Imperceptible Backdoor Attacks for Deep Regression Models

Adapting the SIG Backdoor Attack to the Head Pose Estimation Task

Konstantin Mirinski[1]

Supervisor(s): Guohao Lan[1]**, Lingyu Du**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 24, 2024

Name of the student: Konstantin Mirinski
Final project course: CSE3000 Research Project
Thesis committee: Guohao Lan, Lingyu Du, Sicco Verwer

Abstract

With the rise of deep learning and the widespread use of deep neural networks, backdoor attacks have become a significant security threat, drawing considerable research interest. One such attack is the SIG backdoor attack, which introduces signals to the images. We look into three types of SIG backdoor attacks - ramp, triangle, and sinusoidal signals. Most of the works in the field of AI security, however, have focused on deep classification tasks, leaving deep regression tasks unexplored. In this study, we adapt the SIG backdoor attack for use in a deep regression model (DRM) used to estimate head pose. Our objective is to create a backdoor attack that remains imperceptible to the human eye while being detectable by the DRM. To evaluate the effectiveness of our attack, we employ two approaches: average angular error and accuracy in a discretized continuous space. Additionally, we adapt fine-tuning as a countermeasure against the backdoor attack. By implementing this strategy, we aim to reduce the risk of backdoor attacks and improve the robustness of deep regression models in head pose estimation.

1 Introduction

Deep learning methods are commonly used for a wide range of regression tasks and are known for their effectiveness. The regression task involved in the research is head pose estimation. In the field of computer vision, head pose means the orientation of the person's head represented by three angles - yaw, roll, and pitch. A depicted description can be found in Figure 1 [1]. However, the security of such techniques is exposed to attacks, thus affecting their applicability. One such attack is the backdoor attack.

Backdoor attacks are a newly introduced type of attacks that target deep learning systems. In these attacks, the goal of the attacker is to implant a backdoor within the system, allowing for the manipulation of system outcomes or other desired behaviors during testing [2], [3]. This is achieved by poisoning the dataset. The attacked model yields a predefined outcome for the backdoored instances by assigning them a target label specified by the attacker. The attacker corrupts a percentage of samples in the training set by injecting a backdoor signal and assigning them the target label.

In this research paper we consider the SIG backdoor attack [4], which falls into poisoning the dataset category. Examples of how the attack looks can be seen in Figure 2. In this attack, the attacker is not required to poison the labels of the corrupted samples. In terms of classification, given a classifier, the goal of the attacker is to induce the classifier to decide for a target class $\mathbf{y_t}$ ever when the test sample belongs to a different class. To do so, the attacker corrupts a percentage of the samples from class $\mathbf{y_t}$ with a backdoor signal $\mathbf{v_t}$. The aim is to make the classifier believe that the presence of the signal $\mathbf{v_t}$ is associated with class $\mathbf{y_t}$. At test time, the attacker adds to a sample belonging to a different class $\mathbf{y_x}$ ($\mathbf{x} \neq \mathbf{t}$) the backdoor signal $\mathbf{v_t}$. Even though the signal is nearly imperceptible, the classifier detects its presence and decides that the sample is from class $\mathbf{t}$. Considering that, it is important to mention that an attack on multiple targets is theoretically possible if we use different signals for each target class. An example of such an attack is presented and analyzed in [4].

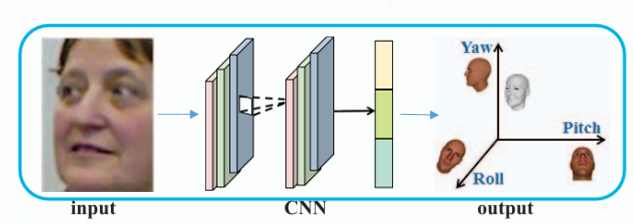

Figure 1: **Simplified Framework of the method used for head pose estimation.** For given a head image, a CNN-based model is used to estimate the angles (yaw, roll, and pitch) of the head position.

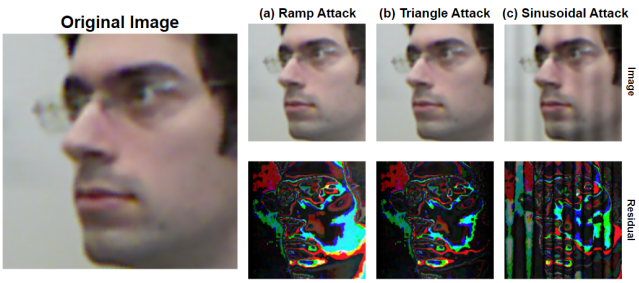

Figure 2: **Comparison between the three types of SIG backdoor attacks, and their respective residuals to the original image.** Given the original image, we generate three signals (which are described formally in Section 4) - namely: (a) ramp signal, (b) triangle signal, and (c) sinusoidal signal. We can see that the signals are nearly imperceptible to the human eye. The residuals are produced using bit-wise XOR.

