BEOWULF: Mitigating Model Extraction Attacks Via Reshaping Decision Regions

Xueluan Gong
School of Computer Science,
Wuhan University,
Wuhan, China
xueluangong@whu.edu.cn

Rubin Wei
School of Cyber Science and
Engineering, Wuhan University,
Wuhan, China
weirubin@whu.edu.cn

Ziyao Wang
School of Cyber Science and
Engineering, Wuhan University,
Wuhan, China
wangziyao@whu.edu.cn

Yuchen Sun
School of Cyber Science and
Engineering, Wuhan University,
Wuhan, China
yuchensun@whu.edu.cn

Jiawen Peng
School of Cyber Science and
Engineering, Wuhan University,
Wuhan, China
nuyoah@whu.edu.cn

Yanjiao Chen*
College of Electrical Engineering,
Zhejiang University,
Hangzhou, China
chenyanjiao@zju.edu.cn

Qian Wang*
School of Cyber Science and
Engineering, Wuhan University,
Wuhan, China
qianwang@whu.edu.cn

ABSTRACT

Machine Learning as a Service (MLaaS) enables resource-constrained users to access well-trained models through a publicly accessible Application Programming Interface (API) on a pay-per-query basis. Nevertheless, model owners may face the potential threats of model extraction attacks where malicious users replicate valuable commercial models based on query results. Existing defenses against model extraction attacks, however, either sacrifice prediction accuracy or fail to thwart more advanced attacks. In this paper, we propose a novel model extraction defense, dubbed BEOWULF[1], which draws inspiration from theoretical findings that models with complex and narrow decision regions are difficult to be *reproduced*. Rather than arbitrarily altering decision regions, which may jeopardize the predictive capacity of the victim model, we introduce a *dummy* class, carefully synthesized using both random and adversarial noises. The random noise broadens the coverage of the dummy class, and the adversarial noise impacts decision regions near decision boundaries with normal classes. To further improve the model utility, we propose to employ data augmentation methods to seamlessly integrate the dummy class and the normal classes. Extensive evaluations on CIFAR-10, GTSRB, CIFAR-100, and ImageNette datasets demonstrate that BEOWULF can significantly reduce the extraction accuracy of 6 state-of-the-art model extraction attacks by as much as 80%. Moreover, we show that BEOWULF can contribute to a modest improvement in the adversarial robustness.

CCS CONCEPTS

• **Security and privacy → Software and application security**; • **Computing methodologies → Computer vision**.

KEYWORDS

Machine Learning as a Service; model extraction attacks, model extraction defenses; decision region reshaping

ACM Reference Format:
Xueluan Gong, Rubin Wei, Ziyao Wang, Yuchen Sun, Jiawen Peng, Yanjiao Chen, and Qian Wang*. 2024. BEOWULF: Mitigating Model Extraction Attacks Via Reshaping Decision Regions. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3658644.3670267

1 INTRODUCTION

Deep learning has attained state-of-the-art performance across various domains. However, these models are trained using vast amounts of sensitive data and often entail substantial computational expenses. Consequently, mainstream cloud providers (e.g., Amazon AWS[2], Microsoft Azure[3], Google Cloud[4], and BigML[5]) have launched Machine Learning as a Service (MLaaS) [51, 68], enabling clients to harness the power of advanced machine learning models through accessible prediction APIs. Users can submit

*Yanjiao Chen and Qian Wang are corresponding authors.

[1] In Anglo-Saxon literature and mythology, "Beowulf" is a heroic figure known for his strength and bravery, defending the kingdom against monsters in an epic tale.

[2] https://aws.amazon.com/cn/machine-learning/

[3] https://azure.microsoft.com/en-us/services/machine-learning/

[4] https://cloud.google.com/

[5] https://cloudacademy.com/blog/bigml-machine-learning/

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3670267

queries to the APIs and receive inference services, streamlining and accelerating the development process. Unfortunately, the significant economic values of such commercial models have incentivized attackers to perform model extraction attacks [12, 36].

In model extraction attacks, the adversary queries the target model with carefully crafted samples to maximally extract internal model information and then uses the returned query results to train a substitute model [45, 47]. The substitute model may be employed to generate future query samples to enhance the extraction effectiveness [11]. Through model extraction attacks, the adversary's gain is two-fold: (a) the adversary may commercialize the substitute model for prediction service [11]; (b) the adversary may construct transferable adversarial examples aiming to mislead the victim model into making erroneous predictions [48].

Substantial efforts have been dedicated to mitigating model extraction attacks, including abnormal query detection [25, 28], query result disruption [33, 46, 61], adversarial training [24], and model watermarking [23, 60]. Abnormal query detection methods assume that the distribution of malicious queries is statistically distinct from those of normal queries. However, this may be bypassed by an attacker who carefully crafts a query sequence that closely mimics the statistical distribution of benign queries. In this case, existing detection methods may have high false positive rates [44, 77]. Query result disruption strategies aim to thwart model extraction attempts by disrupting the query results. For example, Prediction Poisoning (PP) [46] maximized the angle deviation between the original and the perturbed gradient. AM [28] incorporates misleading information into the output of the query, thus preventing attackers from replicating the target model's decision boundary accurately. EDM [26] combats model extraction by constructing a diverse ensemble of models to make it challenging to reconstruct a model with similar functionality. MDP [70] employs a differential privacy mechanism to infuse noise into model responses. However, the introduction of misleading information may adversely affect benign users [28, 46]. Moreover, building and maintaining numerous models incur additional computational and storage overheads. Despite diversity, an ensemble of models may not consistently outperform a well-optimized single model [26]. Recent studies have also leveraged adversarial training to mitigate model extraction attacks, e.g., AMAO [24], but the performance is unsatisfactory under certain attacks. For example, the test accuracy of the extracted substitute model can be as high as 70% under protection. Model watermarking is used by owners to claim ownership of a commercial model after it is stolen, but it cannot prevent model extraction attacks. In addition, most of the existing defenses are ineffective against the latest model extraction attacks, such as D-DAE [4].

In this paper, we propose a novel defense strategy to thwart model extraction attacks by meticulously reshaping the decision regions of the victim model. To build a substitute model that mirrors the functionalities of the target victim model, the crux of model extraction attacks lies in replicating the decision boundaries, i.e., the borders that demarcate different classes within the victim model. Drawing insights from neural network training theories [35, 57] that demonstrate narrow decision regions shaped by fragmented and unstable decision boundaries are hard to *reproduce*[6], we propose to condense the decision regions and complicate the decision boundaries.

To strike a balance between model utility and defense robustness, we tackle the following challenges.

C1: How to restructure the model decision regions to realize the goal of model extraction defense?

Instead of haphazardly reshaping the decision boundaries, which risks compromising the predictive capability of the victim model, we propose to introduce a dummy class to condense the decision regions and enhance the decision boundary's complexity. We compose the dummy class with both random noises and adversarial noises. Random noises expand the diversity and coverage of the dummy class, while adversarial noises specifically focus on regions near the decision boundaries bordering different classes.

C2: How to maintain the utility of the fortified model?

Complex DNN models are known to have feature redundancy [1, 15, 73]. In classification models, a sizable portion of the feature space might not be essential to the primary classification task. Besides, the decision region for a class may encompass sizable redundant areas that are loosely related to the class. By judiciously condensing redundant areas of normal classes, we can maintain a high model prediction accuracy. To realize this objective, rather than directly merging the dummy class into the normal classes for retraining, we propose to employ mixup-based data augmentation methods to seamlessly integrate these datasets. In this way, the fine-tuned model features both high prediction accuracy and strong robustness to model extraction attacks.

We have conducted extensive experiments on CIFAR-10, GTSRB, CIFAR-100, and ImageNette to compare the performance of Beowulf with various model extraction defenses, i.e., Adaptive Misinformation (AM) [28], Ensemble of Diverse Models (EDM), Prediction Poisoning (PP) [46], and Model Orthogonalization (MO) [62], against six state-of-the-art model extraction attacks including KnockoffNets [59], JBDA [48], DFME [64], MAZE [27], DFMS [54], and D-DAE [4]. The results demonstrate that Beowulf significantly reduces the *Extraction Accuracy* (substitute model's accuracy) and *Fidelity* (similarity between the substitute and the victim model) of these attacks by up to 80%. To conclude, we make the following key contributions.

- We devise a novel model extraction defense method named Beowulf by narrowing decision regions and complicating decision boundaries of the victim model, making it harder for attackers to steal the model. We introduce a dummy class constructed with both random noise and adversarial noises to occupy broad decision regions, especially those near the decision boundaries with normal classes.
- We propose the mixup-based data augmentation technique to fuse the dummy class with normal classes for retraining the original model for protection, leading to a notable enhancement in the fortified model's utility.

[6] The meaning of "reproduce" is that given different random initializations, a model can replicate similar decision boundaries twice.

- Extensive experiments on 4 datasets against 6 state-of-the-art model extraction attacks demonstrate that Beowulf surpasses 4 defense baselines by yielding significantly lower extraction accuracy while preserving a high model utility.

2 PRELIMINARIES

2.1 Machine Learning as a Service

Developing and maintaining machine learning models, especially large models, require significant resources, e.g., a substantial amount of high-quality training data and extensive computational power, which are prohibitive for resource-limited users. To deal with this difficulty, many tech giants, e.g., Google and Amazon [51], have launched Machine Learning as a Service (MLaaS), enabling users to access machine learning models through APIs without having to manage underlying infrastructures. The popular pricing mechanism of MLaaS is charging users by queries. For instance, Google Prediction API charges $0.5 per 1,000 predictions[7] and Amazon Rekognition[8] charges $1 per 1,000 image queries for *basic label detection.* The booming large language models follow a similar business model. ChatGPT charges $0.002 per 1,000 tokens[9] and Google's *textbison* charges $0.001 per 1,000 characters[10].

The models are invaluable assets for MLaaS service providers. From the cost perspective, training OpenAI's GPT-3 model is estimated to cost more than $4 million, and processing user queries could have cost OpenAI $40 million per month [34, 66]. From the revenue perspective, it is projected that the number of active users per month for a model as popular as ChatGPT could reach 100 million [66], and the yearly bill for a small business may be as high as a quarter of a million [3].

Unfortunately, the great value of MLaaS models makes them attractive targets of model extraction/stealing attacks, where attackers can replicate a black-box victim model with comparable performance using only a few queries [11, 12, 37]. It is shown that it is feasible to steal crucial hyperparameters of the decoding algorithm of GPT-3 with only $4 [41]. This is not only an infringement on intellectual property but also causes significant economic loss to model owners.

2.2 Model Extraction Attacks

Model extraction attacks aim to construct a substitute model that closely approximates the functionalities of the target black-box victim model. Typically, the adversary can query the victim model to obtain confidence scores or class labels[11]. The adversary may or may not possess a subset of the original training dataset of the victim model but has access to public datasets with a similar distribution to the training samples.

A generic model extraction attack mainly includes three steps, i.e., surrogate model initiation, victim model query, and surrogate model retraining. In the surrogate model initiation step, the adversary may initialize the surrogate model as a pre-trained model or an empty model. In the victim model query step, the adversary carefully selects query samples and obtains query results from the victim model. In the surrogate model retraining step, the adversary retrains the surrogate model based on the query results. The steps of victim model query and surrogate model retraining are performed iteratively until the model converges or the query budget is reached. The query samples are essential for model extraction attacks. They may be natural samples or synthetic samples. Synthetic samples may be constructed based on natural samples or no samples (data-free). It is ideal to use a few query samples to extract the most information from the victim model.

