# Game-Theoretic Unlearnable Example Generator

**Shuang Liu**[1, 2], **Yihan Wang**[1, 2], **Xiao-Shan Gao**[1, 2*]

[1]Academy of Mathematics and Systems Science, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
liushuang2020@amss.ac.cn, yihanwang@amss.ac.cn, xgao@mmrc.iss.ac.cn

## Abstract

Unlearnable example attacks are data poisoning attacks aiming to degrade the clean test accuracy of deep learning by adding imperceptible perturbations to the training samples, which can be formulated as a bi-level optimization problem. However, directly solving this optimization problem is intractable for deep neural networks. In this paper, we investigate unlearnable example attacks from a game-theoretic perspective, by formulating the attack as a nonzero sum Stackelberg game. First, the existence of game equilibria is proved under the normal setting and the adversarial training setting. It is shown that the game equilibrium gives the most powerful poison attack in that the victim has the lowest test accuracy among all networks within the same hypothesis space, when certain loss functions are used. Second, we propose a novel attack method, called the Game Unlearnable Example (GUE), which has three main gradients. (1) The poisons are obtained by directly solving the equilibrium of the Stackelberg game with a first-order algorithm. (2) We employ an autoencoder-like generative network model as the poison attacker. (3) A novel payoff function is introduced to evaluate the performance of the poison. Comprehensive experiments demonstrate that GUE can effectively poison the model in various scenarios. Furthermore, the GUE still works by using a relatively small percentage of the training data to train the generator, and the poison generator can generalize to unseen data well. Our implementation code can be found at https://github.com/hong-xian/gue.

## Introduction

Deep learning has achieved remarkable success in various fields, including computer vision (He et al. 2016), natural language processing, and large language models (Brown et al. 2020), where the acquisition of a substantial amount of training data is typically necessary. However, in practical scenarios, there exists a potential issue of collecting unauthorized private data from the Internet to train these models. This raises significant privacy concerns and underscores the need to address how to protect personal data from exploitation.

To prevent unauthorized use of private data, several poisoning methods (Feng, Cai, and Zhou 2019; Huang et al. 2021; Fowl et al. 2021; Yuan and Wu 2021; Tao et al. 2021; Yu et al. 2022; van Vlijmen et al. 2022; Sandoval-Segura et al. 2022b) have been proposed. These methods involve adding imperceptible perturbations to the training samples, thereby poisoning the models and leading to poor performance on clean test data, thus making the training data unlearnable. However, it is generally agreed that the above methods are vulnerable to adversarial training. In response, novel approaches such as (Fu et al. 2022; Wen et al. 2023) have been proposed to generate robust unlearnable examples that can withstand adversarial training. Commonly referred to as "unlearnable examples attack" or "availability attack", these poison attacks have attracted significant attention.

In previous works (Feng, Cai, and Zhou 2019; Tao et al. 2021; Yu et al. 2022), unlearnable example attacks were usually formulated as a bi-level optimization problem. But directly solving the optimization problem is intractable, and Feng, Cai, and Zhou (2019) used an alternative update and Yuan and Wu (2021) used Neural Tangent Kernels (Jacot, Gabriel, and Hongler 2018) to approximately optimize the bi-level objective. Yu et al. (2022) designed poison perturbations that are easily learned by the model as “shortcuts” to prevent learning the information from the real data.

In this paper, we formulate the unlearnable example attack as a non-zero sum Stackelberg game, in which the attacker, as the leader, crafts poison perturbation on the training data with the aim of decreasing the test accuracy, while the classifier, as the follower, optimizes the network parameters on the poisoned training dataset. We prove the existence of Stackelberg equilibria in both the normal and adversarial training settings. Furthermore, we propose a novel and effective approach, namely Game Unlearnable Example (GUE), in which we directly compute the Stackelberg game equilibrium using a first-order algorithm (Liu et al. 2022) and select an appropriate payoff function for the poison attacker. It is noteworthy that our GUE is not only applicable to standard natural training but can also extend to adversarial training.

Existing unlearnable example attacks typically require modifying the entire training dataset. However, in practical scenarios, the continuous influx of new clean data over time can render the unlearnable effect almost negligible. To mitigate this problem, when there is a continuous stream of new data, the user must go through the generation process for the whole dataset repeatedly, which is time-consuming. In order to overcome this problem, we use an autoencoder-like generator as the attacker, and show that a well-trained

[*] Corresponding author.

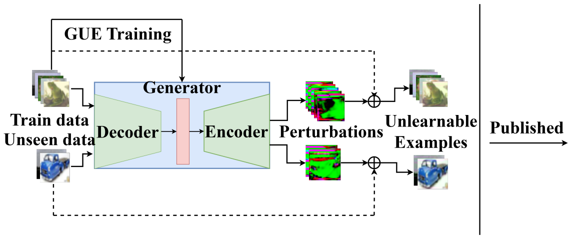


Figure 1: An illustration of GUE attack, where the trained generator can generalize to unseen data well.

attacker can easily generalize to unseen data well. The generator can be used to perform a simple forward propagation on the incoming data to generate the corresponding poison perturbation. Figure 1 illustrates the pipeline of the GUE attack.

In summary, our work has three main contributions:

- The unlearnable example attacks are formulated as a Stackelberg game and the existence of game equilibria under several useful settings are proved. It is further shown that the game equilibrium gives the most powerful poison attack in that the victim has the lowest test accuracy among all networks within the same hypothesis space when certain loss functions are used.
- We propose a new unlearnable example attack, called GUE, which, to the best of our knowledge, is the first poison method directly based on computing the game equilibria. Furthermore, by using an autoencoder-like generator as the attacker, poison perturbations for new data can be generated with simple forward propagation.
- Extensive experiments are used to demonstrate the effectiveness and generalizability of GUE in different scenarios. In particular, it is shown that GUE still works by using a relatively small percentage of the training data to train the generator.

## Related Works

Our GUE attack is intimately related to unlearnable example attacks, the Stackelberg game, and bi-level optimization problems. We first provide a brief overview of the developmental trajectory about unlearnable example attacks and explain their relationship with our approach.

Unlearnable example attacks are data poisoning methods with addictive bounded perturbations that prevent unauthorized model training. Error-minimizing noise (EM) (Huang et al. 2021) generates an imperceptible perturbation by solving a Min-Min optimization. Hypocritical perturbations (HYP) (Tao et al. 2021) follows a similar idea but directly uses a pre-trained model rather than the above Min-Min optimization. Error maximizing noise (TAP) (Fowl et al. 2021) generates adversarial examples as poisoned data. NTGA (Yuan and Wu 2021) generates poison perturbations in an ensemble of neural networks modeled with neural tangent kernels. There exist other methods that do not involve optimization. LSP (Yu et al. 2022) synthesizes linearly separable perturbations as attacks, and AR (Sandoval-Segura et al. 2022b) introduces a generic perturbation that can be utilized in various datasets and architectures.

However, the above methods cannot poison the adversarial trained model, and several novel methods were proposed to mitigate this problem. REM (Fu et al. 2022) generates a stronger perturbation by solving a Min-Min-Max three-level optimization problem. INF (Wen et al. 2023) generates a poison perturbation based on the induction of indiscriminate features between different classes to poison an adversarially trained model under a larger adversarial budget.

