CONFIDENT DATA-FREE MODEL STEALING FOR BLACK-BOX ADVERSARIAL ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep machine learning models are increasingly deployed in the wild, subject to adversarial attacks. White-box model attacks assume to have full knowledge of the deployed target models, whereas the black-box models need to infer the target model via curated labeled dataset or sending abundant data queries and launch attacks. The challenge of black-box lies in how to acquire data for querying target models and effectively learn the substitute model using a minimum number of query data, which can be real or synthetic. In this paper, we propose an effective and confident data-free black-box attack, CODFE, which steals target model by queries of synthetically generated data. The core of our attack is a model stealing optimization consisting of two collaborating models (i) substitute model which imitates the target model and (ii) generator which generates most representative data to maximize the confidence of substitute model. We propose a novel training procedure that steers the synthesizing direction based on the confidence of substitute model and exploit a given set of synthetically generated images by multiple training iterations. We show the theoretical convergence of the proposed model stealing optimization and empirically evaluate its success rate on three datasets. Our results show that the accuracy of substitute models and attack success rate can be up to 56% and 34% higher than the state of the art data-free black-box attacks.

1 INTRODUCTION

Emerging intelligent services, such as Google translate and optical character recognition (Google, 2016), are increasingly powered by trained deep models. Users can easily access those services through public APIs by sending queries, such as getting inference results of classifying users' images. While such an open access to deployed models greatly ease users' experience, it also opens up various vulnerability issues to adversarial attacks. Malicious perturbation on the query images leads the deployed models to the misclassification outcomes (Kurakin et al., 2017; Carlini & Wagner, 2017; Madry et al., 2018), and further makes wrong decision, e.g., self-driving car to misjudge the stop sign with sticks (Papernot et al., 2017).

Adversarial attacks can be either white-box (Ebrahimi et al., 2018; Fang et al., 2019; Truong et al., 2021) or black-box (Ilyas et al., 2018), depending on whether the malicious users have the knowledge of the parameters of deployed models, e.g., network architecture and weights. White-box attacks, e.g., FGSM (Goodfellow et al., 2015) and PGD (Madry et al., 2018), adds perturbation noise on query images based on their gradients of deployed models, assuming a full knowledge of such target models[1]. In contrast, black-box attacks generate adversarial examples by inferring the target models through abundant labeled dataset (Hinton et al., 2015) or data queries (Zhou et al., 2020; Krishna et al., 2020). For each query, malicious users send in real images and retrieves their labels from the target model. Such query-label pairs are then used to train substitute models (Wallace et al., 2020; Krishna et al., 2020), which aim to imitate and steal the target models eventually. Though the prior art has shed the light on stealing target models via the substitute models, it is a standing challenge that how to effectively learn a substitute model using a minimum number of data queries. Moreover, data from privacy- and business-sensitive domains, e.g., bank, and medical sectors, may have limited accessibility to attackers and further exacerbate the difficulty of black-box attacks and model stealing in data-free scenarios.

[1]We interchangeably use terms of target and deployed models.

Recognizing the limited availability of real data for adversarial queries, (Zhou et al., 2020) proposed a data-free model stealing method, DaST, to learn substitute models needed in black-box attacks. DaST adopts the principle of generative adversarial networks (GANs) and has a structure of multiple generators and one substitute model. The generators are trained with the substitute models in a competition game. Although DaST shows the promising results of imitating the target model, it suffers from (i) complex network structure that relies on large amount of query data and long training, and (ii) unstable training due to a large random generating space of synthetic data. Further, it is no mean feat to train GANs for generating adversarial examples even when real data is available (Farnia & Ozdaglar, 2020). Thus open research questions regarding data-free model stealing persist. First, what is the effective network structure that can capture the target models using a minimum number of queries? Secondly, is it necessary to explore the entire generating space? Can one selectively generate examples by exploiting seeds and then lead to better substitute models?

In this paper, we consider a stringent adversarial scenario, a black-box model with no access to the real data for querying the target model. We propose a confident data-free model stealing algorithm, CODFE, which learns a substitute model via querying the target model through the synthetic data from the designed generator. CODFE then generates the adversarial examples by applying the state-of-the-art gradient-based attacks (Goodfellow et al., 2015; Kurakin et al., 2017; Madry et al., 2018) through the trained substitute models. We aim to efficiently imitate the target model through a small number of queries and architect CODFE as two collaborating light-weight models, the generator and substitute model. To efficiently steer the synthetic generation toward an accurate substitute model, we minimize the cross entropy of the substitute model and maximize the confidence of the generated examples. Moreover, to improve the convergence of CODFE, we exploits every set of generated data by multiple iterations of training/updates on the generator and substitute model. We theoretically show the convergence of the proposed stealing process, showing its effectiveness to learn substitute models. Our empirical evaluation on three datasets under both untargeted and targeted attacks shows that CODFE can efficiently steal a target model using a small number of queries and successfully generate adversarial examples using the substitute model. Compared to the state-of-the-art data-free black-box model using three benchmark datasets, CODFE achieves 14%–56% higher accuracy in substitute models using a 37%–95% lower number of synthetic data queries and then leads to higher attack success rate up to 34%.

The contribution of this paper is three fold: (i) efficient confident data-free model stealing framework with a minimum number of queries (Section 3.1), (ii) a theoretical guarantee on the convergence of imitation process (Section 3.3), and (iii) effective and stable black-box attacks on multiple datasets and adversarial scenarios (Section 4).

2 RELATED WORK

Data-free black-box adversarial attacks contains two phases: *stealing process* to imitate the target model and *attacking process* which adds adversarial perturbation so as to mislead the target model. Thus, we organize the related studies according to these two phases.

Model stealing. The goal of model stealing is to imitate the knowledge from a deployed model (target model) and then obtain a highly similar substitute model (Krishna et al., 2020; Jagielski et al., 2020; Chandrasekaran et al., 2020; Zhou et al., 2020). A successful substitute model is able to obtain the implicit mapping function (or knowledge, in high level) of the target model by different network structures (Kariyappa et al., 2021; Orekondy et al., 2019a). There are two types of model stealing methods depending on whether the attackers are able to access the real training data (or part of it). In the case when real data is available, knowledge distilling (Hinton et al., 2015) extracts the knowledge of the target model. The key idea is that the substitute model is trained by class probabilities as well as part of the dataset. The class probabilities (so called "soft targets") are produced by inferring the target model. By this means, the knowledge of the big target model can be transferred to a small substitute model in order to reduce inference cost. When real data is not available for training, attackers can only imitates the target model through querying synthetic examples. (Krishna et al., 2020) uses randomly generated sentences for querying NLP models, but lacks generality beyond NLP tasks. DaST (Zhou et al., 2020) employs the GANs model to generate images by using a large number of queries and experiences the limitation of training instability of GANs. Moreover, when specifically considering classification as the learning task, the querying output of can either be label

or classification probability. Label-query stealing (Jagielski et al., 2020; Chandrasekaran et al., 2020; Zhou et al., 2020) trains the substitute model using the query-label pairs. In query-logit(probability) stealing (Orekondy et al., 2019b), attackers can make use of some unlabelled normal examples (Juuti et al., 2019). Here, we explicitly address the data-free model stealing by a novel generative model that requires a low number of synthetic query examples in scenarios of query-label and query-logit.