Most of the research conducted in this area, however, is engaged only with deep classification tasks. There is very little information about how backdoor attacks affect the behavior of deep regression problems. In this research, we are interested in seeing how the SIG backdoor attack influences the performance of a deep regression model (head pose estimation in our case). Although we previously noted that the SIG backdoor attack does not necessarily require label poisoning, we have decided to include it (only when measuring the performance using the first metric). This decision is based on the theoretical consideration that in the continuous space, there can be infinitely many labels, thus warranting a comprehensive approach. There are two methods for measuring the effectiveness adapted to the backdoor attack that we looked into:

1. **Average angular error metric.** We treat the model as a pure regression task, e.g., the target label is a 3-dimensional vector (representing pitch, yaw, and roll) taken from the continuous space of the problem. Then, we use an average angular error metric to evaluate the model.

2. **Discretization metric.** The idea of this approach is to split the continuous space into I intervals of equal length - each interval is represented by a label (labels go to $0, 1, ..., (I-1)$), where I can be adjusted. We try to find the biggest I, for which the model performs well enough, e.g., yields good performance.

This leads us to the questions we answer to. In this research, we have focused on the behavior of the SIG attack on the head pose estimation task (regression task). To be more specific, we have focused on the following questions:

1. How can the SIG Backdoor Attack method be adapted and applied to compromise a Deep Regression Model (DRM) used to estimate head position?
2. How can we evaluate its effectiveness [1]?
3. Which parameters make the attack successful and imperceptible at the same time?

The rest of the paper is organized as follows. In Section 2 we give a brief overview of the conducted research in the area. In Section 3 we showcase the methodology that was utilized for performing the experiments. In Section 4 we present the implementation details and the results from the experiments. Next, in Section 5 we dive into the responsible and ethical part of the research. Finally, in Section 6 we draw some conclusions and highlight potential future work.

2 Related Works

Backdoor attacks. To our knowledge, the poisoning attacks date back to when data poisoning was used to flip the results of Support Vector Machines (SVM) [5]. More recently, backdoor attacks, where a trigger is used in poisoning the data, are shown in works such as [4], [6], and [7]. Such methods are more practical as the model works well on clean data and the attacks are triggered by introducing a predefined pattern (trigger). The main inspiration drawn for this paper is from [4], as we are adapting this particular backdoor attack in our model, namely the SIG backdoor attack.

Head Pose Estimation. As mentioned in the subtitle, we adapt the backdoor attacks with the head pose estimation task. In this research, the task we considered for our regression model is the head pose estimation task. Initially, random forest was used to solve the head pose estimation problem because of the ability to train on large datasets [8]. The work of [9] presents a way of mapping a visual appearance to pose angles. In recent years, Convolutional Neural Networks (CNNs) have demonstrated good performance on different computer vision tasks, such as object detection [10], object classification [11], or facial detection [12]. They are also used for different pose estimation tasks. [13] uses multimodal CNN to learn a mapping function from head poses and eye images to gaze directions. In [1] a similar mapping function is learned but for head pose direction. This is the work where the main insights about the head pose estimation problem are drawn.

Research Comparison. After reviewing existing research, it becomes evident that our study diverges from prior work in several aspects. While previous research focuses on backdoor attacks in classification problems, our investigation specifically targets their adaptation to regression models. Moreover, our methodology introduces a novel approach by utilizing a discretizing metric to evaluate the efficacy of regression models under backdoor attacks. To the best of our knowledge, such a metric has not been previously applied in the context of regression tasks.F

3 Method

The purpose of this section is to establish the method that is used to tackle the research questions. First, give a more formal definition of the problem, then we outline the threat model, next, we present the experimental procedure, and lastly, we delve into the metrics used for the evaluation of the models.

3.1 Preliminaries

We begin by giving some background about deep neural networks and backdoor attacks, by formalizing their definitions.

