Towards Effective Data Poisoning for Imbalanced Classification

Snigdha Sushil Mishra [1] **Hao He** [2] **Hao Wang** [1]

Abstract

Targeted Clean-label Data Poisoning Attacks (TCPDA) aim to manipulate training samples in a label-consistent manner to gain malicious control over targeted samples' output during deployment. A prominent class of TCDPA methods, gradient-matching based data-poisoning methods, utilize a small subset of training class samples to match the poisoned gradient of a target sample. However, their effectiveness is limited when attacking imbalanced datasets because of gradient mis-match due to training time data balancing techniques like *Re-weighting* and *Re-sampling*. In this paper, we propose two modifications that eliminate this gradient-mismatch and thereby enhance the efficacy of gradient-matching-based TCDPA on imbalanced datasets. Our methods achieve notable improvements of up to 32% (*Re-sampling*) and 51% (*Re-weighting*) in terms of Attack Effect Success Rate on MNIST and CIFAR10.

1. Introduction

Machine Learning (ML) models have made significant advancements, demonstrating improved performance across various real-world tasks in recent years (MetaAI). Consequently, the deployment of ML models in industrial applications has also witnessed a surge (Paleyes et al., 2022). Therefore, it is crucial to develop toolkits for studying threats to trained neural network models. *Targeted Clean-label Data Poisoning Attacks (TCDPA)* represent such threats, employing imperceptible modifications in clean-label training data to poison machine learning models like deep neural networks, specifically targeting a set of test samples. Successful TCDPA attacks alter the labels of the targeted samples without requiring any modifications during deployment. Moreover, these attacks do not diminish the model's performance on a standard test set, rendering them virtually undetectable during deployment (Schwarzschild et al., 2021; Geiping et al., 2021).

[1]Department of Computer Science, Rutgers University - New Brunswick, New Jersey, USA [2]Massachusetts Institute of Technology, Cambridge, Massachusetts, USA. Correspondence to: Snigdha Sushil Mishra <sm2600@rutgers.edu>.

2^{nd} *AdvML Frontiers workshop at* 40^{th} *International Conference on Machine Learning*, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

Existing TCDPA methods mainly fall into three technical categories: Feature Collision (Shafahi et al., 2018), Convex Polytope (Zhu et al., 2019), and Gradient-Matching (Geiping et al., 2021). Feature Collision and Convex Polytope attacks are particularly effective when training only the last layer of the neural network, whereas Gradient-Matching has demonstrated superior performance when training all parameters of the network (Geiping et al., 2021). Considering the wider applicability of the latter scenario, in the paper, we focus on Gradient-Matching. Previous research (Geiping et al., 2021) has solely evaluated gradient-based attacks on balanced datasets such as ImageNet and CIFAR, leaving their effectiveness on imbalanced datasets unknown. However, many real-world applications, including medicine (Matek et al., 2021; Liu et al., 2022), intrusion detection, and anomaly detection, involve imbalanced classes. Consequently, it is crucial to investigate data poisoning attacks in the context of imbalanced datasets. Two main approaches for addressing class imbalance in deep learning are *Re-sampling* and *Re-weighting* (Buda et al., 2018; Cui et al., 2019), with more advanced techniques building upon them. Notably, advanced methods like Focal Loss (Lin et al., 2017) and LDAM (Cao et al., 2019) incorporate Re-sampling as part of their algorithms.

In this study, we investigate gradient-matching based TCDPA within the context of datasets containing imbalanced classes. We conduct a detailed analysis of the training objectives associated with gradient-matching TCDPA and demonstrate their vulnerability to commonly used data balancing techniques, i.e., Re-sampling or Re-weighting. To address this challenge, we propose novel TCDPA attacks that are specifically designed to be invariant to class distribution changes induced by Re-sampling or Re-weighting. Our methods re-formulate the TCDPA attack by removing the attack objective's dependency on class probability or class-specific sample weights. Through empirical evaluations, we demonstrate the superior performance of our proposed methods compared to the Witches Brew algorithm (Geiping et al., 2021) on imbalanced class datasets. Our best method CLI-WB shows improvement of 9% to 51% ASER on datasets like MNIST and CIFAR10.

2. Related Work

Exploits in Neural Networks aim to influence the trained neural net model's deployment behavior or compromise its deployment objectives. Examples of such exploits are Backdoor attacks (Schwarzschild et al., 2021), Data poisoning attacks (Shafahi et al., 2018; Geiping et al., 2021) and Privacy Leakage attacks (Shokri et al., 2017). We focus on Data Poisoning attacks, which are used to poison the training data, usually using imperceptible perturbations. Suciu et al. (2018) and Paudice et al. (2019) used methods like Label Flipping and Watermarking, with training data label modification, or image watermarking to achieve data poisoning. These attacks compromise the natural semantic labels of the training images, and are not clean-label. Methods like Poison Frogs (Shafahi et al., 2018), Convex Polytope (Zhu et al., 2019) and Witches Brew(Geiping et al., 2021) use feature collision, convex polytope and gradient matching, respectively, to achieve a targeted clean-label data poisoning attack.

