2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU) | 979-8-3503-0689-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ASRU57964.2023.10389697

JOINT ENERGY-BASED MODEL FOR ROBUST SPEECH CLASSIFICATION SYSTEM AGAINST DIRTY-LABEL BACKDOOR POISONING ATTACKS

Martin Sustek[1,2], *Sonal Joshi*[1], *Henry Li*[1], *Thomas Thebaud*[1]
Jesús Villalba[1], *Sanjeev Khudanpur*[1], *Najim Dehak*[1]

[1]Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, USA
[2]Faculty of Information Technology, Brno University of Technology, Czechia

ABSTRACT

Our novel technique utilizes a Joint Energy-based Model (JEM) that integrates both discriminative and generative approaches to increase resistance against dirty-label backdoor attacks. Our approach is especially effective when the trigger is short or hardly perceivable. We simulate the attack on the Speech Commands Dataset consisting of 1 s audio clips. During training, we use JEM to model a view of the input implemented by a randomly selected 610 ms window. During inference, we combine all (40) possible views utilizing a generative part of JEM. The resulting system has slightly decreased accuracy but significantly increased resistance shown in multiple scenarios. Interestingly, replacing JEM with a standard discriminative model (Disc) provides increased resistance with a lesser effect compared to JEM but maintains accuracy. We introduce an extension motivated by semi-supervised training that further improves JEM but not Disc. JEM can also benefit from Gaussian noise during evaluation.

Index Terms— joint energy-based model, poisoning attacks, speech commands classification, dirty label backdoor

1. INTRODUCTION

The need for collecting a huge amount of labeled data creates an opportunity for attackers, who seek to alter the behavior of machine learning systems. In data poisoning attacks, an attacker manipulates a subset of collected data. Specifically, in a dirty-label backdoor (DLBD) [1] attack scenario, attackers aim to create a backdoor to the system by adding a *trigger* to a subset of data points from a *source-class*, while simultaneously changing the corresponding ground-truth labels to *target-class* labels. This can alter the behavior of the trained system when it encounters test data containing the trigger [2], which is especially dangerous if the target-class prediction initiates a critical action of the system.

Some previous studies [3, 4, 5, 6] incorporated outlier detection to filter out poisoned examples. In our scenario, we assume that any data can potentially be poisoned. Therefore, we don't have access to *clean data* (a subset of data with a guarantee of not being poisoned) required by these techniques. Nevertheless, poisoned data exhibits two significant irregularities, the presence of the trigger and the mismatch between the provided class label y and the input data $\mathbf{x}$. Only techniques focusing on the latter can be applied against barely noticeable triggers. Using a trigger that is difficult for humans to perceive is more harmful, as it decreases the likelihood of detection, potentially enabling repeated attacks. Large neural network models can achieve nearly 100% accuracy on training data, making even imperceptible triggers capable of changing the behavior of an undefended system. Therefore, defenses that increase effectiveness with decreasing trigger perceptibility show greater promise for future development.

Our research is framed in the DARPA GARD (Guaranteeing AI Robustness Against Deception) program[1], which provides a range of benchmark tasks, attacks, and baseline defenses through the Armory toolkit[2]. Specifically, we focus on the poisoning attacks on Speech Commands dataset [7]. There is limited research on defenses against dirty label poisoning attacks on speech systems. For instance, [8] presents an attack against a speech recognition system, but no defense is proposed. Another recent work [9] offers a defense against clean label poisoning attacks on speaker verification systems, where the attacker replaces some of the attacked speaker's audio files, but the labels stay intact. In contrast, our work addresses dirty label poisoning attacks on speech command classification systems.

Defenses against DLBD attacks have been more extensively investigated within the domain of image classification. Typical approaches are based on a filtering pipeline. It involves training a neural network model on the poisoned dataset and extracting representations of the data. Subsequently, techniques such as [10, 11, 12, 13] are used to remove unusual (potentially poisoned) data points, and the standard classifier is trained on the remaining data. One non-filtering defense is a certified defense called randomized smoothing [14], which involves adding random noise to train and test examples in order to overwhelm the perturbation

[1]`https://www.darpa.mil/program/guaranteeing-ai-robustness-against-deception`
[2]`https://github.com/twosixlabs/armory`

that the attacker adds; however, it has limited effectiveness. Another class of defenses is based on majority voting. For example, Deep Partition Aggregation (DPA) [15] partitions the poisoned dataset into multiple disjoint subsets, then trains a classifier per subset, and the final decision is taken by majority voting of all the classifiers. Finite Aggregation [16] improves DPA by using smaller disjoint subsets and combining the duplicates to train multiple classifiers. Other defenses use differential privacy (DP) to ensure that the predictions by classifier do not depend too much on individual data points [17, 18, 19, 20].

