DEFENDING AGAINST DISTILLATION-BASED MODEL STEALING ATTACKS

DAMITMA YÖNTEMİ İLE MODEL ÇALMA ATAKLARINA KARŞI SAVUNMA

EDA YILMAZ

ASSOC. PROF. DR. HACER YALIM KELEŞ

Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

May 2024

ABSTRACT

DEFENDING AGAINST DISTILLATION-BASED MODEL STEALING ATTACKS

Eda Yılmaz

Master of Science, Computer Engineering

Supervisor: Assoc. Prof. Dr. Hacer YALIM KELEŞ

May 2024, 72 pages

Knowledge Distillation (KD) allows a complex teacher network to pass on its skills to a simpler student network, improving the student's accuracy. However, KD can also be used in model theft, where adversaries try to copy the teacher network's performance. Influenced by the "Stingy Teacher" model, recent research has shown that sparse outputs can greatly reduce the student model's effectiveness and prevent model theft. This work, using the CIFAR10, CIFAR100, and Tiny-Imagenet datasets, presents a way to train a teacher that protects its outputs, inspired by the "Nasty Teacher" concept, to prevent intellectual property theft. To enhance the teacher's defenses, this method mixes sparse outputs from adversarial images with original training data. Additionaly, a new loss function, the Exponential Predictive Divergence (EPD) loss, is introduced to hide the model's outputs without reducing accuracy. This method effectively reduces the EPD loss between the model's responses to adversarial and clean images, allowing the creation of adversarial logits without harming the network's performance.

Keywords: Knowledge distillation, Adversarial examples, Model stealing attacks

ÖZET

DAMITMA YÖNTEMİ İLE MODEL ÇALMA ATAKLARINA KARŞI SAVUNMA

Eda Yılmaz

Yüksek Lisans, Bilgisayar Mühendisliği

Danışman: Doç. Dr. Hacer YALIM KELEŞ

Mayıs 2024, 72 sayfa

Bilgi Damıtma (BD), karmaşık bir öğretmen ağından basit bir öğrenci ağına temel becerilerin aktarılmasını sağlayarak daha yüksek doğruluk sağlar. Ayrıca, BD, düşmancıl çalma saldırıları aracılığıyla öğretmen ağının işlevselliğinin kopyalanmasının hedeflendiği model hırsızlığı senaryolarında kullanılır. *Stingy Teacher* modelinden etkilenen son araştırmalar, seyrek çıktıların öğrenci model etkinliğini önemli ölçüde azaltabileceğini ortaya koymuştur. Bu çalışma, öğretmenin çıktılarını koruyan ve entelektüel mülkiyet hırsızlığı riskini azaltan *Nasty Teacher* konseptinden esinlenerek, CIFAR10, CIFAR100 ve Tiny-Imagenet verisetleri kullanarak, bir öğretmen model eğitme tekniği sunmaktadır. Öğretmenin öğrencilere karşı savunmalarını güçlendirmek için, bu strateji düşmancıl örneklerin seyrek çıkrıları ile orijinal eğitim verilerini benzersiz bir şekilde birleştirir. Yeni bir kayıp fonksiyonu olan Exponential Predictive Divergence (EPD), bu yaklaşımda modelin çıktılarını manipüle etmek için kullanılır ve uygulanırken doğruluğu azaltmaz. Bu strateji, modelin düşmancıl ve orijinal görüntülere verdiği yanıtlar arasındaki EPD kayıp fonksiyonunu etkin bir şekilde azaltarak, ağın performansı üzerinde neredeyse hiç negatif etkisi olmadan düşmacıl çıktıların üretilmesini sağlar.

Keywords: Bilgi damıtma, Düşmancıl örnekler, Model çalma saldırıları

ACKNOWLEDGEMENTS

I extend my heartfelt gratitude and special appreciation to my supervisor, Associate Professor Hacer Yalım Keleş, for her unwavering guidance and encouragement at every phase of this endeavor. Her passion for science has been a constant source of inspiration for me. I am deeply indebted to her for her invaluable support, without which I would not have been able to complete this work.

I would also like to express my deep appreciation to my parents and sister for their steadfast support. I dedicate the values embodied in this work to my mother, Latife Yılmaz, who always had my back and extend great effort for my education. I am immensely grateful to my father, Murat Yılmaz, for sharing my enthusiasm for engineering and being a great vision. Additionally, I extend my heartfelt thanks to my sister, Elif Yılmaz, for always being a wonderful study companion.

I would also like to express my heartfelt gratitude to my beloved Nihat Emre Karakan, whose unwavering support has been my guiding light throughout this journey, as it has always been. His consistent presence by my side has been a source of motivation and inspiration without which I would never have achieved what I have.