Natural samples. To use natural samples for query, the key is sample selection. More specifically, instead of using all available natural samples for the query, the attacker selects the most informative and representative samples. For example, CopyCat [6] employs semi-supervised learning to sift through natural non-labeled samples. KnockoffNets [45] leverages reinforcement learning to select informative samples with a reward function updated by a hierarchical-structure learning strategy. ActiveThief [47] employs active learning for sample selection.

Data-assisted synthetic samples. With a few samples from the training dataset or public dataset, the attacker may synthesize more samples for the query. A common practice is to craft adversarial examples that are considered *hard* to learn. Adversarial examples usually lie close to decision boundaries, and their query results are more likely to carry more useful information about the victim model. Jacobian-based augmentation can be used to create adversarial examples [63] based on the Jacobian matrix of the surrogate model. Fast Sign Gradient Method (FSGM) is also used to create adversarial examples [48]. Jutti *et al.* [25] utilized FSGM and proposed to use cross-validation search to optimize hyperparameters during training. Yu et al. [72] employed C&W attack [2] and Feature Adversary Attack [52] to generate adversarial samples.

Data-free synthetic samples. The victim model may be trained on a rare dataset (e.g., Magnetic Resonance Imaging (MRI) samples for breast cancer detection), and the attacker may have no access to similar datasets. In this case, data-free model extraction attacks [13, 64] use generative models to synthesize query samples that maximize the prediction gap between the surrogate model and the victim model, which facilitates information learning of the surrogate model from the query results. MAZE [27] is the first data-free model extraction method, which employed KL-divergence as the loss function. DFME [64] is based on Generative Adversarial Networks (GAN) [13], using l_1 norm loss to prevent the vanishing gradient problem. Similarly, DFMS [54] also adopts GAN and utilizes the gradient of the *clone network* as an approximation of the victim model's gradient.

2.3 Model Extraction Defenses

As model extraction attacks pose great threats to commercial MLaaS, many defense strategies have been designed to thwart such attacks. Existing defenses can be broadly divided into four categories, i.e., abnormal query detection, query result disruption, model watermarking, and adversarial training. Abnormal query detection methods aim to discover malicious query sequences suspicious of model extraction. Query result disruption methods deliberately add noises

[7] https://www.cleveroad.com/blog/mlaas--machine-learning-as-a-service-solutions-that-promise-to-change-the-way-you-code/
[8] https://aws.amazon.com/cn/rekognition/pricing/?did=ap_card&trk=ap_card
[9] https://openai.com/pricing
[10] https://cloud.google.com/vertex-ai/pricing
[11] In this paper, we mainly consider classification models [42, 43].

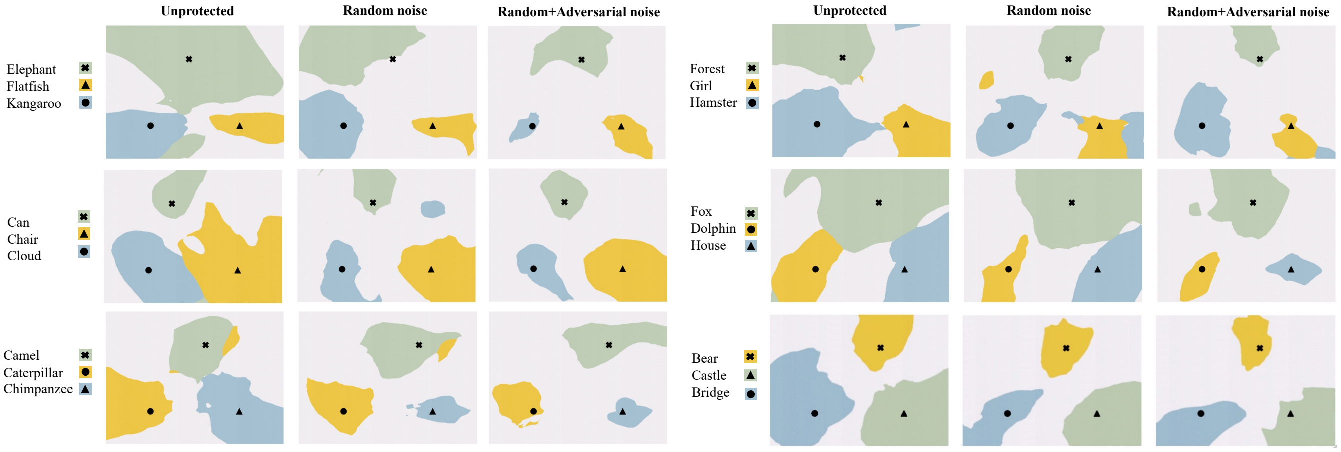

Figure 1: Comparative visualization of model decision regions: No Defense vs. Defense with random noise vs. Defense with combined random and adversarial noise as dummy class.

to query results to mislead the training of surrogate models. Model watermarking methods intend to assist model owners in ex-post identification of stolen models. Adversarial training strengthens the target models by including adversarial examples in the training process.

Abnormal query detection. Since query samples are an important design factor in model extraction attacks, the defenders may examine the statistical characteristics of a query sequence for anomaly detection. If a query sequence is deemed malicious, the defender may choose to disrupt the query result or deny service to the malicious user. For instance, Juuti et al. [25] proposed to detect malicious queries by evaluating the L_2 distance between successive queries. Kariyappa et al. [28] treated malicious queries as out-of-distribution (OOD) samples and employed OOD detection methods to identify them. As an extension, Ensemble of Diverse Models (EDM) [26] used ensemble learning to recognize OOD queries and disrupted the query results. However, abnormal query detection methods may be bypassed by attackers who conceal malicious queries in a seemingly normal query sequence [44, 77].

Query result disruption. Since query results are used to train the surrogate model, the defender may intentionally perturb the query results of all or abnormal queries. Lee et al. [33] attached a Reverse Sigmoid (RS) layer to the victim model with random softmax probabilities that are challenging to reverse. Prediction Poisoning [46] maximized the angle deviation between the original and perturbed gradients. Differential privacy (DP) is commonly used to counter model extraction attacks [70, 79]. Zheng et al. [79] proposed boundary differential privacy (ϵ-BDP), which perturbs predictions near the decision boundary. Yan et al. [70] introduced a monitoring-based differential privacy mechanism that dynamically adjusts the added noise based on the estimated model extraction status. Zhang et al. [77] ensured that query results from a specific category are similar, effectively reducing information leakage. Query result disruption methods may adversely affect the query results of benign users [9, 77, 79]. MLaaS service providers may be reluctant to adopt such methods for fear of degrading their business.

Model watermarking. Model watermarking is used to confirm ownership of an extracted model. Szyller et al. [60] proposed DAWN (Dynamic Adversarial Watermarking of Neural Networks) that dynamically changes the responses for a small subset of queries through a hash function and uses this dataset as a watermark to verify the ownership of the model. Jia et al. [23] proposed Entangled Watermarking Embeddings (EWE) that embedded watermarks into the victim model to be inherited by the stolen surrogate model. Unfortunately, model watermarking methods can only recover some losses of model stealing but cannot prevent model stealing from happening.

Adversarial training. Apart from the above methods, AMAO [24] proposed to use adversarial training to shield victim models from model extraction attacks. AMAO incorporates adversarial samples into the training dataset, and the adversarial samples are generated using methods such as the Fast Gradient Sign Method (FGSM) and Project Gradient Descent (PGD). To further enhance defense performance, AMAO also utilizes malicious query detection, adaptive query responses, and ownership verification strategies. However, AMAO is mainly effective for hard-label attacks, while soft-label KnockoffNets attack still achieves high test accuracy of 70.04% and 57.04% for CIFAR-10 and ImageNette datasets, respectively. In contrast, Beowulf can reduce the test accuracy of KnockoffNets to below 40% for CIFAR-10 and ImageNette.

Note that the goal and strategy of Beowulf differ from adversarial training. Adversarial training aims to bolster model robustness against adversarial examples, while Beowulf targets protecting model privacy against model extraction attacks. Adversarial training usually adds adversarial samples as training samples, while Beowulf introduces samples of a dummy class formed with both random and adversarial noises as training samples. Beowulf makes the trained model *difficult to steal* rather than more robust to adversarial examples.

2.4 Design Rationale of Beowulf

In this paper, we propose Beowulf, an effective model extraction defense strategy tailored to counter model extraction attacks by meticulously reshaping the victim model's decision boundaries. Our design rationale is based on two observations.

(1) **Complicated & compact decision regions**. The main design rationale of Beowulf is to alter the decision boundaries of the victim model to make it more difficult to extract. Our inspiration is that models with more complicated and compact decision regions (i.e., areas formed by decision boundaries) are harder to steal. Existing theoretical works on the geometry of class regions [35, 57] have discovered that narrow decision regions shaped by fragmented and unstable decision boundaries are hard to *reproduce*.

(2) **Redundant decision regions**. To complicate and narrow the decision regions of normal classes of the victim model without affecting the primary task, we propose to introduce a dummy class to model training. This is feasible due to feature redundancy observed in complex DNN models [1, 15, 73]. More specifically, the decision region of a class may contain large redundant areas that are not strongly class-related. By occupying redundant areas of normal classes, the dummy class has little influence on the performance of the primary classification task.

We materialize the above design rationale in Beowulf, as shown in Figure 1. In each subfigure, we illustrate the decision regions of 3 randomly selected classes of a CIFAR-100-trained model [10]. It is demonstrated that the original decision regions of normal classes are broad with relatively smooth decision boundaries (the *unprotected* column). We first create a dummy class with samples consisting of merely random noises. It can be viewed that the decision regions of normal classes shrink and become more irregular (the *random noise* column). We further add adversarial noises to samples of the dummy class. As a result, the dummy class further squeezes the decision regions of normal classes (the *random+adversarial noise* column). In this way, Beowulf alters the decision boundaries of normal classes of the primary task, making it difficult for attackers to extract these useful decision boundaries.

3 SYSTEM MODEL

We consider both the attacker and the defender in our system model.

Attacker. The attacker is an MLaaS user who aims to extract a well-performed surrogate model at a low query budget.

- *Knowledge & capability.* The attack can query the victim model, e.g., via API, and obtain the query results. We assume that the query results contain confidence scores, i.e., soft-label attacks by default[12], which are available in most commercial MLaaS. The attacker can access public datasets that are in-distribution or out-of-distribution of the original training dataset of the victim model. For example, if the victim model is trained with MNIST [32] for optical character recognition tasks, EMNIST [5] can be regarded as an in-distribution query dataset, and ImageNet [8] an out-of-distribution one. The attacker is aware that certain defenses may be adopted by the defender and can craft adaptive attacks to try to bypass the defense. To enhance the model extraction attack, we also consider that the attacker knows the model structure of the target model. Various studies have shown that when the substitute model structure is identical to or in the same family (e.g., VGG-19 for VGG-16) as that of the victim model, the substituted model will be more similar to the victim model [11].
- *Limitations.* Following state-of-the-art model extraction attacks [11, 45, 47], the attacker does not have access to the original training samples of the victim model.

