Final Project Report: Just how much we need to fear Data Poisoning?

Shriya Surusani[†]
School of Computing
Clemson University
Clemson, South Carolina
shriyas@g.clemson.edu

Angel Isaac[‡]
School of Computing
Clemson University
Clemson, South Carolina
angelji@g.clemson.edu

***Abstract*—Data Poisoning as we know it has been a prominent issue as a security threat to machine learning systems where attacks manipulate training data in order to cause models to fail during inference. Though the existing work presents us with comparative study on how much these attacks actually effects the model and how an attacker controls the behavior of a system by manipulating its training data. We can however see that these experimental results are often unreliable. This is majorly for the reason that these poisoning methods have not been tested in consistent or realistic settings. It has been observed that these data poisoning and backdoor attacks are highly sensitive to variations in the testing setup which has not been shown in the current work. Though Data poisoning literature contains attacks in a variety of settings including image classification, facial recognition,and text classification [1] [2], we focus particularly on models based on image classification in this paper.**

Due to these discrepancies in the presented results and also because data poisoning is the number one concern among threats ranging from model stealing to adversarial attacks, we in this paper present Standardized benchmarks that are developed for data poisoning and backdoor attacks such that rigorous tests in a realistic settings are performed to determine the extent to which we should fear these attacks. We would be reproducing the experimental results presented in "Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks" [3]

***Index Terms*—Data Poisoning, Deep Neural Network, Benchmark, Malware, Backdoor, triggered attack, triggerless attack, Machine Learning, .**

I. Introduction

This Data poisoning which is basically an attacker controlling the behavior of a system by manipulating its training data has been up for debate among the researcher community for a while on how to project the effects of it as close to what can be observed in real time. It requires a large amounts of data to train to train the model and this data to train or re-train the model has been scraped from the web which happens to be the main source of collecting the large datasets. As an example we can look at the Open Images and the Amazon Products datasets which collects around 9 million and 233 million samples, respectively, from the sources unknown or insecure [4].

This very same factor is the reason why at this scale, looking out for the authenticity of the data collected and properly vetting the content to make sure there is no malicious payload in it is close to impossible. Furthermore, one of the industrial practice to collect the datasets is to harvest system inputs like e-mails, uploaded files and scraping user-created content like from the profiles, text messages,advertisements without cross checking the collected data leaving the scope for malicious actors to contribute their payload and successfully infiltrate the data collected. The dependence of industrial AI systems on this datasets collected without any manual inspection or any kind of inspection for that matter has lead to rise in fear of producing the faulty models based on this poisoned datasets [5].A recent survey that was conducted among the industry organizations also shows that the fear of data poisoning is more in these companies than that of other threats from adversarial machine learning [3].

Data poisoning attacks as a whole can be put into 2 broad categories as follows:

- Backdoor data poisoning attacks
- Triggerless poisoning attacks

In the case of backdoor data poisoning attacks, the model is made to misclassify the test-time samples that contain a trigger. A trigger is a visual feature in an image or a patch in an image with changed pixels or a particular character sequence in the natural language setting [6] [1]. As an example here, an attacker might tamper with training images such that a vision system fails to identify a stop sign or worse, he might also set a trigger such that the vision system misclassifies it as a different traffic sign which can lead to fatal accidents on road. This threat model works when the attacker modifies data at train time by placing poisons in the dataset and at the inference time by inserting the trigger. It is important in backdoor data poisoning attacks to successfully achieve these both steps in order to make the model faulty and achieve the malicious purpose.

Coming to the triggerless poisoning attacks, the attacker does not require to make any modification at inference time [7] [8] [9] in order to misclassify the results of a model. Though a lot innovative backdoor and triggerless poisoning attacks along with its defense strategies have emerged throughout the years and a lot of existing work talks about it and their findings, there have been a lot of inconsistencies found around these works and how the perfunctory experimentation's has rendered the performance evaluations. The comparisons shown where also misleading since they fail show any evidence about the results being independent of the experimental setup.

