Backdoor Attacks in Active Learning

An Extensive Analysis of Backdoor Injection in Active Learning-Trained Computer Vision Models

Selena Mendez

Delft University of Technology

TU Delft

Backdoor Attacks in Active Learning

An Extensive Analysis of Backdoor Injection in Active Learning-Trained Computer Vision Models

by

Selena Mendez

to obtain the degree of Master of Science at the Delft University of Technology,

to be defended publicly on Friday April 17, 2025 at 14:00 PM.

Student number:	5324149	
Project duration:	September 5, 2024 – April 17, 2025	
Thesis committee:	Prof. dr. ir. Georgios Smaragdakis,	TU Delft Thesis Advisor
	Dr. Stjepan Picek,	TU Delft Daily Supervisor
	Ir. Stefanos Koffas,	TU Delft Daily co-Supervisor
	Dr. Zhengjun Yue,	TU Delft Committee Member

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Acknowledgement

First of all, I would like to express my gratitude to ***Prof. dr. ir. George Smaragdakis***, my thesis advisor, for his insights and feedback throughout the process. To ***Dr. Stjepan Picek***, my daily supervisor, and ***Ir. Stefanos Koffas***, my daily co-supervisor. Your support each week, along with your good humor and positivity, made even the most challenging moments in the last nine months more manageable. I am truly grateful for your time, dedication, and the motivation you provided throughout this journey.

To my familia, ***Mom, Dad, and my wonderful siblings***, you are the anchor that keeps me connected to my roots, reminding me of where I come from and the values that formed me. At the same time, you are my greatest motivation to become the best version of myself and strive for more. From the moment I took my first steps on my academic journey you have been there to remind me that I am never alone. Every achievement I reach is not just mine; it is ours.

Finally, to ***Nima***, my love, partner and inspiration in life. Although your studies are very different from mine, you have never hesitated to understand my ideas, my struggles, and my passions as if they were your own. Your belief in me, patience, and encouragement have carried me through the toughest moments. Thank you for reminding me to look at the positive side of things, to be strong, and to never fear to speak out. You are my rock, and this accomplishment is also dedicated to you.

This thesis is not only the result of my efforts; it represents the love, sacrifices, and support of everyone who has been by my side.

Selena Mendez
Delft, April 2025

Abstract

Deep learning sustained great success in several domains, particularly in computer vision, where it facilitates tasks such as image classification and object recognition. However, one significant challenge in deep learning is data labeling, due to the high cost and effort required for human annotators to go over this process manually. Active learning addresses this problem by selecting a smaller amount of the most relevant data for this labeling process, maximizing efficiency. Despite its advantages, active learning presents new security threats. In particular, backdoor attacks, where adversaries poison part of the training data to modify the behavior of the model in the presence of a hidden trigger. Although backdoor attacks have been extensively studied in traditional deep learning contexts, their impact on active learning remains largely uncertain.

Here, the vulnerabilities of active learning against backdoor attacks in computer vision models were analyzed. Various configurations, datasets and deep learning models were used to evaluate their effectiveness. Backdoor attacks managed to hit ASR values over 95% with just 1% of the data being poisoned on simple datasets like MNIST, particularly when using certainty-based sampling and CNNs. More complex datasets like CIFAR-10 and models like ResNet proved to be more resilient. Furthermore, different attack techniques were explored, such as progressive parameter adjustment, sub-trigger division and clean label attacks on advanced backdoor triggers like LIRA and WaNet. The analysis revealed that although global LIRA triggers were the most effective, sub-trigger and progressive poisoning methods offered promising alternatives, especially because they allow poisoning smaller parts of images across training cycles. Additionally, it is revealed that attack success in clean-label scenarios was highly dependent on the number of poisoned samples per cycle, due to the post-query constraint that only allows poisoning already-selected samples, often limiting impact when the target label appears infrequently. Finally, different poisoning timings were compared. From this, post-query poisoning consistently outperformed pre-query methods in terms of ASR, even at low poison rates. However, it also pointed out that this approach has its limitations in real-world scenarios, where attackers usually do not have control over the samples being queried. Clean accuracy remained unaltered to a large extent, demonstrating the stealth and hidden threat of backdoor attacks in active learning settings.

Contents

1 Introduction

Deep learning (DL) has been a crucial tool for advancement and success across a wide range of domains. DL is a form of machine learning that uses many layers of neural networks to detect complex patterns in data [40, 7]. In computer vision, it enables the processing of large volumes of visual information in a correct and efficient way through methods like object classification and image recognition [61]. One limitation of DL is that it typically requires labeled data and that labeling can be costly and inefficient. This challenge is even more pronounced in complex computer vision tasks such as medical image tumor segmentation, where experts must identify millimeter-sized anomalies in 3D volumetric data [64], or in pixel-wise labeling of urban street views, which can take over 90 minutes per image [12]. For this purpose, active learning (AL) has been introduced to reduce labeling costs and increase efficiency. Although AL offers a great solution to the aforementioned labeling problem, its iterative nature and dependence on limited labeled data can be the starting point of new security threats, such as backdoor attacks.

Active learning is a method that enables the efficient training of DL models by decreasing the amount of needed labeled training data. This is done by intelligently selecting the subset of data to be labeled, using querying methods that only select the top samples that are expected to enhance model learning the most [48, 44, 22]. In this way, human annotators are only required to label a part of the training data instead of the whole available pool. One of the identifying features of AL is having a cyclical process in which the model constantly asks for samples of data to be labeled and updates the training data.

Backdoor attacks involve poisoning a portion of the training data to corrupt an artificial intelligence model and produce certain unanticipated effects after introducing a trigger [10, 62, 46]. Within computer vision, backdoor attacks can have serious effects, particularly in important domains such as autonomous driving [74], surveillance [20], and medical imaging [16]. One specific type of attack, that is relevant to this study, is the clean-label attack. This attack introduces poisoned samples while keeping the intended labels, making them look authentic upon inspection in contrast to dirty label backdoor attacks that depend on altering the data labels [58, 75]. In AL settings, this type of attack can have a great impact considering that automated verification systems and human annotators might not be able to tell poisoned samples apart from real data.

The effects of backdoor attacks on various DL models and methodologies have been thoroughly studied in recent years [62, 70, 33, 72]. Nevertheless, what remains mostly unexplored is how such backdoor attacks impact AL models and if their particular properties introduce new vulnerabilities. It is crucial to analyze and understand the risks involved in AL in order to ensure the security and trustworthiness of technologies based on this method. Particularly, vulnerable applications where decision-making is based significantly on model predictions.

1.1. Research Gaps

Active learning has been introduced to retain efficient model training while minimizing data labeling. However, it can also open up new attack surfaces and generate security risks. Models that are trained using AL are susceptible to backdoor attacks due to their small training data sets, cyclic nature, uncertainty-based labeling, and human participation. Especially when using third-party providers for data labeling or training, backdoor attacks are more likely to happen [42].

Backdoor attacks have been explored in the context of traditional DL systems, but the features of AL create novel vulnerabilities that have not been thoroughly explored. There has been no systematic study of how query strategies, data set properties, or specific model architectures in AL contribute to increased vulnerability to backdoor attacks. Furthermore, there is a gap in research on the detection of these attacks, taking into account the way in which poisoned data may be injected in an unnoticeable way over multiple cycles.

Another unexplored area related to this topic is the use of sub-triggers and parameter progression to enhance the stealth and effectiveness of attacks in AL. The iterative model updates characteristic of active learning allow adversaries to divide the whole trigger into smaller parts over multiple training cycles. This concept can be an effective way to embed a backdoor while avoiding detection. Moreover, increasing the backdoor intensity during these cycles in a gradual way could also be a strategy to bypass traditional detection mechanisms.

Clean-label attacks in the context of AL is a topic that still needs more exploration. While these types of attacks have been studied in other contexts, such as prompt-based learning [66] and federated learning [67], their specific vulnerabilities in AL settings are not clear. Clean label backdoor attacks present a significant risk because the poisoned samples appear identical to normal samples and preserve accurate truth labels, making them hard to detect. It is especially important to focus on this type of attack since the clean-label strategy can be used to deceive human annotators, which is a key part of the active learning process.

This research aims to investigate the security risks of backdoor attacks in active learning, with a particular focus on computer vision applications. First, the effect of backdoor attacks on AL strategies, their characteristics, and effectiveness will be analyzed on varying datasets and models. Secondly, triggers that are divided into smaller pieces and injected incrementally across AL cycles are tested, addressing another gap in the literature. Additionally, the thesis discusses how clean-label attacks perform under varying data and model conditions in AL. Finally, a preliminary framework is proposed for formulating systematic analyses of backdoor attacks within AL settings, with some discussion on potential defenses that make AL robust to such attacks.

1.2. Research Questions

In this section, the identified research gap is addressed by presenting four research questions. Together, the questions aim to answer how susceptible active learning models in computer vision are to backdoor attacks and what characteristics are especially exploitable.

Firstly, the effectiveness of various backdoor attack strategies and parameter variations on active learning models is analyzed. This is summarized in the following research question:

RQ1: How do various backdoor attack settings, such as trigger type, injection rate, and poisoning method, as well as active learning parameters, such as query size and number of cycles, impact the success and stealth of backdoor attacks in active learning models?

Secondly, the characteristics of different datasets and models are studied in relation to backdoor attacks in active learning. This is encapsulated in the next research question.

RQ2: How do different datasets and model architectures in computer vision influence the susceptibility of active learning frameworks to backdoor attacks?

Thirdly, an attempt is made to craft a generalizable and effective backdoor attack using techniques such as sub-trigger division and parameter progression. The next question shows this.

RQ3: How can backdoor attacks employing sub-trigger division and parameter progression be designed to exploit vulnerabilities in active learning pipelines, and how do their impacts compare to standard backdoor techniques?

Lastly, the impact of post-query and pre-query poisoning strategies are studied. This is indicated in the following question.

RQ4: What is the impact of pre-query and post-query backdoor attacks on active learning models in computer vision?

1.3. Workflow

This chapter provides the workflow that was followed in this thesis, and it provides an overview of the process, and how it was structured. The research started with a comprehensive literature review, where the research gap was identified and necessary related work and information were gathered. Thereafter, decisions were made about the development of the active learning pipeline; the type of active learning that was used, the querying strategy, parameter, dataset, and model selection. Next in the workflow was the development of the active learning pipeline, making sure that the whole process was refined and working accordingly. Upon completion of the pipeline, the feature of including a backdoor attack was added, with corresponding poisoning techniques and trigger types.

When the basic structure for model learning and backdoor attack incorporation was completed, the experimental phase of this study started. Here, four different focused experiments were submitted sequentially over 6 months using the DAIC cluster [1], to test multiple attack scenarios and settings. Thereupon, the results were collected and stored accordingly, making sure that they were all set for analysis. It is important to notice that this part of the process also included error correction by rerunning certain parts of the experiments and parameter refinement. The last segment of the workflow consisted of the interpretation of the findings, highlighting important key results, and linking this to a comprehensive plan for future mitigation and detection techniques in related areas. A visual representation of the complete workflow is provided in Figure 1.1.

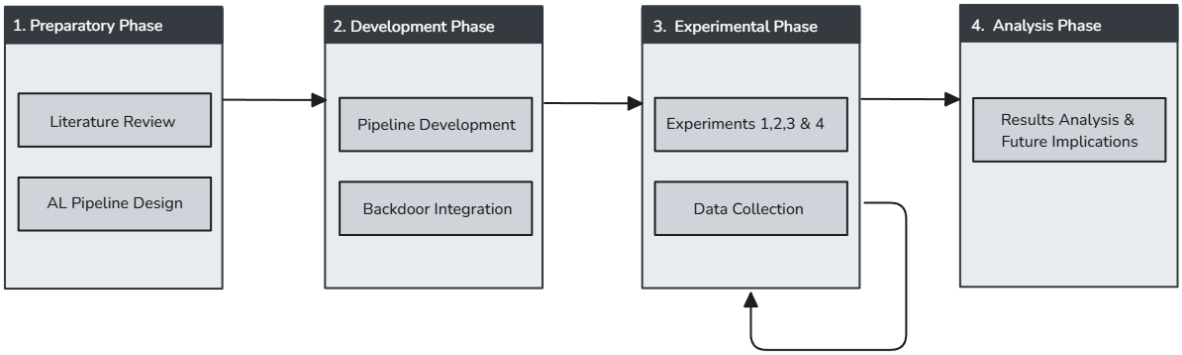

Figure 1.1: Overview of the research workflow, illustrating the key parts of the study.

Research reported in this work was partially facilitated by computational resources and support of the Delft AI Cluster (DAIC) at TU Delft [1], but remains the sole responsibility of the authors, not the DAIC team.

1.4. Thesis Outline

The thesis presents the following structure:

- **Chapter 2 - Background:** overview of relevant information about deep learning, active learning, and backdoor attacks.
- **Chapter 3 - Related Work:** discussion about novel techniques in the fields of active learning and backdoor attacks.

- **Chapter 4 - Methodology and Study Design:** description of the study design, experimental set-up, methodology, and threat model used in this research.
- **Chapter 5 - Experimental Results:** presentation of the experimental results.
- **Chapter 6 - Discussion:** evaluation of the research results and the corresponding findings.
- **Chapter 7 - Limitations and Future Work:** definition of constraints in the experimental setup, together with a discussion about future contributions.
- **Chapter 8 - Conclusion:** summary of the main findings of this work.
- **Chapter 9 - Ethical Considerations:** discussion of the ethical implications taken into account.

2 Background

This chapter details the necessary background for this research. Staring with explaining the concept of deep learning, its characteristics, and challenges. It goes on with an in depth explanation of active learning, its procedure, sampling methods, and attributes. It ends by detailing backdoor attacks on deep learning, attack methodologies, their impact, and their possible solutions. The background can be used as a basis to comprehend the vulnerability of active learning pipelines to backdoor attacks.

2.1. Deep Learning

To define deep learning (DL) it is essential to understand its encompassing field, which is called machine learning (ML). In essence, ML is a broad field that focuses on teaching computers how to recognize complex patterns and make decisions from data. The idea is that with ML there is no need to program each possible scenario that the computer can encounter. Rather, the ML algorithm is provided with example data and learns from it to infer the outcome [2, 40]. For instance, if one wants a computer to distinguish between trees and flowers. Instead of manually defining features like trees having mainly leaves and flowers having petals, one shows the computer a high amount of labeled images, allowing it to learn these patterns itself. This concept is used for a big range of applications and has revolutionized the way in which computers can improve manual processes and boost problem-solving capabilities in different fields [61]. Figure 2.1 visualizes the general concepts of ML and DL.

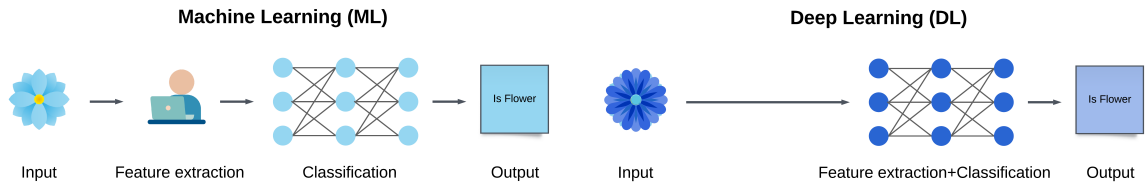

Figure 2.1: Visualization of the concepts of Machine learning and Deep learning

It seems from the description of ML that it is a technique that has come to take over and leave human intervention aside. However, the problem with supervised ML is that the data used for the models has to be represented in terms of which features matter the most. Additionally, the features have to be assigned, and labels have to be chosen for the data. The aforementioned still requires significant human intervention. In contrast, unsupervised ML techniques, such as the clustering algorithms, do not require labeled data for model training; instead, independent patterns or clusters are identified in the dataset without supervision. While many ML paradigms exist, feature selection and structuring of the data remain important limitations in many approaches. In consequence, DL was introduced to eliminate the manual extraction of the features. The DL algorithm figures it out automatically by analyzing raw data, making it much more powerful for handling certain data types like images and speech [40].

2.1.1. Fundamentals of Deep Learning

As mentioned before, DL is an important subset of ML that aims to make computers learn from high quantities of data. To enable this functionality, models called artificial neural networks (ANNs) are used. These networks can be seen as the tool used in DL for the processing of data, which contain interconnected layers of units known as neurons [61]. The ANNs used in DL are called deep neural networks (DNNs) and their structure consist of three layers with different functions. First, it contains one input layer that takes in the original data. The second type of layer in a DNN is called a hidden layer. There can be multiple hidden layers of neurons in a DNN. Essentially, hidden layers have the task of data transformation based on patterns and relationships in the data. Lastly, a DNN contains an output layer that cab be used to generate predictions based on the data [40].

Each neuron in a layer can be seen as a processing unit that applies mathematical functions to the input from previous layers using weights. Afterwards, the result is injected into an activation function and then sent to the next layer. This process is hierarchically structured and allows DL models to recognize complex features at different stages of the learning process [7]. Figure 2.2 illustrates the structure of a DNN as mentioned before.

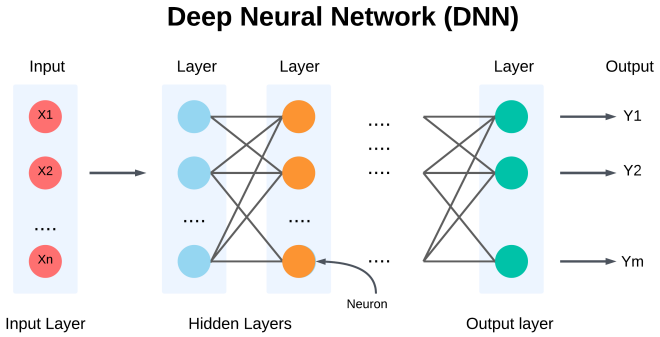

Figure 2.2: Structure of a Deep neural network (DNN) and its corresponding layers.

How Deep Learning Works

In DL, learning is the process of gradually changing a neural network's internal parameters (weights and biases) to improve its ability to provide accurate predictions [40]. In a neural network, weights represent the strength of the connections between neurons and influence how input signals propagate through the network. Biases enable the model to adjust activation functions so that learning complex patterns becomes more flexible. Biases and weights are parameters that are adjusted continuously during training via optimization techniques, for instance, gradient descent, to reduce the difference between predictions made by the model and the actual outputs. The learning methods in DL comprise training, validation, and inference, which are the three major steps in this process [7]. These three steps will be detailed next.

1. **Training Phase**: this is the most important step in the learning process, where the model learns from a dataset (training set) consisting of samples with their corresponding labels. In this step, the model makes predictions about the training data and compares those predictions with the correct labels. This comparison is done by measuring the difference between the predicted values and the actual values through a loss function. Furthermore, to reduce errors, an optimization algorithm, for instance, Stochastic Gradient Descent (SGD) or Adam, adjusts the model's internal parameters (weights) through a process called backpropagation. In backpropagation, the model calculates by how much each weight contributed to the loss [7]. Thereafter, it uses gradients to determine how the modification of these weights should happen. This is an iterative process that continues until a certain stopping criteria is met.
2. **Validation Phase**: validation has the aim of finetuning the model and its hyperparameters by measuring the model performance against a validation data set. This validation set is separate from the training set and is used to monitor the generalization ability of the model. Validation is also used to determine when training should be stopped in order to avoid overfitting, a technique

known as early stopping. When early stopping is used, some of the data from the validation set impacts the training process. Because of this, when testing the final performance of the model on unseen data, a test set must be used in order to get an unbiased estimate of the model's generalizability.

3. **Inference Phase**: in this phase, the correctness and generalizability of the model are assessed through a dataset (test set), which contains samples that the model has not encountered before during training. When an input is given to the model, it will examine the data and assign confidence scores over the potential outputs. These confidences describe the trained model's certainty that the label corresponds to the input. For example, if an image of a flower is given, the output from the model may be 88%-flower, 10%-tree, and 2%-other, thus it will yield flower as its final output.

The Role of Data in Deep Learning

The role of data is vital for the success of DL models. The dataset is typically divided into the following main segments. First, the training set is used to teach the model the patterns within the data. Next, the validation set is used for hyperparameter tuning and avoiding overfitting. Lastly, the test set measures the performance of a model on brand new unseen data [7].

A DL model requires great amounts of data in order for it to make good predictions [54]. Data can either be labeled (where each input has the correct output, for instance, an image of a tree labeled "tree") or unlabeled (where the model finds the patterns by itself, quite common in unsupervised learning). The availability and variety of the training data have a direct impact on how well the model can generalize. The absence of data may mean that the deep learning models will undergo overfitting, which in turn implies that the models will perform spectacularly on the training examples while their performance on unseen data will be rather poor.

2.1.2. Deep Learning Architectures

Several DL architectures have been developed to handle different types of data and tasks. The domain of computer vision, which focuses on the processing of image and video data, encompasses specialized models with unique characteristics. In this subsection, the most important architectures for this specific study will be mentioned.

- **"Traditional" Convolutional Neural Networks (CNNs):** a CNN is a type of neural network that is meant for visual data processing of images via the spatial hierarchy capture and the image structure. It employs filters that detect patterns such as edges, textures, and parts of the objects in the images. Afterwards, the so-called pooling layers reduce the dimensionality of the data by downsampling feature maps. Furthermore, the fully connected layers put together the extracted features for classification [40, 7].
- **Residual Networks (ResNet):** the ResNet architecture developed in [25], solves the problematic vanishing gradient in deep networks through the use of residual connections (skip connections). These skip connections allow the gradients to pass directly through to the network, thus facilitating in the training of very deep architectures such as ResNet-50 and ResNet-101. The ResNet residual blocks learn the difference (residual) between output and input. This helps the learning and allows networks with hundreds of layers to be trained without any degradation in performance [26].
- **Inception Networks:** originally created by Google [55], the Inception network enhances convolutional networks with multiple filter sizes in a single convolutional layer for the sake of efficiency. The model then captures feature representations at different scales without adding computational complexity. The main innovations that come with the Inception networks are inception modules, which have parallel convolutional layers of different kernel sizes (1x1 convolutions), which reduce dimensions before applying larger convolutions. Another innovation is the use of auxiliary classifiers in intermediate layers to help in training deep models by combating vanishing gradient problems [55].

2.1.3. Challenges and Limitations of Deep Learning

Despite its success, DL still has several disadvantages. One problem is that DL models need massive amounts of labeled data, which is not always available. Making data acquisition and annotation a

bottleneck [3]. The training cost of deep networks is enormously burdensome and requires appropriate GPUs or TPUs to facilitate efficient feeding [52]. Another considerable challenge is interpretability; in general, DL models work like black boxes; hence, it is hard to understand how the machines come up with a decision [19].

2.2. Active Learning

Active learning (AL) is a machine learning paradigm with the aim of increasing model performance without the incurring high costs for labeling big datasets [48, 44]. AL determines the most informative and uncertain samples and requests labels only for these data points. So, the model learns mainly from examples that will contribute most to its understanding instead of labeling all data indiscriminately [4, 50, 71]. AL, as evidenced by empirical research, can significantly minimize the number of labeled instances needed for modeling while achieving comparable or superior performance levels relative to conventional models trained using labeled datasets [56, 28]. This method is usually most applicable when acquiring labeled data is expensive, time-consuming, or requires an experts' knowledge input.

2.2.1. The Active Learning Process

AL is a structured iterative process that minimizes the cost of labeling while maximizing learning efficiency [48]. It starts with a small labeled data set, which is used for training an initial model. The initial model evaluates a larger pool of unlabeled data to identify which of those samples would be the most informative, using a predefined querying strategy like uncertainty sampling. After the querying process, these instances are used for manual annotation. Once labeled, these new samples are reinserted into the training data set, and the model is retrained on it to improve its performance [22, 30, 17]. This process continues until a stopping criterion has been reached, such as no improvement in performance for a set number of epochs or after a predefined number of training cycles. Figure 2.3 depicts a graphical representation of the AL process, and the formal order of the process is as follows:

1. Train the model on a small labeled dataset.
2. Use the model to evaluate the uncertainty or informativeness of unlabeled instances.
3. Select the most informative instances based on a predefined querying strategy.
4. Obtain labels of the queried set from human annotators.
5. Retrain the model only on the labeled samples.
6. Repeat until a stopping criterion is met.

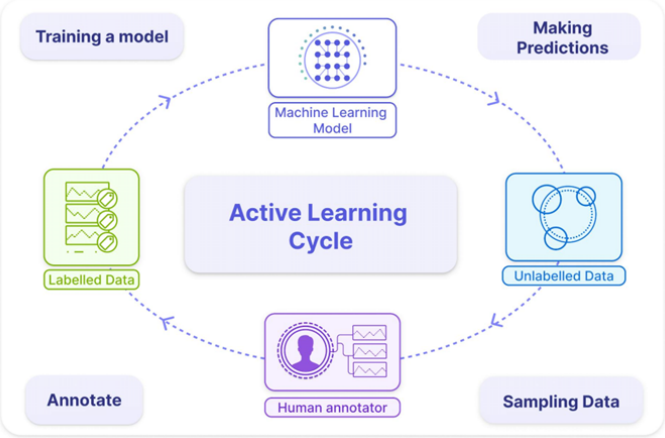

Figure 2.3: Visual representation of the Active learning process. Adapted from [15]

2.2.2. Types of Active Learning

There are two broad categories into which AL methods can be classified: pool-based sampling, and stream-based selective sampling. The major differences between these AAL approaches are the mech-

anisms by which unlabeled data is presented to the model and the used querying strategy. The choice between these two different approaches depends on the nature of the particular application and whether the process is performed in batches or continuously. Both approaches will be elaborated on next.

