefulness for legitimate users. In line with the assumptions presented in existing model extraction defense (Kariyappa & Qureshi, 2020; Kariyappa et al., 2021b; Dziedzic et al., 2022b; Wang et al., 2023), it is presumed that the attack query data consist of out-of-distribution (OOD) samples. This assumption is attributed to several reasons. (1) *Limited Information Exposure*: APIs provide limited information to users, typically only offering access to input-output pairs. This lack of transparency hampers the attacker's ability to gain insights into the specific in-distribution (ID) data used for training (Wang, 2021). (2) *Confidentiality Measures*: APIs do not grant access to their ID training data, which is kept confidential for reasons such as privacy, security, and intellectual property protection. Furthermore, it is crucial to highlight that the *majority of data-based* (Orekondy et al., 2019; Pal et al., 2020) and *all data-free* model extraction (Truong et al., 2021; Kariyappa et al., 2021a; Wang, 2021; Sanyal et al., 2022) methods utilize OOD data to extract the victim model.

4. Method

In this section, we introduce our Bayesian active watermarking method. Specifically, we outline our framework and optimization objective in Section 4.1. Following that, we detail our efficient algorithm for learning the watermark posterior in Section 4.2.

4.1. Bayesian Active Watermarking Framework

Below, we introduce our Bayesian active watermarking framework designed to prevent model extraction while maintaining model utility for legitimate users.

Attack Query Simulation According to the assumptions and reasons presented in Section 3.2, we consider the attack queries to be OOD data. We employ a pseudo-data generator to synthesize OOD data. The OOD data is generated as $\boldsymbol{x} = \mathcal{G}(\boldsymbol{\epsilon}; \boldsymbol{\theta}_G)$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$. $\boldsymbol{\theta}_G$ denotes the parameters of the data generator. Due to space limitations, we put the details about OOD data generation in Appendix D.

Watermark Posterior Learning Objective We propose a **BA**yesian a**C**tive wa**T**ermarking (ACT) defense method to prevent model extraction attack from happening. We opt for a probabilistic approach due to its various advantages: (1) *Increased Uncertainty*: Probabilistic watermark on input queries during deployment adds uncertainty to the extracted model, making it more difficult for attackers to precisely determine the model's parameters. (2) *Complexity for Attackers*: The approach makes the model extraction task more complex by introducing inconsistent/ incorrect information. This complexity can act as a deterrent, requiring attackers to account for the randomness and potentially increasing the difficulty of reverse engineering. (3) *Adaptive Resilience*: The variability introduced by the probabilistic approach can enhance adaptive resilience against different adaptive model extraction techniques. Attackers often rely on consistent patterns/gradients in the model's behavior, which are disrupted by the randomization.

More precisely, given a pre-trained victim model with parameters $\boldsymbol{\theta}_V^0$, the goal is to fine-tune the victim model initialized with $\boldsymbol{\theta}_V^0$ by optimizing the following objective:

$$\max_{q(\boldsymbol{w}|\boldsymbol{x}), \boldsymbol{\theta}_V} \mathcal{L}_d = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{id}} \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})} \log P(y|\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V) \quad (1)$$
$$+ \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{ood}} \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})} \mathbb{KL}(\mathcal{V}(\boldsymbol{x}; \boldsymbol{\theta}_V) || \mathcal{V}(\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V))$$

Here, $\mathcal{D}_{id}$ denotes the ID training data, while $\mathcal{D}_{ood}$ represents synthetic OOD data generated using the method presented earlier. y denotes the ID training data label. The pre-trained model soft-label outputs on synthetic OOD data, denoted as $\boldsymbol{y}'$, are obtained through $\boldsymbol{y}' = \mathcal{V}(\boldsymbol{x}; \boldsymbol{\theta}_V)$. The watermark, denoted by $\boldsymbol{w}$ and matching the input image's size, is treated as a random latent variable. $q(\boldsymbol{w}|\boldsymbol{x})$ denotes the exact watermark posterior distribution. The objective involves maximizing $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{id}} \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})} \log P(y|\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V)$ to ensure model utility on benign queries. Simultaneously, it maximizes $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{ood}} \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})} \mathbb{KL}(\mathcal{V}(\boldsymbol{x}; \boldsymbol{\theta}_V) || \mathcal{V}(\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V))$ to induce maximal changes in model outputs on OOD queries, providing a defense against model extraction. The optimization goal is to find the optimal $q(\boldsymbol{w}|\boldsymbol{x})$ and $\boldsymbol{\theta}_V$, illustrated in Figure 1. However, computing the exact posterior, $q(\boldsymbol{w}|\boldsymbol{x})$, poses a challenge due to the intractable nature of analytically calculating the watermark posterior distribution. Additionally, as the watermark posterior distribution is datapoint-specific (i.e., $q(\boldsymbol{w}|\boldsymbol{x})$), approximating the distribution for each datapoint is computationally intensive.

To address this issue, we propose a novel and efficient amortized variational inference algorithm to significantly alleviate computation costs. Specifically, the watermark posterior is modeled using a stochastic inference neural network, denoted as $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$. This network is structured as an implicit distribution, with $\boldsymbol{\Phi}$ serving as its parameters. The choice of an implicit distribution allows for the representation of a flexible and intricate watermark posterior distribution. This enhances its efficacy in safeguarding against model extraction, primarily attributed to the potential for significantly increased randomness. Specifically, the watermark posterior is represented as the following:

$$\boldsymbol{w} \sim z_{\boldsymbol{\Phi}}(\boldsymbol{x}, \boldsymbol{\gamma}), \text{where}, \boldsymbol{\gamma} \sim \mathcal{N}(0, I) \quad (2)$$

This implicit posterior enables efficient handling of the complexity associated with the posterior density for individual datapoints. The primary objective is to attain an approximate posterior distribution, denoted as $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$, that closely aligns with the exact posterior $q(\boldsymbol{w}|\boldsymbol{x})$. This closeness is measured through the KL divergence, with the aim of minimizing the discrepancy between the approximate and exact posterior distributions as the following:

$$\min_{\boldsymbol{\Phi}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma}) || q(\boldsymbol{w}|\boldsymbol{x})) \quad (3)$$

It is important to emphasize that this inference network is a small-scale convolutional network with less than 1% of the total parameters when compared to the backbone network.

4.2. Efficient Watermark Posterior Learning Algorithm

In this section, we propose an efficient amortized variational inference algorithm for learning the watermark posterior distribution within the probability measure space, i.e., $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$. Before delving into the details, we first introduce some preliminary definitions as follows:

Definition 4.1 (Wasserstein Gradient Flow (Ambrosio et al., 2008))**.** Consider a Wasserstein space $\mathbb{W}^2 = (\mathcal{P}_2(\mathbb{R}^d), W_2)$ (Defined in Appendix B). A curve $(\mu_t)t \geq 0$ of probability measures is identified as a Wasserstein gradient flow for the functional F if it adheres to the following equation:

$$\partial_t \mu_t = \nabla_{W_2} F(\mu_t) := div\left(\mu_t \nabla \frac{\delta F}{\delta \mu}(\mu_t)\right), \quad (4)$$

where $:=$ denotes definition, and $div(\boldsymbol{r}) := \sum_{i=1}^{d} \partial_{\boldsymbol{z}^i} \boldsymbol{r}^i(\boldsymbol{z})$ represents the divergence operator of a vector-valued function $\boldsymbol{r} : \mathbb{R}^d \to \mathbb{R}^d$, with $\boldsymbol{z}^i$ and $\boldsymbol{r}^i$ denoting the i-th element of $\boldsymbol{z}$ and $\boldsymbol{r}$, respectively. The symbol ∇ represents the gradient of a scalar-valued function. The expression $\nabla_{W_2} F(\mu_t) := div(\mu_t \nabla \frac{\delta F}{\delta \mu}(\mu_t))$ defines the Wasserstein gradient of the functional F at the probablity measure μ_t, where $\frac{\delta F}{\delta \mu}(\mu_t)$ stands for the first variation of F at μ_t (defined in Appendix B).

In essence, the Wasserstein Gradient Flow (WGF) can be understood as the trajectory of the probability measure μ_t along the steepest curve of the functional $F(\mu)$ within the Wasserstein space of probability measures (function space). This trajectory initiates at the initial probability measure μ_0 and progressively navigates towards the desired target probability measure π. We choose WGF for learning the watermark posterior for several reasons: (1) *Inference Parameters Efficiency* : WGF can substantially reduce the number of trainable parameters by amortizing the cost across different training data points with only a single set of variational parameters. (2) *Computation Efficiency and Posterior Complexity* : Traditional variational inference needs to trade-off between computation efficiency and posterior complexity. On the contrary, we employ WGF to effectively backpropagate the KL divergence directly into the parameters of the watermark posterior inference network. This network can be selected as arbitrarily complex, thereby preserving the complexity of the watermark posterior while ensuring efficient inference. (3) *Theoretical Guarantees* : WGF is supported by solid theoretical foundations. This provides a principled framework for designing variational inference algorithms and understanding their properties.

To learn the parameters of the approximate watermark posterior distribution $\boldsymbol{\Phi}$, we proceed to calculate the gradient of the KL divergence with respect to $\boldsymbol{\Phi}$ using the chain rule, as outlined below:

$$\begin{aligned}&\nabla_{\boldsymbol{\Phi}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x})) = \\ &\nabla_{\boldsymbol{w}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x})) \frac{\partial \boldsymbol{w}}{\partial \boldsymbol{\Phi}}\end{aligned} \quad (5)$$

Following (Wibisono, 2018), we define the watermark posterior distribution as an energy function $q(\boldsymbol{w}|\boldsymbol{x}) \propto e^{-U}$

$$\begin{aligned}U(\boldsymbol{w}) =& \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{id}} \log P(y|\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V) \\ &+ \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}} \mathbb{KL}(\mathcal{V}(\boldsymbol{x}; \boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x} + \boldsymbol{w}; \boldsymbol{\theta}_V))\end{aligned} \quad (6)$$

To improve computational efficiency, we calculate the Wasserstein gradient $\nabla_{\boldsymbol{w}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x}))$ within the reproducing kernel Hilbert space (RKHS). This choice is motivated by the faster convergence of particles in RKHS compared to MCMC-based sampling methods (Liu et al., 2019; Feng et al., 2017), attributed to the consideration of particle interaction. More precisely, we transform the Wasserstein gradient $\nabla_{W_2} F(\mu_t)$ using the integration transformation $\mathcal{K}_\mu \nabla_{W_2} F(\mu_t) = \int K(\boldsymbol{w}, \boldsymbol{w'}) \nabla_{W_2} F(\mu_t)(\boldsymbol{w'}) d\mu(\boldsymbol{w'})$; where $K(\boldsymbol{w}, \boldsymbol{w'})$ denotes a kernel function and the RKHS space induced by kernel K is denoted as $\mathcal{H}$. In this context, watermark posterior in the kernelized Wasserstein space conforms to the following kernelized WGF: (Liu, 2017):

$$\partial_t \mu_t = div(\mu_t \mathcal{K}_{\mu_t} \nabla \frac{\delta F}{\delta \mu}(\mu_t)). \quad (7)$$

$$F(\mu) := \mathbb{KL}(\mu||\pi) = \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x})) \quad (8)$$

Where μ denotes the distribution of $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$ and π denotes the exact watermark posterior distribution $q(\boldsymbol{w}|\boldsymbol{x})$. Eq. (7) can be interpreted as the WGF in RKHS. This kernelized formulation serves as a deterministic approximation of the WGF in Eq. (4) (Liu & Wang, 2016). By discretizing Eq. (7) and treating each watermark sampled from $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$ as a particle, we derive the following Wasserstein gradient for the watermark posterior. Detailed derivations are provided in Appendix C.

$$\begin{aligned}&\nabla_{\boldsymbol{w}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x})) = \\ &\mathbb{E}_{\boldsymbol{w}\sim z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})} [\underbrace{k(\boldsymbol{w}, \cdot) \nabla_{\boldsymbol{w}} U(\boldsymbol{w})}_{\text{smoothed gradient}} + \underbrace{\nabla_{\boldsymbol{w}} k(\boldsymbol{w}, \cdot)}_{\text{repulsive term}}]\end{aligned} \quad (9)$$

where we use samples $\boldsymbol{w} \sim z_{\boldsymbol{\Phi}}(\boldsymbol{x}, \boldsymbol{\gamma}), \boldsymbol{\gamma} \sim \mathcal{N}(0, I)$ to calculate the expectation in the above equation. The first term in Eq. (9) guides the watermark posterior distribution to simultaneously preserve model utility and resist model extraction. It achieves this by maximizing the energy function of log-likelihood on ID training data and maximizing the change of model outputs on OOD synthetic data. The update is influenced by the kernel-weighted sum of gradients from the watermark sampled from the posterior distribution, resulting in a smoothing effect on the watermark gradients. The second term acts as a repulsive force, preventing the watermark from collapsing into a single mode and thereby promoting diversity in the watermark posterior distribution. In this work, we employ the Gaussian kernel $k(\boldsymbol{w}_i, \boldsymbol{w}_j) = \exp\left(-\frac{(\boldsymbol{w}_i - \boldsymbol{w}_j)^2}{2\sigma^2}\right)$.