In this paper, we try to achieve a transparent development of a framework for benchmarking and evaluating a wide range of poison attacks on image classifiers and provide a way to compare attack strategies and shed light on the differences between them. In order to achieve this, we reproduce the goals presented in ”Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks” [3] such that we also present if our experimental results are similar to the ones presented in the paper to achieve this goal and also talk about any discrepancies found or limitations of the solution presented in the paper.

The goals of our experimental setup is to address the following weaknesses in the current literature:

- The reported success rates of the poisoning attacks in the current literature is presented based on an unrealistic and far fetched network architecture choice and training protocol which is no where close to what is observed in the real-world setting.
- The standard percentage of training data that an attacker can access to carry out his malicious intent is presented in the poisoning literature is actually not very useful to present an unbiased representation of the effects of poisoning. The flaw in this standardization is that even with a fixed percentage of the dataset poisoned, we argue that the success rate is still highly dependent on the dataset size which has not been standardized in any of the experimentation presented in the literature.
- Some claim that the attacks contains ”Clean labels” which means that even after poisoning is done by the attack. The changes made are not visible by the human eye. This is in-fact not true. The changes made by the attack are clearly visible upon human inspection. For example, in the figure 1 shown on the right, we can see the image before poisoning (top) and the image after poisoning (below) and can clearly make out the difference between them and find that the data has been tampered with.

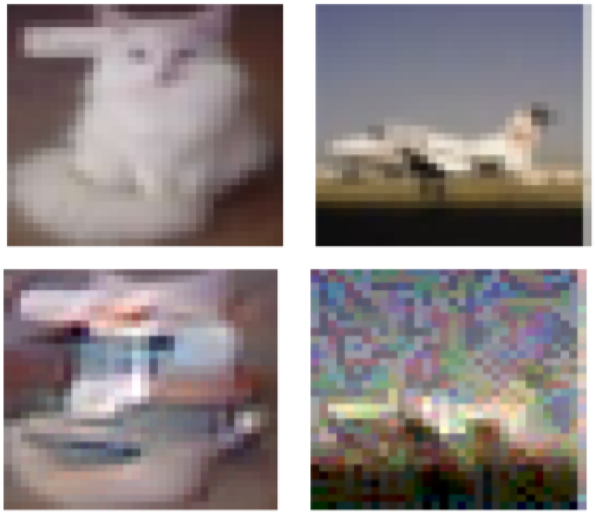


Fig. 1. Bases (top) poisons (bottom)

The benchmarks we are reproducing is to measure the effectiveness of attacks in standardized scenarios using modern network architectures in white-box transfer learning settings. We benchmark to perform from-scratch training scenarios while we constrain the poisoned images to be clean in the sense of small perturbations. Although the data poisoning attacks has a wide range of modalities, such as image classification, facial recognition, and text classification [2] [1] [6] along with development with attacks on the fairness of models, speech recognition, and recommendation engines [10] [11] [12]. Our benchmark reproduced focuses on attacks on image classifiers that only modify data. This is because it is the most common setting in the existing literature and also among these attacks and yet there has not been a standard comparison metric for it. We mainly focus on the attacks with a common goal and the sensitivities of it to the experimental setup.

II. Background

Earlier, the poisoning attacks that where targeted, supported the vector machines and simple neural networks [7] [13]. With gaining popularity of these poisoning attacks, the strategies for triggerless attacks on deep architectures have emerged drastically [9] [14] [15]. In the early time the backdoor attacks contained triggers in the poisoned content and sometimes also in the label making them not clean-label [16] [17] [18]. Also however, the models after poisoning which do not visibly contain a trigger also show positive results [19] [20]. With these raise in strategies, several defense mechanisms have also arise to detect them. But these sanitization-based defenses may also be overwhelmed by some attacks [21] [22].

What we are focusing on in this paper is the attacks that achieve targeted misclassification which means that for both the triggerless and backdoor threat models the motive of the attacker is to cause a target sample to be misclassified as another specified class. We also show how there is an overall decrease in the test accuracy. In these both types of data poisoning attacks, we need to take note that the clean images are called as base images and the images that are tampered by the attacker for an alternate motive that are chosen from a single class is known as base class. This class chosen is often the class into which the attacker aims to target the image or class to be misclassified.

