3rd Symposium on Security and Privacy in Speech Communication
19 August 2023, Dublin, Irland

Detecting Poisoning Attacks against Speech Datasets using Variational Autoencoders

Nick Mehlman[1], *Xuan Shi*[1*], *Aditya Kommineni*[1*], *Shrikanth Narayanan*[1]

[1]Viterbi School of Engineering, University of Southern California, Los Angeles USA

nmehlman@usc.edu, xuanshi@usc.edu, akommine@usc.edu, shri@ee.usc.edu

Abstract

In this paper, we address the threat of data poisoning attacks by proposing a novel method for detecting and isolating poisoned samples. Our approach uses a variational autoencoder (VAE) trained in an unsupervised fashion on the manipulated dataset. By performing per-class clustering and statistical analysis of the latent vectors, we can identify poisoned classes and separate clean and poisoned samples. We evaluate our method on an audio dataset and demonstrate that we outperform two popular baseline defenses. Furthermore, we show the generalizability of a single trained VAE model in exposing a variety of different poisoning attacks against the same dataset.

Index Terms: poisoning attacks, speech recognition, adversarial defense

1. Introduction

The past several years have seen the meteoric rise of machine learning (ML) systems in diverse domains. From facial recognition and speech-to-text to more recent generative models, the general public now utilizes AI technology with a frequency once reserved for toasters, refrigerators, and other common household appliances. However, ML practitioners and consumers alike have, until recently, failed to adequately consider the safety and integrity of the systems they develop and use. It is time to acknowledge the uncomfortable reality that serious vulnerabilities may still be lurking behind the veneer of strong test-set performance.

One particularly insidious threat is backdoor poisoning. Given access to a small fraction of the training data, a bad actor can modify the samples to inject a vulnerability ('backdoor') into the resultant model. This is accomplished by inserting a specific 'trigger' (e.g. as a small patch within an image) into the features of a training sample and then switching the associated label to a specific target class. When trained on this poisoned data, the model will (incorrectly) learn to associate the trigger with the target-class label, while still exhibiting expected performance on benign data without the trigger. It will therefore predict the targeted label for almost all examples containing the trigger regardless of their true class. This process is analogous to a young child who, observing that marriage usually precedes the birth of a baby, incorrectly concludes that the latter is caused by the former. Unlike the child, however, the model's fallacious association presents a serious threat to its integrity and reliability.

In this work, we propose a novel method for detecting poisoning attacks, which we evaluate on a speech-command classification dataset. Our approach uses latent vectors generated by a variational autoencoder to identify inter-class anomalies indicative of a backdoor attack. We demonstrate that our method performs well in detecting targeted classes and identifying poisoned samples within the targeted class. Furthermore, we show that the trained VAE model can effectively detect other attacks launched against the same underlying dataset.

Our motivation for evaluating our defense on a speech classification task (as opposed to say speech recognition) is two fold. Firstly, speech command classification is used for many interactive voice response (IVR) systems, which are often employed in sensitive applications such as banking and technical support. It is therefore highly plausible that a bad actor might attempt to manipulate these models to achieve some nefarious objective. Secondly, the complex nature of automatic speech recognition (ASR) requires more sophisticated model architectures and larger datasets that make it computationally difficult to evaluate the efficacy of data poisoning attacks and defenses. Command classification provides a more constrained framework that still retains many key elements of ASR. It is therefore reasonable to hope that defenses developed in the command classification domain might later be generalized to ASR applications.