CONTENTS

Page

TABLES

FIGURES

ABBREVIATIONS

API	:	**A**pplication **P**rogramming **I**nterface
AST	:	**A**dversarial **S**parse **T**eacher
CNN	:	**C**onvolutional **N**eural **N**etwork
EPD	:	**E**xponential **P**redictive **D**ivergence
IP	:	**I**ntellectual **P**roperty
KD	:	**K**nowledge **D**istillation
KL	:	**K**ullback **L**iebler
MLaaS	:	**M**achine **L**earning **a**s **a** **S**ervice
NT	:	**N**asty **T**eacher
NN	:	**N**eural **N**etwork
ST	:	**S**tingy **T**eacher
STT	:	**S**tingy **T**rained **T**eacher

1. INTRODUCTION

As the volume of data increases exponentially and technical capabilities continue to progress, deep learning models are becoming more and more essential in the artificial intelligence area. These models have demonstrated amazing effectiveness in a variety of applications because of their capacity to learn hierarchical data representations. These models, enables the extraction of high-level complex abstractions as data passes through their successive layers. As a result, deep learning models have become essential in advancing AI research and applications. They play a key role in driving innovation and improving machines' ability to understand the complex world around us. The impressive capabilities of deep learning models come with a significant demand for computational power and extensive data. This poses challenges for deploying these models on smaller or mobile devices. Consequently, there is a need for smaller, more efficient models that can achieve similar performance to their advanced counterparts. Knowledge Distillation (KD) is a method that compresses the knowledge embedded within large, complex models into a smaller network. Hinton *et al.* introduced KD for the first time. This technique generalizes model compression using soft targets [3]. Most classifiers employ the softmax function in their final layer. However, the output of the softmax function tends to be overly confident, with low entropy, resulting in almost zero probabilities for incorrect classes. The intuition behind knowledge distillation lies in leveraging these incorrect class probabilities to learn the similarities and differences between classes from this distribution. KD addresses this issue by utilizing 'soft targets', which are softened versions of the output logits, achieved by introducing a temperature parameter. This temperature parameter reduces the distance between correct and incorrect classes within probability distribution. Subsequently, KD aims to reduce the Kullback-Leibler (KL) divergence between the outputs of a complex model, referred to as the 'teacher' network, and a simpler 'student' model. Through this method, the performance of a student network may be improved by transferring information from the outputs of a complicated neural network to it. Success of KD has caused a growing interest among researchers. This method has been applied to different kinds

of knowledge, distillation algorithms, teacher-student architectures, varying numbers of teacher-student pairs, and across various domains. Traditional KD uses output responses of a neural network as knowledge. However, in classification problems with a small number of classes, response-based distillation may become inefficient. To address this issue, feature and relationship-based distillation algorithms have been developed. Feature-based methods leverage intermediate features to inherit the functionality of a model, while relationship-based techniques utilize both intermediate features and output knowledge, including relationships between different layers. Furthermore, this method has been applied to improve student's prediction accuracy without the requirement for a teacher network. Self-distillation transfers information from the last layers to the previous levels, compressing the network itself [4]. Additionally, attention mechanisms have been used for improving the performance of KD by learning the behavior of attention maps from the teacher model [5]. These advancements have contributed to the effectiveness and efficiency of knowledge distillation in various machine learning applications. However, leveraging from only the outputs of a neural networks poses risks to the privacy and security of the intellectual property associated with these sophisticated, complex, and expensive models. While deep learning methods and models becomes developed also the techniques that tries to steal, harm or tries to make them wrong prediction developed. As deep learning methods and models evolve, there are also advancements in techniques for unauthorized access to models, replicating functionality, and accessing the sensitive data they use. Additionally, there is a growing concern regarding the potential for adversarial actions to harm the model, leading to incorrect predictions [6]. Adversarial examples are anomalies carefully created to trick models into making incorrect predictions. Its been proved that most high-performing classifiers are vulnerable to adversarial examples [7]. Adversarial images are often imperceptible to humans, yet machines frequently misclassify these examples. The transferability of adversarial images is an interesting characteristic. An image generated to mislead one network also tends to deceive another network into misclassifying it. This characteristic of adversarial images, underscores the generalizability of the ability to confuse deep neural networks across various models with diverse structures and training data [8]. To develop more robust models against adversarial examples the most known technique

called ”adversarial training” can be used. Adversarial training makes networks familiar with adversarial examples by using them while training alongside with original images. With these images as training material, the model learns key aspects and becomes capable of accurate prediction.