The major differences between the triggerless and backdoor threat models in the literature that are worth noticing are as follows:

- The backdoor attacks alter their targets during inference by adding a trigger. These triggers are planted by adding a small patch of pixels to the images [19]. Where, triggerless attacks contain nothing of this sort.
- The backdoor attacks cause a victim to misclassify any image containing the trigger instead of a particular sample. Whereas, triggerless attacks cause the victim to misclassify an individual image called the target image.

Although the second difference mentioned does not have any significant value to it; for example, triggerless attacks could be designed to cause the victim to misclassify a collection of images rather than a single target. We focus on triggerless attacks that targets individual samples and backdoor attacks that targets whole classes of images in order to be consistent with the poisoning literature while looking at it in the bigger picture.

Our focus as mentioned will be on the clean-label backdoor attack and the hidden trigger backdoor attack. We craft the poisons with optimization procedures which do not contain the visual patches that are easily noticeable [20]. For the triggerless attacks, our focus will be on feature collision and convex polytope methods. We also include the recent triggerless methods Bullseye Polytope (BP) and Witches' Brew (WiB) in our experiment. The following section details the attacks that serve as the subjects of our experiments that we are reproducing.

Technical details: The following are the notations that we will be needing to know before proceeding with the technical details of the attacks. In the equations presented X(c) is the set of all clean training data. x(t) is the target image and the labels are denoted by y and Y for a single image and a set of images, respectively, and are indexed to match the data.We also use f to denote a feature extractor network.

Feature Collision (FC): In this type of attack, the poisons are crafted by adding small perturbations to base images such that their feature representations lie extremely close to that of the target [2]. Each poison in this is the solution to the optimization problem shown in figure 2.

$$x_p^{(j)} = \underset{x}{\operatorname{argmin}} \|f(x) - f(x_t)\|_2^2 + \beta \|x - x_b^{(j)}\|_2^2.$$

Fig. 2. Optimization problem 1

Convex Polytope (CP): In this type of attack, the poisons are crafted such that the target's feature represents a convex combination of the poisons' feature. It is represented by solving the the optimization problem [23] shown in figure 3.

$$\begin{aligned} X_p = \underset{\{c_j\},\{x^{(j)}\}}{\operatorname{argmin}} \quad & \frac{1}{2}\frac{\|f(x_t)-\sum_{j=1}^{J} c_j f(x^{(j)})\|_2^2}{\|f(x_t)\|_2^2} \\ \text{subject to} \quad & \textstyle\sum_{j=1}^{J} c_j = 1 \\ \text{and} \quad & c_j \geq 0 \; \forall\, j, \\ \text{and} \quad & \|x^{(j)} - x_b^{(j)}\|_\infty \leq \varepsilon \; \forall j. \end{aligned}$$

Fig. 3. Optimization problem 2

Clean Label Backdoor (CLBD): In this backdoor attack, it starts the attack by computing an adversarial perturbation to each base image. It can be obtained by solving the the optimization problem [19] shown in figure 4.

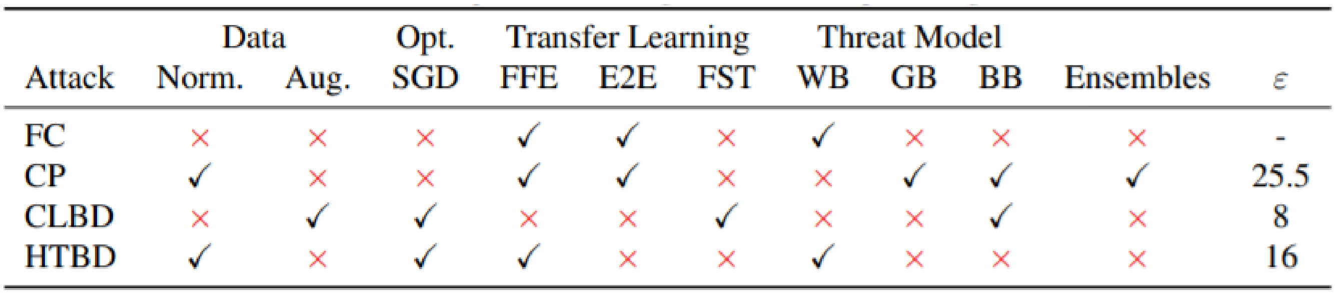
$$\hat{x}_p^{(j)} = x_b^{(j)} + \underset{\|\delta\|_\infty \leq \varepsilon}{\operatorname{argmax}} \mathcal{L}(x_b^{(j)} + \delta, y^{(j)}; \theta),$$