Defender. The defender is an MLaaS service provider who aims to offer high-quality services to benign users but prevents malicious attackers from extracting a well-performed surrogate model.

- *Knowledge & capability.* The defender has white-box access to the victim model and is able to retrain the victim model.
- *Limitations.* The defender does not know the exact model extraction strategy adopted by the attacker.

4 BEOWULF: DETAILED CONSTRUCTION

Beowulf realizes the defense objective through two major phases, as shown in Figure 2.

- *Generating dummy class.* The dummy class performs the role of altering decision regions of normal classes. To ensure that the dummy class samples are non-overlapping with normal class samples, we construct dummy class samples with solely noises. We adopt two kinds of noises, i.e., random noises from a Gaussian distribution and adversarial noises generated using Projected Gradient Descent (PGD) [40]. Random noises diversify the coverage of the dummy class. Adversarial noises affect decision regions near the decision boundary between the dummy class and normal classes.
- *Fortifying model retraining.* We retrain the victim model with both dummy class data and normal class data. Rather than directly merging the samples from the dummy class and normal classes, we adopt CutMix (a data augmentation method) for seamless integration. It helps preserve the model utility and improve the defense performance.

4.1 Generating Dummy Class

Rather than introducing a foreign class from an auxiliary dataset, we propose to synthesize noises to construct the dummy class. The goal is to maximize the distinction between samples from normal classes and the dummy class, minimizing overlap in the feature space and thereby reducing the impact on the original classification task. Additionally, utilizing noises helps reinforce the decision boundary between clean samples and various transformations, enhancing robustness to out-of-distribution (OOD) images, such as those transformed with noise filters or rotations.

To generate dummy class samples, we construct two kinds of noises, i.e., random noises and adversarial noises. Random noise ensures the diversity of the samples of the dummy class. In this way, the dummy class covers a broad region of the feature space and

[12] Unlike hard-label attacks that only obtain the predicted label of the query, soft-label attacks can access the probability vector of the query, providing more information to attackers.

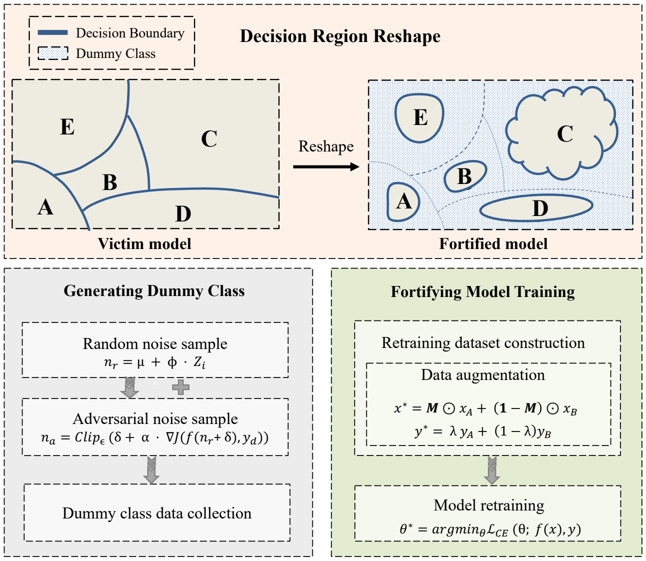
Decision Boundary
Dummy Class
Decision Region Reshape
E
C
B
A
D
Reshape
E
C
B
A
D
Victim model
Fortified model
Generating Dummy Class
Random noise sample
$n_r = \mu + \phi \cdot Z_i$
Adversarial noise sample
$n_a = Clip_\epsilon(\delta + \alpha \cdot \nabla J(f(n_r + \delta), y_d))$
Dummy class data collection
Fortifying Model Training
Retraining dataset construction
Data augmentation
$x^* = \boldsymbol{M} \odot x_A + (\boldsymbol{1} - \boldsymbol{M}) \odot x_B$
$y^* = \lambda y_A + (1 - \lambda) y_B$
Model retraining
$\theta^* = argmin_\theta \mathcal{L}_{CE}(\theta; f(x), y)$

Figure 2: Overview of Beowulf.

occupies redundant space of normal classes. Nevertheless, only the random noise cannot make the decision regions of normal classes adequately complex and narrow. To address this limitation, we further generate a batch of adversarial noises based on random noises. The detailed process of generating these two kinds of noise is as follows.

Random noise generation. We generate the random noise following a Gaussian distribution with a mean of 0 and a standard deviation ϕ as

$$n_r = \mu + \phi \cdot Z_i, \tag{1}$$

where n_r represents the generated random noise sample, and μ is the mean of the normal distribution, typically set to 0. ϕ is the standard deviation of the normal distribution, which controls the amplitude of the noise. Z_i is a randomly selected sample from the standard normal distribution, which can be generated using a random number generator.

Note that we choose Gaussian distribution since the training data also undergoes similar preprocessing of subtracting the mean and dividing by the standard deviation, i.e., normalization [50]. It can help prevent certain features from dominating the training process with exceptionally large magnitudes.

Adversarial noise generation. After generating random noise, we further compress the region of normal classes by introducing adversarial noises. We generate the adversarial noise based on the previously generated random noises. We observed that adversarial noises generated by targeted attacks tend to impact the model utility more than untargeted attacks. Moreover, generating adversarial noises for untargeted attacks is less costly. Thus, we use untargeted attacks to generate adversarial noises.

Specifically, we employ the loss-maximization process of Projected Gradient Descent (PGD) attack on the random noises as

$$n_a = Clip_\epsilon(\delta + \alpha \cdot \nabla J(f(n_r + \delta), y_d)), \tag{2}$$

where f represents the victim model, n_a denotes the adversarial noise generated by clipping the perturbation δ, and n_r represents the randomly generated noise sample. The label y_d corresponds to the dummy class. The function $Clip_\epsilon$ constrains the adversarial noise within a predetermined range ϵ, typically implemented through truncation. The parameter α signifies the step size. ∇J represents the gradient of the adversarial noise that maximizes the loss of $n_r + \delta$ on the dummy class. This loss-maximization process aims to generate the adversarial example from n_r that can cross the decision boundary between the dummy class and the normal classes but is still labeled as the dummy class in the model training process. Thus, the dummy class will compress the region of the normal classes by these adversarial examples and significantly reduce the availability of malicious query samples near this decision region. However, there is a trade-off between model performance and the strength of these adversarial noises. Therefore, it is essential to employ a proper method to balance the adversarial noise and the random noise, preserving the utility of the victim model.

Algorithm 1: The fortified model training process.

```
Input: Victim model $f(\cdot)$ and its parameters $\theta$, benign
       dataset $D = \{(x^{(t)}, y^{(t)})\}_{t=1}^{m}$, mean $\mu$ and standard
       deviation $\phi$ for random noise, PGD step size $\alpha$,
       mixing parameters $\beta_r$ and $\beta_a$, CutMix mask $\mathbf{M}$,
       dummy class $y_d$
Result: Fortified model $f(\cdot)$ and its parameters $\theta^*$
# Generate dummy class data
$n_r = \mu + \phi \cdot Z_i$;
$n_a = Clip_\epsilon(\delta + \alpha \cdot \nabla J(f(n_r + \delta), y_d))$;
# Form the dummy class dataset
$N \longleftarrow (\beta_r n_r + \beta_a n_a, y_d)$;
# CutMix data augmentation
for $(x_A, y_A) \in N$ and $(x_B, y_B) \in D$ do
    $x^* = \mathbf{M} \odot x_A + (\mathbf{1} - \mathbf{M}) \odot x_B$;
    $y^* = \lambda y_A + (1 - \lambda) y_B$;
    $D_{train} \longleftarrow (x^*, y^*)$
end
# Model training
$\theta^* = \arg\min_{\theta} \mathcal{L}_{CE}(\theta; f(x^*), y^*),\ (x, y) \in D_{train}$;
```

Random and adversarial noise combination. To keep both the victim model utility and ensure the defense capability, we combine the random noise and the adversarial noise as

$$N = \beta_r n_r + \beta_a n_a. \tag{3}$$

With a smaller random noise, the overall coverage of the dummy class will shrink. With a smaller adversarial noise, the dummy class cannot effectively occupy the regions near the decision boundaries of normal classes. Through empirical experiments, we have found that a $1:1$ mixing ratio has a satisfactory performance, i.e., $\beta_r = \beta_a = 0.5$.

Note that the number of dummy class samples we generate is the same as that of the original training dataset to effectively occupy most of the redundant decision regions. We assign a new label to the dummy class. For example, in the CIFAR-10 dataset, we can assign it as label 10.

Table 1: Comparison of Beowulf with baselines against 6 state-of-the-art model extraction attacks in CIFAR-10 and GTSRB.

Dataset	Defenses	Metrics[†]	KnockoffNets [59]	DFME [64]	JBDA [48]	MAZE [27]	D-DAE [4]	DFMS [54]
CIFAR-10	Unprotected	ACC (↑)	94.63%	94.63%	94.63%	94.63%	94.63%	94.63%
		EACC (↓)	79.99%	82.18%	51.47%	81.85%	71.86%	84.67%
		Fidelity (↓)	84.53%	90.12%	56.44%	89.76%	78.81%	92.58%
	AM [28]	ACC (↑)	85.24%	85.24%	85.24%	85.24%	85.24%	85.24%
		EACC (↓)	65.80%	**11.99%**	**6.16%**	32.03%	86.30%	75.36%
		Fidelity (↓)	77.19%	**12.58%**	**7.22%**	37.57%	99.24%	88.40%
	EDM [26]	ACC (↑)	86.95%	86.95%	86.95%	86.95%	86.95%	86.95%
		EACC (↓)	71.56%	36.93%	21.61%	34.45%	74.98%	38.95%
		Fidelity (↓)	82.30%	42.47%	24.85%	39.62%	86.23%	44.79%
	PP [46]	ACC (↑)	91.20%	91.20%	91.20%	91.20%	91.20%	91.20%
		EACC (↓)	52.96%	59.67%	65.42%	36.32%	83.20%	72.94%
		Fidelity (↓)	58.07%	51.62%	12.34%	39.82%	91.23%	79.97%
	Mo [62]	ACC (↑)	91.52%	91.52%	91.52%	91.52%	91.52%	91.52%
		EACC (↓)	60.20%	70.26%	49.22%	51.87%	45.30%	55.76%
		Fidelity (↓)	65.77%	76.77%	53.78%	56.67%	49.50%	60.92%
	Beowulf	ACC (↑)	**91.76%**	**91.76%**	**91.76%**	**91.76%**	**91.76%**	**91.76%**
		EACC (↓)	**39.27%**	30.26%	9.02%	**27.42%**	**34.05%**	**10.04%**
		Fidelity (↓)	**41.57%**	32.03%	9.80%	**29.12%**	**36.10%**	**11.55%**
Dataset	Defenses	Metrics[†]	KnockoffNets [59]	DFME [64]	JBDA [48]	MAZE [27]	D-DAE [4]	DFMS [54]
GTSRB	Unprotected	ACC (↑)	94.22%	94.22%	94.22%	94.22%	94.22%	94.22%
		EACC (↓)	92.40%	88.64%	26.80%	94.20%	92.16%	91.61%
		Fidelity (↓)	94.07%	90.25%	27.29%	95.91%	93.83%	93.27%
	AM [28]	ACC (↑)	89.80%	89.80%	89.80%	89.80%	89.80%	89.80%
		EACC (↓)	51.53%	39.73%	7.80%	44.16%	83.65%	45.41%
		Fidelity (↓)	57.38%	44.24%	8.69%	49.12%	93.15%	50.56%
	EDM [26]	ACC (↑)	91.65%	91.65%	91.65%	91.65%	91.65%	91.65%
		EACC (↓)	47.35%	74.47%	10.80%	68.52%	53.69%	**11.70%**
		Fidelity (↓)	51.66%	81.25%	11.78%	74.76%	58.58%	**12.77%**
	PP [46]	ACC (↑)	94.40%	94.40%	94.40%	94.40%	94.40%	94.40%
		EACC (↓)	79.24%	59.32%	18.90%	58.19%	92.16%	81.64%
		Fidelity (↓)	81.35%	60.90%	19.40%	59.74%	94.62%	83.81%
	Mo [62]	ACC (↑)	**91.88%**	**91.88%**	**91.88%**	**91.88%**	**91.88%**	**91.88%**
		EACC (↓)	64.96%	44.09%	**7.10%**	64.72%	55.75%	83.09%
		Fidelity (↓)	70.70%	47.99%	**7.73%**	70.44%	60.68%	90.43%
	Beowulf	ACC (↑)	91.87%	91.87%	91.87%	91.87%	91.87%	91.87%
		EACC (↓)	**38.46%**	**36.42%**	9.60%	**33.20%**	**26.78%**	29.67%
		Fidelity (↓)	**41.86%**	**39.64%**	10.44%	**36.13%**	**29.15%**	32.29%