However, this process requires creating an additional dataset and also greater computation power. Since training a large network is expensive, needing more computational power is infeasible but defense is still necessary. Some improvements on this field has been made to create more efficient defenses. One of the example techniwues named Adversarial Logit Pairing (ALP) also uses adversarial examples to make model more robust against them but it doesn't feed them to the model. It reduces the difference between model's output probability distributions for clean images and their adversarial counterparts by minimizing the L2-norm between them. This process widens the decision boundary and regularizes the model. With these model trained with ALP achieves adversarial robustness [9]. Furthermore, combining logit pairing methods with adversarial training techniques has shown improved accuracy compared to using either method alone [10]. Another growing concern about privacy and security of deep learning models is model stealing or extraction attacks. As artificial Intelligence (AI) systems become more common and advanced in various domains, the risk of protecting Intellectual Property (IP) and integrity of these models arise. Model stealing and extraction attacks represent sophisticated methods used by adversaries to identify weaknesses in AI systems. Therefore, they may copy the functionality, parameters or hyperparameters of complex black-box models with much less cost. Nevertheless, sharing a model black-box does not mean that model is robust against stealing attacks. Adversaries can steal from only the hard labels of a neural network. A cloud-based service known as ”Machine Learning as a Service” (MLaaS) enables customers to access machine learning tools and algorithms online without having to pay for expensive software or hardware infrastructure. However, MLaaS companies face the risk of model theft when providing access to their customers. They must carefully strike a balance between protecting their intellectual property and offering value to clients [11]. Model stealing attacks, often leverage the predictions obtained from queries exploiting MLaaS's

Application Programming Interface (API) which operates on a pay-for-query basis most of the time. Attackers obtain output label or probability of the specific class. By sending queries and observing the model's responses, attackers aim to reconstruct the model or copy the performance of it, potentially obtaining sensitive information such as training data, hyperparameters, or even the model itself. For instance, Orekondy *et al.* trains a "knockoff" model from image-prediction pairs obtained from original model by applying black-box version of KD. Milli *et al.* reconstructs the victim model by querying the gradient of it [12].

Various defense methods has been developed against model stealing attacks. Among these techniques, some focuses on verification of ownership. Watermarking, for instance, adds a unique identifier named watermark to network to proof ownership. In cases where verification remains insufficient, passport-based methods can be used. These methods involves adding a passport to the network, which contains information for verifying ownership. However, unauthorized access can cause the network to fail. Additionally, there are methods similar to passport-based methods that fail when they detect an adversary but without an identifier. These methods includes a trigger mechanism for detecting out-of-distribution harmful activities. When an attack is detected, it causes the model to fail for protecting. Recently, some defensive methods have been developed against distillation-based attacks that do not trigger failure. These techniques involve creating a model that maintains its behavior regardless of whether it is under attack. These defensive teachers produce outputs that provide correct predictions but mislead student models attempting to replicate the teachers' functionality from their outputs. Nasty Teacher (NT), for example, creates a defensive version of a neural network by increasing the distance between logits of the base model and NT while maintaining performance with minimizing cross-entropy loss between ground truth and predictions [13]. Another method, Stingy Teacher (ST), achieves defense by revealing only a small portion of the output probability distribution, which is softened with a temperature parameter, while putting zero on the remaining distribution. This strategy not only reveals the hard label but also top n predictions in the distribution, causing the student to fail in knowledge transfer [14]. Additionally,

Semantic Nasty Teacher calculates semantic relationship within dataset to decide the weight of KL divergence in the loss function then uses a similar combined objective function with NT to prevent model compression attacks [15].

The motivation behind our study is to prevent model stealing attacks, even in fully revealed white-box scenarios, by developing a defensive teacher model. The proposed teacher model, which we refer as Adversarial Sparse Teacher (AST), shares similarities with NT. Both models maintain consistent behavior across various circumstances and lack an adversary detection mechanism. Moreover, they both leverage logit responses to enhance their defensiveness. However, while NT focuses on maximizing the KL divergence between the output responses of the teacher and student models with one term while minimizing cross-entropy loss with another, AST has a different primary objective. AST aims to achieve a more stable loss function by minimizing all terms in its combined loss function. Adversarial examples are known to trick image classifiers by making minimal modifications to original images. Thus, AST minimizes the distance between the output probabilities of models for original images and their adversarial versions. The methodology involves training a base model using the original dataset and then synthesizing adversarial versions of this dataset. Subsequently, the teacher model is trained with the objective of minimizing the Exponential Predictive Divergence (EPD) loss between adversarial and clean images, while simultaneously minimizing cross-entropy loss between the labels and predictions. In this thesis, a novel loss function, EPD, is proposed alongside with the training mechanism of the AST. It has been observed that using KL divergence while training AST, can lead to performance degradation. The proposed EPD loss addresses this issue effectively, providing a more robust training mechanism for AST. More detailed explanation about EPD is given in Chapter 4..

1.1. Scope Of The Thesis

This thesis primarily concentrates on model stealing attacks, exploring the vulnerabilities of models to these attacks and proposing strategies to mitigate these vulnerabilities. This study

covers a novel and efficient technique that can be used for protecting deep neural networks against theft attempts. Moreover, alongside with this approach, a new objective function is introduced which helps to model to enhance its performance while increasing its robustness.