Lu, Kamath, and Yu (2022) formulated the traditional data poison attacks as a Stackelberg game and used total gradient descent to solve the game, while they considered adding a small portion of the poisoned data to the training set rather than modifying the training samples. And they did not consider the existence of game equilibrium.

The works most relevant to ours are DeepConfuse (Feng, Cai, and Zhou 2019) and ShortcutGen(van Vlijmen et al. 2022). Feng, Cai, and Zhou (2019) formulated the unlearnable examples attack as a bi-level optimization problem, and optimized the bi-level problem by decoupling the alternating update procedure. However, this method cannot guarantee the convergence theoretically and requires multiple rounds of optimization, which is very time-consuming. An autoencoder-like generator was also used as the attacker in DeepConfuse, but the generalizability of the generator was not discussed. van Vlijmen et al. (2022) used a static randomly initialized discriminator to train a poison generator, in order to encourage the generator to learn spurious shortcuts in the Min-Min framework (Huang et al. 2021). It is based on their conjecture that a randomly initialized discriminator provides a noisy mapping between images and labels and thus lacks theoretical guarantee.

## Unlearnable Example Game

We formulate the unlearnable example attack as a Stackelberg game, called *unlearnable example game*, denoted as $\mathcal{G}$. Then, we prove the existence of its equilibrium in the general case, which is extended to various attacking scenarios.

### Unified Game Framework

**Settings.** The game $\mathcal{G}$ has two participants: a poison attacker and a victim classifier. Let $\mathcal{S}$ be a finitely sampled clean training dataset from a data distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subset \mathbb{R}^n$ is the dataset and $\mathcal{Y} = [K] = \{i\}_{i=1}^{K}$ is the label set for $K \in \mathbb{N}_+$. The attacker modifies $x$ in each sample-label pair $(x, y) \in \mathcal{S}$ by an imperceptible perturbation $\mathcal{A}(x, y)$ such that

$$||\mathcal{A}(x, y)||_\infty \leq \epsilon.$$

In our method proposed in Algorithm 1, $\mathcal{A}$ will be specified as an encoder-decoder generator. Here, it can be simply regarded as a map $A(x, y) : \mathcal{S} \to \mathbb{R}^n$, where $n$ is the dimension of the data. The victim classifier only has access to the poisoned training set on which a classifier $f$ with parameters $\theta$ will be trained. The attacker's goal is to destroy the performance of $f_\theta$. We denote the loss function used in classifier training by $\mathcal{L}_c$ and the loss function used in the evaluation of attack performance by $\mathcal{L}_a$.

**Game model.** In the presence of a poisoning $\mathcal{A}$ introduced by the game leader, the payoff function of the victim is the empirical risk in the training process:

$$\mathcal{J}_c(\mathcal{A},\theta) := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{S}}\big[\mathcal{L}_c(x+\mathcal{A}(x,y),y;\theta)\big].$$

As the follower of the game, the victim should choose one of the *best responses*:

$$\theta^* \in \mathrm{BR}(\mathcal{A}) := \mathop{\arg\inf}_{\theta'} \mathcal{J}_c(\mathcal{A},\theta').$$

Due to the fact that neural network training usually reaches local optima, we consider the *$\eta$-approximately best responses*:

$$\theta^* \in \mathrm{BR}_\eta(\mathcal{A}) := \{\theta | \mathcal{J}_c(\mathcal{A},\theta) < \inf_{\theta'} \mathcal{J}_c(\mathcal{A},\theta') + \eta\}.$$

The payoff function of the attacker is the negative value of population risk:

$$\mathcal{J}_a(\mathcal{A},\theta) := \sup_{\theta\in\mathrm{BR}_\eta(\mathcal{A})} \{-\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\big[\mathcal{L}_a(x,y;\theta)\big]\}. \quad (1)$$

The attacker should choose a *Stackelberg strategy*

$$\mathcal{A}^* \in \mathop{\arg\inf}_{\mathcal{A}} \mathcal{J}_a(\mathcal{A},\theta). \quad (2)$$

Though the Stackelberg strategy is not ensured unique, the Stackelberg cost is unique. For a Stackelberg strategy $\mathcal{A}^*$, each $\theta^* \in \mathrm{BR}_\eta(\mathcal{A}^*)$ is an approximate optimal strategy for the victim with respect to $\mathcal{A}^*$. We call the action profile $(\mathcal{A}^*,\theta^*)$ a *Stackelberg equilibrium* without distinction.

## Equilibrium Existence

One of the primary concern for the game $\mathcal{G}$ is whether this game possesses an equilibrium. We will prove the existence of the equilibrium, consequently providing a well-defined target for learning. Next section will delve into the methods for solving this game.

Assume that the parameters $\theta$ of the victim classifier are restricted in a feasible strategy space $\Theta \subset \mathbb{R}^N$, where $N$ is the number of parameters. The following common assumptions are required for the existence of a Stackelberg equilibrium.

**Assumption 1.** *The strategy space $\Theta$ is compact, for instance $\Theta = [-E,E]^N$ for $E \in \mathbb{R}_+$.*

**Assumption 2.** *The loss functions $\mathcal{L}_a(x,y;\theta)$ and $\mathcal{L}_c(x,y;\theta)$ are continuous in $x$ and $\theta$.*

**Theorem 3.** *Under Assumptions 1 and 2, the unlearnable example game $\mathcal{G}$ has a Stackelberg equilibrium $(\mathcal{A}^*,\theta^*)$.*

*Proof sketch..* Let $\Gamma$ be the feasible space for the perturbations on the training set, that is, $\mathcal{A}(\mathcal{S}) \subset \Gamma$. With a slight abuse of notation, we denote $\mathcal{A} \in \Gamma$.

By Assumptions 1 and 2, $\mathcal{J}_c(\mathcal{A},\theta)$ is continuous, $\Theta$ and $\Gamma$ are compact. Then, for any $\mathcal{A}$, $\mathrm{BR}_\eta(\mathcal{A})$ is non-empty. Let $\mathcal{J}(\theta) := -\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\big[\mathcal{L}_a(x,y;\theta)\big]$. Thus the existence of Stackelberg equilibrium is equivalent to the existence of $\mathcal{A}^* \in \Gamma$ such that

$$\sup_{\theta\in\mathrm{BR}_\eta(\mathcal{A}^*)} \mathcal{J}(\theta) = \inf_{\mathcal{A}} \sup_{\theta\in\mathrm{BR}_\eta(\mathcal{A})} \mathcal{J}(\theta).$$

Since $\mathcal{J}(\theta)$ is continuous in $\Theta$, the marginal function

$$V(\mathcal{A}) := \sup_{\theta\in\mathrm{BR}_\eta(\mathcal{A})} \mathcal{J}(\theta)$$

is lower semi-continuous. Then $V(\mathcal{A})$ can attain the minimum as $\Gamma$ is compact, that is, there exists an $\mathcal{A}^*$ such that

$$V(\mathcal{A}^*) = \inf_{\mathcal{A}\in\Gamma} V(\mathcal{A}) = \sup_{\theta\in\mathrm{BR}_\eta(\mathcal{A}^*)} \mathcal{J}(\theta).$$

Since $\mathrm{BR}_\eta(\mathcal{A})$ is not empty, there exists a $\theta^* \in \mathrm{BR}_\eta(\mathcal{A}^*)$, and the theorem is proved. Details of the proof are given in subsection . □

## Various Attacking Scenarios

We have proved the existence of Stackelberg equilibrium with respect to general loss functions $\mathcal{L}_c$ and $\mathcal{L}_a$. Next, we will consider specific loss functions for different scenarios below. As long as $\mathcal{L}_c$ and $\mathcal{L}_a$ satisfy Assumption 2, the specified unlearnable example game has an equilibrium.