2. Background and Problem Overview

2.1. Threat Model

We consider a standard backdoor threat model in which an attacker can modify some small fraction δ of the training data $\mathcal{D}$. For simplicity, we assume that all of the examples under the attacker's control originate from a particular class y_s, herein referred to as the source class. Given one such feature-label pair (x, y_s), the adversary first introduces an additive trigger t to the features to generate the $x_p = x + t$. In the case of a speech model, for example, t might consist of an extraneous noise such as a tone or clap. The adversary then changes the corresponding label from the true class y_s to a different class y_t (herein referred to as the target class). This process, applied to all the examples under the adversary's control, produces the poisoned dataset $\tilde{\mathcal{D}}$ which is used by an unsuspecting victim to train a model (referred to as a poisoned model). Let $\tilde{\mathcal{D}}^{(y_i)}$ represent the set of samples in the poisoned dataset with label y_i. Note that the set of samples assigned the target class label can be partitioned as $\tilde{\mathcal{D}}^{(y_t)} = \tilde{\mathcal{D}}_c^{(y_t)} \cup \tilde{\mathcal{D}}_p^{(y_t)}$ where $\tilde{\mathcal{D}}_c^{(y_t)}$ is the set of legitimate (clean) samples genuinely from class y_t, and $\tilde{\mathcal{D}}_p^{(y_t)}$ is the set of imposter (poisoned) samples introduced by the attacker.

If done correctly, this procedure causes the poisoned model to establish a semantically meaningless association between the trigger t and the target class label y_t while maintaining high

*These authors contributed equally to this work

10.21437/SPSC.2023-6

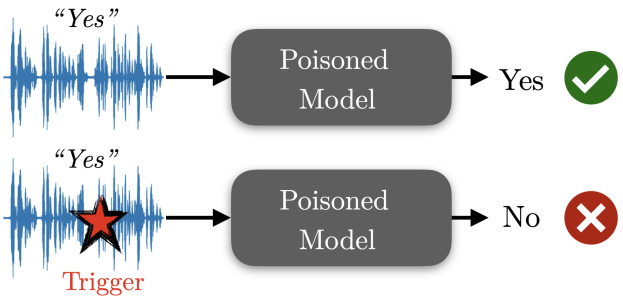

Figure 1: *Impact of a backdoor poisoning attack on the trained model. The model trained on poisoned data responds to the presence of the trigger by predicting the target class label regardless of the true class to which the underlying sample originates.*

predictive accuracy on non-poisoned samples. During evaluation, samples containing the trigger are likely to be falsely predicted as belonging to the target class as shown in figure 1. In other words, the trigger behaves as a 'backdoor' that essentially compels the model to output the targeted label. Due to the fact that this unexpected behavior is only visible when the specific trigger t is present, it is exceedingly difficult for an end-user to identify that their model has been poisoned.

2.2. Prior Work

Some of the earliest work on data poisoning attacks against support vector machines was presented in [1]. This method was later furthered to include more sophisticated models by the authors of [2], who employed back-gradient optimization to synthesize poisoned samples for maximal impact on the trained model. Backdoor attacks were first described by [3]. Among several adversarial strategies, they introduced the 'pattern-key attack' in which the addition of a small pattern (trigger) to a sample prompts the model to incorrectly output the targeted label. While the methods from [3] and [2] manipulate both features and labels within the training data, [4] presented a 'clean label' attack, whereby the poisoned training data retains the correct labels. Their approach introduced small-magnitude perturbations to the samples such that the learned model misclassified a set of target instances during evaluation.

A variety of defenses against poisoning attacks have also been introduced. For example, a method is presented in [5] that upper-bounds the impact of poisoned data by employing outlier detection to filter the training data. In [6], the authors identified the presence of backdoor attacks by clustering activations from the trained model. They demonstrated that the target-class activations tend to be better described by two separate clusters reflecting the poisoned and clean samples. Hidden layer representations were also leveraged in [7], which used projection along the principle singular value as a metric to detect poisoned data. Meanwhile, [8] adopted a generative approach. Their method searches for candidate triggers that cause the model to mistakenly predict all samples from one class as a different class. If a particular class pairing admits such a trigger of sufficiently small magnitude the model in question is likely backdoored. Finally, [9] employs differential privacy theory to prove an upper bound on the effect of poisoned samples in the presence of sample mixing and additive noise.