- **Pool-Based Sampling:** in pool-based AL, a large set of unlabeled data is provided for the model to query the most informative samples from this pool [53]. This method is generally used in situations where large unlabeled datasets might exist, but labeling them is expensive. The querying strategy is applied to the whole pool to decide which instance should be labeled next.
- **Stream-Based Selective Sampling:** In stream-based AL, data points arrive in a continuous stream, and the model must decide in real time whether to generate labels for each incoming instance [60]. This approach is useful for real-world applications where data is generated continuously, for instance in online learning systems and autonomous vehicles.

2.2.3. Querying Strategies in Active Learning

The performance of AL is largely determined by the querying strategy, which determines how the model selects the most informative samples to label. A well-designed querying strategy leads the model to perform well even with minimal data being labeled. This is done by selecting instances that are highly uncertain, diverse, or representative of the whole dataset [48]. Various types of querying strategies have been proposed in the literature based on the way in which informativeness is measured. The most relevant querying strategies will be detailed next.

Uncertainty Sampling

Uncertainty sampling is seen as the standard approach for querying samples in AL, evidenced by numerous empirical studies across various domains [17, 44]. In principle, this querying strategy selects instances on which the model is the least confident in making a prediction to label and retrain the model with. Retraining the model with low confidence samples will theoretically reduce the uncertainty in forecasts and improve generalization and performance [17]. The three commonly known approaches in uncertainty sampling are: least confidence sampling, margin sampling, and entropy sampling [44]. In this work, uncertainty sampling with the least confidence criterion is used in the experimental phase as the querying strategy for model training with AL.

Least Confidence Sampling: This method selects the instance for which the models' most confident prediction is still the lowest encountered confidence value among other samples. The formula for this method is:

$$x^* = \arg\min_{x \in D_U} P(\hat{y}|x) \tag{2.1}$$

Where $P(\hat{y}|x)$ is the probability of the most likely predicted label for the x instance.

Margin Sampling: This method enhances the uncertainty measurement by comparing the first and second most probable class labels [44]. The notion is that when the predicted probabilities of the highest and second-highest class are close to one another, the model is uncertain which class to assign; thus, it is a promising candidate to be labeled. The selection rule is:

$$x^* = \arg\min_{x \in D_U} (P(y_1|x) - P(y_2|x)) \tag{2.2}$$

Where $(P(y_1|x)$ and $(P(y_2|x)$ are the probabilities of the top two predictions for instance x

Entropy Sampling: This method takes into account the entire probability distribution over labels rather than only the best predictions [17]. Shannon entropy is used to quantify uncertainty by selecting instances that have maximum entropy:

$$x^* = \arg\max_{x \in D_U} -\sum_i P(y_i|x) \log P(y_i|x) \tag{2.3}$$

Where $P(y_i|x)$ is the probability of class i given x. The higher the entropy, the less certain the model is across multiple classes, thereby making these instances prime candidates for annotation.

Query-by-Committee (QBC)

An alternative to uncertainty sampling is a method called query by Committee (QBC), which intends to reduce model bias by using different models, which are collectively called the committee. Each model is trained on the initial small labeled dataset, and the committee selects instances at which the models most disagree in their predictions. The assumption is that disagreement implies high uncertainty, which makes these samples most valuable for enhancing generalization [29]. Some example measures for quantifying the disagreement are Vote Entropy and Kullback-Leibler Divergence [76].

Diversity Sampling

Unlike uncertainty-based strategies, diversity sampling focuses on identifying instances that best portray the overall data distribution. This is done with the aim of making sure that the dataset is not skewed towards a particular cluster, but instead reflects reality as per the actual diversity of data [14]. Techniques like clustering algorithms (K-Means) can be used to find representative points in different clusters or distance-based methods where the selected samples are different from labeled data already present [38].

2.2.4. Challenges and Considerations

Active learning, has certain challenges and limitations. One of the challenges of AL is the so-called ”cold start”. Because the initial-model performance heavily depends on a very small labeled dataset, poor performance in the initial stages is expected [22]. Another challenge is the computational complexity of this method, due to the multiple evaluations required for the querying of the data. The last challenge is the bias in the selection caused by the repeated querying of uncertain samples on unbalanced datasets. This shows why careful selection strategies and efficient sampling methods are needed to balance label costs over the performance of the model [4].

2.3. Backdoor Attacks in Deep Learning

Backdoor attacks can be categorized into data poisoning [20], code poisoning [5], and model poisoning [27]. In this work, we focus on data poisoning, where malicious samples are introduced into the training dataset, which embed hidden triggers. When the model is exposed to the previously mentioned triggers during the inference phase, it performs some unexpected behavior that is intended by the attacker [10, 20]. Backdoor attacks are a great threat in the context of image classification because the attacker can add almost unnoticeable changes to an image to manipulate the output [62]. The primary challenge in detecting these attacks is that they are often hidden and undetectable by standard validation processes [9].

How Backdoor Attacks are Performed

The process of executing a backdoor attack can be divided into different stages. First, a trigger pattern is chosen by the adversary; for example, for images, a small patch or pixel modification may be used. Thereafter, a portion of the training data is modified to contain the mentioned trigger, representing a target class chosen by the attacker. Next, such a poisoned dataset is used to train the DL model. Because the fraction of poisoned data is small, the model achieves high accuracy on clean data while including the backdoor. Finally, once the model is deployed, the adversary can activate the backdoor by inserting the trigger into any input, thus enforcing misclassification [36, 10, 20]. Figure 2.4 shows a visual representation of the steps taken to embed a backdoor attack.

Backdoor Attacks in Computer Vision

Backdoor attacks in computer vision form a serious threat due to the complexity of image classification models and the high-dimensional aspect of images [33]. Different techniques and tactics for such attacks are possible, including altering pixels of images or somehow manipulating learned representations of features. One common form of a backdoor attack consists of embedding patch-based triggers, in which a small region of a given image is altered [20, 34]. Such triggers can be made deceptively subtle yet effective in causing a classification error. Another method is the manipulation of feature space,

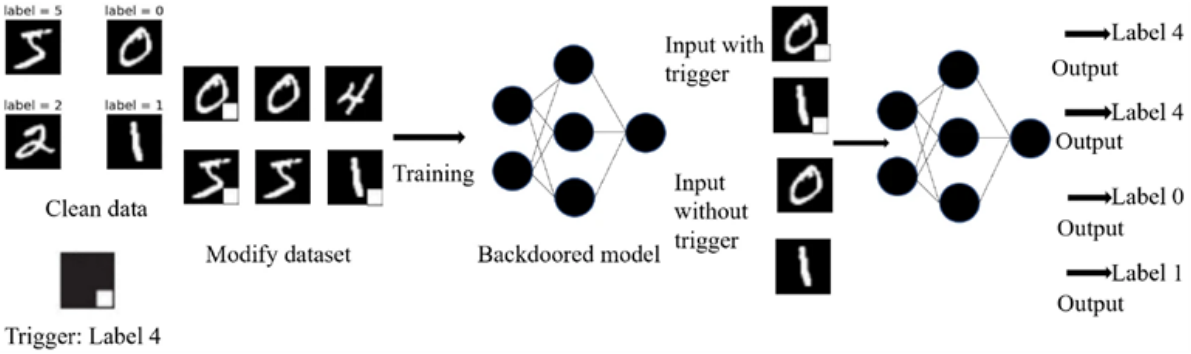

Figure 2.4: Visual representation of a patch-based backdoor injection process on the MNIST dataset. Adapted from [33]

wherein the attacker chooses to modify high-level image attributes like textures and edges without applying pixel-wise manipulation [70]. This technique makes the backdoor even harder to detect while still being highly effective.

Backdoor attacks have been shown to be effective in real world situations. For instance, these attacks can affect facial recognition systems, enabling unauthorized access through manipulated images [37, 45]. Another equally severe risk is posed for autonomous driving vehicles, where injected backdoor triggers can cause misclassifications of road signs and other important signals [74]. Studies indicate that models trained on just 1-5% of poisoned data almost always exhibit backdoor success rates near 100% while maintaining a relatively high clean accuracy [10]. The effectiveness of these backdoor attacks justifies an urgent need for strategic counteractions, protecting the deep learning models against adversarial manipulation.

2.3.1. Advanced Types of Backdoor Attacks in Computer Vision

Backdoor attack techniques have evolved, following new stealthy and robust approaches beyond traditional patch-based methods, that are now standards. Two of the most advanced and efficient types of backdoor attacks in computer vision are the Learnable, Imperceptible, and Robust Backdoor Attack (LIRA) [13], and Warping-Based Backdoor Attack (WaNet) [41]. These methods introduce sophisticated triggers that evade detection, withstand known defenses, and remain effective with little poisoning. The two aforementioned attack types are studied in this research because of their applicability to incremental learning models such as AL pipelines.

Learnable, Imperceptible, and Robust Backdoor Attack (LIRA)

LIRA introduces an elaborate form of invisible backdoor attack in which the trigger does not comprise a fixed pattern, but it is rather learned and embedded within the model's latent space. LIRA creates small alterations in images that are imperceptible and almost impossible to detect in manual inspection. LIRA has demonstrated a near 100% attack success rate with less than 1% poisoned data [13]. The attack remains robust against common backdoor defenses (Neural Cleanse, STRIP), making it one of the stealthiest attack methods.

For the trigger generation, LIRA maximizes a perturbation mask δ on input images in a bounded perturbation budget ϵ, guaranteeing imperceptibility:

$$\min_{\delta} \mathcal{L}_{\text{target}}(f(x+\delta), y_{\text{target}}) + \lambda \|\delta\|_p \tag{2.4}$$

where $\mathcal{L}_{\text{target}}$ forces misclassification to target label y_{target}, and imperceptibility through a perturbation budget ϵ, which limits the alteration of each pixel (usually employing an L_∞-norm). The term $\|\delta\|_p$ is employed to constrain the perturbation size, such that alterations are kept covert. The trade-off between attack strength and imperceptibility is governed by λ , the regularization parameter [13].

Warping-Based Backdoor Attack (WaNet)

WaNet introduces its backdoors by subtly adding spatial distortions (e.g., slight pixel displacement or warping) to image content instead of embedding patterns visible to the eye. In this form of attack, the effectiveness comes from the fact that it modifies the geometry of the image rather than the pixel values, thus preventing the common defense mechanisms from detecting it. Experimental results show that WaNet attacks obtain a success rate of 80% to 95% while remaining highly resistant to standard backdoor defense techniques [41].

In order to produce the trigger, WaNet uses a learnable warping function $W(x;\theta)$ that displaces pixel coordinates of images by a distortion map θ:

$$\min_{\theta} \mathcal{L}_{\text{target}}(f(W(x;\theta)), y_{\text{target}}) + \beta \cdot \text{Distortion}(\theta) \tag{2.5}$$

Within this setup, $\mathcal{L}_{\text{target}}$ forces the model to classify distorted images as y_{target}, while Distortion(θ) is used to control the warping transformation to maintain imperceptibility. The displacement map θ specifies how far pixel positions are moved within the image. The magnitude of this displacement is limited by the attack hyperparameter δ_{max}, thus ensuring that the warping effect is inconspicuous. The smoothness of the transformation is controlled by the attack's hyperparameter s, avoiding abrupt or unrealistic modifications. Moreover, the spatial extension of the warping effect is controlled by the kernel size k, another hyperparameter that controls the interaction among neighboring pixels. Lastly, the regularization parameter β balances the effectiveness of the attack with the preservation of a visually coherent transformation [41].

2.3.2. Clean vs. Dirty Label Backdoor Attacks

Backdoor attacks can be categorized into dirty and clean label attacks. In the dirty label approach, adversaries introduce corrupted instances into the training dataset together with a modification to the original labels of the data. This causes the model to link a particular activation pattern with a target label selected by the adversary. The existence of a trigger can then cause the model to misclassify to the selected target label during the inference phase [39]. In contrast, clean label attacks poison the training data and do not change the initial labels. Opponents modify the inputs and introduce a backdoor trigger, while keeping the labels in accordance with the true class of the input [58, 67, 75]. This approach makes it difficult for human annotators to identify clean labeling attacks, as the poisoned samples closely resemble the genuine ones.

Within the AL framework, clean label attacks represent a considerable risk. Active learning approaches prioritize the selection of informative and representative samples for labeling, with the aim of efficiently optimizing model performance. They are likely to trick the human annotators in the AL process, as clean-labeled poisoned samples are almost indistinguishable from real data. This choice allows the backdoor to spread within the model without raising doubts, thus putting the integrity of the model at risk in a way that is difficult to identify.

2.3.3. Countermeasures Against Backdoor Attacks

Backdoor attack detection and mitigation in DL is a difficult problem for AI system security, especially in environments with ongoing model updates, such as AL. Backdoors may be concealed during training and triggered with the introduction of a special trigger, which makes conventional validation procedures ineffective for detection. The increasing complexity of such attacks requires an integrated solution that combines detection and prevention strategies for defense against adversarial attacks. This subsection addresses the typical techniques used for backdoor attack detection and prevention.

Detection Techniques

One of the fundamental methods for preventing backdoor attacks is the detection of backdoor triggers or poisoned samples prior to or at the moment of model inference. Data filtering methods, including spectral analysis and outlier detection, can be applied for detecting and eliminating suspicious data points before they are utilized in the training procedure [57]. These methods examine the statistical characteristics of the dataset for uncovering anomalies resulting from poisoned samples. Nevertheless,

filtering approaches may be ineffective at identifying advanced clean-label attacks where poisoned samples are very close to real data. Another approach to detecting backdoor attacks is to monitor model behavior at the inference time. Methods such as activation clustering [9] cluster clean and poisoned input activation responses, thus identifying systematic differences caused by backdoor triggers. Through highlighting inconsistencies, methods can be applied to detect adversarial trained models before they are deployed.

Prevention Techniques

Prevention strategies attack models by making them more inherently resistant to backdoor attacks through modifications to training processes and the introduction of defensive strategies. One effective prevention technique is the fine-tuning of models on a new clean dataset to reduce the impact of the backdoors. In reality, exposing the model to additional clean data results in less dependence on poisoned samples, and, in many cases, it entirely removes the backdoor effect [73]. Another technique is to retrain the model under differential privacy so that the model will be less dependent on any single data point, thus decreasing the likelihood of an attacker using it to implant backdoors [43]. Furthermore, adversarial training also entails the addition of the dataset with adversarial manipulated instances to make the model more robust against manipulated inputs. In the paradigm of backdoor defenses, adversarial training can be supplemented with the employment of backdoor-activated samples, which compels the model to learn a more generalized decision boundary [18]. Another method, named defensive distillation, is the practice of training an auxiliary model to learn a smoothed variant of the decision space of the main model, which makes it more difficult for backdoors to survive [68].

3

Related Work

This chapter provides more insights into current research on backdoor attacks, active learning and the intersection of both topics. The chapter starts with an introduction to the approaches to active learning and applications, continuing with backdoor attacks in deep learning in terms of effective methods and their impact. The chapter provides a general introduction to current research on backdoor attacks on active learning and brings to light weaknesses and variety in techniques.

3.1. Deep Active Learning

Active Learning (AL) has been widely applied in the field of deep learning (DL) as a means of overcoming the huge costs of data annotation while maintaining the effectiveness of the model. The technique is of special importance in fields like medical imaging, where labeling large datasets requires expertise in specialized knowledge [63]. Likewise, in the field of autonomous driving, AL is useful by allowing the effective labeling of rare but critical traffic scenarios [24]. Additionally, AL is used in natural language processing and speech recognition to minimize the necessity of manual annotation while maintaining linguistic diversity in the datasets [49, 23]. In recent years, AL has branched out into areas like object detection and segmentation, thus showing its adaptability in various machine learning applications [11].

One of the basic methods in deep AL is Bayesian Deep Active Learning, which employs Bayesian neural networks with dropout methods to enable uncertainty estimation. This method demonstrated improvements in high-dimensional domains like medical imaging [17]. Furthermore, the approach described in [47], proposes AL as a selection problem, where a model trained on a carefully picked subset is theoretically proven to perform nearly as well as one trained on the whole dataset. Another key contribution was the creation of the learning loss framework for AL, which included a loss module that was able to predict the expected loss for every unlabeled sample. This helped with effective sampling irrespective of specific metrics such as uncertainty [69].

Later developments included adversarial and semi-supervised learning methods incorporated in the AL framework. A good example is the variational adversarial active learning method, which utilized an autoencoder alongside an adversarial discriminator to separate labeled and unlabeled samples and improve representation learning and optimize sample selection efficiency [50]. Another interesting development is the batch-mode active learning method presented in [4], which utilizes gradient embedding to simultaneously select uncertain and diverse samples. Subsequently, eliminating the need for hyperparameters and optimizing batch query strategies for different DL models and datasets.

Earlier deep AL techniques like Bayesian deep active learning [17] and Core-set selection [47] struggled with uncertainty querying methods as well as batch selection. More contemporary techniques (adversarial learning, meta-learning loss, gradient embeddings) have offered greater effectiveness and scalability. Furthermore, researchers are increasingly testing on realistic and large-scale labeled datasets to prove that AL decreases labeling costs without sacrificing accuracy [69, 50]. In this work, traditional AL is used together with uncertainty sampling as the querying strategy. While more advanced methods like adversarial learning or gradient embeddings offer great advancements, they lead to increased

computational costs and complexity. Thus, traditional AL with uncertainty sampling is used to maintain performance with limited computational resources.

3.2. Backdoor Attacks in Deep Learning

Backdoor attacks are serious threats that allow the embedding of malicious functionality within different DL models. These types of attacks are especially dangerous in areas such as autonomous vehicle operation, where malicious inputs might cause catastrophic misclassifications [74]. Another example is the case of facial recognition systems for security, where facial analysis might be targeted to allow unauthorized permissions [37, 45]. Furthermore, the presence of backdoor vulnerabilities within medical AI systems creates safety menaces that can compromise patients' well-being [16].

Several works have studied and recognized the potential dangers associated with backdoor attacks. BadNets, one of the earliest examples, demonstrates the idea of using simple visual patterns to act as trigger signals that induce misclassification while retaining good performance for non-manipulated inputs [20]. More recently, backdoor attacks on federated learning have shown the possibility of poisoning collaborative models in an effective way [6]. An additional example is supply chain attacks, where backdoors are introduced into DL models using compromised training datasets, often found in open source repositories [65]. These have been recognized as a considerable risk that could also occur in real life scenarios.

Recent developments in backdoor attack techniques have led to increasingly complex and difficult-to-detect scenarios. For instance, clean label backdoor attacks successfully overcame the main drawback of previous approaches by labeling the poisoned samples correctly, thus ensuring that poisoned data went undetected during human evaluations [59, 67, 58]. Another difficult to detect method is the LIRA attack, which strategically alters representations of the latent space to infiltrate triggers into images [13]. WaNet is another sophisticated technique that uses unique spatial modifications. This attack integrates triggers into visual data to create patterns that blend in perfectly with the original information [41]. Both techniques have demonstrated effectiveness in bypassing many widely used defense mechanisms that depend on identifying inconsistencies in the data.

3.3. Backdoor Attacks on Active Learning

Even though backdoor attacks have been extensively researched in typical DL settings, much remains unknown about their implications in AL. The unique selection mechanism in AL, through which models choose the most uncertain or informative instances, offers adversaries new directions to corrupt the training pipeline.

Recent studies have begun to investigate backdoor threats in AL. For instance, the Double-Cross Attack [60], demonstrates that AL-based selection processes had the potential to improve poisoning effectiveness using a small poison rate. This attack relied on injecting adversarial samples that purposely attracted AL queries, therefore boosting the effect of the backdoor with minimum interference. As opposed to traditional poisoning schemes that indiscriminately inject backdoor triggers within the dataset, this approach ensures that the AL process itself selects and amplifies the attack. Experiments also found that typical anomaly detection models failed to prune poisoned samples, showing the need for more robust AL protections against these attacks [60].

While existing work has explored backdoor attacks on AL [60], our approach introduces key differences and improvements. In [60] a generator is trained to produce the trigger, this is computationally expensive and uses massive model queries (up to 520K inputs queried). Our attack does not produce as much overhead, making it more applicable in resource-limited situations. Moreover, we do not rely on a pre-specified loss term for stealthiness, such as the L2 norm of trigger pixel values used in their approach, so we can adopt a more adaptive and flexible attack plan. In contrast to their static trigger strategy, where a single trigger is applied to all poisoned samples, we investigate the effect of dynamic triggers with WaNet and LIRA and study how these vary across AL rounds. Also, their attack is restricted to clean-label poisoning, whereas our approach takes more general poisoning into account.

Another major difference is in the scope of poisoning. Their attack employs a pre-query poisoning strategy, where poisoned samples are distributed across the entire available data pool before AL queries.

Our approach employs a post-query poisoning strategy, where only a fraction of the previously queried samples are poisoned. Experimentally, their work focuses on low poisoning rates (as little as 0.1% on ImageNet and as high as 4.2% on SVHN), we examine similar and higher poisoning rates in order to set upper bounds on an attacker's abilities. While they are reporting poisoning rates of 0.7%, 0.4%, and 0.2% on CIFAR-10, our effective poisoning rates for CIFAR-10 would be 1% and 5%.

Besides, their approach employs stream-based AL with margin sampling as the query strategy, while we explore batch-based AL and the uncertainty querying method. In their gray-box scenario, they assume the attacker knows the sampling mechanism, whereas the attacker is not aware of any such mechanism in the black-box scenario. Finally, they remove the original clean sample when poisoning a sample, which we also do in our implementation, but our querying mechanism is different. They initially poison all images of the target class to test their electability, and we choose poisoned samples based on uncertainty, certainty, or randomly, thus examining systematically what choice method performs best for poisoning in an AL setting.

To defuse backdoor attacks in AL, new protective mechanisms have been proposed. The work in [35] highlighted how an attacker could strategically insert poisoned samples in decision boundary regions, increasing the chances that the poisoned inputs would be queried and labeled, thereby reinforcing the backdoor. Furthermore, adversarial training is used for the AL pipeline so that less labeled data is required to achieve high test accuracy [35]. The proposed AL approach in [35] manages to boost accuracy to 89% on average, compared to about 50% with a baseline random sampling strategy. Another mitigation technique is Density Based Data Selective Sampling (DRE). This technique uses data density and entropy to maintain low chances of selecting adversary-designed samples. The DRE method improves robustness by up to 24.40% compared to existing methods [21]. Although adversarial training can offer strong protection, it often comes with significant computational overhead, making the DRE-based approach more realistic for many real-world AL scenarios where resources are limited. Present-day defenses remain sparse in how they tend to address particular techniques of attacks rather than providing a complete solution for backdoor AL vulnerabilities.

Our work extends the analysis of backdoor threats in AL for computer vision into multiple configurations (for example, certainty-based sampling, post-query poisoning) and advanced triggers such as LIRA and WaNet. Here, with only 1 percent data poisoning on simple datasets, we demonstrate an ASR of greater than 95 percent on CNNs. Additionally, the cyclic nature of AL is exploited through techniques like sub-trigger division and progressive parameter adjustment. We do not include defenses (DRE or adversarial training) in our analysis, which points to a pertinent future research area. Despite this, our findings confirm that clean accuracy is mostly preserved, which speaks to the stealthiness of these types of backdoor attacks and the need for strong, multi-faceted defenses to secure active learning pipelines.

4

Methodology and Study Design

In this chapter, the experimental setup and methodology used to examine the impact of backdoor attacks in active learning are explained. The chapter begins with the datasets and experimental configurations used. The second section is about the active learning framework design, describing each part of the training process. Afterwards, the threat model considered for this study is presented. Furthermore, the four main experiments of this study are explained in detail. Lastly, the experimental execution and the computational resources for these experiments are reviewed.

4.1. Data Collection and Model Variation

To ensure that the results of this study are more generalizable and complete, three distinct data sets were utilized. Initially, the MNIST dataset was selected for preliminary experimentation and baseline measurements. The gray-scale images in this dataset represent ten classes of handwritten digits (0-9) [32]. Next, the GTSRB dataset was chosen, which contains 43 categories of color images of varying sizes, representing different traffic signs [51]. Lastly, the CIFAR-10 dataset was employed, containing 10 different class images of various objects [31]. Table 4.1 summarizes the most important features of the used datasets.

Table 4.1: Summary of used datasets

Dataset	#Set Size	Image Size
MNIST	60,000	32X32
GTSRB	39,209	15x15 - 250x250
CIFAR-10	50,000	32X32

Furthermore, three types of deep learning models were trained to observe how the different structures influence susceptibility to backdoor attacks. A simple CNN model was used primarily to establish an initial understanding of vulnerabilities in simpler convolutional structures. Moreover, the ResNet [25] and Inception [55] model architectures were used to provide insights into vulnerabilities of more complex models. Table 4.2 encapsulates the most important features of each model.

Table 4.2: Summary of used models. *The number input channels for GTSRB and CIFAR-10 is three, given the RGB channels of the images. ** For MNIST and CIFAR-10 datasets we have 10 output classes, while for GTSRB dataset we consider 43.

Model	Input Channels	Conv Layers	Fully Connected
CNN	1 / 3*	2 (32, 64 filters)	2 (128, 10 / 43)**
ResNet	1 / 3*	ResNet18 Backbone	1 (Adapted FC)
Inception	1 / 3*	InceptionV3 Backbone	1 (Adapted FC)

Table 4.3 provides an overview of trainable parameters per model across different datasets. The complexity of the models varies significantly, with CNNs having the least amount of parameters, hence being less computationally intensive and suitable for small-scale problems, while ResNet and Inception models have significantly more parameters, reflecting their deeper structures and greater representational capacity.

Table 4.3: Number of trainable parameters for each model across different datasets.