**Standard setting.** The most common attack scenario for classification tasks leverages cross-entropy loss as both $\mathcal{L}_c$ and $\mathcal{L}_a$. In this case, the attacker aims at reducing standard accuracy on the original data distribution while the victim trains classifier $f_\theta$ to improve standard accuracy on the poisoned training set. We have the following corollary as a direct consequence of Theorem 3.

**Corollary 4.** *Let $\mathcal{L}_c = \mathcal{L}_a = \mathcal{L}_{ce}$, then the unlearnable example game $\mathcal{G}$ has a Stackelberg equilibrium.*

In practice, solving this game also requires taking into account the effectiveness of the algorithm. Searching for a Stackelberg strategy of the attacker refers to minimizing its payoff function in Equation (1) involving $\mathcal{L}_{ce}$. However, the cross-entropy loss has no upper bound, and thus naturally lacks a criterion for convergence of such an optimization. Furthermore, it results in a gradient explosion, as will be discussed in the experimental section.

To tackle this issue, we instead leverage a surrogate loss function which is upper-bounded and is equivalent to cross-entropy loss for solving the game.

**Surrogate loss.** We propose a novel loss function for the attacker to measure the poison performance:

$$\mathcal{L}_{sur}(x,y;\theta) := -\max_{k\in[K]\backslash\{y\}} \mathcal{L}_{ce}(x,k;\theta). \quad (3)$$

Intuitively, maximizing $\mathcal{L}_{sur}$ means minimizing the cross-entropy loss of $x$ with respect to all other labels instead of $y$, leading to the misclassification of the model trained on the poisoned training set. When maximizing $\mathcal{L}_{ce}$, it often forms a deep valley on the true label in the label confidence distribution. On the other hand, when maximizing $\mathcal{L}_{sur}$, in addition to forming a deep valley on the true label, it also makes the confidence levels of other labels uniform.

The following properties of surrogate loss allow our method to effectively solve the game. Moreover, an equilibrium still exists for surrogate loss.

**Proposition 5.** *Assume that the data have $K$ classes.*

*1. $\mathcal{L}_{sur}$ is upper-bounded:*

$$\mathcal{L}_{sur}(x, y; \theta) \leq -\log(K-1).$$

*2. $\mathcal{L}_{ce}$ grows as $\mathcal{L}_{sur}$ grows:*

$$\mathcal{L}_{ce}(x, y; \theta) \geq -\log(1-(K-1)e^{\mathcal{L}_{sur}(x,y;\theta)}).$$

**Corollary 6.** *Let $\mathcal{L}_c = \mathcal{L}_{ce}$ and $\mathcal{L}_a = \mathcal{L}_{sur}$. We denote this game as $\mathcal{G}_{ue}$. Then this unlearnable example game $\mathcal{G}_{ue}$ has a Stackelberg equilibrium $(\mathcal{A}^*_{ue}, \theta^*_{ue})$.*

**Adversarial setting.** Recent studies (Tao et al. 2021) have suggested that adversarial training can mitigate the impact of unlearnable example attacks. Then poisoning approaches (Fu et al. 2022; Wen et al. 2023) were proposed to work for adversarially trained models.

Our unified game framework also includes the adversarial scenario in which the victim is aware of the potentially poisoned training set and decides to deploy adversarial training and the attacker still wants to prevent the victim from obtaining an available model.

We need only to modify the training loss $\mathcal{L}_c$ to adversarial settings. Here we consider the two widely-used adversarial losses:

- Adversarial Loss (Madry et al. 2017):

$$\mathcal{L}_{adv}(x, y; \theta) := \max_{||\mu||_\infty \leq \epsilon_d} \mathcal{L}_{ce}(x+\mu, y; \theta)$$

  where $\epsilon_d$ is the adversarial training radius.

- TRADES Loss (Zhang et al. 2019):

$$\begin{aligned}\mathcal{L}_{tra}(x, y; \theta) :=& \frac{1}{\lambda} \max_{||\mu||_\infty \leq \epsilon_d} \mathrm{KL}(f_\theta(x)||f_\theta(x+\mu)) \\ &+ \mathcal{L}_{ce}(x, y; \theta).\end{aligned}$$

The following lemma states that $L_{adv}$ and $L_{tra}$ satisfy Assumption 2.

**Lemma 7.** *$\mathcal{L}_{adv}(x, y; \theta)$ and $\mathcal{L}_{tra}(x, y; \theta)$ are continuous.*

As a consequence of Theorem 3, the equilibrium exists in the adversarial scenario.

**Corollary 8.** *Let $\mathcal{L}_c = \mathcal{L}_{adv}$ or $\mathcal{L}_{tra}$, and $\mathcal{L}_a = \mathcal{L}_{sur}$. We denote this game by $\mathcal{G}_{at}$. Then this unlearnable example game $\mathcal{G}_{at}$ has a Stackelberg equilibrium $(\mathcal{A}^*_{at}, \theta^*_{at})$.*

It implies that there exists a poisoning attack which can degrade the classification performance of a model that obtains approximately optimal parameters through adversarial training on the poisoned training set. We mainly consider the case with $\mathcal{L}_c = \mathcal{L}_{tra}$ in the following paper.

**Remark 9.** *Similar to (Gao, Liu, and Yu 2022), by using the loss function $\mathcal{L}_{cw}(f_\theta(x), y) = \max_{l \neq y}[f_\theta(x)]_l - [f_\theta(x)]_y$, we can prove that the corresponding game equilibrium exists and gives the most powerful poison attack in that the victim has the lowest test accuracy among all networks within the hypothesis space, when trained on poisoned training set.*

# Unlearnable Example Generator

In the preceding section, we theoretically established the unlearnable example game framework and demonstrated the existence of equilibrium in various scenarios.

In this section, by solving the game using a first-order algorithm, we propose a novel unlearnable example generator that can effectively generalize to future data added to the training set. By doing so, we aim to reduce the high dependency of unlearnable examples on the poisoning ratio to better align with the real-world need for protecting unauthorized data. Moreover, this game-theoretic approach can extend to the adversarial setting.

## Protection of Unauthorized Data

In the era of big data, countless personal data is uploaded to servers through various applications, published on the Internet, and can be accessed by others without restrictions. If these data are obtained without authorization by malicious individuals and used to train machine learning models, there is a risk that the user's personal privacy information being misused. As a means of protecting unauthorized data, unlearnable examples are used to preprocess upcoming data before its release, rendering models trained on the processed data unusable.

**Challenges.** However, there are some common issues that hinder the deployment and application of existing unlearnable example methods in the real world. Challenges include

- Dependency on extremely high poisoning ratio.
- Inefficiency of poison generation.
- Vulnerability to recovery approaches.

First, even a slight decrease in the poisoning ratio can result in a substantial decrease in unlearnability (Huang et al. 2021). When a poisoned training set is diluted by a few new clean samples, the protection of the data will be destroyed. It brings an essential obstacle to the application of unlearnable examples in the real world. Furthermore, most of the existing unlearnable example attacks are based on an enormous number of iterative propagation and back-propagation (Fowl et al. 2021). When it comes to redeploying poisoning to new data flow, the issue of inefficiency becomes more prominent. Last but not least, adversarial training can restore the usability of models trained on training sets polluted by poisons that have not been specifically designed for adversarial training (Tao et al. 2021). Class-wise perturbations are easier to recover compared to sample-wise perturbations (Sandoval-Segura et al. 2022a).