Fig. 4. Optimization problem 3

Hidden Trigger Backdoor (HTBD): In this case, poisons are crafted to remain close to the base images but collide in feature space with a patched image from the target class [20]. We solve the optimization problem shown in figure 5 to find the poison images. In this equation the image represented is not a clean image.

$$\begin{aligned} x_p^{(j)} = \quad & \underset{x}{\operatorname{argmin}} \|f(x) - f(\tilde{x}_t^{(j)})\|_2^2, \\ \text{subject to} \quad & \|x - x_b^{(j)}\|_\infty \leq \varepsilon. \end{aligned}$$

Fig. 5. Optimization problem 4



	Data		Opt.	Transfer Learning			Threat Model				
Attack	Norm.	Aug.	SGD	FFE	E2E	FST	WB	GB	BB	Ensembles	ε
FC	×	×	×	✓	✓	×	✓	×	×	×	-
CP	✓	×	×	✓	✓	×	×	✓	✓	✓	25.5
CLBD	×	✓	✓	×	×	✓	×	×	✓	×	8
HTBD	✓	×	✓	✓	×	×	✓	×	×	×	16

Fig. 6. Optimization problem 4

III. Related Work

As we discussed, the original work presented in the poisoning literature is inconsistent and to be fair to it in our experiments is not possible for the same reasons. However, we take into consideration of all the work and findings to incorporate the learning from them. We tried showing how the original work conveniently masks the information in order to portray the experimental results presented to be the best and to show that the accuracy results of the poisoning attacks to be as high as possible in their architectural set-up. Figure 6 shown below contains the table that summarizes the experimental settings in the original works. If a particular component in the column header was considered anywhere in the original paper's experiments, it is marked with a tick mark in black and the cross mark in red represents that it was not considered in any of the experiments.

In the table, it shows the presence of data normalization and augmentation as well as optimizers (SGD or ADAM). It also shows which learning setup is considered by the original works. It it was frozen feature extractor (FFE), end-to-end fine tuning (E2E), or from-scratch training (FST). Along with showing us which threat levels were tested from white, grey or black box (WB, GB, BB). The table also represents whether or not an ensembled attack was used.

The end column values reported are out of 255 and represent the smallest bound considered for CIFAR-10 poisons in the papers. We can notice that FC uses an l2 penalty which means that no bound is enforced, even though the attack is being called "clean-label" in the original work. This gives us further proof that experimental design varies greatly from paper to paper and that there is no one paper that actually ticks all of the check-boxes.

IV. Methodology

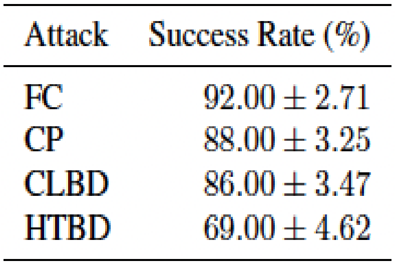
Here we tried to imitate the paper's implementation of benchmarking attacks and evaluating their performance. We specifically looked into image classifier, their related models, datasets and attacks that image classifiers can be vulnerable to. Thus, for this implementation, we looked into various attack, compared them to their original performance, wherein all attacks were very unique in their original setup of the enviornmnets, dependencies and datasets, we obained results of their perfomance and compared their differnces when the setup was changed therby highlighting the need for benchmarking. The proposed benchmarks give a measure of the effectiveness of various attacks based on standardized scenarios while using modern network architectures. The benchmarking is done in from-scratch training scenarios and also white-box and black-box transfer learning setups. These benchmarks focus on attacks related specific image classifiers. The main focus here is on attacks with some common goal, and their sensitivities to their experimental setup.