There is limited amount of work on data poisoning methods in the context of imbalanced class datasets. Peri et al. (2020) analyse defences against data poisoning methods like (Shafahi et al., 2018) on imbalanced datasets. Their study is limited to convex polytope and feature collision based attacks in the transfer learning scenario. In contrast, our method focuses on the more widely used scenario, where all layers of the neural network are trained. Jin et al. (2021) uses imbalanced dataset evaluation, but their focus is not on studying the effect of data-imbalance on data poisoning.

3. Background

3.1. Threat Model

The threat model for TCDPA is as follows:
Goal. Attacker wishes to induce misclassification of a targeted sample $(x_i^t, y_i^t)_{i=1}^T$ to the class y_i^{adv}. Here, x is the input sample, y is the class label, i is the corresponding sample out of T targeted samples, and y_i^{adv} is the adversarial label to which we want the input sample x_i^t to be misclassified as.

Attacker. The capabilities of the attacker are

1. Read-only access to training data.
2. Read-only access to model specification.
3. Write access and a limited budget for modifying a small set of training samples.

Victim. The capabilities of the victim are:

1. Complete control over model training procedure.
2. Read and write access to entire training set.
3. Victim verifies the collected training data through a human expert.

To ensure that the attack is invisible to the Victim side subject expert, any changes to the training data should be label-consistent and imperceptible.

3.2. Gradient-Matching based Data Poisoning

Gradient-Matching based TCDPA methods like Witches Brew (WB) assume gray-box access to the model. The attacker has information like model initialization, architecture and optimization procedure. Increased access to the aforementioned information increases the effectiveness of the attack (Geiping et al., 2021).

The Witches Brew algorithm requires access to the training data and the model architecture. It selects a set of P samples $\{x_i, y_i\}_{i=1}^P$ from the training set of N samples. Here, $x_i \in \mathbb{R}^K$ and $y \in Y$, where Y is the set of labels . A perturbation Δ_i is applied to each data sample. The attack process should optimize Δ such that for a set of test samples $(x_i^t, y_i^t)_{i=1}^T$, the model $f_{\theta(\Delta)}$ parameterized by $\theta(\Delta)$ misclassifies the test samples as the target class y_i^{adv}. More formally, the bi-level optimization is as follows:

$$\min_{\Delta} \sum_{i=1}^{T} \mathcal{L}(f_{\theta(\Delta)}(x_i^t), y_i^{adv})$$
$$\text{s.t. } \theta(\Delta) \in \text{argmin}_\theta \frac{1}{N}[\sum_{i=1}^{P} \mathcal{L}(f_\theta(x_i + \Delta_i), y_i) \quad (1)$$
$$+ \sum_{i=P+1}^{N} \mathcal{L}(f_\theta(x_i), y_i)]$$

Here, Δ can take any value from $\mathbb{R}^{K\times P}$ with the constraint that $||\Delta||_\infty \leq \delta$, where $\delta \in \mathbb{R}^+$ is the allowed radius of poisoning perturbation. $\mathcal{L}$ is the loss function. The perturbation Δ_i for any $i > P$ is zero, i.e. Witches Brew only uses P samples for poisoning. Similar to (Geiping et al., 2021), our experiments will also focus on a test sample size of $T = 1$. The attack can be extended to multiple targets by increasing the budget P of poisoned samples.

Witches Brew optimizes the objective in Equation 1 by using poisoned sample gradients to mimic the adversarial gradient of target samples for any θ during model training as follows:

$$\nabla_\theta \mathcal{L}(f_\theta(x^t, \theta), y^{adv}) \approx \frac{1}{P}\sum_{i=1}^{P} \nabla_\theta \mathcal{L}(f_\theta(x_i + \Delta_i), y_i) \quad (2)$$

Witches Brew objective increases the alignment of LHS and RHS for a specific θ. The optimization objective is

$$\mathcal{B}(\Delta, \theta) = 1 - \frac{\langle \hat{g}_\theta, \sum_{i=1}^{P} \nabla_\theta \mathcal{L}(f_\theta(x_i + \Delta_i), y_i)\rangle}{||\sum_{i=1}^{P} \nabla_\theta \mathcal{L}(f_\theta(x_i + \Delta_i), y_i)||} \quad (3)$$

Here $g_\theta = \nabla_\theta \mathcal{L}(f_\theta(x^t), y^{adv})$, is the adversarial target gradient and $\hat{g}_\theta := \frac{g_\theta}{\|g_\theta\|_2}$ is the unit vector along its direction.