[†] (↑) signifies that a higher value is preferable, while (↓) indicates that a lower value is more desirable.

4.2 Fortifying Model Retraining

After generating dummy class samples, we meticulously integrate them into the retraining dataset to fine-tune the model to enhance its robustness to model extraction attacks.

Retraining dataset construction. To create the retraining dataset, we need to merge the normal classes with the dummy class. We have discovered that directly combining samples from the dummy class and normal classes significantly impairs the model's prediction accuracy.

Inspired by Mixup [76] that blends two images according to their features and labels, we propose to use a mixup-based data augmentation technique. Recent studies affirm that data augmentation methods can be employed to enhance the decision boundary between clean samples and arbitrary transformations, effectively fortifying the model's performance against out-of-distribution images [71]. Moreover, data augmentation can enhance the model's resilience to adversarial examples [49].

Through extensive experiments, we adopt CutMix [74] to merge samples from normal classes and the dummy class. Specifically, for each sample (x_A, y_A) in normal classes, we randomly select a noise sample (x_B, y_B) from the dummy class. These two samples are then combined to form a new sample (x^*, y^*) as

$$\begin{aligned} x^* &= \mathbf{M} \odot x_A + (\mathbf{1} - \mathbf{M}) \odot x_B, \\ y^* &= \lambda y_A + (1 - \lambda) y_B, \end{aligned} \tag{4}$$

where the binary mask $\mathbf{M} \in \{0, 1\}^{W \times H}$ indicates where to drop out and fill in from the two samples. $\odot$ is the element-wise multiplication. CutMix replaces the deleted regions with a patch of another sample rather than simply deleting pixels of the original image. The corresponding labels are also blended proportionally.

Note that we also explore the effectiveness of other data augmentation methods in the experiments, including MixUp [76], FMix [16], and SaliencyMix [65].

Model retraining. We retrain the victim model with a standard cross-entropy loss $\mathcal{L}_{CE}$ on the retraining dataset. The model parameters are calculated by minimizing the loss function $\mathcal{L}$ by stochastic gradient descent (SGD) as

$$\theta^* = \arg\min_{\theta} \mathcal{L}_{CE}(\theta; f(x), y), \tag{5}$$

where θ is the parameters of the victim model, θ^* represents the protected parameters after retraining, $f(\cdot)$ is the protected model, and (x, y) is the retraining sample drawn from the data distribution

Table 2: Comparison of Beowulf with baselines against 6 state-of-the-art model extraction attacks in CIFAR-100 and ImageNette datasets.

Dataset	Defenses	Metrics[†]	KnockoffNets [59]	DFME [64]	JBDA [48]	MAZE [27]	D-DAE [4]	DFMS [54]
CIFAR-100	Unprotected	ACC (↑)	77.79%	77.79%	77.79%	77.79%	77.79%	77.79%
		EACC (↓)	43.49%	54.62%	1.00%	50.19%	49.60%	56.87%
		Fidelity (↓)	55.91%	70.21%	1.28%	64.52%	63.79%	73.11%
	AM [28]	ACC (↑)	57.56%	57.56%	57.56%	57.56%	57.56%	57.56%
		EACC (↓)	37.50%	**10.81%**	2.10%	20.07%	32.18%	38.82%
		Fidelity (↓)	65.14%	**18.78%**	3.64%	34.86%	55.90%	67.44%
	EDM [26]	ACC (↑)	56.96%	56.96%	56.96%	56.96%	56.96%	56.96%
		EACC (↓)	36.52%	18.74%	1.06%	35.91%	39.08%	**25.67%**
		Fidelity (↓)	64.12%	32.90%	1.86%	63.04%	68.61%	42.81%
	PP [46]	ACC (↑)	67.23%	67.23%	67.23%	67.23%	67.23%	67.23%
		EACC (↓)	39.60%	17.78%	3.60%	26.67%	42.30%	48.01%
		Fidelity (↓)	58.90%	26.44%	5.35%	39.66%	62.91%	71.41%
	Mo [62]	ACC (↑)	73.26%	73.26%	73.26%	73.26%	73.26%	73.26%
		EACC (↓)	49.21%	41.68%	1.21%	43.87%	35.66%	54.78%
		Fidelity (↓)	67.17%	56.90%	1.65%	59.88%	48.68%	74.78%
	Beowulf	ACC (↑)	**77.80%**	**77.80%**	**77.80%**	**77.80%**	**77.80%**	**77.80%**
		EACC (↓)	**24.99%**	23.83%	**1.67%**	**18.94%**	**20.49%**	30.27%
		Fidelity (↓)	**32.12%**	30.63%	**2.11%**	**24.34%**	**26.66%**	**30.91%**
Dataset	Defenses	Metrics[†]	KnockoffNets [59]	DFME [64]	JBDA [48]	MAZE [27]	D-DAE [4]	DFMS [54]
ImageNette	Unprotected	ACC (↑)	89.06%	89.06%	89.06%	89.06%	89.06%	89.06%
		EACC (↓)	78.58%	51.02%	16.80%	49.89%	83.39%	68.28%
		Fidelity (↓)	85.94%	57.29%	18.37%	56.02%	91.20%	74.68%
	AM [28]	ACC (↑)	85.45%	85.45%	85.45%	85.45%	85.45%	85.45%
		EACC (↓)	68.80%	31.92%	12.60%	42.56%	81.56%	29.87%
		Fidelity (↓)	80.51%	37.36%	14.75%	49.81%	97.81%	34.97%
	EDM [26]	ACC (↑)	87.94%	87.94%	87.94%	87.94%	87.94%	87.94%
		EACC (↓)	69.72%	42.66%	11.50%	40.51%	32.50%	44.10%
		Fidelity (↓)	79.28%	48.51%	13.08%	46.06%	36.96%	50.14%
	PP [46]	ACC (↑)	90.43%	90.43%	90.43%	90.43%	90.43%	90.43%
		EACC (↓)	74.14%	55.51%	12.10%	44.89%	82.96%	14.93%
		Fidelity (↓)	76.88%	57.56%	12.54%	46.55%	86.03%	15.48%
	Mo [62]	ACC (↑)	89.76%	89.76%	89.76%	89.76%	89.76%	89.76%
		EACC (↓)	70.08%	27.66%	13.40%	45.12%	26.10%	70.96%
		Fidelity (↓)	78.07%	30.82%	14.93%	50.27%	29.08%	79.06%
	Beowulf	ACC (↑)	**91.43%**	**91.43%**	**91.43%**	**91.43%**	**91.43%**	**91.43%**
		EACC (↓)	**35.90%**	**24.08%**	**10.60%**	**25.00%**	**13.20%**	**22.51%**
		Fidelity (↓)	**39.26%**	**26.40%**	**11.62%**	**27.40%**	**14.47%**	**24.62%**

[†] (↑) signifies that a higher value is preferable, while (↓) indicates that a lower value is more desirable.

Table 3: Impact of target model structure.

Dataset	VGG-19-Unprotected			VGG-19-Beowulf			DenseNet-Unprotected			DenseNet-Beowulf			WideResnet-Unprotected			WideResnet-Beowulf		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
CIFAR-10	92.10%	71.83%	77.99%	90.22%	28.79%	31.91%	89.28%	76.54%	85.73%	86.62%	21.26%	24.83%	96.21%	88.69%	92.18%	94.59%	36.13%	36.88%
CIFAR-100	74.94%	47.21%	62.99%	71.53%	12.53%	18.02%	76.19%	44.91%	58.94%	73.16%	20.89%	29.77%	78.07%	50.97%	65.28%	75.45%	23.81%	33.79%
GTSRB	93.12%	89.31%	95.91%	93.60%	32.57%	34.79%	92.56%	90.61%	97.88%	91.68%	28.57%	31.16%	91.49%	85.37%	93.31%	89.93%	27.68%	30.77%
ImageNette	92.20%	50.30%	54.55%	90.70%	19.40%	21.39%	91.60%	48.70%	53.16%	91.80%	25.60%	27.88%	90.60%	41.50%	45.81%	91.40%	21.90%	23.96%

(X, Y). We conclude the overall algorithm of Beowulf in Algorithm 1.

5 EVALUATIONS

5.1 Experimental Setup

Models and datasets. We conducted experiments using four commonly-used datasets: CIFAR-10 [30], CIFAR-100 [30], GTSRB [58], and ImageNette[13] [20]. The details of the datasets and model training process are listed in the appendix.

To steal these victim models, we consider a powerful attacker capable of using the same model structure as the victim model.

Evaluation metrics. To evaluate the defense performance, we employ *Target Accuracy* (ACC), *Extraction Accuracy* (EACC), and *Fidelity* in the experiments.

Target Accuracy. It measures the prediction accuracy of the fortified models, which is calculated as the fraction of correctly-labeled test samples by the fortified model. The defender should maintain a high utility of the fortified model.

Extraction Accuracy. It measures the fraction of correctly-labeled test samples by the substitute model. Since the adversary's objective is to steal the functionality of the target victim model, high extraction accuracy is desirable. In contrast, a lower extraction accuracy indicates better defense performance.

[13] ImageNette is a representative subset of ImageNet.

Fidelity. It measures the fraction of test samples whose prediction label given by the substitute model is the same as that given by the target victim model. A high fidelity is ideal if the purpose of the adversary is to mount downstream attacks, such as adversarial example attacks, against the black-box victim model. A lower fidelity indicates better defense performance.

Model extraction attacks and baselines. To evaluate the defense performance of Beowulf and baselines, we consider six state-of-the-art model extraction attacks, including KnockoffNets (soft-label attack) [59], JBDA (soft-label attack) [48], DFME (soft-label attack) [64], MAZE (soft-label attack) [27], DFMS (hard-label attack) [54], and D-DAE (soft-label attack) [4].