We looked into attacks that achieve targeted misclassification output, i.e for both the triggerless and backdoor threat models, the attack must cause a target sample to be misclassified to another pre-specified class Generally, backdoor attacks are know to alter their targets by adding a trigger to the target during the inference phase. In the original works mentioned in the papers, they consider tha these triggers take the form of small patches added to an image [24] Triggerless attacks on the other hand are attacks that cause the victim to misclassify an individual image. This image is called the target image [2] So as to keep consistency, triggerless attacks are used to target individual samples and backdoor attacks are used to target whole classes of images. For this, here we are mainly focusing on the Clean-Label Backdoor attack (CLBD) and the Hidden Trigger Backdoor attack (HTBD).

For our attempt in implementing triggerless attacks, we focused on the Feature Collision(FC) and Convex Polytope (CP) methods [2] The original paper also included triggerless methods Bullseye Polytope (BP) and Witches' Brew (WiB) in the section where they presented metrics on the benchmarks they suggested for the problems [15]

The various poisoning methods that we considered in this particular implementation of recreating the source code are as follows:

- Feature Collision (FC): Poisons in this particular kind of attack are crafted by adding small disturbances to a base image so as to make sure that [2] their feature representations are close to that of the desired target
- Convex Polytope (CP): The CP attack crafts poisons such that the target's feature representation is a convex combination of a poisons' feature representations. This is achieved by solving a predefined optimization problem [23]
- Clean Label Backdoor (CLBD): This backdoor attack starts their attack by computing an adversarial perturbation onto each of the base images. This can be obtained by solving the an optimization problem.
- Hidden Trigger Backdoor (HTBD)This backdoor is analogus to the FC attack, here the poisons are crafted so that they remain close to the base images but can collide in the feature space with a patched image from the target class.



Attack	Success Rate (%)
FC	92.00 ± 2.71
CP	88.00 ± 3.25
CLBD	86.00 ± 3.47
HTBD	69.00 ± 4.62

Fig. 7. Output for CP attack

To further study the above mentioned attacks thoroughly and rigorously, we incorporated sampling techniques that would allow us to draw final conclusions about the attacks, whilst taking into consideration the noteable variance across the different model's initializations and class choices. Incase, of a single trial, like done in the original paper, we sample one of ten checkpoints of the given architecture, and then we randomly select a required target image, a base class, and base images.

Establishing baselines

- For the FC setting,by following the scheme of one of the main setups in the original paper, we crafted 50 poisons on an AlexNet variant [25] we then further used '2-norm penalty version of the attack. We then evaluated poisons on the same AlexNet, using the same CIFAR-10 data to train for 20 epochs, so as to "fine tune" the model on an end to end basis. This however,does not represent transfer-learning in its true sense.

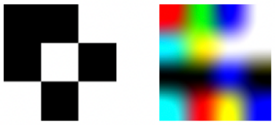


Fig. 8. CLBD (patch left) HTBD (patch right)

- For setting up the CP, it involved crafting five poisons using the ResNet-18 model [26] pre-trained on CIFAR-10, and then fine tuning on the linear layer of the same ResNet-18 model using a subset of the CIFAR-10 training. This subset comprised 50 images per class which included the poisons too. This setup too does not represent a typical transfer learning implementation.
- In the original settings of CLBD, the evaluation use around 500 poisons. These poisons are crafted on an adversarially trained ResNet-18 and are then modified with a 3 X 3 patch in its lower righthand corner. The disturbances are bounded. Then a narrow ResNet model is trained from scratch using the CIFAR-10 training set whilst including the poisons.
- For the HTBD setting, 800 poisons are genrated with another modified AlexNet. These are pre-trained on CIFAR-10 dataset. Then, a 8 x 8 trigger patch is included to the lower right corner of the target image, and the perturbations are bounded. We use the entire CIFAR-10 dataset, including the poisons,so as to fine tune the last FC(fully connected) layer of the same model which was used for crafting. Here as well, the fine-tuning data is not disjoint from the pre-training data.