Model	MNIST	GTSRB	CIFAR-10
CNN	421,642	549,355	545,098
ResNet	11,175,370	11,190,891	11,173,962
Inception	21,805,482	21,873,675	21,805,994

4.2. Active Learning Pipeline Settings

An active learning (AL) pipeline was developed in order to emulate the iterative training mentioned in Section 2.2. This pipeline allowed the injection of backdoor attacks to examine vulnerabilities or to train the model without poisoning, thus facilitating comparative analysis. A pool-based AL approach was used, where the model was initially trained on a small labeled subset (approx. 2% of the dataset) and then iteratively retrained on new samples. For the query strategy, uncertainty sampling was chosen, which selected the least confident predictions to be classified by human annotators. In this research, the human labeling process was simulated by automatically assigning ground truth labels to the dataset.

In each cycle, the model queried a specific number of data points and was retrained for five epochs. The epochs were chosen to achieve a balance between computational overhead and sufficient gradient updates to facilitate learning advancement without overfitting or long training durations. In the experiment, a portion of the previously queried set was poisoned by inserting a backdoor trigger. The poisoning strategy was based on the specific experiment and determined which of the pre-queried samples was chosen for poisoning. The first poisoning strategy, uncertainty-based poisoning, applied triggers to samples with higher uncertainty scores. Certainty-based poisoning, the second option, focused on data points that the model had ranked with a high level of confidence. The third option, random poisoning, was used as a baseline and arbitrary selected poisoned samples. The last option, clean label poisoning, adopted a similar random selection method, but maintained consistency with natural class labels. Random poisoning was used in the clean label experiments based on previous results that suggested that its performance falls between uncertainty and certainty poisoning, making it a good baseline. After each cycle, the model was evaluated, and this process continued over multiple retraining cycles. The number of cycles used to retrain the model depends on the experiment. Detailed settings are summarized in Table 4.4.

Table 4.4: Initial dataset splits and model performance metrics used. *Initial validation accuracy and validation loss were averaged over five different seeds.

Dataset	Init. set	Pool	Test	Val	Model	Init. acc.	Init. loss
MNIST	1200	47400	5400	6000	CNN	0.881	0.549
					ResNet	0.852	0.744
					Inception	0.760	0.840
GTSRB	785	31024	3927	3534	CNN	0.602	1.441
					ResNet	0.307	2.283
					Inception	0.190	3.657
CIFAR-10	1200	47400	5400	6000	CNN	0.387	1.927
					ResNet	0.228	2.069
					Inception	0.141	3.776

4.2.1. Evaluation Methods

The accuracy of the model and the success rate of the attacks were evaluated to measure the performance of the AL framework on different datasets, models and attack configurations. This study tried to identify the most vulnerable combinations to those backdoor attacks, in order to provide valuable information on potential weaknesses. The evaluation was also conducted over each cycle to track the accuracy and success of attacks during different iterations. The key metric used in this analysis was the Attack Success Rate (ASR), serving as an indication of how vulnerable a model is to a certain attack. In this case, higher ASR values represent more vulnerability to the attack and are calculated as follows:

$$ASR = \left(\frac{Number\ of\ Successful\ Attacks}{Total\ Number\ of\ Attacks} \right) \times 100\% \tag{4.1}$$

This thesis measured the stealthiness of backdoor attacks through the clean accuracy drop (CAD), which is the difference of the model accuracy on not poisoned and correctly labeled test datasets. The basic assumption is that a good and stealthy backdoor attack should have little effect on the model's effectiveness on normal tasks since any appreciable degradation would raise suspicion.

4.2.2. Threat Model

This subsection establishes the threat model considered in this study, focusing particularly on backdoor attacks on active learning under the hypothesis of post-query poisoning. This implies that an attacker injects poisoned samples into the dataset after completion of the querying process. This approach allows for a more direct assessment of how AL trained models react to different poisoning effects. Post-query injection of poisoned samples allows us to isolate the impact of the selection strategy on ASR. Additionally, post-query poisoning is particularly relevant in scenarios where data labeling or collection is outsourced, and hence becomes an even more realistic and stealthier attack channel in AL pipelines.

Attacker's Goal

Imagine a context where an attacker has partial control over the labeled dataset used for training, but has no direct impact on the query strategy. This implies that the choice of data points remains a legitimate AL process. After the querying process, the attacker injects a minimal portion of the poisoned data into the set before using it for training. The first goal of the attacker is to maintain good model performance on clean samples to avoid suspicion, while making sure that without the trigger, the backdoor stays inactive. Another goal is to incorporate the backdoor in a way that bypasses common detection methods.

Capabilities of the Attacker

It is assumed that the attacker has the following capabilities. First, the attacker has the ability to alter a section of the labeled data before it is used for training. In the scenario of a clean label attack, the data can be altered prior to the labeling process because the label does not need to be altered. Secondly, a gray box threat model is considered, where the opponent knows the structure of the model and the training process, but does not have full access to the model parameters. This differs from white-box threat models that allow complete visibility of the model, including parameters and gradients, and black-box settings where nothing about the model structure or training process is known [8]. Lastly, the attacker has the ability to introduce meticulously prepared poison samples into the training dataset after the querying process. In contrast to conventional data poisoning attacks, where attackers manipulate the dataset prior to the query, in this scenario the poisoning only occurs after the query process. This is particularly significant in real-world contexts where AL is applied in areas such as medical imaging and cybersecurity, where data tagging is performed by trusted specialists, but adversaries can compromise the integrity of the tagged dataset at a later stage.

4.2.3. Experiment 1: Evaluating Backdoor Attacks and Model Susceptibility

The goal of this experiment was to evaluate the effectiveness of various backdoor attack strategies and parameter variations in the AL scenario. Precisely, **RQ1** was partly addressed, which discusses how various settings of backdoor attacks influence ASR on AL models. In addition, **RQ2** was explored, which

analyzes how dataset characteristics and model structures affect the vulnerability of active learning frameworks to backdoor attacks.

The standard active learning settings described in Table 4.5 were used for Experiment 1. This follows the standard settings described in Section 4.2. The analyzed poisoning methods were uncertainty-based, certainty-based, and random selection, allowing for a comparison of how different query strategies influence ASR. The poisoning percentages were 1% or 5%, which represent the fraction of the queried set that are poisoned in every cycle. To evaluate how the poisoning process evolves, the experiment was conducted across 5, 10, and 15 cycles. Besides, various types of triggers were also considered, which are small, large, or specialized, with different levels of visibility. The query size, which determines how many samples are chosen for each cycle, was set at 500, 1000, and 1500 samples, allowing different AL settings to test. Finally, to provide reproducibility of the results, each experiment was run with five varying random seeds.

Table 4.5: Parameter settings used for Experiment 1.

Parameter	Values
#Seeds	5
Poison Rate	1%, 5%
Cycle Count	5, 10, 15
Query Size	500, 1000, 1500
Trigger Type	small, big, specialized
Poison Method	uncertainty, certainty, random

The first goal of this experiment was to examine how different backdoor configurations influence the effectiveness of the attack, including trigger type, injection rate, poisoning method and query size. Three different trigger designs were used. Figure 4.1 graphically shows the used trigger types for this experiment. These three trigger designs, small, large, and specialized (checkerboard pattern), were chosen to evaluate their impact on ASR in active learning. Small and large triggers evaluate the impact of trigger size on ASR by employing less-visible small triggers and potentially increased attack effectiveness with larger triggers. The checkerboard pattern evaluates if structured patterns influence ASR differently by affecting the feature extraction of the model.

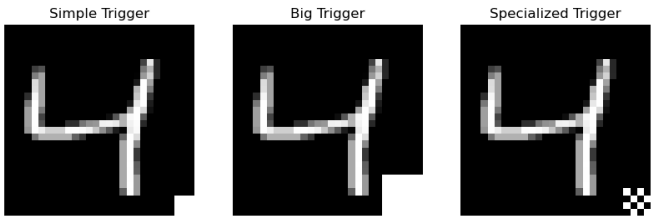

Figure 4.1: The simple trigger is a 3×3 white square placed in the bottom-right of the image. The big trigger increases this to a 6×6 white square, with higher visibility. The specialized trigger contains a 4×4 checkerboard pattern for more complexity.

The second goal of this experiment was to investigate how dataset characteristics and model structures affect the vulnerability of AL frameworks to backdoor attacks. Subsection 4.1 details the model and dataset combinations that were employed in this experiment. It is important to note that the attacks performed in this experiment were dirty label attacks. Initially, early stopping was used, but was later changed to a fixed number of active learning cycles to ensure consistency across all executions. In addition, batch processing was implemented to increase computational efficiency, especially on larger

datasets. To reduce bias and improve reproducibility, all experiments were performed with five different random seeds.

4.2.4. Experiment 2: Sub-Triggers and Progressive Parameter Adjustments

The objective of **RQ3** was to analyze tailored backdoor attack designs within the context of AL. This experiment evaluated progressive parameter adjustment and trigger subdivision, along with the use of advanced types of backdoor triggers, such as LIRA and WaNet. First, preliminary tests were conducted using pixel-based methods to assess feasibility. Subsequently, more detailed experiments were carried out using LIRA and WaNet. Note that different datasets, models, active learning settings, and backdoor parameters were used. Following **RQ1** and **RQ2**.

Table 4.6 presents the parameter settings used in Experiment 2. The complete description of the standard parameters can be found in Section 4.2. The random poisoning method was employed because it demonstrated intermediate performance between certainty and uncertainty poisoning in Experiment 1, serving as a good baseline. As for the poisoning rate, the same predefined rates of either 1% or 5% were used. Each cycle queries 1000 or 1500 samples and was repeated over 15 cycles to assess the persistence of the attack. Note that taking 15 cycles allowed comparisons to 5 and 10 cycles by taking intermediate results. To ensure generalizability, the experiments were run using five different random seeds.

Various trigger types were explored, including sub-trigger LIRA, sub-trigger WaNet, Global LIRA, Global WaNet, progressive LIRA, and progressive WaNet, each with distinct characteristics. The sub-trigger and progressive parameter adjustment methods will be explained later in this Section. The parameter related to the sub-trigger method is the sub-size of either 5 or 8 pixels. The sub-size represents the area that is affected by each sub-trigger. The perturbation strength (s) in WaNet attacks is set to 0.2 or 0.5, while the grid size (k), takes values of 4 or 8. For LIRA-based attacks, the perturbation bound (ϵ), is set to 0.1 or 0.2. The explanation of the inherent parameters of WaNet and LIRA are described in Section 2.3.1. Since global and progressive triggers modify the complete image rather than a small subset, they do not require a sub-size parameter.

Table 4.6: Parameter settings used for Experiment 2.

Parameter	**Values**
#Seeds	5
Cycle Count	15
Poison Rate	1%, 5%
Poison Method	random
Query Size	1000, 1500

Trigger Type	**sub size**	**s**	**k**	ϵ
Sub-trigger LIRA	5, 8	-	-	0.1, 0.2
Sub-trigger WaNet	5, 8	0.2, 0.5	4, 8	-
Global LIRA	-	-	-	0.2
Global WaNet	-	0.5	8	-
Progressive LIRA	-	-	-	0.2
Progressive WaNet	-	0.5	8	-

Progressive Parameter Adjustment

Instead of introducing triggers with constant intensity from the start, the progressive parameter adjustment strategy consists of progressively boosting the trigger strength over the active learning cycles. This method ensured that the backdoor modification evolved over time, given that the intensity of the trigger was computed based on the cycle. An example of the progressive parameter adjustment in WaNet-based poisoning is shown in Figure 4.2.

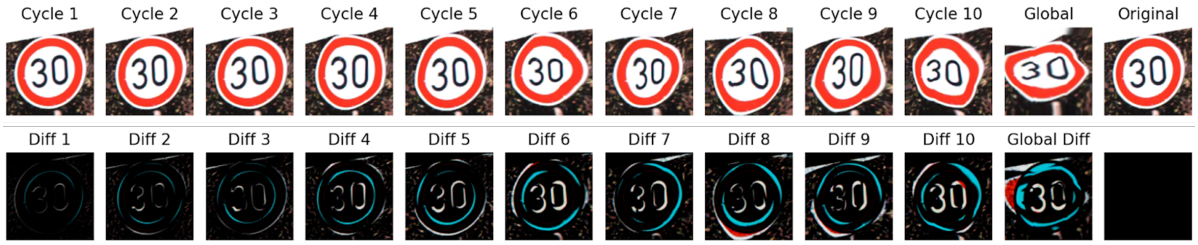

Figure 4.2: WaNet-based progressive parameter adjustment is visualized with a control grid size of k=4 and a scaling factor of s=0.2. The top row shows how the backdoor pattern is gradually embedded as the geometric distortion gets more intense over cycles. The original image and the cumulative global warping effect (applying the maximum transformation at each pixel location across all cycles) are depicted in the final two photographs.

To gradually increase the intensity of the backdoor patterns without losing stealth, key parameters were gradually adjusted during the AL cycles. The LIRA- and WaNet-based settings follow a similar principle, where the intensity of the applied modification grows linearly with the cycle index t, reaching its maximum at the final cycle T.

$$\theta_t = \theta_{\max} \cdot \frac{t}{T} \tag{4.2}$$

Where θ_t represents the progressively adjusted parameter at cycle t. For LIRA-based adjustment, θ_t corresponds to the perturbation intensity ϵ_t, with $\theta_{\max} = \epsilon_{\max}$. For WaNet-based adjustment, θ_t corresponds to the warping intensity s_t, with $\theta_{\max} = s_{\max}$. T is the total number of active learning cycles.

In the LIRA-based adjustment, the parameter ϵ_t controlled the magnitude of additive noise applied to the image. In the WaNet-based adjustment, the parameter s_t regulated the strength of spatial transformations, causing progressively more pronounced geometric distortions. By gradually modifying these settings for every round, the backdoor effect is initially concealed but becomes more powerful over the cycles. A key aspect of this strategy was that although each cycle only introduced a small increase in poisoning intensity, the model was evaluated on cumulative poisoning, meaning that the test set contained images poisoned with the full summed-up intensity of all previous cycles.

Sub-Trigger Division

Unlike standard backdoor triggers, which use a single identifiable modification on the whole image. Sub-trigger division fragments a trigger into multiple smaller parts, each injected into different poisoned samples. This approach assessed whether models could recognize and activate a backdoor with partial triggers, and how the backdoor effect changed when all sub-triggers were present in an image. Figure 4.3 shows the principle of sub-triggering in LIRA-based attacks.

During training, each cycle included one individual sub-trigger in the image, allowing the model to gradually learn the poisoned features. However, during inference time and testing, the model was evaluated with images showing all sub-triggers accumulated over the cycles. This allowed us to determine whether the models needed the full trigger pattern to activate the backdoor, or whether smaller patterns could still cause misclassification.

Comparison to Global Triggers

To contrast with these progressive approaches, global triggers were also tested, where a complete LIRA or WaNet trigger was applied from the start of the attack. As opposed to progressive poisoning, where the strength of the trigger varied with time, global poisoning ensured a consistent backdoor pattern during training. This provided a baseline to compare against, allowing to determine if sub-triggers and gradual poisoning were more or less effective than a consistent full-strength trigger.

Figure 4.3: Visualisation of the LIRA-based subtrigger division with ϵ=0.5 and an 8×8 subtrigger pattern. The top row illustrates the progressive application of the backdoor subtriggers over 10 active learning cycles, where the two final images reflect the overall cumulative effect together with the original image.

4.2.5. Experiment 3: Investigating the Effects of Clean-Label Attacks

The goal of experiment 3 was to extend the analysis of **RQ1** an **RQ2** by focusing on clean-label poisoning methods, which are especially relevant in AL contexts. This experiment conducted trials with LIRA and WaNet backdoor triggers, using global and sub-trigger splitting methods. Unlike dirty label attacks, clean label poisoning maintains the correct class labels, which makes identification more difficult and improves threat representation in real-world situations. In contrast to previous experiments, where inconsistencies in labels could indicate possible poisoning, clean label poisoning ensured that poisoned samples were still correctly labeled, making it difficult for human annotators to identify them.

Table 4.7 summarizes the key parameters for Experiment 3. This experiment investigates clean-label backdoor attacks and introduces the target-only poison method in comparison to the poison methods described in Section 4.2. Target only refers to poisoning samples that belong to the target class. The poisoning rate is 2% and 5%, which refers to the percentage of queried samples modified each cycle. Query size 1500 was used to have enough queried samples per cycle for more stable evaluation of poisoning efficacy. The experiment was conducted for 15 cycles to capture long-term poisoning effects, and global LIRA and WaNet triggers were used for maximizing the backdoor effect by altering the whole image instead of localized areas.

Table 4.7: Parameter settings used for Experiment 3.

Parameter	Values
#Seeds	5
Cycle Count	15
Query Size	1500
Poison Rate	2%, 5%
Poison Method	target-only
Trigger Type	sub-trigger/global LIRA-WaNet

In this experiment, only the images of the target attack class were poisoned during training, while the other classes remained unchanged. In this way, the model learned to link only the backdoor trigger to the target class. To evaluate the effectiveness of the attack, images of the non-target classes were tested using the same trigger. If the attack was successful, these images were misclassified as the target class. Figure 4.4 illustrates this process.

Figure 4.4: Example of a clean label attack. The first column presents the original image with the corresponding label. In the second column, a trigger is applied during training, while keeping the original labels. The third column shows how in the testing phase the triggered image causes model misclassifications.

4.2.6. Experiment 4: Pre-query vs Post-query Poisoning

The fourth experiment in this thesis aligns with **RQ4**, **RQ1** and **RQ2**. This experiment compared the effects of using pre-query poisoning in contrast to post-query poisoning. The study conducted in [60] uses the pre-query method, while all previous experiments in this study employ a post-query method. This analysis was done to understand how the poisoning strategy affects the success rate of backdoor attacks. As explained in Section 4.2, active learning selects a subset of available data based on an uncertainty criteria. Thus, the timing of the poisoning could significantly affect the model's learning and its vulnerability to backdoor attacks.

Pre-query poisoning assumes that the pool of data available for selection is already corrupted. This means that when the model queries a sample from the data pool to be labeled, the uncertainty criterion is also computed for the poisoned samples. As a result, it is not guaranteed that all poisoned samples will be queried and used for retraining. Figure 4.5 illustrates the pre-query method.

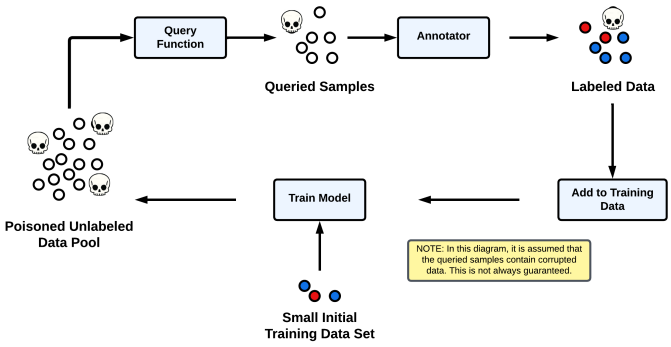

Figure 4.5: Diagram showing theprequery poisoning strategy.

In contrast, the post-query poisoning strategy applies the trigger to the already queried samples, meaning that the model will always be retrained with the poisoned data. Note that the post-query strategy assumes a scenario in which the adversary has control over the queried set. Figure 4.6 shows the post-query method.

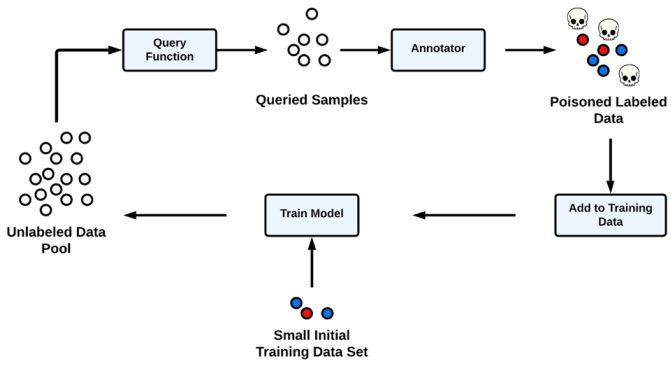

Figure 4.6: Diagram showing the postquery poisoning strategy

To evaluate the poisoning strategies in a comparable way, two identical active learning models were trained in parallel. Specifically, the CNN model architecture was used. One model used the pre-query poisoning strategy and the other post-query poisoning. Table 4.8 presents the parameter settings used for Experiment 4. The random poisoning method was used as a baseline at a fixed rate of 1% or 5%, for consistency across pre-query and post-query scenarios. Note that for the pre-query poisoning the rate was computed over the whole data pool, whereas for the post-query poisoning it was computed over the queried set. To ensure reproducibility, five different random seeds were used in the training runs. Additionally, this experiment was done using the LIRA and WaNet triggers over 15 training cycles, with each cycle querying 1500 new samples.

Table 4.8: Parameter settings used for Experiment 4.

Parameter	Values
Cycle Count	15
Query Size	1500
Poison Rate	1%, 5%
Poison Method	random
Trigger Type	global LIRA-WaNet

4.2.7. Computational Resources and Experimental Execution

As mentioned before, this thesis consisted of four different experiments. Because of the different parameter settings, poisoning strategies, datasets, and models, high scale computational resources were needed to run these experiments. For this purpose, the DAIC cluster at TU Delft was used, which ensured efficient, parallel computing and replicability of the process. Using the DAIC cluster significantly reduced the runtime of the process, improved the tracking of the processed jobs, and enabled error correction.

Experiments were structured into job arrays, where each job represented a single execution of an active learning pipeline with a specific parameter and dataset-model configuration. Within a job, the model was retrained for an amount of cycles, the total number of AL retraining cycles resulted in **112185 runs**. Each job was submitted via the SLURM workload manager, which allows batch execution of experiments and automated resource allocation. The average runtime per job was approximately **6 hours**, with total experimental completion spanning **44874 hours**. CUDA-enabled GPUs were used to accelerate the deep learning computations, ensuring the on time execution of the experiments. Lastly, a CUDA environment was configured using Apptainer to ensure consistency between local development and cluster executions.

The main programming language used for the development of the pipeline, the set-up of the experiments, the data management, and the overall analysis was Python (version 3.12). Furthermore, PyTorch (version 2.4) was used to train the deep learning models and to perform the backdoor attacks. The entire code for this project is available at: https://github.com/SelenaMendez2801/master_thesis

Table 4.9 provides an overview of the parameter settings, the submission of jobs, and the total model retraining for each experiment. The column parameter combinations present the total variations of experimental setups. The column total jobs presents the variations of these parameter combinations over the various datasets and models. The column total retraining provides the number of times a model was trained and updated over all AL cycles in the experiment. The total number of all distinct models learned across all data and experiments were **7479 models**. It indicates the extensive computational burden used in this study.

Table 4.9: Overview of parameter combinations, job submissions and total model retraining per experiment. Not including reruns of experiments or failed jobs.

Experiment	Parameter Comb.	Total Jobs	Total Retraining
Experiment 1	285	2565	38475
Experiment 2	340	3060	45900
Experiment 3	190	1710	25650
Experiment 4	48	144	2160
Total	863	7479	112185

5

Experimental Results

This chapter presents the outcomes that were obtained for the four experiments that were conducted. The main evaluation measure used for this study was the attack success rate across various models, datasets, and poisoning techniques. Each section of this chapter explains the result of its corresponding experiment, with the main findings consistent with the aim of the experiments.

5.1. Results of Experiment 1

This section presents the results of Experiment 1. This experiment had two different purposes. The first purpose was to examine the influence of different parameter settings such as trigger type, poisoning strategy, query size, and poisoning rate on the vulnerability of the model that was trained under active learning. The second goal was to evaluate how different datasets and models affect the success of backdoor attacks in the active learning process. The detailed tabulated results of the attack success rates (ASR) for this experiment are provided in Appendix A, Tables A.1–A.3.

ASR Evolution Across Active Learning Cycles

Based on the results, Figure 5.1 presents the ASR trends across multiple active learning cycles for different models, datasets, poisoning methods, and trigger types. The results show clear differences in attack effectiveness depending on the unique combinations.

As expected, for most combinations, the ASR increased as the number of active learning cycles progressed. However, the pace of such an increase differs from dataset to dataset and even among models. For the datasets, MNIST was the fastest in ASR progress in just a few iterations. For example, CNN on MNIST with uncertainty-based poisoning reached an ASR of 100% by cycle 4, while CNN on CIFAR-10 stayed around 20–30% ASR even after 15 cycles. This can be because of relatively lower complexity and more homogeneous images in MNIST, such that backdoor triggers can evolve into a dominant feature more easily. In contrast, CIFAR-10 was quite robust to attacks even after a number of iterations. Most likely due to its higher complexity and higher diversity of information, texture of objects, and colors that can make it more difficult to connect to a particular class. Furthermore, GTSRB had average ASR growth compared to the other two datasets, perhaps due to it being composed of structured yet varied traffic sign images, where certain signs might have some similarities that make backdoor patterns less immediately dominant but still influential over time.

As for model variations, the findings highlight that CNN models were consistently more prone to backdoor attacks than ResNet models, quickly attaining higher values of ASR earlier. This can be explained by the simpler architecture of CNNs, which has a tendency to make them more vulnerable to learning incorrect patterns such as backdoor triggers. On both MNIST and GTSRB, CNNs showed rapid ASR progression, especially with certainty and random-based poisoning. However, on CIFAR-10, CNNs showed much lower ASR overall, suggesting that the increased image complexity, texture variation, and color diversity in CIFAR-10 reduce the model's capacity to learn and apply the backdoor trigger in a consistent way. This shows a feature in CNNs when dealing with more complex visual features, which may inherently reduce the effectiveness of the trigger.