1.2. Contributions

In this work, these shortcomings of other methods are addressed by introducing an innovative and effective method. The following is a summary of this paper's main contributions:

- We provide *Adversarial Sparse Teacher* (AST), a novel training paradigm that makes use of sparse logits, adversarial examples. By strengthening the model's robustness against stealing attacks, this approach seeks to solve a critical issue in modern model security, as proposed in our related paper [1].
- A new divergence metric Exponential Predictive Divergence (EPD) is proposed. This function provides a novel technique for distance reduction between predicted and target probability distributions. Particularly in our study this metric enables the improve security while keeping accuracy high. The metric is also introduced in our related paper [1].
- In contrast to other approaches that involve creating defensive teachers, AST does not rely on training a secondary defensive teacher model from the base model. Instead, AST utilizes sparse logits of adversarial examples during training to enhance robustness. This methodology aims to fortify model resilience directly through the incorporation of adversarial examples, rather than relying on an additional teacher model.

1.3. Organization

The organization of the thesis is as follows:

- Section 1 covers the reasons behind the study, gives a general summary, discusses the contributions, and outlines the scope of the thesis.
- Section 2 provides detailed, technical background overview of this thesis to understand the technical aspects of proposed methods.
- Section 3 provides a detailed review of the literature relevant to each category this study addresses.
- Section 4 introduces our novel defense method and objective function.
- Section 5 demonstrates experimental setup.
- Section 6 outlines the results of our defense method across various network architectures and datasets. Additionally, this section includes an analysis of the impact of different parameters.
- Section 7 summarizes the thesis and suggests possible directions for future research.

2. BACKGROUND OVERVIEW

In this section, the basics of our subject matter is explored. A comprehensive overview of key concepts and relevant theoretical frameworks is provided. The objective is to equip the readers with the essential knowledge required to navigate through the complexities of our discussion as well as understand the background of our study. This section dives into the background information that shapes understanding and guides the study.

2.1. Neural Networks

Neural Networks (NNs) are a machine learning method that replicates the human brain by creating mathematical representations of neurons and the connections between them. Their capacity to extract complicated patterns and representations from input via a network of interconnected neurons is what makes them unique. In image classification the network takes an image as input, with each pixel value typically representing a feature. In hidden layers connected neurons process the input data. Subsequently, the final layer produces the network's output, which is a probability distribution over possible classes in image classification task [16].

2.2. Knowledge Distillation

Working process of KD is illustrated in Figure 2.1. Training a student network, symbolized as $f_S(\cdot)$, to extract the information inside the outputs of a teacher network, denoted as $f_T(\cdot)$, is the main objective of KD. Each network operates with its unique set of parameters, where θ_T indicates the parameters of the teacher and θ_S indicates the student's. The dataset used for training, denoted by $\mathcal{X}$, consists of images and their respective labels, expressed as $(x^{(i)}, y^{(i)})$. For every sample $x^{(i)}$, the output response generated by the network $f(\cdot)$ is represented by the symbol $f(x^{(i)}; \theta)$. The softmax temperature function, denoted as $\sigma_\tau(\cdot)$ which is introduced by Hinton *et al.* [3], plays a crucial role in this process. The softmax function typically yields a highly confident prediction for the correct class, while assigning near-zero probabilities to

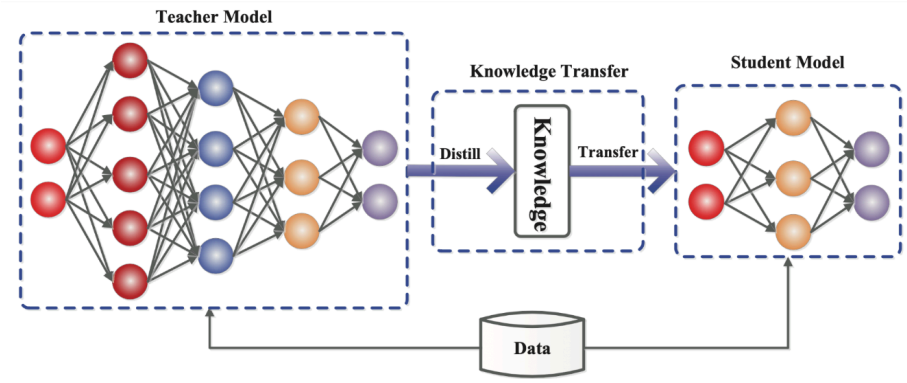

Figure 2.1 The framework for KD. Diagram is obtained from [2].