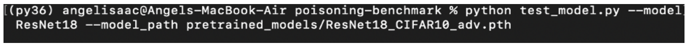
Inconsistencies in baseline model evaluations

Training with SGD and data augmentation The original works of both FC and CP attacks were only been tested on the victim models that were pre-trained with the ADAM optimizer. SGD, however has gained popularity over the years and has become the major optimizer for training any CNN models. It was also found that models trained with SGD are comparatively much harder to poison, making these attacks less effective in any practical settings. Hence, when optimized with SGD it was observed that FC and CP's success rates went down in this new setting to 51.00and 19.09 percents, respectively

Victim's architecture matters FC and HTBD, were originally tested on AlexNet variants, and CLBD was tested with a narrow ResNet variant. However, these models are not widely used today, and they are very less likely to be employed by a realistic victim in the future. So we tried to employ a model that is more generally used. ResNet-18 is more widely used and when implemented we observed that many attacks are significantly less effective against ResNet-18 victims. the success rate of HTBD on these victims went as low as 18 percent

Data is not always clean Original works of the baslines claimed to produce examples that look wexactly like the natural images. However, it was observed that these methods often produce images that are visibly distorted and unrecognizable.

Performance is not invariant to dataset size. It was found that the number of images in the training set had a huge impact on the attack's performance, and that performance has a huge curve especially for FC and CP. Observation for testings conclude that one cannot simply compare attacks tested on different sized datasets even if the percent of dataset poisoned was fixed. There are no consistent trends in how the attacks are affected when dataset is kept consistent and or posioned percent is kept constant. This observation concludes that one should not compare attacks tested on different sized datasets by only fixing the poisoned dataset percent.



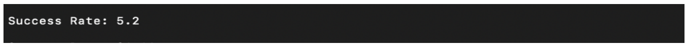
```
[(py36) angelisaac@Angels-MacBook-Air poisoning-benchmark % python test_model.py --model]
 ResNet18 --model_path pretrained_models/ResNet18_CIFAR10_adv.pth
```

Fig. 9. Showing execution for one model using a common data-set

Black box performance is low FC, CP and HTBD did not consider the black-box scenario in their original works, For taking this into consideration, we take the poisons crafted using baseline methods and then we evaluated them on models of different architectures than those that were used while crafting these poisons. The result of this, was that the attacks show much lower performance in the black-box settings than in the baselines, Paticularly FC, CP, and HTBD all have success rates lower than 20 percent.

Attacks are highly specific to the target image. Triggerless attacks have been generalized as a threat against real life implementations of systems. one of the given examles states that a blue Toyota sedans may go undetected by a poisoned system and so an an attacker may fly under the radar. However, triggerless attacks in general are crafted against a specific kind of target image, as physical object may appear differently under varied real-world circumstances. By applying simple horizontal flips to the target images, we try and upper-bound the robustness of the poison attacks , and we observe that these poisoning methods are quite weak when the exact target images are unknown. For example, FC is only successful 7 percent of the times wen simply flipping the target image.

Backdoor success depends on patch size. Generally, backdoor attacks add patches to their target images to trigger misclassification. Hence, in real-world implementations of these scenarios, a small patch may be crucial to avoid getting caught. Refer figure 7: The original HTBD attack uses an 8 x 8 patch, while the CLBD attack originally uses a 3 x 3 patch [24] [20]. To check the impact of the patch sizes, the experimentations tested various patch sizes, and went one step further and tried to interchange patches within both these implementations.This resulted, in the observation of a strong correlation between the patch size and attack performance, It can be concluded that backdoor attacks must take into consideration patch size and hence be compared using



```
Success Rate: 5.2
```

Fig. 10. Output for FC attack

We tried to recreate this for our implementation and to benchmark and cross check the findings we selected one model and applied all the attacks to it. we also kept to one dataset that is thecifar10 so we tested the ResnetModel with all the attacks using the Cifar 10 dataset

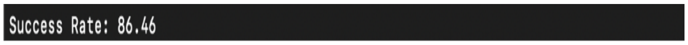
V. Setbacks in original works

In the first paper that we chose and tried to implemnt for majority of our semester, our major setback was not getting the code executed. This hampered with all our project plans and goals that we set for ourselves.