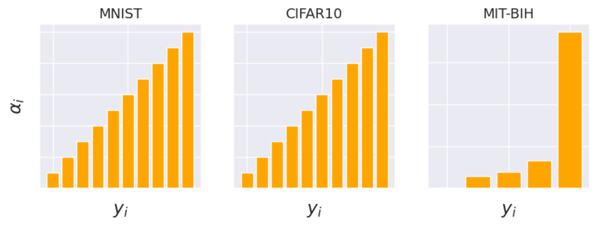

Figure 1: Class importance weights $\boldsymbol{\alpha}$ in si-MNIST, si-CIFAR10 and MIT-BIH during training. Details of datasets is in Section 5.1. Here α_i is the class-specific weight for the i^{th} class and y_i is that class label. The $\boldsymbol{\alpha}$ ensures equal contribution of each class to the loss. Witches Brew sets $\alpha_i = 1$ for all classes during the poisoning generation, which may lead to sub-optimal attack efficacy.

4. Gradient-Matching based Data Poisoning for Imbalanced Data

In a standard neural network based classification, a neural network $f_\theta(.)$ is trained to minimize the loss, i.e., $\min_\theta \mathbb{E}_{y\in P(y)} \left[\mathbb{E}_{(x)\in P(x|y)} \mathcal{L}(f_\theta(x), y)\right]$ which equals to,

$$\arg\min_\theta \sum_y P(y) \left[\sum_{(x)\in D(y)} P(x|y)\mathcal{L}(f_\theta(x), y)\right] \quad (4)$$

Here $D(y)$ denotes the set of samples in the dataset D which have the class label y. In case of a balanced dataset $P(y) = 1/|Y|$ where $|Y|$ is the number of class labels. In case of imbalanced datasets, $P(y)$ may be very small for minority classes. This is especially true for anomaly detection datasets in domains like security healthcare and medicine. For instance, in the MIT-BIH (Moody & Mark, 2001) for arrhythmia detection, all non-normal class (class V,Q,S,F) probabilities are lower than 0.05 (in a 5 class classification problem). While the majority class (class N) has a class probability higher than 0.85. (Refer Table 1 and Table 5)

Re-sampling or Re-weighting methods try to change the importance of the rare class samples in order to prioritize their loss during the training procedure. Re-weighting is done by explicitly replacing $P(y)$ with a class-specific parameter α_y. Using a value of α_y that is greater than or less than $P(y)$ increases or decreases the importance of that sample. In contrast to Re-weighting, Re-sampling changes the class-specific parameter α_y implicitly. It does so by increasing or decreasing the number of class-specific samples instead of changing the importance of each sample. We will design our proposed methods for Re-weighting and assume that the class-specific parameters ($\boldsymbol{\alpha} = \left[\alpha_1, \alpha_2 ... \alpha_{|Y|}\right]$) are changed explicitly. Towards the end of this section we will discuss how our method can be easily extended to the Re-sampling case.

Class-Weight-Invariant Witches Brew for Re-weighting

In this section, we formulate our class weight invariant data poisoning method for the Re-weighting case. Gradient-Matching methods like Witches Brew (Geiping et al., 2021) are not invariant to different $\boldsymbol{\alpha}$ values. To show this, let us rewrite the objective for the attack optimization provided in Equation 3, using class importance weights $\boldsymbol{\alpha}$.

$$\mathcal{B}(\Delta, \theta, \alpha) = 1 - \frac{\langle \hat{g}_\theta, \sum_y \alpha_y \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)\rangle}{|| \sum_y \alpha_y \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)||} \quad (5)$$

Here, $x_{(y,i)}$ is the i^{th} sample of class y. For ease of notation we replace $x_{(y,i)} + \Delta_i$ with $x'_{(y,i)}$. The parameters α_y are the class weight values for each class y. The objective $\mathcal{B}(\Delta, \theta, \alpha)$ is a function of α. This means that if there is a mismatch between the α used during the attack construction and the α' value used to train the final model, reducing the objective $\mathcal{B}(\Delta, \theta, \alpha)$ may no longer induce gradient-matching. The mismatch in α and α' can be significant, as shown in Figure 1.

The values of α can be frequently changed during model development, depending on the precision and recall required by the model developers. As a result, we see that loss re-weighting may cause misalignment of target and poisoned gradients. To account for this we propose a perturbation that makes our new objective $\mathcal{B}'$ independent of $\boldsymbol{\alpha}$.