In summary, Algorithm 1 presents our Bayesian active watermarking algorithm. In lines 3 to 5, the computation involves calculating the gradient with respect to the parameters $\boldsymbol{\Phi}$ of the watermark posterior network. Then, lines 6 to 7 update the parameters $\boldsymbol{\Phi}$ and victim model parameters $\boldsymbol{\theta}_V$.

Bayesian Active Watermarking Deployment During deployment, the watermark $\boldsymbol{w}$ is sampled from the posterior distribution $z_{\mathbf{\Phi}}(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{\gamma})$ and is incorporated into each query, resulting in $\boldsymbol{y}' = \mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V)$ being delivered to users.

Algorithm 1 Active watermarking for model extraction defense.

```
1: REQUIRE: Pre-trained victim model parameters $\boldsymbol{\theta}_V^0$, OOD data generator, learning rate $\eta$, number of iterations $M$, randomly initialized watermark posterior network parameters $\mathbf{\Phi}$.
2: for $m = 1$ to $M$ do
3:     Sample a mini-batch ID training data and generate synthetic OOD data as $\boldsymbol{x} = \mathcal{G}(\boldsymbol{\epsilon};\boldsymbol{\theta}_G)$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$.
4:     Sample watermark from the posterior for $\boldsymbol{x}$ with $\boldsymbol{w} = z_{\mathbf{\Phi}}(\boldsymbol{x},\boldsymbol{\gamma})$, where, $\boldsymbol{\gamma} \sim \mathcal{N}(0, I)$
5:     Calculate $\nabla_{\mathbf{\Phi}}\mathbb{KL}(z_{\mathbf{\Phi}}(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x}))$, according to Eq. (5) and (9).
6:     Update the watermark posterior network parameters $\mathbf{\Phi}_{m+1} = \mathbf{\Phi}_m - \eta\nabla_{\mathbf{\Phi}}\mathbb{KL}(z_{\mathbf{\Phi}}(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x}))$
7:     Update the victim model parameters by gradient descent $\boldsymbol{\theta}_V^{m+1} = \boldsymbol{\theta}_V^m - \frac{\partial\mathcal{L}_d}{\partial\boldsymbol{\theta}_V}$, where $\mathcal{L}_d$ is calculated by Eq. (1)
8: end for
```

5. Theoretical Analysis

In this section, we provide theoretical analysis for why Bayesian active watermarking can defend against model extraction. First, we prove that $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V))$ on OOD data will decrease the clone model quality on ID test data (Theorem 5.1) and impact of query budget on the clone model generalization error (Theorem 5.2).

Given two distributions $\mathcal{P}$, $\mathcal{Q}$ and the their probability density $p(x)$ and $q(x)$, the total variation distance between $\mathcal{P}$ and $\mathcal{Q}$ is defined as: $\mathbb{TV}(\mathcal{P},\mathcal{Q}) = \frac{1}{2}\mathbb{E}_x[|p(x)-q(x)|]$. Denote by $l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})$ the non-negative learning objective used by attacker, which aims at increasing the similarity between $\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)$ and $\boldsymbol{y}$. During the model extraction attack, the attacker optimizes the following objective:

$$\min_{\boldsymbol{\theta}_C}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]. \tag{10}$$

where $\boldsymbol{y} = \mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V)$. Under our active watermarking settings, the attacker minimizes the following objective:

$$\begin{aligned}L_{\mathcal{D}_{ood}}(\mathcal{C}) :=&\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\\ &\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V))].\end{aligned} \tag{11}$$

The goal of model extraction is to attain high test accuracy with clone model on the ID data. To measure the effectiveness of model extraction, we utilize the disparity in loss between victim model and clone model on the ID data as:

$$Q(\mathcal{C},\mathcal{V}) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{id}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y}) - l(\mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V),\boldsymbol{y})],$$

larger $Q(\mathcal{C},\mathcal{V})$ indicates worse clone model quality.

Theorem 5.1. *Assuming attacker uses cross entropy loss l, we have:*

$$\begin{aligned}&Q(\mathcal{C},\mathcal{V}) \geq -4A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id}) - B+\\ &\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V))\end{aligned} \tag{12}$$

where $B = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}[\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V))]$ is a constant only related to the victim model $\mathcal{V}$.

Theorem 5.1 asserts that maximizing $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}$ $\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V))$ leads to an increase in $Q(\mathcal{C},\mathcal{V})$. Consequently, this will result in a decrease in the quality of the clone model.

Impact of query budget on the clone model generalization In this part, we study the relation between the generalization error of clone model and the number of queries. Attacker employs a set of n examples $S := (\boldsymbol{x}_i,\mathcal{V}(\boldsymbol{x}_i+\boldsymbol{w}_i;\boldsymbol{\theta}_V))_{i=1}^n$ sampled from $\mathcal{D}_{ood}$ to optimize Eq. 11. The optimization goal is expressed as follows:

$$L_S(\mathcal{C}) := \frac{1}{n}\sum_{i=1}^{n}[l(\mathcal{C}(\boldsymbol{x}_i,\boldsymbol{\theta}_C),\mathcal{V}(\boldsymbol{x}_i+\boldsymbol{w}_i,\boldsymbol{\theta}_V))] \tag{13}$$

Let k be the dimension of $\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)$ and $\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)$), i.e., number of classes. Usually the loss l is independently applied to each entry (denoted by $\mathcal{C}_i$ and $\mathcal{V}_i$, $i\in[k]$) i.e.,

$$l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)) = \sum_{i=1}^{k} l_i(\mathcal{C}_i(\boldsymbol{x},\boldsymbol{\theta}_C),\mathcal{V}_i(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V))$$

We follow the commonly used assumption, K-Lipschitz parametrization (Bubeck & Sellke, 2021; Wu et al., 2023), for studying the generalization of clone model. Denote by $\mathcal{C}_i^p := \{\mathcal{C}_i(\boldsymbol{x},\boldsymbol{\theta}_C) : \boldsymbol{\theta}_C\in\mathbb{R}^p, ||\boldsymbol{\theta}_C||_\infty \leq W, i\in[k]\}$ the parameterized hypothesis space, we say $\mathcal{C}_i^p$ is K-Lipschitz parameterized if and only if $\forall\mathcal{C}_i(\cdot,\boldsymbol{\theta}_C),\mathcal{C}_i(\cdot,\boldsymbol{\theta}_C')\in\mathcal{C}_i^p$,

$$||\mathcal{C}_i(\cdot,\boldsymbol{\theta}_C)-\mathcal{C}_i(\cdot,\boldsymbol{\theta}_C')||_{\mathcal{F}} \leq K||\boldsymbol{\theta}_C-\boldsymbol{\theta}_C'||_2,$$

Theorem 5.2. *Denoted by $\mathcal{C}^p := \mathcal{C}_1^p\times\mathcal{C}_2^p\times\cdots\times\mathcal{C}_k^p$. Given an input space $[0,1]^d$. Assume the error functions $l_i(s,y), i\in[k]$ are C-Lipschitz w.r.t. s and $\sup|l_i|\leq a, i\in[k]$. Then, with probability at least $1-\delta$, one has simultaneously for all $\mathcal{C}\in\mathcal{C}^p$:*

$$\begin{aligned}L_{\mathcal{D}_{ood}}(\mathcal{C}) - L_S(\mathcal{C}) \leq& \frac{8akC}{\sqrt{n}}(1+\sqrt{p\ln(K^2W^2pn/a^2)})\\ &+4ak\sqrt{\frac{2\ln(2k/\delta)}{n}}\end{aligned}$$

This theorem indicates that clone model's generalization error on the OOD dataset is limited by $O(\sqrt{\ln(n)}/\sqrt{n})$, where n denotes the number of queries. Within the range $[c,\infty)$, $\sqrt{\ln(n)}/\sqrt{n}$ decreases as n increases, suggesting that more queries result in the clone model's performance alignment with the watermarked OOD distribution. However, attacker aims to enhance the clone model's accuracy on the ID dataset. Overfitting to watermarked OOD data can lead to poorer performance on non-watermarked OOD dataset since our active watermarking aims to maximize the KL-divergence between the scores of watermarked and non-watermarked OOD data. Furthermore, Theorem A.1 states that a decline in the clone model's effectiveness on original OOD distribution correlates with an increase in $Q(\mathcal{C}, \mathcal{V})$, thereby undermining the performance of clone model. Due to the space limitation, we put detailed theorem proof in Appendix A.

Table 1. Accuracy of clone model following the application of defense methods on *CIFAR-10* and *CIFAR-100* using ResNet34-8x as the victim model, which provides **soft labels** to users.

Attack	Defense	**CIFAR10** Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFME	undefended ↓	$87.36 \pm 0.78\%$	$75.23 \pm 1.53\%$	$73.89 \pm 1.29\%$
	RandP ↓	$84.28 \pm 1.37\%$	$70.56 \pm 2.23\%$	$70.03 \pm 2.38\%$
	P-poison ↓	$78.06 \pm 1.73\%$	$66.32 \pm 1.36\%$	$68.75 \pm 1.40\%$
	GRAD ↓	$79.33 \pm 1.68\%$	$65.82 \pm 1.67\%$	$69.06 \pm 1.57\%$
	MeCo ↓	$51.68 \pm 1.96\%$	$46.53 \pm 2.09\%$	$61.38 \pm 2.41\%$
	ACT(Ours) ↓	$\mathbf{46.57 \pm 2.83\%}$	$\mathbf{40.32 \pm 2.96\%}$	$\mathbf{49.25 \pm 2.67\%}$
MAZE	undefended ↓	$45.17 \pm 0.73\%$	$23.28 \pm 1.67\%$	$20.03 \pm 1.79\%$
	RandP ↓	$28.76 \pm 2.38\%$	$22.03 \pm 1.50\%$	$18.79 \pm 1.38\%$
	P-poison ↓	$26.81 \pm 2.19\%$	$20.89 \pm 1.58\%$	$\mathbf{17.08 \pm 2.28\%}$
	GRAD ↓	$26.06 \pm 1.81\%$	$21.18 \pm 1.58\%$	$18.09 \pm 1.72\%$
	MeCo ↓	$21.89 \pm 2.07\%$	$18.75 \pm 2.11\%$	$17.95 \pm 1.46\%$
	ACT(Ours) ↓	$\mathbf{18.09 \pm 2.31\%}$	$\mathbf{16.57 \pm 2.06\%}$	$17.21 \pm 1.97\%$