other classes. This scenario makes the distillation process more difficult because it limits the model to learn from the probabilities of incorrect classifications. This function converts logits into soft probabilities under the influence of a temperature parameter τ, typically set higher than 1. When the temperature τ equals 1, the function defaults to the standard softmax operation, $\sigma(\cdot)$. For a deeper understanding of the softmax and temperature scaled softmax activation functions, detailed information is provided in the Section 2.3.. The loss function of KD, which is developed for the effective training of the student, is displayed in Equation 1.

$$L_{KD} = \omega\tau^2 \; L_{KL}(\sigma_\tau(f_T(x^{(i)};\theta_T)), \sigma_\tau(f_S(x^{(i)};\theta_S)) + (1-\omega) L_{CE}(\sigma((f_S(x^{(i)};\theta_S)), y^{(i)}) \quad (1)$$

This equation is containing two segments. The initial part quantifies the KL divergence. This divergence is calculated between the temperature scaled outputs from the teacher, $(\sigma_\tau(f_T(x^{(i)};\theta_T))$, and the student, $(\sigma_\tau(f_S(x^{(i)};\theta_S))$. This divergence serves as a measure of the variance in the soft probability distributions yielded by the two models. Moreover, the second part of the equation assesses the cross-entropy loss between the soft probabilities derived from the student model, $(\sigma_\tau(f_S(x^{(i)};\theta_S))$, and the true labels. This improves the model's prediction accuracy by ensuring that the student maintains a direct connection with the true labels, in addition to learning from the teacher.

The balancing act between reducing the distance between predicted and target probability distributions and cross-entropy between labels and predictions is controlled by a

hyperparameter, ω. By adjusting ω, one can fine-tune the training process to either increase the importance of direct label prediction or prioritize the mimicking of the teacher's output distribution, thereby optimizing the student network's performance according to the particular demands of the task. This approach underscores the flexibility and adaptability inherent in knowledge distillation strategies, making them particularly useful for compressing complex models into more efficient, easy to deploy counterparts.

2.3. Softmax and Temperature Scaled Softmax Activation Functions

The softmax activation function is a mathematical function commonly used in machine learning and neural networks. It operates on a vector of real-valued inputs and produces an output vector whose elements represent probabilities. These probabilities are normalized in such a way that they sum up to 1, ensuring that the output vector forms a valid probability distribution. It is commonly used in classification tasks, where it transforms the raw output of a neural network into probabilities corresponding to each class. Since the softmax function produces a probability distribution as an output, wherein the model must estimate the likelihood that each class is right, it is appropriate for multi-class classification tasks. The formula for the softmax function is provided in equation (2)

$$\sigma(z)_i = \frac{e^{(z_i)}}{\sum_{j=1}^{K} e^{(z_j)}} \tag{2}$$

$\sigma(z)_i$, denotes the probability distribution of the raw output, represented by z, of the neural network. K represents the number of classes. Each member z_i of the input vector z is subjected to the exponential function, and the resultant values are normalized by dividing by the total of the exponentials. In essence, the softmax function turns the raw output scores into probabilities, making it easier to interpret and compare the model's predictions across different classes [16].

In the context of KD, information related to incorrect class predictions holds significant value. The student model enhances its performance not solely from the hard label

information but also from learning through these incorrect class probabilities. A confident distribution is usually produced using the softmax activation function, which gives the right class a probability near to one and the remaining classes a probability close to zero. However, the softmax temperature function simplifies KD by scaling this distribution with a temperature parameter. Dividing the logits by a constant narrows the gap between probabilities, effectively reducing the distance between them. Softmax temperature function is given in equation (3).

$$\sigma_\tau(z)_i = \frac{e^{(\frac{z_i}{\tau})}}{\sum_{j=1}^{K} e^{(\frac{z_j}{\tau})}} \tag{3}$$

The expression $\sigma_\tau(z)_i$ indicates the temperature scaled probability distribution, where z represents the raw output of the neural network. z_i resperesents each element of the raw output and K indicates how many classes there are. Additionally, the temperature parameter τ scales the distribution by dividing all values by this constant. When the temperature value rises, it increases the entropy of the distribution, making it more akin to a uniform distribution. Conversely, a decrease in this value reduces entropy, resulting in a distribution with higher confidence. Setting the value to 1 causes the function to behave just like the standard softmax function [3].