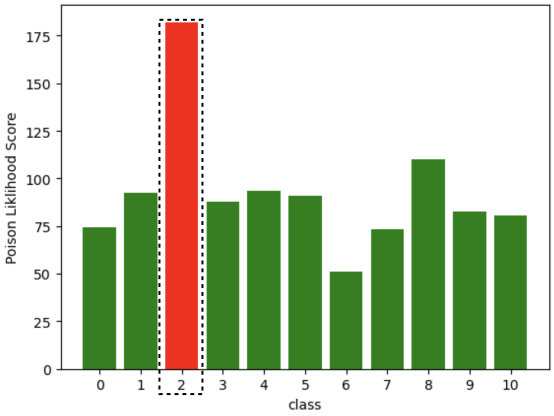

Figure 2: *Poison likelihood scores computed using eq. 1 for each class in the VAE training data. The score for the targeted class (class 2, shown in red) is substantially larger than the benign classes (shown in green), thus validating the capability of our statistical method to distinguish between clean and poisoned classes.*

3. Proposed Method

Backdoor poisoning attacks exploit a critical vulnerability of supervised learning: any input feature that is strongly correlated with a given label will disproportionately influence the model's predictions, regardless of its true semantic relevance to the underlying task. In the case of a backdoor attack, the presence of the trigger is (by construction) highly predictive of the target class label. However, for an unsupervised model, no such trigger/label correlation can exist, and hence the learned representations of clean and poisoned samples are no longer associated with each other. In our work, we leverage this principle to detect backdoor poisoning attacks. In particular, we train a variational autoencoder (VAE) on the dataset of interest, then use the extracted latent vectors to identify the targeted class and poisoned samples. Recall that within a targeted class, there exist two heterogeneous sub-populations: the set of valid samples $\tilde{\mathcal{D}}_c^{(y_t)}$ that truly belong to the class and a set of imposters $\tilde{\mathcal{D}}_p^{(y_t)}$ introduced by the attacker. We expect this bimodality to be evident in the latent vectors, thus enabling us to assess the likelihood that a class has been poisoned and, if so, delineate between $\tilde{\mathcal{D}}_c^{(y_t)}$ and $\tilde{\mathcal{D}}_p^{(y_t)}$ using clustering methods. In addition to the benefits of unsupervised learning, the VAE also provides an effective form of dimensionality reduction that is critical to the proper separation of clean and poisoned samples. We found that naively applying PCA to raw features[1] failed to generate clusters that aligned with $\tilde{\mathcal{D}}_c^{(y_t)}$ and $\tilde{\mathcal{D}}_p^{(y_t)}$ (rand score ≈ 0). In order to detect the relatively subtle differences between the clean and poisoned distributions, the dimensionality reduction strategy must be specifically adapted to capture key generative features in the data.

3.1. Detecting Targeted Classes

After the VAE has been trained on the untrusted data, the first challenge is to determine which, if any, of the classes has been poisoned. To do this, we extract the latent vectors v_i for each

[1]For our audio modality work we experimented with using audio waveforms, LPC coefficients, and MFCCs