Existing non-filtering approaches either focus on provable rather than empirical performance or are specifically designed for images and cannot be directly applied to speech. Moreover, the latter methods rely on specific data augmentation that might harm the task in certain settings. Filtering approaches are effective in removing poisoned data, but even a minor leak can be sufficient to establish a backdoor entry into the system[3]. Moreover, implementing filtering methods may not be feasible in specific scenarios. For instance, in continual learning [21, 22], data is gradually collected over time. However, the filtering requires training a system on all available data first. This motivates us to propose a new method for training a robust classifier against DLBD attacks and demonstrating its effectiveness on a speech classification task. Our standalone approach is versatile as it does not depend on any specific architecture or training paradigm. As a result, it is also well-suited to serve as the final step in a filtering pipeline or as a potential replacement for a classifier employed in other non-filtering methods. Our technique employs JEM which modifies the training of a standard classifier to additionally provide the score proportional to likelihood. We introduce a novel way of determining the class membership during evaluation which is distinct from the one optimized during training. It is based on a proper probabilistic inference when JEM is used, but we empirically show the usefulness even for a standard classifier. We additionally propose an extension of our approach motivated by semi-supervised training and show its effectiveness in more challenging scenarios. We also demonstrate that during inference, augmenting the input with an appropriate amount of Gaussian noise or modifying it in a direction that maximizes the likelihood provides an enhancement to robustness.

2. GENERATIVE MODELING

2.1. Motivation

Discriminative models excel in classifying training data based on provided labels by extracting relevant information from the input. However, they are prone to relying on easily recognizable features, which we illustrate by the analogy of classifying paintings by authors. When the author's signature is present in the training data, the classifier may rely solely on it, resulting in reduced performance when the signature is later absent. This issue is typically addressed by regularization techniques that weaken the link to the signature by forcing the classifier to consider only part of each painting or by augmenting each painting during training. In this analogy, the clean data setup corresponds to datasets of paintings with no signature. When data is poisoned, some paintings are enriched with the signature (trigger) of different authors and are falsely labeled as belonging to them. Poisoning an example can significantly reduce the classification difficulty, as it only lies in extracting the signature.

We argue that if the classifier additionally needs to reconstruct the entire painting, the presence of the corresponding author's signature no longer simplifies the classification task. Poisoning an example actually increases the difficulty of classifying and at the same time reconstructing this painting, as the classifier is forced to take into consideration the entire painting. The part of the poisoned painting that does not contain the signature of a different author provides evidence supporting the correct author ownership. For the classifier, this evidence is interfering with its objective of classifying the poisoned painting as belonging to a falsely provided author. We believe that this effect should be magnified for hardly perceivable triggers[4].

Generative models describe the data distribution, which implicitly requires paying attention to the whole input and we perceive them as a proxy for the above-mentioned painting reconstruction. Specifically, we investigate Joint Energy-based Model (JEM) [23] that modifies the training of a standard classifier by directly embedding unscaled likelihood values into the logits. The resulting model can function as a classifier, with the embedded generative part ensuring that the entire input is considered during the classification process. We anticipate that this approach enhances natural defense against DLBD attacks or unreliable labels, but the effectiveness of this approach is subject to this work.

2.2. Joint Energy-Based Model

The standard classifier typically utilizes a neural network (NN) $f_{\boldsymbol{\theta}}(\mathbf{x})\colon\ \mathbf{x} \mapsto \mathbf{q}$ with parameters $\boldsymbol{\theta}$ to map input example $\mathbf{x} \in \mathbb{R}^{D_x}$ to a vector of logits $\mathbf{q} \in \mathbb{R}^{D_y}$, where D_x is the dimensionality of the input example and D_y corresponds to the number of classes. Denoting q_i as i-th index of $\mathbf{q}$, posterior distribution over labels $y \in \mathbb{N}^+, y \leq D_y$ is calculated as

$$p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) = \text{Softmax}(\mathbf{q})_y = \frac{e^{q_y}}{\sum_{i=1}^{D_y} e^{q_i}}. \tag{1}$$

JEM uses the same NN $f_{\boldsymbol{\theta}}(\mathbf{x})$ to model the joint distribution of $\mathbf{x}$ and y via the energy $E_{\boldsymbol{\theta}}(\mathbf{x}, y)$ as

$$p_{\boldsymbol{\theta}}(\mathbf{x}, y) = \frac{e^{-E_{\boldsymbol{\theta}}(\mathbf{x},y)}}{Z_{\boldsymbol{\theta}}} = \frac{e^{f_{\boldsymbol{\theta}}(\mathbf{x})_y}}{Z_{\boldsymbol{\theta}}} = \frac{e^{q_y}}{Z_{\boldsymbol{\theta}}}. \tag{2}$$

[3]https://github.com/twosixlabs/armory/blob/master/docs/baseline_results/speech_commands_poison_results.md

[4]Note that the case of asymptotically decreasing the strength of the trigger up to zero corresponds to the common task of dealing with unreliable labels, suggesting an additional utilization of the proposed technique.