Attack	Defense	**CIFAR100** Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFME	undefended ↓	$58.72 \pm 2.82\%$	$28.36 \pm 1.97\%$	$27.28 \pm 2.08\%$
	RandP ↓	$41.69 \pm 2.91\%$	$22.75 \pm 2.19\%$	$23.61 \pm 2.70\%$
	P-poison ↓	$38.72 \pm 3.06\%$	$20.87 \pm 2.61\%$	$21.89 \pm 2.93\%$
	GRAD ↓	$39.07 \pm 2.72\%$	$20.71 \pm 2.80\%$	$22.08 \pm 2.78\%$
	MeCo ↓	$29.57 \pm 1.97\%$	$12.18 \pm 1.05\%$	$10.79 \pm 1.36\%$
	ACT(Ours) ↓	$\mathbf{23.95 \pm 2.38\%}$	$\mathbf{10.09 \pm 2.53\%}$	$\mathbf{6.26 \pm 2.38\%}$

6. Experiments

6.1. Setup

Baselines: We compare with the following SOTA model extraction attack and defense baselines. **Attack baselines :** *Soft-label attack*: MAZE (Kariyappa et al., 2021a) and DFME (Truong et al., 2021). *Hard-label attack*: DFMS-HL (Sanyal et al., 2022). We do not compare with ZSDB3KD (Wang, 2021) due to its substantial query requirements and notably slow performance. **Defense baselines :** Random output perturbation (RandP), Prediction-poisoning (P-poison) (Orekondy et al., 2020), GRAD (Mazeika et al., 2022) and MeCo (Wang et al., 2023). Following (Wang et al., 2023), we set an output-perturbation budget of 1.0, i.e., $||\boldsymbol{y} - \vec{\boldsymbol{y}}||_1 \leq 1.0$, where $\boldsymbol{y}$ and $\vec{\boldsymbol{y}}$ represent unperturbed and perturbed model outputs, respectively. This is applied to RandP, P-poison, and GRAD to enhance their defense by inducing more substantial output perturbations.

Datasets We assess various defense methods using datasets such as MNIST, CIFAR10, CIFAR100 (Krizhevsky, 2009), MiniImageNet (Vinyals et al., 2016).

Implementation Details The watermark posterior inference network is a small-scale two-block residual network, similar to the noise2net structure (Hendrycks et al., 2021) but with a much smaller number of parameters, which only accounts for less than 1% parameters of the pre-trained victim model. For soft-label setting, following (Truong et al., 2021), we set the attack query budget to be 20M for CIFAR10, 200M for CIFAR100 and 200M for Mini-ImageNet, respectively. For hard-label setting, following (Sanyal et al., 2022), we set the attack query budget to be 8M for CIFAR10, 10M for CIFAR100 and 10M for Mini-ImageNet. We conduct each experiment for five runs and report the mean and the standard deviation of the results.

6.2. Results of Defensive Performance against DFME

Defense Effectiveness Evaluation To assess the efficacy of various defense methods, we compare the test accuracy of the clone model across different defense strategies, aiming for lower clone model accuracy as an indication of improved defense performance. For CIFAR10, CIFAR100 and Mini-ImageNet datasets, we employ the ResNet34-8x (He et al., 2016) as the victim model, while utilizing ResNet18-8x (He et al., 2016), MobileNetV2 (Sandler et al., 2018), and DenseNet121 (Huang et al., 2017) as diverse clone model architectures. For the MNIST dataset, LeNet5 (LeCun et al., 1998) serves as the victim model, with LeNet5, LeNet5-Half, and LeNet5-1/5 serving as distinct clone model architectures. LeNet5-Half and LeNet5-1/5 have half and one-fifth of the number of convolutional filters in each layer compared to LeNet5, respectively. The defense performance of various methods on CIFAR10 and CIFAR100 with *soft-label* is presented in Table 1. We do not compare various defense methods w.r.t the MAZE attack on CIFAR100 due to the low clone model accuracy achieved by MAZE, which is less than 6%. Additionally, the defense performance with *hard-label* is shown in Table 2. The results on Mini-ImageNet is shown in Table 8 in Appendix. Notably, our method (ACT) exhibits a significant improvement over the SOTA defense method by 6% to 12%, showing the superiority of our approach.

This superiority stems from our method's ability to learn watermark posteriors that maximally alter model outputs on OOD data. In contrast, the SOTA method, MeCo (Wang et al., 2023) lacks explicit optimization on OOD query data, resulting in inferior performance. On the other hand, (1) RandP, P-poison, and GRAD typically preserve the hard-label for most benign and attack queries, rendering these defense methods ineffective as attackers can still extract valuable information. (2) P-poison requires a randomly initialized surrogate attacker model, functioning as an adversary model. (3) GRAD necessitates knowledge of the attack query set to train the surrogate model, resulting in surrogates with significant gaps compared to the attacker model. These methods exhibit worse performance because the distribution of attack query data and the attacker model remain

Table 2. Accuracy of clone model by applying various defense methods on *CIFAR-10* and *CIFAR-100* with ResNet34-8x serving as the victim model, providing users with **hard labels**.

Attack	Defense	**CIFAR10** Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	$84.67 \pm 1.90\%$	$79.28 \pm 1.87\%$	$68.87 \pm 2.08\%$
	RandP ↓	$84.02 \pm 2.31\%$	$78.71 \pm 1.93\%$	$68.16 \pm 2.23\%$
	P-poison ↓	$84.06 \pm 1.87\%$	$79.02 \pm 1.96\%$	$68.05 \pm 2.17\%$
	GRAD ↓	$84.28 \pm 1.95\%$	$78.83 \pm 1.91\%$	$68.11 \pm 1.93\%$
	MeCo ↓	$76.86 \pm 2.09\%$	$\mathbf{71.22 \pm 1.87\%}$	$62.33 \pm 2.01\%$
	ACT(Ours) ↓	$\mathbf{73.93 \pm 2.67\%}$	$71.97 \pm 2.08\%$	$\mathbf{61.08 \pm 2.39\%}$
Attack	**Defense**	**CIFAR100** Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	$72.57 \pm 1.28\%$	$62.71 \pm 1.68\%$	$63.58 \pm 1.79\%$
	RandP ↓	$72.43 \pm 1.43\%$	$62.06 \pm 1.82\%$	$63.16 \pm 1.73\%$
	P-poison ↓	$71.83 \pm 1.32\%$	$61.83 \pm 1.79\%$	$62.73 \pm 1.91\%$
	GRAD ↓	$71.89 \pm 1.37\%$	$62.60 \pm 1.71\%$	$62.57 \pm 1.80\%$
	MeCo ↓	$59.30 \pm 1.70\%$	$55.32 \pm 1.65\%$	$56.80 \pm 1.86\%$
	ACT(Ours) ↓	$\mathbf{55.38 \pm 1.97\%}$	$\mathbf{51.46 \pm 1.89\%}$	$\mathbf{52.29 \pm 2.03\%}$

unknown to the defender in DFME. In contrast, ACT excels without the need for a surrogate model. (1) Our method includes a probabilistic watermark on input queries during deployment introduces uncertainty to the extracted model, rendering it more challenging for attackers to precisely discern the functionality of the victim model. (2) Bayesian active watermarking complicates the model extraction task by introducing inconsistent or incorrect gradient information for attackers. They must contend with randomness, potentially heightening the difficulty of reverse engineering.

Model Utility Evaluation. To assess the utility of the victim model incorporating various defense methods, we evaluate the benign test accuracy using various defense strategies, as presented in Table 3. It is noted that ACT maintains comparable model utility compared to other defense methods.

Table 3. Evaluation of victim model's utility through test accuracy.

Method	MNIST	CIFAR10	CIFAR100
undefended(upper bound)	$98.91 \pm 0.16\%$	$94.91 \pm 0.37\%$	$76.71 \pm 1.25\%$
RandP ↑	$98.52 \pm 0.19\%$	$93.98 \pm 0.28\%$	$75.23 \pm 1.39\%$
P-poison ↑	$98.87 \pm 0.35\%$	$94.58 \pm 0.61\%$	$75.42 \pm 1.21\%$
GRAD ↑	$98.73 \pm 0.31\%$	$\mathbf{94.65 \pm 0.67\%}$	$75.60 \pm 1.45\%$
MeCo ↑	$98.63 \pm 0.28\%$	$94.17 \pm 0.56\%$	$75.36 \pm 0.68\%$
ACT(Ours) ↑	$\mathbf{98.90 \pm 0.37\%}$	$94.31 \pm 0.75\%$	$\mathbf{75.78 \pm 0.73\%}$

Legitimate Use in Edge Case We perform a set of experiments to evaluate the edge case where the benign data is different from the ID data. Specifically, the pre-trained victim model is trained on CIFAR10. After adding defense, the protected model is tested on CINIC-10 (Darlow et al., 2018), which consists of images from both CIFAR10 and ImageNet. The images from ImageNet are resized to the same size as CIFAR10. Since the test images consist of images from ImageNet and the pre-trained victim model is trained on CIFAR10, this naturally simulates the natural distribution shift. These queries represent the benign queries that are similar to the ID training data but different from it, representing the edge cases in real applications. The results are presented in table 9 in Appendix. We can observe that our approach only slightly affects those edge cases.

6.3. Results of Defensive Performance against DBME

We utilize traditional Knockoff Nets (Orekondy et al., 2019) and Jacobian-Based Dataset Augmentation (JBDA) (Papernot et al., 2017) with real dataset for data-based model extraction (DBME). The employed attack query dataset is similar to the training data used for training the victim model. The results are presented in Table 10 in Appendix and show that our method outperforms baselines by more than 6% in many cases when defending against DBME.

6.4. Robustness to Adaptive Attack

To assess the robustness of our approach against adaptive attacks performed by potential adversaries, we conduct two distinct types of adaptive attack scenarios. In the first scenario, we consider an adversary who possesses knowledge of the defender's strategy. In this setting, the attacker not only has insight into the defender's approach but also incorporates watermarking into their input data as an adaptive countermeasure. In the second scenario, the attacker adopts a strategy where they solely rely on the top-1 class label (hard-label) instead of using soft-label information to carry out model stealing. The results are presented in Table 11 and 12 in Appendix, respectively. The findings affirm the robustness of our approach against these adaptive attacks. This resilience can be attributed to two key factors: (1) our principled Bayesian approach, introducing significant uncertainty and randomness to confound attackers, and (2) our watermark learning objective, which maximally alters model outputs on OOD queries, generating a wealth of misleading and inconsistent gradient information for attackers.

Stronger Adaptive Attack We opt to diversify the architectures of the surrogate models by training a collection of surrogate models $A_1, A_2, \cdots, A_{10}$, including ResNet-18, ResNet-34, ResNet-50, MobileNetV2 and DenseNet121 network. We train two surrogate models for each network architecture and have 10 surrogate models. For each surrogate model, we randomly sample a subclasses of labeled data from ImageNet (Deng et al., 2009). The surrogate model is trained on the ImageNet subset with the same learning objective and training procedure as our victim model fine-tuning objective for learning the watermark posterior.

During model extraction, the attacker samples a watermark from each watermark posterior inference network associated with each surrogate model. They then utilize the expectation of these sampled watermarks and combine them with the query inputs to form the inputs to the clone model. Subsequently, the attacker engages in the same clone model optimization procedure to extract the victim model. We refer to this adaptive attack as the *Ensemble Adaptive Attack*.

This adaptive attack is substantially stronger since the attacker knows all the details about our defense strategy including fine-tuning objective and watermark posterior inference network architecture. By utilizing an ensemble adaptive attack strategy, the attacker can achieve a more precise estimation of the watermark posterior. Empirically, it significantly increases the clone model accuracy by more than 8% in Table 4. However, ACT still significantly outperforms these adaptive attacks since our probabilistic active watermarking varies with input. We can sample different watermark samples from different posterior. It is difficult for attacker to correctly restore the victim model outputs.