We propose a new objective $\mathcal{B}'(\Delta, \theta) = (1/|Y|) \sum_y \mathcal{B}'_y(\Delta, \theta)$, where $\mathcal{B}'_y(\Delta, \theta)$ is

$$\begin{aligned} \mathcal{B}'_y(\Delta, \theta) =& 1 - \frac{\langle \hat{g}_\theta, \alpha_y \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)\rangle}{||\alpha_y \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)||} \\ =& 1 - \frac{\langle \hat{g}_\theta, \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)\rangle}{|| \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)||} \end{aligned} \quad (6)$$

This objective isolates samples from each class and separately aligns them with the target gradient. Implementing the Witches Brew procedure with the modified objective in Equation 6, we get our Class Weight Invariant (CLI) TCDPA procedure. We refer to this procedure as **CLI-WB**.

It should be noted here that for CLI-WB, each attack optimization step will require $|Y|$ differentiation steps, one for each class. Each differentiation step calculates $\sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$ for a specific class. While this does not result in an increase in asymptotic complexity w.r.t the number of samples, a large number of classes $|Y|$ may increase the practical run-time of our algorithm multiple-fold.

If we modify the Equation 6 such that $\mathcal{B}'_y(\Delta, \theta)$ is linear in $\nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$, we can remove the factor of $|Y|$ from the practical run-time of the method. We can do this by using projections instead of normalized inner products, to induce matching between target gradient and the poisoned gradients. Instead of increasing the normalized inner product between $\hat{g}_\theta$ and $\nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$ (as proposed in Witches Brew), we can instead increase the projection of $\nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$ on the unit vector $\hat{g}_\theta$. This results in an objective that is linear in $\nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$ as follows:

$$\mathcal{B}''_y(\Delta, \theta) = 1 - \frac{\langle \hat{g}_\theta, \alpha_y \sum_{x_{(y,i)}} \nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y) \rangle}{\alpha_y \sum_{x_{(y,i)}} 1} \quad (7)$$

Here, the mean of each class' projection is divided by the class importance weights to ensure that the proposed objective is still invariant to changes in class weights. It is important to understand that the value of α_y used in Equation (7) is the one used in the attack. Therefore, we can replace it by the value of 1.0, if we construct our attack without any cost weighting. The attack based on this objective is referred to as CWI-WB-Projection (**CWI-WB-P**). This objective is linear in $\nabla_\theta \mathcal{L}(f_\theta(x'_{(y,i)}, y)$, which means that we can use a single differentiation of classification loss to compute the complete objective $\mathcal{B}''(\Delta, \theta) = (1/|Y|) \sum_y \mathcal{B}''_y(\Delta, \theta)$. This reduces the practical run-time of our algorithm.

The objective $\mathcal{B}''(\Delta, \theta)$ also provides an additional advantage. It optimizes the projection of the poisoned sample gradients on the target gradient. This incentivizes the poisoned samples to have a larger component vector in the direction of the target gradient. As a result, CWI-WB-P may amplify the poisoning effect of the resultant poisoned gradient.

Class-Weight-Invariant Witches Brew for Re-sampling

Re-sampling can be implemented using sampling with and without replacement. Let us first consider the case where Re-sampling without replacement is used to increase the priority of the rarer class during the neural network model training. In this case, the class specific parameter values are the sampling probabilities of each class during stochastic gradient descent. The class probabilities are the parameter vector of a multinomial distribution ($\hat{\boldsymbol{\alpha}} = \left[\hat{\alpha_1}, \hat{\alpha_2} ... \hat{\alpha_{|Y|}}\right]$), where is $\sum_{i=1}^{|Y|} \hat{\alpha_i} = 1$. The gradient mismatch in this case comes from the fact that this multinomial distribution used during model training, may be different that the distribution used during the poison construction.

Let us assume without loss of generality that the first class (y_1) is the most abundant class in the training dataset. Then for each class y_i $(i \neq 1)$, its samples will be repeated roughly $\frac{\hat{\alpha_i}}{\hat{\alpha_1}}$ number of times in a stochastic gradient descent epoch. When we plug this in Equation 6, the class specific multipliers $\frac{\hat{\alpha_i}}{\hat{\alpha_1}}$ in the numerator and denominator cancel out. This gives use the same $\mathcal{B}'_y(\Delta, \theta)$ formulation as before.

Table 1: Number of data samples (train/test) for the head and tail class in si-MNIST, si-CIFAR10 and MIT-BIH.

	si-MNIST	si-CIFAR10	MIT-BIH
Head class	5923 / 980	5000 / 1000	76554 / 13142
Tail class	592 / 98	500 / 100	411 / 97

The same procedure can be shown for Equation 7. So we can apply CLI-WB and CLI-WB-P procedures for both Re-sampling and Re-weighting based training methods. It should be noted that in Re-sampling, due to the stochasticity introduced by the sampling procedure, the alignment of poison gradient and the target gradient may not be exact. This is especially true for Re-sampling with replacement schemes. As a result, Re-sampling may be more challenging for gradient-matching methods.