Adversarial attacks. Adversarial attacks aim to generate visual indistinguishable adversarial examples to fool target models. Attacks can be under white-box setting (Dong et al., 2018; Goodfellow et al., 2015; Kurakin et al., 2017; Papernot et al., 2016; Carlini & Wagner, 2017; Szegedy et al., 2014), or black-box attack settings (Ilyas et al., 2018). As the crucial difference among them is that white-box attackers can access the target model, most white-box attack methods can be applied for black-box setting using a substitute model. Thus, in the following, we assume there exists a substitute model so that it is unnecessary to discuss white-box or black-box attacks separately. Adversarial attacks can be summarized into two categories: optimization- and gradient-based. The optimization-based methods (Szegedy et al., 2014; Carlini & Wagner, 2017) minimize the distance between the normal and adversarial examples while considering the misclassification of adversarial examples (Szegedy et al., 2014), or find perturbation which minimizes the distance of the examples (Carlini & Wagner, 2017). The gradient-based approaches (Dong et al., 2018; Goodfellow et al., 2015; Kurakin et al., 2017; Papernot et al., 2016) are extensively explored. FGSM (Goodfellow et al., 2015) is an one-step attack that generates the noise according to the gradient signs of the examples and adds the noise into the normal examples to obtain the corresponding adversarial examples. BIM (Kurakin et al., 2017) is an enhanced iterative version of FGSM while MI-FGSM (Dong et al., 2018) elaborates the momentum on BIM for higher transferability. Till date, the strongest adversarial attack is PGD (Madry et al., 2018; Croce & Hein, 2020), state-of-the-art iterative version of FGSM, initializes the example at a random point and does random restarts in each iteration. The substitute model obtained through the proposed modeling stealing optimizaiton is compatible and applicable with these gradient-based adversarial methods.

3 METHODOLOGY

3.1 ADVERSARIAL MODELS

Here, a target model $\mathcal{T}$ that conducts classification tasks is deployed and accessible for data query, e.g., inferencing class labels for query data. Given an input $x \in X$, $\mathcal{T}(x) = y$, where y is the inference output[2] of model $\mathcal{T}$. There are two types of data to $\mathcal{T}$: benign examples and adversarial examples, denoted by $\bar{x}$ and $\widehat{x}$, respectively. Let $\bar{y}$ be the inference label of $\mathcal{T}$ for $\bar{x}$. Adversarial attacks aims to generate a visual indistinguishable example $\widehat{x} = \bar{x} + \epsilon$ to fool the target model, where ϵ is the perturbation of the normal example $\bar{x}$. $\widehat{x}$ is the corresponding adversarial example. There are two types of attacks: untargeted and targeted, where the former misleads the target model to misclassify the adversarial example, and the later leads the target model to a particular type of classification. In an untargeted attack, attackers tries to misleads the target model to misclassify the adversarial example by minimizing $\|\epsilon\|$ such that $\mathcal{T}(\widehat{x}) \neq \bar{y}$. In the *targeted* attacks, adversarial attack tries to make the target model to classify the adversarial example to a particular label l, i.e., y_l, $\mathcal{T}(\widehat{x}) = y_l$. Attackers take a benign example and then adds the perturbation to the gradient of $\bar{x}$ in the target model by the function $\epsilon = P(\nabla_{\bar{x}}\mathcal{T}(\bar{x}))$, e.g., FGSM (Goodfellow et al., 2015), where P is the perturbation function over the gradient. Thus, for both targeted and untargeted attacks, generating ϵ for $\bar{x}$ requires the knowledge of target model which can be either known or unknown to attackers, corresponding to white or black-box attacks respectively. In white-box settings, attackers can directly access $\mathcal{T}$ to generate adversarial examples by adding ϵ. However, in black-box attacks, as the attackers can't access the target model, a substitute model $\mathcal{S}$ that imitates the target model is required to generate adversarial examples,.

Here,we particularly consider black-box attacks, where $\mathcal{T}$ is unknown to attackers. Moreover, we assume that attackers do not have any real-world dataset to train the substitute model. In this black-box setting, though attackers can't access the target model, they can query it and get the inference results. We consider two types of inference results: the label-only and the probability-only. In the label-only scenario, attackers can get the corresponding inference labels of the query examples from

[2] $\mathcal{T}(x)$ is label in label-only scenario, and is probability vector for probability-only scenario. In this section we refer it as label for simplifications.

$\mathcal{T}$. While in the probability-only scenario, the output probability of $\mathcal{T}$ can be obtained. The main difference for label-only and probability-only is that the label-only feedback is less informative, and thus more difficult to attack.

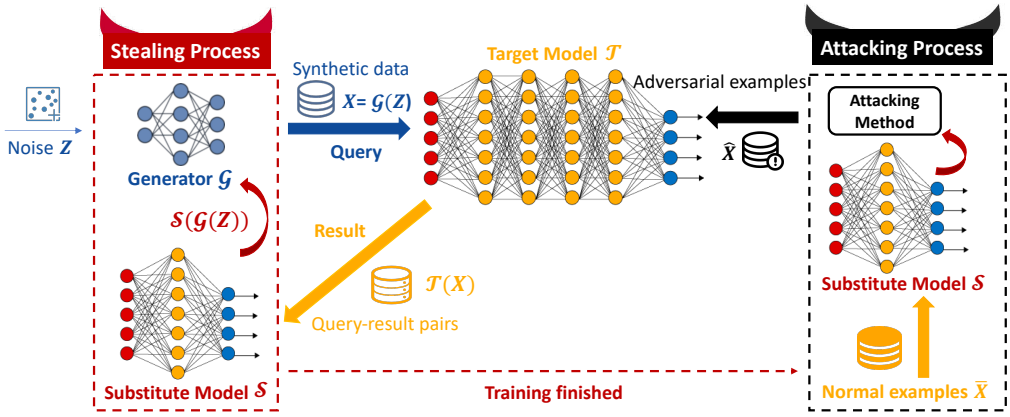


Figure 1: CoDFe framework: data-free model stealing process and attacking process.

3.2 Confident Data Free Attacks (CoDFe)

Our goal is to learn a substitute model that steals the knowledge of the target model with no real data and via a minimum number of queries. Then, we can use the substitute models to launch gradient based attacks, e.g.,FGSM, BIM and PGD .