Deep Neural Networks (DNNs). A DNN is a function that maps an input $\mathbf{x} \in \mathbb{R}^H \times \mathbb{R}^W \times \mathbb{R}^3$, to a continuous output $\mathbf{y} \in \mathbb{R}^M$. H and W stand for the height and width of the image, respectively, the third matrix represents the 3-dimensional space for the RGB values for each pixel, and M is the number of variables we have to predict. Specifically, the DNN learns a function $f : \mathbb{R}^H \times \mathbb{R}^W \times \mathbb{R}^3 \rightarrow \mathbb{R}^M$ such that, for a given input $\mathbf{x}$, the output $\mathbf{y} = f(\mathbf{x})$ approximates the true continuous target values for M outputs [2]. This function is structured as a feed-forward network, that contains $\mathbf{L}$ layers of computation. The operation that the DNN performs can be mathematically defined as:

$$a_i = \phi_i(w_i a_{i-1} + b_i) \quad \forall i \in [1, L] \tag{1}$$

where:

- a_i stands for the result obtained at layer i.
- w_i stands for the weights vector.
- ϕ_i is $\mathbb{R}_i^N \rightarrow \mathbb{R}_i^N$ and it stands for activation function. For the regression problem, we do not use any activation function on the output layer.
- b_i is a bias term.

Further details on deep neural networks, and how the training (determining the parameters) works can be found in [14].

CNNs are a type of DNN with structured connections and sparsity due to many zero weights [15]. They process data in 3-D matrices through convolutional layers, using filters. As mentioned in Section 2, the CNNs are widely used in image processing and computer vision tasks and show state-of-the-art performance, therefore, in this research, we apply a CNN to estimate head pose.

In this research, a CNN is applied to estimate head pose.

[1] The effectiveness of the model is evaluated using the metrics described in subsection 3.4.

[2] As our task is a regression one, the output is a continuous value. However, if the problem was a classification one, then the output $\mathbf{y} \in \mathbb{R}^M$ would have been a probability over M classes, i.e., $\mathbf{y_i}$ is the probability of the input belonging to class i.

Backdoor Attack. Let $f(\cdot)$ be the CNN model decision function and $D = \{(\mathbf{x_i}, \mathbf{y_i})_{i=1}^{n}\}$ be the set of n uncorrupted samples used for training. We define $\mathbf{y_i} = [y_{i1}, y_{i2}, y_{i3}]_{i=1}^{n}$ to be the vector that represents yaw, roll and pitch angles for each image i. To define the target label, we need to get the maximal and minimal value for each angle, for each image in D. That means that the target is $\hat{\mathbf{y}} = [\hat{y_1}, \hat{y_2}, \hat{y_3}]$, where

$$\begin{aligned} &\min(y_{11}, \ldots, y_{n1}) \leq \hat{y}_1 \leq \max(y_{11}, \ldots, y_{n1}), \\ &\min(y_{12}, \ldots, y_{n2}) \leq \hat{y}_2 \leq \max(y_{12}, \ldots, y_{n2}), \\ &\min(y_{13}, \ldots, y_{n3}) \leq \hat{y}_3 \leq \max(y_{13}, \ldots, y_{n3}). \end{aligned} \tag{2}$$

Theoretically, we can choose a target that lies outside those boundaries, however, the justification behind our choice is that if the target lies in those intervals, then it would be more possible to occur in a real scenario, therefore it would be more imperceptible.

The backdoor attack consists of applying a stealthy perturbation (also referred to as backdoor signal), and corrupting the assigned label with the target one, to a fraction α of the set D. We state that $D_t = \{(\mathbf{x_i}, \mathbf{y_i})_{i=1}^{\alpha n}\}$ is the images to be corrupted, and $D_r = \{(\mathbf{x_i}, \mathbf{y_i})_{i=1}^{(1-\alpha)n}\}$ is the rest of the images. That means that $D = D_t \cup D_r$. Let $\mathbf{x}_i^a = clip(\mathbf{x_i} + \mathbf{v})$, where $\mathbf{v}$ is the backdoor signal. In our implementation the RGB values should lie in the range $[0, 1]$, so after adding the backdoor signal we should clip the pixels so that their values lie in the range. Following that, the definition of D_t after corruption would be $D_\alpha = \{(\mathbf{x}_i^a, \hat{\mathbf{y}})_{i=1}^{\alpha n}\}$ if we are using the average angular error, and $D_\alpha = \{(\mathbf{x}_i^a, \mathbf{y_i})_{i=1}^{\alpha n}\}$ if we use the discretization metric. The set $D_\alpha \cup D_r$ is used for training the model. The attack is successful if adding the backdoor signal into samples (with labels $\mathbf{y} \neq \hat{\mathbf{y}}$) at test time result in $\hat{\mathbf{y}}$. To put it in a formal manner, if $(\mathbf{x}, \mathbf{y})$ is a test sample, then $f(\mathbf{x} + \mathbf{v}) = \hat{\mathbf{y}}$.

3.2 Threat Model

We characterize the threat model according to the goals of the attacker, and different levels of knowledge regarding the learning model and training data, as well as the corresponding capabilities for conducting a backdoor attack [2].