## Generalizable Poison Generator

To mitigate the aforementioned problems, we propose a novel approach that differs from existing methods. Instead of empirically optimizing perturbations to create shortcuts for learning, we suggest using a game-theoretic framework to train an unlearnable example generator that can generalize to unseen data effectively.

In detail, our approach involves having the attacker in the game utilize a poison generator $g_\omega$ based on an encoder-decoder with parameters $\omega$. That is, $\mathcal{A}(x, y) = g_\omega(x)$ which outputs sample-wise perturbations. The activation for the final layer is $\epsilon \cdot \tanh(\cdot)$ to control the perturbation within the poison radius $\epsilon$, i.e. $||g_w(x)||_\infty \leq \epsilon$.

Now the classifier aims at minimizing the payoff function:

$$\mathcal{J}_c(w, \theta) = \mathop{\mathbb{E}}_{(x,y)\sim S} \big[\mathcal{L}_c(x + g_w(x), y; \theta)\big]$$

by choosing parameters

$$\theta^* \in \mathrm{BR}_\eta(w) = \{\theta | \mathcal{J}_c(w, \theta) < \inf_{\theta'} \mathcal{J}_c(w, \theta') + \eta\}.$$

The attacker chooses generator parameters $w^*$ to minimize the payoff function:

$$\mathcal{J}_a(\omega, \theta) := \sup_{\theta \in \mathrm{BR}_\eta(\omega)} \{- \mathop{\mathbb{E}}_{(x,y)\sim \mathcal{S}} \big[\mathcal{L}_a(x, y; \theta)\big]\}. \quad (4)$$

To make it feasible in practical computation, here we consider the loss on clean training set $\mathcal{S}$ instead of the data distribution $\mathcal{D}$ in Equation (1).

Once a generator $g_\omega$ is well trained on a given training set, it costs only forward propagation to generate poisons for images in the training set and potentially for future images that may be added to the training set. Due to the flexible selection of $\mathcal{L}_c$ and $\mathcal{L}_a$, the generator is able to deal with different scenarios, including standard and adversarial settings.

## Compute the Game Equilibrium

In fact, a Stackelburg game can be simplified to a bi-level optimization problem:

$$\begin{aligned} &\min\nolimits_{w,\theta} l(w, \theta) \\ \text{s.t.} \quad \theta \quad &\in \arg\min\nolimits_{\theta'} h(w, \theta'). \end{aligned} \quad (5)$$

In the context of an unlearnable example game, we have $l = \mathcal{J}_a$ and $h = \mathcal{J}_c$. Directly solving a bi-level optimization problem requires the calculation of the second-order Hessian matrices, which makes it impractical for common machine learning tasks with high-dimensional inputs and parameters.

We leverage previously proposed BOME (Liu et al. 2022) and dynamic barrier gradient descent algorithm (Gong, Liu, and Liu 2021) to efficiently train an unlearnable example generator, the detailed algorithm is presented in algorithm 1. BOME is a recently developed first-order bi-level optimization algorithm that employs a value function approach to transform the bi-level problem into a single-level constrained optimization problem. Then the single-level problem is solved by the dynamic barrier gradient descent algorithm, which has the notable advantage of not requiring the lower-level problem to possess a unique solution.

**BOME.** Assume $h(w, \cdot)$ can attain a minimum for each $w$. Problem (5) is equivalent to the following constrained optimization (even for nonconvex $h$):

$$\begin{aligned} &\min_{w,\theta} l(w, \theta) \\ \text{s.t.} \quad &q(w, \theta) := h(w, \theta) - h(w, \theta^*(w)) \leq 0, \end{aligned}$$

where $h(w, \theta^*(w)) = \min_\theta h(w, \theta)$. In practice, we approximate $\theta^*(w)$ by $\theta^T(w)$ which is the $T$ step gradient descent of $h(w, \cdot)$ over $\theta$ for some step size $\alpha > 0$ :

$$\theta^{t+1}(w) = \theta^t(w) - \alpha \nabla_\theta h(w, \theta^t(w)). \quad (6)$$

Then we obtain an estimate of $q(w, \theta)$:

$$\widehat{q}(w, \theta) = h(w, \theta) - h(w, \theta^T(w)).$$

Algorithm 1: Training of unlearnable example generator

```
Input: Training set $\mathcal{S}$, inner step $T$, inner and outer step size $\alpha, \beta$, batch size $b$, train epochs $e$.
Output: Learned poison Generator $g_w$
Initialize classifier $f_\theta$ and attacker $g_w$
for $k = 1$ to $e$ do
    Sample a mini-batch $\{(x_i, y_i)\}_{i=1}^b \sim \mathcal{S}$
    Compute $\theta^T(w_k)$ by $T$ steps gradient descent on $J_c(w_k, \cdot)$ starting from $\theta_k$(like Eq.(6))
    set $\widehat{q}(w_k, \theta_k) = J_c(w_k, \theta_k) - J_c(w_k, \theta^T(w_k))$
    Update $(w, \theta)$:
        $\theta_{k+1} \leftarrow \theta_k - \beta(\nabla J_a(\theta_k) + \lambda_k \nabla_{\theta_k} \widehat{q}(w_k, \theta_k))$
        $w_{k+1} \leftarrow w_k - \beta(\lambda_k \nabla_{w_k} \widehat{q}(w_k, \theta_k))$
    where $\lambda_k = \max(\frac{\phi_k - \langle \nabla J(\theta_k), \nabla_{\theta_k} \widehat{q}(w_k, \theta_k) \rangle}{||\nabla \hat{q}(w_k, \theta_k)||^2}, 0)$ and $\phi_k = \rho||\nabla \widehat{q}(w_k, \theta_k)||^2$.
    (Set $\rho = 1.5$ and $T = 10$ as default)
end for
return $g_w$
```

**Dynamic barrier gradient descent.** The method is to iteratively update $(w, \theta)$ with step size $\beta > 0$ to reduce $l$ while controlling the decrease of $q$:

$$(w_{k+1}, \theta_{k+1}) \leftarrow (w_k, \theta_k) - \beta(\nabla l(w_k, \theta_k) + \lambda_k \nabla \widehat{q}(w_k, \theta_k))$$

where

$$\lambda_k = \max(\frac{\phi_k - \langle \nabla l(w_k, \theta_k), \nabla \widehat{q}(w_k, \theta_k) \rangle}{||\nabla \widehat{q}(w_k, \theta_k)||^2}, 0)$$

and $\phi_k = \rho||\nabla \widehat{q}(w_k, \theta_k)||^2$ by default with a hyperparameter $\rho > 0$.

# Experiments

In this section, we conduct comprehensive experiments to validate the effectiveness of GUE on popular benchmark datasets. Here, we consider not only standard training but also adversarial training.

## Experiment Settup

We mainly conduct experiments on image classification datasets: CIFAR-10, CIFAR-100 which have been commonly used in the poisoning literature. We use ResNet-18 (He et al. 2016) as classifier $f(\theta)$ and U-Net (Ronneberger, Fischer, and Brox 2015) as poison generator $g_w$ during training. If not explicitly mentioned, we focus on the reasonable setting with poison radius $\epsilon = 8/255$.