The architecture of CoDFe is shown in Figure 1, containing a stealing stealing process and an attacking process. For stealing process, we first generate synthetic examples $X = \mathcal{G}(Z)$ from noises Z to query $\mathcal{T}$ and get the output $\mathcal{T}(X)$. Next, X associated with $\mathcal{T}(X)$ is feed into $\mathcal{S}$ for training. The output of the substitute model $\mathcal{S}(\mathcal{G}(Z))$ is then used to update $\mathcal{G}$ so as to generate better synthetic examples. When the stealing is finished, normal examples $\bar{X}$ and the stolen $\mathcal{S}$ can be combined jointly to generate adversarial examples $\widehat{X}$ using adversarial attack methods. We note that the proposed modeling stealing approaches are compatible with existing white-box attack approaches, which rely on the information of target models.

Notations: In this data-free approach, a set of noise vectors $Z = \{z_1, ..., z_M\}$ where M is the number of noise vectors, is generated as the input data of $\mathcal{G}$. $X = \{x_i \mid x_i = \mathcal{G}(z_i), i \in [M]\}$ denotes the corresponding synthetic examples generated by the generator. X is used to query the target model so that we can obtain the query results $\mathcal{T}(X) = \{\mathcal{T}(x_i) \mid i \in [M]\}$. In the probability-only scenario $\mathcal{T}(x)$ is the output probability vector of the input example x. While in the label-only scenario, $\mathcal{T}(x)$ is the one-hot vector of the predicted labels. In this paper, we assume $\mathcal{T}$ is a model of common classification task, with N number of classes. We use $\mathcal{T}_j(x)$ to denote the j-th ($j \in N$) element of the output.

The objective of the substitute model: Since $\mathcal{S}$ is a substitute of $\mathcal{T}$, their outputs shall be as consistent as possible. Inspired by knowledge distillation (Hinton et al., 2015), $\mathcal{S}$ imitates the outputs of $\mathcal{T}$ through the cross entropy loss. According to the definition of $\mathcal{T}(x)$, the following loss function (1) can be applied to both the probability-only scenario and the label-only scenario.

$$\mathcal{L}_{\mathcal{S}} = \mathrm{CE}(\mathcal{T}(X), \mathcal{S}(X)) = -\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{T}_j(x_i)\log \mathcal{S}_j(x_i), \tag{1}$$

where CE denotes cross entropy. The objective is to learn the generalization ability of $\mathcal{T}$. Using the cross entropy loss, $\mathcal{S}$ can generalize in a similar way as $\mathcal{T}$. Thus we can say $\mathcal{S}$ gets the knowledge transferred from $\mathcal{T}$ and $\mathcal{S}$ can be used to represent $\mathcal{T}$ when generating adversarial examples in the black-box setting.

The objective of the generator: The generator $\mathcal{G}$ is responsible for generating synthetic examples to query the target model. The key challenge is to generate representative synthetic examples. If

we use a real-world example $\bar{x}$ to query $\mathcal{T}$ and assume that $\mathcal{T}$ is well trained, the confidence of the output $\mathcal{T}(\bar{x})$ is expected to be high where the confidence is defined as the biggest element of the output $\{\mathcal{T}_k(\bar{x}) \mid \forall_j : \mathcal{T}_j(\bar{x}) \leq \mathcal{T}_k(\bar{x})\}$. In our attack setting, we only have synthetic examples to query $\mathcal{T}$. Our goal is to simulate the queries of using real-world examples. Given a synthetic example $x = \mathcal{G}(z)$, where z is a noise vector. If the confidence of $\mathcal{T}(x)$ is high, x is be regarded as a representative example (Li & Sethi, 2006; **?**). Consequently, the objective of $\mathcal{G}$ is to generate x that maximizes the confidence of $\mathcal{T}(x)$. However, we can't straightforwardly make use of $\arg\max_{\theta_g} \{\mathcal{T}_k(\mathcal{G}(z)) \mid \forall_j : \mathcal{T}_j(\mathcal{G}(z)) \leq \mathcal{T}_k(\mathcal{G}(z))\}$ to be the optimization goal, where θ_g is the model parameters of $\mathcal{G}$. The reason is that the backpropagation of this optimization goal requires the gradient information of $\mathcal{T}$, but the target model is not accessible in our setting. Since $\mathcal{S}$ mimics the outputs of $\mathcal{T}$ in the training process, we use $\mathcal{S}(x)$ to approximate $\mathcal{T}(x)$. Thus the loss function of $\mathcal{G}$ can be defined as Equation (2).

$$\mathcal{L}_{\mathcal{G}} = -\frac{1}{M}\sum_{i=1}^{M}\{\log \mathcal{S}_k(\mathcal{G}(z_i)) \mid \forall_j : \mathcal{S}_j(\mathcal{G}(z_i)) \leq \mathcal{S}_k(\mathcal{G}(z_i))\}, \tag{2}$$

where $\mathcal{S}_j(\mathcal{G}(z))$ represents the j-th element of the output $\mathcal{S}(\mathcal{G}(z))$.

Stealing algorithm: In our approach, $\mathcal{S}$ and $\mathcal{G}$ are trained together in the stealing procedure shown in Algorithm 1. We generate a set of random noise vectors Z in the beginning. Then we have multiple rounds to iteratively train $\mathcal{S}$ and $\mathcal{G}$. Specifically, the noise vector is sent as an input into $\mathcal{G}$ for generating a fixed set of synthetic examples X at the beginning of each round. We then use the synthetic examples to query $\mathcal{T}$ and obtain the query-result pairs $(X, \mathcal{T}(X))$. The pairs are not just used to update one step (e.g., one SGD step) of $\mathcal{S}$. Instead, they can be applied to train $\mathcal{S}$ for multiple iterations within a round. We apply mini-batch training. Then a local optima of $\mathcal{S}$ for $(X, \mathcal{T}(X))$ can be found, which is good for the **stability** of the convergence process and we also empirically evaluate it in the experiment section. In each round, $\mathcal{S}$ is updated before updating $\mathcal{G}$, because in the loss $\mathcal{L}_{\mathcal{G}}$ we use $\mathcal{S}(x)$ to be the approximation of $\mathcal{T}(x)$. Thus, $\mathcal{S}$ should be firstly updated to imitate the output of $\mathcal{T}$. We explicitly exploit the input noise set Z for multiple rounds. In each round, the generator $\mathcal{G}$ is trained to obtain new synthetic examples which are more representative to query the target model in the next round. After the model stealing training, $\mathcal{S}$ is applied to produce adversarial examples and launch attacks on $\mathcal{T}$.