The intractable partition function $Z_{\boldsymbol{\theta}} = \sum_y \int_{\mathbf{x}} e^{-E_{\boldsymbol{\theta}}(\mathbf{x},y)} d\,\mathbf{x}$ ensures that the probability distribution is properly normalized. Expressing the posterior distribution $p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$ in JEM factorizes $Z_{\boldsymbol{\theta}}$ out, resulting in the expression equivalent to the standard classifier (Equation 1). This demonstrates that the standard classifier and JEM differ only in the training procedure as both are represented by the same $f_{\boldsymbol{\theta}}(\mathbf{x})$. The outcome of the JEM training procedure is that e^{q_y} becomes proportional to $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$. However, in the case of the standard classifier, there is no guarantee of any inherent relationship between e^{q_y} and $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$.

2.3. JEM Training

The objective of JEM maximum likelihood training can be factorized as $\log p_{\boldsymbol{\theta}}(\mathbf{x}, y) = \log p_{\boldsymbol{\theta}}(\mathbf{x}) + \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$. The optimization of the last term corresponds to the training of the standard classifier and $\log p_{\boldsymbol{\theta}}(\mathbf{x})$ can be maximized by approximating the intractable expectation in

$$\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) = \nabla_{\boldsymbol{\theta}} g_{\boldsymbol{\theta}}(\mathbf{x}) - \mathbb{E}_{p_{\boldsymbol{\theta}}(\tilde{\mathbf{x}})}\left[\nabla_{\boldsymbol{\theta}} g_{\boldsymbol{\theta}}(\tilde{\mathbf{x}})\right] \quad (3)$$

via Stochastic Gradient Langevin Dynamics (SGLD) [24] samples, where $g_{\boldsymbol{\theta}}(\mathbf{x}) = \log \sum_y \exp f_{\boldsymbol{\theta}}(\mathbf{x})_y$. Improper samples of SGLD can lead to training instabilities, but the use of training modifications from [25] resolves this issue. Instead of minimizing $-\log p_{\boldsymbol{\theta}}(\mathbf{x}, y)$, they also suggest minimizing modified loss

$$\mathrm{L} = -\log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{x} \mid y), \quad (4)$$

which results in updates from Equation 3 to be replaced by

$$\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{x} \mid y) = \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\mathbf{x})_y - \mathbb{E}_{p_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}|y)}\left[\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}})_y\right]. \quad (5)$$

As suggested in [25], we occasionally replace class conditional samples from $p_{\boldsymbol{\theta}}(\mathbf{x} \mid y)$ with unconditional samples from $p_{\boldsymbol{\theta}}(\mathbf{x})$, but only with a probability of 0.1. Per each minibatch of 64 examples, we generate 6 samples[5] via iterative SGLD procedure for $1 \leq i \leq D_x$ as

$$\mathbf{x}^t = \mathbf{x}^{t-1} + \frac{\alpha^t}{2} \nabla_{\mathbf{x}} F_{\boldsymbol{\theta}}(\mathbf{x}^{t-1}) + \mathbf{u}^t, \qquad u_i^t \sim \mathcal{N}(0, \alpha^t). \quad (6)$$

To sample from $p_{\boldsymbol{\theta}}(\mathbf{x})$, we set $F_{\boldsymbol{\theta}}(\mathbf{x}) = g_{\boldsymbol{\theta}}(\mathbf{x})$, whereas setting $F_{\boldsymbol{\theta}}(\mathbf{x}) = f_{\boldsymbol{\theta}}(\mathbf{x})_y$ produces samples from $p_{\boldsymbol{\theta}}(\mathbf{x} \mid y)$.

3. EXPERIMENTAL SETUP

3.1. Poisoned Speech Commands Dataset

We simulate dirty-label poisoning attacks on Speech Commands Dataset [7]. This dataset consists of 1 s long audio clips $\mathbf{x}$ belonging to 12 different keyword classes $y \in \{0, 1, .., 11\}$. Classes 0 to 9 are single-word commands[6], while class 10 represents silence and background noise, and class 11 represents all other words. The last category consists of 25 different short words. Classes 0 to 9 are roughly equally distributed in the training set, but class 11 comprises 63.24 % of 85511 the training examples. In contrast, the test set is nearly balanced, with each class accounting for 8.1 % to 8.7 % of 4890 test set utterances. It's worth noting that class 11 is intended to be the source class for poisoning.