5. Experiments

To study the effectiveness of WB, CWI-WB and CWI-WB-P attacks, we run experiments on MNIST and CIFAR datasets [1]. We evaluate our methods on models trained using both Re-sampling and Re-weighting. We also run a smaller set of experiments on the MIT-BIH clinical dataset (Moody & Mark, 2001), using only Re-sampling training to see the effect of our system on a real-world dataset.

5.1. Datasets

MNIST and CIFAR10 have been widely used for data poisoning evaluation (Geiping et al., 2021; Shafahi et al., 2018). We use these two datasets to evaluate our TCDPA methods. Since these datasets have a uniform class distribution, we use a sub-sampling scheme to artificially induce class imbalance. For our experiments, we choose a class imbalance ratio of 0.1 for the smallest (tail) and the largest (head) classes. For the remaining sections of this paper, we use si-MNIST and si-CIFAR10 to refer to the simulated-imbalanced-MNIST and CIFAR10. Details for class imbalance simulation are in Appendix A.1.

MIT-BIH (Moody & Mark, 2001) is an ECG dataset of normal and abnormal heart beats. We use a coarse classification of heart beat types (Das & Ari, 2014) which classifies the ECG sample into 5 heart beat type. MIT-BIH dataset has naturally imbalanced classes, with the majority class (the Normal heart beat class "N"), consisting of more than 85% of the training samples. Head and tail class samples counts are provided in 1. Additional details are in Appendix A.2.

[1]Code Repository : https://github.com/snigdhas-mishra/Data_Poisoning_Imbalanced_Classifiers

Table 2: Macro and Weighted F1-scores of different models (ResNet and CNN1D) trained on clean datasets.

Model	Dataset	Macro F1	Weighted F1
ResNet Models trained with Re-Weighting			
ResNet18	si-MNIST	0.53	0.55
ResNet18	si-CIFAR10	0.22	0.29
ResNet50	si-MNIST	0.54	0.57
ResNet50	si-CIFAR10	0.17	0.19
ResNet Models trained with Re-Sampling			
ResNet18	si-MNIST	0.95	0.97
ResNet18	si-CIFAR10	0.48	0.55
ResNet50	si-MNIST	0.86	0.90
ResNet50	si-CIFAR10	0.39	0.46
CNN1D models trained with Re-Sampling			
CNN1D	MIT-BIH	0.47	0.72
CNN1D+Res.	MIT-BIH	0.39	0.66

Algorithm 1 TCDPA Procedure

```
1: Accept the target sample $(x^t, y^t)$, and target label $y^{adv}$ as input.
2: Train a clean model on the training dataset $\{x_i, y_i\}_{i=1}^N$ to obtain the clean model parameters $\theta^*$.
3: for $i = 1$ to $k$ do
4:     for $h \in \mathcal{H}$ do
5:         Select $P$ training samples to poison and estimate $\Delta = \Delta_{i=1}^P$ values by optimizing the objective $\mathcal{B}$ using the hyper-parameter setting $h$.
6:         Store the final attack objective value.
7:     end for
8: end for
9: Select the poison with the lowest attack loss.
10: Retrain the model using poisoned samples.
11: Evaluate retrained model output on the target sample.
```

5.2. Neural Network Models

We used ResNet18 and ResNet50 models to train classifiers for si-CIFAR10 and si-MNIST models. And, we use two Convolutional Neural Network models CNN1D and CNN1D+Res to model the classification of ECG signals. Our CNN1D model uses 4 CNN layers and 1 linear output layer. Our CNN1D + Residual model uses 3 CNN layers. The middle CNN layer is repeated with residual connections. ResNet models were trained for 50 epochs using Adam optimizer with learning rate of 1e-3. Both MIT-BIH models were trained using Stochastic Gradient Descent for 50 epochs, with learning rate of 1e-4 with learning rate decay.

We use both Re-Sampling and Re-Weighting to maintain a balanced class distribution in the training data. The F1-score performance of all models is reported in Table 2. For Re-sampling, we sample each class uniformly when constructing the batches. For Re-weighting, we compute the class importance weights α in a way such that each class has equal contribution over the total training loss. This induces a $P(y) = 1/5$ on every class for MIT-BIT and $P(y) = 1/10$ on every class in si-MNIST, si-CIFAR10 datasets. Additional details about the Neural Networks and their performance analysis is provided in Appendix B. We use these models to evaluate the TCDPA attack efficacy.