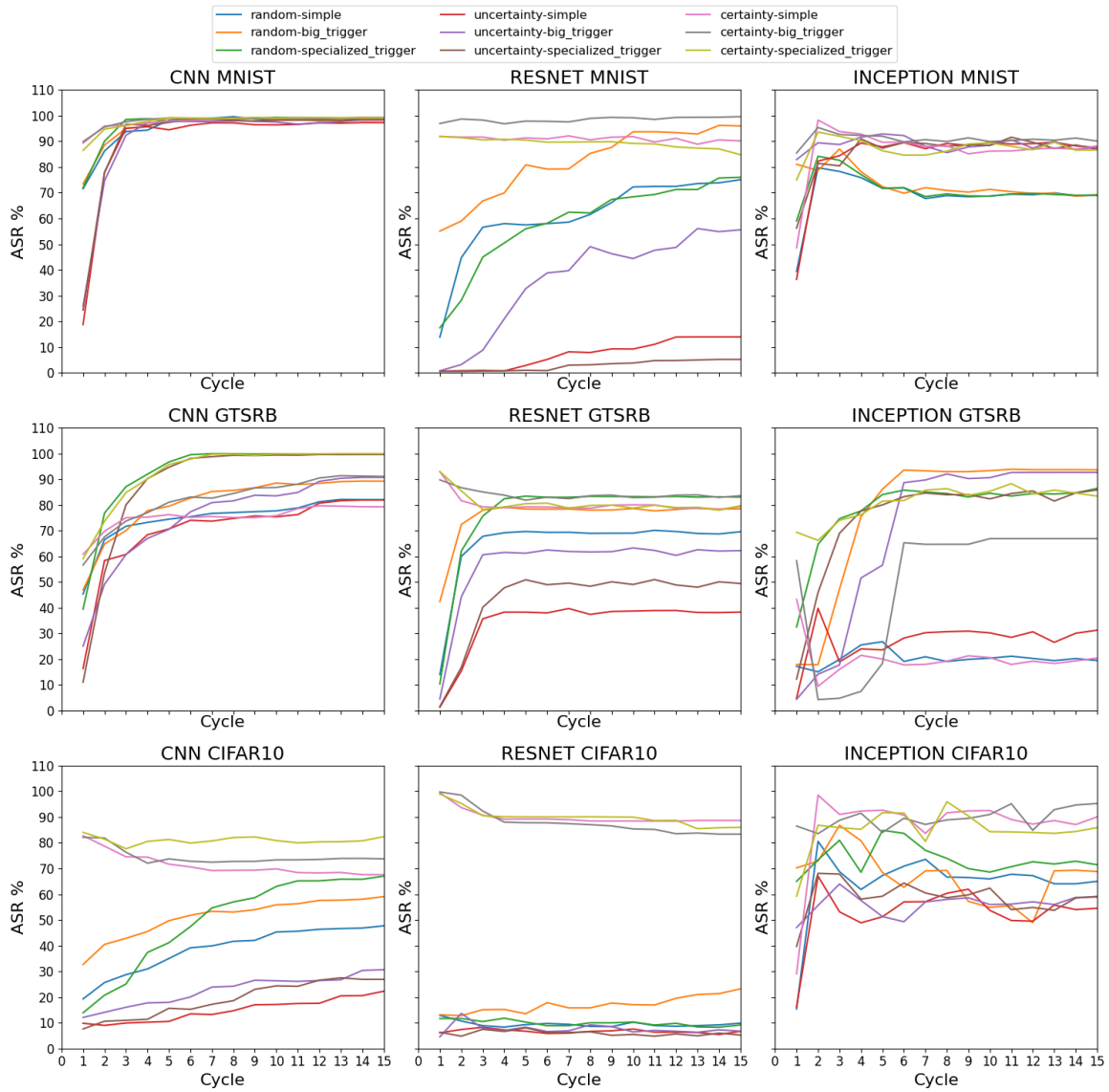

Figure 5.1: ASR progression over active learning cycles for different models and datasets. Furthermore, each line represent an unique combination of poison method and trigger type

For ResNet models, ASR improvement was generally lower and slower for all datasets, with greater robustness to backdoor poisoning. For instance, ResNet on CIFAR-10 under uncertainty poisoning reached a final ASR of less than ≈10%, while certainty poisoning improved this to ≈90%. The architectural depth and use of residual connections likely help prevent overfitting to the poisoned data, especially in the early learning stages. However, performance on CIFAR-10 was particularly poor, with most poisoning methods leading to very low ASR throughout the cycles. In particular, only the certainty-based poisoning method achieved high ASR values, validating the assertion that samples confidently selected as poisoned have higher probability of impacting a stable model like ResNet. This highlights the manner in which both data complexity and the model architecture efficiently disrupt backdoor attacks from being successful in active learning.

For the Inception model, ASR trends differed notably across datasets. On MNIST, ASR was high across most methods, but random-based poisoning consistently had the lowest ASR, suggesting that the type of selection limits the effectiveness of the attack, even on a simple dataset. For GTSRB, ASR increased steadily, but the simple trigger type consistently led to the lowest ASR, indicating that more complex or specialized triggers are more effective on structured yet diverse images like traffic signs.

Bigger triggers again stood out, confirming its advantage in targeting impactful samples. On CIFAR-10, ASR was more spread out, yet the trend among the other models was again replicated: poisoning by uncertainty worked most poorly, then random, then certainty. Counterintuitively, compared to CNN and ResNet, Inception actually returned generally higher ASR on CIFAR-10, showing that its multiscale feature extraction remains capable of even better triggering and spreading detection with complicated data sets.

The parameter choices, such as poisoning strategy and trigger type, showed impact on the ASR trends. In terms of the poisoning strategy, certainty poisoning was apparently the most effective, achieving high ASR with fewer cycles for most of the combinations. Random based poisoning followed with varying ASR ratings that progressed more slowly than the certainty based method. As last comes uncertainty poisoning, which showed to be the least effective overall. Furthermore, the chosen trigger type also showed recurring patterns in terms of ASR.

ASR Dependence on the Number of Poisoned Samples

Figure 5.2 presents the relationship between number of poisoned samples and the final ASR (at cycle 15) for different models. It shows how poisoned samples can highly increase the ASR, especially in simpler datasets like MNIST.

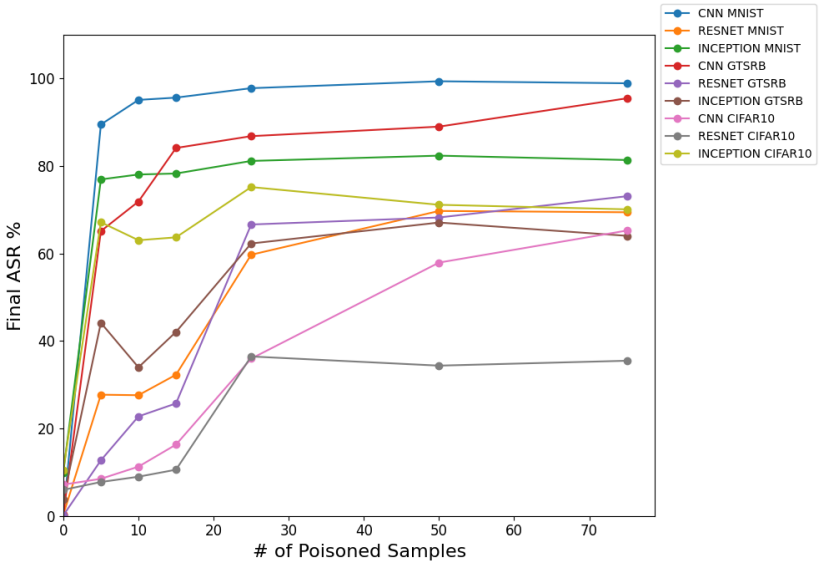

Figure 5.2: Final ASR as a function of the number of poisoned samples for different models and datasets.

The figure provides several important insights. First, attacks on MNIST worked very well with low poisoning rates, a high ASR may be obtained with just a few poisoned samples. This is likely because the images of MNIST are simple, representing grayscale digits that have minimal intra-class variability. As a result, the model primarily picks up edge-based and shape-based features, so even a simple backdoor trigger can become a common feature. For example, with just 5 poisoned samples, CNN on MNIST already reached ≈90% ASR, whereas CNN on CIFAR-10 only achieved ≈10% ASR under the same condition. With fewer features competing for space, a few poisoned samples will induce rapid ASR growth.

Second, compared to MNIST, GTSRB exhibited a comparable but slower trend, requiring more poisoned samples to obtain a high ASR. In contrast to MNIST, GTSRB features color images of traffic signs with greater intra-class variability in the size, shape, and background complexity. This added variance forces the model to capture more robust features, so that it is worse at directly mapping the backdoor trigger to some particular class. Hence, the model has to learn from more poisoned samples before capturing the backdoor pattern sufficiently enough to achieve notable ASR. For instance, GTSRB with Inception reached ≈60% ASR with 50+ poisoned samples, compared to only ≈40% with 25 poisoned samples.

Thirdly, CIFAR-10 was more resilient to the attack, even considering more poisoned samples. The greater resistance of CIFAR-10 models against backdoor attacks is likely due to the fact that the dataset has higher feature diversity, like objects with intricate textures, different light conditions, and realistic backgrounds. The model needs to focus on a higher number of discriminative patterns to classify images correctly, making it harder for a small backdoor trigger to be learned as a dominant feature. This effect is even more pronounced on ResNet models, which, due to their deeper architecture and skip connections, are designed to generalize and be resistant to learning spurious correlations.

The final ASR values for CNN models were consistently higher across datasets, suggesting that more complex architectures may offer superior protection against backdoors and making CNN models typically more susceptible than ResNet and Inception. This is an interesting observation because it is different from the general expectation that models capable of learning faster and more deeply, like ResNet and Inception would be more vulnerable to backdoor attacks. One possible explanation is that the combination of active learning and complex architectures changes the outcome of the attack, perhaps due to the fact that active learning provides better generalization and robustness in the early phases of learning. Additionally, it is known that creating a backdoor after the model has already been partially trained is more difficult than when training from scratch, which might explain why ResNet and Inception, which often benefits from deeper representations, shows more resistance in this setting.

Another visible outcome is the gap between CNN and ResNet performance for different datasets. For instance, CNN on GTSRB at 75 poisoned samples achieved $\approx$90% ASR, while ResNet only reached $\approx$70%. One hypothesis is that the CNN model can overfit poisoned samples in this dataset more easily since it has a less complex structure and less regularization, while ResNet is able to generalize better and hence suppress the backdoor signal better. However, the class imbalance and higher number of classes in GTSRB might also influence how effectively poisoned samples are integrated into training, especially in a post-query clean-label setting, where the success of the attack depends on how often the target class appears in the queried set. For Inception, the ASR increases with more poisoned samples, but the difference between datasets is less strong than in the case of CNN and ResNet. For MNIST, the ASR is relatively high, but less so than in the case of CNN or ResNet, which mirrors the fact that Inception is more resilient to small-scale triggers on simpler datasets. For both GTSRB and CIFAR-10, the ASR curves are essentially similar, with smooth rises and final ASR values close together. This suggests that Inception's architecture, in its ability to process multiscale features, responds more similarly across datasets, with less dataset-dependent variability in susceptibility. Although this may be evidence of more consistent generalization, it further suggests that once a trigger has been learned, it can linger regardless of dataset difficulty.

Clean Accuracy Drop (CAD)

Although ASR results showed high attack effectiveness for certain settings, the classification accuracy on clean data remained largely unaffected. As seen in Appendix B, Tables B.1–B.3, the clean accuracy drop (CAD) was minimal, suggesting that the attacks were stealthy and did not compromise the general performance of the model.

5.2. Results of Experiment 2

This section explains the result of Experiment 2, in which the subtrigger division and parameter adjustment methods were tested together with the LIRA and WaNet triggers. The corresponding parameters of the LIRA and WaNet triggers are: perturbation magnitude (ϵ), scaling factor (s), and number of transformations (k). These parameters were varied to get a more complete overview of the effects of the backdoor attack. In Appendix A, Tables A.4 and A.5 provide a full tabular summary of the ASR values observed in this experiment.

ASR Evolution Across Active Learning Cycles

Figure 5.3 shows the ASR trends across the selected datasets and models for the trigger types evaluated in Experiment 2. Each curve in the subplots represents a specific trigger type, allowing the comparative assessment of how ASR develops over time in response to active learning updates.

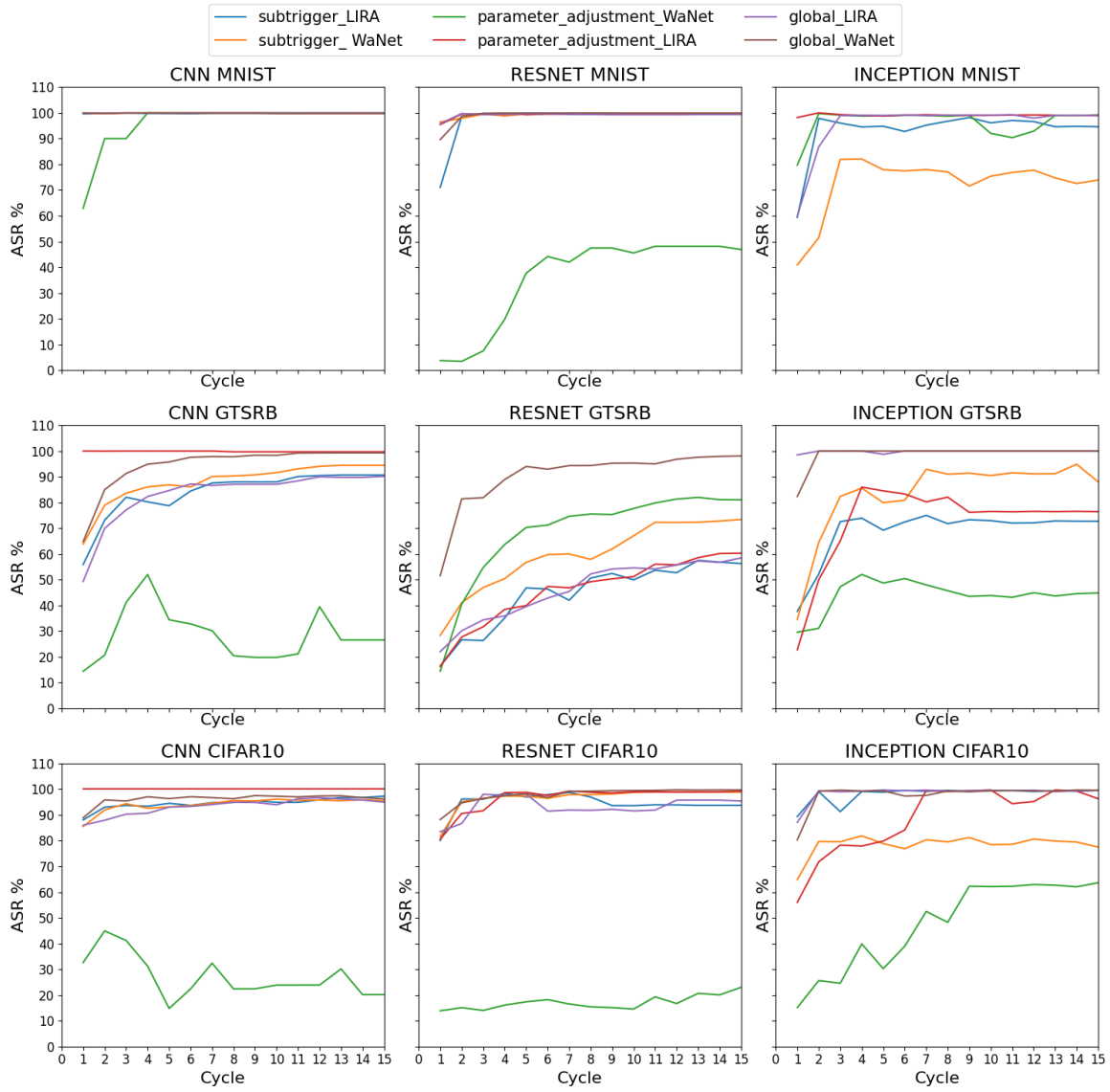

Figure 5.3: ASR progression over active learning cycles for different models, datasets, and poisoning strategies.

Figure 5.3 shows clear differences in how the ASR progressed over the cycles for the global, subtrigger, and progressive approaches. For the global triggers, it is evident that higher ASR results were obtained, when compared to the subtrigger and progressive methods. This is expected, as the global approach exposes the full-strength trigger from the beginning of training, giving the model ample time to learn the backdoor signal reliably. For example, on Inception with GTSRB, global WaNet and LIRA attained ≈100% ASR within the first 3 cycles, while sub-trigger WaNet and LIRA attained only ≈70-90% ASR even at cycle 15. However, there are exceptions, such as in the ResNet, GTSRB setting, where global LIRA underperforms compared to progressive and subtrigger WaNet in later cycles. This can be attributed to the combination of dataset complexity and model depth: the 43 GTSRB classes require longer adaptation to distinguish subtle backdoor cues, and ResNet's deeper structure may initially resist overfitting to static global triggers, while benefiting more from gradually reinforced or diversified poisoning signals. The subtrigger method, specifically, shows varied results and generally takes longer than the global method to achieve a higher ASR value. This delay is due to the fact that the full trigger is never shown to the model during training; only fragmented sub-parts are injected in separate samples. As a result, the model must implicitly learn to associate these disjoint fragments with the target class, which requires repeated exposure and makes the learning process slower and less stable.

Furthermore, the progressive poisoning method also had intermediate performance with steady ASR improvement that typically takes longer to converge. As the strength of the trigger increasingly grows across the training iterations, it delays the point at which the backdoor reaches enough strength to consistently affect predictions.

A comparison between WaNet and LIRA shows that LIRA obtains a higher ASR across all datasets and model architectures. This result is true for almost all poisoning tactics used. The rationale behind this lies in the inherent differences in the working mechanism of the two triggers: LIRA uses adversarial attacks with careful design that move samples outside the decision boundary. Such attacks are deeply intertwined with the model's internal gradients, making them extremely effective—no matter the size, fragmentation, or sequential usage. On the other hand, WaNet-based attacks show higher variability, especially on more complex datasets such as CIFAR-10, where the ASR remains lower and less stable over cycles. This may be due to more complex datasets like CIFAR-10, where images are noisy, textured, and contain much visual variation. The spatial warping in WaNet make it harder for the model to reliably associate a single class, resulting in higher variability and lower stability in ASR across cycles. The difference between WaNet and LIRA is especially noticeable in the subtrigger and progressive poisoning methods, where LIRA obtains larger ASR values at a faster pace while WaNet is slower to train to the same extent of attack efficacy. This is because WaNet's trigger is based on spatial transformations that when applied as subtriggers, the individual warped fragments may not be visually or functionally significant enough to guide the model toward a consistent backdoor behavior.

A few notable outliers were observed in the ASR outcomes, especially in the progressive parameter adjustment WaNet setups of CNN and ResNet with CIFAR-10. These specific setups showed low or varying ASR across the cycles. This behavior can be caused by the interaction of the two different backdoor methods. The progressive method gradually modifies trigger parameters across the cycles (gradually increasing warping strength in WaNet). For CIFAR-10 with progressive WaNet, the low ASR may be due to the high inter-class visual variability in the dataset, combined with the fact that early-cycle perturbations in the progressive schedule may have been too subtle to be learned effectively as a backdoor signal, leading to a failure in establishing the attack early on. Comparison across models shows that the biggest difference is that Inception consistently had on average smaller ASR values and more variability from cycle to cycle. Compared to CNN, which quickly reached high ASR, Inception's ASR increased slowly and was lower overall. This was clear in the subtrigger WaNet settings, where ASR stayed significantly below the other models throughout the cycles. The ASR curves of Inception also fluctuated more from cycle to cycle, indicating less stable learning of the backdoor trigger. Overall, though Inception was less vulnerable to backdoor attacks, it also showed slower and less stable ASR growth, which is different from the more regular pattern of CNN and ResNet.

ASR Dependence on the Number of Poisoned Samples

Figure 5.4 explores the relationship between number of poisoned samples and the final ASR (at cycle 15) for different trigger types, by datasets. The results disclose key highlights in ASR growth, based on the number of poisoned samples and trigger types. In MNIST, ASR reaches almost 100% with very few poisoned samples, for all strategies excluding the progressive approaches. This can be attributed to the nature of the dataset: MNIST is composed of low-resolution grayscale images with digit classes that are well-separated, and hence it is easy for even a limited number of poisoned samples to control the model. Global and sub-trigger approaches can inject a strong and consistent backdoor signal from the start, leading to the rapid saturation in ASR. In contrast, progressive strategies start at reduced trigger strength and only increment by cycles, deferring the buildup of the poisoned signal, which explains their slightly slower ASR increase.

For the GTSRB dataset, ASR increases with the number of poisoned samples, showing a peak for the global WaNet approach. GTSRB is a more complex dataset: it contains 43 visually similar traffic sign classes, most with minor variations, for which a stronger and stronger poisoning signal is required to deceive the model consistently. The spatial warping used in WaNet is especially effective in GTSRB, possibly due to its ability to mimic real-world image distortions (like camera blur or rotation), making the trigger more ”natural” and harder to ignore during training. The global version of WaNet applies the full-strength warping from the beginning, which explains its consistently superior performance in this setting. In CIFAR-10 we observe a performance between MNIST and GTSRB, reflecting the impact of the higher level of complexity and noise in the data set. CIFAR-10 is a dataset of natural images with

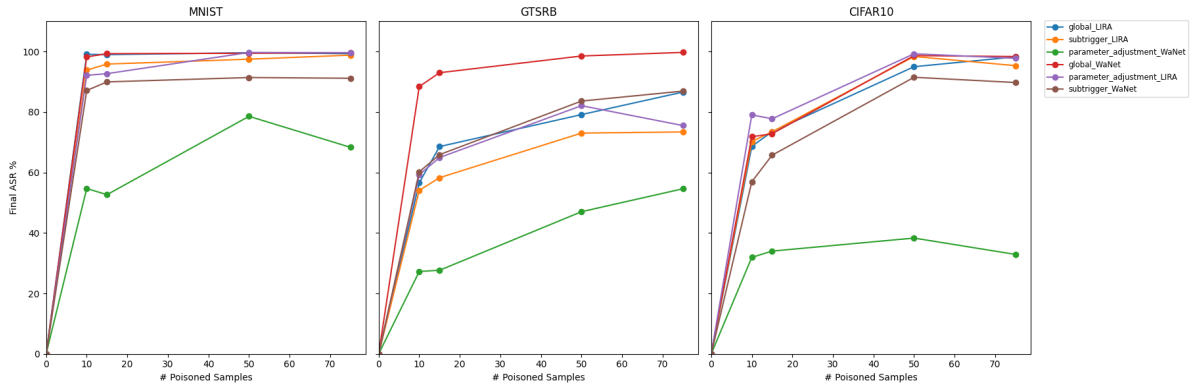

Figure 5.4: Final ASR as a function of the number of poisoned samples for different datasets and poisoning strategies.

high intraclass variability and low resolution (32x32 color images), this is possibly the reason for the weakened poisoned signal in the case of the WaNet progressive approach. Furthermore, CIFAR-10 contains just 10 classes, but the high visual richness creates difficulties for generalizing to unknown patterns, inclusive of subtle or dynamic triggers. Overall, progressive WaNet shows substantially lower ASR growth than all the other strategies, highlighting that this attack is less effective over all datasets. This can be attributed to the nature of the attack. Global and subtrigger approaches can inject a strong and consistent backdoor signal from the start, leading to the rapid saturation in ASR. In contrast, progressive strategies start at reduced trigger strength and only increment by cycles, deferring the buildup of the poisoned signal, which explains their slightly slower ASR increase.

5.2.1. Results of Progressive Parameter Adjustment

Figure 5.5 presents the results of the progressive parameter adjustment approach over active learning cycles. The visualization is divided into subplots by dataset and each curve in the subplot represents a different trigger (WaNet or LIRA). It is demonstrated how the progressive LIRA approach is consistently more effective than the progressive WaNet approach over all the datasets. This difference can be attributed to the nature of each attack. LIRA is based on adversarial perturbations that are crafted using gradients that exploit the model's vulnerabilities. When iteratively scaled up, such perturbations are increasingly successful at shifting the decision boundary of the model towards the target class. WaNet, however, relies on spatial warping of the input image, which, when applied with low intensity in the early cycles, may not be strong enough to leave a meaningful imprint on the model. As the warping intensity grows, it becomes more detectable but not necessarily better in terms of ASR.

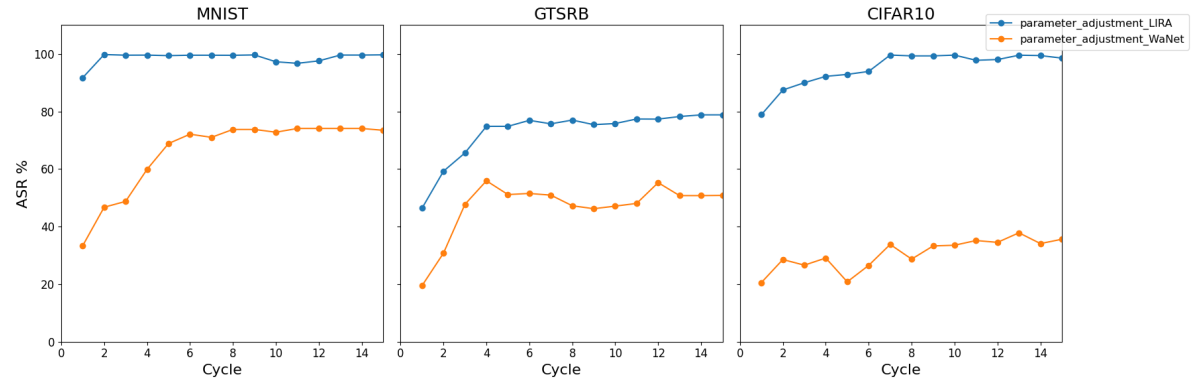

Figure 5.5: ASR progression for different progressively adjusted poisoning strategies across datasets.