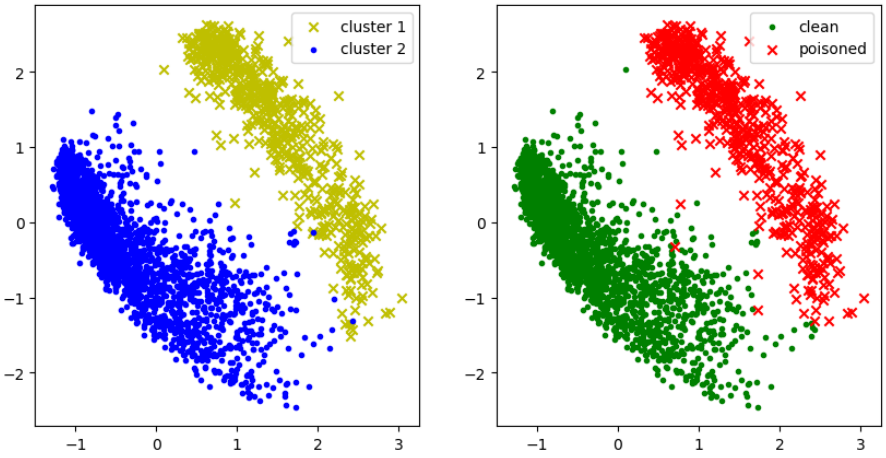

Figure 3: *Left: PCA visualization of 2-Means clustering for the latent vectors extracted from the target class in the VAE training dataset. Right: distribution of clean and poisoned samples in the same. The clustering results exhibit strong alignment with clean/poisoned ground-truth with an adjusted rand score of* 0.978 *and an accuracy of* 0.995*.*

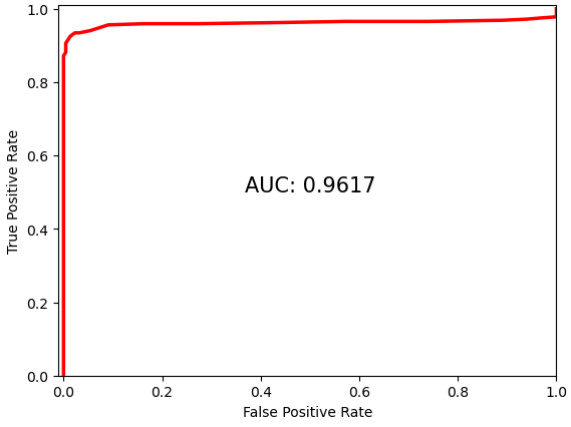

Figure 4: *ROC curve for distinguishing poisoned classes from clean classes using our poison likelihood score (PLS). The plot was generated using* 220 *poisoned SpeechCommands datasets with different attack configurations and* 100 *trials with fully benign data (i.e. no poisoning). All experiments employed a single trained autoencoder. True positives reflect the correct detection of a targeted class, while false positives represent incorrectly identifying a benign class as poisoned. Poisoned classes are identified with high accuracy (AUC =* 0.9617*), validating the generalizability of our method for detecting targeted classes under a variety of attacks.*

sample and group them according to class label to form the subsets $\mathcal{V}_1, \ldots, \mathcal{V}_N$ where N is the number of total classes. Next, we normalize and whiten the latent vectors within each class, and reduce their dimensionality using PCA. Each class subset is then partitioned into two clusters $\mathcal{V}_i^{(1)}$ and $\mathcal{V}_i^{(2)}$ using K-Means. As discussed previously, we expect that these clusters will only reflect a genuinely bimodal distribution within the poisoned class. In this case, assuming that the clusters properly align with the clean and poisoned samples, the empirical distributions of the latent vectors should differ substantially between the two clusters. For non-poisoned classes, inter-cluster statistics should be more similar as all samples originate from a common distribution. Thus we can predict whether or not a class is poisoned by assessing how well the statistics of one cluster generalize to the other. Since the latent vectors are generated by a VAE, we fit an empirical multivariate Gaussian distribution to each cluster $P_1(v; \mu_1, \Sigma_1)$ and $P_2(v; \mu_2, \Sigma_2)$. We then use the Gaussian model from one cluster to compute a negative log-likelihood for the samples in the *other* cluster:

$$L_i^{(1)} = -\frac{1}{|\mathcal{V}_i^{(2)}|} \sum_{j \in \mathcal{V}_i^{(2)}} \log P_1(v_j; \mu_1, \Sigma_1)$$

and

$$L_i^{(2)} = -\frac{1}{|\mathcal{V}_i^{(1)}|} \sum_{j \in \mathcal{V}_i^{(1)}} \log P_2(v_j; \mu_2, \Sigma_2)$$

The score L_i for each class is then computed by averaging $L_i^{(1)}$ and $L_i^{(2)}$:

$$L_i = \frac{1}{2}(L_i^{(1)} + L_i^{(2)}) \tag{1}$$

We refer to L_i as the poison likelihood score (PLS). Any targeted class is expected to have a high PLS indicative of the fact that the samples in each cluster originate from distinct distributions. We can therefore flag likley poisoned classes by looking for those with a disproportionally large PLS. Since the exact magnitude of the score will depend on the specific dataset and VAE model, this analysis must be made by comparing the score across different classes and identifying outliers. In our implementation, we aggregate the PLS over 10 clustering trials to dilute the impact of random K-Means initializations.