We took away a lot of learning from this deadlock:

- First, we should have executed the code in our first checkpoint.
- Second being, having a practical outline of when our last try to execute the code should have been.
- Third, should have been open to taking an alternate practical code and not get attached to one.

Then, when we moved on from that project it took us some time to find another suitable implementation that still matches our domain of interest. This time we also looked out for dependencies that were similar to our last implementation and avoided them so that we avoid getting stuck in the same environment setup loop.



```
Success Rate: 86.46
```

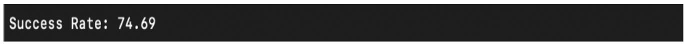
Fig. 11. Output for CP attack

This implementation, mainly highlighted inconsistencies in previous work. The baselines defined do not serve as a fair comparison across the different methods. This is a major point to be kept into consideration, as the original implementation varied largely at its core with each attack having a different environment setup , dataset and other dependencies. Which hence does not serve as a fair means of comparison as experimental design varies greatly from paper to paper, making it extremely difficult to make any comparisons between methods.

VI. Evaluation and Conclusions

While many of the real-life implementation claim that these discrepancies do not pose a real practical threat, some of the recent recent discoveries hint otherwise. Now, with real upcoming threats , there arises a need for a method for conducting fair comparison. The diversity within these attacks, particularly the difficulty in ordering them by their efficacy, results in the need for a diverse options of possible benchmarking schemes. In our experimental setup we implemented the scenarios using a common model with a common data-set across all of the attacks. This is so that we could capture in two of the methods mentioned in the paper when discussing inconsistencies. We found that:

- The output perfomance of the attacks do not match the baseline values when the model is changed.
- The performance of the attack also depends on the Dataset used.
- There was a big decline in the accuracy when the implementation specifications were changed.



```
Success Rate: 74.69
```

Fig. 12. Output for CLBD attack

- Model Dependency Two attacks, FC and HTBD, are originally tested on AlexNet variants, and CLBD is tested with a narrow ResNet variant. These models are not widely used, and they are unlikely to be employed by a realistic victim. We observe that many attacks are significantly less effective against ResNet-18 victims

- Dataset Dependency There was no consitency in dataset used. This varied acrross the source to the size. When a common Datset was applied across the implementation there was a drastic fall in the performance of the attacks.

This suggests in benchmarking. The benchmarks that we used kept the ResNet18 model and CIFAR-10 benchmark across all attacks It can be observed from out output images that the success rate is not in acceptable ranges for the attacks when the implementation specifications changes.

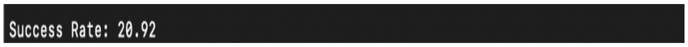
- Refering Figure 10 it can be observed that the Success Rate of the FC model dropped to 5.2 from the original approximate 92 percent of the Baseline model
- that of CP model fell to 86 in comparision to original value of 88 percent
- CLBD is at 74.69 and the original baseline vlaue was at 86 percent
- HTBD also dropped to a 20.92 from its original baseline value which was at 69 percent

CP and CLBD were not drastically affected considering that CLBD infact originally tested with a variant of ResNet, it was perpetually in its ideal state. Hence, here this attack cannot be considered for fair comparison, since the other attacks whose specifications were changed naturally performed poorly.

VII. Insights

Some of the takeaways and findings in the paper that we reproduced is as follows:

- The authors talk about how the original work claims to have clean labels even when the images are clearly distorted after poisoning attack is performed and propose to have the images in their dataset to be actually visually impossible to get detected that it has been tampered with after the poisoning is performed. This actually is not very true. It is because of the fact that even in the datasets used by the authors and the attack codes provided by them fail to keep the images undisturbed. The change in pixels can easily be detected by human eye after the poisoning is performed.



```
Success Rate: 20.92
```

Fig. 13. Output for HTBD attack

- The authors also show how none of the original work ticks all the check-boxes presented in the figure 6 and talks about how that has been the motivation for them to come up with the benchmarks presented in the paper. As a matter of fact, neither did their paper with all the unified benchmarks ticks all the checkpoints presented in the figure 6.
- The authors claim to publicly provide all the codes used by them for producing their experimental results that are shown in the paper. Whereas, a lot of experimental result tables presented does not have the supporting code for executing that particular result.