We consider four state-of-the-art model extraction defenses as the baselines, including AM [28], EDM [26], PP [46], and Mo [62]. Due to the source codes of [24] are not open, we did not compare Beowulf with it. But note that TABLE XIII in [24] shows that KnockoffNets achieves high test accuracy of 70.04% and 57.04% for CIFAR-10 and ImageNette respectively under AMAO. In contrast, Beowulf can reduce the test accuracy of KnockoffNets to below 40% for CIFAR-10 and ImageNette.

More details of the model extraction attacks and baseline defenses considered in the experiments are demonstrated in the appendix.

5.2 Comparison with Baseline Defenses

We compare the defense performance of Beowulf with four state-of-the-art baselines against six representative model extraction attacks. The comparison results are shown in Table 1 and Table 2. "Unprotected" defense represents that no defense mechanisms are applied. The ACC of the "Unprotected" defense denotes the original prediction accuracy of the target victim model. The EACC and Fidelity of the "Unprotected" defense indicate the attack performance of various model extraction attacks before the defense.

We can see that Beowulf consistently and markedly reduces EACC and Fidelity in substitute models trained with different attacks across various datasets. In most cases, Beowulf outperforms baselines, achieving lower EACC and Fidelity. It effectively neutralizes all attacks, while baseline defenses are ineffective against one or more listed attacks, particularly D-DAE. Furthermore, Beowulf minimally affects the utility of the victim model, keeping the prediction accuracy drop to less than 3%. In contrast, other baseline defenses, especially AM and EDM, significantly undermine the model's utility. For instance, when applied to the CIFAR-10 dataset, Beowulf reduces the EACC and Fidelity of D-DAE from 71.86% and 78.81% to 34.05% and 36.10%, respectively. Conversely, D-DAE can also maintain a high EACC of 86.30%, 74.98%, 83.20%, and 45.30%, as well as a high Fidelity of 99.24%, 86.23%, 91.23%, and 49.50%, even after defense from AM, EDM, PP, and Mo, respectively. Moreover, Beowulf achieves the highest ACC of 91.76% compared to the baselines. In this case, AM and EDM achieve ACC rates of only 85.24% and 86.95%, respectively. Even for more complex datasets like ImageNette, Beowulf can also significantly reduce the EACC and ACC of different model extraction attacks to less than 30% in most cases, which is more effective than the baselines.

5.3 Impact of the Structure of Target Model

Table 4: Number of test samples classified to the dummy class by the target model and surrogate models stolen by various model extraction attacks.

Dataset	Target model	KnockoffNets	DFME	JBDA	MAZE	D-DAE	DFMS
CIFAR-10	1	0	2	1	1	0	3
CIFAR-100	1	1	1	0	2	1	0
GTSRB	0	0	1	0	2	1	2
ImageNette	0	0	0	1	1	2	1

In addition to ResNet-18 and ResNet-50, we also examine the efficacy of Beowulf in safeguarding models based on other architectural frameworks, such as VGG-19 [56], DenseNet [22], and WideResNet [75]. To evaluate the defense capabilities of Beowulf, we deploy it to protect models trained on these diverse architectures and initiate model extraction attacks according to MAZE [27].

As depicted in Table 3, the results affirm the defense robustness of Beowulf under different model structures. For instance, in experiments on the CIFAR-10 dataset, Beowulf reduces the EACC from 71.83% to 28.79% for VGG-19, from 76.54% to 21.26% for DenseNet, and from 88.69% to 36.13% for WideResNet.

5.4 Impact of Dummy Class

To explore the impact of the dummy class, we present the number of test samples that are classified as dummy class by the target model and the surrogate model in Table 4. It is shown that an extremely small number of test samples are classified into the dummy class during the model extraction process. Take CIFAR-10 as an example, for the target model, the number is 1, and for the surrogate models extracted using different extraction attacks, the numbers are 0 for KnockoffNets, 2 for DFME, 1 for JBDA, 1 MAZE, 0 for D-DAE, 3 for DFMS. The probable reason is that the dummy class samples are carefully designed to alter the decision boundary, creating a substantial difference between dummy class samples and actual samples. This shows that the introduction of dummy class samples has little impact on honest clients. Besides, the extraction attacks will not succeed even if the attacker removes the query samples classified into the dummy class.

5.5 Effectiveness of Different Mixup-based Data Augmentation Methods

We explore the effectiveness of Beowulf when using various state-of-the-art mixup-based data augmentation strategies, such as CutMix [74], MixUp [76], FMix [16], and SaliencyMix [65]. The defense performance is shown in Table 5. We can see that each of these mixup-based data augmentation strategies demonstrates an obvious reduction in the EACC and Fidelity. Among these data augmentation strategies, CutMix can achieve the best defense performance in most cases. Although in certain scenarios, Mixup (D-DAE in ImageNette), FMix (DFME in CIFAR-10 and CIFAR-100, JBDA in CIFAR-100), and SaliencyMix (D-DAE in ImageNette, MAZE in CIFAR-100, and DFME in CIFAR-100) have demonstrated the capability to decrease EACC and Fidelity more significantly than that of Beowulf, Beowulf can attain a superior ACC compared to them. In Beowulf, we default to using CutMix. Note that the defenders have the flexibility to choose the most suitable data augmentation strategy based on the defense performance.

Table 5: Defense performance of Beowulf when using various mixup-based data augmentation methods, including CutMix [74], MixUp [76], FMix [16], and SaliencyMix [65].

Attacks								CIFAR-10							
	Unprotected			CutMix [74]			MixUp[76]			FMix [16]			SaliencyMix [65]		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
KnockoffNets	94.63%	79.99%	84.53%	91.76%	**39.27%**	**41.57%**	79.34%	60.61%	76.39%	**94.16%**	53.24%	56.54%	83.63%	64.83%	77.52%
DFME	94.63%	82.18%	90.12%	91.76%	30.26%	32.03%	79.34%	**17.99%**	**22.67%**	**94.16%**	21.88%	26.41%	83.63%	30.86%	32.26%
JBDA	94.63%	51.47%	56.44%	91.76%	**9.02%**	**9.80%**	79.34%	13.56%	17.38%	**94.16%**	15.21%	16.16%	83.63%	11.54%	13.81%
MAZE	94.63%	81.85%	89.76%	91.76%	**27.42%**	**29.12%**	79.34%	66.68%	84.04%	**94.16%**	58.87%	68.24%	83.63%	60.04%	71.29%
D-DAE	94.63%	71.86%	78.81%	91.76%	**34.05%**	**36.10%**	79.34%	49.80%	63.84%	**94.16%**	43.46%	45.69%	83.63%	53.78%	64.33%
DFMS	94.63%	83.56%	85.89%	91.76%	**10.04%**	**11.55%**	79.34%	11.56%	14.57%	**94.16%**	17.59%	86.75%	83.63%	15.65%	18.51%

Attacks								CIFAR-100							
	Unprotected			CutMix [74]			MixUp [76]			FMix [16]			SaliencyMix [65]		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
KnockoffNets	77.79%	43.49%	55.91%	**77.80%**	24.99%	**32.12%**	58.81%	37.87%	64.39%	61.39%	42.13%	69.24%	50.26%	**24.37%**	48.64%
DFME	77.79%	54.62%	70.21%	**77.80%**	23.83%	30.63%	58.21%	20.91%	35.92%	61.39%	**11.54%**	**18.80%**	50.26%	13.22%	26.30%
JBDA	77.79%	1.00%	1.28%	**77.80%**	**1.67%**	2.11%	58.81%	2.16%	2.46%	61.39%	1.93%	2.34%	50.26%	**1.67%**	**1.95%**
MAZE	77.79%	50.19%	64.52%	**77.80%**	18.94%	**24.34%**	58.81%	25.38%	43.16%	61.39%	17.35%	28.26%	50.26%	**14.74%**	29.33%
D-DAE	77.79%	49.60%	63.79%	**77.80%**	20.49%	**26.66%**	58.81%	31.05%	52.80%	61.39%	34.55%	56.78%	50.26%	**19.98%**	38.24%
DFMS	77.79%	56.87%	73.11%	**77.80%**	30.27%	30.91%	58.81%	**27.69%**	47.08%	61.39%	**20.47%**	33.34%	50.26%	23.61%	46.98%

Attacks								GTSRB							
	Unprotected			CutMix [74]			MixUp [76]			FMix [16]			SaliencyMix [65]		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
KnockoffNets	94.22%	92.40%	94.07%	91.87%	**38.46%**	**41.86%**	92.44%	40.44%	43.07%	90.25%	47.13%	52.22%	81.45%	56.35%	69.18%
DFME	94.22%	88.64%	90.25%	91.87%	**36.42%**	**39.64%**	**92.44%**	57.93%	66.44%	90.25%	48.48%	55.81%	81.45%	51.32%	63.01%
JBDA	94.22%	26.80%	27.29%	91.87%	9.60%	10.44%	**92.44%**	9.80%	11.21%	90.25%	**8.70%**	**9.64%**	81.45%	15.40%	18.89%
MAZE	94.22%	94.20%	95.91%	91.87%	**33.20%**	**36.13%**	**92.44%**	67.22%	77.10%	90.25%	56.51%	65.05%	81.45%	48.29%	59.28%
D-DAE	94.22%	92.16%	93.83%	91.87%	**26.78%**	**29.15%**	**92.44%**	38.90%	44.52%	90.25%	51.20%	56.73%	81.45%	59.60%	73.17%
DFMS	94.22%	91.61%	93.27%	91.87%	**29.67%**	**32.29%**	**92.44%**	47.70%	50.69%	90.25%	65.05%	69.03%	81.45%	47.92%	58.83%

Attacks								ImageNette							
	Unprotected			CutMix [74]			MixUp [76]			FMix [16]			SaliencyMix [65]		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
KnockoffNets	89.06%	78.58%	85.94%	91.43%	**35.90%**	**39.26%**	91.81%	52.67%	57.37%	**92.00%**	46.80%	50.86%	89.20%	86.60%	97.08%
DFME	89.06%	51.02%	57.29%	91.43%	24.08%	26.40%	91.81%	26.33%	30.65%	**92.00%**	**14.35%**	**15.58%**	89.20%	19.35%	21.69%
JBDA	89.06%	16.80%	18.86%	91.43%	**10.60%**	**11.62%**	91.81%	11.20%	12.19%	**92.00%**	10.80%	11.74%	89.20%	14.60%	16.36%
MAZE	89.06%	49.89%	56.02%	91.43%	25.00%	**27.40%**	91.81%	**24.60%**	28.67%	**92.00%**	29.60%	32.17%	89.20%	34.00%	38.11%
D-DAE	89.06%	83.29%	97.60%	91.43%	13.20%	14.47%	91.81%	**9.90%**	**10.77%**	**92.00%**	10.20%	11.09%	89.20%	10.70%	11.99%
DFMS	89.06%	68.28%	74.68%	91.43%	22.51%	24.62%	91.81%	38.87%	43.19%	**92.00%**	**14.31%**	**16.46%**	89.20%	37.29%	41.43%

Table 6: Impact of noise magnitude.