We use the test accuracy on clean test set to assess the effectiveness of unlearnable example attacks; the lower accuracy implies that the attack is stronger to prevent the model from learning information from the poisoned training dataset.

## Standard Training

In this subsection, we conduct experiments under standard training setting to verify the effectiveness of our GUE attack by directly solving the game $\mathcal{G}_{ue}$ defined in Corollary 6.

| Poison method | CIFAR10 | CIFAR100 |
|---|---|---|
| None(Clean) | 92.36 | 70.59 |
| EM | 10.16 | 1.90 |
| TAP | 20.28 | 15.40 |
| DeepConfuse | 22.74 | 25.73 |
| ShortcutGen | 24.42 | 8.62 |
| GUE | 13.25 | 8.35 |

Table 1: Test accuracy of ResNet-18 trained on poisoned data from different unlearnable example attacks on CIFAR.

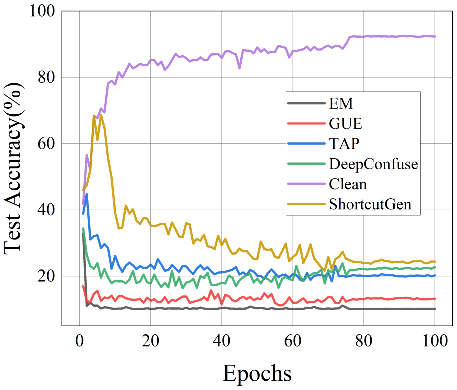


Figure 2: Test accuracy curves of ResNet-18 trained on poisoned data from different unlearnable example attacks on CIFAR-10.

We train 50 epochs to generate GUE with algorithm 1, using SGD optimizer with learning rate 0.01 for classifier $f_\theta$ and SGD optimizer with learning rate 0.1 for attacker $g_w$. And use Adam (Kingma and Ba 2014) with learning rate 0.001 for the inner T steps approximation. And for evaluation of unlearnable examples, we train a model on poisoned dataset for 100 epochs using SGD optimizer with an initial learning rate of 0.01 that is decayed by a factor of 0.1 at the 75-th and 90-th training epochs. The optimizer is set with momentum 0.9 and weight decay $5 \times 10^{-4}$.

**Compare to different unlearnable example attacks.** We first evaluate the performance of our GUE and different state-of-the-art poisoning methods under standard training, including DeepConfuse (Feng, Cai, and Zhou 2019), Unlearnable Examples(EM) (Huang et al. 2021), Targeted Adversarial Poisoning(TAP) (Fowl et al. 2021) and ShortcutGen (van Vlijmen et al. 2022). As shown in Table 1, our GUE can substantially decrease the clean test accuracy and performs better than all methods except EM. Furthermore, as we can see from Figure 2, our GUE remains unlearnably effective throughout all training epochs, while all other attacks exhibit an accuracy peak at the beginning of the training.

**Different percentages of data to train $g_w$ and generalizability.** In practice, it is not always possible that the attacker has access to the full training dataset, the victim classifier may collect more data to train a model. It has been observed that other unlearnable example attacks such as EM or TAP would fail to poison the model when there is a small proportion of clean data, such as 10% from the training dataset. Thus, when there are more clean data, these attacks require regenerating corresponding unlearnable examples on the entire training data after adding new clean data. This motivates us to examine the generalizability of the poison generator $g_w$ trained on only a proportion of randomly selected training examples, and when there are more clean data, we can use $g_w$ to generate unlearnable examples for these data.

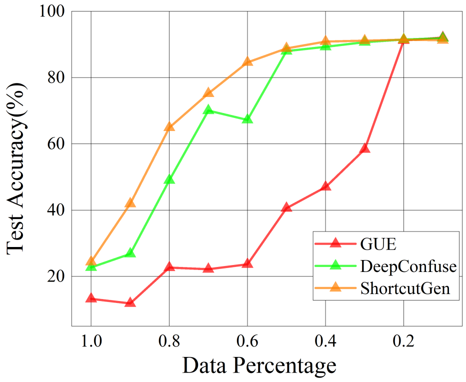


Figure 3: Test accuracy of ResNet-18 trained on poisoned CIFAR-10, where the poison generator is trained on different percentage of training data.

We use different percentages of training data to train the poison generator, then use the corresponding generator to generate unlearnable examples on the entire training dataset. The results are shown in Figures 3 and 4. We can observe that our GUE's generalizability is much better than DeepConfuse and ShortcutGen. Specifically, our GUE is still effective in degrading the test accuracy to about 20% when we only use 60% of the training data in CIFAR10. This indicates that the GUE-trained poison generator can generalize to unseen data well.

**Transferability on different model architectures.** We generate a GUE with fixed classifier structure as ResNet-18. For this reason, a natural question to ask is whether the poison generator trained with one model architecture is still effective when the victim classifier adopts a different architecture. Table 2 shows that the poisoning effects of our GUE optimized against a ResNet-18 classifier can be transferred to other model architectures. (To save space, we denote DeepConfuse and ShorcutGen as DC and SG.) Specifically, we can observe that the transferability of our GUE is more stable than that of DeepConfuse and ShortcutGen. Our GUE is capable of reducing the test accuracy to $12.97\% \sim 14.68\%$ across various model architectures.

**Effects of defenses against GUE.** We have demonstrated the efficacy of our GUE in the setting where the victim classifier only uses standard training. However, the victim classi-

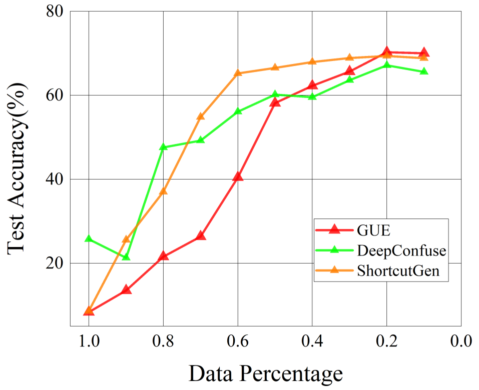


Figure 4: Test accuracy of ResNet-18 trained on poisoned CIFAR-100, where the poison generator is trained on different percentage of train data.

| Poison method | Clean | GUE | DC | SG |
|---|---|---|---|---|
| ResNet-18 | 92.36 | 13.25 | 22.74 | 24.42 |
| VGG16 | 91.18 | 13.72 | 25.35 | 12.32 |
| ResNet-50 | 92.14 | 12.97 | 20.56 | 17.35 |
| DenseNet-121 | 92.10 | 13.71 | 21.44 | 16.59 |
| WRN-28-10 | 92.86 | 14.68 | 29.02 | 25.09 |

Table 2: Transferability of GUE from ResNet-18 to other model architectures on CIFAR-10.

fier could employ several defenses proposed against poison attacks. Thus, we test the effectiveness of several popular defenses against our GUE.

Here, we firstly study whether data augmentations can mitigate our GUE. Following previous work, we test a diverse set of data augmentations, including Mixup (Zhang et al. 2017), Cutout (DeVries and Taylor 2017), and Cutmix (Yun et al. 2019). Since adversarial training is widely considered as an effective defense against unlearnable examples attacks, we also test adversarial training with adversarial radius $\epsilon_d = 2/255$ and $4/255$.