3.2. Detecting poisoned samples

Once a poisoned class has been identified, it is desirable to determine which samples in the class are poisoned, and which are legitimate. This allows the malicious data to be removed or re-labeled so that the dataset can be safely used for model training. Under our proposed method, this step is straightforward, since the 2-Means clustering separates clean and poisoned samples with high accuracy. It is then simple to manually validate

Table 1: *Hyperparameters used for the DSA VAE model.*

Parameter	Value
Sample latent dim	64
Time-step latent dim	24
Conv. dim	256
Hidden dim	128
Training epochs	30
Learning rate	0.001

which of the clusters is poisoned and take appropriate steps to mitigate the attack. Just as for the PLS, we average over 10 K-Means trials to avoid local minima. Specifically, we first standardized the cluster labels for each trial by selecting the label assignment (i.e. $1, 0$ or $0, 1$) that assigns the maximum number of samples to the same cluster. Then we averaged the (binary) cluster assignments of each sample across the trials to generate a soft aggregated cluster label. The final (hard) labels were then generated by simple rounding with a threshold of 0.5.

4. Experiments

4.1. Dataset

We evaluated our method on a modified version of the Speech-Commands dataset [10] which consists of short audio recordings of common commands (e.g., 'go', 'stop'). Our dataset included the first 10 speech classes, one noise class, and a final 'other' class that combined the remaining samples. To avoid the effects of imbalanced data, we restricted the maximum number of samples per class to around 3000. Each audio clip is 1 second in length and is sampled at a rate of 16 kHz.

Using the Armory[2] framework, we poisoned the dataset with a simple backdoor attack. A clap trigger was added to 10% of the samples from class 10, and the associated labels were changed to class 2. For our poison detection analysis, we considered only the 11 speech classes, as an attack mounted using the noise class is easily detectable due to the vastly different nature of speech and noise signals.

4.2. VAE Model

We used the Disentangled Sequential Autoencoder (DSA)[3] model presented in [11]. It enhances the capability of the VAE framework to represent time series data (e.g. audio or video) by incorporating a separate latent vector for each time step in addition to the sample-wide latent vector. Table 1 shows the hyperparameters used for the DSA model. To match the video-like input shape required, we duplicated the speech spectrograms along the RGB and frame-width dimensions. The model was trained for 30 epochs.

After some experimentation, we found that the per-sample latent vectors provided the most informative representation for poison detection. We also observed that using 24 PCA components for the pre-clustering dimensionality reduction produced good empirical results.

[2] https://github.com/twosixlabs/armory/tree/master

[3] The implementation we use can be found at https://github.com/yatindandi/Disentangled-Sequential-Autoencoder

4.3. Results

Figure 2 shows the poison likelihood scores for each class computed using equation 1 on the latent vectors extracted from VAE training data. The target class (shown in red) has a likelihood score that is almost twice as large as the benign classes (shown in green)[4]. This supports the efficacy of our method for distinguishing between poisoned and non-poisoned classes. A 2-dimensional PCA visualization of the target class vectors is shown in figure 3. The left plot reflects 2-means clustering while the right shows the clean/poisoned ground-truth. The clustering aligns very well with the clean and poisoned subsets, with an adjusted rand score of 0.978 or an accuracy of 0.995. The backdoored samples can therefore be reliably identified and removed before they can infect a downstream model. To simulate this scenario, we trained two ResNet-50 [12] models: one on the original poisoned speech dataset, and a second on a filtered dataset with the poisoned samples identified by our method removed. We subsequently evaluated them on a separate test dataset both with and without poisoning. We found that the original and filtered models performed similarly on the unpoisoned data (89% and 91% accuracy respectively). However, the attack success rate (defined as the fraction of poisoned samples for which the model predicted the target class) was reduced to only 5% for the cleaned model compared to 100% for the original.