Algorithm 1: CODFE

Let θ_s be the model parameters of $\mathcal{S}$ and θ_g be the model parameters of $\mathcal{G}$
Generate a noise set $Z = \{z\}$
for *number of rounds* **do**
    Generate examples $X = \mathcal{G}(Z)$ using the generator
    Get the query results $\mathcal{T}(X)$ from the target model
    /* train the substitute model */
    for *number of iterations* **do**
        Sample a batch $\tilde{X}$ from X
        Get the corresponding $\mathcal{T}(\tilde{X})$
        Get the output of the substitute model $\mathcal{S}(\tilde{X})$
        Update θ_s by minimizing $\mathcal{L}_{\mathcal{S}}$ using first order stochastic optimization
    /* train the generator */
    for *number of iterations* **do**
        Sample a batch $\tilde{Z}$ from Z
        Get the output of the substitute model $\mathcal{S}(\mathcal{G}(\tilde{Z}))$
        Update θ_g by minimizing $\mathcal{L}_{\mathcal{G}}$ using first order stochastic optimization

3.3 ANALYSIS

The cross entropy loss: In order to imitate the target model, the output of the substitute model needs to be as similar to the output of the target model as possible. The two models can be regarded

as two distributions. The most straightforward way to enforce their similarity is to minimize the Kullback-Leibler divergence (Kullback & Leibler, 1951) between them. Here, we make use of cross entropy to transfer the generalization ability of the target model to the substitute model (Hinton et al., 2015). In our black-box setting, the KL-divergence loss and the cross entropy loss can be regarded as equivalent loss functions for stochastic optimization. But cross entropy can avoid gradient vanishing in white-box cases (see the Appendix for more details).

Convergence: In Algorithm 1, $\mathcal{S}$ and $\mathcal{G}$ are updated iteratively in each round t. The algorithm proceeds by coordinate descent. Updating $\mathcal{S}$ minimizes $\mathcal{L}_{\mathcal{S}}$ while updating $\mathcal{G}$ also indirectly decreases the value of $\mathcal{L}_{\mathcal{S}}$ because it maximizes the confidence of the outputs. Let $\theta_s^{(t)}$ and $\theta_g^{(t)}$ be the model parameters of $\mathcal{S}$ and $\mathcal{G}$ after the training of round t. To analyze the convergence of Algorithm 1, we make the following assumptions according to the optimization objectives of $\mathcal{S}$ and $\mathcal{T}$. In round t, given a noise vector $z \in Z$, we assume that after the training of $\mathcal{S}$ (line 6-10), $\arg\max_i \mathcal{S}_i(\mathcal{G}(z;\theta_g^{(t-1)});\theta_s^{(t)}) = \arg\max_i \mathcal{T}_i(\mathcal{G}(z;\theta_g^{(t-1)}))$. And after the training of $\mathcal{G}$ (line 11-14), the confidence of $\mathcal{T}$'s output is higher. Then, we can have Theorem 1 that shows $\mathcal{L}_{\mathcal{S}}$ can converge in our proposed algorithm. We also verify the convergence performance of our algorithm in the experiment part.

Theorem 1. Given a noise vector $z \in Z$. Let $f\left(\theta_s^{(t)}\right) = \mathrm{CE}\left(\mathcal{T}\left(\mathcal{G}(z;\theta_g^{(t)})\right), \mathcal{S}\left(\mathcal{G}(z;\theta_g^{(t)});\theta_s^{(t)}\right)\right)$. Training the substitute model by Algorithm 1, we have $\lim_{t\to\infty} f(\theta_s^{(t)}) = \epsilon^*$, where $\epsilon^* \geq 0$. ***(Proof in Appendix B)***

4 EVALUATION

In this section, we comprehensively evaluate the attacking effectiveness and convergence performance of CODFE under various attacking scenarios and methods. As our main contributions lie on the model stealing, we stress the model stealing accuracy of CODFE compared to DaST (Zhou et al., 2020), the state of the art data-free model stealing approach. Last, we demonstrate that only a small number of queries is needed for learning substitute models and crafting adversarial examples.

4.1 EXPERIMENTAL SETUP

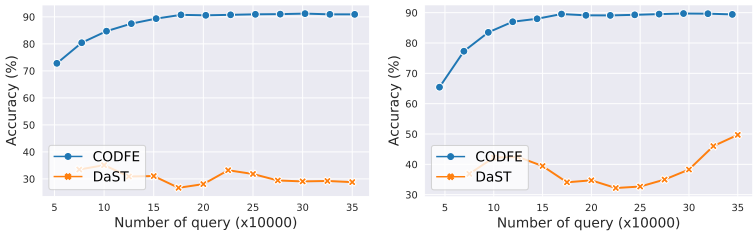
Dataset and Model Structure: We evaluate our proposed method on three datasets: MNIST (LeCun et al., 1998), Fashion-MNIST (Xiao et al., 2017) and CIFAR-10 (Krizhevsky et al., 2009). Note that for each dataset we use different network structures for target and substitute models as the attackers lack of the prior knowledge of the target structure. For MNIST, we utilize a lightweight CNN of four and three convolutional layers for target and substitute models, respectively. Both $\mathcal{S}$ and $\mathcal{T}$ structures of Fashion-MNIST are the same with MNIST. As for Cifar-10, the commonly adopted ResNet34 is used for $\mathcal{T}$ and we select CNN of four convolutional layers for $\mathcal{S}$ (as DaST) in the experiments.

Attacking Scenarios and Methods: Two types of scenarios are considered: **label-only**, where the attackers can access the output label only, and **probability-only**, where the accessible output is classification probabilities. We denote them as CODFE-L and CODFE-P respectively. Here we also assume that the attacks in both scenarios are free to query $\mathcal{T}$ unlimited times. To generate adversarial examples, we apply three existing attacking methods including **FGSM** (Goodfellow et al., 2015), **BIM** (Kurakin et al., 2017), and projected gradient descent (**PGD**) (Madry et al., 2018).

Evaluation Criteria: The goal of our confident data-free attack is to mislead $\mathcal{T}$ by two types of attacks. The **targeted attack** aims at leading $\mathcal{T}$ to output specific label (In the experiments, we use the second class as target for all three datasets.). However, the **untargeted attack** just aims at making $\mathcal{T}$ take wrong class label. The criteria to evaluate both types of attacks are attack success rate (**ASR**), denoted by n_{suc}/n_{all}, where n_{all} is the number of generated adversarial examples to fool $\mathcal{T}$, and n_{suc} is the number of successful attempts. For attacking efficiency, we use the number of queries to reach the maximum accuracy of substitute model.