Table 3 shows that all data augmentation methods fail to mitigate our GUE. Adversarial training can counteract the poison effect, but our GUE is still effective when the adversarial radius is small $\epsilon_d = 2/255$. It is more effective than other existing methods, such as EM, TAP, both of which achieve test accuracy exceeding $70\%$.

## Adversarial Training

We find that our GUE cannot defend against adversarial training for a large adversarial radius in the preceding subsection. To solve this problem, we consider solving the game $\mathcal{G}_{at}$ in which the victim classifier also adopts adversarial training. And we denote the attacks as GUE-AT($\epsilon$, $\epsilon_d$) where $\epsilon$ is the poison radius and $\epsilon_d$ is the adversarial radius in the game $\mathcal{G}_{at}$.

We train 150 epochs to generate GUE-AT with algorithm 1, the optimizers are set as the standard training. We set $\lambda = 1$ in trade-off loss, the inner maximization (that is, generation of adversarial examples) of the adversarial training is solved by 10-step PGD with a step size of $\epsilon_d/4$. For adversarial training evaluation experiments, all training settings are the same as the standard training except that the initial learning rate is changed to 0.1.

| Defenses | Clean test accuracy |
|---|---|
| Baseline(Clean) | 92.36 |
| Mixup | 11.83 |
| Cutout | 14.71 |
| Cutmix | 19.84 |
| Adv training($\epsilon_d = 2/255$) | 22.55 |
| Adv training($\epsilon_d = 4/255$) | 76.96 |

Table 3: Evaluating GUE against different defenses.

| Poison method | $\epsilon_d = 0$ | $\epsilon_d = 2/255$ | $\epsilon_d = 4/255$ |
|---|---|---|---|
| None(Clean) | 92.36 | 92.41 | 89.82 |
| INF | 88.78 | 80.91 | 77.96 |
| REM(8-2) | 21.14 | 29.75 | 74.20 |
| GUE-AT(8-2) | 16.11 | 22.27 | 60.11 |
| REM(8-4) | 26.15 | 30.90 | 44.91 |
| GUE-AT(8-4) | 17.86 | 21.19 | 52.79 |

Table 4: Test accuracy of ResNet-18 adversarially trained with radius $\epsilon_d = 0, 2/255, 4/255$. INF is from (Wen et al. 2023).

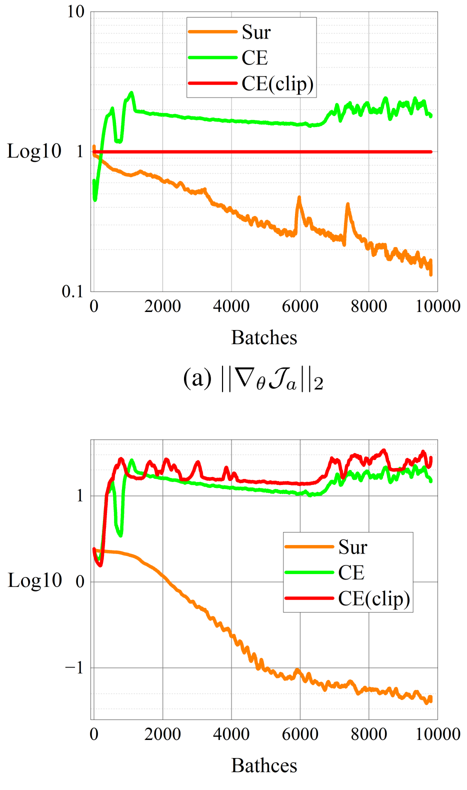
**Different adversarial training perturbation radii.** We train models using different adversarial training perturbation radii on these unlearnable examples. The adversarial training radius $\epsilon_d$ ranges from $0/255$ to $4/255$. It is important to note that when $\epsilon_d = 0$, the models are trained using the standard training method. Table 4 reports the accuracies of the trained models under different unlearnable example attacks.

We can see that our GUE-AT outperforms the SOTA method REM over $7.48\% - 9.71\%$ when adversarial radius $\epsilon_d = 2/255$. For a larger adversarial radius, our GUE-AT is also effective. In total, these experiments show that our method can be applied to different adversarial training perturbation radii while keeping a significant poison effect.

## Ablation Studies

In this subsection, we conduct experiments to provide an empirical understanding of the proposed surrogate loss. As discussed when introducing the surrogate loss, the cross-entropy loss has no upper bound and does not provide a criterion for convergence. Therefore, we propose the surrogate loss $\mathcal{L}_{sur}$, which is upper bounded and equivalent to $\mathcal{L}_{ce}$. Empirically, we compute the game equilibrium with $L_a = \mathcal{L}_{ce}$ and $\mathcal{L}_a = \mathcal{L}_{sur}$, and observe the gradient of $\mathcal{J}_a$ and the poisoned loss $\mathcal{J}_c(w, \theta)$ in each batch of data during training.

Figure 5 (a) shows that $||\nabla_\theta \mathcal{J}_a||_2$ with $\mathcal{L}_{ce}$ is approximately two orders of magnitude larger than with $\mathcal{L}_{sur}$, and



(a) $||\nabla_\theta \mathcal{J}_a||_2$

(b) Poisoned loss $\mathcal{J}_c(w,\theta)$

Figure 5: The training curves of $||\nabla_\theta \mathcal{J}_a||_2$ and $\mathcal{J}_c(w,\theta)$ when we use different loss function in $\mathcal{J}_a$ to compute the equilibrium: $\mathcal{L}_{ce}, \mathcal{L}_{sur}$ and $\mathcal{L}_{ce}$ with gradient clipping. We take $\log_{10}$ of all values for better visualization.

diverges. A straightforward method to overcome the explosion of gradients is gradient clipping. Hence, we use gradient clipping to control the gradient norm $||\nabla_\theta \mathcal{J}_a||_2 = 10$. However, in Figure 5(b) it can be seen that, as a necessary condition for convergence, poisoned loss $\mathcal{J}_c(w,\theta)$ cannot converge when we use $\mathcal{L}_{ce}$ or $\mathcal{L}_{ce}$ with gradient clipping. Only our proposed loss has guarantee of convergence. This verifies the necessity of our proposed loss $\mathcal{L}_{sur}$ to compute the equilibrium.

## Proofs

In this section, we give proofs of the theoretical results in the paper and give a complete statement of Remark 9.

### Proof of Theorem 3

We first introduce some lemmas about the semicontinuity of set-valued maps and then give the complete proof of the existence of the Stackelberg equilibrium of unlearnable examples game.

Similar to Lemma 3.2 in (Gao, Liu, and Yu 2022), we can prove the following properties of payoff functions.

**Lemma 10.** *The payoff functions $J_c(\mathcal{A},\theta)$ and $J_a(\theta)$ are continuous.*

**Lemma 11.** *Let $X, Y$ be two Hausdorff spaces and $X$ compact, $F : X \times Y \to \mathbb{R}$ continuous. Then $\inf\limits_{x\in X} F(x,y)$ is continuous on $Y$.*

*Proof.* It is equivalent to prove that $\forall \epsilon > 0, \exists \delta > 0$, for any $||y_1 - y_2|| \leq \delta$, we have $|\inf_{x\in X} F(x,y_1) - \inf_{x\in X} F(x,y_2)| \leq \epsilon$. Since $F(x,y)$ is continuous and $X$ is compact, there exists an $x^*$ for any $y$ such that $F(x^*,y) = \inf_{x\in X} F(x,y)$, so there exist $x_1^*, x_2^*$ correlated with $y_1, y_2$, respectively.