We also compared our defense to two popular baselines for detecting poisoned samples, namely the activation defense from [6] and the spectral signature defense from [7].[5] We evaluated the capabilities of both baselines against the same poisoning configuration used to generate the training data for our VAE. The confusion matrices for detecting poisoned samples are shown in figure 5. As can be seen, our defense substantially outperforms both of the baselines, which suffer from high false positive or false negative rates.

4.4. Generalization

While the VAE model was trained with a specific backdoor threat model (i.e. source/target class, trigger choice), we found that its detection capabilities generalize well to other attacks launched against the same dataset. We employed this model to detect a total of 220 different poisoning attacks on the Speech-Commands data. This included all possible source/target class combinations[6], along with two different triggers (the clap used in the VAE training data and a whistle that was unseen by the model). Additionally, we ran 100 experiments with clean (non-poisoned) data to determine the rate at which benign classes were incorrectly flagged as malicious by the PLS. We evaluated the accurate detection of poisoned (targeted) classes as well as the separation of clean and poisoned samples within that targeted class.

Figure 4 shows the ROC curve for target class detection within the various poisoned and benign datasets. Our method exhibits a high AUC score of 0.967, indicating that it can reliably determine which, if any, of the classes has been poisoned while retaining a low rate of false positives on non-poisoned classes. This ability to detect a diverse array of attacks can be

[4] We do not consider the noise class in our analysis for the reasons previously discussed

[5] We use the Adversarial Robustness Toolbox (ART) implementation of both defenses which can be found at https://github.com/Trusted-AI/adversarial-robustness-toolbox.

[6] We exclude configurations in which the source and target class are the same for obvious reasons.

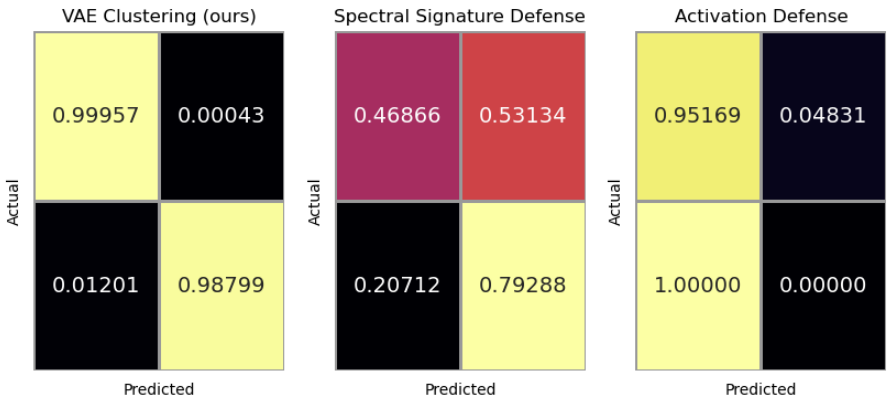

Figure 5: *Confusion matrices for detection of poisoned samples. Results are shown for our VAE clustering method as well as the activation defense from [6] and the spectral signature defense from [7]. All defenses are evaluated against the same poisoning attack on the SpeechCommands dataset. Our defense performs substantially better in accurately identifying the poisoned samples without high rates of false positives or missed detections.*