Furthermore, the choice of dataset also has an effect on the performance, showing especially high ASR for the progressive LIRA in MNIST and CIFAR-10. The MNIST dataset, characterized by its grayscale

and low-resolution images alongside just 10 distinctly separated classes, possesses a comparatively simpler structural design. Such simplicity facilitates the effectiveness of even moderately robust triggers, such as those generated by WaNet. Conversely, the GTSRB dataset comprises 43 traffic sign classes, which introduces considerable inter-class similarity and real-world noise, potentially reducing the efficacy of an individual evolving trigger. CIFAR-10 is particularly challenging for WaNet: it consists of small, full-color natural images of highly diverse object categories. The poor performance of progressive WaNet in CIFAR-10 and may be explained by the combination of factors: the color complexity, the high-frequency content of natural images, and the low resolution all make spatial warping less distinguishable or learnable as a trigger. Additionally, the progressive intensity in WaNet may introduce noise that the model fails to interpret as a meaningful signal tied to a target class. For further analysis, Figure 5.6 and Figure 5.7 compare the ASR of different progressive triggers over cycles categorized by model and dataset.

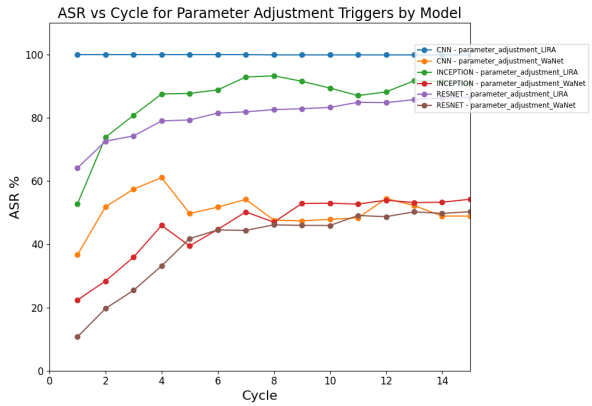

Figure 5.6: ASR over cycles for different trigger types, by model.

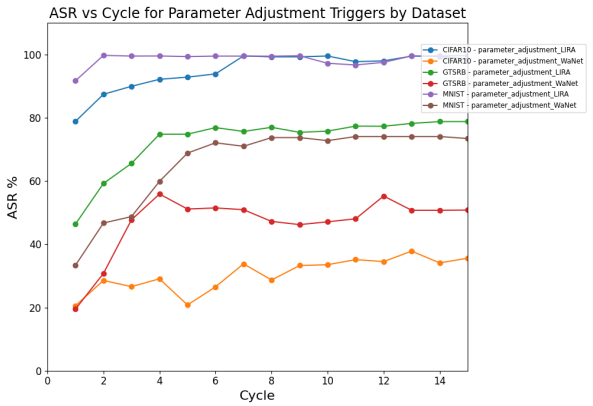

Figure 5.7: ASR over cycles for different trigger types, by dataset.

Figure 5.6 shows that CNN obtained higher ASR compared to ResNet models across all cycles, with progressive LIRA performing better than progressive WaNet. Increased ASR scores reported in CNNs over ResNet models could be a result of architectural variations in feature extraction and robustness. Skip connections and deeper layers of ResNet are intended to learn hierarchical and higher-level representations that can result in improved generalization and robustness against finer perturbations such as those utilized in backdoor attacks. In contrast, CNNs may overfit more easily to the poisoned patterns, especially when the poisoned signal is subtle or scattered across training cycles, as in progressive and sub-trigger settings. For Inception, ASR progression was between ResNet and CNN for both progressive WaNet and progressive LIRA triggers. Even though it was unable to keep pace with the higher ASR of CNN, it outperformed ResNet in some cases, especially during the early to mid-cycles. As with the other models, progressive LIRA consistently achieved higher ASR than progressive WaNet, demonstrating its effectiveness across all architectures.

In Figure 5.7, the dataset variations reveal that MNIST reached an almost maximal ASR early on except for the progressive approach, while the GTSRB dataset had gradual increases. Progressive LIRA maintained the highest ASR compared to progressive WaNet. The superior performance of progressive LIRA over progressive WaNet across both architectures can be attributed to how each attack method responds to gradual increases in trigger intensity. In LIRA, the perturbations are crafted based on the model's gradients. As the perturbation strength increases with each cycle, the poisoned signal becomes more pronounced and better aligned with the model's decision boundaries, thereby enhancing the backdoor effect over time. By contrast, WaNet employs spatial warping; although greater warping intensity can render the trigger more perceptible, it can simultaneously distort the input in manners that impede learning over early cycles. This makes WaNet's progressive adjustment less stable or effective, particularly under clean-label constraints. The disparity in the ASR between GTSRB and CIFAR-10, despite their shared characteristic as colored datasets, is most likely a consequence of variations in class coarseness and visual complexity. The GTSRB dataset has 43 distinct classes, a

number of which have subtle visual distinctions, thereby complicating the model's ability to correctly link a progressively reinforced but static trigger to a single target class. Additionally, the dataset includes more real-world variability (lighting, blur, angle), which may interfere with the warping-based triggers of WaNet or the fine-tuned perturbations of LIRA. In contrast, CIFAR-10 has fewer classes with more distinct object categories, allowing a stronger trigger to more easily dominate the classification decision as its intensity increases across cycles.

5.2.2. Results of Sub-trigger Division

Figure 5.8 and Figure 5.9 show the results of the sub-trigger division method over the used datasets and models. The variation of ASR over the subtrigger configurations is depicted as well as the different LIRA and WaNet triggers. These configurations are specifically the variations in perturbation strength (ϵ), spatial transformation parameters (s-k), and the size of the sub-triggers (sub_size). Figure 5.8 displays the ASR distribution for sub-trigger poisoning, grouped by different hyperparameters. The left plot presents the trends for ASR in the LIRA subtrigger, based on varying perturbation strength. The right plot examines the ASR of the WaNet subtrigger under its own specific hyperparameters s and k. For LIRA, a lower perturbation strength (eps=0.1) results in lower ASR results for almost all the combinations. In contrast, increasing eps to 0.2 stabilizes ASR across models, but does not always guarantee higher succes. For WaNet, changes in scaling factor (s) and kernel size (k) show that smaller s values (0.2) lead to more variability in ASR.

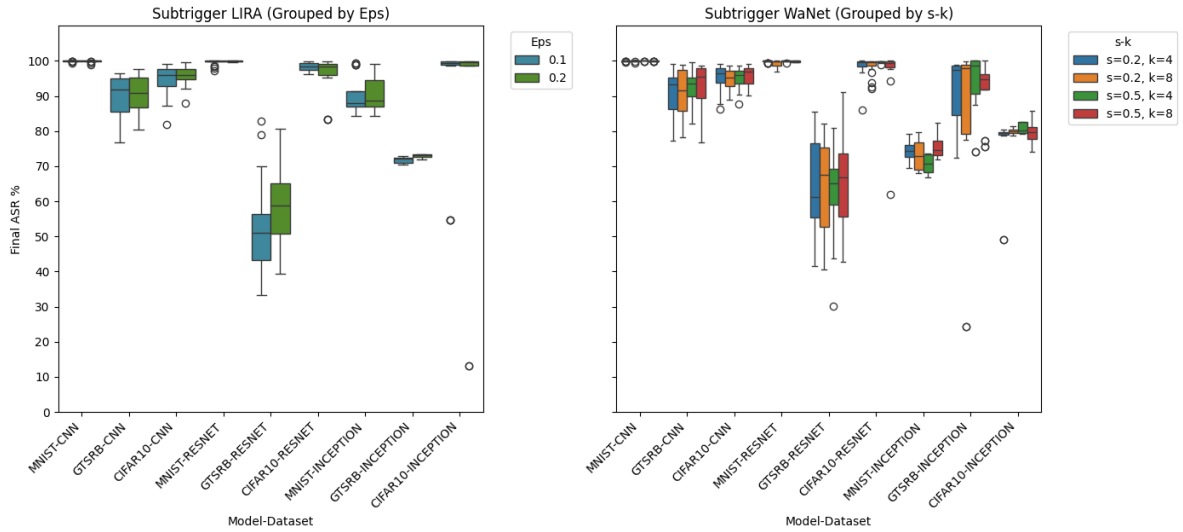

Figure 5.8: ASR distribution for sub-trigger poisoning across different datasets and models.

Figure 5.9 further extends the analysis of sub-triggers by visualizing the impact of the sub-trigger size on ASR. In the case of WaNet, the increase from 5 to 8 in size enhances the ASR in some instances marginally, but at times introduces more variability. For instance, increasing LIRA's sub-trigger size from 5 to 8 improved ASR from 85% to 95% on Inception with MNIST. For WaNet, when combined across training cycles, larger warped areas may help the model learn a more consistent backdoor signal, but they can also lead to noisier gradients or conflicting spatial cues, especially in deeper models like Inception that focus on spatial hierarchies, explaining the higher variability. In contrast, LIRA applies adversarial perturbations that are fine-tuned to the model's gradients and do not rely on spatial contiguity. Increasing the size of each sub-trigger has less of a negative effect on LIRA, since the model learns these perturbations independently, and their accumulation does benefit from a larger spatial footprint in contrast to WaNet.

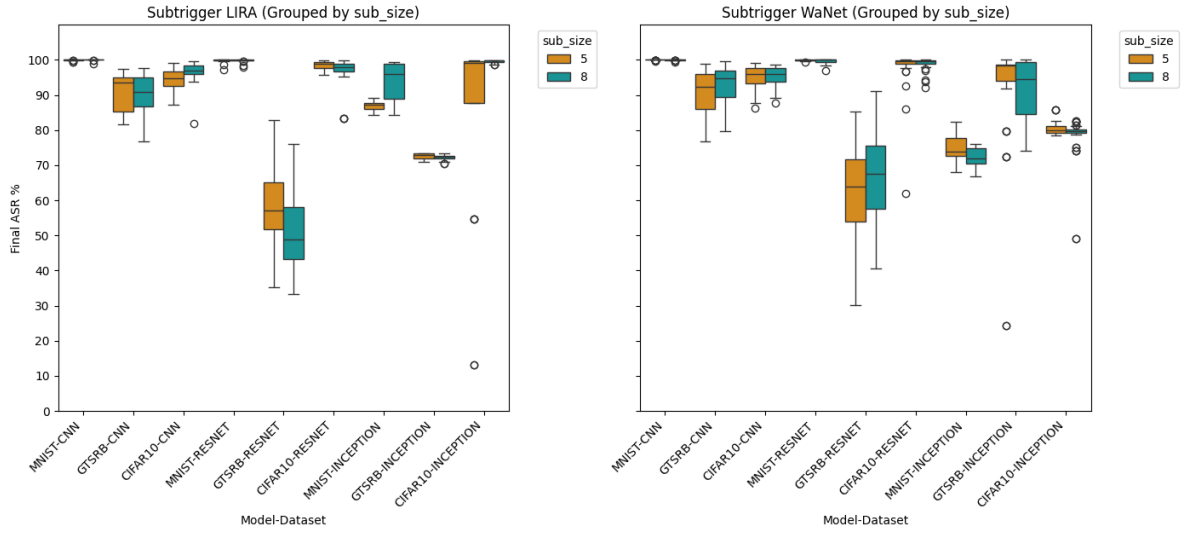

Figure 5.9: Impact of different sub-trigger configurations on ASR trends.

Clean Accuracy Drop (CAD)

Besides the ASR trends shown before, model accuracy on unpoisoned inputs was not significantly impacted. The CAD values reported in Appendix B, Tables B.4 and B.5 confirm that the poisoning strategies maintained relatively stable performance on clean data, reinforcing their subtlety.

5.3. Results of Experiment 3

This section presents the results of Experiment 3, the focus of the experiment was on clean label attacks and their effects on active learning models. In this experiment the LIRA and WaNet triggers were considered again. Note that only the global (original) methods were considered and the parameters for the LIRA and WaNet triggers were varied. For a complete breakdown of the ASR across the configurations tested here, refer to Table A.6 in Appendix A.

ASR Evolution Across Active Learning Cycles

Figure 5.10 presents the evolution of ASR along active learning iterations for different poisoning tactics in a clean-label attack, over a number of datasets and model configurations. The plot compares target-only subtrigger LIRA, target-only subtrigger WaNet, target-only global LIRA, and target-only global WaNet attacks. The analysis shows that global poisoning tended to achieve higher ASR than subtrigger techniques, especially on the MNIST and GTSRB datasets. For instance, CNN on MNIST with global WaNet reached ≈95% ASR, while with subtrigger WaNet it reached around ≈60%. The low ASR observed in certain situations may be due to the number of poisoned samples, which is sometimes not sufficient to achieve a high level of attack effectiveness. Since clean-label attacks with postquery poisoning rely on corrupting only within the queried set, there are occasions when the amount of target label samples in the selected batch is too low to significantly affect the model. This limitation is especially evident in GTSRB, likely due to the higher number of classes in the dataset. In addition, attacks using WaNet show higher ASR trends compared to LIRA, indicating that some poisoning techniques may be more vulnerable to the constraints imposed by clean-label poisoning.

In the clean-label poisoning setting, it is interesting to note that WaNet had better ASR trends than LIRA, a result that is opposite to earlier findings in dirty-label settings where LIRA performed better. This difference can be attributed to the nature of the clean-label constraint: LIRA relies on adversarial perturbations that are specifically tailored to flip predictions at training time, a technique that can be less effective when the label is not changed and the poisoning is limited to a small queried batch. In contrast, WaNet introduces a universal warping-based trigger that affects the input distribution more consistently across samples. This makes it more robust under clean-label constraints, particularly when the number

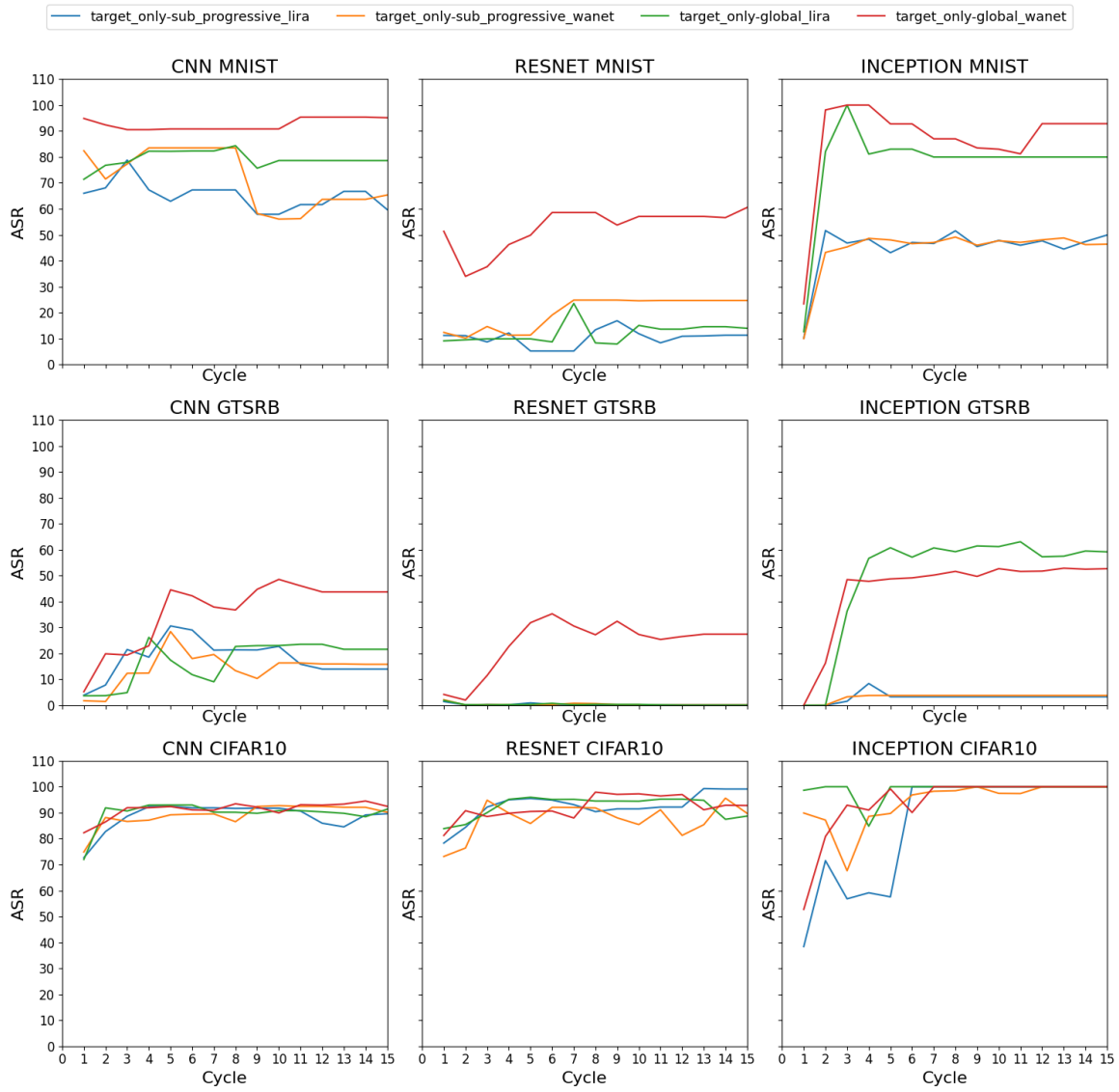

Figure 5.10: ASR progression for different sub-trigger poisoning strategies across datasets.

of poisonable samples is limited. Thus, under these stricter conditions, the visual and structural nature of the WaNet trigger may result in higher effectiveness despite LIRA's usual advantage in more flexible settings. In all datasets, CNN models consistently exhibit the highest ASR, particularly for MNIST and CIFAR-10, against LIRA and WaNet attacks. This implies that CNN models are more vulnerable to clean label attacks, most significantly in settings with simpler or more structured input distributions like MNIST. Respectively, ResNet models illustrate the lowest ASR on GTSRB, particularly in the case of subtrigger attacks, reflecting superior robustness, possibly as a result of the deeper structure and residual connections. Inception models illustrate a more subtle pattern: they fail to achieve the same ASR as CNNs, global triggers (particularly WaNet) working significantly better than subtrigger ones.

ASR Dependence on the Number of Poisoned Samples

Figure 5.11 shows the number of poisoned samples in each active learning cycle for various trigger attack tactics in a clean-label setting. This figure illustrates that the poisoned samples, during the training process influence the ASR patterns seen in Figure 5.10.

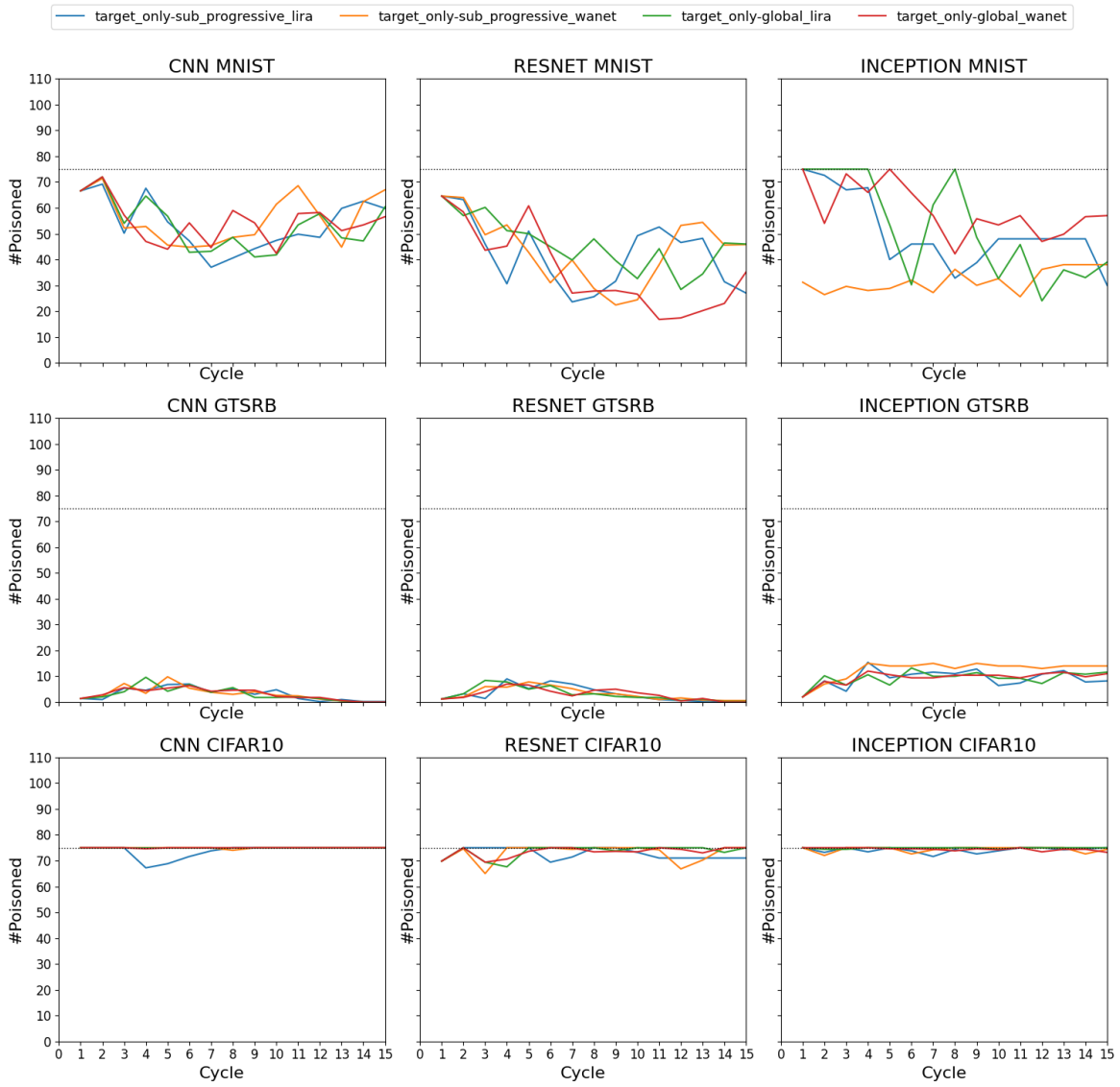

Figure 5.11: Number of poisoned samples per cycle for different sub-trigger attack strategies.

The curves in the plots represent how the number of poisoned samples differs across cycles for each combination. It reflects that there are variations in the selection of queried samples containing the target label and therefore variations in how many samples are poisoned. The dotted line in each plot represents the maximum possible number of poisoned samples per cycle, serving as a reference for comparisons. It is shown that there are notable differences in the number of samples poisoned every cycle on datasets, even more for GTSRB and MNIST. As seen in Figure 5.10, this probably contributes to the low ASR seen in these datasets. On the other hand, the high ASR of CIFAR-10 in the clean-label scenario, is consistent with the amount of poisoned samples every cycle. For certain models, like ResNet on MNIST, the number of poisoned samples decreases, indicating that fewer target-labeled samples are chosen later on, which may reduce the attack's efficacy.

Clean Accuracy Drop (CAD)

In this clean-label attack scenario, although backdoor effectiveness varied, the clean data accuracy remained relatively stable. Appendix B, Table B.6 indicate that the CAD remained modest across most configurations, reflecting a limited impact on overall model accuracy.

5.4. Results of Experiment 4

This section presents the results of Experiment 4, which focused on the comparison of pre-query and post-query poisoning methods. For this experiment, the global versions of LIRA and WaNet were considered, and the poison rate was varied. The experiment was carried out on the CNN model and the three standard datasets that were used for the previous experiments. The ASR results discussed in this section are also presented in tabular form in Appendix A, Table A.7.

ASR Dependence on the Number of Poisoned Samples

Figure 5.12 shows how the ASR progresses over cycles for both pre-query and post-query poisoning approaches. Note that the lines in each plot represent the unique poison rates and trigger type combinations. For this experiment, the global LIRA and WaNet trigger methods were used, as well as poison rates of 1% and 5%. The evolution of the ASR is shown over 15 active learning cycles for pre-query and post-query poisoning attacks against the MNIST, the GTSRB, and the CIFAR-10 data sets. A clear pattern is seen where the post-query poisoning reaches higher and more stable ASR levels than pre-query poisoning, notably through the latter cycles. This can be explained by the nature of post-query poisoning: the attacker poisons only those samples that the model has already selected based on its uncertainty, ensuring that every poisoned sample is used in retraining. In contrast, pre-query poisoning injects poisoned data into the pool before selection. Since the active learning query strategy is unaware of which samples are poisoned, there is no guarantee that the poisoned samples will be selected, leading to underexposure or inconsistent learning of the backdoor pattern, especially in early cycles. This effect accumulates across cycles, contributing to the lower and less stable ASR observed in pre-query settings.