4.2 MODEL STEALING PERFORMANCE

Model Stealing Accuracy. Here we evaluate the accuracy of model stealing results compared to baseline model of DaST on both label-only and probability-only scenarios in Figure (2). As DaST and CODFE apply different updating strategies and batch sizes, it is hard to uniform them by training iterations or rounds. For a fair comparison, we use number of the queries as the X-



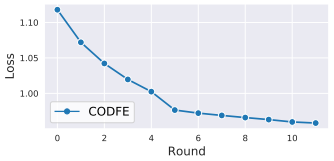
(a) MNIST, probability-only (b) MNIST, label-only

(c) Fashion-MNIST, probability-only (d) Fashion-MNIST, label-only

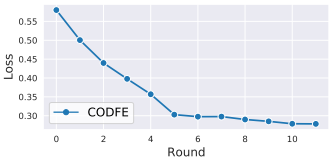
Figure 2: Accuracy of substitute models.

axis[3]. We further note that DaST trains the entire model via one training iteration per set of synthetically generated data, whereas CODFE updates the entire model via multiple training iterations per set of synthetic data. From Figure (2a) to (2d), it is clear that the substitute model accuracy of CODFE outperforms DaST with a significant gap (89 v.s.52 for label-only and 91 v.s.30 for probability-only on MNIST as example). Moreover, CODFE can quickly reach high accuracy with a much smaller number of queries than DaST. It strongly demonstrates the effectiveness of CODFE in stealing the target model in a data-free manner. Another worth mentioning observation is that CODFE converges to the maximum accuracy steadily in both label-only and probability-only scenarios, whereas DaST apparently vibrates in both. Similar observation can be made for datasets of Fashion-MNIST. The only difference is that due to different task complexity, the accuracy of stolen model of Fashion-MNIST is lower than that of MNIST using the same CNN. We also provide the long run training process of DaST to verify our statement on its model stealing performance.

Convergence Process. For DaST, we can see from Figure (2) that the training accuracy fluctuates and doesn't stop at a good local optima along the entire training procedure of 350,000 queries for (2a)(2b) (450,000, 500,000 for (2c)(2d) respectively). The accuracy of substitute model does not show an increasing trend in Figure (2a)(2c). Nevertheless DaST saves the substitute model each iteration, and choose the one of highest accuracy to be the final results. We argue the it is infeasible to select the best model for DaST because attackers do not have real data to evaluate their saved models. Another issue is that what stopping criteria shall be used for terminating the training especially when the accuracy does not converge stably, e.g., shown in Figure. (2c). Further, once the model stealing process ends, it requires additional resources and efforts to store and to select the best substitute model. This further leads to the unstable the training process of DaST. Comparatively, the convergence process of CODFE is almost monotonic. The training accuracy increase smoothly during the whole training phrase, and converge to local optima at around 200,000 queries for MNIST and 400,000 for Fashion-MNIST. The loss of substitute models shown in Figure 4 stays consistent with our theoretical analysis on convergence guarantee in Section. 3.3.



(a) Probability-only:MNIST



(b) Label-only:MNIST

Figure 3: Convergence of CODFE over training rounds: loss of the substitute model

[3]Here we limit the number of queries for plotting according to the convergence of CODFE. Actually the training process of DaST is far longer however its modeling stealing accuracy stays fluctuating below CODFE. The entire results is provided in Appendix.

Table 1: Attack success rates on three datasets: CODFE v.s. DaST under adversarial scenarios of label-only and probability-only.

Dataset	Attack	Untargeted				Targeted			
		DaST-P	CODFE-P	DaST-L	CODFE-L	DaST-P	CODFE-P	DaST-L	CODFE-L
MNIST	*FGSM*	24.65	**52.00**	20.22	**53.72**	6.33	**16.04**	6.32	**16.47**
	BIM	27.93	**61.76**	27.83	**65.26**	2.49	**15.42**	2.07	**18.94**
	PGD	28.42	**60.92**	27.73	**65.19**	2.78	**15.33**	2.13	**18.97**
Fashion-MNIST	*FGSM*	87.76	**89.08**	**88.77**	86.55	2.10	**30.74**	9.81	**17.61**
	BIM	88.50	**90.42**	85.10	**87.10**	6.18	**37.28**	23.46	**28.46**
	PGD	89.93	**92.07**	85.60	**88.69**	5.71	**37.75**	23.55	**28.63**
Cifar-10	*FGSM*	56.06	**58.60**	54.62	**57.99**	0.20	**0.22**	0.16	**0.22**
	BIM	58.54	**60.18**	56.25	**60.02**	0.27	0.27	0.17	**0.20**
	PGD	58.52	**59.95**	56.15	**59.50**	**0.29**	0.25	0.12	**0.19**

There are multiple reasons behind such differences in convergence between DaST and CODFE. First, CODFE generates a set of synthetic data to update the generator and the substitute model via multiple training iterations, aiming to find the local optima for each set. On the contrary, DaST generates a new set of examples in each iteration and the substitute model and the generator are updated through this set once. As a result, each set is used to guide only one training iteration. However, such one-iteration update provides limited guidance to the training, even worse, the knowledge learned by one update can be dominated by random factors of a new set of synthetic data. Second, the training process in DaST is similar to GANs (Cha et al., 2021; Jeong & Shin, 2021) and inherit the limitations of GANs. Under GANs, real examples are applied to guide the training. Real examples are limited in a small region of feature space. However, for synthetic noise pictures, each new set generated from different seeds can be totally different from the others, which brings instability especially in the case of synthetic examples from a large random generating space. Thus, in CODFE, we only use a fixed set of random seeds and fully exploit them in multiple rounds. Thirdly, the objective function of the generator of DaST are trained in a competition game with $\mathcal{S}$, but for our confidence-based objective, the training procedure is a collaborative game among them. The optimization objectives of $\mathcal{S}$ and $\mathcal{G}$ are complimentary rather than contradictory, and thus enhances stability.

4.3 ATTACK SUCCESS RATES

In this part, we evaluate the attack success rate in both *targeted* and *untargetd* adversarial under label-only and probability-only scenarios on three datasets. The experimental results by ASR are summarized in Table 1. Cifar-10 dataset is chosen here to show the results of the data-free attack among more complex network and task. Prior to discussing the ASR of CODFE, we first show the ASR achieved by real data to steal the target model for MNIST dataset in Table 2. Here we use real data to query the target model so as to train the substitute model. Then we apply the same attack methods for generating adversarial examples. It can be regarded as the upper-bound of ASR for our data-free settings. Overall, the attack success rate of CODFE reaches around 85% of the real data case for untargeted attacks and worse results for targeted attacks depending on the attack methods. From Table 1, it is clear to see that ASR of CODFE significantly outperforms DaST from two to fifteen times high. Comparing untargeted or targeted attacks, both CODFE and DaST show higher ASR for untargeted attack. This can be explained by that untargeted attacks only need classification to shift out of the correct label while the targeted attack requires an elaborate ϵ to lead to the specific misclassification outcome. As a result, more significant ASR improvement resulted from CODFE can be observed for the challenge targeted cases than the untargeted ones. Also, the result that CODFE achieves about 55% high of real data scenario is as expected, because more elaborate ϵ is required but too demanding for synthetic data.

Table 2: MNIST: ASR using unlabeled real data to steal model.

Attack	Untargeted		Targeted	
	Real-P	Real-L	Real-P	Real-L
FGSM	63.95	61.80	16.38	15.68
BIM	79.02	80.24	29.95	30.06
PGD	78.77	80.46	29.48	30.20

When it comes to difference between label-only and probability-only, higher attack success rates are achieved under the probability-only scenarios. This can be explained by that in label-only scenarios, attackers train the substitute model by pairs of synthetic data X and querying output labels. Specifically, the learning of the substitute model is to imitate the output label of $\mathcal{T}$ by cross entropy loss. Thus, when outputting the same label (class element of maximum probability), it does not provide information for training. On the contrary, the feedback the probability vector of all classes provides much more supervision to the substitute model. When $\mathcal{S}$ and $\mathcal{T}$ output the same label of maximum probability, $\mathcal{S}$ continues to learn until all of the probabilities become similar. It is in line with our previous statement that probability-only scenarios are more informative, and achieve stronger attacks.

