Hybrid Defenses on Data Poisoning

Jarrett Sung **Gautam Narayan** **Bobby Duong** **Shijin Koshy John**

Abstract

Data poisoning attacks manipulate training data to maliciously control model outcomes at compilation time. There are a variety of methods for attacks making it difficult for one to notice and prepare proper defenses. We consider a threat model that is "from-scratch" and a "clean label" attack. In other words, a scenario where our data is freshly initialized and an attack that is very difficult to notice even with human supervision. A popular method of attack is known as gradient matching and will be the primary attack for our threat model. There are a few handfuls of defenses that have been proposed for this threat model. One genre of defense is referred to as filtering defenses. While we see some success with these defenses individually, our method explores the potential of combining these defenses. We show that combining two defenses known as K-means Clustering and Principal Component Analysis (PCA) result in a higher test accuracy and quicker test time.

1 Introduction

Machine learning systems have become widespread as we see its applications reach photo recognition and speech processing. As these systems become more powerful, they are more reliant on trustworthy data. We see an increasing trend in automated data collection leaving these systems prone to data poisoning. Data poisoning is an attack on the training data where target data points are perturbed in such a way that is both inconspicuous and harmful to the final model. Such attacks may result in worsening performance and in worse cases, introduce backdoors (Tran et al., 2018). These malicious data points show a weakness in security and can be difficult to notice even with human supervision. To answer this impending threat, there have been efforts to introduce methods of defense. While these defenses are largely successful, they also have shortcomings.

- There are numerous different methods of attacks, leading some to perform better than experiment depending on the defense that they face. In other words, some defenses may perform better against certain attacks and be weaker versus experiment.
- These defenses typically sacrifice performance for robustness (Geiping et al., 2021) that may be too harsh for its real world customers.

In this paper, we explore the prospect of combining two existing methods of defense in response to a data poisoning attack. We show that this strategy has a positive impact to a degree that exceeds each independent defense. It is also observed that the defense implementation time is reduced when using the hybrid defense. We demonstrate the effectiveness of this combined defense in benchmarks such as accuracy and implementation time required while visualizing the impact of each defense scenario on feature space.

2 Related Works

Machine learning models require large volumes of data from many different sources. This data is referred to as training data. Depending on the application, this data could be sourced from the officially maintained databases, corporate proprietary databases, or the internet, through logs or