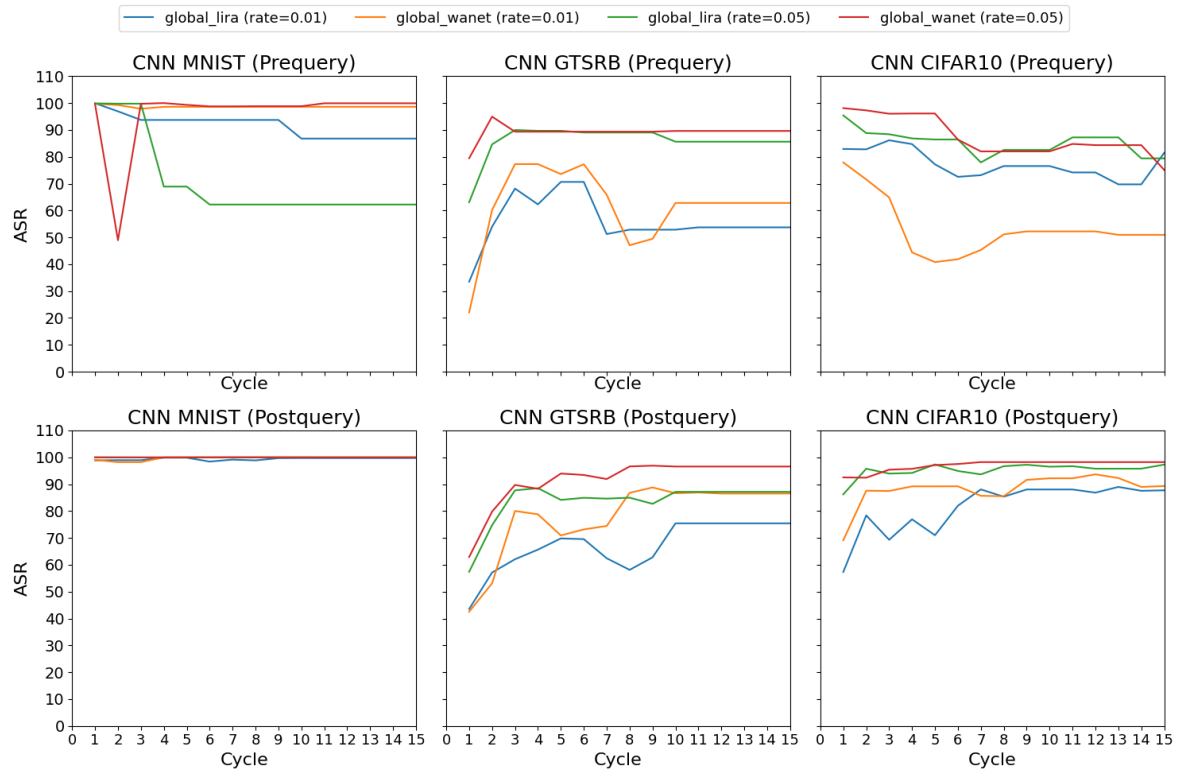

Figure 5.12: ASR progression for pre-query and post-query poisoning across datasets, trigger methods and poison rates.

For the MNIST dataset, pre-query poisoning shows high fluctuations in ASR through the first cycles, notably with the use of the WaNet trigger. This is likely due to MNIST's low visual complexity: the images are simple and uniform, so slight perturbations introduced by the WaNet warping (especially in early cycles) may be too subtle to be reliably learned unless the samples are actually queried. If those poisoned samples are missed in pre-query, the model does not easily link the trigger to the

target label. Contrary, the post-query poisoning remains high right from the start because the poisoned samples are guaranteed to be queried and used for model updates, allowing the backdoor to be learned consistently from the first cycle onward. For instance, in MNIST, post-query WaNet reached ≈100% ASR, while pre-query fluctuated around ≈60–100%. For the CIFAR-10 and the GTSRB data sets, the post-query poisoning reaches a high and steady ASR earlier than the pre-query poisoning does, with higher variability before settling. This is because these datasets are visually more complex and diverse than MNIST, so learning a consistent backdoor pattern requires repeated and reliable exposure to poisoned examples. In the pre-query setting, the uncertainty-based selection may omit poisoned samples or introduce them sporadically, causing convergence of the backdoor to be delayed. The post-query approach avoids this by targeting exactly those samples where the model is most uncertain, so that poisoned inputs are conspicuous and will likely affect training.

In addition, the poisoning rate significantly impacts the robustness and performance of ASR. A review across all the data sets indicates a high poisoning rate of 5% generally causes higher and steadier ASR values compared with 1% poisoning, shown most prominently in the pre-query context, where lower poisoning rates cause increased variability across many cycles. On the contrary, post-query poisoning shows a consistent effect regardless of the poisoning level, with minimal fluctuations across the cycles. That is because even with fewer poisoned samples, the attacker forces them directly into the queried set, so they will always be part of the training procedure. Therefore, while the absolute number of poisoned samples fluctuates from 1% to 5%, their impact per cycle is always maximized in post-query poisoning, which is responsible for the more stable and efficient ASR evolution.

Clean Accuracy Drop (CAD)

The comparison of pre-query and post-query poisoning revealed some differences in ASR, yet the model's classification accuracy on clean samples was not drastically affected. As shown in Appendix B, Table B7 B.7, the CAD remained low, underscoring the covert nature of the attacks.

6 Discussion

This chapter provides an in-depth analysis of the outcomes obtained from experiments conducted in this research work. The obtained findings are described in relation to the existing literature and highlight significant contributions of this research. Moreover, effective defense mechanisms against backdoor attacks in active learning and the initial framework to research these vulnerabilities are addressed.

6.1. Addressing the Research Questions

RQ1: Backdoor Attack Settings and AL parameters

Experiment 1 was conducted to answer RQ1 by exploring the effect of different backdoor attack parameters on ASR. The study demonstrates that the type of trigger that is selected for the backdoor attack greatly affects its effectiveness. From the results, larger triggers are generally more likely to produce a higher ASR than smaller subtle triggers. However, the larger triggers can cost the attack in terms of stealthiness, making it more detectable. This aligns with current literature, such as [20], where the visibility of the trigger is normally sacrificed against detectability. Interestingly, when the specialized pattern trigger was employed, no difference in ASR was found with respect to the large trigger, suggesting that while complexity in trigger design may have some effect, size remains the dominant factor in determining ASR. Meaning that it has a greater effect on the feature space, embedding itself more solidly in the learned decision boundaries and making it more likely that backdoor activation will succeed.

As for the poisoning methods, there is a clear difference in terms of effectiveness. The certainty poisoning method obtained the highest ASR, followed by random poisoning and uncertainty poisoning. The previous statement indicates that in the AL process, where the most uncertain samples are typically queried, the poisoning of the most certain samples among them becomes the most efficient attack strategy. This can be explained by the fact that highly certain samples are more likely to be learned confidently and retained by the model, making them ideal carriers for the backdoor signal. When these confidently predicted (but poisoned) samples are added to the training set, the model reinforces the backdoor pattern more strongly. In contrast, uncertain samples may be more ambiguous or noisy, leading to weaker integration of the backdoor trigger during training. This result turned out to be a first contribution to the field, as it proposes an attack strategy that exploits unique characteristics of the active learning pipeline.

As is expected for backdoor attacks, higher poisoning rates increased the ASR. The percentage of poisoned samples is calculated based on the queried set size and not on the entire pool set. Even with a smaller poisoning percentage, a significant ASR improvement is observed, corroborating the fact that minimal poisoning can have a dramatic impact on active learning pipelines. Interestingly, the results reveal that relatively low numbers of poisoned samples are needed to achieve high ASR, especially for the MNIST and GTSRB data sets. This shows how these data sets are somewhat susceptible to backdoor attacks. The number of active learning cycles is another crucial element in the observed

ASR progression. For nearly all the different dataset-model combinations in Experiment 1, the first 5-10 cycles see the greatest leap in ASR, afterwards the ASR almost remains constant. The aforementioned shows that the attack was most effective in the earliest phases of training, likely because the model is still learning its internal representations and is more vulnerable to the impact of new data. As training continues and the model becomes increasingly confident and stable in its predictions, the impact of additional poisoned samples diminishes, hence the plateau in ASR that we observe in later cycles. This particular finding is relevant for the creation of defense mechanisms in AL, as it indicates that countermeasures should be targeted at these early active learning cycles.

It is shown that clean-label backdoor attacks can achieve a relatively high ASR, especially in the case of the CIFAR-10 dataset, where the ASR was found to be even higher than that of MNIST. This is interesting since previous experiments indicated that CIFAR-10 is comparatively more resistant to backdoor attacks. The success of the CIFAR-10 dataset can be explained by the fact that it has a balanced class distribution, which increases the likelihood that the target class appears frequently in the queried set. Since the attack in this experiment relies on post-query poisoning with clean labels, its effectiveness depends on the number of target class samples present in the queried set. In datasets like GTSRB, which have class imbalance and a larger number of classes, fewer target class samples may be available per query batch, limiting the number of poisoned samples and resulting in a lower ASR compared to CIFAR-10. Both the sub-trigger method and the global methods for LIRA and WaNet were analyzed in clean label serrings. The ASR of these methods was as expected, considering the amount of poisoned samples that were actually used. The clear disadvantage of clean-label poisoning in this study's setting is that the number of poisoned samples for the retraining of the model can not be guaranteed. Clean-label backdoor attacks poison samples from one particular target class in the queried set. Thereafter, the number of available samples can be significantly less than in standard poisoning approaches due to the label constraint. This could limit the effectiveness of an attack when insufficient poisoned samples are queried in the AL pipeline. One potential solution for the aforementioned limitation can be the utilization of diversity-based poisoning techniques, in which the querying will be more evenly spread across classes. This could reduce sparse target class samples in the queried set and improve ASR for clean-label poisoning.

Interestingly, [21] proposes Density-based Representative Sampling (DRE) to improve robustness against adversarial data manipulations. DRE tries to maintain a balance between the queried set and the entire data pool [21]. Therefore, it could be interesting to explore whether the introduction of clean-label attacks decreases the mitigation performance or not. Another way to increase the success of clean-label attacks could be the use of pre-query poisoning, while ensuring that the poisoned samples have a higher chance of being selected in AL. This could be done through optimization of the visual or feature-based characteristics of the trigger itself, thus enhancing its impact while not sacrificing label integrity. Pre-query poisoning, as demonstrated in [60], solves the limitation of the post-query strategy, where the number of available target-class samples for poisoning may be small.

RQ2: How do different datasets and model architectures in computer vision influence the susceptibility of active learning frameworks to backdoor attacks?

All the experiments addressed in a way RQ2 through evaluating model architecture and dataset complexity impact on ASR in active learning models. The results outline differences in ASR development based on dataset-model selection. The used dataset within active learning plays a crucial role in attack effectiveness. Among the tested datasets, the simplest dataset, MNIST, achieved the maximum average ASR for almost all trigger type and poison method combinations (excluding the ResNet model). The attack easily approached 100% ASR in the first active learning rounds of poisoning, indicating the susceptibility of simple datasets to backdoor attacks. For the GTSRB dataset, ASR was also relatively high, but less than that observed for MNIST. In contrast, CIFAR-10 obtained the lowest ASR (excluding clean label attacks) and exhibited significant variation across active learning cycles. This indicates that while backdoor attacks remain powerful for more complex datasets, their effectiveness decreases with the complexity of the data. Furthermore, the lower ASR of CIFAR-10 proves that models that have been trained on more diverse and detailed image distributions are more resistant to poisoning attacks within active learning settings.

In terms of the models, they exhibit varying degrees of susceptibility to backdoor attacks. From Experiment 1, the CNN models reach high ASR within shorter time across datasets when compared to ResNet

models. ASR for CNN models increases steeply in the first 5-10 active learning cycles, with stabilization near 100% for MNIST and GTSRB datasets. In ResNet models ASR increases slower, particularly for CIFAR-10, where the ASR remains much lower throughout the learning process. However, even for simpler datasets such as MNIST and GTSRB, ResNet models reach high ASR in later cycles, though at a slightly lower rate than CNN models. This may be related to the influence of the active learning loop on model capacity. More complicated models like ResNet have more parameters, and therefore new, subtle patterns such as backdoor triggers may hardly impact overall behavior within a short period. It may therefore require more training effort to insert new ”knowledge”, like a backdoor. On the other hand, smaller models like CNNs can be easily controlled and are more prone to forget earlier patterns; thus they are more susceptible to rapid backdoor injection during early cycles. The Inception model showed mixed results across the experiments. In Experiment 1, it achieved relatively high ASR on MNIST, even outperforming CNN and ResNet in some poisoning settings. This shows that Inception is more vulnerable in complex datasets, most likely due to the fact that it is capable of extracting multiscale features, thus being able to learn even stealthy backdoor patterns. On GTSRB and CIFAR-10, however, its ASR was lower and less consistent, especially with random and uncertainty poisoning. In Experiment 2, which tested sub-trigger and progressive poisoning methods, Inception underperformed compared to CNN and ResNet, especially with sub-progressive WaNet, where its ASR was low across cycles consistently. This indicates that Inception is less reactive to fragmented or progressing triggers and performs better when the entire trigger is shown early.

RQ3: How can backdoor attacks employing sub-trigger division and parameter progression be designed to exploit vulnerabilities in active learning pipelines, and how do their impacts compare to standard backdoor techniques?

Experiment 2 was designed to answer RQ3, which focuses on the progressive parameter adjustment and sub-trigger method in terms of effectiveness. These experiments produced a considerably higher ASR than those seen in Experiment 1. However, the sub-trigger and parameter adjustment methods showed a small difference in ASR to their global counterparts. This means that besides the trigger type, the proposed techniques can be considered as good alternatives to static methods. By incrementally introducing the backdoor over multiple active learning cycles, the progressive approaches might be able to evade detection mechanisms more effectively than global triggers. Recent work by [35] shows that adversarial retraining and robust active learning techniques can defend against poisoning-based attacks in AL. One possible reason why progressive approaches could serve as a good counterpart to adversarial retraining is that they introduce the backdoor signal more subtly and gradually, potentially avoiding the sharp changes in loss or gradients that adversarial retraining mechanisms are designed to detect and suppress. Whether progressive parameter adjustment and sub-trigger methods would remain undetectable with the use of adversarial retraining as a defense is uncertain. Future research may compare these methods' detectability with that of baseline global trigger methods' detectability. Experiment 2 highlights the possibility of increased stealthiness of the progressive sub-trigger division and parameter adjustment methods. One main contribution of this study is showing that it is possible to divide triggers and poison data with its small individual subcomponents. Moreover, at the inference stage all the subcomponents can be combined to trigger the backdoor while obtaining high ASR. It can be worthwhile to examine the detectability of these mechanisms in comparison to conventional global trigger methods for determining whether they provide a substantial breakthrough in evading detection mechanisms.

The progressive parameter adjustment in WaNet had the worst performance in Experiment 2. The hypothetical reason for this bad performance is that the initial perturbations in WaNet are not strong enough for the attack to persist in subsequent stages. If the attack is not affecting the model initially, it may not develop into a strong backdoor over the cycles. The progressive parameter adjustment in LIRA does not show the same weaknesses as WaNet. The LIRA approach has high ASR overall, meaning that its progressive implementations are more effective at introducing the backdoor gradually without impacting the attack's effectiveness, possibly due to LIRA directly manipulating feature space representations in such a way that the model is more capable of internalizing the backdoor pattern more consistently even under minimal early poisoning. In comparison to WaNet's image-space perturbations that are likely too subtle in early phases to be learned appropriately. The results provide evidence that sub-trigger and progressive parameter adjustment methods can be effective alternatives to standard

global poisoning methods. Though their ASR improvements over traditional techniques are not significant, their stealthy growth potential is something to investigate further. The ability of these methods to gradually add backdoor patterns during active learning cycles without sacrificing high ASR is a valuable insight into how backdoor attacks can be optimized. Future work could examine the detectability of these techniques in order to evaluate their value in realistic adversarial situations.

RQ4: What is the impact of pre-query and post-query backdoor attacks on active learning models in computer vision?
Experiment 4 explored the differences in active learning pipelines between the pre-query and post-query poisoning techniques. Post-query poisoning shows higher and more uniform ASR across all datasets, following the results. This is likely because the attacker poisons the samples after the model selects them to guarantee that they're used in training. However, this approach assumes a high level of access to the model's internal querying process, making it less realistic in many real-world attack scenarios. By contrast, pre-query poisoning, as used in the Double-cross paper [60], reflects a more practical threat model in which the attacker can only poison the unlabeled data pool. However, since selection is based on model uncertainty, poisoned samples may not be chosen, especially in early cycles, leading to lower and more variable ASR. The results also show that higher poisoning rates are necessary to achieve reasonable effectiveness in the pre-query setting, as they increase the chance that at least some poisoned samples are selected. To make pre-query attacks more useful in a real-world scenario, future studies could look to design triggers that make a sample more probable to be queried. These triggers could incrementally increase model uncertainty and thus make the poisoned samples preferable for the active learner to incorporate them better in the learning process. In summary, post-query poisoning is stronger and suitable for extensive studies, while pre-query poisoning is more covert and realistic but requires more careful design and greater poisoning rates to be effective.

6.2. Implications for Defense Strategies

The experimental findings of this thesis have clear consequences for defense mechanism design and tuning in AL pipelines regarding backdoor attacks. This research confirms that the early active learning cycles are especially vulnerable to backdoor poisoning. Therefore, AL pipelines must prioritize defense mechanisms in the initial iterations. A few of the techniques that could be used are: sanitizing data at the start of training or auditing samples by using tools like Spectral Signatures [57] or Activation Clustering [9]. Observing model behavior can be used to catch anomalies early on before the poisoned samples are internalized by the model. Secondly, model architectures influence vulnerability, showing the need for model specific defenses. Small models like CNNs were discovered to be more susceptible to backdoor attacks in initial iterations, while models like ResNet or Inception required more training for the attack to be successful. Consequently, lightweight warning systems may suffice for CNNs, but richer feature monitoring and layered defenses may be required for larger models. Third, a major insight from the study in [60] is that anomaly-based filtering defenses, which are effective in other contexts, may directly conflict with the goals of active learning. AL pipelines intentionally select "uncertain" data points, often the same data that anomaly filters would discard. Removing such samples can greatly degrade AL's effectiveness and learning trajectory [60]. Future research must resolve this by developing detection methods that distinguish between useful anomalies and maliciously crafted ones.

Another finding is the effectiveness of clean-label and progressive/subtrigger backdoor attacks, which are inherently harder to detect. Clean-label poisoning evades standard label consistency checks and sub-trigger or progressive methods introduce the backdoor gradually, avoiding sudden spikes in training loss or prediction shifts. Temporal analysis methods observing the model's internal representations across AL iterations are needed for this. For example, measuring the feature activations or attention maps over time may help flag slow evolving corruption patterns before they fully become a functional backdoor. Lastly, educating human annotators on trigger detection may be promising in hybrid settings, especially for clean-label attacks where human annotators are unaware of malicious intent. However, the feasibility of this approach is limited by the imperceptibility of many modern triggers.

6.3. Proposed Framework for Analyzing Backdoor Attacks in AL

Based on the insights obtained from this thesis, a preliminary framework is put together to systematically analyze backdoor attacks within active learning (AL) pipelines. This framework has three major components:

1. Attack Surface and Poisoning Strategy Characterization

Before evaluating backdoor attacks, one must map where and how the attacker can interfere with the AL pipeline. This involves identifying the attack timing (pre-query or post-query), the type of label manipulation (clean-label or label-flipping), and the degree to which the trigger is perceivable or visible. Triggers may be static and obvious, or incrementally added and practically invisible. In addition, poisoning mechanisms vary in sample choice criteria, ranging from random and uncertainty-based to certainty-based. The knowledge of the attacker about the pipeline also influences the attack surface. All these factors together provide a systematic basis for investigating the feasibility, stealth, and effectiveness of different attack scenarios in active learning.

2. Trigger Effectiveness and Persistence Analysis

These properties assess how well a backdoor trigger continues to work with iterations of adversarial learning, it helps predictably at test time, and evades detection successfully. Several metrics, such as ASR improvement, visibility of the trigger, and interaction with specific dataset-model combinations, help quantify the resilience and stealth of different trigger approaches.

3. Dataset and Model Sensitivity Layer

Here, the framework explores the interaction of dataset complexity and model architecture in terms of their influence on vulnerability to backdoor attacks. Specifically, simple datasets like MNIST, when combined with CNNs, show high vulnerability early on; in contrast, more complex models like ResNet on datasets like CIFAR-10 show lagged but still high vulnerability.

6.4. Contributions

The aim of this thesis was to widen the understanding of backdoor attacks in the domain of computer vision within active learning trained models. The research contributed to the security of these type of models, by presenting the following main contributions:

- Demonstrating how the active learning pipeline could be exploited by backdoor attacks on computer vision model training.
- Investigating the impact of attack parameters (trigger type, injection rate, poisoning method, and query size) on the effectiveness and stealth of backdoor attacks in AL.
- Investigating how the different model and dataset choices affect the AL system and make it more vulnerable to backdoor attacks.
- Illustrating how techniques, such as sub-trigger splitting and parameter adjustment, can be utilized in the field of backdoor attacks to exploit some weaknesses.
- Testing the impact of clean label attacks and their relevance to active learning trained models.
- Suggest different methods to mitigate backdoor attacks in active learning based on the obtained results and highlights of the study.
- Creating an awareness of security threats related to active learning and promoting the formulation of detection and mitigation plans specific to the recognized vulnerabilities.

7

Limitations and Future Work

There are several limitations of this work that are important to consider. First of all, the experiments were conducted in a closed environment with pre-defined datasets and model architectures, and thus such behavior might not be exactly comparable to real world applications. The impact of backdoor attacks and countermeasures may rely on the data distribution, more complex neural architectures, and active learning pipelines in real-world scenarios. Second, while in this study various backdoor attack strategies were explored, adaptive adversarial strategies with dynamic adaptation to deployed defenses were not considered. Attackers could adaptively modify poisoning strategies over time, and therefore real-world adversarial settings are more difficult than those considered here. Furthermore, pre-query poisoning strategies, where sample selection is poisoned before querying, were not exhaustively explored, even though they have been shown to enhance the efficiency of such attacks.

Moreover, this study is primarily based on computer vision tasks, where data characteristics such as spatial structure and pixel correlations allow for backdoor attacks that exploit subtle visual triggers. Domains like natural language processing or speech recognition show different representations, such as text tokens or audio signals. Therefore, the attack and defense mechanisms in the scenario of backdoor attacks may be significantly different. Attack vectors and defenses designed for computer vision are not immediately relevant to these other areas and need to be adapted to each domain and explored further. Future work must explore how active learning based backdoor vulnerabilities manifest in those domains. Finally, computational constraints limited the scope of experimentation, particularly in scaling up active learning iterations and testing with a broader variety of model architectures.

Future work must focus on expanding knowledge about backdoor attacks using a broader set of data, models, and deep learning frameworks to achieve generalization. Investigating adversarial tactics that adapt to countermeasures will provide more light on general resilience to attacks in real-world settings. Besides this, employing stronger safeguard mechanisms such as differential privacy, adversarial training or anomaly detection could improve the security of active learning models. Further research into the role of human annotators in active learning, especially in high-risk applications such as medical imaging and autonomous systems, could allow for more effective security measures to be created.

8

Conclusion

This thesis investigated the vulnerability of active learning pipelines to backdoor attacks in computer vision. It was demonstrated that the intrinsic properties of the iterative nature of active learning, its reliance on small sets of labeled data, and sample selection processes can be exploited by attackers to embed backdoors effectively even in covert situations. Analysis indicates that some attack parameters, such as the size of a trigger, mechanism of poisoning, and injection time, constitute a core function in determining the attacks success rate. Simpler models and data are more susceptible to attacks, especially at the start of active learning cycles. Clean-label attacks and incremental mechanisms of poisoning also become very difficult to prevent because they can potentially evade normal means of detection.

Among the most important contributions of this work is the exploration of progressive triggers and sub-trigger partitioning techniques that introduce the backdoor signal progressively across a sequence of active learning cycles. These methods have proved to be effective alternatives to global attacks with the additional benefit of flexibility and show that the cyclical nature of active learning can be exploited by attackers. This contribution enhances an overall sense of security vulnerabilities in active learning systems and acts as a foundation for stronger defenses. An initial analysis framework was proposed for determining and grouping threats in active learning to make future work better at protecting against these vulnerabilities. In general, this thesis points out the requirement of security-oriented design in active learning, particularly for critical applications like autonomous vehicles, surveillance, and medical imaging.

9

Ethical Considerations

This thesis examined backdoor attacks in active learning to identify vulnerabilities and develop mitigation plans. All experiments were conducted in a closed environment and did not impact real-world applications or the operation of AI systems. The study followed responsible AI principles such as transparency, reproducibility, and ethical use of the data obtained. The aim was to improve the security of active learning trained models rather than make it simpler for adversaries to abuse it. Moreover, only public datasets were used, and the results were presented with a defensive point of view.

In addition, this study highlighted the importance of the trustworthiness and safety of machine learning systems. This research went along with ethical guidelines in artificial intelligence research and in cybersecurity, alerting the dangers that active learning trained models might bring. This is especially important in critical real-world applications such as autonomous systems and the healthcare field. In summary, this study contributed to building safer artificial intelligence applications. The knowledge gained will support better strategies toward protecting active learning models against emerging security threats.

References

[1] Delft AI Cluster (DAIC). *The Delft AI Cluster (DAIC)*. RRID:SCR_025091. High Performance Computing cluster consisting of Linux compute servers with processing power and memory for running large, long, or GPU-enabled jobs. 2024. DOI: 10.4233/rrid:scr_025091. URL: https://daic.tudelft.nl/.

[2] Ethem Alpaydin. *Machine learning*. MIT press, 2021.

[3] Laith Alzubaidi et al. "A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications". In: *Journal of Big Data* 10 (Apr. 2023). DOI: 10.1186/s40537-023-00727-2.

[4] Jordan T. Ash et al. *Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds*. 2020. arXiv: 1906.03671 [cs.LG]. URL: https://arxiv.org/abs/1906.03671.

[5] Eugene Bagdasaryan and Vitaly Shmatikov. *Blind Backdoors in Deep Learning Models*. 2021. arXiv: 2005.03823 [cs.CR]. URL: https://arxiv.org/abs/2005.03823.