Table 4. Accuracy of clone model following the application of the **ensemble adaptive attack** by attacker on *CIFAR-10*, using ResNet34-8x as the victim model.

Attack	Defense	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFME	undefended ↓	$87.36 \pm 0.78\%$	$75.23 \pm 1.53\%$	$73.89 \pm 1.29\%$
	ACT ↓	$46.57 \pm 2.83\%$	$40.32 \pm 2.96\%$	$49.25 \pm 2.67\%$
	Ensemble Adaptive ↑	$\mathbf{52.68 \pm 2.91\%}$	$\mathbf{45.19 \pm 3.07\%}$	$\mathbf{57.83 \pm 2.82\%}$

6.5. Ablation Study

Impact of query budget on clone model performance To assess the impact of varying query budgets on the performance of clone model, we experiment with different query budgets adopted by the attacker. The results are visualized in Figure 2 in Appendix, showing that ACT significantly outperforms SOTA defenses across different query budgets.

Impact of victim model architecture To assess the influence of various victim model architectures, we modified the victim model to be GoogLeNet (Szegedy et al., 2015). The results can be found in Table 6 in Appendix.

Training Efficiency Evaluation To assess the training cost in comparison to the SOTA method, MeCo (Wang et al., 2023), we conduct an efficiency evaluation presented in Table 14 with A6000 GPU. It is evident that our method significantly reduces the training cost by more than 87%.

Table 5. Efficiency evaluation (seconds) during deployment

Algorithm	CIFAR10	CIFAR100
P-poison	223.68 ± 2.78	1126.83 ± 8.71
GRAD	107.29 ± 1.66	317.83 ± 4.29
MeCo	6.06 ± 0.80	6.53 ± 0.82
ACT (Ours)	$\mathbf{6.03 \pm 0.91}$	$\mathbf{6.42 \pm 0.87}$

Test-time Efficiency Evaluation To assess the test-time efficiency of various defense methods, we conduct an evaluation on test data. The results, presented in Table 5, show that our method achieves $17\times \sim 172\times$ speed up compared to P-poison and GRAD while maintaining comparable efficiency to MeCo. For memory efficiency, we provide the results in Table 15 in Appendix.

7. Conclusion

We introduce an innovative Bayesian active watermarking framework designed to thwart model extraction. We also present an efficient amortized variational inference algorithm for learning the watermark posterior distribution. The method only requires minimal fine-tuning of the victim model. Notably, in contrast to existing watermark-based approaches employing passive defense strategies, our method can actively prevent model extraction. Through rigorous theoretical analysis and extensive experiments encompassing diverse model extraction scenarios and datasets, our method demonstrates remarkable effectiveness and efficiency.

Limitations A limitation of our current work is that the victim model is not sufficiently large. In future work, we plan to address this by integrating a mixture of experts (MoE) (Chowdhury et al., 2023) to upscale the victim model. This enhancement will increase the model's capacity and expand its applicability in real-world scenarios. We will leave the detailed and deeper exploration of this expansion for future research efforts.

Developing defense methods for model extraction can potentially serve as a foundation for attackers to devise new attack strategies. By enhancing the complexity and flexibility of the watermark posterior distribution, we aim to fortify our defense against these potential threats. Specifically, we plan to propose the design of a hierarchical Bayesian model, where the global component remains fixed, encapsulating the invariant active watermarking defense that serves as the defense mechanism foundation. Meanwhile, the local components are designed to be more flexible, adapting over time to counter new emerging attacks while preserving the core defense strategy. This hierarchical Bayesian active watermarking mechanism will facilitate minimal yet efficient adaptations of our defense strategy, ensuring robust protection against novel attacks.

Impact Statement

Positive Impact: This paper proposes a method to prevent API from extraction. Many organizations invest significant resources in developing advanced machine learning models. By defending against model extraction, they can safeguard their intellectual property and proprietary algorithms. Furthermore, defending against model extraction helps prevent unauthorized access to the sensitive information contained in API. *Negative Impact:* These model extraction defenses can make it difficult for researchers to study and improve upon existing black-box pre-trained models. This can slow down the overall progress of AI research. Furthermore, Limiting the ability to extract the victim model can reduce the dissemination of knowledge, making it harder to learn from and build upon existing pre-trained models.

Acknowledgements

This work was partially supported by NSF IIS 2347592, 2347604, 2348159, 2348169, DBI 2405416, CCF 2348306, CNS 2347617.

References

Adi, Y., Baum, C., Cisse, M., Pinkas, B., and Keshet, J. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX Security 18)*, 2018.

Ambrosio, L., Gigli, N., and Savare, G. Gradient flows: In metric spaces and in the space of probability measures. *(Lectures in Mathematics. ETH)*, 2008.

Bartlett, P. For valid generalization the size of the weights is more important than the size of the network. *Advances in neural information processing systems*, 9, 1996.

Bubeck, S. and Sellke, M. A universal law of robustness via isoperimetry. *Advances in Neural Information Processing Systems*, 34:28811–28822, 2021.

Chen, X. and He, K. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Chewi, S., Le Gouic, T., Lu, C., Maunu, T., and Rigollet, P. Svgd as a kernelized wasserstein gradient flow of the chi-squared divergence. In *Advances in Neural Information Processing Systems*, 2020.

Chowdhury, M. N. R., Zhang, S., Wang, M., Liu, S., and Chen, P.-Y. Patch-level routing in mixture-of-experts is provably sample-efficient for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6074–6114. PMLR, 2023.

Darlow, L. N., Crowley, E. J., Antoniou, A., and Storkey, A. J. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dziedzic, A., Dhawan, N., Kaleem, M. A., Guan, J., and Papernot, N. On the difficulty of defending self-supervised learning against model extraction. In *International Conference on Machine Learning*, pp. 5757–5776. PMLR, 2022a.

Dziedzic, A., Kaleem, M. A., Lu, Y. S., and Papernot, N. Increasing the cost of model extraction with calibrated proof of work. In *International Conference on Learning Representations*, 2022b.

Feng, Y., Wang, D., and Liu, Q. Learning to draw samples with amortized stein variational gradient descent. *The Conference on Uncertainty in Artificial Intelligence*, 2017.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2014.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8340–8349, 2021.

Hong, Z., Wang, Z., Shen, L., Yao, Y., Huang, Z., Chen, S., Yang, C., Gong, M., and Liu, T. Improving non-transferable representation learning by harnessing content and style. In *The Twelfth International Conference on Learning Representations*, 2024.

Hu, Z., Shen, L., Wang, Z., Wu, B., Yuan, C., and Tao, D. Learning to learn from APIs: Black-box data-free meta-learning. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 13610–13627. PMLR, 2023.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A., and Papernot, N. High accuracy and high fidelity extraction of neural networks. In *USENIX Security 2020*, 2020.

Jia, H., Choquette-Choo, C. A., Chandrasekaran, V., and Papernot, N. Entangled watermarks as a defense against model extraction. In *30th USENIX Security Symposium*, 2021a.

Jia, H., Yaghini, M., Choquette-Choo, C. A., Dullerud, N., Thudi, A., Chandrasekaran, V., and Papernot, N. Proof-of-learning: Definitions and practice. In *42nd IEEE Symposium on Security and Privacy*, 2021b.

Juuti, M., Szyller, S., Marchal, S., and Asokan, N. Prada: Protecting against dnn model stealing attacks. In *IEEE European Symposium on Security and Privacy*, 2019.

Kariyappa, S. and Qureshi, M. K. Defending against model stealing attacks with adaptive misinformation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Kariyappa, S., Prakash, A., and Qureshi, M. K. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021a.

Kariyappa, S., Prakash, A., and Qureshi, M. K. Protecting {dnn}s from theft using an ensemble of diverse models. In *International Conference on Learning Representations*, 2021b.

Krizhevsky, A. Learning multiple layers of features from tiny images, 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, G., Xu, G., Guo, S., Qiu, H., Li, J., and Zhang, T. Extracting robust models with uncertain examples. In *International Conference on Learning Representations*, 2023.

Liu, C., Zhuo, J., Cheng, P., Zhang, R., and Zhu, J. Understanding and accelerating particle-based variational inference. In *International Conference on Machine Learning*, pp. 4082–4092. PMLR, 2019.

Liu, Q. Stein variational gradient descent as gradient flow. *Advances in Neural Information Processing Systems*, 2017.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in Neural Information Processing Systems*, 2016.

Lowd, D. and Meek, C. Adversarial learning. In *ACM SIGKDD international conference on Knowledge discovery in data mining*. Association for Computing Machinery, 2005.

Maini, P., Yaghini, M., and Papernot, N. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021.

Mazeika, M., Li, B., and Forsyth, D. How to steer your adversary: Targeted and efficient model stealing defenses with gradient redirection. In *International Conference on Machine Learning*, 2022.

Oh, S. J., Augustin, M., Schiele, B., and Fritz, M. Towards reverse-engineering black-box neural networks. In *International Conference on Learning Representations*, 2018.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Orekondy, T., Schiele, B., and Fritz, M. Knockoff nets: Stealing functionality of black-box models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Orekondy, T., Schiele, B., and Fritz, M. Prediction poisoning: Towards defenses against dnn model stealing attacks. In *International Conference on Learning Representations*, 2020.

Pal, S., Gupta, Y., Shukla, A., Kanade, A., Shevade, S., and Ganapathy, V. Activethief: Model extraction using active learning and unannotated public data. In *AAAI Conference on Artificial Intelligence*, volume 34, 2020.

Pal, S., Gupta, Y., Kanade, A., and Shevade, S. Stateful detection of model extraction attacks, 2021.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In *ACM Asia Conference on Computer and Communications Security*, 2017.

Rahimian, H. and Mehrotra, S. Distributionally robust optimization: A review. 2019.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Sanyal, S., Addepalli, S., and Babu, R. V. Towards data-free model stealing in a hard label setting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Srebro, N., Sridharan, K., and Tewari, A. Smoothness, low noise and fast rates. *Advances in neural information processing systems*, 23, 2010.

Sun, Y., Liu, T., Hu, P., Liao, Q., Fu, S., Yu, N., Guo, D., Liu, Y., and Liu, L. Deep intellectual property protection: A survey. 2023.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Szyller, S., Atli, B. G., Marchal, S., and Asokan, N. Dawn: Dynamic adversarial watermarking of neural networks. In *ACM Multimedia*, 2021.

Tang, M., Dai, A., DiValentin, L., Ding, A., Hass, A., Gong, N. Z., and Chen, Y. Modelguard: Information-theoretic defense against model extraction attacks. *USENIX Security 2024*, 2024.

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. Stealing machine learning models via prediction apis. In *USENIX Security*, 2016.

Truong, J.-B., Maini, P., Walls, R. J., and Papernot, N. Data-free model extraction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 2016.

von Luxburg, U. and Bousquet, O. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–695, 2004.

Wang, B. and Gong, N. Z. Stealing hyperparameters in machine learning. In *In the 39th IEEE Symposium on Security and Privacy*, 2018.

Wang, Z. Zero-shot knowledge distillation from a decision-based black-box model. In *International Conference on Machine Learning*, 2021.

Wang, Z., Shen, L., Liu, T., Duan, T., Zhu, Y., Zhan, D., Doermann, D., and Gao, M. Defending against data-free model extraction by distributionally robust defensive training. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Wibisono, A. Sampling as optimization in the space of measures: The langevin dynamics as a composite optimization problem. In *Conference on Learning Theory*, pp. 2093–3027. PMLR, 2018.

Wu, Y. and Yang, Y. *Information-theoretic methods in high-dimensional statistics S*. http://www.stat.yale.edu/ yw562/teaching/598/lec14.pdf, 2016.

Wu, Y., Huang, H., and Zhang, H. A law of robustness beyond isoperimetry. In *International Conference on Machine Learning*, pp. 37439–37455. PMLR, 2023.