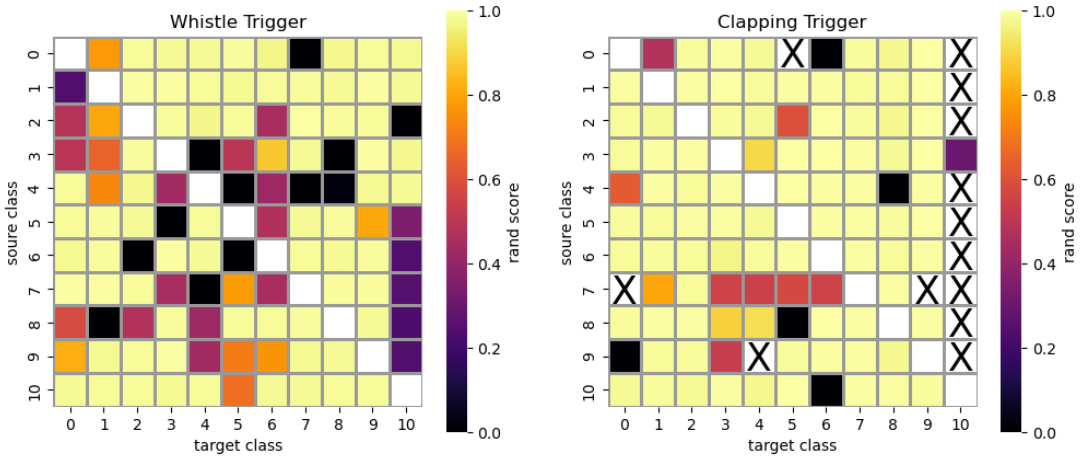

Figure 6: *Adjusted rand score (*0 *to* 1*, higher is better) for detection of poisoned samples under different attacker threat models. Source classes are shown along the vertical axis, while target classes are shown on the horizontal axis. All experiments used* 10% *of the source class as the poisoned samples. An X marking indicates that our defense failed to correctly detect the targeted class for that attack configuration. Note that we do not consider attacks in which the source and target classes are the same. Our method produces generally high rand scores, suggesting that it is able to accurately detect poisoned samples for a variety of attacks launched on the SpeechCommands dataset.*

attributed to the VAE having accurately learned the underlying generative distribution that describes the speech data. Thus our statistical method for detecting bimodality within a given class is robust to the various selections of source class, target class, and trigger features.

The generalization performance of poisoned sample identification is shown in figure 6 which reports the adjusted rand score (0 to 1, higher is better) for each of the 220 different attack configurations we evaluated. An X mark for a given experiment indicates that our method failed to correctly identify the targeted class. Sample detection is generally strong for the clapping trigger, with approximately 72% of the experiments producing a rand score greater than or equal to 0.95. Detection accuracy for the whistle trigger is somewhat weaker, with only around 62% of experiments exceeding the 0.95 threshold. This drop in performance is likely attributable to the fact that the dataset used to train the VAE was poisoned with the clapping trigger, thus rendering the model more sensitive to the presence of the clap than to the previously unseen whistle. Furthermore, the clap and whistle differ substantially in terms of both their spectral content and temporal envelope.

Figure 6 also reveals that the majority of failure cases in which our defense fails to identify the target class occur in experiments with the clapping trigger when the attacker selects target class 10. Class 10 represents the 'other' class, that contains a combination of different speech commands. Our method for identifying poisoned classes relies on the fact that the clean samples share a common underlying distribution, distinct from that of the poisoned samples. This assumption is clearly violated for class 10, which is already highly heterogeneous even in the absence of any poisoning attack. Interestingly, under the whistle trigger, our defense does correctly detect target class 10, although sample detection performance remains relatively poor. This may be due to the tonal nature of the whistle trig-

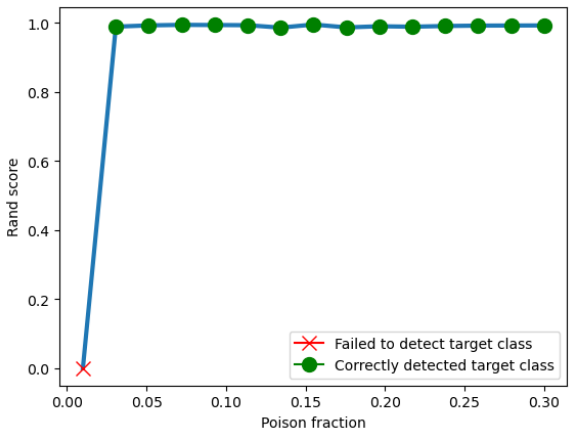

Figure 7: *Adjusted rand score (*0 *to* 1*, higher is better) for poison sample detection with differing amounts of poisoned data. The poison fraction (x axis) reflects the fraction of the source class used for the poisoning attack. All experiments used source class* 10*, target class* 2 *and the clapping trigger to match the dataset used to train the VAE model. Our defense is able to reliably detect the poisoned samples for all but the smallest attack.*

ger which closely resembles the harmonic structure of voiced speech. Since the VAE model is naturally attuned to such speech-relevant features, the presence of the whistle may be sufficient to differentiate the distribution of poisoned samples from that of the clean speech.