2.4. Generation of Adversarial Examples

Projected Gradient Descent (PGD) emerges as a pivotal method in the realm of adversarial machine learning, especially in the crafting of adversarial examples. It initiates its process from a randomly chosen point within a specified range near original input x, encapsulated by what is known as the ϵ-ball. The core of this technique involves iteratively modifying the input towards increasing the loss function, where each iteration's modification magnitude is carefully controlled by a predetermined parameter, α. The equation that dictates how each modification is made in each iteration is shown in (4).

$$x^{(t+1)} = \text{Project}_\epsilon \left(x^{(t)} + \alpha \cdot \text{sign} \left(\nabla_x \mathcal{L}(\theta, x, y) \right) \right) \tag{4}$$

The function 'Project' ensures that each step's alterations are carefully recalibrated within the specified allowable range, as defined by the ϵ-ball, thus preventing any going beyond the setted ϵ threshold [17]. The findings of Mandry et al. [17] highlight the significant impact of incorporating adversarial examples, generated via the PGD methodology, into the training of classifiers. This strategic approach has the potential to greatly improve classifier robustness, making them more resilient against various types of adversarial attacks. Our approach is specifically designed to undermine the student's capacity to properly copy the teacher's functioning by utilizing such examples. The property of transferability associated with adversarial examples stands out as a intresting characteristic, where such examples, though crafted for a particular network, frequently result in misclassification by different network models as well [8]. This characteristic of transferability suggests an intriguing ability of adversarial perturbations to navigate across diverse neural network architectures, maintaining their disruptive effect. This characteristic of adversarial examples may reduce the computation time for techniques that use adversarial images to enhance the robustness of a neural network.

2.5. Sparse Logits

The concept introduced in the *Stingy Teacher* framework posits that enhancing the smoothness within model's output distribution can lead to significant improvements in the student model's performance during the Knowledge Distillation (KD) process. The pivotal element that makes a teacher model less susceptible to effective distillation lies in the sparsity characteristic of its output signals. Achieving this sparsity involves selectively maintaining only a small portion of the top class probabilities within the logits and setting rest of it to the zero. This inherently shifts the focus of the model towards the most critical classes, thereby pushing the less important classes to the background. This methodological approach serves to deceive the distillation process, effectively diminishing the adversary's ability to steal the model's performance [14].

ST, maintains its performance while preventing model stealing attacks through distillation. Unlike other methods that manipulate outputs, ST adopts a more different approach. It selectively removes the probability from less accurate classes while revealing the remaining insights. Moreover, ST employs temperature scaling to adjust the output distribution, enhancing the confidence of classes other than the correct one. In conclusion, ST keeps accuracy high by always keeping the most confident class and prevents adversaries from stealing the model by emphasizing the most relevant classes with high confidence while filtering out less confident ones. This approach ensures that the adversary fails to extract valuable insights, thereby safeguarding the model's integrity.

3. RELATED WORKS

3.1. Knowledge Distillation

Knowledge distillation involves optimizing deep learning models with the goal of transferring information from a bigger, more complicated ”teacher” network to a simpler, more streamlined ”student” network. This technique not only improves the efficiency of smaller models but also deals with limitations regarding computational resources and deployment in environments with restricted resource. The concept of Knowledge Distillation (KD) was first introduced by Buciluǎ et al. [18] and later expanded upon by Hinton et al. [3]. Since then, many researchers have been working to improve and broaden its uses and effectiveness. KD can generally be classified by the kind of knowledge being transferred: this includes output responses [3, 19], features [5, 20, 21], and relationships [22–24]. Each type aims to capture a distinct dimension of the teacher's understanding, from its final output decisions and the connections it forms internally, to how it processes and relates different pieces of information [2]. In KD approaches that utilizes information from features, a student is trained to copy the internal features that the teacher has learned. This makes the richer and more detailed representations available to the student model, that the teacher model has learned, which may boost the student model's accuracy on a variety of tasks [20]. Relationship-based knowledge distillation examines the relationships between the various layers of the model and makes use of both the intermediate features and output responses. The student model can perform better on a variety of tasks and obtain a comprehensive understanding of the data's underlying structure by utilizing these relationships. When dealing with classification models that have a small number of classes, output-based KD techniques might not be feasible since they cannot efficiently learn from inaccurate class probabilities. Relationship and feature-based knowledge distillation techniques therefore provide more workable and efficient solutions in these kinds of situations. An innovative approach within KD is called Self Distillation, in which a model is designed to learn from

its own predictions, thereby enhancing its accuracy and performance without relying on a separate, more complex teacher model [4].

3.1.1. Distance Metrics for Knowledge Transfer

KD algorithms differs from each other in the terms of knowledge, method, number of teachers and more. The distance metric used for copying the knowledge from is also a parameter that affects the performance of KD. Traditional KD employs KL divergence as distance metric [3]. Aguilar *et al.* also utilize this divergence function but with internal representations as knowledge, different from traditional KD [25]. In the *Nasty Teacher* method which shares similar loss function structure with response-based KD, KL divergence is employed as a part of the objective function. In most of the KD related works $L1$ and $L2$ distance related metrics are used by researcher [26]. For instance, Romero *et al.* [20], Komodakis and Zaguruyko [5], Gao *et al.* [27] and Changyong *et al.* [28] utilizes $L2$ distance while Wang *et al.* [29] and Zhang *et al.* [30] employs $L1$ distance instead. Additionaly, Adversarial Logit Pairing (ALP), also employs the $L2$ distance for reducing the divergence between the pairs of logits [9]. Utilizing KL divergence is more efficient when knowledge comes from output representations as probability distribution [31]. However, while transferring knowledge from features and internal relationship $L1$ and $L2$ distance based metrics is more usable. Moreover, some researchers like Shin *et al.* [32] and Park *et al.* [22] utilizes cosine-similarity for calculating similarities in their works.