3.2. Threat model

The default trigger is a short audio of a clap sound placed at the beginning of a certain portion of utterances $\mathbf{x}$ belonging to the source class 11, whose label y is changed to target class 2. The trigger has comparable energy to $\mathbf{x}$ but it is scaled by 0.1 before mixing it with the original audio. During the evaluation, the same trigger employed in the training phase is added to the source class test utterances, and the Accuracy measured with respect to the correct labels along with Attack Success Rate (ASR) is reported. ASR measures the proportion of source class examples misclassified as the target class.

3.3. Model Descriptions

The baseline system (**Base**) is a ResNet50 [26] with 23.9 M parameters trained to predict the class label given input spectrogram $\mathbf{x}$. It is a Pytorch reimplementation of the DARPA GARD TensorFlow baseline system provided in the armory toolkit[7].

We use the default JEM system from [23, 25], which employs 28-layer deep, 10-k-wide WideResNet[8] architecture [27] with 36.6 M parameters with removed batch normalization. We keep the same features $\mathbf{x}$ used by [25] for speech applications, which are log-mel filter banks normalized over the dataset.

4. PROPOSED METHOD

In Section 2.1, we argued that poisoning an example might decrease the task difficulty when using the standard classifier, but increase it for JEM. Our goal is not just increasing the difficulty, but ideally building a classifier capable of outputing the correct class during evaluation when the trigger is presented. To achieve that, we propose a technique to further weaken the link between the trigger and the target class.

4.1. JEM Applied over Multiple Views (MVJ)

JEM provides access to unscaled likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}, y)$. Similarly to $p_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_y p_{\boldsymbol{\theta}}(\mathbf{x}, y)$, we can introduce another variable and marginalize it out by applying the sum rule, which is a straightforward way for combination when using generative models. Instead of directly modeling 1 s input utterance, we propose to use multiple views τ of each input. Input views could be in general obtained by any transformation[9] of the input signal. Our approach aims at the similarity between the original input and the input corrupted by the trigger and we expect that it can be better exploited in lower dimensional

[5]We maintain the default settings for the rest of the SGLD parameters.

[6]From 0 to 9: down, go, left, no, off, on, right, stop, up, yes.

[7]`https://github.com/twosixlabs/armory/blob/master/docs/scenarios.md`

[8]Attempt to use WideResNet and its setup instead of ResNet50 as a baseline architecture decreased the accuracy of about 2 % with negligible improvement in ASR, e.g. 100 % → 98 %.

[9]Active dropout, data augmentation, jitter, adding noise, stochastic denoiser, or considering only particular bandwidth to name a few.

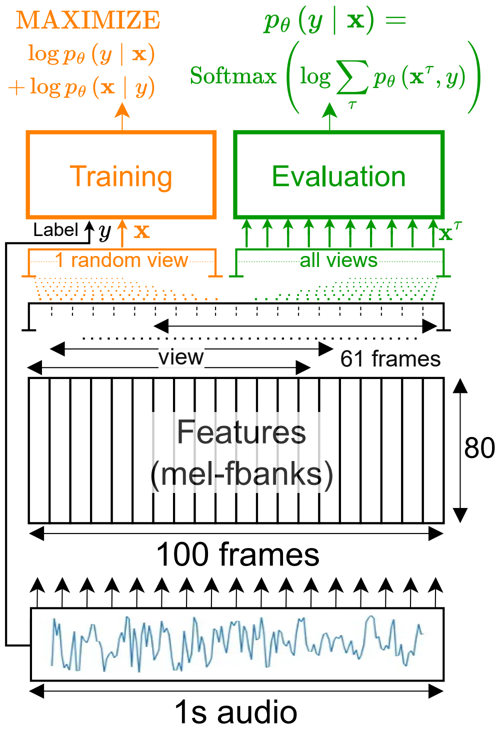
MAXIMIZE
$\log p_\theta(y \mid \mathbf{x})$
$+\log p_\theta(\mathbf{x} \mid y)$
$p_\theta(y \mid \mathbf{x}) =$
Softmax $\left(\log \sum_\tau p_\theta(\mathbf{x}^\tau, y)\right)$
Training
Evaluation
Label y $\mathbf{x}$
$\mathbf{x}^\tau$
1 random view
all views
view
61 frames
Features (mel-fbanks)
80
100 frames
1s audio

Fig. 1. Training and subsequent evaluation schema of the proposed system. All utterances are 1 s long with corresponding 100 frames. Out of 61 consecutive frames, we compose one view. During the training, mini-batches contain only one random view per recording. Evaluation combines all 40 views.

spaces. For that reason, we implement view as a segment of an input with a length[10] of around 610 ms (61 consecutive frames) instead of the full 1 s. This creates 40 distinct views per input since the time shift between features is 10 ms. We expect that having multiple views increases the difficulty of trigger extraction and introduces the possibility of obtaining some views entirely free of the trigger.