Dataset	Unprotected			$0.5n_r$			n_r			$2n_r$		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
CIFAR-10	94.63%	79.99%	89.53%	86.46%	47.04%	61.52%	**91.76%**	**39.27%**	**41.57%**	90.31%	46.37%	61.57%
CIFAR-100	77.79%	43.49%	55.91%	73.72%	30.19%	49.72%	**77.80%**	**24.99%**	**32.12%**	72.00%	27.62%	48.46%
GTSRB	94.22%	92.40%	94.07%	85.12%	37.91%	44.53%	**91.87%**	38.46%	**41.86%**	90.79%	**31.26%**	56.03%
ImageNette	89.06%	78.58%	85.94%	85.68%	42.18%	49.22%	**91.43%**	**35.90%**	**39.26%**	88.58%	38.42%	43.37%

Table 7: Impact of maxing parameter.

Dataset	Unprotected			0.3			0.5			0.7		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
CIFAR-10	94.63%	79.99%	84.53%	87.99%	47.73%	61.20%	**91.76%**	**39.27%**	**41.57%**	89.21%	49.11%	61.99%
CIFAR-100	77.79%	43.49%	55.91%	74.12%	33.54%	52.30%	**77.80%**	**24.99%**	**32.12%**	72.06%	36.53%	58.86%
GTSRB	94.22%	92.40%	94.07%	90.93%	**22.17%**	**24.38%**	**91.87%**	38.46%	41.86%	87.26%	28.69%	42.65%
ImageNette	89.06%	78.58%	85.94%	84.98%	39.72%	46.74%	**91.43%**	**35.90%**	**39.26%**	89.87%	40.12%	44.63%

5.6 Impact of Noise Magnitude

By default, we generate the random noise n_r following a normal distribution with a mean of 0 and a standard deviation of 1. To study the impact of noise magnitude on Beowulf, we test different magnitudes of noise values, i.e., $0.5 \times n_r$ and $2 \times n_r$.

As presented in Table 6, random noises of varying magnitudes can notably reduce model extraction performance. Notably, the normal distribution with a mean of 0 and a standard deviation of 1 consistently exhibits superior performance across most scenarios. Therefore, we adopt a default setting where random noise n_r follows a normal distribution.

Table 8: Impact of strength of adversarial noise.

Dataset	Unprotected			$0.15n_a$			$0.20n_a$			$0.25n_a$			Beowulf		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity	ACC	EACC	Fidelity
CIFAR-10	94.63%	79.99%	84.53%	91.32%	62.08%	76.34%	90.29%	57.90%	70.36%	91.72%	59.81%	65.22%	**91.76%**	**39.27%**	**41.57%**
CIFAR-100	77.79%	43.49%	55.91%	71.66%	29.45%	47.76%	73.54%	30.79%	49.23%	72.29%	31.71%	43.87%	**77.80%**	**24.99%**	**32.12%**
GTSRB	94.22%	92.40%	94.07%	89.01%	65.95%	79.45%	89.76%	61.09%	72.94%	90.93%	65.34%	71.86%	**91.87%**	**38.46%**	**41.86%**
ImageNette	89.06%	78.58%	85.94%	83.97%	60.14%	74.28%	85.27%	58.35%	73.61%	89.83%	64.91%	72.26%	**91.43%**	**35.90%**	**39.26%**

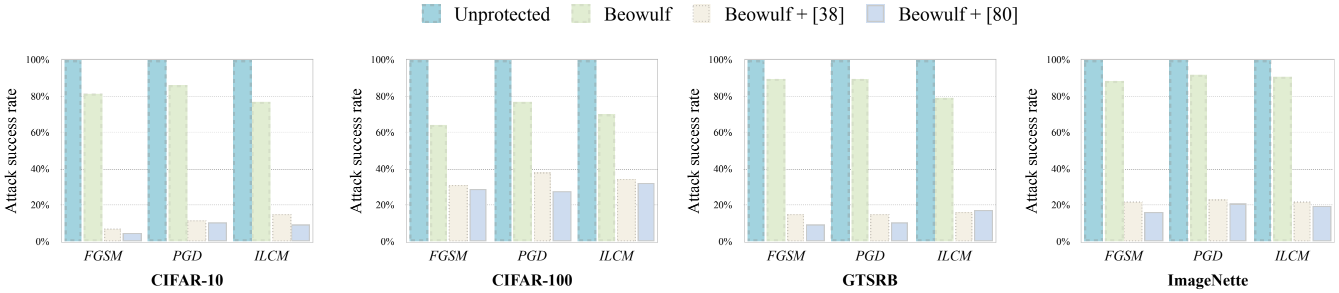

Figure 3: Robustness against adversarial example attacks. A lower attack success rate reflects stronger resistance to such attacks.

5.7 Impact of Maxing Parameter

The “maxing parameter" refers to the proportion of original data to dummy data in the retraining dataset. In this part, we explore the impact of the “maxing parameter" on Beowulf by varying its number as 0.3, 0.5, and 0.7, respectively. The results are shown in Table 7. We can see that a value of 0.5 consistently achieves the optimal defense performance across various scenarios in most cases. For GTSRB, a value of 0.3 can decrease the EACC and Fidelity more than that of the value of 0.5, but its ACC is relatively lower. Consequently, we set the maxing parameter as 0.5 in the experiments. Note that the defender can choose the most suitable value of the maxing parameter according to the defense performance.

5.8 Strength of Adversarial Noise

To evaluate the impact of added noises on the defense performance, we generate perturbations that approach the decision boundary but do not cross it by multiplying the adversarial noise by coefficients of 0.15, 0.20, and 0.25, respectively. We have ensured that these noises will not change a sample's original label, i.e., do not cross the decision boundary. We then use these new noises to form the dummy class. The defense performance comparison between Beowulf and such perturbations is shown in Table 8.

We can see that the perturbations that do not cross the decision boundary fail to yield comparable results with Beowulf. For example, the original ACC, EACC, and Fidelity of the KnockoffNets attack are 94.63%, 79.99%, and 84.53% on CIFAR-10. Under the protection of $0.25n_a$, the ACC, EACC, and Fidelity are 91.72%, 59.81%, and 65.22%, while Beowulf has achieved 91.76%, 39.27%, and 41.57% respectively. This is probably because the perturbations that do not cross the decision boundary cannot effectively reshape the decision boundary and squeeze the normal classes.

5.9 Adversarial Robustness of Beowulf

We test the adversarial robustness of Beowulf. For each dataset, we randomly selected 1,000 samples and generated corresponding adversarial examples using FGSM [14], PGD [40], and ILCM [31] attacks, respectively. These adversarial examples are fed into both unprotected models and models protected by Beowulf to assess the attack success rate. Since Beowulf and adversarial example detection techniques are complementary, we further analyzed the robustness of combinations of Beowulf with different adversarial example detection methods (Beowulf + [38] and Beowulf + [80]). The first approach [38] introduces an energy score to differentiate Out-of-Distribution (OOD) examples as potential adversarial examples. The second method [80] detects adversarial samples based on the assumption that adversarial examples are more susceptible to perturbations than clean samples. We use adversarial example detection techniques to identify and remove potentially malicious input samples, and then feed the remaining samples into Beowulf-protected models.

We consider untargeted adversarial attacks where an attack is considered to be successful if the predicted label of the adversarial example differs from its true label. Note that untargeted adversarial attacks are easier to perform by attackers and harder to be detected. Additionally, if the sample is classified into the dummy class in the Beowulf-protected model, the attack is deemed unsuccessful, as the image is considered unrelated to the model's task. The results are shown in the Figure 3.

It is revealed that nearly all unprotected models are susceptible to the three adversarial attacks, with success rates of almost 100%. In contrast, models protected by Beowulf (light green pillar) exhibit significantly lower attack success rates, which even drop to around 60%. When Beowulf is further equipped with adversarial example detection methods, the adversarial attack success rates further decrease to below 37%. This suggests that Beowulf contributes to a modest improvement in the model's adversarial robustness, while its integration with adversarial example detection techniques further strengthens the model's adversarial robustness.

6 DISCUSSION

6.1 Theoretical Foundations of Beowulf

The effectiveness of Beowulf can be supported by information theory [1]. By introducing a dummy class, Beowulf increases the entropy of the model's output, making the outputs less predictable near decision boundaries for the attacker. Additionally, Beowulf reduces mutual information between inputs and outputs, further obstructing the extraction process.

Enhancing output entropy. The conditional entropy, $H(Y|X)$, quantifies the uncertainty of the model's output Y given an input X. By introducing a dummy class with random and adversarial noises, Beowulf tends to increase the conditional entropy $H'(Y|X) > H(Y|X)$, enhancing the model's resistance against extraction attacks.

Reducing mutual information. The mutual information $I(X;Y) = H(Y) - H(Y|X)$ quantifies the amount of information about the input X that can be inferred from the output Y. Beowulf reduces the mutual information to ensure that less information about the model is revealed through its outputs, increasing the difficulty of model extraction attacks.

Let $P(X,Y)$ represent the joint probability distribution of inputs and outputs, and $P(Y|X)$ denote the conditional probability distribution of outputs given inputs. Mutual information is formally defined as:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right). \tag{6}$$

Upon applying Beowulf, the joint and conditional probabilities become $P'(X,Y)$ and $P'(Y|X)$, respectively, resulting in a new mutual information $I'(X;Y)$:

$$I'(X;Y) = \sum_{x\in X}\sum_{y\in Y} P'(x,y)\log\left(\frac{P'(x,y)}{P'(x)P'(y)}\right). \tag{7}$$

The change in mutual information due to Beowulf is $\Delta I = I'(X;Y) - I(X;Y)$. The Kullback-Leibler (KL) divergence [18], a metric of the difference between two probability distributions, can be used to quantify the information change:

$$D_{KL}(P(Y|X)||P'(Y|X)) = \sum_{x\in X}\sum_{y\in Y} P(y|X=x)\log\left(\frac{P(y|X=x)}{P'(y|X=x)}\right). \tag{8}$$

The KL divergence is non-negative and represents the increase in conditional entropy due to Beowulf. The inequality $H'(Y|X) \geq H(Y|X) + D_{KL}(P(Y|X)||P'(Y|X))$ indicates that the changed distribution $P'(Y|X)$ is farther away from the original distribution $P(Y|X)$ than the KL divergence alone, reflecting the additional uncertainty introduced by Beowulf.

6.2 Extend to NLP Domain

We also extend Beowulf to text classification tasks in natural language processing (NLP), using the BERT structure. To generate enough dummy class data, we utilize an unrelated text dataset, Yelp Polarirty [78], to enlarge the random noise data. To ensure the defense performance, we set the proportion of dummy class data to normal class data as 7:1.

Table 9: Extend Beowulf to NLP domain.

Dataset	Unprotected			Beowulf-NLP		
	ACC	EACC	Fidelity	ACC	EACC	Fidelity
Blog	97.12%	88.28%	90.90%	95.94%	31.22%	32.54%
TP-US	85.50%	85.30%	99.76%	83.66%	6.07%	7.25%
AG	94.52%	88.65%	93.78%	94.15%	34.35%	36.48%

We test the defense performance of Beowulf under a state-of-the-art model extraction attack [17] across three text classification tasks, including Blog [55], TP-US [19], and AG [7]. As shown in Table 9, Beowulf can decrease the EACC from 88.28%, 85.30%, and 88.65% to 31.22%, 6.07% and 34.35% for Blog, TP-US, and AG datasets, respectively. Besides, Beowulf can decrease the Fidelity from 99.90%, 99.76%, and 93.78% to 32.54%, 7.25%, and 36.48%, while maintaining a high ACC of the protected models. The results show that Beowulf can be effectively extended to the NLP domain.