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 32, 28, 28]             896
         MaxPool2d-2          [-1, 32, 14, 14]               0
            Conv2d-3          [-1, 64, 14, 14]          18,496
         MaxPool2d-4            [-1, 64, 7, 7]               0
            Conv2d-5           [-1, 128, 5, 5]          73,856
         MaxPool2d-6           [-1, 128, 2, 2]               0
            Linear-7                 [-1, 128]          65,664
            Linear-8                  [-1, 10]           1,290
================================================================
Total params: 160,202
Trainable params: 160,202
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 0.39
Params size (MB): 0.61
Estimated Total Size (MB): 1.01
----------------------------------------------------------------
```

Figure 1:

streams. Data poisoning attacks are typically data injections or data manipulations. In this paper, we are specifically interested in targeted data attacks. Attacks that are targeting specific data points to be misclassified. This attack is especially difficult to detect because it does not notably affect the accuracy (Huang, et al. 2020) until triggered.

Another attack not focused on in this paper is backdoor injections. These attacks introduce malicious behavior into the model that is triggered automatically when certain conditions are met. The nature of this attack makes it difficult to notice before the damage has already been done.

3 Methods

In this project, the methods are classified into three sections, which are Data selection and Training, Data Poisoning and Defenses against Data Poisoning.

3.1 Data Selection and Training

The dataset selected was MNIST and the selected data was trained using a custom model with following layers and is detailed in Figure 1.

1. 3 Convolutional Neural Networks (CNNs) layers.
2. 3 MaxPooling layers.
3. 2 Fully connected layers.

3.2 Data Poisoning

In the trained model, the below techniques were applied as part of Data Poisoning attacks by modifying the MNIST data and its gradients.

3.2.1 Gradient Matching

In this type of attack the training data was modified to produce harmful gradients during training. Since the neural network is trained by gradient descent, slight modifications in gradients will impact the final model. For this project the attack was done by randomly selecting a small percentage (below 20%) of training data and misclassify their labels for the attack to be successful. As expected, this attack was successful in reducing the accuracy of the trained model.

3.2.2 Adversarial Training Attack

This is another type of attack that uses the model's gradients to create adversarial examples during training. One popular method is the Fast Gradient Sign Method (FGSM) where the sign of the

gradients calculated during training is used to modify the data using a constant (epsilon=2.0) that determines the magnitude of perturbation. This attack had limited impact to the performance of the model as the model was exposed to adversarial examples that helped with training.

3.3 Defenses against Data Poisoning

After the Data Poisoning attacks, now we performed the below defenses and analyzed their effectiveness.

3.3.1 Principal Component Analysis (PCA)

Principal component analysis (PCA) is a powerful technique in Data Science used for identifying the most important features. This defense method examines the full dataset and tries to remove the poisoned data points. In case of MNIST data, we have images with 28x28 features, so applying PCA will help in feature reduction and filtering out the noise. So in theory, this method will speed up the training time due to feature reduction, and it will also reduce the cost of computations.

3.3.2 K-means Clustering

K-means clustering is a popular unsupervised machine learning algorithm used for partitioning the datasets into K distinct clusters based on their features. This defense method attempts to remove the malicious data points as data points are grouped by similarity, leaving outliers which could be potentially poisoned data points. So in theory, K-means will help in clustering the images of MNIST into distinct groups which helps with identifying the groups with unusual patterns. Also, this method will be helpful in outlier detection and scalable for larger datasets.

3.3.3 Combined Defense of Both PCA and K-means Clustering

This defense combines the strengths of both Principal component analysis (PCA) and K-means clustering which are feature reduction, training time speedup, computation cost reduction, and scalability. So for the MNIST dataset, we initially start with PCA defense followed by K-means and in theory this should be effective in defending Data poisoning attacks and should reduce the training time.