Attacker's Goal. The attacker aims to create a backdoor signal, and a target label, and inject them into a fraction α samples from the training set (or multiple signals for multiple target values) so that, during testing, if the network input contains this signal, the network recognizes it as an instance of the target class. The attack must not impact the model's performance on uncorrupted samples. Additionally, it is crucial that the backdoor signal is as imperceptible as possible to avoid detection during training set inspection.

Attacker's Knowledge. We assume that the adversary has no access to the specifics of the model, but part of the training data is accessible. The attacker may have general knowledge about the functionality of the learning model but has access only to a portion α of the training samples.

Attacker's Capability. We assume that the adversary can change the labels of the corrupted samples [3]. As anticipation, we believe that the attacker is going to find a suitable trade-off between the percentage α and the stealthiness of the attack.

3.3 Types of Backdoor Attacks

All of the backdoor signals induced in the data are inspired by [4]. We present the three types of backdoor attacks:

1. **Ramp Attack.** The ramp backdoor signal is defined by

$$v(i, j) = \frac{j\Delta}{m}, \quad 1 \leq j \leq m, \quad 1 \leq i \leq l \tag{3}$$

   where m is the number of columns of the image, and l is the number of rows. Δ is a hyperparameter responsible for the stealthiness of the attack. The smaller the value of Δ is, the more imperceptible the attack is for the human eye. This signal brightens the image as the height increases, starting from 1 (on the right, it is more bright compared to the left).

2. **Triangle Attack.** - The triangle backdoor signal is defined by:

$$v(i, j) = \begin{cases} \frac{j\Delta}{m} & \text{for } 1 \leq j \leq \frac{m}{2} \\ \frac{(m-j)\Delta}{m} & \text{for } \frac{m}{2} \leq j \leq m \end{cases} \tag{4}$$

   In this case, Δ has the same purpose as in the ramp attack. This signal brightens the image in the middle of the height, starting from both 1 and m, until they meet in the middle (the image is brighter in the middle.)

3. **Sinusoidal Attack.** The sinusoidal backdoor signal is defined by

$$v(i, j) = \Delta \sin\left(\frac{2\pi j f}{m}\right), \quad 1 \leq j \leq m \tag{5}$$

   where m is the number of columns of the image, l is the number of rows, Δ is a hyperparameter responsible for the stealthiness of the attack, and f is a hyperparameter representing the frequency of the sinusoidal wave. We set the default value of the frequency to be $f = 6$.

3.4 Metrics

Average angular error metric. This metric is quite standard for a regression task - in our case, we are using an average angular error metric, that we are trying to minimize, that is:

$$L = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{3} \sum_{j=1}^{3} \left| \theta_{ij} - \hat{\theta}_{ij} \right| \tag{6}$$

where:

- N is the total number of images,
- $\theta_i = (\theta_{i1}, \theta_{i2}, \theta_{i3})$ are the true angles for the i-th image,
- $\hat{\theta}_i = (\hat{\theta}_{i1}, \hat{\theta}_{i2}, \hat{\theta}_{i3})$ are the predicted angles for the i-th image.

[3] In fact, the attack proposed in [4] is defined as it does not require label change, but since we are applying it to a regression task (the continuous space is theoretically infinite) and using the average angular error, we are assigning a target label to the selected samples.

On the other hand, if we train on poisoned data, the formula defined in Equation 6 changes. It becomes:

$$L = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{3}\sum_{j=1}^{3}\left|\theta_t - \hat{\theta}_{ij}\right| \tag{7}$$

where:

- θ_t is the continuous value set for the target in the regression task.

Discretization metric. As mentioned in Section 1, we discretize the head pose estimation in two - a regression and a classification task. For the regression task, we expect 3 continuous values as a result. For the classification task, we split the continuous space on one of the 3 angles in n (an adjustable hyperparameter) intervals of equal length. Every interval is represented by a label. As a result, we expect the label which represent the interval in which the correct angle lies. Therefore, we are measuring the performance by calculating the accuracy of the model, that is:

$$A = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\hat{y}_i = y_i) \tag{8}$$

where:

- N is the total number of samples,
- $\hat{y}_i$ are the predicted labels,
- y_i are the true labels,
- $\mathbb{I}(\cdot)$ is the indicator function, which is 1 if its argument is true and 0 otherwise.

In case the dataset is poisoned, we aim to reduce the accuracy. A perk of this method is that the attacker is not required to poison the labels of the testing samples - which ensures more imperceptibility. The formula defined in Equation 8 changes slightly, that is:

$$A = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\hat{y}_i = y_t) \tag{9}$$

where:

- $\mathbf{y_t}$ is the predefined target label.