Since $F(x,y)$ is continuous, $\forall \epsilon > 0, \exists \delta(x) > 0$, such that when $||y_1 - y_2|| \leq \delta(x)$, we have $|F(x,y_1) - F(x,y_2)| \leq \epsilon$. Hence $\forall \epsilon > 0$, there exits a $\delta(x_2^*)$, such that

$$\inf_{x\in X} F(x,y_1) - \inf_{x\in X} F(x,y_2) \leq F(x_2^*,y_1) - F(x_2^*,y_2) \leq \epsilon$$

and there exits a $\delta(x_1^*)$ such that

$$\begin{aligned} &\inf_{x\in X} F(x,y_1) - \inf_{x\in X} F(x,y_2) \\ &\geq F(x_1^*,y_1) - F(x_1^*,y_2) \geq -\epsilon. \end{aligned}$$

Let $\delta = \min\{\delta(x_1^*), \delta(x_2^*)\}$. Then $\forall \epsilon > 0, \exists \delta > 0$, such that for any $||y_1 - y_2|| \leq \delta$, $|\inf_{x\in X} F(x,y_1) - \inf_{x\in X} F(x,y_2)| \leq \epsilon$ holds, that is, $\inf\limits_{x\in X} F(x,y)$ is continuous on $Y$. □

**Lemma 12.** *(Aubin and Ekeland 1984) Let $X, Y$ be two Hausdorff spaces, $G$ a set-valued map from $Y$ to $X$, and $W$ a real-valued function defined on $X$. Suppose that $W$ is lower semicontinuous, $G$ is lower semicontinuous on $Y$. Then the marginal function $V(y) = \sup\limits_{x\in G(y)} W(x)$ is lower semicontinuous on $Y$.*

**Lemma 13.** *Let $X, Y$ be two compact subsets in a metric space, $F : X \times Y \to \mathbb{R}$ a continuous function, and $G(y) = \{x \in X : F(x,y) < \inf\limits_{z\in X} F(z,y) + \eta\}$ where $\eta > 0$ is a constant. Then $G(y) : Y \to X$ is lower semicontinuous.*

*Proof.* The fact that $G(y)$ is lower semicontinuous is equivalent to proving for any $y \in Y$, any sequence $\{y^m\}_{m=1}^\infty$ convergent to $y$, and $\forall x^0 \in G(y)$, there exists a sequence $\{x^m\}_{m=1}^\infty$ converging to $x^0$ such that $x^m \in G(y^m)$ holds for sufficiently large $m$.

Since $x^0 \in G(y)$, we have $F(x^0,y) < \inf\limits_x F(x,y) + \eta$, and $X$ is compact, so there exists a sequence $\{x^m\}_{m=1}^\infty$ that converges to $x^0$. Because $x^m \to x^0$ and $y^m \to y$, and $F(x,y)$ are continuous, we have $\lim\limits_{m\to\infty} F(x^m,y^m) = F(x^0,y)$. According to the Lemma 11, since $X$ is compact, $\inf\limits_{x\in X} F(x,y)$ is continuous on $Y$. Then

$$\begin{aligned} \lim_{m\to\infty} F(x^m,y^m) &= F(x^0,y) < \inf_x F(x,y) + \eta \\ &= \lim_{m\to\infty} \inf_{x\in X} F(x,y^m) + \eta. \end{aligned}$$

Then for a sufficiently large $m$, we have $F(x^m,y^m) < \inf\limits_{x\in X} F(x,y^m) + \eta$, i.e. $x^m \in G(y^m)$ for a sufficiently large $m$. Therefore, $G(y)$ is lower semi-continuous on $y$. □

Now we can prove the main theorem:

**Theorem 14.** *(Restate of Theorem 3) The unlearnable example game $\mathcal{G}$ has a Stackelberg equilibrium.*

*Proof.* By Lemma 10, we have $J_c(\mathcal{A}, \theta)$ is continuous, and by Assumption 1, $\Theta$ and $\Gamma$ are compact. Then, for any $\mathcal{A}$, $R_\eta(\mathcal{A})$ is nonempty. Let $J(\theta) := -\underset{(x,y)\sim\mathcal{D}}{\mathbb{E}}\big[\mathcal{L}_a(x, y; \theta)\big]$. Then the existence of a Stackelberg equilibrium is equivalent to $\exists \mathcal{A}^* \in \Gamma$ such that:

$$\sup_{\theta \in R_\eta(\mathcal{A}^*)} J(\theta) = \inf_{\mathcal{A}} \sup_{\theta \in R_\eta(\mathcal{A})} J(\theta).$$

By Lemma 13, we have that $R_\eta(\cdot)$ is lower semicontinuous. And since $J(\theta)$ is lower semi-continuous on $\Theta$, by Lemma 12, $V(\mathcal{A}) = \sup_{\theta \in R_\eta(\mathcal{A})} J(\theta)$ is lower semicontinuous. Then $V(\mathcal{A})$ can reach the minimum as $\Gamma$ is compact. Then $\exists \mathcal{A}^*$ such that

$$V(\mathcal{A}^*) = \inf_{\mathcal{A} \in \Gamma} V(\mathcal{A}) = \sup_{\theta \in R_\eta(\mathcal{A}^*)} J(\theta).$$

Since $R_\eta(\mathcal{A})$ is a nonempty subset, there exists a $\theta^* \in R_\eta(\mathcal{A}^*)$. Thus, we have proved that there exist $\mathcal{A}^* \in \Gamma$ and $\theta^* \in R_\eta(\mathcal{A}^*)$ such that $(\mathcal{A}^*, \theta^*)$ is a Stacklberg equilibrium. □

## Proof of Lemma 7

**Lemma 15.** *(Restatement of Lemma 7) The adversarial loss function $L_{adv}(x, y; f_\theta) = \max_{||\mu||_\infty \le \epsilon_d} L_{ce}(f_\theta(x+\mu), y)$ is continuous.*

*Proof.* We show that $L_{adv}(x, y; f_\theta)$ is continuous at $\theta$, and the continuity for $x$ is similar. It is clear that $L_{ce}(f_\theta(x+\mu), y)$ is continuous on $\theta$. Therefore, for any $\gamma > 0$, there exists a $\Delta(x) > 0$ (depending on $x$), such that for any $||\theta_1 - \theta_2|| \le \Delta(x)$ we have $|L_{ce}(x, \theta_1) - L_{ce}(x, \theta_2)| \le \gamma$.