A. Theorem Proofs

Theorem A.1. *If* $\sup l \leq A$

$$Q(\mathcal{C},\mathcal{V}) \geq \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - 4A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id})$$

Proof. First we consider $|\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{id}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]|$

$$\begin{aligned}
&|\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{id}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]| && (14)\\
&= |\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})p_{id}(\boldsymbol{x},\boldsymbol{y})] - \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})p_{ood}(\boldsymbol{x},\boldsymbol{y})|] && (15)\\
&= |\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})(p_{id}(\boldsymbol{x},\boldsymbol{y}) - p_{ood}(\boldsymbol{x},\boldsymbol{y}))]| && (16)\\
&\leq \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[|l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})||p_{id}(\boldsymbol{x},\boldsymbol{y}) - p_{ood}(\boldsymbol{x},\boldsymbol{y})|] && (17)\\
&\leq A\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[|p_{id}(\boldsymbol{x},\boldsymbol{y}) - p_{ood}(\boldsymbol{x},\boldsymbol{y})|] && (18)\\
&= 2A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id}). && (19)
\end{aligned}$$

Analogously, we have

$$|\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{id}}[l(\mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V),\boldsymbol{y})] - \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V),\boldsymbol{y})]| \leq 2A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id}).$$

With the above derivation, we have

$$\begin{aligned}
Q(\mathcal{C},\mathcal{V}) &= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{id}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y}) - l(\mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]\\
&\geq \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - 2A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id}) - \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - 2A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id})
\end{aligned} \quad (20)$$

Notice when $\boldsymbol{y} = \mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V)$, $\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{V}(\boldsymbol{x},\boldsymbol{\theta}_V),\boldsymbol{y})] = 0$. Thus,

$$Q(\mathcal{C},\mathcal{V}) \geq \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})] - 4A\mathbb{TV}(\mathcal{D}_{ood},\mathcal{D}_{id}) \quad (21)$$

□

A.1. Proof of Theorem 5.1.

Recall with the active watermarking, the attacker will optimize the objective Eq. 11. Assuming the attacker utilizes cross-entropy loss as the learning objective l, we have

Lemma A.2. *The optimal solution of Eq. 11 with cross-entropy objective is* $\forall \boldsymbol{x}$,

$$\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C) = \mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)].$$

Proof. If the learning objective is cross-entropy loss, Eq. 11 becomes

$$\min_{\boldsymbol{\theta}_C} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood},\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[-\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)\cdot\log\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)],$$

where $\cdot$ represent dot product of two vectors. Assuming the representation capability of the clone model is large enough, we have

$$\begin{aligned}
&\min_{\boldsymbol{\theta}_C} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood},\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[-\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)\cdot\log\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)]\\
&= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[\min_{\boldsymbol{\theta}_C}\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[-\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)\cdot\log\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)]]\\
&= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[\min_{\boldsymbol{\theta}_C} -\log\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)\cdot\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)]]
\end{aligned}$$

The solution of $\min_{\boldsymbol{\theta}_C} -\log\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)\cdot\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)]$ is $\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C) = \mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)]$. □

Lemma A.3. *Assuming the attacker achieve optimal model extraction solution (Lemma A.2), we have*

$$\begin{aligned}&\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]\\ \geq&\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V))-B,\end{aligned} \tag{22}$$

where $B=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V))]$ is a constant only related to the victim model $\mathcal{V}$.

Proof. Given the optimal solution

$$\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C)=\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)].$$

$$\begin{aligned}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V))]\\&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[-\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C))]\\&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[-\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log(\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)])].\end{aligned} \tag{23}$$

by Jensen's inequality we have

$$\begin{aligned}&\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[l(\mathcal{C}(\boldsymbol{x},\boldsymbol{\theta}_C),\boldsymbol{y})]\\&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[-\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log(\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)])]\\&\geq\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[-\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[\log\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)]]\\&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}[-\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\cdot\log\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V)\\&=\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}\mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\boldsymbol{x})}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V))-\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_{ood}}[\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)\log(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_T))]\end{aligned} \tag{24}$$

□

Based on Theorem A.1 and Lemma A.3, we can derive the present theorem.

A.2. Proof of Theorem 5.2.

We first introduce some preliminary knowledge about the complexity bound. Rademacher complexity (Bartlett, 1996) measures the richness of a function class. For a set of vectors $\mathcal{A}\subset\mathbb{R}^n$, the Rademacher complexity is defined as $R(\mathcal{A}):=\frac{1}{n}\mathbb{E}_{\sigma_1,\ldots,\sigma_n\in\{-1,1\}}\left[\sup_{a\in\mathcal{A}}\sum_{i=1}^n\sigma_i a_i\right]$. Given a loss function l, a hypothesis class $\mathcal{F}$, and a training set $S=\{(x_1,y_1),...,(x_n,y_n)\}$, denote by $l\circ\mathcal{F}:=\{l(f(\cdot),\cdot):f\in\mathcal{F}\}$ and $l\circ\mathcal{F}\circ S:=\{(l(f(x_1),y_1),...,l(f(x_n),y_n)):f\in\mathcal{F}\}$. The Rademacher complexity of the set $l\circ\mathcal{F}\circ S$ is given by $R(l\circ\mathcal{F}\circ S):=\frac{1}{n}\mathbb{E}_{\sigma_1,\ldots,\sigma_n\in\{-1,1\}}\left[\sup_{f\in\mathcal{F}}\sum_{i=1}^n\sigma_i l(f(x_i),y_i))\right]$.

For an arbitrary function $f\in\mathcal{F}$, the generation gap between the population error $\mathcal{L}_{\mathcal{D}}(f)$ and the training error $\mathcal{L}_S(f)$ is bounded by the Rademacher complexity of the function space $l\circ\mathcal{F}\circ S$. Assume that $\forall f\in\mathcal{F},\forall x\in\mathcal{X},|l(f(x),y)|\leq a$. According to Theorem 26.5 of (Shalev-Shwartz & Ben-David, 2014), with probability at least $1-\delta$, for all $f\in\mathcal{F}$,

$$\mathcal{L}_{\mathcal{D}}(f)-\mathcal{L}_S(f)\leq 2R(l\circ\mathcal{F}\circ S)+4a\sqrt{\frac{2\ln(2/\delta)}{n}}. \tag{25}$$

The contraction lemma of Rademacher complexity (Lemma 26.9 of (Shalev-Shwartz & Ben-David, 2014)) shows that for a given space $\mathcal{A}$ and a L-lipschitz function $h:\mathcal{A}\to\mathbb{R}$, we have $R(h\circ\mathcal{A})\leq L\cdot R(\mathcal{A})$. Thus, if the error function $l(s,y)$ is C-Lipschitz *w.r.t.* $\forall s\in\{f(x):f\in\mathcal{F},x\in\mathcal{X}\}$, we have

$$R(l\circ\mathcal{F}\circ S)\leq C\cdot R(\mathcal{F}\circ S). \tag{26}$$

Notice, many popular error functions $l(s,y)$, *e.g.* cross-entropy loss with softmax activation and mean squared loss, is $\Theta(1)$-Lipschitz *w.r.t.* $\forall s\in\{f(x):f\in\mathcal{F},x\in\mathcal{X}\}$.

To estimate the Rademacher complexity, we rely on another complexity measure named covering number. Given a space $\mathcal{A}$, denote by $\mathcal{N}(\mathcal{A},\epsilon,||\cdot||)$ the minimum number of $||\cdot||$-norm balls with radius ϵ needed to completely cover $\mathcal{A}$. With Dudley's integral, (von Luxburg & Bousquet, 2004; Srebro et al., 2010) showed that the Rademacher complexity of a given set is upper bounded by a function related to the number of covering of the set. Based on their results, we have the following generalization gap:

Lemma A.4 (Theorem 16 of (von Luxburg & Bousquet, 2004))**.** *Denote by $\mathcal{X} = [0,1]^d$ the input space, given a hypothesis space $\mathcal{F} = \{f : \mathcal{X} \to [0,1]\}$ and the norm $||\cdot||_\mathcal{F} : ||f||_\mathcal{F} = \sup_{x\in\mathcal{X}} |f(x)|$. If $l(s,y)$ is C-Lipschitz w.r.t. s for all $y \in [0,1]$ and $\sup|l| < a$. With probability $1-\delta$, we have for all $f \in \mathcal{F}$,*

$$L_\mathcal{D}(f) - L_S(f) \le 4a\sqrt{\frac{2\ln(2/\delta)}{n}} + C\inf_{\epsilon>0}\left\{8\epsilon + \frac{4\sqrt{2}}{\sqrt{n}}\int_\epsilon^{2a}\sqrt{\ln\mathcal{N}(\mathcal{F},u,||\cdot||_\mathcal{F})}\,du\right\}, \tag{27}$$

where n is the number of training samples.

Lemma A.5 (Covering number of $\mathcal{C}_i^p$)**.** *Given an input space $[0,1]^d$ and a K-Lipschitz parameterized hypothesis space $\mathcal{C}_i^p$, we have for every $\epsilon > 0$,*

$$\mathcal{N}(\mathcal{C}_i^p,\epsilon,||\cdot||_\mathcal{F}) \le (\frac{KW\sqrt{p}}{\epsilon})^p,$$

where W is the ℓ_∞ bound of model parameters, i.e., $||\theta||_\infty \le W$

where $||f||_\mathcal{F} := \sup_{\boldsymbol{x}} |f(\boldsymbol{x})|$ is the norm of the function.

Proof.

Lemma A.6 (Theorem 14.2 of (Wu & Yang, 2016))**.** *Given a space $V \in \mathbb{R}^d$. Denote by $\mathcal{N}(V,\epsilon,||\cdot||_p)$ the ϵ-covering of V with norm $||\cdot||_p, p \ge 1$, we have*

$$\mathcal{N}(V,\epsilon,||\cdot||_p) \le (\frac{3}{\epsilon})^d\frac{\mathrm{vol}(V)}{\mathrm{vol}(B_p)},$$

where $\mathrm{vol}(V)$ is the volume of V and B_p is a unit $||\cdot||_p$-norm ball.

Denote by $\mathcal{W}_\epsilon$ the ϵ/K-covering of the parameter space $\mathcal{W} := \{\theta \in \mathbb{R}^p, ||\theta||_\infty \le W\}$ under ℓ_2-norm. We will show $\mathcal{F}_\epsilon := \{f_\theta : \theta \in \mathcal{W}_\epsilon\}$ is an ϵ-covering of $\mathcal{C}_i^p$.