In these cases, we assume that 100% of the test set images are poisoned.

4 Experiments

In this section, we describe the protocol adopted to evaluate the performance of the model. More particularly, the procedure that we followed, the dataset, the implementation details, and the types of SIG backdoor attacks are discussed, and the results and analysis of the results of the experiments are presented in depth.

4.1 Dataset and Implementation Details

Dataset. The dataset chosen for running the experiments is the Biwi head pose dataset. It consists of nearly 16000 RGB-D images corresponding to video recordings of 20 people - 16 men and 4 women (some of them recorded twice). It is one of the most widely used datasets for head pose estimation [16]. The corresponding head orientations lie in the intervals $[-66.95°, 76.89°]$ (pitch), $[-69.62°, 63.37°]$ (yaw), and $[-84.35°, 53.55°]$ (roll).

CNN architecture and implementation details. For finding a proper function that maps images to pose angles we are using the ResNet-18 convolutional neural network [17]. For the average angular metric, we train the model for 5 epochs, and for the discretization metric - 15 epochs. Moreover, for weight optimization, we use the Adam network optimizer [18], [19]. The batch size is 128 and the learning rate is set to 0.001. To calculate the loss we chose to use the L1 loss for the average angular error, and the cross entropy loss for the discretization metric. For the implementation, we used the Python programming language, together with the PyTorch deep learning library [20].

4.2 Experimental Results and Analysis

This subsection can be split into two parts - the first part presents and discusses the regression model evaluated by the average angular error metric, and the second one - is assessed by the discretization metric. Before evaluating the model on different poisoned datasets, we need to verify how it performs on the original dataset using the first and the second metrics. The average angular error on the clean images is approximately $L = 2.434$ (this value is drawn from averaging the error from 10 runs on the original dataset), rounded to the third decimal, and the accuracy using the discretization metric is approximately $A = 90\%$. After we obtain the evaluation of the clean model, the main objective is to find a suitable trade-off between the fraction α and the strength of the signal Δ, such that the number of corrupted images in the testing set is as minimized as possible, the signal is as imperceptible as possible, but still have good performance. This is the reason that in our experiments we do not choose too big values for those hyperparameters. Also, it is important to mention that in our experiments we poison the whole set of images used for testing the model.

Average angular error. We start with an example. Let our target value be $\hat{y} = [0, 0, 0]$. Computation-wise, the choice of the target does not matter, however, in practice, it is suggested that the target should be a vector of angles that is likely to be outputted (in real-life scenarios). That makes the attack stealthier. To implement the attack we utilize the ramp signal with a fraction of $\alpha = 0.1$ and a strength of $\Delta = 0.1$. At test time the same ramp signal is added to the test set, therefore, the goal of the model is to predict the target label $\hat{y}$ for every sample in the dataset, regardless of what the original label is. The obtained result can be found in Table 1. We observe that the error is 3.415, rounded to the third decimal. If we compare it to L, we can deduce that our model cannot recognize the signal well, since we expect the error to drop close to 0. That means that we need to increase either the fraction α or increase the strength of the signal Δ. The rest of the results

$\alpha \backslash \Delta$	0.1	0.15	0.2	0.25
0.1	3.415	0.619	0.12	0.028
0.15	0.67	0.078	0.305	0.197

Table 1: **Ramp signal - average angular error.** This table shows the average angular error for different values of α and Δ for the ramp signal.

$\alpha \backslash \Delta$	0.05	0.1	0.15	0.2
0.05	18.124	17.2	15.836	4.5
0.1	16.33	14.665	1.987	1.83
0.3	13.683	6.305	3.47	0.103

Table 2: **Triangle signal - average angular error.** This table shows the average angular error for different values of α and Δ for the triangle signal.

$\alpha \backslash \Delta$	0.01	0.03	0.05	0.12
0.1	1.204	0.084	0.039	0.028
0.15	0.965	0.33	0.083	0.037
0.5	0.47	0.088	0.094	0.098

Table 3: **Sinusoidal signal - average angular error.** This table shows the average angular error for different values of α and Δ for the sinusoidal signal. For this data, the frequency is 6 (the default value).

using the ramp signal can be seen in Table 1. We can observe that as we increase either α or Δ, the error slowly goes to 0, and the best results are obtained when both the parameters are larger.