3.2. Model Stealing Attacks

Deep learning model stealing attacks, also known as model extraction attacks, are a type of security threat where an adversary attempts to replicate a target deep learning model by observing its behavior, such as querying it with inputs and analyzing its outputs. When the target model is proprietary or has been trained on sensitive data, these attacks are very concerning. Recent research has confirmed that adversaries that only access the responses of the model can effectively steal its functionality. For instance Tramer *et al.* [33] demonstrated

the vunerability of machine learning models to stealing attacks and adversaries could steal the models by prediction APIs. Employing the concept of KD [3], multiple studies have showed the feasibility of extracting the functionality of deep neural networks. Orekondy *et al.* [34] showcased that even with weakly associated queries, or in some cases, entirely random queries [35], it is possible to replicate the performance of the targeted model. However, it's important that attacks employing queries more closely aligned with the target task tend to provide better outcomes. Similarly, Lopes *et al.* [36] introduced a KD method to extract model without original training data. This technique only uses some extra information by getting metadata from pretrained teacher's activation layers. Additionaly, Chen *et al.* [37] employed a Generative Adversarial Network (GAN) to enable data-free KD. In this method, a pretrained teacher model serves as the discriminator while a generator creates training data. In certain instances, these methods facilitate the compression of models trained with extensive datasets, offering significant reductions in model size and easier deployment. However, it is important to acknowledge that these techniques also pose a risk to the intellectual property of machine learning models.

3.3. Security and Privacy of Machine Learning Models

The rise of deep learning models has revolutionized various industries, from healthcare to finance and beyond. These models have become necessary tools for solving complex problems and extracting valuable knowledge from huge amounts of data. Nonetheless, the necessity to preserve the intellectual property (IP) of these models grows along with their importance and complexity. Developing deep learning models requires a combination of expert knowledge, access to sensitive data, and substantial computational resources. As such, it represents a significant investment for organizations and researchers alike. It is crucial to safeguard this investment and the innovations included in these models. Some methods had been developed to protect the IP of these valuable models [38]. These methods can be categorized and examined under several key categories. Watermarking methods adds an identifier inside the model's parameters or outputs. These watermarks serve as unique signatures, enabling the verification of model ownership and tracing unauthorized usage

[39, 40]. Nevertheless, the majority of watermarking techniques do not prevent model theft; rather, they just confirm ownership. Passport-based defense methods involve embedding a digital passport within the deep learning model. This passport serves as a key that must be provided for the model to function properly. Without a valid passport, unauthorized access results in the model to fail. This approach effectively prevents the from being used illegaly an guarantees that only users with permission can access to the functioning model [41, 42]. Backdoor attacks, conversely, operate effectively under normal circumstances but fail to perform when they detect adversarial behavior or an attack [43]. Similarly, Kariyappa and Qureshi proposes a defense strategy that incorporates system which detects out-of-distribution and identify adversarial behaviour. The approach dynamically handles these behaviours by providing incorrect predictions from an secondary ”misinformation model” which generates predictions that is uncorrelated with original predictions [44].

Juuti *et al.* introduced PRADA, a detection mechanism designed to identify model stealing attacks. It examines the pattern of successive API requests and triggers an alert when this pattern diverges from typical harmless behavior [45]. Recently some works presented defensive teacher networks that cannot be distilled . *Nasty Teacher*, trains a defensive teacher from base model by increasing the distance between logits of base and defensive teacher while keeping high performance by decreasing the cross-entropy loss between training images and labels [13]. *Stingy Teacher* is a theoretical method that shows making outputs of the base model sparse and soft creates an effective defense [14]. Furthermore, while training the teacher model, Semantic Nasty Teacher deconstructs the semantic relationships within the output logits.It first converts class names into vectors with n dimensions using a Word2Vec model, and then it uses a cosine similarity metric to evaluate the logical ties within the classes and it dynamically sets the weight of KL divergence loss depending on class similarities [15].