[6] Eugene Bagdasaryan et al. "How To Backdoor Federated Learning". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 2938–2948. URL: https://proceedings.mlr.press/v108/bagdasaryan20a.html.

[7] Anurag Bhardwaj, Wei Di, and Jianing Wei. *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*. Packt Publishing Ltd, 2018.

[8] Anirban Chakraborty et al. *Adversarial Attacks and Defences: A Survey*. 2018. arXiv: 1810.00069 [cs.LG]. URL: https://arxiv.org/abs/1810.00069.

[9] Bryant Chen et al. "Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering". In: *ArXiv* abs/1811.03728 (2018). URL: https://api.semanticscholar.org/CorpusID:53250337.

[10] Xinyun Chen et al. "Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning". In: *ArXiv* abs/1712.05526 (2017). URL: https://api.semanticscholar.org/CorpusID:36122023.

[11] Jiwoong Choi et al. *Active Learning for Deep Object Detection via Probabilistic Modeling*. 2021. arXiv: 2103.16130 [cs.CV]. URL: https://arxiv.org/abs/2103.16130.

[12] Marius Cordts et al. *The Cityscapes Dataset for Semantic Urban Scene Understanding*. 2016. arXiv: 1604.01685 [cs.CV]. URL: https://arxiv.org/abs/1604.01685.

[13] Khoa D Doan et al. "LIRA: Learnable, Imperceptible and Robust Backdoor Attacks". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 11946–11956. URL: https://api.semanticscholar.org/CorpusID:244397177.

[14] Paul Doucet et al. *Bridging Diversity and Uncertainty in Active learning with Self-Supervised Pre-Training*. 2025. arXiv: 2403.03728 [cs.LG]. URL: https://arxiv.org/abs/2403.03728.

[15] Encord. *Active Learning Loop*. [Online; accessed March 3, 2025]. 2023. URL: https:https://encord.com/blog/active-learning-machine-learning-guide/.

[16] Yu Feng et al. "FIBA: Frequency-Injection Based Backdoor Attack in Medical Image Analysis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 20876–20885.

[17] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. "Deep Bayesian Active Learning with Image Data". In: *ArXiv* abs/1703.02910 (2017). URL: https://api.semanticscholar.org/CorpusID:6318455.

[18] Yinghua Gao et al. “On the Effectiveness of Adversarial Training Against Backdoor Attacks”. In: *IEEE Transactions on Neural Networks and Learning Systems* 35.10 (2024), pp. 14878–14888. DOI: 10.1109/TNNLS.2023.3281872.

[19] Leilani H. Gilpin et al. *Explaining Explanations: An Overview of Interpretability of Machine Learning*. 2019. arXiv: 1806.00069 [cs.AI]. URL: https://arxiv.org/abs/1806.00069.

[20] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. *BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain*. 2019. arXiv: 1708.06733 [cs.CR]. URL: https://arxiv.org/abs/1708.06733.

[21] Yuejun Guo et al. “DRE: density-based data selection with entropy for adversarial-robust deep learning models”. In: *Neural Computing and Applications* 35.5 (Feb. 2023), pp. 4009–4026. ISSN: 1433-3058. DOI: 10.1007/s00521-022-07812-2. URL: https://doi.org/10.1007/s00521-022-07812-2.

[22] Guy Hacohen, Avihu Dekel, and Daphna Weinshall. “Active Learning on a Budget: Opposite Strategies Suit High and Low Budgets”. In: *ArXiv* abs/2202.02794 (2022). URL: https://api.semanticscholar.org/CorpusID:246634642.

[23] Dilek Hakkani-Tur and Allen Gorin. “Active Learning For Automatic Speech Recognition”. In: *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on* (Sept. 2002). DOI: 10.1109/ICASSP.2002.5745510.

[24] Elmar Haussmann et al. “Scalable Active Learning for Object Detection”. In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. 2020, pp. 1430–1435. DOI: 10.1109/IV47402.2020.9304793.

[25] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[26] Kaiming He et al. “Identity Mappings in Deep Residual Networks”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 630–645. ISBN: 978-3-319-46493-0.

[27] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. *Handcrafted Backdoors in Deep Neural Networks*. 2022. arXiv: 2106.04690 [cs.CR]. URL: https://arxiv.org/abs/2106.04690.

[28] Vishal Kaushal et al. “Learning From Less Data: A Unified Data Subset Selection and Active Learning Framework for Computer Vision”. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2019), pp. 1289–1299. URL: https://api.semanticscholar.org/CorpusID:53980344.

[29] Seho Kee, Enrique del Castillo, and George Runger. “Query-by-committee improvement with diversity and density in batch active learning”. In: *Information Sciences* 454-455 (2018), pp. 401–418. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2018.05.014. URL: https://www.sciencedirect.com/science/article/pii/S0020025518303700.

[30] Ksenia Konyushkova, Raphael Sznitman, and Pascal V. Fua. “Learning Active Learning from Data”. In: *Neural Information Processing Systems*. 2017. URL: https://api.semanticscholar.org/CorpusID:5878784.

[31] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. “CIFAR-10 (Canadian Institute for Advanced Research)”. In: (). URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[32] Yann LeCun, Corinna Cortes, and CJ Burges. “MNIST handwritten digit database”. In: *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[33] Yudong Li et al. “ Backdoor Attacks to Deep Learning Models and Countermeasures: A Survey ”. In: *IEEE Open Journal of the Computer Society* 4.01 (Jan. 2023), pp. 134–146. ISSN: 2644-1268. DOI: 10.1109/OJCS.2023.3267221. URL: https://doi.ieeecomputersociety.org/10.1109/OJCS.2023.3267221.

[34] Yudong Li et al. “Backdoor Attacks to Deep Learning Models and Countermeasures: A Survey”. In: *IEEE Open Journal of the Computer Society* PP (Jan. 2023), pp. 1–12. DOI: 10.1109/OJCS.2023.3267221.

[35] Jing Lin, Ryan Luley, and Kaiqi Xiong. *Active Learning Under Malicious Mislabeling and Poisoning Attacks*. Dec. 2020. DOI: 10.48550/arXiv.2101.00157.

[36] Yingqi Liu et al. “Trojaning Attack on Neural Networks”. In: *Network and Distributed System Security Symposium*. 2018. URL: https://api.semanticscholar.org/CorpusID:31806516.

[37] Giulio Lovisotto, Simon Eberz, and Ivan Martinovic. “Biometric Backdoors: A Poisoning Attack Against Unsupervised Template Updating”. In: *2020 IEEE European Symposium on Security and Privacy (EuroSP)*. 2020, pp. 184–197. DOI: 10.1109/EuroSP48549.2020.00020.

[38] Katerina Margatina et al. “Active Learning by Acquiring Contrastive Examples”. In: *Conference on Empirical Methods in Natural Language Processing*. 2021. URL: https://api.semanticscholar.org/CorpusID:237442331.

[39] Orson Mengara. “A Backdoor Approach with Inverted Labels Using Dirty Label-Flipping Attacks”. In: *IEEE Access* PP (Jan. 2024), pp. 1–1. DOI: 10.1109/ACCESS.2024.3382839.

[40] Chandrahas Mishra and D. Gupta. “Deep Machine Learning and Neural Networks: An Overview”. In: *IAES International Journal of Artificial Intelligence (IJ-AI)* 6 (June 2017), p. 66. DOI: 10.11591/ijai.v6.i2.pp66-73.

[41] Anh Nguyen and Anh Tran. *WaNet – Imperceptible Warping-based Backdoor Attack*. 2021. arXiv: 2102.10369 [cs.CR]. URL: https://arxiv.org/abs/2102.10369.

[42] Huaibing Peng et al. “On Model Outsourcing Adaptive Attacks to Deep Learning Backdoor Defenses”. In: *IEEE Transactions on Information Forensics and Security* PP (Jan. 2024), pp. 1–1. DOI: 10.1109/TIFS.2024.3349869.

[43] Fereshteh Razmi, Jian Lou, and Li Xiong. *Does Differential Privacy Prevent Backdoor Attacks in Practice?* 2023. arXiv: 2311.06227 [cs.CR]. URL: https://arxiv.org/abs/2311.06227.

[44] Pengzhen Ren et al. “A Survey of Deep Active Learning”. In: *ACM Computing Surveys (CSUR)* 54 (2020), pp. 1–40. URL: https://api.semanticscholar.org/CorpusID:221397441.

[45] Quentin Le Roux et al. “A Comprehensive Survey on Backdoor Attacks and Their Defenses in Face Recognition Systems”. In: *IEEE Access* 12 (2024), pp. 47433–47468. DOI: 10.1109/ACCESS.2024.3382584.

[46] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. “Hidden Trigger Backdoor Attacks”. In: *ArXiv* abs/1910.00033 (2019). URL: https://api.semanticscholar.org/CorpusID:203610516.

[47] Ozan Sener and Silvio Savarese. “Active Learning for Convolutional Neural Networks: A Core-Set Approach”. In: *arXiv: Machine Learning* (2017). URL: https://api.semanticscholar.org/CorpusID:3383786.

[48] Burr Settles. “Active Learning Literature Survey”. In: 2009. URL: https://api.semanticscholar.org/CorpusID:324600.

[49] Yanyao Shen et al. *Deep Active Learning for Named Entity Recognition*. 2018. arXiv: 1707.05928 [cs.CL]. URL: https://arxiv.org/abs/1707.05928.

[50] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. “Variational Adversarial Active Learning”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 5971–5980. URL: https://api.semanticscholar.org/CorpusID:90258881.

[51] J. Stallkamp et al. “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition”. In: *Neural Networks* 0 (2012), pp. -. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2012.02.016. URL: http://www.sciencedirect.com/science/article/pii/S0893608012000457.

[52] Emma Strubell, Ananya Ganesh, and Andrew McCallum. *Energy and Policy Considerations for Deep Learning in NLP*. 2019. arXiv: 1906.02243 [cs.CL]. URL: https://arxiv.org/abs/1906.02243.

[53] Masashi Sugiyama and Shinichi Nakajima. “Nakajima, S.: Pool-based active learning in approximate linear regression. Mach. Learn. 75, 249-274”. In: *Machine Learning* 75 (June 2009), pp. 249–274. DOI: 10.1007/s10994-009-5100-3.

[54] Chen Sun et al. “Revisiting Unreasonable Effectiveness of Data in Deep Learning Era”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[55] Christian Szegedy et al. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 31.1 (Feb. 2017). DOI: 10.1609/aaai.v31i1.11231. URL: https://ojs.aaai.org/index.php/AAAI/article/view/11231.

[56] Rinyoichi Takezoe et al. “Deep Active Learning for Computer Vision: Past and Future”. In: *APSIPA Transactions on Signal and Information Processing* 12.1 (2023). ISSN: 2048-7703. DOI: 10.1561/116.00000057. URL: http://dx.doi.org/10.1561/116.00000057.

[57] Brandon Tran, Jerry Li, and Aleksander Madry. “Spectral Signatures in Backdoor Attacks”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al. 2018, pp. 8011–8021. URL: https://proceedings.neurips.cc/paper/2018/hash/280cf18baf4311c92aa5a042336587d3-Abstract.html.

[58] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. “Clean-Label Backdoor Attacks”. In: 2018. URL: https://api.semanticscholar.org/CorpusID:53636567.

[59] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. *Label-Consistent Backdoor Attacks*. 2019. arXiv: 1912.02771 [stat.ML]. URL: https://arxiv.org/abs/1912.02771.

[60] Jose Rodrigo Sanchez Vicarte, Gang Wang, and Christopher W. Fletcher. “Double-Cross Attacks: Subverting Active Learning Systems”. In: *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 1593–1610. ISBN: 978-1-939133-24-3. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/vicarte.

[61] Athanasios Voulodimos et al. “Deep Learning for Computer Vision: A Brief Review”. In: *Computational Intelligence and Neuroscience* 2018 (2018). URL: https://api.semanticscholar.org/CorpusID:3557281.

[62] Bolun Wang et al. “Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks”. In: *2019 IEEE Symposium on Security and Privacy (SP)* (2019), pp. 707–723. URL: https://api.semanticscholar.org/CorpusID:67846878.

[63] Haoran Wang et al. “A comprehensive survey on deep active learning in medical image analysis”. In: *Medical image analysis* 95 (2023), p. 103201. URL: https://api.semanticscholar.org/CorpusID:269779048.

[64] Shanshan Wang et al. “Annotation-efficient deep learning for automatic medical image segmentation”. In: *Nature Communications* 12 (Oct. 2021), p. 5915. DOI: 10.1038/s41467-021-26216-9.

[65] Dominik Wermke et al. “”Always Contribute Back”: A Qualitative Study on Security Challenges of the Open Source Supply Chain”. In: *2023 IEEE Symposium on Security and Privacy (SP)* (2023), pp. 1545–1560. URL: https://api.semanticscholar.org/CorpusID:259373507.

[66] Xiaopeng Xie et al. *Shortcuts Arising from Contrast: Effective and Covert Clean-Label Attacks in Prompt-Based Learning*. 2024. arXiv: 2404.00461 [cs.LG]. URL: https://arxiv.org/abs/2404.00461.

[67] Jie Yang et al. “Clean□label poisoning attacks on federated learning for IoT”. In: *Expert Systems* 40 (2022). URL: https://api.semanticscholar.org/CorpusID:252883608.

[68] Zonghao Ying and Bin Wu. “NBA: defensive distillation for backdoor removal via neural behavior alignment”. In: *Cybersecurity* 6.1 (July 2023). ISSN: 2523-3246. DOI: 10.1186/s42400-023-00154-z. URL: http://dx.doi.org/10.1186/s42400-023-00154-z.

[69] Donggeun Yoo and In-So Kweon. “Learning Loss for Active Learning”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 93–102. URL: https://api.semanticscholar.org/CorpusID:148571749.

[70] Yi Yu et al. “Backdoor Attacks Against Deep Image Compression via Adaptive Frequency Trigger”. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 12250–12259. URL: https://api.semanticscholar.org/CorpusID:257232876.

[71] Xueying Zhan et al. “A Comparative Survey of Deep Active Learning”. In: *ArXiv* abs/2203.13450 (2022). URL: https://api.semanticscholar.org/CorpusID:247749004.

[72] Hangfan Zhang et al. “A3FL: Adversarially Adaptive Backdoor Attacks to Federated Learning”. In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=S6ajVZy6FA.

[73] Jiale Zhang et al. *Fine-tuning is Not Fine: Mitigating Backdoor Attacks in GNNs with Limited Clean Data*. 2025. arXiv: 2501.05835 [cs.LG]. URL: https://arxiv.org/abs/2501.05835.

[74] Yan Zhang et al. “Towards Backdoor Attacks against LiDAR Object Detection in Autonomous Driving”. In: *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. SenSys ’22. Boston, Massachusetts: Association for Computing Machinery, 2023, pp. 533–547. ISBN: 9781450398862. DOI: 10.1145/3560905.3568539. URL: https://doi.org/10.1145/3560905.3568539.

[75] Shihao Zhao et al. “Clean-Label Backdoor Attacks on Video Recognition Models”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 14431–14440. URL: https://api.semanticscholar.org/CorpusID:212628208.

[76] Yue Zhao, Ciwen Xu, and Yongcun Cao. “Research on Query-by-Committee Method of Active Learning and Application”. In: *Advanced Data Mining and Applications*. Ed. by Xue Li, Osmar R. Zaïane, and Zhanhuai Li. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 985–991. ISBN: 978-3-540-37026-0.

A

Tables of ASR Results

Tables A.1, A.2 and A.3 present the results of Experiment 1, evaluating the final (last cycle) attack success rate (ASR) over different query sizes. The tables summarize the ASR for various models and datasets at different poison rates. The results are further divided by poison method and trigger type.

Table A.1: Results of experiment 1 showcasing the final ASR of the model with poison rates of 1% and 5% over a query size of 500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	93.12	97.15	61.33	86.18	8.98	37.76	76.81	65.46	16.00	71.81	7.55	11.46	68.21	69.49	44.32	66.46	70.33	76.30
Certainty	90.18	98.43	70.30	86.01	8.25	54.08	3.74	83.68	8.99	65.25	9.49	92.57	85.65	85.63	26.82	53.04	86.54	89.95
Uncertainty	85.10	97.72	63.65	88.21	8.22	16.02	2.57	29.94	13.14	62.77	6.23	5.31	76.88	88.28	61.28	67.30	44.50	59.24
Trigger Type																		
Specialized	91.73	98.82	81.49	99.77	5.87	33.28	21.37	42.86	5.25	66.74	8.75	35.36	76.41	80.77	70.42	84.49	61.77	68.91
Big	91.80	97.59	55.85	85.44	12.60	43.42	32.45	88.38	27.90	70.15	7.18	36.65	76.75	81.78	48.55	82.17	65.97	87.36
Simple	84.87	96.90	57.94	75.19	6.97	31.16	29.31	47.85	4.98	62.94	7.34	37.33	77.59	80.84	13.45	20.15	73.62	69.22

Table A.2: Results of experiment 1 showcasing the final ASR of the model with poison rates of 1% and 5% over a query size of 1000 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	96.04	99.41	74.50	88.65	11.51	63.27	28.61	90.28	28.33	80.12	5.98	12.03	70.00	68.32	34.23	67.66	64.60	64.31
Certainty	95.90	99.67	71.42	89.19	14.41	79.83	38.01	93.44	21.32	82.79	13.43	84.94	87.16	88.86	34.57	62.55	75.93	89.55
Uncertainty	93.30	98.96	69.65	89.10	7.87	30.61	16.17	25.40	18.61	41.73	7.47	6.09	76.96	89.86	33.01	70.36	48.52	59.49
Trigger Type																		
Specialized	96.10	99.20	88.29	99.63	7.49	64.46	4.11	57.73	11.99	74.36	10.71	35.18	78.55	81.83	74.86	87.05	59.39	68.56
Big	94.21	99.23	66.70	88.50	13.72	57.83	70.26	84.93	38.31	76.35	7.61	34.31	78.97	81.43	3.85	94.01	68.32	68.09
Simple	94.93	99.61	60.59	78.80	12.58	51.42	8.42	66.46	17.95	53.93	8.58	33.57	76.60	83.78	17.73	23.70	61.34	76.68

Table A.3: Results of experiment 1 showcasing the final ASR of the model with poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	96.56	99.54	86.03	96.33	16.48	72.77	37.37	91.38	25.67	79.12	5.05	18.65	70.46	69.45	43.99	65.43	57.28	64.65
Certainty	96.59	99.41	85.58	95.10	21.23	89.76	42.66	97.47	32.79	94.62	21.99	80.53	88.14	90.43	34.87	54.56	85.51	91.75
Uncertainty	93.67	97.70	80.68	94.91	11.20	33.14	16.79	19.37	18.69	45.47	4.67	7.23	76.18	84.16	47.32	72.08	48.23	53.83
Trigger Type																		
Specialized	96.25	99.13	98.96	99.99	12.20	78.55	12.35	65.45	10.37	71.07	7.48	29.82	80.19	80.02	83.80	84.13	65.05	79.03
Big	94.99	98.75	81.78	97.08	19.21	62.17	66.59	77.87	42.62	77.77	13.66	42.32	79.03	83.85	25.37	80.60	59.76	67.51
Simple	95.58	98.78	71.55	89.27	17.51	54.95	17.89	64.90	24.16	70.37	10.57	34.27	75.57	80.17	17.01	27.34	66.21	63.70

Table A.4 and A.5 present the final (last cycle) ASR results of Experiment 2 over different query sizes. In this experiment the subtrigger division and parameter progression methods were tested. The results are categorized into three groups: Global, Subtrigger, and Progressive poisoning methods. The different triggers were the WaNet and the LIRA trigger and their unique parameters s, k, eps and ss (sub-size) were varied in this experiment.

Table A.4: Results of experiment 2 showcasing the final ASR of the model with poison rates of 1% and 5% over a query size of 1000 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Global																		
LIRA (eps=0.2)	89.99	99.99	88.81	97.29	88.04	97.82	98.21	99.28	23.56	63.81	39.61	99.42	99.32	99.12	87.90	82.07	99.21	96.78
WaNet (s=0.5, k=8)	89.87	99.99	47.99	66.62	50.66	58.15	62.25	68.24	75.24	93.30	26.95	58.00	98.97	98.97	57.75	71.58	82.60	80.72
Subtrigger																		
LIRA (eps=0.1, ss=5)	99.94	99.89	80.27	95.17	76.00	94.27	99.38	99.98	31.18	59.10	37.32	98.82	87.29	87.13	71.49	71.60	95.54	99.35
LIRA (eps=0.2, ss=5)	99.70	99.97	80.10	95.56	81.97	96.01	98.43	99.88	26.88	58.78	30.80	97.87	87.37	85.59	72.85	73.42	99.68	65.17
LIRA (eps=0.1, ss=8)	99.30	100.0	78.95	93.47	80.26	97.94	97.72	99.96	22.11	44.27	36.04	98.18	87.76	99.18	71.53	71.54	78.45	99.03
LIRA (eps=0.2, ss=8)	99.99	99.98	81.09	94.86	81.64	97.20	99.18	99.97	22.25	53.08	39.64	89.46	88.40	96.48	71.49	72.27	99.10	99.36
WaNet (s=0.2, k=4, ss=5)	99.98	99.92	82.75	94.83	87.32	96.28	99.52	100.0	26.82	66.42	38.24	97.05	74.07	75.38	85.42	82.98	79.84	79.56
WaNet (s=0.2, k=4, ss=8)	98.60	99.96	78.95	96.61	85.66	96.91	96.48	100.0	34.57	74.01	43.92	98.73	75.25	71.68	84.38	97.46	79.61	67.44
WaNet (s=0.2, k=8, ss=5)	99.83	99.99	81.48	97.16	86.89	94.79	97.94	99.85	33.19	66.74	48.12	98.15	74.72	77.21	74.12	57.60	78.88	80.90
WaNet (s=0.2, k=8, ss=8)	98.48	99.98	83.07	94.66	82.52	95.14	97.67	99.69	24.64	68.33	41.82	98.25	75.36	71.54	96.48	86.50	79.47	79.79
WaNet (s=0.5, k=4, ss=5)	99.66	100.0	83.15	93.62	82.79	95.06	98.00	99.91	30.07	65.35	33.17	99.83	73.61	73.48	89.78	96.73	79.80	81.20
WaNet (s=0.5, k=4, ss=8)	99.97	99.95	86.92	96.96	84.37	94.43	97.30	99.89	24.53	62.13	36.58	99.91	75.07	70.67	95.82	82.12	75.17	80.80
WaNet (s=0.5, k=8, ss=5)	99.72	100.0	74.45	95.75	88.86	95.56	99.45	99.67	28.02	50.72	47.49	91.32	76.10	76.63	98.55	96.99	73.89	80.12
WaNet (s=0.5, k=8, ss=8)	99.85	99.99	81.86	97.68	77.18	95.44	99.43	99.96	34.14	77.07	41.43	99.31	70.62	73.54	81.71	86.00	78.55	74.53
Progressive																		
LIRA (eps=0.2)	89.99	99.99	88.81	97.29	88.04	97.82	98.21	99.28	23.56	63.81	39.61	99.42	99.32	99.12	87.90	82.07	99.21	96.78
WaNet (s=0.5, k=8)	89.87	99.99	47.99	66.62	50.66	58.15	62.25	68.24	75.24	93.30	26.95	58.00	98.97	98.97	57.75	71.58	82.60	80.72

Table A.5: Results of experiment 2 showcasing the final ASR of the model with poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Global																		
LIRA (eps=0.2)	89.99	99.99	88.81	97.29	88.04	97.82	98.21	99.28	23.56	63.81	39.61	99.42	99.30	99.08	84.88	77.59	99.14	96.00
WaNet (s=0.5, k=8)	89.87	99.99	47.99	66.62	50.66	58.15	62.25	68.24	75.24	93.30	26.95	58.00	74.90	64.55	57.10	72.05	78.48	76.05
Subtrigger																		
LIRA (eps=0.1, ss=5)	99.94	99.89	80.27	95.17	76.00	94.27	99.38	99.98	31.18	59.10	37.32	98.82	87.30	86.97	71.59	71.47	94.53	99.40
LIRA (eps=0.2, ss=5)	99.70	99.97	80.10	95.56	81.97	96.01	98.43	99.88	26.88	58.78	30.80	97.87	87.43	85.91	72.85	73.42	99.68	56.48
LIRA (eps=0.1, ss=8)	99.30	100.0	78.95	93.47	80.26	97.94	97.72	99.96	22.11	44.27	36.04	98.18	87.96	99.22	71.59	71.36	73.15	98.96
LIRA (eps=0.2, ss=8)	99.99	99.98	81.09	94.86	81.64	97.20	99.18	99.97	22.25	53.08	39.64	89.46	88.49	95.93	71.72	72.36	99.18	99.24
WaNet (s=0.2, k=4, ss=5)	99.98	99.92	82.75	94.83	87.32	96.28	99.52	100.0	26.82	66.42	38.24	97.05	74.83	74.41	87.66	85.63	79.73	79.48
WaNet (s=0.2, k=4, ss=8)	98.60	99.96	78.95	96.61	85.66	96.91	96.48	100.0	34.57	74.01	43.92	98.73	75.96	72.17	86.88	97.44	79.61	64.38
WaNet (s=0.2, k=8, ss=5)	99.83	99.99	81.48	97.16	86.89	94.79	97.94	99.85	33.19	66.74	48.12	98.15	74.85	77.07	74.04	52.07	79.30	80.76
WaNet (s=0.2, k=8, ss=8)	98.48	99.98	83.07	94.66	82.52	95.14	97.67	99.69	24.64	68.33	41.82	98.25	75.04	72.15	97.00	88.72	79.39	79.72
WaNet (s=0.5, k=4, ss=5)	99.66	100.0	83.15	93.62	82.79	95.06	98.00	99.91	30.07	65.35	33.17	99.83	73.68	73.44	87.67	95.91	79.52	80.87
WaNet (s=0.5, k=4, ss=8)	99.97	99.95	86.92	96.96	84.37	94.43	97.30	99.89	24.53	62.13	36.58	99.91	74.65	70.64	94.78	80.79	75.75	81.10
WaNet (s=0.5, k=8, ss=5)	99.72	100.0	74.45	95.75	88.86	95.56	99.45	99.67	28.02	50.72	47.49	91.32	75.57	77.60	98.79	97.49	73.52	79.87
WaNet (s=0.5, k=8, ss=8)	99.85	99.99	81.86	97.68	77.18	95.44	99.43	99.96	34.14	77.07	41.43	99.31	71.19	73.52	81.26	84.56	78.54	74.62
Progressive																		
LIRA (eps=0.2)	89.99	99.99	88.81	97.29	88.04	97.82	98.21	99.28	23.56	63.81	39.61	99.42	99.30	99.08	84.88	77.59	99.14	96.00
WaNet (s=0.5, k=8)	89.87	99.99	47.99	66.62	50.66	58.15	62.25	68.24	75.24	93.30	26.95	58.00	74.90	64.55	57.10	72.05	78.48	76.05

Table A.6 presents a comprehensive overview of Experiment 3, in which clean-label attacks are tested. In the table, the results are categorized by trigger type and all the important parameters for the Lira and WaNet triggers are included.