During training, per mini-batch, we randomly[11] choose only a single view of each utterance and minimize L (Equation 4). During inference, we treat the index of the view as a random variable τ and interpret $p(\mathbf{x}, y)$ of the view τ as $p_{\boldsymbol{\theta}}(\mathbf{x}, y \mid \tau)$. We set $p_{\boldsymbol{\theta}}(\tau)$ to be a non-informative (uniform) prior, which allows us to obtain the posterior by marginalizing over τ as

$$p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) = \text{Softmax}\left(\log \sum_\tau \exp f_{\boldsymbol{\theta}}(\mathbf{x}^\tau)\right)_y , \quad (7)$$

where $\mathbf{x}^\tau$ denotes the view of $\mathbf{x}$ with the index τ. The overall training and evaluation schema of Multiple View JEM (**MVJ**) is depicted in Figure 1. We train our model for 26 epochs and since we have 40 different views, it is unlikely for the same view to be chosen many times. We anticipate that in the absence of the poisoned label, JEM would assign a high likelihood to the original class for all views of the poisoned example. Due to the fact that we select only a single view per mini-batch during training, we believe that JEM might still assign a high likelihood to the original class for other views of the poisoned example. For the success of this approach, it is crucial that Equation 7 is not directly optimized during training. The evaluation of multiple views at once generally scales linearly with the number of views. However, the type of views we employ incurs only minimal overhead ($\approx 2\times$ slowdown) due to a large overlap of views and the utilization of a specialized architecture that shares most computation across different views.

[10]The length is a hyperparameter and could be set arbitrarily.
[11]We sample from a uniform distribution, but any distribution is allowed.

4.2. Discriminative Model and Multiple Views (MVD)

We demonstrated that the standard discriminative model differs from JEM only in the type of training. Instead of $\log p_{\boldsymbol{\theta}}(\mathbf{x}, y) = \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x})$, or in our case, instead of minimizing Equation 4, the classifier is trained to maximize $\log p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$. Unlike for JEM, combining views by applying Equation 7 has no mathematical foundation for the standard classifier. However, the computation can be performed and seen as an ad hoc logit combination. We use the same protocol for training and evaluation as for MVJ, but instead of training JEM, we train the standard discriminative model. The resulting approach (**MVD**) can also be seen as training a standard discriminator but treating it as JEM in the context of our multiple views method. MVD takes 12 h to train, ¼ of MVJ training time.

4.3. Extention Inspired by Semi-Supervised Training

Denoting p_d as empirical data distribution (with labels), previously trained JEM minimized $\mathbb{E}_{p_d(\mathbf{x},y)}[\mathrm{L}]$, with L defined in Equation 4. Unlabeled data can be incorporated into JEM training [25] by minimizing $\mathbb{E}_{\mathbf{x}\sim p_d(\mathbf{x})}\left[\sum_y p_{\boldsymbol{\theta}}(y \mid \mathbf{x})\,\mathrm{L}\right]$ for such data. This encourages the model to assign a high value of posterior distribution to a class of its own choice while preventing posterior collapse. Our extension uses weight γ to combine both losses and minimize them at the same time for the data from the poisoned dataset. We expect that $p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$ will match the given label for clean data. Meanwhile, for poisoned data, $p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$ should also assign a large value to the true label, which then causes a large supervised loss value, motivating JEM to intervene. The proposed extension aims at increasing tolerance toward this kind of "error" by a relative decrease in the loss value associated with this case.

The optimal solution of unlabeled loss is reached when JEM assigs the same number of examples to each class, i.e. when $p_{\boldsymbol{\theta}}(y)$ is uniform. To deal with class imbalance, we add an additional term to the loss, such that the optimum is reached when $p_{\boldsymbol{\theta}}(y)$ distributes as $p_d(y)$. This term is the cross-entropy between $p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$ and $p_d(y)$. This is equivalent to setting $\tilde{\mathrm{L}} = \mathrm{L} - \log p_d(y)$ and minimizing

$$(1-\gamma)\,\mathbb{E}_{p_d(\mathbf{x},y)}[\mathrm{L}] + \gamma\,\mathbb{E}_{p_d(\mathbf{x})}\left[\sum_y p_{\boldsymbol{\theta}}(y \mid \mathbf{x})\,\tilde{\mathrm{L}}\right]. \quad (8)$$

Table 1. Accuracy and Attack Success Rate (ASR) at different trigger scales on the default scenario – 10 % poisoning of source-class 11, target class 2, clap at the beginning.