7 CONCLUSION AND FUTURE WORK

This paper presents an effective defense strategy to counteract model extraction attacks by reshaping decision regions. Instead of randomly changing decision regions, we carefully design a dummy class to condense the decision regions and complexify the decision boundaries, making model extraction more challenging. Besides, we propose to use the mixup-based data augmentation technique to ensure a seamless fusion of the dummy class with original datasets, leading to a notable enhancement in the fortified model's utility. Comprehensive experiments across 4 datasets, testing against 6 state-of-the-art attacks, show the superior performance of Beowulf compared to 4 state-of-the-art model extraction defenses.

Beowulf has several limitations that we plan to address in subsequent research endeavors. First, we mainly evaluate the defensive capabilities of Beowulf in the visual domain but not in other domains, e.g., voice and video. However, given that model extraction attacks [21, 29] and feature redundancy [53, 67, 81] can occur in various domains, there's potential for Beowulf to be more broadly applicable. In the future, we aim to adapt Beowulf to diverse data types. Second, Beowulf employs a meticulously designed dummy class to condense the decision regions and intricate the decision boundary. Nonetheless, Beowulf may slightly diminish the prediction accuracy of the protected models in some cases, albeit by less than 3%. In the future, we will further optimize Beowulf, especially the retraining dataset construction process, to bolster the accuracy of the reinforced model. Third, it is worth exploring effective adaptive attacks that can bypass Beowulf. Last but not least, we will further ameliorate the fortifying methodology of Beowulf to make it resilient against other inference-time attacks, such as adversarial example attacks.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their valuable comments. Qian Wang's work was partially supported by the National Key R&D Program of China (2020AAA0107701) and the NSFC under Grants U20B2049 and U21B2018.

REFERENCES

[1] Babajide O Ayinde, Tamer Inanc, and Jacek M Zurada. 2019. Redundant feature pruning for accelerated inference in deep neural networks. *Neural Networks* 118 (2019), 148–158.

[2] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*. 39–57.

[3] Lingjiao Chen, Matei Zaharia, and James Zou. 2023. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176* (2023).

[4] Yanjiao Chen, Rui Guan, Xueluan Gong, Jianshuo Dong, and Meng Xue. 2023. D-DAE: Defense-penetrating model extraction attacks. In *IEEE Symposium on Security and Privacy*. 382–399.

[5] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *IEEE International Joint Conference on Neural Networks*. 2921–2926.

[6] Jacson Rodrigues Correia-Silva, Rodrigo F Berriel, Claudine Badue, Alberto F de Souza, and Thiago Oliveira-Santos. 2018. Copycat CNN: Stealing knowledge by persuading confession with random non-labeled data. In *IEEE International Joint Conference on Neural Networks*. 1–8.

[7] Gianna M Del Corso, Antonio Gulli, and Francesco Romani. 2005. Ranking a stream of news. In *International Conference on World Wide Web*. 97–106.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*. 248–255.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*. Springer, 265–284.

[10] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. 2018. Empirical study of the topology and geometry of deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*. 3762–3770.

[11] Xueluan Gong, Yanjiao Chen, Wenbin Yang, Guanghao Mei, and Qian Wang. 2021. INVERSENET: Augmenting model extraction attacks with training data inversion. In *International Joint Conference on Artificial Intelligence*. ijcai.org, 2439–2447.

[12] Xueluan Gong, Qian Wang, Yanjiao Chen, Wang Yang, and Xinchang Jiang. 2020. Model extraction attacks and defenses on cloud-based machine learning models. *IEEE Communications Magazine* 58, 12 (2020), 83–89.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[15] Yiwen Guo, Chao Zhang, Changshui Zhang, and Yurong Chen. 2018. Sparse dnns with improved adversarial robustness. *Advances in Neural Information Processing Systems* 31 (2018).

[16] Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, Adam Prügel-Bennett, and Jonathon Hare. 2020. FMix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047* (2020).

[17] Xuanli He, Lingjuan Lyu, Qiongkai Xu, and Lichao Sun. 2021. Model extraction and adversarial transferability, your BERT is vulnerable! *arXiv preprint arXiv:2103.10013* (2021).

[18] John R Hershey and Peder A Olsen. 2007. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 4. IV–317.

[19] Dirk Hovy, Anders Johannsen, and Anders Søgaard. 2015. User review sites as a resource for large-scale sociolinguistic studies. In *International Conference on World Wide Web*. 452–461.

[20] Jeremy Howard. 2019. Imagenette: A smaller subset of 10 easily classified classes from Imagenet. (2019).

[21] Tsu-Yuan Hsu, Chen-An Li, Tung-Yu Wu, and Hung-yi Lee. 2022. Model extraction attack against self-supervised speech models. *arXiv preprint arXiv:2211.16044* (2022).

[22] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. 2014. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869* (2014).

[23] Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. 2021. Entangled watermarks as a defense against model extraction. In *USENIX Security Symposium*. 1937–1954.

[24] Wenbo Jiang, Hongwei Li, Guowen Xu, Tianwei Zhang, and Rongxing Lu. 2023. A comprehensive defense framework against model extraction attacks. *IEEE Transactions on Dependable and Secure Computing* (2023).

[25] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. 2019. PRADA: Protecting against DNN model stealing attacks. In *IEEE European Symposium on Security and Privacy*. 512–527.

[26] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. 2020. Protecting DNNs from theft using an ensemble of diverse models. In *International Conference on Learning Representations*.

[27] Sanjay Kariyappa, Atul Prakash, and Moinuddin K Qureshi. 2021. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13814–13823.

[28] Sanjay Kariyappa and Moinuddin K Qureshi. 2020. Defending against model stealing attacks with adaptive misinformation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 770–778.

[29] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2019. Thieves on sesame street! model extraction of bert-based apis. *arXiv preprint arXiv:1910.12366* (2019).

[30] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[31] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *International Conference on Learning Representations*. OpenReview.net.

[32] Yann LeCun, Corinna Cortes, and Chris Burges. 2010. MNIST handwritten digit database. (2010).

[33] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. 2019. Defending against machine learning model stealing attacks using deceptive perturbations. In *IEEE Security and Privacy Workshops*.

[34] Chuan Li. 2020. OpenAI's GPT-3 language model: A technical overview. *Blog Post* (2020).

[35] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, Vol. 31.

[36] Zeyu Li, Chenghui Shi, Yuwen Pu, Xuhong Zhang, Yu Li, Jinbao Li, and Shouling Ji. 2023. MEAOD: Model extraction attack against object detectors. *arXiv preprint arXiv:2312.14677* (2023).

[37] Zongjie Li, Chaozheng Wang, Pingchuan Ma, Chaowei Liu, Shuai Wang, Daoyuan Wu, and Cuiyun Gao. 2023. On the feasibility of specialized ability stealing for large language code models. *arXiv preprint arXiv:2303.03012* (2023).

[38] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems* 33 (2020), 21464–21475.

[39] Shaohao Lu, Yuqiao Xian, Ke Yan, Yi Hu, Xing Sun, Xiaowei Guo, Feiyue Huang, and Wei-Shi Zheng. 2021. Discriminator-free generative adversarial attack. In *ACM International Conference on Multimedia*. 1544–1552.

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*.

[41] Ali Naseh, Kalpesh Krishna, Mohit Iyyer, and Amir Houmansadr. 2023. Stealing the decoding algorithms of language models. In *ACM SIGSAC Conference on Computer and Communications Security*. 1835–1849.

[42] Jasmina Dj Novaković, Alempije Veljović, Siniša S Ilić, Željko Papić, and Milica Tomović. 2017. Evaluation of classification models in machine learning. *Theory and Applications of Mathematics & Computer Science* 7, 1 (2017), 39.

[43] Kavi B Obaid, Subhi Zeebaree, Omar M Ahmed, et al. 2020. Deep learning models based on image classification: A review. *International Journal of Science and Business* 4, 11 (2020), 75–81.

[44] Daryna Oliynyk, Rudolf Mayer, and Andreas Rauber. 2023. I know what you trained last summer: A survey on stealing machine learning models and defences. *Comput. Surveys* (2023).

[45] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff Nets: Stealing functionality of black-box models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4954–4963.

[46] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2020. Prediction poisoning: Towards defenses against DNN model stealing attacks. In *International Conference on Learning Representations*.

[47] Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. 2020. Activethief: Model extraction using active learning and unannotated public data. In *AAAI Conference on Artificial Intelligence*, Vol. 34. 865–872.

[48] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *ACM on Asia Conference on Computer and Communications Security*. 506–519.

[49] Han Qiu, Yi Zeng, Tianwei Zhang, Yong Jiang, and Meikang Qiu. 2020. Fencebox: A platform for defeating adversarial examples with data augmentation techniques. *arXiv preprint arXiv:2012.01701* (2020).

[50] VN Ganapathi Raju, K Prasanna Lakshmi, Vinod Mahesh Jain, Archana Kalidindi, and V Padma. 2020. Study the influence of normalization/transformation process on the accuracy of supervised classification. In *International Conference on Smart Systems and Inventive Technology*. IEEE, 729–735.

[51] Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz. 2015. MLaaS: Machine learning as a service. In *IEEE International Conference on Machine Learning and Applications*. 896–902.

[52] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. 2016. Adversarial manipulation of deep representations. In *International Conference on Learning*

Representations.

[53] Sherine Nagi Saleh and Yasser El-Sonbaty. 2007. A feature selection algorithm with redundancy reduction for text classification. In *International Symposium on Computer and Information Sciences.* IEEE, 1–6.

[54] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu. 2022. Towards data-free model stealing in a hard label setting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 15284–15293.

[55] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging.. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, Vol. 6. 199–205.

[56] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations.* OpenReview.net.

[57] Gowthami Somepalli, Liam Fowl, Arpit Bansal, Ping Yeh-Chiang, Yehuda Dar, Richard Baraniuk, Micah Goldblum, and Tom Goldstein. 2022. Can neural nets learn the same model twice? Investigating reproducibility and double descent from the decision boundary perspective. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 13699–13708.

[58] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2011. The German traffic sign recognition benchmark: A multi-class classification competition. In *IEEE International Joint Conference on Neural Networks.* 1453–1460.

[59] Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning.* Vol. 135. MIT press Cambridge.

[60] Sebastian Szyller, Buse Gul Atli, Samuel Marchal, and N Asokan. 2021. Dawn: Dynamic adversarial watermarking of neural networks. In *ACM International Conference on Multimedia.* 4417–4425.

[61] Minxue Tang, Anna Dai, Louis DiValentin, Aolin Ding, Amin Hass, Neil Zhenqiang Gong, and Yiran Chen. 2024. MODELGUARD: Information-theoretic defense against model extraction attacks. In *USENIX Security Symposium.*

[62] Guanhong Tao, Yingqi Liu, Guangyu Shen, Qiuling Xu, Shengwei An, Zhuo Zhang, and Xiangyu Zhang. 2022. Model orthogonalization: Class distance hardening in neural networks for better security. In *IEEE Symposium on Security and Privacy.* 1372–1389.