5.3. Experimental Setup

Our experimental setup to evaluate TCDPA methods borrows the procedure from Witches Brew (Geiping et al., 2021) with a few modifications to allow for imbalanced data. Our overall attack procedure is described in Algorithm 1. Here k is the number of repeats for designing the poison and the set $\mathcal{H}$ is the set of hyper-parameter combinations used to obtain the poison. As mentioned in the previous subsection, during model training we use data-balancing techniques to ensure a balanced training set. However, our poison construction is done using uniform α values for all classes. This shift simulates the α discrepancy between attack model training and victim neural model training in our experiments.

5.4. Evaluation Metric

TCDPA are usually evaluated using Attack Success Rate (ASR). Attack Success Rate is defined by the proportion of attacks that have successfully flipped the victim sample's label from the correct class to the target class. Using ASR metric in our experiments may lead to biased results due to the high misclassification rate on rarer class samples in imbalanced data classifiers. Table 2 shows the test-set performance of our models. The macro-averaged F1-score for all models except si-MNIST trained models is less that 0.5. This shows that there are a significant number of misclassifications in our neural model predictions. Therefore, it is likely that the targeted sample for our TCDPA may also be misclassified due to model's erroneous output. This in turn may lead to biased attack success evaluation. Additionally, in domains with class-imbalance, label-shift correction algorithms such as (Lipton et al., 2018) may be used to account for the class re-weighting during testing. Label-shift correction will change the output probabilities and therefore may change the attack success rate.

Therefore, we define a new metric, *Attack Effect Success Rate (AESR)*. For an attack on a target (x^t, y^t) for target label y^{adv}, the TCDPA attack is considered *effective* if $(f_\theta(x^t)[y^t] - f_{\theta(\Delta)}(x^t)[y^t]) > \epsilon$ and $(f_\theta(x^t)[y^{adv}] - f_{\theta(\Delta)}(x^t)[y^{adv}]) < \epsilon$. This means that the attack is *effective* if it reduces the probability of target samples actual label y^t and simultaneously increases the probability of label y^{adv}. AESR is defined as the proportion of attacks that are *effec-*

Table 3: TCDPA attack results (AESR) on si-MNIST and si-CIFAR10. Each experiment consists of 150 trials. Best performance is in **bold**.

Model	TCDPA	Dataset	
		si-MNIST	si-CIFAR10
ResNet Models trained with Re-sampling			
ResNet18	Witches Brew	0.253	0.226
	CWI-WB	0.320	**0.333**
	CWI-WB-P	**0.333**	0.200
ResNet50	Witches Brew	0.200	0.193
	CWI-WB	**0.282**	0.226
	CWI-WB-P	0.220	**0.286**
ResNet Models trained with Re-weighting			
ResNet18	Witches Brew	0.306	0.213
	CWI-WB	**0.453**	**0.233**
	CWI-WB-P	0.360	0.193
ResNet50	Witches Brew	0.286	0.233
	CWI-WB	0.320	0.260
	CWI-WB-P	**0.333**	**0.346**

tive. To reduce the possibility of counting random effects as changes in model output, we use a threshold ϵ to ensure that we do not count minor changes in a model's output probability. We use an epsilon of $\epsilon = 0.01$. AESR is not affected by label-shift correction, because it only uses the relative changes in each class' output.

6. Results

We use the Witches Brew method as the baseline. Our proposed methods CWI-WB and CWI-WB-P, use the objective provided in Equation 6 and 7 respectively. For each experiment, we run $k \times |\mathcal{H}|$ attacks and then select the attack with the best attack objective. We then retrain the neural model on the poisoned dataset (as shown in Section 5.3). To estimate AESR, we repeat this experiment N times. We run a comprehensive set of experiment with N = 150 for si-MNIST and si-CIFAR10 Dataset. The victim samples are selected from 3 different classes, (50 samples each). The target class y^{adv} for all victim samples is Class "0" for si-MNIST and "airplane" for si-CIFAR10. For our experiments, $k = 10$ and $|\mathcal{H}| = 3$.

The results of these experiments are provided in Table 3. We see that our proposed method CWI-WB consistently outperforms the baseline Witches Brew for both ResNet models and datasets. The improvements show similar trends for both Re-sampling and Re-weighting based neural nets. Additionally, our proposed simplification to CWI-WB, the CWI-WB-P model is either competitive or better than the baseline. CWI-WB model does seem to perform better overall as compared to CWI-WB-P, but that comes at the cost of a higher practical run-time for attack optimization. Since for each experiment, the attack optimization is executed $k \times |\mathcal{H}|$, this cost can result in significantly increased overall runtime.

Table 4: TCDPA attack results (AESR) on MIT-BIH for Re-sampling. Each experiment consists of 10 trials.