Let us zoom into the difference among datasets. In general, CODFE greatly outperforms DaST in all three datasets. Compared to MNIST, the attack success rate of Fashion-MNIST is higher although the accuracy of $\mathcal{S}$ in Figure. 2 is lower. It is because that the classification task of MNIST is very easy, adding perturbation helps more for attacking when the task is not that easy. However, despite the fact that Cifar-10 task is more challenging than Fashion-MNIST, the attack success rate on Cifar-10 appears to be lower. The reason of such results lies on the fact of low accuracy of the substitute model. Training on Cifar-10 even with real data and deep networks can be difficult (Çalik & Demirci, 2018; Thakkar et al., 2018). Without surprise, shallow substitute models trained from synthetic data thus reach even lower accuracy. This observation unfortunately implies the limitation of data-free adversarial attacks in complex learning tasks that need deeper network structures.

4.4 ATTACKING EFFICIENCY

Another performance advantage of CODFE is attacking efficiency, compared to the other data-free black-box attacks. We zoom into the metrics to show efficiency of the proposed model stealing process: the number of queries needed to achieve the maximum accuracy for substitute model accuracy. The results of number of queries and training time of two version of CODFE are summarized in Table 3. One can see that CODFE requires 37%–95% fewer queries to train the substitute models. As CODFE has light-weight networks, a smaller number of query pairs is needed and the required training time is thus lower. Another reason lies on the convergence speed. As the training of substitute model of DaST fluctuates a lot, it is hard to find the terminating point of training process so that more iterations are needed to obtain better $\mathcal{S}$. Moreover, as the generator of DaST contains several different components, e.g., one generator per class, it requires a large number of query pairs to train such complex networks and faces the difficulty of slow convergence or even non-convergence. In contrast, CODFE converges steadily and fast. In a nutshell, CODFE not only requires a lower number of synthetic query examples but also training time than DaST.

Table 3: Number of queries to achieve the maximum substitute model accuracy.

Dataset	DaST-P	CODFE-P	DaST-L	CODFE-L
MNIST	5900000	**352452**	21500000	**344414**
Fashion-MNIST	900000	**445488**	750000	**475487**
Cifar-10	1000000	**166410**	300000	**195466**

5 CONCLUSION

It's challenging to design adversarial attacks without the knowledge of the target model and the access to real data. In this paper, we proposed a confident data-free model stealing framework, CODFE, which learns a substitute model through querying the target model via synthetically generated data. CODFE is composed of two collaborating models, a generator and a substitute model, aiming to maximize the confidence of generating synthetic images and minimizing the cross entropy loss of substitute model. We design a training procedure to explicitly exploit every set of synthetic images and theoretically prove its convergence. We empirically demonstrate that CODFE effectively steals the target model using a low number of queries for three data sets. Under various of adversarial scenarios we show that the substitute model achieves 14%–56% higher accuracy and up to 34% higher attack success rate than the state of the art data-free model stealing adversarial attacks.

REFERENCES

Rasim Caner Çalik and M. Fatih Demirci. Cifar-10 image classification with convolutional neural networks for embedded systems. In *15th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2018, Aqaba, Jordan, October 28 - Nov. 1, 2018*, pp. 1–2. IEEE Computer Society, 2018. doi: 10.1109/AICCSA.2018.8612873. URL https://doi.org/10.1109/AICCSA.2018.8612873.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 39–57. IEEE Computer Society, 2017. doi: 10.1109/SP.2017.49. URL https://doi.org/10.1109/SP.2017.49.

Sungmin Cha, Taeeon Park, Byeongjoon Kim, Jongduk Baek, and Taesup Moon. GAN2GAN: generative noise learning for blind denoising with single noisy images. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=SHvF5xaueVn.

Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. Exploring connections between active learning and model extraction. In Srdjan Capkun and Franziska Roesner (eds.), *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pp. 1309–1326. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/chandrasekaran.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2206–2216. PMLR, 2020. URL http://proceedings.mlr.press/v119/croce20b.html.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 9185–9193. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00957. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Dong_Boosting_Adversarial_Attacks_CVPR_2018_paper.html.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pp. 31–36. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-2006. URL https://aclanthology.org/P18-2006/.

Gongfan Fang, Jie Song, Chengchao Shen, Xinchao Wang, Da Chen, and Mingli Song. Data-free adversarial distillation. *CoRR*, abs/1912.11006, 2019. URL http://arxiv.org/abs/1912.11006.

Farzan Farnia and Asuman E. Ozdaglar. Gans may have no nash equilibria. *CoRR*, abs/2002.09124, 2020. URL https://arxiv.org/abs/2002.09124.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6572.

Cloud Team Google. Google cloud vision api for ocr. https://cloud.google.com/vision/docs/ocr, 2016.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2142–2151. PMLR, 2018. URL http://proceedings.mlr.press/v80/ilyas18a.html.

Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. High accuracy and high fidelity extraction of neural networks. In Srdjan Capkun and Franziska Roesner (eds.), *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pp. 1345–1362. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/jagielski.

Jongheon Jeong and Jinwoo Shin. Training gans with stronger augmentations via contrastive discriminator. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=eo6U4CAwVmg.

Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. PRADA: protecting against DNN model stealing attacks. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pp. 512–527. IEEE, 2019. doi: 10.1109/EuroSP.2019.00044. URL https://doi.org/10.1109/EuroSP.2019.00044.

Sanjay Kariyappa, Atul Prakash, and Moinuddin K. Qureshi. MAZE: data-free model stealing attack using zeroth-order gradient estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 13814–13823. Computer Vision Foundation / IEEE, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Kariyappa_MAZE_Data-Free_Model_Stealing_Attack_Using_Zeroth-Order_Gradient_Estimation_CVPR_2021_paper.html.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Byl5NREFDr.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HJGU3Rodl.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Mingkun Li and Ishwar K. Sethi. Confidence-based active learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(8):1251–1261, 2006. doi: 10.1109/TPAMI.2006.156. URL https://doi.org/10.1109/TPAMI.2006.156.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4954–4963. Computer Vision Foundation / IEEE, 2019a. doi: 10.1109/CVPR.2019.00509. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Orekondy_Knockoff_Nets_Stealing_Functionality_of_Black-Box_Models_CVPR_2019_paper.html.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4954–4963. Computer Vision Foundation / IEEE, 2019b. doi: 10.1109/CVPR.2019.00509. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Orekondy_Knockoff_Nets_Stealing_Functionality_of_Black-Box_Models_CVPR_2019_paper.html.

Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pp. 372–387. IEEE, 2016. doi: 10.1109/EuroSP.2016.36. URL https://doi.org/10.1109/EuroSP.2016.36.

Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi (eds.), *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pp. 506–519. ACM, 2017. doi: 10.1145/3052973.3053009. URL https://doi.org/10.1145/3052973.3053009.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6199.

Vignesh Thakkar, Suman Tewary, and Chandan Chakraborty. Batch normalization in convolutional neural networks—a comparative study with cifar-10 data. In *2018 fifth international conference on emerging applications of information technology (EAIT)*, pp. 1–5. IEEE, 2018.

Jean-Baptiste Truong, Pratyush Maini, Robert J. Walls, and Nicolas Papernot. Data-free model extraction. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 4771–4780. Computer Vision Foundation / IEEE, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Truong_Data-Free_Model_Extraction_CVPR_2021_paper.html.

Eric Wallace, Mitchell Stern, and Dawn Song. Imitation attacks and defenses for black-box machine translation systems. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 5531–5546. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.446. URL https://doi.org/10.18653/v1/2020.emnlp-main.446.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Mingyi Zhou, Jing Wu, Yipeng Liu, Shuaicheng Liu, and Ce Zhu. Dast: Data-free substitute training for adversarial attacks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 231–240. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.00031. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Zhou_DaST_Data-Free_Substitute_Training_for_Adversarial_Attacks_CVPR_2020_paper.html.

A CROSS ENTROPY AND KL-DIVERGENCE

Let $\mathcal{S}(x)$ and $\mathcal{T}(x)$ to be the softmax outputs of the substitute model and the target model. The expressions of the KL-divergence loss and the cross entropy loss are shown in the following.

$$\mathcal{L}_{\mathrm{KL}}(x) = \sum_{i=1}^{N} \mathcal{T}_i(x) \log\left(\frac{\mathcal{T}_i(x)}{\mathcal{S}_i(x)}\right)$$

$$\mathcal{L}_{\mathrm{CE}}(x) = -\sum_{i=1}^{N} \mathcal{T}_i(x) \log \mathcal{S}_i(x)$$

In the black-box setting of this paper, $\mathcal{T}$'s model parameters are unavailable. Therefore $\mathcal{T}(x)$ is a constant vector. In this case, we have

$$\nabla_x \mathcal{L}_{\mathrm{KL}}(x) = \nabla_x \mathcal{L}_{\mathrm{CE}}(x) = -\sum_{i=1}^{N} \mathcal{T}_i(x) \nabla \left[\log \mathcal{S}_i(x)\right]$$

But an interesting observation is that in white-box settings ($\mathcal{T}$ is derivable) cross entropy won't suffer from vanishing gradients but KL-divergence can suffer from vanishing gradients. The justification is in the following. Taking the gradient over $\mathcal{L}_{\mathrm{KL}}(x)$, we have

$$\begin{aligned}
\nabla_x \mathcal{L}_{\mathrm{KL}}(x) &= \sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \log \frac{\mathcal{T}_i}{\mathcal{S}_i} + \frac{\partial \mathcal{T}_i}{\partial x} - \frac{\partial \mathcal{S}_i}{\partial x} \frac{\mathcal{T}_i}{\mathcal{S}_i} \\
&= \sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \log \frac{\mathcal{T}_i}{\mathcal{S}_i} - \frac{\partial \mathcal{S}_i}{\partial x} \frac{\mathcal{T}_i}{\mathcal{S}_i},
\end{aligned}$$

where $\sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} = 0$ because $\sum_{i=1}^{N} \mathcal{T}_i = 1$. When $\mathcal{S}$ converges to $\mathcal{T}$ we have $\mathcal{T}_i(x) = \mathcal{S}_i(x)\left(1 + \epsilon_i(x)\right)$ where $\epsilon_i(x)$ tends to 0 during the convergence process. When $\epsilon_i(x)$ tends to 0, $\log\left(1 + \epsilon_i(x)\right) \approx \epsilon_i(x)$. Then we have

$$\begin{aligned}
\nabla_x \mathcal{L}_{\mathrm{KL}}(x) &\approx \sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \epsilon_i - \frac{\partial \mathcal{S}_i}{\partial x}\left(1 + \epsilon_i\right) \\
&\approx \sum_{i=1}^{N} \epsilon_i \left(\frac{\partial \mathcal{T}_i}{\partial x} - \frac{\partial \mathcal{S}_i}{\partial x}\right),
\end{aligned} \tag{3}$$

where we have applied $\sum_{i=1}^{N} \frac{\partial \mathcal{S}_i}{\partial x} = 0$ because $\sum_{i=1}^{N} \mathcal{S}_i = 1$. According to Equation (3), the gradient of $\mathcal{L}_{\mathrm{KL}}(x)$ will gradually vanish after many iterations. Then taking the gradient over $\mathcal{L}_{\mathrm{CE}}(x)$, we have

$$\begin{aligned}
\nabla_x \mathcal{L}_{\mathrm{CE}}(x) &= -\sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \log \mathcal{S}_i + \frac{\partial \mathcal{S}_i}{\partial x} \frac{\mathcal{T}_i}{\mathcal{S}_i} \\
&\approx -\sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \log \mathcal{S}_i - \sum_{i=1}^{N} \frac{\partial \mathcal{S}_i}{\partial x} - \sum_{i=1}^{N} \epsilon_i \frac{\partial \mathcal{S}_i}{\partial x} \\
&= -\sum_{i=1}^{N} \frac{\partial \mathcal{T}_i}{\partial x} \log \mathcal{S}_i - \sum_{i=1}^{N} \epsilon_i \frac{\partial \mathcal{S}_i}{\partial x}
\end{aligned}$$

where the first term won't vanish during the iterations. Therefore the cross entropy loss won't suffer from vanishing gradients. Note that we have applied $\sum_{i=1}^{N} \frac{\partial \mathcal{S}_i}{\partial x} = 0$.

B PROOF OF THEOREM 1

B.1 NOTATIONS AND LEMMAS

For simplicity, we use $\mathcal{T}\left(z;\theta_g^{(t)}\right)$ to denote $\mathcal{T}\left(\mathcal{G}(z;\theta_g^{(t)})\right)$ and $\mathcal{S}\left(z;\theta_g^{(t)},\theta_s^{(t)}\right)$ to denote $\mathcal{S}\left(\mathcal{G}(z;\theta_g^{(t)});\theta_s^{(t)}\right)$. In order to prove Theorem 1, we firstly derive Lemma 1 according to the assumptions in Section 3.3.

Lemma 1. *After the training of $\mathcal{G}$ (line 11-14, Algorithm 1) in round t, given $z \in Z$, we have* $\mathrm{CE}(\mathcal{T}(z;\theta_g^{(t)}),\mathcal{S}(z;\theta_g^{(t)},\theta_s^{(t)})) \leq \mathrm{CE}(\mathcal{T}(z;\theta_g^{(t-1)}),\mathcal{S}(z;\theta_g^{(t-1)},\theta_s^{(t)}))$.