Given $\mathcal{C}_i(\cdot,\theta) \in \mathcal{C}_i^p$, we can find $\theta' \in \mathcal{W}_\epsilon$ such that $||\theta-\theta'||_2 \le \epsilon/K$. By K-Lipschitz parametrization we can immediately get

$$||\mathcal{C}_i(\cdot,\theta) - f_{\theta'}||_\mathcal{F} \le K||\theta-\theta'||_2 \le \epsilon.$$

Because $f_{\theta'} \in \mathcal{F}_\epsilon$, $\mathcal{F}_\epsilon$ is an ϵ-covering of $\mathcal{C}_i^p$, which yields

$$\mathcal{N}(\mathcal{C}_i^p,\epsilon,||\cdot||_\mathcal{F}) \le |\mathcal{F}_\epsilon| = \mathcal{N}(\mathcal{W},\epsilon/K,||\cdot||_2).$$

By Lemma A.6, we have $\mathcal{N}(\mathcal{W},\epsilon/K,||\cdot||_2) \le (\frac{KW\sqrt{p}}{\epsilon})^p$, thus

$$\mathcal{N}(\mathcal{C}_i^p,\epsilon,||\cdot||_\mathcal{F}) \le (\frac{KW\sqrt{p}}{\epsilon})^p$$

. □

We show that an ϵ/K-covering of the parameter space is an ϵ-covering of $\mathcal{C}_i^p$. In this way, we successfully find an upper bound of the covering number of $\mathcal{C}_i^p$. Combining Lemma A.4 and Lemma A.5, we can derive the generalization gap of the clone model. Now we start our proof, we first focus on the space $\mathcal{C}_i^p, i \in [k]$. By Lemma A.5 we have $\forall i \in [k], \mathcal{N}(\mathcal{C}_i^p,\epsilon,||\cdot||_\mathcal{F}) \le (\frac{KW\sqrt{p}}{\epsilon})^p$. So

$$\begin{aligned}
&8\epsilon + \frac{4\sqrt{2}}{\sqrt{n}}\int_\epsilon^{2a}\sqrt{\ln\mathcal{N}(\mathcal{C}_i^p,u,||\cdot||_\mathcal{F})}\,du \\
\le&8\epsilon + \frac{4\sqrt{2}}{\sqrt{n}}\int_\epsilon^{2a}\sqrt{p\ln(KW\sqrt{p}/u)}\,du \\
\le&8\epsilon + a\frac{8\sqrt{2}}{\sqrt{n}}\sqrt{p\ln(KW\sqrt{p}/\epsilon)},
\end{aligned}$$

Denote by $L_{\mathcal{D}_{ood}}(\mathcal{C}_i) := \mathbb{E}_{\mathcal{D}_{ood}}\mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{x})}[l_i(\mathcal{C}_i(\boldsymbol{x},\boldsymbol{\theta}_C),\mathcal{V}_i(\boldsymbol{x}+\boldsymbol{w},\boldsymbol{\theta}_V))]$ and $L_S(\mathcal{C}_i) := \sum_{i=1}^n l_i(\mathcal{C}_i(\boldsymbol{x}_i,\boldsymbol{\theta}_C),\mathcal{V}_i(\boldsymbol{x}_i+\boldsymbol{w}_i,\boldsymbol{\theta}_V))$. Taking $\epsilon = \frac{a}{\sqrt{n}}$ and combining it with Lemma A.4, we have with probability $1-\delta$, for all $\mathcal{C}_i \in \mathcal{C}_i^p$,

$$\begin{aligned} L_{\mathcal{D}_{ood}}(\mathcal{C}_i) - L_S(\mathcal{C}_i) \leq \frac{8aC}{\sqrt{n}}(1+\sqrt{p\ln(K^2W^2pn/a^2)}) \\ + 4a\sqrt{\frac{2\ln(2/\delta)}{n}}. \end{aligned} \tag{28}$$

Given $\mathcal{C} = (\mathcal{C}_1,...,\mathcal{C}_k) \in \mathcal{C}^p$, we have $L_{\mathcal{D}_{ood}}(\mathcal{C}) = \sum_{i=1}^k L_{\mathcal{D}_{ood}}(\mathcal{C}_i)$ and $L_S(\mathcal{C}) = \sum_{i=1}^k L_S(\mathcal{C}_i)$. By applying the union bound over the function space $\mathcal{C}_i^p, i \in [k]$, we have with probability $1-k\delta$, for all $\mathcal{C} \in \mathcal{C}^p$

$$\begin{aligned} L_{\mathcal{D}_{ood}}(\mathcal{C}) - L_S(\mathcal{C}) &= \sum_{i=1}^k (L_{\mathcal{D}_{ood}}(\mathcal{C}_i) - L_S(\mathcal{C}_i)) \\ &\leq \frac{8akC}{\sqrt{n}}(1+\sqrt{p\ln(K^2W^2pn/a^2)}) + 4ak\sqrt{\frac{2\ln(2/\delta)}{n}}. \end{aligned} \tag{29}$$

Taking $\delta = \frac{\delta}{k}$ yields the result.

B. Preliminary Definition

In this section, we establish the definition of the Wasserstein space. We represent $\mathcal{P}_2(\mathbb{R}^d)$ as the space comprising probability measures on $\mathbb{R}^d$ with finite second-order moments. Each element $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ is a probability measure, characterized by its density function $\mu : \mathbb{R}^d \to \mathbb{R}$, relative to the Lebesgue measure dx. The Wasserstein distance between two probability measures μ_1 and μ_2 in $\mathcal{P}_2(\mathbb{R}^d)$ is defined as the following:

$$W_2(\mu_1,\mu_2) = \left(\min_{\omega\in\prod(\mu_1,\mu_2)} \int ||\boldsymbol{x}-\boldsymbol{x}'||^2 d\omega(\boldsymbol{x},\boldsymbol{x}')\right)^{1/2},$$

where $\prod(\mu_1,\mu_2) = \{\omega|\omega(A\times\mathbb{R}^d) = \mu_1(A), \omega(\mathbb{R}^d\times B) = \mu_2(B)\}$. ω represents the joint probability measure with marginal measures μ_1 and μ_2 respectively. Consequently, $\mathbb{W}^2 = (\mathcal{P}_2(\mathbb{R}^d), W_2)$ constitutes a metric space.

Definition B.1 (First Variation)**.** The first variation of a functional $F(\mu)$ at the probability measure μ is defined as:

$$\frac{\delta F}{\delta \mu}(\mu) = \lim_{\epsilon\to 0}\frac{F(\mu+\epsilon\psi)-F(\mu)}{\epsilon}, \tag{30}$$

where ψ could be chosen as an arbitrary function.

C. Derivations of Watermark Posterior Distribution Evolution

The Wasserstein gradient flow (WGF) is defined as the following:

$$\partial_t\mu_t = \nabla W_2 F(\mu_t) := \nabla\cdot\left(\mu_t\nabla\frac{\delta F}{\delta\mu}(\mu_t)\right), \tag{31}$$

According to (Ambrosio et al., 2008), the first variation of KL divergence, $\frac{\delta\mathbb{KL}(\mu||\pi)}{\delta\mu}$ is equal to the following:

$$\frac{\delta\mathbb{KL}(\mu||\pi)}{\delta\mu} = \log\frac{\mu}{\pi}+1; \tag{32}$$

In accordance with (Wibisono, 2018), we define the watermark posterior distribution as an energy function $q(\boldsymbol{w}|\boldsymbol{x}) \propto e^{-U}$

$$U(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{id}}\log P(y|\boldsymbol{x}+\boldsymbol{w}) + \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_{ood}}\mathbb{KL}(\mathcal{V}(\boldsymbol{x};\boldsymbol{\theta}_V)||\mathcal{V}(\boldsymbol{x}+\boldsymbol{w};\boldsymbol{\theta}_V)) \tag{33}$$

We transform the Wasserstein gradient $\nabla_{W_2} F(\mu_t)$ in Eq. (31) through the application of the transformation $\mathcal{K}_\mu \nabla_{W_2} F(\mu_t)$, employing the integral operator $\mathcal{K}_\mu f(\boldsymbol{w}) = \int K(\boldsymbol{w}, \boldsymbol{w}') f(\boldsymbol{w}') d\mu(\boldsymbol{w}')$, where $\mathcal{H}$ denotes the RKHS induced by the kernel K. Subsequently, we express the WGF in Eq. (31) in terms of the kernelized WGF (Liu, 2017) as the following:

$$\partial_t \mu_t = div(\mu_t \mathcal{K}_{\mu_t} \nabla \frac{\delta F}{\delta \mu}(\mu_t)). \tag{34}$$

The above Eq. (34) implies that the random variable W_t, representing the posterior distribution of the watermark $z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})$ and describing the evolved watermark $\boldsymbol{w}$ for the data point $\boldsymbol{x}$ at time t, follows according to the following differential equation (Liu, 2017; Chewi et al., 2020):

$$\frac{dW}{dt} = -[\mathcal{K}_\mu \nabla \frac{\delta F}{\delta \mu}(\mu_t)](W). \tag{35}$$

The kernelized Wasserstein gradient can be expressed by the following equation:

$$\mathcal{K}_{\mu_t} \nabla \frac{\delta \mathbb{KL}(\mu||\pi)}{\delta \mu} = \mathcal{K}_{\mu_t} \nabla \log \frac{\mu_t}{\pi}(\boldsymbol{w}) := \int K(\boldsymbol{w}, \cdot) \nabla \log \frac{\mu_t}{\pi} = \int K(\boldsymbol{w}, \cdot) \nabla U d\mu_t - \int \nabla_2 K(\boldsymbol{w}, \cdot) d\mu_t \tag{36}$$

In Eq. (36), we apply the technique of integration by parts in the third identity. Here, ∇_2 denotes the gradient of the kernel with respect to the second argument.

Substituting the expression from Eq. 36 into Eq. 35, we derive the following differential equation, which describes the continuous evolution of the watermark posterior distribution:

$$\frac{dW}{dt} = -\int K(\boldsymbol{w}, \cdot) \nabla U d\mu_t + \int \nabla_2 K(\boldsymbol{w}, \cdot) d\mu_t \tag{37}$$

We proceed to discretize the aforementioned equation in Eq. (37), treating each sampled watermark from the posterior distribution as an individual particle. This process leads us to the following expression for the Wasserstein gradient of the watermark posterior:

$$\nabla_{\boldsymbol{w}} \mathbb{KL}(z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})||q(\boldsymbol{w}|\boldsymbol{x})) = \mathbb{E}_{\boldsymbol{w} \sim z_{\boldsymbol{\Phi}}(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{\gamma})} [\underbrace{k(\boldsymbol{w}, \cdot) \nabla_{\boldsymbol{w}} U(\boldsymbol{w})}_{\text{smoothed gradient}} + \underbrace{\nabla_{\boldsymbol{w}} k(\boldsymbol{w}, \cdot)}_{\text{repulsive term}}] \tag{38}$$

D. OOD Data Generation

To ensure comprehensive coverage of OOD data, we treat the parameters of the pseudo-data generator, denoted as $\boldsymbol{\theta}_G$, as random variables. Specifically, for the i-th layer of the generator network, $\boldsymbol{\theta}_G^i$, we sample the parameters from a normal distribution, i.e., $\boldsymbol{\theta}_G^i \sim \mathcal{N}(0, I)$. The OOD data is generated as $\boldsymbol{x} = \mathcal{G}(\boldsymbol{\epsilon}; \boldsymbol{\theta}_G)$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$. The underlying rationale is that a fixed set of parameters can generate a specific data distribution, whereas introducing a distribution over the generator parameters can produce an infinite set of OOD data distributions, thereby aiding in the coverage of a broader range of OOD scenarios.