[63] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction APIs. In *USENIX Security Symposium.* 601–618.

[64] Jean-Baptiste Truong, Pratyush Maini, Robert J Walls, and Nicolas Papernot. 2021. Data-free model extraction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 4771–4780.

[65] AFM Uddin, Mst Monira, Wheemyung Shin, TaeChoong Chung, Sung-Ho Bae, et al. 2021. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations.*

[66] Jonathan Vanian. 2023. ChatGPT and generative AI are booming, but the costs can be extraordinary. (2023).

[67] De Wang, Feiping Nie, and Heng Huang. 2015. Feature selection via global redundancy minimization. *IEEE Transactions on Knowledge and Data Engineering* 27, 10 (2015), 2743–2755.

[68] Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. 2022. MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters. In *USENIX Symposium on Networked Systems Design and Implementation.* 945–960.

[69] Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. 2021. Patchguard: A provably robust defense against adversarial patches via small receptive fields and masking. In *USENIX Security Symposium.*

[70] Haonan Yan, Xiaoguang Li, Hui Li, Jiamin Li, Wenhai Sun, and Fenghua Li. 2021. Monitoring-based differential privacy mechanism against query flooding-based model extraction attack. *IEEE Transactions on Dependable and Secure Computing* (2021).

[71] Yaoqing Yang, Rajiv Khanna, Yaodong Yu, Amir Gholami, Kurt Keutzer, Joseph E Gonzalez, Kannan Ramchandran, and Michael W Mahoney. 2020. Boundary thickness and robustness in learning models. In *Advances in Neural Information Processing Systems.* 6223–6234.

[72] Honggang Yu, Kaichen Yang, Teng Zhang, Yun-Yun Tsai, Tsung-Yi Ho, and Yier Jin. 2020. CloudLeak: Large-scale deep learning models stealing through adversarial examples.. In *Annual Network and Distributed System Security Symposium.* The Internet Society.

[73] Lei Yu and Huan Liu. 2003. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *International Conference on Machine Learning.* 856–863.

[74] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. CutMix: Regularization strategy to train strong classifiers with localizable features. In *IEEE/CVF International Conference on Computer Vision.* 6023–6032.

[75] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146* (2016).

[76] H Zhang, M Cisse, Y Dauphin, and D Lopez-Paz. 2018. mixup: Beyond empirical risk management. In *International Conference on Learning Representations.* 1–13.

[77] Jiliang Zhang, Shuang Peng, Yansong Gao, Zhi Zhang, and Qinghui Hong. 2023. APMSA: Adversarial perturbation against model stealing attacks. *IEEE Transactions on Information Forensics and Security* 18 (2023), 1667–1679.

[78] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems* 28 (2015).

[79] Huadi Zheng, Qingqing Ye, Haibo Hu, Chengfang Fang, and Jie Shi. 2019. Bdpl: A boundary differentially private layer against machine learning model extraction attacks. In *European Symposium on Research in Computer Security.* Springer, 66–83.

[80] Qifei Zhou, Rong Zhang, Bo Wu, Weiping Li, and Tong Mo. 2020. Detection by attack: Detecting adversarial samples by undercover attack. In *European Symposium on Research in Computer Security.* Springer, 146–164.

[81] Chen Zhu, Xiao Tan, Feng Zhou, Xiao Liu, Kaiyu Yue, Errui Ding, and Yi Ma. 2018. Fine-grained video categorization with redundancy reduction attention. In *European Conference on Computer Vision.* 136–152.

APPENDIX

A DATASETS AND MODELS

We conducted experiments using four commonly-used datasets: CIFAR-10 [30], CIFAR-100 [30], GTSRB [58], and ImageNette [20].

CIFAR-10. CIFAR-10 [30] contains 60,000 images belonging to 10 classes. Each sample has a dimension of 32 × 32. We randomly select 50,000 samples as the training set, and the remaining 10,000 samples as the test set. In the experiments, we employ the ResNet-18 architecture to train the victim model. The training process spans a considerable 50 epochs to ensure accuracy. During this training phase, we adopt a learning rate of 0.1. Additionally, to optimize the training, we utilize the momentum stochastic gradient descent (SGD) method with a momentum value set at 0.9.

CIFAR-100. CIFAR-100 dataset [30] closely resembles CIFAR-10, differing in the number of classes. It comprises 100 classes, each containing 600 images. The dataset is split into 500 training images and 100 testing images per class, amounting to a comprehensive set of diverse visual data for classification tasks. In the experiments, we employ the ResNet-50 architecture to train the victim model. The training process spans 100 epochs to ensure accuracy. Throughout this training phase, a learning rate of 0.1 is employed. To further optimize the training process, we leverage the momentum stochastic gradient descent (SGD) method, with a momentum value set at 0.9. Additionally, we incorporate a weight decay of 5e-4 to regulate the model's parameters and enhance generalization.

GTSRB. GTSRB [58] comprises images of German traffic signs distributed across 43 distinct classes. It consists of 39,209 training samples and 12,630 test samples. Leveraging the annotated details, we meticulously cropped each image to focus on its primary content and subsequently resized them to a uniform dimension of 32 × 32 pixels. We train a victim model using the ResNet-18 network on the training set. With a base learning rate at 0.1, a batch size at 128, a momentum of 0.9, we train the model for 50 epochs. In addition, we use CosineAnnealingLR as our lr scheduler.

ImageNette. ImageNette [20] is a subset of ImageNet, widely used in the research community [39, 69]. ImageNette includes 9,469 training samples and 3,925 test samples. Each image has a high resolution with a dimension of 224 × 224. In the experiments, we train a ResNet-18 model for 100 epochs as the victim model. We set the learning rate as 0.001, the batch size as 16 (due to the limited GPU source), the momentum of stochastic gradient descent as 0.9, and weight decay as 0.0005.

In our experiments, we employed ResNet-18, ResNet-50, ResNet-18, and ResNet-18 architectures to train models for these datasets, respectively.

B MODEL EXTRACTION ATTACKS

We consider six state-of-the-art model extraction attacks in the experiments.

KnockoffNets. KnockoffNets [59] is a natural sample-based model extraction attack that queries the victim model with natural samples selected using reinforcement learning. In our experiments, we used CIFAR-100, CIFAR-10, CIFAR-100, and Tiny-Imagenette datasets as query datasets for the victim models trained on CIFAR-10, CIFAR-100, GTSRB, and ImageNette, respectively. Note that the training samples of the victim model and the querying samples have no overlap.

JBDA. JBDA [48] utilizes Jacobian-based augmentation to produce synthetic samples from a limited set of unlabeled *seeds*, assuming they share a distribution similar to the training samples of the victim model. In our implementation, the seed set size is 100 for CIFAR-10, GTSRB, CIFAR-100, and ImageNette. Other parameters remain consistent with those specified in the original paper.

DFME. DFME [64] is a data-free model extraction method that incorporates techniques from the field of data-free knowledge distillation. This involves utilizing a generative model to synthesize queries, maximizing disagreement between the student and teacher models. The student model underwent training with an initial learning rate of 0.1, weight decay of 5.10^{-4}, and a learning rate scheduler that multiplied the learning rate by a factor of 0.3 at 0.1×, 0.3×, and 0.5× the total training epochs. The default query budget for all datasets in the experiment is set to 20M. In the gradient approximation, one random direction is sampled ($m = 1$), and the step size is set to $\varepsilon = 10^{-3}$. Other parameters remain consistent with those in the original paper.

MAZE. MAZE [27] is a data-free model extraction attack employing zero-step estimation. This approach utilizes a generative model to generate synthetic data for model stealing without the need for any actual data. In our implementation, the query budget is set to 30M for CIFAR-10, 30M for CIFAR-100, 30M for GTSRB, and 20M for Imagenette. Other parameters are configured to the default values as specified in the original paper.

DFMS. DFMS [54] is a data-free model extraction technique based on the generative adversarial network (GAN) framework. It employs the gradient of the *clone* network as an approximation of the gradient of the targeted victim model, simultaneously training both the student and the generator to effectively extract the model. For the CIFAR-10, GTSRB, and ImageNette target victim models, CIFAR-100 is used as the query dataset, while for CIFAR-100, CIFAR-10 serves as the query set. The clone model is trained using the SGD optimizer with a momentum of 0.9, a maximum learning rate of 0.1, and a weight decay of 5×10^{-4}.

D-DAE. D-DAE [4] is an advanced defense-penetrating model extraction attack framework that aims to break disruption-based defenses. It first utilizes the disruption detection module to infer the defense mechanism adopted by the defender and then uses the disruption recovery module to restore a clean query result from the disrupted query result with well-designed generative models. In the experiments, we apply D-DAE to enhance Activethief [47], a natural sample-based model extraction method that employs active learning to choose query samples from public datasets. We utilize hyperparameters (lr=1e-3) to train each meta-classifier for each dataset. For the generative model, we use the SGD optimizer (lr=0.01) with a momentum of 0.9 to train each generative model.

C BASELINE MODEL EXTRACTION DEFENSES

Adaptive Misinformation (AM). AM [28] employs an Out-of-Distribution (OOD) detector to differentiate between benign and adversarial queries, disrupting only those queries identified as OOD. The hyperparameter τ is crucial for balancing security and accuracy. In our experiments, we set $\tau = 0.90, 0.05, 0.10$, and 0.95 for CIFAR-10, CIFAR-100, GTSRB, and ImageNette, respectively. AM needs an OOD dataset to train the outlier detector. In the experiments, we use CIFAR-100 for CIFAR-10, CIFAR-10 for CIFAR-100, SVHN for GTSRB, and Tiny ImageNette for ImageNette.

Ensemble of Diverse Models (EDM). EDM [26] trains an ensemble of diverse models to achieve high accuracy in in-distribution data and diversity in out-of-distribution data. During the inference phase, a hash function is employed to select a member of the ensemble to respond to the query. In our implementation, the parameter λ_D, which governs the ratio of importance between diversity and accuracy objectives, is chosen with the constraint of limiting the degradation in benign accuracy of the target model to less than 0.5%. The ensemble size is 5, consistent with the original paper. The auxiliary OOD datasets are selected in the same manner as for AM.

Prediction Poisoning (PP). PP [46] introduces perturbations to the query results based on a hyperparameter ϵ, representing the magnitude of the perturbation. We evaluate the defense at $\epsilon = 0.5, 0.99, and 1.1$. Remarkably, we observed that the defense effectiveness peaked at $\epsilon = 1.1$. Consequently, we default set $\epsilon = 1.1$ for the four datasets and maintain all other parameters consistent with the original paper.

Model Orthogonalization (MO). Mo [62] is a novel model fortifying technique that is introduced as an additional training step for pretrained models. This approach significantly enhances class distances while incurring reasonable training costs and minimal accuracy degradation. Although initially designed to fortify models against backdoor attacks, it can also exhibit notable resilience against model extraction attacks. In our implementation, to fortify the victim models, we apply an L^∞ bound of $\frac{4}{255}$ for CIFAR-10, CIFAR-100, and ImageNette, and 0.03 for GTSRB. We conduct the fortifying process for each class pair through 100 iterations.