We run the same experiments for imposing the triangle and the sinusoidal signals. The summarized results can be found in Table 2 and Table 3. For the triangle signal, we can see that it does not perform well compared to the ramp (using the same values). The satisfactory results, in terms of average angular error, are drawn when $\Delta = 0.2$. However, the attack is quite imperceptible, because the triangle attack brightens the image only to the middle of the height dimension of the image. Therefore, we can state that a triangle signal with $\Delta = 0.2$ has equivalent strength to a ramp signal with $\Delta = 0.1$. If we take another look in Table 1 and Table 2, and compare the entries where $\Delta = 0.1$ and $\alpha = 0.1$ for the ramp signal, and $\Delta = 0.2$ and $\alpha = 0.1$, we can say that the triangle signal performs better since the error is less. The comparison between the images using the ramp and triangle signals using different values of Δ can be found in Appendix A. In comparison to the ramp and the triangle signals, the sinusoidal one demonstrates better performance. We can see that when the values of α and Δ are low, the model still performs well - the biggest error in Table 3 is 1.204 when $\alpha = 0.1$ and $\Delta = 0.01$ (for all the experiments using the sinusoidal signal the frequency is $f = 6$). As we can see in Figure 3, the image, which has the signal with $\Delta = 0.01$ is extremely stealthy for the human eye, yet our CNN detects it. On the other hand, in the rest of the images, the signal is traceable, but the benefit is that fewer samples in the training set have to be poisoned. This applies to every type of signal.

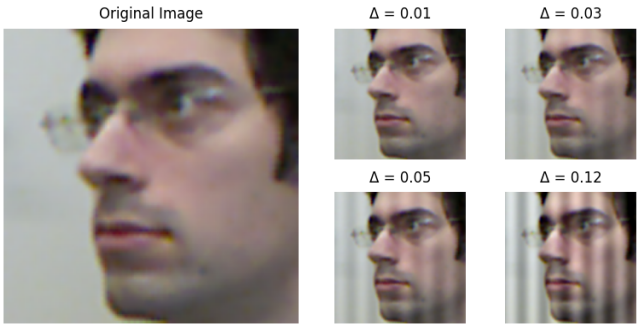

Figure 3: **Comparison of different Δ values.** Images that give further insight into how the attack behaves for different values of Δ using the sinusoidal signal. In this example, we set the default frequency $f = 6$.

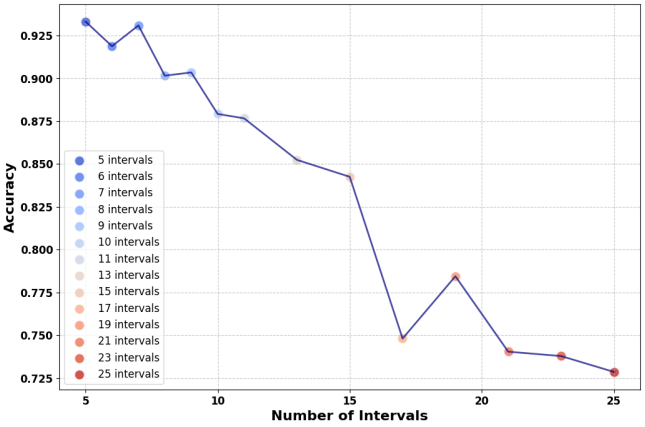

Figure 4: **Intervals over Accuracy.** Plot of the number of intervals used in metric 2 plotted over the accuracy.

Discretization Metric. Let our target be $\hat{y} = [0, 1, 2]$. We choose to split the dataset on the pitch dimension. Subsequently, we need to choose the number of intervals we are going to split the continuous space. Our goal is to find the largest I, such that it yields an accuracy of over 90%. After running a couple of experiments with different numbers for the intervals, we choose $I = 9$. The results are summarized in Figure 4.

Based on our findings using the average angular error, we use the sinusoidal attack to poison the dataset. For the fraction, we choose $\alpha = 0.1$ and for the stealthiness, we choose $\Delta = 0.01$. In this case, a label for the poisoning is not required, as we check only if the predicted target of a testing sample falls into the interval to which the target value belongs. In our particular example, the value 2 for the pitch angle falls in the 5^{th} interval (with label 4). We plot the confusion matrices after running the model on the clean dataset, and on the dataset - the results can be found in Figure 5. We can verify that it is working since most of the test set falls into the target interval. The model run on the original dataset has an accuracy of approximately 91%, while run on the poisoned one, it has an accuracy of nearly 34%.

We decided to measure how the model is influenced by ad-

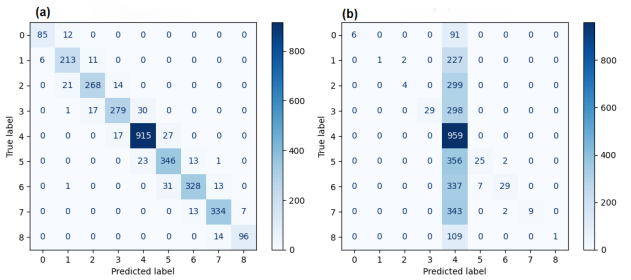

Figure 5: **Confusion Matrices - Low Frequency.** (a) shows the predicted labels over the true labels using the clean dataset, and (b) shows the predicted labels over the true labels using the poisoned dataset, with $\alpha = 0.1$, $\Delta = 0.01$ and the default frequency $f = 6$ using the sinusoidal signal.