E. Model Extraction Method Description

E.1. Soft-Label DFME

In the soft-label setting (Truong et al., 2021; Kariyappa et al., 2021a), the black-box API offers softmax probabilities outputs corresponding to input queries. Employing a synthetic data generator $\boldsymbol{x} = G(\boldsymbol{\epsilon}, \boldsymbol{\theta}_G)$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, the attacker utilizes synthetic data $\boldsymbol{x}$ to query the black-box victim model, obtaining soft-labels $\boldsymbol{y}_V = \mathcal{V}(\boldsymbol{x}; \boldsymbol{\theta}_V)$ and acquiring outputs from the clone model $\boldsymbol{y}_C = \mathcal{C}(\boldsymbol{x}, \boldsymbol{\theta}_C)$. Attacker then minimizes the following loss function to train the clone model and data generator:

$$\mathcal{L}_C = \mathbb{KL}(\boldsymbol{y}_V, \boldsymbol{y}_C), \quad \mathcal{L}_G = -\mathbb{KL}(\boldsymbol{y}_V, \boldsymbol{y}_C) \tag{39}$$

E.2. Decision-based DFME (DFMS-HL)

DFMS-HL (Sanyal et al., 2022) employs 10 unrelated classes for pre-training the Generative Adversarial Network (GAN) (Goodfellow et al., 2014). Considering a random noise variable $z \sim \mathcal{N}(0, I)$, pseudo data is generated through $\boldsymbol{x} = G(\boldsymbol{z}; \boldsymbol{\theta}_G)$, where $\boldsymbol{z} \sim \mathcal{N}(0, \mathbf{I})$, and $\boldsymbol{\theta}_G$ represents the generator parameters. Subsequently, the generated pseudo data is utilized to query the victim model, with $y(\boldsymbol{x}) = \arg\max_i \mathcal{V}_i(\boldsymbol{x}; \boldsymbol{\theta}_V)$ denoting the predicted label by the victim model on the pseudo data. The clone model then minimizes the following loss function to learn its parameters:

$$\mathcal{L}(\mathcal{C}(\boldsymbol{x}, \boldsymbol{\theta}_C), y(\boldsymbol{x})) \tag{40}$$

Rather than training the pseudo-data generator from scratch, DFMS-HL learns the data generator from a set of unrelated classes of images using the following loss function, akin to the one used in GAN (Goodfellow et al., 2014):

$$\mathcal{L}_{real} = \mathbb{E}_{\boldsymbol{x} \sim P_{real}(\boldsymbol{x})}[\log \mathcal{D}(\boldsymbol{x})] \tag{41}$$

$$\mathcal{L}_{fake} = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0,I)}[\log(1 - \mathcal{D}(\boldsymbol{x}))] \tag{42}$$

where $\mathcal{D}(\boldsymbol{x})$ denotes the discriminator function. To enhance the diversity of the generated data, DFMS-HL introduces a diversity loss as follows:

$$\mathcal{L}_{div} = \sum_{j=0}^{j=K} \alpha_j \log \alpha_j; \tag{43}$$

$$\alpha_j = \frac{1}{N} \sum_{i=1}^{i=N} softmax(C(\boldsymbol{x}, \boldsymbol{\theta}_C)) \tag{44}$$

The pseudo-data generator and discriminator respectively optimize the following loss function:

$$\mathcal{L}_G = \mathcal{L}_{fake} + \lambda \mathcal{L}_{div} \tag{45}$$

$$\mathcal{L}_D = \mathcal{L}_{real} + \mathcal{L}_{fake} \tag{46}$$

E.3. Defense against Hard-Label DFME

We denote the ground-truth label for the query data $\boldsymbol{x}$ as $y(\boldsymbol{x})$. The predicted label by the victim model with parameters $\boldsymbol{\theta}_V$ is determined as $y(\boldsymbol{x}) = \arg\max_i \mathcal{V}_i(\boldsymbol{x}, \boldsymbol{\theta}_V)$. Active watermarking aims to maximize the change in model outputs on simulated out-of-distribution (OOD) data, significantly increasing the likelihood of incorrect prediction labels on this OOD data. Let the wrongly predicted label be denoted as $y'(\boldsymbol{x}) = \arg\max_i \mathcal{V}_i(\boldsymbol{x}, \boldsymbol{\theta}_V)$, where $y'(\boldsymbol{x}) \neq y(\boldsymbol{x})$. Consequently, minimizing the loss $\mathcal{L}(\mathcal{C}(\boldsymbol{x}, \boldsymbol{\theta}_C), y'(\boldsymbol{x}))$ leads to the attacker learning incorrect information from the victim model.

F. Additional Experimental Results

F.1. Model Extraction Defense Baselines

- **RandP** (Orekondy et al., 2020): This defense method involves the random perturbation of the victim model output. Specifically, we introduce random noise $\widehat{\boldsymbol{a}_V}$ to the prediction logits, where $\boldsymbol{a}_V$ represents the victim model output logit. The perturbed victim model output logit, denoted as $\widehat{\boldsymbol{a}_V}$, is obtained by adding the random noise vector $\boldsymbol{\lambda}_T$ to $\boldsymbol{a}_V$. To recover this perturbed logit from the probability outputs $\boldsymbol{y}$, we use the expression $\boldsymbol{a} = \log \frac{\boldsymbol{y}}{1-\boldsymbol{y}}$. The random noise vector is denoted as $\boldsymbol{\lambda}_T$. The logits undergo renormalization, resulting in $\widehat{\boldsymbol{y}} = \frac{1}{1+e^{-\widehat{\boldsymbol{a}}}}$, which is then presented to the users.

- **Prediction Poisoning** (Orekondy et al., 2020): They introduce an objective for perturbing the outputs of the victim model, termed Maximizing Angular Deviation (MAD). The aim of MAD is to modify the prediction probabilities $\boldsymbol{y}_V$ of the victim model in a way that generates an adversarial perturbed gradient with maximal deviation from the original gradient of the victim model.

- **GRAD** (Mazeika et al., 2022): This defense mechanism utilizes gradient redirection to enable the adversary to update the clone model gradient in any arbitrary direction.

- **MeCo** (Wang et al., 2023) is the SOTA model extraction defense method which employees the distributionally robust defensive training.

F.2. Data-Based Model Extraction Baselines

We employ Knockoff Nets (Orekondy et al., 2019) and Jacobian Based Dataset Augmentation (JBDA) (Papernot et al., 2017) to extract the target victim model, and the results are presented in Table 10.

- Knockoff Nets (Orekondy et al., 2019): This method extracts the target black-box model using a relevant surrogate dataset to query the target model. Subsequently, the attacker trains a clone model with the surrogate dataset and incorporates the target model predictions on the surrogate dataset as the corresponding data labels.

- Jacobian Based Dataset Augmentation (JBDA) (Papernot et al., 2017): In this approach, the attacker initially queries the target model with a small subset of in-distribution data examples to construct a labeled dataset. Subsequently, the attacker trains the clone model on the constructed labeled dataset. Additionally, the dataset is augmented with additional synthetic examples generated by perturbing the raw data input using the Jacobian of the loss function.

F.3. Modify the victim model to be GoogLeNet

Table 6. Clone model accuracy after applying different defense methods on **CIFAR10** against existing DFME methods with **GoogLeNet** as the target model

Attack	**Defense**	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended ↓	$61.38 \pm 1.93\%$	$60.23 \pm 1.82\%$	$58.37 \pm 2.19\%$
	RandP ↓	$61.06 \pm 2.18\%$	$59.82 \pm 1.87\%$	$58.03 \pm 2.31\%$
	P-poison ↓	$60.73 \pm 1.95\%$	$59.07 \pm 1.93\%$	$58.09 \pm 2.33\%$
	GRAD ↓	$59.41 \pm 1.90\%$	$58.72 \pm 1.91\%$	$58.18 \pm 1.98\%$
	MeCo ↓	$53.77 \pm 2.21\%$	$53.89 \pm 1.96\%$	$53.28 \pm 2.20\%$
	ACT(Ours) ↓	**51.32 ± 2.38%**	**52.51 ± 2.53%**	**51.07 ± 2.75%**
DFME	undefended ↓	$74.67 \pm 1.35\%$	$75.23 \pm 1.51\%$	$70.96 \pm 0.78\%$
	RandP ↓	$70.05 \pm 1.97\%$	$70.54 \pm 1.72\%$	$66.78 \pm 2.07\%$
	P-poison ↓	$67.32 \pm 1.89\%$	$69.28 \pm 1.82\%$	$64.90 \pm 2.01\%$
	GRAD ↓	$69.32 \pm 1.91\%$	$66.32 \pm 1.97\%$	$65.73 \pm 1.77\%$
	MeCo ↓	$54.53 \pm 2.17\%$	$50.28 \pm 2.32\%$	$59.42 \pm 2.51\%$
	ACT(Ours) ↓	**49.38 ± 2.83%**	**46.53 ± 3.06%**	**55.26 ± 3.16%**
MAZE	undefended ↓	$23.18 \pm 2.37\%$	$19.01 \pm 1.09\%$	$21.28 \pm 3.16\%$
	RandP ↓	$22.03 \pm 2.16\%$	$17.24 \pm 1.16\%$	$19.23 \pm 1.38\%$
	P-poison ↓	$21.28 \pm 2.33\%$	$17.28 \pm 1.22\%$	$17.76 \pm 1.16\%$
	GRAD ↓	$20.79 \pm 2.07\%$	$16.96 \pm 1.27\%$	$18.09 \pm 1.34\%$
	MeCo ↓	$19.65 \pm 2.61\%$	**14.16 ± 2.30%**	$18.31 \pm 2.75\%$
	ACT(Ours) ↓	**17.81 ± 2.93%**	$15.35 \pm 2.09\%$	**17.06 ± 2.18%**

F.4. DFME Defense on MNIST and MiniImageNet

We also assess the performance of the DFME defense on MNIST, as presented in Table 7.

Table 7. Accuracy of the clone model following the application of various defense methods on the **MNIST** dataset using LeNet5 as the target model.

Attack	Defense	Clone Model Architecture		
		LeNet5	LeNet5-Half	LeNet5-1/5
DFME	undefended $\downarrow$	$98.76 \pm 0.27\%$	$96.65 \pm 0.43\%$	$94.62 \pm 0.69\%$
	RandP $\downarrow$	$92.25 \pm 0.32\%$	$91.86 \pm 0.49\%$	$90.37 \pm 0.73\%$
	P-poison $\downarrow$	$88.34 \pm 0.78\%$	$86.09 \pm 0.96\%$	$84.98 \pm 1.07\%$
	GRAD $\downarrow$	$87.22 \pm 0.70\%$	$85.38 \pm 0.91\%$	$84.23 \pm 1.16\%$
	MeCo $\downarrow$	$85.07 \pm 0.87\%$	$82.93 \pm 1.27\%$	$82.57 \pm 1.53\%$
	ACT $\downarrow$	$\mathbf{81.67 \pm 0.96\%}$	$\mathbf{80.18 \pm 1.38\%}$	$\mathbf{80.09 \pm 1.76\%}$
MAZE	undefended $\downarrow$	$98.09 \pm 0.26\%$	$95.91 \pm 0.36\%$	$89.41 \pm 0.38\%$
	(RandP, **C**) $\downarrow$	$92.77 \pm 0.41\%$	$88.71 \pm 0.90\%$	$86.38 \pm 0.95\%$
	P-poison $\downarrow$	$85.34 \pm 0.78\%$	$87.05 \pm 0.86\%$	$86.18 \pm 0.97\%$
	GRAD $\downarrow$	$85.63 \pm 0.91\%$	$87.31 \pm 0.75\%$	$86.42 \pm 0.78\%$
	MeCo $\downarrow$	$88.57 \pm 0.87\%$	$86.99 \pm 0.78\%$	$85.03 \pm 0.68\%$
	ACT $\downarrow$	$\mathbf{83.56 \pm 0.93\%}$	$\mathbf{83.32 \pm 0.91\%}$	$\mathbf{82.53 \pm 0.82\%}$

Table 8. Accuracy of clone model following the application of various defense methods on **MiniImageNet**, using ResNet34-8x as the victim model.

Attack	Defense	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFMS-HL	undefended $\downarrow$	$46.72 \pm 4.86\%$	$40.35 \pm 4.97\%$	$38.71 \pm 3.85\%$
	RandP $\downarrow$	$45.09 \pm 4.93\%$	$39.51 \pm 4.83\%$	$38.08 \pm 3.95\%$
	P-poison $\downarrow$	$45.16 \pm 5.03\%$	$39.06 \pm 4.72\%$	$37.78 \pm 4.26\%$
	GRAD $\downarrow$	$45.32 \pm 5.21\%$	$39.17 \pm 4.85\%$	$37.85 \pm 4.32\%$
	MeCo $\downarrow$	$39.23 \pm 4.83\%$	$35.81 \pm 4.69\%$	$32.30 \pm 4.56\%$
	ACT(Ours) $\downarrow$	$\mathbf{33.28 \pm 4.57\%}$	$\mathbf{31.61 \pm 4.38\%}$	$\mathbf{27.32 \pm 4.75\%}$
DFME	undefended $\downarrow$	$35.89 \pm 3.97\%$	$28.71 \pm 3.25\%$	$25.05 \pm 3.68\%$
	RandP $\downarrow$	$30.76 \pm 4.09\%$	$22.06 \pm 3.83\%$	$20.23 \pm 3.97\%$
	P-poison $\downarrow$	$29.36 \pm 4.23\%$	$21.83 \pm 3.77\%$	$20.01 \pm 3.89\%$
	GRAD $\downarrow$	$29.87 \pm 3.76\%$	$21.65 \pm 3.75\%$	$19.82 \pm 3.77\%$
	MeCo $\downarrow$	$23.29 \pm 3.83\%$	$17.83 \pm 3.67\%$	$16.73 \pm 3.88\%$
	ACT(Ours) $\downarrow$	$\mathbf{19.81 \pm 3.57\%}$	$\mathbf{15.36 \pm 3.56\%}$	$\mathbf{12.95 \pm 3.28\%}$

F.5. Legitimate Use in Edge Case

Table 9. Evaluation of victim model's utility through test accuracy in edge cases on CINIC-10.