Table A.6: Results of experiment 3 showcasing the final ASR of the model with poison rates of 2% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR		Final ASR	
2%	2% 5%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%
Global																		
LIRA (eps=0.2)	63.95	78.61	14.69	21.60	80.62	91.37	1.61	13.95	0.04	0.04	55.64	88.70	100.0	74.96	56.40	59.27	100.0	100.0
WaNet (s=0.5, k=8)	90.70	95.13	29.92	43.72	88.42	92.47	30.75	60.53	19.22	27.41	54.50	90.97	50.80	91.04	49.23	51.83	99.93	100.0
Subtrigger																		
LIRA (eps=0.1, ss=5)	34.03	65.96	12.63	16.02	80.36	89.36	7.33	21.30	0.37	0.37	62.42	97.82	46.28	48.15	1.57	3.12	66.29	100.0
LIRA (eps=0.2, ss=5)	60.85	55.60	15.65	10.04	80.03	90.81	3.64	17.72	0.05	0.08	58.13	98.27	47.00	47.38	3.16	3.25	100.0	100.0
LIRA (eps=0.1, ss=8)	55.11	75.25	14.09	17.74	69.03	89.68	11.66	26.59	0.19	0.11	59.98	97.91	47.11	46.52	3.16	3.90	87.29	100.0
LIRA (eps=0.2, ss=8)	48.13	59.71	13.06	13.98	76.13	89.59	3.98	11.28	0.17	0.21	54.83	99.12	45.67	49.87	3.18	3.18	100.0	100.0
WaNet (s=0.2, k=4, ss=5)	44.89	69.45	12.40	17.46	84.12	94.12	7.28	28.56	0.07	0.07	72.63	88.83	48.19	48.37	3.44	1.88	100.0	100.0
WaNet (s=0.2, k=4, ss=8)	68.85	58.05	24.32	19.92	81.67	87.95	4.13	14.24	0.46	0.37	58.03	92.58	46.79	47.35	3.18	3.34	100.0	100.0
WaNet (s=0.2, k=8, ss=5)	51.39	56.23	13.76	11.47	86.16	94.59	12.52	18.17	0.32	0.19	65.06	88.74	46.61	48.22	3.68	3.68	100.0	90.82
WaNet (s=0.2, k=8, ss=8)	63.05	67.60	15.31	16.04	80.06	89.73	7.78	25.00	0.34	0.28	34.49	99.55	46.04	46.61	2.65	3.67	100.0	100.0
WaNet (s=0.5, k=4, ss=5)	53.47	67.34	20.31	25.80	76.41	93.58	6.64	16.46	0.13	0.08	77.37	90.04	46.44	47.94	1.02	2.90	100.0	99.50
WaNet (s=0.5, k=4, ss=8)	86.91	55.88	8.63	11.66	82.62	88.64	8.84	22.46	0.26	0.16	48.53	95.27	46.17	49.80	3.62	3.65	100.0	100.0
WaNet (s=0.5, k=8, ss=5)	48.39	71.89	16.76	16.42	86.47	89.42	4.90	35.05	0.06	0.07	56.34	73.32	47.03	49.48	1.90	3.27	100.0	100.0
WaNet (s=0.5, k=8, ss=8)	72.40	65.33	14.44	15.81	82.57	90.19	5.06	24.67	0.05	0.14	66.92	89.74	47.32	46.23	3.72	3.79	97.07	100.0

Table A.7 presents a complete overview of the results of Experiment 4, which focuses on pre and post-query poisoning in AL. The results are categorized by trigger type and poison rates.

Table A.7: Results of experiment 4 showcasing the final ASR of the model for the pre-query and post-query approaches. Results are categorized by trigger methods (Lira and WaNet) and poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		**GTSRB CNN**		**CIFAR10 CNN**	
	Final ASR		Final ASR		Final ASR	
	1%	5%	1%	5%	1%	5%
Prequery						
LIRA (eps=0.2)	86.78	62.22	53.74	85.61	81.47	79.41
WaNet (s=0.5, k =4)	98.60	99.91	62.81	89.61	50.95	75.10
Postquery						
LIRA (eps=0.2)	99.72	99.96	75.43	87.15	87.72	97.29
WaNet (s=0.5, k =4)	99.98	100.0	86.51	96.57	89.30	98.22

B

Tables of CAD Results

Tables B.1, B.2 and B.3 present the results of Experiment 1, evaluating the final (last cycle) clean accuracy drop (CAD) over different query sizes. The tables summarize the CAD for various models and datasets at different poison rates. The results are further divided by poison method and trigger type.

Table B.1: Results of experiment 1 showcasing the final CAD of the model with poison rates of 1% and 5% over a query size of 500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	-0.0011	-0.0001	0.0243	0.0268	-0.0025	0.0221	-0.0216	-0.0041	-0.0626	-0.0714	0.0012	0.0032	-0.0245	-0.0252	0.1988	0.1836	0.0044	0.0193
Certainty	-0.0008	-0.0002	0.0308	0.0406	0.0039	0.0369	-0.0027	0.1603	-0.0516	-0.0178	0.0138	0.1992	-0.0287	-0.0134	0.1893	0.2051	-0.0082	-0.0003
Uncertainty	-0.0008	0.0003	0.0307	0.0313	0.0039	0.0101	-0.0036	-0.0047	-0.0808	-0.0477	-0.0161	0.0145	-0.0254	-0.0051	0.1935	0.2210	0.0012	0.0296
Trigger Type																		
Specialized	-0.0009	-0.0005	0.0152	0.0239	0.0018	0.0154	-0.0097	0.1013	-0.0774	-0.073	-0.0003	0.0755	-0.0256	-0.0162	0.3380	0.3467	0.0026	0.0172
Big	-0.0006	0.0008	0.0248	0.0368	-0.0041	0.0262	-0.0086	-0.0055	-0.0581	-0.0716	-0.0051	0.0655	-0.0271	-0.0084	0.0165	0.0129	-0.0041	0.0172
Simple	-0.0012	-0.0003	0.0458	0.038	0.0076	0.0276	-0.0096	0.0557	-0.0595	0.0077	0.0043	0.076	-0.0259	-0.0192	0.0108	-0.0021	-0.0012	0.0138

Table B.2: Results of experiment 1 showcasing the final CAD of the model with poison rates of 1% and 5% over a query size of 1000 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	0.0017	0.0009	0.0149	0.0522	-0.014	-0.0029	0.0067	0.0003	-0.1064	-0.0866	-0.0085	-0.0068	0.0064	0.0042	0.109	0.1209	-0.0227	0.0105
Certainty	0.001	0.0016	0.0168	0.0423	0.0007	0.0573	0.0127	0.3271	-0.1246	0.1154	0.0328	0.2643	0.006	0.0128	0.1389	0.1382	-0.0086	-0.0306
Uncertainty	0.0012	0.0013	0.0255	0.0326	-0.0156	-0.0014	0.0007	-0.0011	-0.0689	0.0055	-0.0077	0.0015	0.0049	0.0202	0.1039	0.1156	-0.0156	-0.0013
Trigger Type																		
Specialized	0.0013	0.0013	0.0165	0.0321	-0.0145	0.0184	0.0157	0.1724	-0.0842	-0.0068	0.0156	0.0932	0.0029	0.0143	0.3297	0.3421	-0.024	-0.0049
Big	0.0016	0.0012	0.0338	0.0327	-0.0103	0.0142	-0.0045	-0.0021	-0.1084	-0.0118	0.0058	0.08	0.0078	0.0092	0.0018	0.0101	-0.0199	-0.0127
Simple	0.001	0.0013	0.0069	0.0623	-0.0041	0.0204	0.009	0.156	-0.1073	0.053	-0.0049	0.0859	0.0066	0.0137	-0.0051	0.0055	-0.003	-0.0038

Table B.3: Results of experiment 1 showcasing the final CAD of the model with poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Poison Method																		
Random	-0.0002	0.001	0.0450	0.0207	0.0002	0.0189	0.0041	0.0031	0.0851	0.1307	0.0137	0.0208	-0.0032	-0.0003	0.1184	0.0721	0.0211	0.0235
Certainty	-0.001	0.0006	0.0363	0.0350	0.0391	0.0899	0.0948	0.1824	0.1449	0.6631	0.1078	0.3199	-0.0015	0.0140	0.0714	0.0872	-0.0044	0.0142
Uncertainty	-0.0001	-0.0003	0.0315	0.0428	-0.0104	0.0047	0.0061	0.0027	0.0612	0.0551	0.0071	0.0113	0.0018	0.0158	0.0835	0.0948	0.0084	0.0285
Trigger Type																		
Specialized	-0.0006	-0.0003	0.0383	0.0249	0.0064	0.0295	0.0482	0.0828	0.1049	0.2850	0.0335	0.1157	-0.0013	0.0117	0.2667	0.2534	-0.0004	0.0265
Big	-0.0001	0.0011	0.0336	0.0353	0.0160	0.0428	0.0045	-0.0001	0.0908	0.2487	0.0624	0.1140	0.0020	0.0112	-0.0019	0.0035	0.0140	0.0040
Simple	-0.0006	0.0005	0.0409	0.0383	0.0065	0.0412	0.0523	0.1056	0.0955	0.3153	0.0327	0.1222	-0.0036	0.0067	0.0083	-0.0029	0.0115	0.0357

Table B.4 and B.5 present the final (last cycle) CAD results of Experiment 2 over different query sizes. In this experiment the subtrigger division and parameter progression methods were tested. The results are categorized into three groups: Global, Subtrigger, and Progressive poisoning methods.

Table B.4: Results of experiment 2 showcasing the final CAD of the model with poison rates of 1% and 5% over a query size of 1000 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Global																		
LIRA (eps=0.2)	-0.001	-0.0001	0.0203	0.0486	-0.0153	-0.0092	0.0005	-0.0009	0.0172	0.0321	-0.0131	0.0043	0.0058	0.0061	0.006	-0.0053	0.017	0.009
WaNet (s=0.5, k=8)	-0.0009	0.0	0.0047	0.0052	-0.0134	-0.0108	-0.0001	-0.0005	-0.0254	0.0071	-0.0002	-0.0007	0.0058	0.0152	0.0008	-0.0007	-0.0231	0.0072
Subtrigger																		
LIRA (eps=0.1, ss=5)	-0.0014	0.0011	0.0295	0.0112	-0.0112	-0.0172	0.0007	-0.0003	-0.0061	0.0135	-0.0827	-0.0905	0.0024	0.0073	0.0155	0.0278	0.037	0.0389
LIRA (eps=0.2, ss=5)	-0.0015	-0.0003	0.035	0.01	-0.0079	-0.0065	-0.0003	-0.0009	0.0563	0.0389	-0.0916	-0.09	0.0009	-0.0018	0.0188	0.0055	0.0126	0.0207
LIRA (eps=0.1, ss=8)	-0.0019	0.0015	0.0423	0.0375	-0.0099	-0.0166	-0.0013	-0.0007	0.0182	0.0089	-0.1364	-0.0816	0.0068	0.0027	0.0238	0.0285	0.0263	0.0246
LIRA (eps=0.2, ss=8)	-0.0002	0.0003	0.0092	0.042	0.0006	-0.0017	-0.0014	-0.0001	-0.0299	0.0228	-0.097	-0.0789	0.0053	0.0147	0.0158	-0.0031	0.0027	0.0219
WaNet (s=0.2, k=4, ss=5)	-0.0008	-0.0022	0.0404	0.0369	-0.0145	-0.0017	-0.0021	-0.0006	-0.0363	0.0185	-0.0102	-0.024	0.0023	0.0026	-0.0023	-0.0081	0.0253	-0.0002
WaNet (s=0.2, k=4, ss=8)	0.0003	-0.0009	0.0234	0.0421	-0.014	0.0018	-0.0013	-0.0006	-0.0248	0.0518	0.01	0.0019	0.0054	0.0032	-0.0025	0.0064	0.0217	0.0113
WaNet (s=0.2, k=8, ss=5)	-0.0017	-0.0002	-0.0112	0.0019	-0.0175	-0.001	-0.0002	0.0001	0.0151	0.0027	-0.0081	-0.0097	0.0042	0.0072	0.0372	-0.009	-0.0084	-0.0109
WaNet (s=0.2, k=8, ss=8)	-0.0019	-0.0006	0.0053	0.0347	-0.0128	-0.005	0.0003	-0.001	-0.0209	0.039	-0.0117	-0.0108	0.0018	0.0088	-0.0073	-0.0013	-0.0103	-0.0322
WaNet (s=0.5, k=4, ss=5)	-0.0004	-0.001	-0.0178	0.0259	-0.0271	-0.0052	0.0001	-0.001	0.0363	-0.005	-0.0055	-0.0135	0.002	0.0039	-0.0067	0.0028	0.0105	0.0157
WaNet (s=0.5, k=4, ss=8)	-0.0002	-0.0008	0.0056	0.0291	-0.0321	-0.0072	-0.0017	0.0007	-0.0139	-0.0364	-0.0106	-0.0199	0.0032	0.0056	0.0028	-0.0014	-0.0261	0.0231
WaNet (s=0.5, k=8, ss=5)	-0.0007	0.0008	-0.0072	0.058	-0.0185	-0.0027	-0.001	0.0007	-0.0221	-0.0266	-0.0118	-0.014	0.0017	0.0122	0.0003	0.0066	-0.0248	-0.0012
WaNet (s=0.5, k=8, ss=8)	-0.0012	0.0004	0.0277	0.0214	-0.0079	0.0013	-0.0008	-0.0009	0.045	0.002	0.0075	-0.0049	0.0047	0.0159	0.0378	0.0122	-0.0115	-0.0152
Progressive																		
LIRA (eps=0.2)	-0.001	-0.0001	0.0203	0.0486	-0.0153	-0.0092	0.0005	-0.0009	0.0172	0.0321	-0.0131	0.0043	0.0058	0.0061	0.006	-0.0053	0.017	0.009
WaNet (s=0.5, k=8)	-0.0009	0.0	0.0047	0.0052	-0.0134	-0.0108	-0.0001	-0.0005	-0.0254	0.0071	-0.0002	-0.0007	0.0058	0.0152	0.0008	-0.0007	-0.0231	0.0072

Table B.5: Results of experiment 2 showcasing the final CAD of the model with poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%	1%	5%
Global																		
LIRA (eps=0.2)	-0.0011	-0.0006	0.0347	0.0303	0.0065	0.0157	-0.0008	-0.0006	0.0505	0.0421	0.0081	0.0172	0.0016	-0.0028	-0.0015	-0.0038	0.0065	0.0157
WaNet (s=0.5, k=8)	-0.0004	-0.0002	0.0261	0.0013	-0.0009	0.0076	0.0004	0.001	0.0339	0.0348	0.0213	0.0281	-0.0006	0.0135	0.0105	-0.0132	-0.0009	0.0076
Subtrigger																		
LIRA (eps=0.1, ss=5)	-0.0005	0.0007	0.0423	0.0506	0.0035	0.0081	-0.0	-0.0	0.0502	0.0847	-0.0312	-0.0474	-0.0062	-0.002	0.0145	0.0141	-0.0395	0.0248
LIRA (eps=0.2, ss=5)	-0.001	-0.0001	0.0293	0.0253	0.0014	0.0181	-0.0004	-0.0007	0.0962	0.0676	-0.036	-0.0529	0.0024	0.0024	-0.0045	-0.0088	0.0062	0.0177
LIRA (eps=0.1, ss=8)	-0.0022	-0.0003	0.0318	0.0342	0.0041	0.0159	0.0002	0.0006	0.0484	0.0721	-0.0522	-0.0246	-0.0072	-0.004	-0.013	-0.0029	0.0367	0.006
LIRA (eps=0.2, ss=8)	-0.0004	-0.0004	0.0234	0.0452	0.0123	0.0298	0.0006	-0.0004	0.0444	0.1167	-0.0493	-0.0113	-0.0014	0.0037	0.0019	0.0307	-0.0183	0.0141
WaNet (s=0.2, k=4, ss=5)	-0.0001	-0.0005	0.0267	0.0249	0.0211	0.0278	0.0003	0.0013	0.0547	0.082	0.0128	0.016	-0.002	-0.0028	0.0324	-0.0136	-0.0197	-0.0195
WaNet (s=0.2, k=4, ss=8)	-0.0004	-0.0002	0.0357	0.0322	0.0003	0.0132	-0.0005	-0.002	0.0211	0.0332	-0.0019	0.0105	-0.0071	-0.0061	-0.0155	0.0056	-0.0193	0.0046
WaNet (s=0.2, k=8, ss=5)	-0.0005	0.001	0.0106	0.0338	0.0013	0.0195	0.0004	-0.0011	0.0338	0.0439	0.0167	0.0078	-0.0002	-0.0028	-0.017	-0.0034	-0.0034	0.0428
WaNet (s=0.2, k=8, ss=8)	-0.0008	0.0009	0.0515	0.0257	0.0055	0.0031	-0.0009	-0.0006	0.0563	0.0394	0.0059	0.015	-0.0003	0.0048	-0.0234	-0.0204	-0.0067	0.0029
WaNet (s=0.5, k=4, ss=5)	-0.0015	-0.0005	0.0123	0.0469	-0.0057	0.0141	-0.0005	0.0035	0.0387	0.0342	0.0161	0.0126	-0.007	-0.0037	-0.0145	-0.0143	-0.0483	0.0291
WaNet (s=0.5, k=4, ss=8)	-0.0003	-0.0006	0.0054	0.0115	-0.0059	0.0118	-0.0005	0.0006	0.0278	0.0815	0.0122	0.0144	-0.004	-0.0052	-0.0196	-0.0203	0.0116	-0.0152
WaNet (s=0.5, k=8, ss=5)	0.0007	-0.0002	0.0183	0.0303	0.0164	0.0174	-0.0001	0.0006	0.0317	0.0532	0.0052	-0.0053	-0.0004	-0.0029	-0.0094	0.0189	-0.0202	0.0056
WaNet (s=0.5, k=8, ss=8)	-0.0017	-0.0015	0.0163	0.0224	-0.005	0.0197	-0.0008	0.0002	0.0412	0.0272	0.0202	0.0147	0.0019	-0.0031	0.0543	0.0031	0.004	0.0109
Progressive																		
LIRA (eps=0.2)	-0.0011	-0.0006	0.0347	0.0303	0.0065	0.0157	-0.0008	-0.0006	0.0505	0.0421	0.0081	0.0172	0.0016	-0.0028	-0.0015	-0.0038	0.0249	0.0077
WaNet (s=0.5, k=8)	-0.0004	-0.0002	0.0261	0.0013	-0.0009	0.0076	0.0004	0.001	0.0339	0.0348	0.0213	0.0281	-0.0006	0.0135	0.0105	-0.0132	-0.0006	0.0135

Table B.6 presents a comprehensive overview of Experiment 3, in which clean-label attacks are tested.

In the table, the results are categorized by trigger type and all the important parameters for the Lira and WaNet triggers are included.

Table B.6: Results of experiment 3 showcasing the final CAD of the model with poison rates of 2% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN		MNIST ResNet		GTSRB ResNet		CIFAR10 ResNet		MNIST Inception		GTSRB Inception		CIFAR10 Inception	
	Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD		Final CAD	
	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%	2%	5%
Global																		
LIRA (eps=0.2)	0.0059	0.0085	-0.023	-0.0531	-0.018	-0.0162	0.0006	0.0030	-0.0175	-0.0175	-0.0057	0.0221	-0.0066	0.0024	-0.0015	-0.0038	-0.018	-0.0162
WaNet (s=0.5, k=8)	0.0008	0.0107	-0.031	-0.0299	-0.0177	-0.0178	0.0055	0.0154	0.0220	0.0160	0.0053	0.0154	0.0030	-0.0059	0.0336	0.0336	-0.0177	-0.0178
Subtrigger																		
LIRA (eps=0.1, ss=5)	0.0027	0.0103	-0.0385	-0.0276	-0.018	-0.0061	0.0006	0.0128	-0.0183	-0.0224	0.0090	0.0114	-0.0011	-0.0045	0.1058	0.0168	-0.018	-0.0061
LIRA (eps=0.2, ss=5)	0.0041	0.0104	-0.0264	-0.0377	-0.0048	-0.0066	0.0157	0.0140	0.0200	0.0275	0.0028	0.0125	-0.0001	0.0047	0.0284	0.0128	-0.0048	-0.0066
LIRA (eps=0.1, ss=8)	0.0035	0.0096	-0.0316	0.0020	-0.016	-0.0108	0.0053	0.0048	-0.0073	0.0014	-0.0014	0.0195	-0.0026	-0.0007	-0.013	-0.0029	-0.016	-0.0108
LIRA (eps=0.2, ss=8)	0.0027	0.0067	-0.0083	0.0048	-0.0242	-0.0056	-0.0011	0.0028	-0.0124	-0.0325	-0.0022	0.0036	-0.0062	-0.0032	0.0330	0.0453	-0.0242	-0.0056
WaNet (s=0.2, k=4, ss=5)	0.0030	0.0105	-0.0356	-0.0537	-0.0070	0.0047	-0.0071	0.0140	-0.0107	0.0064	0.0107	0.0207	-0.0051	-0.0054	0.0050	0.0355	-0.0070	0.0047
WaNet (s=0.2, k=4, ss=8)	0.0040	0.0081	-0.0162	-0.0176	-0.0169	-0.0048	-0.0017	-0.0062	0.0183	0.0078	0.0042	0.0125	-0.0003	0.0010	-0.0237	0.0119	-0.0169	-0.0048
WaNet (s=0.2, k=8, ss=5)	0.0015	0.0099	-0.0145	-0.0236	-0.0214	-0.014	-0.0062	0.0128	0.0161	0.0395	-0.0187	0.0291	-0.0047	-0.0001	-0.036	0.0045	-0.0214	-0.014
WaNet (s=0.2, k=8, ss=8)	0.0035	0.0132	-0.0366	-0.0051	-0.0145	0.0059	0.0068	0.0094	0.0019	-0.0033	0.0163	0.0069	-0.0002	0.0084	0.0451	0.0354	0.0332	0.0344
WaNet (s=0.5, k=4, ss=5)	0.0030	0.0102	-0.0583	-0.0418	-0.017	-0.0021	0.0074	0.0086	-0.0103	-0.0158	0.0176	0.0088	-0.0038	0.0016	-0.0165	-0.0111	-0.0170	-0.0021
WaNet (s=0.5, k=4, ss=8)	-0.0011	0.0104	-0.0454	-0.0284	-0.0208	-0.0015	-0.0012	0.0016	0.0042	0.0143	-0.0041	-0.0049	-0.0022	-0.0047	-0.0055	-0.0235	-0.0208	-0.0015
WaNet (s=0.5, k=8, ss=5)	0.0024	0.0103	-0.0096	-0.0389	-0.0077	-0.0062	0.0056	0.0135	-0.0139	-0.0185	-0.0122	0.0257	-0.0011	-0.0036	0.0111	-0.0217	-0.0077	-0.0062
WaNet (s=0.5, k=8, ss=8)	0.0026	0.0075	-0.0334	-0.0146	-0.0026	-0.001	-0.0028	0.0035	0.0285	0.0014	-0.0052	-0.0085	-0.0005	-0.0021	0.0121	0.0121	0.0047	0.0041

Table B.7 presents a complete overview of the results of Experiment 4, which focuses on pre and post-query poisoning in AL. The results are categorized by trigger type and poison rates.

Table B.7: Results of experiment 4 showcasing the final CAD of the model for the pre-query and post-query approaches. Results are categorized by trigger methods (LIRA and WaNet) and poison rates of 1% and 5% over a query size of 1500 samples.

	MNIST CNN		GTSRB CNN		CIFAR10 CNN	
	Final CAD		Final CAD		Final CAD	
	1%	5%	1%	5%	1%	5%
Prequery						
LIRA (eps=0.2)	0.0158	0.0256	0.2075	0.3203	0.0448	0.0585
WaNet (s=0.5, k=4)	0.0129	0.0125	0.1645	0.2218	0.0274	0.0837
Postquery						
LIRA (eps=0.2)	0.0066	0.0087	0.1110	0.1353	-0.0053	0.0092
WaNet (s=0.5, k=4)	0.0052	0.0074	0.0995	0.0922	0.0011	0.0146