Clean Training Data and Trigger at Various Scales						
Variant / Scale	**Base**	**MVD**	**MVJ**	**Base**	**MVD**	**MVJ**
	$\downarrow$ **ASR [%]**			$\uparrow$ **Accuracy [%]**		
`clean`	-	-	-	94.6	**95.1**	91.9
`0.02`	100.0	11.3	**0.3**	86.5	**94.1**	89.5
`0.1`	99.8	27.5	**1.0**	86.9	**92.2**	90.8
`0.5`	100.0	87.0	**69.9**	86.9	**87.5**	85.5

We set $\gamma = 0.1$ at the beginning of the training and reach $\gamma = 0.475$ in the final epoch by the constant increase of 0.015 per epoch. We also apply the same extension to MVD, for which the losses in Equation 8 becomes $\tilde{\mathrm{L}} = \mathrm{L} = -\log p_{\boldsymbol{\theta}}(y \mid \mathbf{x})$. We refer to this extension as **SemiSup** to distinguish it from the previously introduced version **Basic**.

4.4. Extention – Augmentation during Evaluation

In our framework, data augmentation [28] could be incorporated during training to obtain multiple views. Instead, we propose to use data augmentation only during evaluation, specifically, we consider Gaussian noise and the SGLD step. Each SGLD step is composed of two sub-steps, changing the input $\mathbf{x}$ in the direction that increases the likelihood the most and adding Gaussian noise. We expect that corrupting test data by Gaussian noise weakens the trigger signal more than the desired signal as long as the trigger scale remains small. Moreover, we hope that a subsequent SGLD step[12] (Equation 6) for $F_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{\tau} f_{\boldsymbol{\theta}}\left(\mathbf{x}^{\tau}\right)$ denoted as `SGLD` will have a further positive effect on ASR and possibly accuracy. In our setup, we first apply Gaussian noise and/or SGLD step and then construct views, i.e. there is one Gaussian noise and SGLD step per recording rather than per view. The SGLD step is obtained by summing contributions from all views.

5. EXPERIMENTS

5.1. Trigger Affecting Only a Subset of Views

We evaluate the default scenario described in Section 3.2 poisoning 10 % of the source class, which causes 64 % of all examples labeled as the target class to be poisoned. Evaluation with a short trigger at the beginning of the utterance is suitable to simulate a scenario where the trigger affects only a subset of all input views, as almost ⅔ of views of each input do not contain any trigger. In Table 1, we present the accuracy for the system trained on clean data and the impact of varying the default trigger scale of 0.1.

[12]Last SGLD step typically doesn't add noise, i.e. it is equivalent to SGD.

Table 2. Accuracy and Attack Success Rate (ASR) for target class 5 with 10 % poisoning of various source classes for 0.1 scaled clap trigger placed at the utterance beginning.

Target Class 5, Various Source Classes						
Variant / Source	**Base**	**MVD**	**MVJ**	**Base**	**MVD**	**MVJ**
	$\downarrow$ **ASR [%]**			$\uparrow$ **Accuracy [%]**		
class **`3`**	100.0	10.4	**0.3**	86.8	**94.0**	88.3
class **`11`**	99.5	37.0	**6.4**	87.6	**91.9**	90.4

Since the source class 11 represents 8.3 % of the test set, it is noteworthy that the Base system consistently maintains an accuracy of approximately 95 % on the provided (poisoned) labels. Interestingly, applying the combination proposed for JEM (Equation 7) to a discriminative model (MVD) provides high accuracy with increased natural resistance toward DLBD attacks compared to Base, especially for low trigger scales. The ability of MVJ to model $p_{\boldsymbol{\theta}}(\mathbf{x})$ hurts test accuracy across all cases, but we also observe a significant decrease in ASR compared to MVD when sufficiently small trigger scales are evaluated.

We change the target class to 5, and we additionally use a smaller source class (3). Poisoning 10 % of source class 3 corresponds to only 9 % of the total number of examples from target class 5 to be poisoned. The results in Table 2 are consistent with the previous observations. Moreover, as expected, reducing the number of poisoned examples leads to improved performance. Since both MVD and MVJ systems perform well when the trigger scale is low, they are capable of ignoring incorrect labels, as this corresponds to the trigger scale of 0.

5.2. More Challenging Triggers

We considered three additional triggers for our experiments: the first trigger, the sound of a whistle (`Whistle`), is placed close to the end of the utterance, but only about ⅕ of the views do not contain the trigger. The second trigger (`6 Claps`) consists of six consecutive claps and the third trigger concatenated a clap sound with a car horn sound (`Clap+Horn`). The last two triggers affect the entire signal. Given the increased difficulty of this setting and the fact that our approach performs better with fewer poisoned cases and higher attenuation of the trigger, we evaluated these triggers with a trigger scale of 0.02 and a poisoning rate of 1 % on source class 11, which results in 15 % of all examples from target class 2 being poisoned. The results are presented in Table 3. When the trigger corrupts the entire signal, both MVJ and MVD still offer benefits over Base. However, their effectiveness is reduced compared to the previous cases.