Method	CINIC-10
undefended(upper bound)	75.28%
RandP $\uparrow$	74.31%
P-poison $\uparrow$	74.17%
GRAD $\uparrow$	**74.69%**
MeCo $\uparrow$	73.72%
ACT(Ours) $\uparrow$	74.56%

F.6. Defense against DBME

Table 10. Clone model accuracy after applying defense methods against DBME on *CIFAR-10* comparing to EDM (Kariyappa et al., 2021b)

Dataset	KnockoffNets			JBDA		
	undefended	EDM	EDM + ACT	undefended	EDM	EDM + ACT
MNIST	90.18	51.34	**45.78**	88.48	79.04	**65.82**
CIFAR10	85.39	68.50	**60.21**	22.03	22.01	**21.65**
CIFAR100	53.04	41.16	**34.67**	4.09	**3.69**	3.75

F.7. Results of Adaptive Attack

Table 11. Accuracy of clone model following the application of the **adaptive attack** by attacker on *CIFAR-10*, using ResNet34-8x as the victim model.

Attack	Defense	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFME	undefended ↓	$87.36 \pm 0.78\%$	$75.23 \pm 1.53\%$	$73.89 \pm 1.29\%$
	ACT ↓	$46.57 \pm 2.83\%$	$40.32 \pm 2.96\%$	$49.25 \pm 2.67\%$
	Adaptive ↓	$\mathbf{23.97 \pm 1.65\%}$	$\mathbf{18.72 \pm 1.56\%}$	$\mathbf{21.08 \pm 1.95\%}$

Table 12. Accuracy of clone model when the attacker solely utilizes the **hard label** provided by the victim model, instead of output probabilities, on *CIFAR-10*.

Attack	Defense	Clone Model Architecture		
		ResNet18-8X	MobileNetV2	DenseNet121
DFME	undefended ↓	$87.36 \pm 0.78\%$	$75.23 \pm 1.53\%$	$73.89 \pm 1.29\%$
	ACT, soft label ↓	$46.57 \pm 2.83\%$	$40.32 \pm 2.96\%$	$49.25 \pm 2.67\%$
	ACT, hard label ↓	$\mathbf{40.09 \pm 1.08\%}$	$\mathbf{38.59 \pm 1.23\%}$	$\mathbf{39.82 \pm 1.97\%}$

F.8. Defending Self-Supervised Learning Model

To assess the effectiveness of our method against self-supervised learning pre-trained model, we perform model extraction and defense on the encoder trained with SimSiam (Chen & He, 2021) on ImageNet-1K. Following the procedure in (Dziedzic et al., 2022a), we evaluate the quality of the extracted encoder by various downstream task performance on CIFAR10, CIFAR100 and SVHN. The results are shown in Table 13. We can observe that under various defense strategies, our method substantially outperforms the SOTA defense method by up to 5.6%. More details can be found in the following.

Table 13. Downstream task performance by performing different defenses on self-supervised learning encoder on **ImageNet-1K**

Method	CIFAR10	CIFAR100	SVHN
undefended ↓	$71.2 \pm 0.8\%$	$48.6 \pm 0.5\%$	$73.6 \pm 0.6\%$
RandP ↓	$69.6 \pm 0.9\%$	$47.6 \pm 0.8\%$	$72.4 \pm 0.5\%$
P-poison ↓	$69.1 \pm 0.5\%$	$46.9 \pm 0.9\%$	$71.8 \pm 0.9\%$
GRAD ↓	$68.9 \pm 0.5\%$	$46.2 \pm 0.4\%$	$70.2 \pm 0.7\%$
MeCo ↓	$69.8 \pm 0.6\%$	$46.5 \pm 0.5\%$	$70.7 \pm 0.3\%$
ACT (Ours) ↓	$\mathbf{66.1 \pm 0.7\%}$	$\mathbf{40.6 \pm 0.6\%}$	$\mathbf{65.2 \pm 0.5\%}$

Following (Dziedzic et al., 2022a), we perform model extraction on the encoder which is pre-trained on ImageNet-1K dataset (Deng et al., 2009) with the self-supervised learning method, SimSiam (Chen & He, 2021).

To train a stolen model, we employ comparable hyperparameters to those used in training the victim models, including a batch size of either 64 or 256, an initial learning rate of 0.0001, and the Adam optimizer. In instances of stealing from the ImageNet victim model, we opt for a larger learning rate of 0.1 or 1.0 and a batch size ranging from 256. The query budget is set to be 50000.

In downstream tasks, we assess the performance of the stolen model against the victim model by evaluating their performance on CIFAR10, SVHN, STL-10, and Fashion MNIST datasets. The standard linear evaluation protocol is employed, involving the addition of an extra linear layer to the representation model while keeping all other layers frozen. Subsequently, the network is optimized using labeled data specific to the downstream task. For models stolen from a CIFAR10 or SVHN victim model, we utilize a learning rate of 0.0001 with the SGD optimizer during linear evaluation, adjusting the parameters based on the victim model. These tuned parameters remain constant when evaluating models stolen from the same victim model. In the case of the ImageNet victim model, a learning rate of 1.0 and a batch size of 256 are employed. Attacker uses the InfoNCE loss (Oord et al., 2018) to extract the encoder. In all instances, the top-1 test accuracy on the particular downstream task is reported and serves as the benchmark for comparing the performance between the victim and stolen models.

The energy function is defined as the following:

$$U(\boldsymbol{w}) = -\mathbb{E}_{\boldsymbol{x}\in\mathcal{D}_{id}}\left[\log\frac{\exp(sim(\boldsymbol{x},\boldsymbol{x}+\boldsymbol{w})/\tau)}{\sum_j \exp(sim(\boldsymbol{x},\boldsymbol{x}_j+\boldsymbol{w}_j)/\tau)}\right] + \mathbb{E}_{\boldsymbol{x}\in\mathcal{D}_{ood}}\left[\log\frac{\exp(sim(\boldsymbol{x},\boldsymbol{x}+\boldsymbol{w}')/\tau)}{\sum_j \exp(sim(\boldsymbol{x},\boldsymbol{x}_j+\boldsymbol{w}'_j)/\tau)}\right] \tag{47}$$

The notation $\boldsymbol{x}+\boldsymbol{w}$ represents a randomly watermarked data point on $\boldsymbol{x}$, while $\boldsymbol{x}_j+\boldsymbol{w}_j$ denotes other negatively watermarked instances. The function $sim(\boldsymbol{u},\boldsymbol{v}) = \frac{\boldsymbol{u}\cdot\boldsymbol{v}}{||\boldsymbol{u}||||\boldsymbol{v}||}$ signifies the normalized cosine similarity, and τ represents the temperature constant. The rationale behind optimizing this objective is to minimize the disparity between the feature representations of watermarked ID data and non-watermarked ID data, ensuring that benign users can effectively utilize the valuable representations generated by the victim model. Conversely, the second term in Equation (47) encourages maximizing the distance between the feature representation of watermarked OOD data and non-watermarked OOD data, aiming for an effective defense mechanism.

Subsequently, the parameters of the watermark posterior network are updated using the same methodology as in the supervised learning process outlined in Algorithm 1 in the main text.

F.9. Impact of query budget on clone model performance

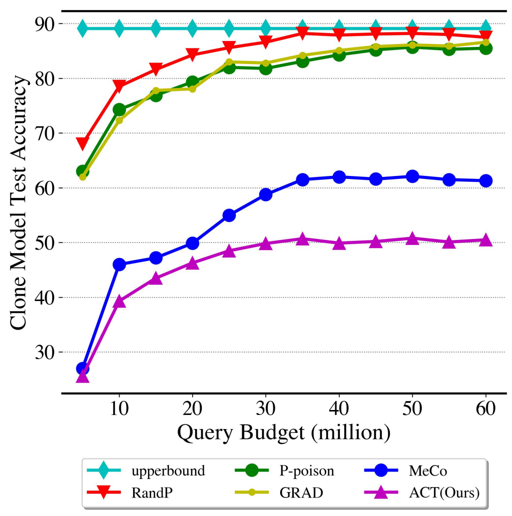

Figure 2. The test accuracy of the clone model on CIFAR10, where lower values are preferable, shows fluctuations with varying query budgets following the application of different defense methods to the victim model. Notably, our defense method, ACT, outperforms a range of other defense methods significantly.

F.10. Training Efficiency Evaluation

We perform training efficiency evaluation compared to the SOTA defense method, MeCo (Wang et al., 2023). We can observe that our method significantly reduces the training cost of MeCo by more than 87%.

Table 14. Training Time on CIFAR10

Algorithm	total training time (hours)
MeCo	5.68
ACT (Ours)	0.72

F.11. Test Time Memory Consumption Evaluation

To assess test-time memory efficiency in comparison to baseline defense methods, the results are presented in Table 15. It is evident that ACT exhibits lower memory usage than P-poison, as P-poison requires multiple forward and gradient computations during test time. Furthermore, ACT demonstrates significantly superior memory efficiency compared to

GRAD, given that GRAD involves memory-intensive double backpropagation. ACT maintains a comparable level of memory consumption to MeCo.

Table 15. Evaluation of memory efficiency during test time.

Algorithm	CIFAR10
P-poison	2.2 GB
GRAD	10.83 GB
MeCo	2.03 GB
ACT	2.03 GB

F.12. Effect of Fine-Tuning Victim Model

We assess performance through a comparison of fixed and fine-tuned victim models, with the results presented in Table 16. It is evident that fixing the victim model leads to a decrease in model utility. Therefore, fine-tuning the victim model becomes essential to preserve its utility.

Table 16. Model utility comparisons with fixed and updating the victim model.

Algorithm	CIFAR10	CIFAR100
fix victim model	$89.76 \pm 0.87\%$	$65.32 \pm 0.68\%$
update victim model	$\mathbf{94.25 \pm 0.75\%}$	$\mathbf{75.65 \pm 0.73\%}$

F.13. l_1 Norm Utility Evaluation

We additionally assess the l_1 norm output perturbation magnitude denoted by $||\boldsymbol{y} - \vec{\boldsymbol{y}}||_1$ (lower values are preferable), where $\boldsymbol{y}$ and $\vec{\boldsymbol{y}}$ represent the original and perturbed output probabilities, respectively. The outcomes are detailed in Table 17. The findings indicate that our approach provides enhanced defense with a reduced magnitude of output perturbation. This is advantageous for benign users as they may require meaningful output logits to analyze valuable knowledge within the model outputs.

Table 17. **victim model utility** measured by the l_1 norm of the model output probability with/without perturbation

Method	MNIST	CIFAR10	CIFAR100
RandP $\downarrow$	1.0	1.0	1.0
P-poison $\downarrow$	1.0	1.0	1.0
GRAD $\downarrow$	1.0	1.0	1.0
MeCo $\downarrow$	0.0126	0.099	**0.312**
ACT $\downarrow$	**0.0117**	**0.093**	0.328