Model	TCDPA	V → N	V → S
CNN1D +Residual	Witches Brew	6/10	2/10
	CWI-WB	4/10	7/10
CNN1D	Random Sel.	4/10	6/10
	CWI-WB	5/10	6/10

We also ran a smaller set of experiments on the MIT-BIH dataset to access the applicability of our methods on a real-world domain with high negative costs associated with vulnerable systems. For MIT-BIH, we evaluate our TCDPA with the more challenging Re-sampling based training. We construct two test scenarios. In the scenario "V→N", we choose a minority class (Class V) test sample as the victim sample and the head class (Class N) as the target label (y^{adv}). In the scenario "V→S" we choose the same minority class (Class V) as the victim sample and use the tail class (Class S) as the target label. This mimics the two cases of poisoning a data sample towards a false negative ("V→N") and incorrect medical diagnosis ("V→S"). We again use $k = 10$ and $|\mathcal{H}| = 3$. We run this experiment for the baseline and our best performing proposed method CWI-WB. We repeated our experiment 10 times for both scenarios. Table 4 shows that our proposed method CWI-WB provides better or comparable attack success in both scenarios.

Additionally, we also observed that the F-1 score performance of the poisoned models using WB, CLI-WB and CLI-WB-P was similar to the clean model performance shown in Table 2. We provide the F1-score of models trained on CLI-WB poisoned data in the Appendix (Table 6).

7. Discussion

CWI-WB vs CWI-WB-P: Table 3 and 4 show that our proposed models exhibit comparable or higher Attack Effect Success Rate (AESR) as compared to the baseline Witches Brew. The method CWI-WB shows a more consistent improvement over witches brew, as compared CWI-WB-P. CWI-WB tries to induce higher alignment between the target and the poisoned gradients, while CWI-WB-P tries to increase the component of the poisoned gradient vectors that is parallel to the target gradient. It is unclear why the CWI-WB works better than CWI-WB-P, since they both are approximate solutions to the optimization in Equation 1.

Evaluation Metric: Our choice of attack evaluation metric (AESR) is different from the standard attack evaluation metric of ASR. As mentioned before, this is because 1) machine learning models trained on imbalanced data may already have very low performance on the minority classes, and 2) label-shift correction methods may bias the ASR evaluation metric. Our proposed method accounts for both these factors. However, a disadvantage of our metric is that an attack's ASER may be high even when the attack may not result in a high number of label flips. ASER is useful in studies that need to establish comparisons between different TCDPA methods. However, ASER is not an accurate assessment of an attack's practical efficacy.

8. Conclusion

In our study, we explored the application of gradient-matching TCDPA methods on imbalanced data classification models using Re-sampling or Re-weighting techniques. Our analysis revealed that gradient misalignment hampers the effectiveness of TCDPA in these scenarios. To mitigate this issue, we proposed two novel methods that effectively enhance the performance of gradient-matching TCDPA attacks, as demonstrated in our experimental results.

Acknowledgements

All experiments for this work were done using the ILab cluster resources at Department of Computer Science, Rutgers University-New Brunswick.

References

Buda, M., Maki, A., and Mazurowski, M. A. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

Das, M. K. and Ari, S. Ecg beats classification using mixture of features. *International scholarly research notices*, 2014, 2014.

Geiping, J., Fowl, L. H., Huang, W. R., Czaja, W., Taylor, G., Moeller, M., and Goldstein, T. Witches' brew: Industrial scale data poisoning via gradient matching. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=01olnfLIbD.

Jin, C., Sun, M., and Rinard, M. Provable guarantees against data poisoning using self-expansion and compatibility. *arXiv preprint arXiv:2105.03692*, 2021.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Lipton, Z., Wang, Y.-X., and Smola, A. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pp. 3122–3130. PMLR, 2018.

Liu, P., Sun, X., Han, Y., He, Z., Zhang, W., and Wu, C. Arrhythmia classification of lstm autoencoder based on time series anomaly detection. *Biomedical Signal Processing and Control*, 71:103228, 2022.

Matek, C., Krappe, S., Münzenmayer, C., Haferlach, T., and Marr, C. Highly accurate differentiation of bone marrow cell morphologies using deep neural networks on a large image data set. *Blood, The Journal of the American Society of Hematology*, 138(20):1917–1927, 2021.

MetaAI. State of the art —papers with code. URL https://paperswithcode.com/sota.

Moody, G. B. and Mark, R. G. The impact of the mit-bih arrhythmia database. *IEEE engineering in medicine and biology magazine*, 20(3):45–50, 2001.

Paleyes, A., Urma, R.-G., and Lawrence, N. D. Challenges in deploying machine learning: a survey of case studies. *ACM Computing Surveys*, 55(6):1–29, 2022.

Paudice, A., Muñoz-González, L., and Lupu, E. C. Label sanitization against label flipping poisoning attacks. In *ECML PKDD 2018 Workshops: Nemesis 2018, UrbReas 2018, SoGood 2018, IWAISe 2018, and Green Data Mining 2018, Dublin, Ireland, September 10-14, 2018, Proceedings 18*, pp. 5–15. Springer, 2019.