Proof. After training $\mathcal{G}$ we have

$$\mathcal{S}_{i^*}(z,\theta_g^{(t)},\theta_s^{(t)}) \geq \mathcal{S}_{i^*}(z,\theta_g^{(t-1)},\theta_s^{(t)}),$$
$$\mathcal{T}_{i^*}(z,\theta_g^{(t)},\theta_s^{(t)}) \geq \mathcal{T}_{i^*}(z,\theta_g^{(t-1)},\theta_s^{(t)}),$$

where $i^* = \arg\max_i \mathcal{S}_i(\mathcal{G}(z;\theta_g^{(t-1)});\theta_s^{(t)}) = \arg\max_i \mathcal{T}_i(\mathcal{G}(z;\theta_g^{(t-1)}))$. Then we have

$$\begin{aligned}
\mathrm{CE}(\mathcal{T}(z;\theta_g^{(t)}),\mathcal{S}(z;\theta_g^{(t)},\theta_s^{(t)})) &= -\sum_{i=1}^{N}\mathcal{T}_i(z;\theta_g^{(t)})\log\mathcal{S}_i(z;\theta_g^{(t)},\theta_s^{(t)}) \\
&\leq -\sum_{i=1}^{N}\mathcal{T}_i(z;\theta_g^{(t-1)})\log\mathcal{S}_i(z;\theta_g^{(t)},\theta_s^{(t)}) \\
&\leq -\sum_{i=1}^{N}\mathcal{T}_i(z;\theta_g^{(t-1)})\log\mathcal{S}_i(z;\theta_g^{(t-1)},\theta_s^{(t)}) \\
&= \mathrm{CE}(\mathcal{T}(z;\theta_g^{(t-1)}),\mathcal{S}(z;\theta_g^{(t-1)},\theta_s^{(t)}))
\end{aligned}$$

B.2 COMPLETING THE PROOF

Theorem 1. *Given $z \in Z$. Let $f\left(\theta_s^{(t)}\right) = \mathrm{CE}\left(\mathcal{T}\left(\mathcal{G}(z;\theta_g^{(t)})\right),\mathcal{S}\left(\mathcal{G}(z;\theta_g^{(t)});\theta_s^{(t)}\right)\right)$. Training the substitute model by Algorithm 1, we have $\lim_{t\to\infty} f(\theta_s^{(t)}) = \epsilon^*$, where $\epsilon^* \geq 0$.*

Proof. We can simplify $f\left(\theta_s^{(t)}\right)$ as

$$f\left(\theta_s^{(t)}\right) = \mathrm{CE}\left(\mathcal{T}\left(z;\theta^{(t)}\right),\mathcal{S}\left(z;\theta_g^{(t)},\theta_s^{(t)}\right)\right),$$

where t is used to index the training rounds. Using Lemma 1, We have

$$f\left(\theta_s^{(t+1)}\right) \leq \mathrm{CE}\left(\mathcal{T}\left(z;\theta g^{(t)}\right),S\left(z;\theta_g^{(t)},\theta_s^{(t+1)}\right)\right).$$

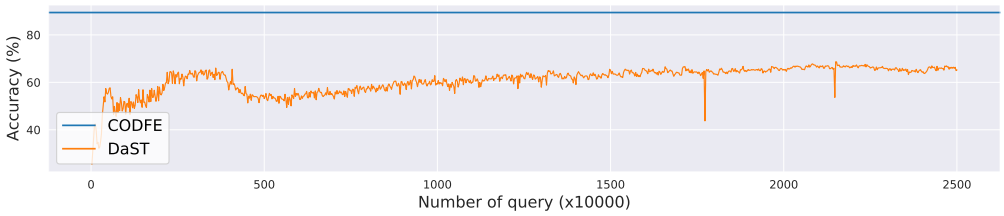
Since the cross entropy loss is the loss function of $\mathcal{S}$, we have

$$\begin{aligned}
f\left(\theta_s^{(t+1)}\right) &\leq \mathrm{CE}\left(\mathcal{T}\left(z,\theta g^{(t)}\right),\mathcal{S}\left(z,\theta_g^{(t)},\theta_s^{(t+1)}\right)\right) \\
&\leq \mathrm{CE}\left(T\left(z,\theta g^{(t)}\right),S\left(z,\theta_g^{(t)},\theta_s^{(t)}\right)\right) \\
&= f\left(\theta_s^{(t)}\right)
\end{aligned}$$

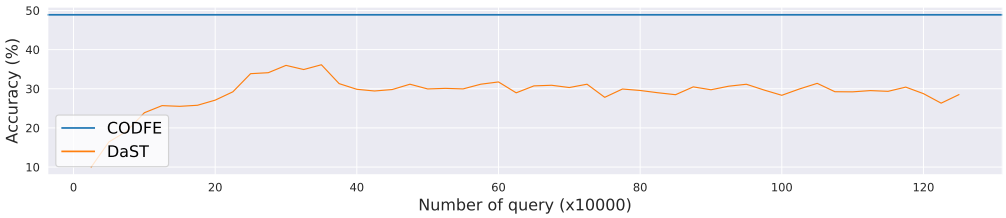
Therefore, we know that $f(\theta_s)$ is monotone decreasing during the training. $f(\theta_s) = 0$ if and only if $\mathcal{T}\left(z;\theta^{(t)}\right) = \mathcal{S}\left(z;\theta_g^{(t)},\theta_s^{(t)}\right)$. Otherwise $f(\theta_s) > 0$. Since $f(\theta_s) \geq 0$, it will converge. However the outputs of $\mathcal{S}$ and $\mathcal{T}$ usually won't be exactly the same. Then the convergence can be formally represented as $\lim_{t\to\infty} f(\theta_s^{(t)}) = \epsilon^*$, where $\epsilon^* \geq 0$.

C ADDITIONAL EXPERIMENTAL RESULTS

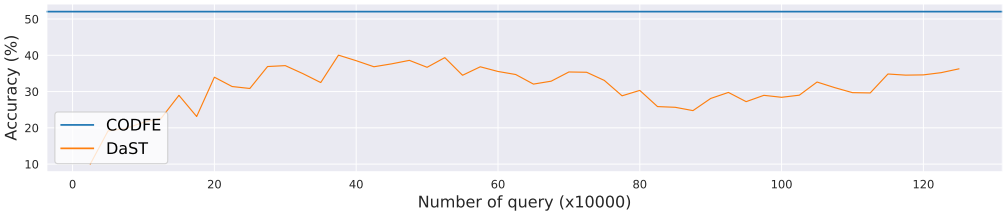
(a) Probability-only:MNIST



(b) Label-only:MNIST



(c) Probability-only:Fashion-MNIST



(d) Label-only:Fashion-MNIST

Figure 4: Substitute model accuracy.