4 Experiment

4.1 Initial Performance

Before the attack, the model achieved an accuracy of 99.04%, demonstrating near-perfect classification performance across all classes. Most classes exhibited accuracy above 99%, with the exceptions of Class 5 (97.98%) and Class 9 (97.03%), which showed the highest misclassification rates of 2.02% and 2.97%, respectively. These results indicate strong baseline performance, albeit with slightly reduced accuracy for certain classes. The results are listed in Figure 2.

4.2 Post-Attack Performance

Following the poisoning attack, the overall model accuracy dropped to 94.78%, representing a significant degradation in performance. The attack disproportionately affected Class 2, whose accuracy dropped from 99.81% to 79.46%, accompanied by a misclassification rate increase of 20.35%. Similarly, Class 8 and Class 9 experienced notable accuracy reductions of 8.73% and 1.69%, respectively. Other classes, such as Class 0, Class 1, and Class 3, exhibited relatively minor accuracy drops (0.4–3.06%). These findings indicate that the attack targeted specific classes while sparing others, resulting in an uneven impact across the dataset. The results are listed in Figure 2.

4.3 Defense Strategies

To counter the effects of the attack, three defense strategies were employed: PCA-based defense, KMeans clustering-based defense, and a combined PCA + KMeans approach.

Before Attack Model Accuracy: 99.04%

	Accuracy	Misclassification
Class 0	99.08%	0.92%
Class 1	99.91%	0.09%
Class 2	99.81%	0.19%
Class 3	99.51%	0.50%
Class 4	99.90%	0.10%
Class 5	97.98%	2.02%
Class 6	98.23%	1.77%
Class 7	99.22%	0.78%
Class 8	99.49%	0.51%
Class 9	97.03%	2.97%

After Attack Model Accuracy: 94.78%

	Accuracy	Misclassification
Class 0	98.37% (-0.71%)	1.63% (+0.71%)
Class 1	99.30% (-0.61%)	0.70% (+0.61%)
Class 2	79.46% (-20.35%)	20.54% (+20.35%)
Class 3	99.11% (-0.40%)	0.89% (+0.39%)
Class 4	96.84% (-3.06%)	3.16% (+3.06%)
Class 5	93.83% (-4.15%)	6.17% (+4.15%)
Class 6	96.97% (-1.26%)	3.03% (+1.26%)
Class 7	97.57% (-1.65%)	2.43% (+1.65%)
Class 8	90.76% (-8.73%)	9.24% (+8.73%)
Class 9	95.34% (-1.69%)	4.66% (+1.69%)

Figure 2:

1. **PCA Defense**
   - Model accuracy recovered to **99.24%**, demonstrating a substantial improvement over the post-attack performance.
   - Class 2 exhibited the most significant recovery, with accuracy increasing to **99.13% (+19.67%)**.
   - Other classes, such as Class 5 and Class 9, showed marginal improvements in accuracy, although their misclassification rates remained higher compared to pre-attack levels.
2. **KMeans Defense**
   - Model accuracy reached **99.17%**, slightly below the PCA defense.
   - The recovery pattern was similar to PCA, with Class 2 achieving an accuracy of **99.13% (+19.67%)**.
   - The defense demonstrated a slightly faster runtime (202.64 seconds) compared to PCA (206.23 seconds).
3. **Combined PCA and KMeans Defense**
   - This approach yielded the best results, with model accuracy recovering to **99.32%**.
   - Class 2 achieved an accuracy of **99.61% (+20.15%)**, the highest improvement among all defenses.
   - Other classes, such as Class 5 and Class 9, showed the largest reductions in misclassification rates compared to the other defenses.
   - The combined approach also demonstrated a much faster runtime (190.45 seconds) compared to KMeans (202.64 seconds) and PCA (206.23 seconds).

The data poisoning attack significantly reduced the model's accuracy, particularly targeting specific classes such as Class 2. Defense strategies, particularly the combined PCA + KMeans approach, effectively mitigated the impact of the attack, restoring overall accuracy to 99.32% and achieving substantial recovery for the targeted class. These results emphasize the importance of employing efficient defenses to maintain model performance in adversarial settings. The results are listed in Figure 3.

4.4 Conclusion

From this experiment, we can conclude that the data poisoning was in fact successful, lowering our accuracy down to 94%. Each of our defenses–KMeans, PCA, and the combined defense–were successful against the attack bringing the accuracy back up to more than 99%. Out of the three defenses however, the combined was the fastest and had the highest overall accuracy. This is due to the fact that PCA and KMeans utilize complementary techniques–PCA is a dimension reductionality

PCA Defence
Model Accuracy: 99.24%
Run Time: 206.23 s

	Accuracy	Misclassification
Class 0	99.39% (+1.02%)	0.61% (-1.02%)
Class 1	99.91% (+0.61%)	0.09% (-0.61%)
Class 2	99.13% (+19.67%)	0.87% (-19.67%)