5.3. Extensions

We test the extension SemiSup proposed in Section 4.3 on the variant `Clap+Horn` reported in Table 3 and on the modified

Table 3. Accuracy and Attack Success Rate (ASR) for target class 2 poisoning source class 11 for various triggers.

Challenging Triggers, 1 % Poisoned, 0.02 Scale						
Trigger \ Variant	Base	MVD	MVJ	Base	MVD	MVJ
	↓ ASR [%]			↑ Accuracy [%]		
`Whistle`	88.2	78.7	**4.4**	86.5	88.3	**88.5**
`6 Claps`	94.9	82.6	**49.3**	87.8	**87.9**	85.4
`Clap+Horn`	96.6	81.1	**46.6**	85.7	**88.0**	86.5

default setup (scale 0.1 reported in Table 1). In the modified setup, we place the trigger at random positions in the signal rather than at the beginning (`Random-Clap`), which significantly reduces the performance of our approaches when the extension is not applied (Basic).

For each variant in every setting, we choose two appropriate Gaussian noise levels added to features, lower `NoisLo` and higher `NoisHi`, and evaluate the trade-off between accuracy and ASR. We use the architecture and features of MVD/MVJ for Base when we add Gaussian noise. Additionally, after the noise is inserted, we consider modifying the utterance $\mathbf{x}$ by performing one SGLD step denoted as `SGLD`. More details in Section 4.4.

Results in Table 4 indicate that SemiSup has a positive effect on both accuracy and ASR of MVJ, but it does not improve MVD. In addition, a suitable amount of noise can significantly reduce ASR for MVJ with a reasonably small decrease in accuracy, especially when combined with `SGLD`. The same holds for Base and MVD but with a significantly worse trade-off between the accuracy and ASR. The main reason for reporting these results is to show the potential of MVJ as we use an oracle to determine suitable noise levels and SGLD step sizes α^t. Choosing a proper strategy for selecting these is out of the scope of this work.

6. CONCLUSION

We proposed a novel approach to train a system for speech classification with increased robustness against dirty-label backdoor poisoning attacks. Our method involves dividing 1 s long input recordings into 40 overlapping 610 ms long views. During training, we construct mini-batches by randomly selecting one view per recording. During testing, we combine all views. We use a Joint Energy-based Model (JEM) that fuses a discriminative and generative model, and the view combination is motivated by proper inference that leverages the generative part of the MVJ system. Interestingly, applying the same inference to the standard discriminative system MVD yields benefits in both accuracy and resistance against the attack in all settings compared to the baseline system. While MVJ provides the best resistance against poisoning attacks in all settings, its accuracy is slightly reduced. Moreover, we extended MVJ training inspired by semi-supervised training. This variant further increases the accuracy and resistance but does not improve the MVD system. Finally, we demonstrated that adding a proper (obtained by an oracle) amount of Gaussian noise and subsequent SGLD step of a proper size during inference can significantly boost MVJ's resistance but it provides only a limited boost to the baseline system and MVD. Overall, we provided ample evidence that our incorporation of MVJ is effective against backdoor attacks and we propose future research directions.

Table 4. Accuracy and Attack Success Rate (ASR) of leveraging a semi-supervised loss **`SemiSup`**, including lower `NoisLo` and higher `NoisHi` noise levels during testing, and performing 1 SGLD step `SGLD` for target class 2 and source class 11, tested with a clap at a random location `Random-Clap`, and clap with horn `Clap+Horn` triggers.

Extensions \ Variant	Base	MVD	MVJ	Base	MVD	MVJ
	↓ ASR [%]			↑ Accuracy [%]		
Extensions, `Random-Clap`, 10 % Poisoned, 0.1 Scale						
Basic	97.3	78.4	**58.1**	86.6	**88.1**	85.1
`+NoisLo`	91.9	45.3	**31.4**	82.3	**85.8**	84.5
`+NoisLo, SGLD`	-	51.5	**28.2**	-	**87.4**	86.6
`+NoisHi`	46.3	27.5	**4.7**	70.2	75.3	**82.3**
`+NoisHi, SGLD`	-	29.7	**3.2**	-	79.7	**85.3**
SemiSup	-	83.6	**35.5**	-	87.8	**88.1**
`+NoisLo, SGLD`	-	69.6	**9.1**	-	87.4	**88.2**
`+NoisHi, SGLD`	-	33.8	**0.3**	-	82.2	**83.7**
Extensions, `Clap+Horn`, 1 % Poisoned, 0.02 Scale						
Basic	96.6	81.1	**46.6**	85.7	**88.0**	86.5
`+NoisLo`	67.9	29.9	**14.2**	82.0	**89.1**	88.5
`+NoisLo, SGLD`	-	33.1	**6.9**	-	**89.8**	89.0
`+NoisHi`	43.4	16.2	**1.0**	72.7	82.9	**85.8**
`+NoisHi, SGLD`	-	15.7	**0.5**	-	85.4	**87.0**
SemiSup	-	82.1	**38.5**	-	87.6	**88.3**
`+NoisLo, SGLD`	-	42.2	**13.7**	-	85.8	**90.1**
`+NoisHi, SGLD`	-	40.0	**1.5**	-	82.7	**88.3**