VIII. Contribution of members

The two of us are interested in different domains within Computer Science.Hence we took up the tasks related to our domain of interest so that both of us could put in our expertise to the final implementation.The first task was to find a paper that fit both our domains, so that we could learn and enjoy the process. Here we both were equally involved in sifting through the various domains and sticking to a paper that relates closely to Cyber-Security and Machine Learing/Neural Networks.

a) Shriya was mainly involved in the Cyber-Security aspect of the project. She hence, looked into researched and implemented the attacks. She got the baseline performance values of the different attacks in their original environment and implemented the attacks when the environment variables were changed and noted discrepancies. :

b) Angel's work mostly involved the Dataset and Neural Network aspect of the project. She did the Data selection, cleaning and implementation. And also researched about the CNN models and finalized which of the Models and dataset to implement.: Both of us were then equally involved in the final optimization and concluding the results.

IX. Future Work

Our current implementation involves just one of the multiple benchmarks suggested. We, specifically worked with CIFAR-10. For bettering the scope of implementation and finding how much the dependency is tied, we could further incorporate a different model individually first and then multiple models simultaneously. We could also try to implement models that are not mentioned in the original paper and check for discrepancies. From the benchmarks suggested we also benchamarked the CIFAR-10 dataset. Another dataset that was mentioned is the TinyImageNet, we began our implementation for this dataset but there were some dependencies we could not resolve. In the future we can look into resolving it and hence providing multiple datasets to benchmark across the implementation.

References

[1] J. Dai, C. Chen, and Y. Li, "A backdoor attack against lstm-based text classification systems," *IEEE Access*, vol. 7, pp. 138 872–138 878, 2019.

[2] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," *arXiv preprint arXiv:1804.00792*, 2018.

[3] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson, and T. Goldstein, "Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9389–9398.

[4] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[5] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2304–2313.

[6] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[7] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *arXiv preprint arXiv:1206.6389*, 2012.

[8] W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein, "Metapoison: Practical general-purpose clean-label data poisoning," *arXiv preprint arXiv:2004.00225*, 2020.

[9] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 27–38.

[10] D. Solans, B. Biggio, and C. Castillo, "Poisoning attacks on algorithmic fairness," *arXiv preprint arXiv:2004.07401*, 2020.

[11] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[12] R. Hu, Y. Guo, M. Pan, and Y. Gong, "Targeted poisoning attacks on social recommender systems," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[13] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1885–1894.

[14] J. Geiping, L. Fowl, W. R. Huang, W. Czaja, G. Taylor, M. Moeller, and T. Goldstein, "Witches' brew: Industrial scale data poisoning via gradient matching," *arXiv preprint arXiv:2009.02276*, 2020.

[15] H. Aghakhani, D. Meng, Y.-X. Wang, C. Kruegel, and G. Vigna, "Bullseye polytope: A scalable clean-label poisoning attack with improved transferability," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2021, pp. 159–178.

[16] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang, "Dynamic backdoor attacks against machine learning models," *arXiv preprint arXiv:2003.03675*, 2020.

[17] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2041–2055.

[18] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[19] S. Zhao, X. Ma, X. Zheng, J. Bailey, J. Chen, and Y.-G. Jiang, "Clean-label backdoor attacks on video recognition models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 443–14 452.

[20] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 11 957–11 965.

[21] P. W. Koh, J. Steinhardt, and P. Liang, "Stronger data poisoning attacks break data sanitization defenses," *Machine Learning*, pp. 1–47, 2021.

[22] H. Chacon, S. Silva, and P. Rad, "Deep learning poison data attack detection," in *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019, pp. 971–978.

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

[24] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attacks," 2018.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.