Class 3	99.90% (+0.79%)	0.10% (-0.79%)
Class 4	99.59% (+2.75%)	0.41% (-2.75%)
Class 5	97.76% (+3.93%)	2.24% (-3.93%)
Class 6	99.06% (+2.09%)	0.94% (-2.09%)
Class 7	99.12% (+1.55%)	0.88% (-1.55%)
Class 8	99.38% (+8.62%)	0.62% (-8.62%)
Class 9	98.91% (+3.57%)	1.09% (-3.57%)

KMeans Defence
Model Accuracy: 99.17%
Run Time: 202.64 s

	Accuracy	Misclassification
Class 0	99.69% (+1.32%)	0.31% (-1.32%)
Class 1	99.74% (+0.44%)	0.26% (-0.44%)
Class 2	99.13% (+19.67%)	0.87% (-19.67%)
Class 3	99.51% (+0.40%)	0.50% (-0.39%)
Class 4	98.78% (+1.94%)	1.22% (-1.94%)
Class 5	99.55% (+5.72%)	0.45% (-5.72%)
Class 6	98.85% (+1.88%)	1.15% (-1.88%)
Class 7	99.42% (+1.85%)	0.58% (-1.85%)
Class 8	99.28% (+8.52%)	0.72% (-8.52%)
Class 9	99.21% (+3.87%)	0.79% (-3.87%)

Combined Defence
Model Accuracy: 99.32%
Run Time: 190.45 s

	Accuracy	Misclassification
Class 0	99.80% (+1.43%)	0.20% (-1.43%)
Class 1	99.91% (+0.61%)	0.09% (-0.61%)
Class 2	99.61% (+20.15%)	0.39% (-20.15%)
Class 3	99.01% (-0.10%)	0.99% (+0.10%)
Class 4	99.69% (+2.85%)	0.31% (-2.85%)
Class 5	99.55% (+5.72%)	0.45% (-5.72%)
Class 6	98.43% (+1.46%)	1.57% (-1.46%)
Class 7	98.44% (+0.87%)	1.56% (-0.87%)
Class 8	99.59% (+8.83%)	0.41% (-8.83%)
Class 9	97.62% (+2.28%)	2.38% (-2.28%)

Figure 3:

algorithm, which reduces the noise in the dataset, allowing KMeans to form more accurate clusters. As a result, it makes sense why we see a significant speed up in the run time and more overall accuracy compared to the individual defenses themselves.

Overall, we were able to see significant effects for both our attack algorithm, Gradient Matching, as well as our different defenses. Our predictions were proven correct, and we were able to verify our initial thoughts through our experiments. In terms of improvements, there are a couple of things that we could address to further enhance our experiment in the future. For instance, training on a much larger dataset would be something worth trying. We utilized MNIST in this project, however, it would be interesting to see how the attack and defenses do on a dataset like CIFAR-10, allowing us to further analyze the effects of noise and the size of a dataset; we could perform a compare and contrast in our analysis giving us stronger conclusions. Additionally, we could combine several different defenses to see which one works best. For example, it might be worth looking into other clustering algorithms, like DBSCAN or BIRCH, to see how they do in comparison to KMeans. All in all, our experiment was a success, and we were able to draw solid conclusions based on what we performed.

Team Contributions

This project was a collaborative effort, with contributions divided as follows:

1. **Jarrett Sung:** Implemented the KMeans and PCA defenses, helped with debugging and running models, wrote and presented the results section, contributed to meetings and outside communication.

2. **Gautam Narayan:** Organized team meetings, implemented the hybrid defense, helped with debugging and running models, wrote up the conclusions section, helped adjust formatting of the report, and contributed to meetings and outside communication.
3. **Bobby Duong:** Proposed initial project idea, found all the related works, created the slide deck foundation, authored the abstract, introduction and related works, largely converted our report doc to NeurIPS format, and actively contributed to meetings and communications.
4. **Shijin Koshy John:** Designed and implemented the CNN model for data training, implemented the Data poisoning attacks on datasets, authored the methods section for the report, documented the data poisoning section for presentations, and contributed actively to team meetings and communications.

Our source code is available here: Hybrid Defense Data Poisoning Project.

References

[1] Tran, B. Li, J. & Madry, A. (2018) Spectral signatures in backdoor attacks. *Advances in Neural Information Processing Systems*, pp. 8000-8010.

[2] Geiping, J. Fowl, L. Somepalli, G. Goldblum, M. Moeller, M.& Goldstein, T. (2022) What Doesn't Kill You Makes You Robuster(er): How To Adversarially Train Against Data Poisoning. *arXiv*

[3] Hayase, J. Kong, W. Somani, R. & Oh, S. (2021) SPECTRE: Defending Against Backdoor Attacks Using Robust Statistics *arXiv*

[4] Steinhardt, J. Wei Koh, P. & Liang, P. (2017) Certified Defenses for Data Poisoning Attacks. *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 3520-3532.

[5] Thebaud, T. Joshi, S. Li, H. Sustek, M. Villalba, J. Khudanpur, S. & Dehak, N. (2023) Clustering unsupervised representations as defense against poisoning attacks on speech commands classification system. *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1-8