7. ACKNOWLEDGEMENTS

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0114. Any opinions, findings, conclusions, or recommendations expressed here are those of the author(s) and do not necessarily reflect the views of DARPA.

8. REFERENCES

[1] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.

[2] P.Y. Chen and C.J. Hsieh, *Adversarial Robustness for Machine Learning*, Elsevier Science, 2022.

[3] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, and Jaehoon Amir Safavi, "Mitigating poisoning attacks on machine learning models: A data provenance based approach," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 103–110.

[4] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang, "Certified defenses for data poisoning attacks," *Advances in neural information processing systems*, vol. 30, 2017.

[5] Marius Kloft and Pavel Laskov, "Online anomaly detection under adversarial impact," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 405–412.

[6] Marius Kloft and Laskov, "Security analysis of online centroid anomaly detection," in *The Journal of Machine Learning Research*. JMLR, 2012, pp. 3681–3724.

[7] Pete Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[8] Hojjat Aghakhani, Lea Schönherr, Thorsten Eisenhofer, Dorothea Kolossa, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna, "Venomave: Targeted poisoning against speech recognition," in *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2023.

[9] Ke Li, Cameron Baird, and Dan Lin, "Defend data poisoning attacks on voice authentication," *arXiv preprint arXiv:2209.04547*, 2022.

[10] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," in *Workshop on Artificial Intelligence Safety*. CEUR-WS, 2019.

[11] Brandon Tran, Jerry Li, and Aleksander Madry, "Spectral signatures in backdoor attacks," *Advances in neural information processing systems*, vol. 31, 2018.

[12] Charles Jin, Melinda Sun, and Martin Rinard, "Leveraging incompatibility to defend against backdoor poisoning," in *International Conference on Learning Representations*.

[13] Neehar Peri, Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John P Dickerson, "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*. Springer, 2020, pp. 55–70.

[14] Binghui Wang, Xiaoyu Cao, Neil Zhenqiang Gong, et al., "On certifying robustness against backdoor attacks via randomized smoothing," in *CVPR Workshop on Adversarial Machine Learning in Computer Vision*, 2020.

[15] A Levine and S Feizi, "Deep partition aggregation: Provable defense against general poisoning attacks," in *International Conference on Learning Representations (ICLR)*, 2021.

[16] Wenxiao Wang, Alexander J Levine, and Soheil Feizi, "Improved certified defenses against data poisoning with (deterministic) finite aggregation," in *International Conference on Machine Learning*. PMLR, 2022, pp. 22769–22783.

[17] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein, "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[18] Sanghyun Hong, Varun Chandrasekaran, Yiğitcan Kaya, Tudor Dumitraş, and Nicolas Papernot, "On the effectiveness of mitigating data poisoning attacks with gradient shaping," *arXiv preprint arXiv:2002.11497*, 2020.

[19] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta, "Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3855–3859.

[20] Eitan Borgnia, Jonas Geiping, Valeriia Cherepanova, Liam Fowl, Arjun Gupta, Amin Ghiasi, Furong Huang, Micah Goldblum, and Tom Goldstein, "Dp-instahide: Provably defusing poisoning and backdoor attacks with differentially private data augmentations," *International Conference on Learning Representations (ICLR), Workshop on Security and Safety in Machine Learning Systems*, 2021.

[21] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, 2019.

[22] Zhizhong Li and Derek Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[23] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky, "Your classifier is secretly an energy based model and you should treat it like one," in *International Conference on Learning Representations ICLR*, 2020.

[24] Max Welling and Yee W Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 681–688.

[25] Martin Sustek, Samik Sadhu, Lukas Burget, Hynek Hermansky, Laureano Moro-Velazquez, Jesus Villalba, and Najim Dehak, "Stabilized training of joint energy-based models and their practical applications," *arXiv preprint arXiv:2303.04187*, 2023.

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

[27] Sergey Zagoruyko and Nikos Komodakis, "Wide residual networks," *Proceedings of the British Machine Vision Conference BMVC*, 2016.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.