Peri, N., Gupta, N., Huang, W. R., Fowl, L., Zhu, C., Feizi, S., Goldstein, T., and Dickerson, J. P. Deep k-nn defense against clean-label data poisoning attacks. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 55–70. Springer, 2020.

Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J. P., and Goldstein, T. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9389–9398. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/schwarzschild21a.html.

Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., and Goldstein, T. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31, 2018.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Suciu, O., Marginean, R., Kaya, Y., Daume III, H., and Dumitras, T. When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 1299–1316, 2018.

Zhu, C., Huang, W. R., Li, H., Taylor, G., Studer, C., and Goldstein, T. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pp. 7614–7623. PMLR, 2019.

Table 5: Number of Train and Test samples for our used split of Train and Test Patients in MIT-BIH dataset.

Class	# Train Samples	# Test Samples
N	76554	13142
V	5357	4147
Q	3881	376
S	2291	1470
F	411	97

A. Dataset Details

A.1. MNIST and CIFAR10

MNIST and CIFAR10 datasets have a uniform class distribution. Therefore we use a stratified sub-sampling scheme to artificially induce class imbalance. We choose a class imbalance ratio of 0.1 for the smallest (tail) and the largest (head) classes. This means that the tail class' sample size is 10% of the sample size for the head class. The sample size of the remaining classes is evenly distributed between the classes. For instance, the training set sample size of classes for our simulated-imbalanced-MNIST is $5923, 5330, 4738, 4146, 3553, 2961, 2369, 1776, 1184, 592$.

A.2. MIT-BIH

MIT-BIH (Moody & Mark, 2001) is an ECG dataset of 44 patients. The dataset annotates normal and abnormal heart beat types with 12 types of beats. We use a coarse classification of heart beat types used by (Das & Ari, 2014). They use the AAMI classification of heart beat types into Normal (N), Supraventricular ectopic beat (S), Ventricular ectopic beat (V), Fusion beat (F) and Unknown beat (Q). We select 8 patients and use their samples to form the test set. The remaining 36 patients are in the training dataset. We split the train and test set by patients, to mimic the real-world use case where the model is deployed for previously unseen patients. We preprocessed the ECG dataset by segmenting each heart-beat into a single sample. [2]

As can be seen from Table 5, MIT-BIH dataset has imbalanced classes. The majority class is the (N) Normal heart beat class, consisting of more than 85% of the training samples.

B. Neural Network Performance Details.

We used ResNet18 and ResNet50 models to train classifiers for si-CIFAR10 and si-MNIST models. The performance of our Neural Network models trained on clean datasets (as seen in Table 2 vary between high F1-scores in case of ResNet18 trained on MNIST to very low F1 values in the case of ResNet50 trained on CIFAR10. The low performance of some of these models is likely due to the over-fitting and instability issues of Re-weighting and Re-sampling. Re-sampling techniques may sometimes overfit the model on minority classes. Similarly, Re-weighting techniques tend to destabilize the neural network training due to very high weights on minority class samples. These issues are exacerbated in our models, because we keep the α values fixed, in order to ensure a fair TCDPA attack evaluation.

The F1-scores of neural net models trained on datasets that were poisoned using CLI-WB are provided in Table 6. We can see that F1-scores of Poisoned models are very close to their clean counterparts in Table 2.

[2]Our pre-processing code is based on a third-party github repository `https://github.com/eddymina/ECG_Classification_Pytorch`.

Table 6: Average Macro and Weighted F1-scores of different models (ResNet and CNN1D) trained on datasets poisoned using CLI-WB. Results are averaged over 10 trials, evaluate seperately from the experiments used for Table 3.

Model	**Dataset**	**Macro F1**	**Weighted F1**
ResNet Models trained with Re-Weighting			
ResNet18	si-MNIST	0.52	0.57
ResNet18	si-CIFAR10	0.22	0.30
ResNet50	si-MNIST	0.52	0.57
ResNet50	si-CIFAR10	0.17	0.19
ResNet Models trained with Re-Sampling			
ResNet18	si-MNIST	0.95	0.97
ResNet18	si-CIFAR10	0.47	0.55
ResNet50	si-MNIST	0.85	0.91
ResNet50	si-CIFAR10	0.38	0.45
CNN1D models trained with Re-Sampling and V→N poisoning.			
CNN1D	MIT-BIH	0.47	0.74
CNN1D+Res.	MIT-BIH	0.40	0.67
CNN1D models trained with Re-Sampling and V→S poisoning.			
CNN1D	MIT-BIH	0.47	0.72
CNN1D+Res.	MIT-BIH	0.36	0.68