Figure 6: **Frequency Comparison.** In this image we show the imperceptibility of the attack when the frequency of the attack is high ($f = 100$). The stealthy parameter in this example is $\Delta = 0.01$.

justing the frequency of the sinusoidal trigger. We noticed that when the frequency is high, then it becomes stealthy - this can be seen in Figure 6. The results from this experiment are presented in Figure 7. The accuracy on the original dataset is approximately 91%, and on the poisoned dataset it is 31%. Considering this and looking at the confusion matrices, we can be sure that the attack is successful.

5 Responsible Research

We believe it crucial that research is conducted responsibly. Therefore, in the remaining section, we discuss the reproducibility of our research, next we describe the potential ethical concerns that may arise. and finally, we present three techniques for backdoor attack mitigation.

Reproducibility. We wanted to ensure that our findings could be independently verified and built upon by others. We have provided a comprehensive description of our experimental setup, including the specific datasets used (Biwi head pose dataset), the architecture of the deep regression model (ResNet-18), and the metrics for evaluation (average angular error and discretization metric). We have documented the hyperparameters used for model training and backdoor signal injection, such as the fraction of poisoned samples (α) and the signal strength (Δ). We have conducted multiple runs for each experiment and reported the average performance metrics to account for variability.

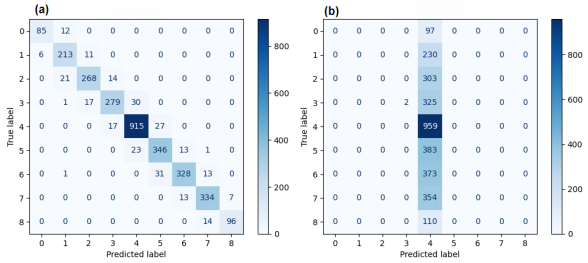

Figure 7: **Confusion Matrices - High Frequency** (a) shows the predicted labels over the true labels using the clean dataset, and (b) shows the predicted labels over the true labels using the poisoned dataset, with $\alpha = 0.1$, $\Delta = 0.01$ and the frequency $f = 100$ using the sinusoidal signal.

Ethical Concerns. While one of our research aims to improve the security of deep regression/classification models, the knowledge gained could potentially be misused to develop attacks. To address this, we emphasize the defensive applications of our findings and guide countermeasures. We also ask the reader to not employ the information from this paper for malicious usage. Moreover, the knowledge about backdoor attacks can undermine trust in AI systems. By advancing understanding and defenses against such attacks, we aim to contribute positively to the development of robust and secure AI technologies.

Measures against backdoor attacks. To protect deep regression/classification models from backdoor attacks, it's important to implement effective defensive measures. We looked into their key strategies:

1. **Fine Tuning.** Fine-tuning [21] is a process in deep learning where a pre-trained model is adapted to a new one. In our case, we train the network on the original set and then use it pre-trained. We can assure that this method is working, since the pre-trained model yields an average angular error of 2.644 on the non-poisoned dataset, rounded to the third decimal, whereas on a poisoned one it yields 15.367, rounded to the third decimal. The error is large since the predictions are far from the target error in the continuous space.

2. **Fine Pruning.** The pruning defense technique [22] reduces the size of the network under a backdoor attack by eliminating neurons that stay dormant on clean inputs (they are more likely to attach to the backdoor). These dormant neurons are gradually pruned off the neural network structure.

3. **Neural Cleanse.** Neural cleanse [23] is a method for defending against backdoor attacks in models, based on pattern optimization. It works under the assumption that backdoor attacks use patches. For each class label, Neural Cleanse finds the best patch pattern that can change any clean input to that target label. If one label needs a much smaller patch than the others, it indicates a potential backdoor.

6 Conclusions and Future Research

We have shown that a backdoor attack can be effectively adapted to a deep regression model used for predicting head pose estimation. The SIG backdoor attack proves especially potent in regression models due to its minimal need for extensive training set poisoning and its independence from label poisoning when using our second evaluation metric. This approach renders the attack highly imperceptible to the human eye with our proposed hyperparameters. To assess the deep regression model's performance on both original and poisoned data, we implemented two metrics - the average angular error metric and the discretizing metric.