Let $L_{adv}(x_1, \theta_1) = \max_{||\mu||_\infty \le \epsilon_d} L_{ce}(f_{\theta_1}(x_1 + \mu), y)$ and $L_{adv}(x_2, \theta_2) = \max_{||\mu||_\infty \le \epsilon_d} L_{ce}(f_{\theta_2}(x_2 + \mu), y)$. Then for $||\theta_1 - \theta_2|| \le \Delta(x_2)$:

$$\begin{aligned} &L_{ce}(x_1, \theta_1) - L_{ce}(x_2, \theta_2) \\ &\ge L_{ce}(x_2, \theta_1) - L_{ce}(x_2, \theta_2) \ge -\gamma \end{aligned}$$

and for $||\theta_1 - \theta_2|| \le \Delta(x_1)$:

$$\begin{aligned} &L_{ce}(x_1, \theta_1) - L_{ce}(x_2, \theta_2) \\ &\le L_{ce}(x_1, \theta_1) - L_{ce}(x_1, \theta_2) \le \gamma. \end{aligned}$$

Thus, for any $\gamma > 0$, there exists a constant $\Delta = \min\{\Delta(x_1), \Delta(x_2)\} > 0$, such that for any $||\theta_1 - \theta_2|| \le \Delta$ we have $|\max_{x'} L_{ce}(x', \theta_1) - \max_{x'} L_{ce}(x', \theta_2)| \le \gamma$, which means $L_{adv}(x, y; f_\theta)$ is continuous at $\theta$. □

Similarly, we can prove that the trade-off adversarial loss is continuous:

**Lemma 16.** *The TRADES Loss function $L_{tra}(x, y; f_\theta) = L_{ce}(f_\theta(x), y) + \frac{1}{\lambda} \max_{||\mu||_\infty \le \epsilon_d} KL(f_\theta(x)||f_\theta(x+\mu))$ is continuous.*

## Discrete Loss for Lowest Accuracy

As we stated in Remark 9, in order to exactly degrade the victim classifier's test accuracy, we consider a discrete loss function similar to (Gao, Liu, and Yu 2022):

$$\mathcal{L}_{acc}(f_\theta(x), y) = \begin{cases} 0 & \mathcal{L}_{cw}(f_\theta(x), y) \ge 0 \\ -1 & \mathcal{L}_{cw}(f_\theta(x), y) < 0 \end{cases}$$

where $\mathcal{L}_{cw}(f_\theta(x), y) = \max_{l \ne y}[f_\theta(x)]_l - [f_\theta(x)]_y$ is the Carlini-Wagner loss (Carlini and Wagner 2017). Then the negative value of population risk: $-\underset{(x,y)\sim\mathcal{D}}{\mathbb{E}}\big[\mathcal{L}_{acc}(x, y; \theta)\big]$ is exactly the classification accuracy.

Though here $\mathcal{L}_{acc}$ is not continuous, we also can extend our existence Theorem 3 to this case:

**Corollary 17.** *Let $L_a = L_{acc}$, $L_c = L_{ce}$, we denote this Stackelberg game as $\mathcal{G}_{acc}$. Then the unlearnable example game $\mathcal{G}_{acc}$ has a Stackelberg equilibrium $(\mathcal{A}^*_{acc}, \theta^*_{acc})$ .*

*Proof.* Since $L_{cw}(f_\theta(x), y)$ is continuous at $\theta$, it is easy to verify that $-L_{acc}(f_\theta(x), y)$ is lower semi-continuous on $\theta$. Furthermore $J(\theta)$ is lower semi-continuous on $\theta$. And the remaining proof is similar to before. □

# Conclusion

While unlearnable example attacks have been well studied in various formulations, a unified theoretical framework remains unexplored. In this paper, we formulate the unlearnable example attack as a Stackelberg game that encompasses a range of scenarios. We propose a novel attack GUE by directly computing the Stackelberg equilibrium using a first-order optimization method and by using a better loss function. Our approach possesses a stronger theoretical intuition, whereas other methods are primarily based on empirical practices. Furthermore, we show that the game equilibrium gives the most powerful poison attack in the sense that the victim neural network has the lowest test accuracy among all networks within the same hypothesis space.

Extensive experiments demonstrate the effectiveness of GUE in different scenarios, in particular, the generalizability of poison generator trained with low percentage of training data and its effectiveness against adversarial training.

**Limitations** Our algorithm cannot converge well when the adversarial radius is larger, hence not effective to poison adversarial trained model with more aggressive budgets. Future research is needed to find more effective algorithms to solve the unlearnable example games.

# Acknowledgements

This work is partially supported by the NSFC grant No.11688101 and NKRDP grant No.2018YFA0306702. The authors thank anonymous referees for their valuable comments.

# References

Aubin, J.-P.; and Ekeland, I. 1984. *Applied Nonlinear Analysis*. Wiley, New York.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. Ieee.

DeVries, T.; and Taylor, G. W. 2017. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.

Feng, J.; Cai, Q.-Z.; and Zhou, Z.-H. 2019. Learning to confuse: generating training time adversarial data with autoencoder. *Advances in Neural Information Processing Systems*, 32.

Fowl, L.; Goldblum, M.; Chiang, P.-y.; Geiping, J.; Czaja, W.; and Goldstein, T. 2021. Adversarial Examples Make Strong Poisons. In *Advances in Neural Information Processing Systems*, volume 34, 30339–30351.

Fu, S.; He, F.; Liu, Y.; Shen, L.; and Tao, D. 2022. Robust unlearnable examples: Protecting data against adversarial learning. *arXiv preprint arXiv:2203.14533*.

Gao, X.-S.; Liu, S.; and Yu, L. 2022. Achieving optimal adversarial accuracy for adversarial deep learning using Stackelberg games. *Acta Math Sci*, 2399–2418.

Gong, C.; Liu, X.; and Liu, Q. 2021. Automatic and harmless regularization with constrained and lexicographic optimization: A dynamic barrier approach. *Advances in Neural Information Processing Systems*, 34: 29630–29642.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Huang, H.; Ma, X.; Erfani, S. M.; Bailey, J.; and Wang, Y. 2021. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Liu, B.; Ye, M.; Wright, S.; Stone, P.; and Liu, Q. 2022. Bome! bilevel optimization made easy: A simple first-order approach. *Advances in Neural Information Processing Systems*, 35: 17248–17262.

Lu, Y.; Kamath, G.; and Yu, Y. 2022. Indiscriminate Data Poisoning Attacks on Neural Networks. *arXiv preprint arXiv:2204.09092*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Proceedings, Part III 18*, 234–241. Springer.

Sandoval-Segura, P.; Singla, V.; Fowl, L.; Geiping, J.; Goldblum, M.; Jacobs, D.; and Goldstein, T. 2022a. Poisons that are learned faster are more effective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 198–205.

Sandoval-Segura, P.; Singla, V.; Geiping, J.; Goldblum, M.; Goldstein, T.; and Jacobs, D. 2022b. Autoregressive perturbations for data poisoning. *Advances in Neural Information Processing Systems*, 35: 27374–27386.

Tao, L.; Feng, L.; Yi, J.; Huang, S.-J.; and Chen, S. 2021. Better safe than sorry: Preventing delusive adversaries with adversarial training. *Advances in Neural Information Processing Systems*, 34: 16209–16225.

van Vlijmen, D.; Kolmus, A.; Liu, Z.; Zhao, Z.; and Larson, M. 2022. Generative Poisoning Using Random Discriminators. *arXiv preprint arXiv:2211.01086*.

Wen, R.; Zhao, Z.; Liu, Z.; Backes, M.; Wang, T.; and Zhang, Y. 2023. Is Adversarial Training Really a Silver Bullet for Mitigating Data Poisoning? In *The Eleventh International Conference on Learning Representations*.

Yu, D.; Zhang, H.; Chen, W.; Yin, J.; and Liu, T.-Y. 2022. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2367–2376.

Yuan, C.-H.; and Wu, S.-H. 2021. Neural tangent generalization attacks. In *International Conference on Machine Learning*, 12230–12240. PMLR.

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, 6023–6032.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; El Ghaoui, L.; and Jordan, M. 2019. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, 7472–7482. PMLR.