We also ran an ablation study to evaluate how well out method performed with varying amounts of poisoned data. Using a fixed source class, target class, and trigger (the same as in the VAE training data) we varied the fraction of the source class used for poisoning from 0.01 to 0.3. The resulting rand scores are shown in figure 7. Our defense identified the poisoned samples with high accuracy for all but the smallest poison fraction, where it failed to correctly detect the target class. These results suggest that our method is relatively insensitive to the exact amount of poisoning data injected by the attacker, further validating it generalizability.

5. Conclusion

In this paper, we have introduced a novel method for detecting poisoning attacks using latent vectors extracted from a variational autoencoder trained on the poisoned data. Our approach uses clustering within label groups to identify targeted classes and subsequently separate clean and poisoned samples within the poisoned class. We evaluate the performance of our defense on an audio dataset and show that it outperforms two popular baselines. Additionally, we demonstrate that our trained VAE model is capable of generalizing to detect different poisoning attacks provided the underlying dataset is the same as the one on which it was trained.

Our defense offers a number of benefits over other methods for poison sample detection. First, the per-class poison likelihood score implicitly assigns a 'risk level' that reflects the integrity of the underlying training data. An end user can therefore feasibly assess the likelihood of poisoned data within a large dataset and take appropriate action as fit their particular use case and risk tolerance. Furthermore, the robustness of our defense in detecting targeted classes enables this kind of trustworthiness test to be performed even in cases where the poisoned samples themselves cannot be adequately separated. Secondly, the generalizability our defense exhibits means that a single autoencoder can be used to detect a variety of poisoning attacks, without the need for re-training. This is especially useful in the case where a commonly used dataset (e.g. MNIST [13]) is available from multiple different online sources of unknown repute.

Future work should consider whether a sufficiently powerful VAE might be used to detect attacks across different datasets of the same modality (e.g. image, audio), perhaps with a few epochs of fine-tuning. This would further reduce the computational cost required for training a separate model for poison detection. Another important question is what other unsupervised methods and models might be leveraged to identify poisoned samples. Finally, we have yet to explore the generative capabilities of the VAE architecture. It may be possible to generate reconstructions of poisoned samples without the trigger, thus avoiding the data loss that results from simply discarding poisoned data.

6. Acknowledgements

This work was supported by DARPA (Grant No. HR00112020009), The USC Amazon Center for Secure and Trusted Machine Learning, and an Annenberg Fellowship (to N.M.).

7. References

[1] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," p. 1467–1474, 2012.

[2] 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.

[3] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 12 2017.

[4] 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," *Advances in neural information processing systems*, vol. 31, 2018.

[5] J. Steinhardt, P. W. Koh, and P. Liang, "Certified defenses for data poisoning attacks," pp. 3520–3532, 2017.

[6] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," 2019.

[7] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Neural Information Processing Systems*, 2018.

[8] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, 2019.

[9] E. Borgnia, J. Geiping, V. Cherepanova, L. Fowl, A. Gupta, A. Ghiasi, F. Huang, M. Goldblum, and T. Goldstein, "Dp-instahide: Provably defusing poisoning and backdoor attacks with differentially private data augmentations," 2021. [Online]. Available: https://arxiv.org/abs/2103.02079

[10] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018.

[11] Y. Li and S. Mandt, "Disentangled sequential autoencoder," in *International Conference on Machine Learning*, 2018.

[12] 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.

[13] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.