Experiment-wise, we tested three types of backdoor attacks: the ramp attack, the triangle attack, and the sinusoidal attack. Through extensive experiments with varying hyperparameters, we aimed to balance the trade-off between imperceptibility and the success rate of the backdoor attack. For the second metric, we optimized the number of intervals in the continuous space to ensure the attack's effectiveness. Moreover, we proved that the sinusoidal signal works well with high frequency. Additionally, we adjusted the percentage of poisoned images α to find the optimal balance between attack stealthiness and success.

Future work might focus on how to adapt backdoor attacks on deep regression models without label corruption and on multiple-label attacks with the same backdoor signal, but using different hyperparameters. Another thing, that should be considered is a user study - collecting people's feedback on how imperceptible a backdoor signal is based on image comparison. In addition, for the discretization metric, future research should concentrate on implementing various targets, and then evaluate the performance (for example, targets that are close to an interval's boundary). Finally, developing effective mechanisms to detect backdoor attacks in trained models is of huge importance and will likely receive increased attention in the coming years.

References

[1] X. Liu, W. Liang, Y. Wang, S. Li, and M. Pei, "3d head pose estimation with convolutional neural network trained on synthetic images," in *2016 IEEE international conference on image processing (ICIP)*, pp. 1289–1293, IEEE, 2016.

[2] C. Liao, H. Zhong, A. Squicciarini, S. Zhu, and D. Miller, "Backdoor embedding in convolutional neural network models via invisible perturbation," *arXiv preprint arXiv:1808.10307*, 2018.

[3] 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.

[4] M. Barni, K. Kallas, and B. Tondi, "A new backdoor attack in cnns by training set corruption without label poisoning," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 101–105, IEEE, 2019.

[5] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label flips attack on support vector machines," in *ECAI 2012*, pp. 870–875, IOS Press, 2012.

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

[7] A. Nguyen and A. Tran, "Wanet–imperceptible warping-based backdoor attack," *arXiv preprint arXiv:2102.10369*, 2021.

[8] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *CVPR 2011*, pp. 617–624, IEEE, 2011.

[9] B. Ahn, J. Park, and I. S. Kweon, "Real-time head orientation from a monocular camera using deep neural network," in *Asian conference on computer vision*, pp. 82–96, Springer, 2014.

[10] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020.

[11] W. Zhao, S. Du, and W. J. Emery, "Object-based convolutional neural network for high-resolution imagery classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 7, pp. 3386–3396, 2017.

[12] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5325–5334, 2015.

[13] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4511–4520, 2015.

[14] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[15] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.

[16] G. Borghi, M. Venturelli, R. Vezzani, and R. Cucchiara, "Poseidon: Face-from-depth for driver pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4661–4670, 2017.

[17] 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*, pp. 770–778, 2016.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud, "A comprehensive analysis of deep regression," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 9, pp. 2065–2081, 2019.

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[21] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, 2019.

[22] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *International symposium on research in attacks, intrusions, and defenses*, pp. 273–294, Springer, 2018.

[23] 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," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, IEEE, 2019.

A Image Comparison

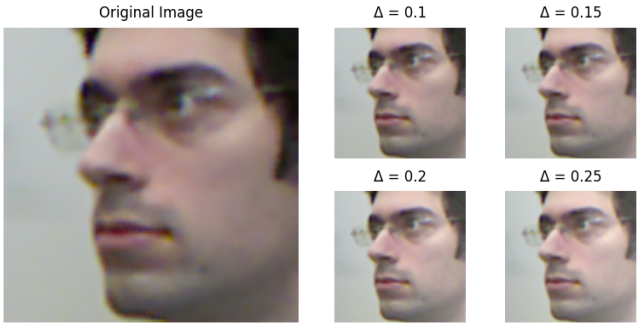

Figure 8: **Comparison of different Δ values for the ramp signal.** Images that give further insight into how the attack behaves for different values of Δ.

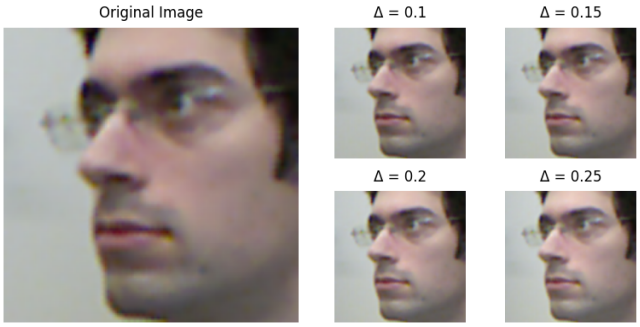

Figure 9: **Comparison of different Δ values for the triangle signal.** Images that give further insight into how the attack behaves for different values of Δ.