3.4. Adversarial Examples

Adversarial examples represent a fascinating yet concerning aspect of machine learning models, particularly neural networks. These examples are carefully crafted input images that cannot be distinguished with human eye, yet capable of causing the model to make significant mistakes in its predictions. Fast Gradient Sign Method (FGSM) is a technique to synthesis these examples [8]. The underlying concept is straightforward: following the computation of the objective function derived from the classification of the input image, the gradient of this loss is computed. However, rather than employing this gradient to minimize the loss, it is utilized to maximize it. Accordingly, adjustments are made to the pixels in a manner that increases the loss, thereby leads model to misclassify the image [8]. Basic Iterative Method (BIM) is very similar to FGSM but it changes the image iteratively with small steps instead of single step [46]. Projected Gradient Descent (PGD), is almost works like BIM. It applies perturbations iteratively with small steps until it reaches the maximum permissible magnitude. The difference between BIM and PGD is PGD initializes in a random point within ϵ-ball while BIM starts in a fixed point [17].

3.5. Defense Against Adversarial Attacks

Defense mechanisms against adversarial examples are essencial for making machine learning models more robust and reliable. Adversarial training is a widely used approach to protect neural networks from the effects of adversarial images. In this approach, models are trained with both clean and adversarial images. By teaching the model with adversarial examples during training, it learns to classify these images, thus improving its robustness [47]. However, adversarial training is highly time consuming and expensive process. To tackle this issue, Kannan *et al.* introduced the Adversarial Logit Pairing (ALP) technique. Unlike adversarial training, ALP does not involve feeding the model with adversarial examples. Instead, it focuses on minimizing the L2 distance between the model's responses to clean and adversarial images. Additionally, in the same study, pairing only logits of two clean images was found to enhance the model's robustness against adversarial examples [9]. Our

research is inspired by the logit pairing method, which demonstrates that models can be made more robust by adjusting their logits and thereby altering the decision boundary.

4. PROPOSED METHOD

4.1. Adversarial Sparse Teacher

This research's main goal is to construct a teacher network capable of producing adversarial output probabilities designed to hinder the extraction of information by models analyzing these outputs. The effectiveness of this approach can be measured based on the performance of the defensive teacher compared to non-defensive networks with similar structures. Hence, the primary design objective is to deceive student models while maintaining performance. First, a base model is trained with original dataset. This model has the same architecture with *Adversarial Sparse Teacher (AST)*. Then, an adversarially perturbed dataset which has identical structure with the clean one is generated from this base model. This dataset, which has been adversarially augmented, is subsequently utilized as a supplementary dataset during the training process of the proposed defensive teacher network. Given that adversarial images are created to increase the objective function of the network, their output probabilities shares the same features as they. Within this framework, our goal is to reduce the divergence between the AST's logits for both clean and perturbed images, thereby achieving adversarially perturbed output responses to clean images. The training diagram of AST is illustrated in Figure 4.1.

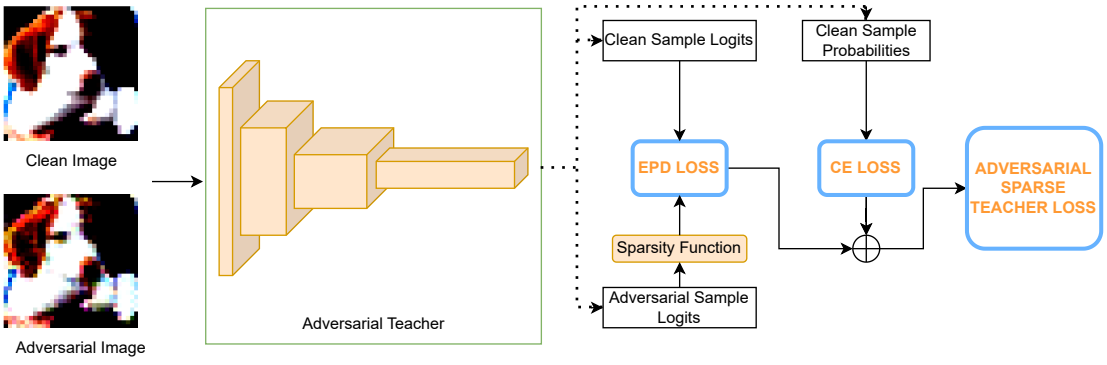

Figure 4.1 Training scheme for *AST*. KL loss refers to the Kullback-Leibler divergence loss, which utilizes logits from both adversarial and original images. CE represents the cross-entropy objective function, which is calculated using the probabilities of clean samples and their corresponding labels. The *AST* loss is the aggregate of these two loss components [1].

Inspired by the *Stingy Teacher* concept, AST also employs sparseness to logits and it accomplishes protection in a defensive manner. Unlike ST, which manipulates its outputs by retaining only the small portion of its output probability distribution with the highest probabilities and cutting out the rest, our proposed method involves training a defensive teacher model that naturally produce confusing output distribution. This is accomplished by concurrently feeding the algorithm with sparse versions of the adversarial image logits that were generated during training. Furthermore, AST maintains its performance while enhancing robustness, demonstrating similar accuracy levels compared to the non-defensive base network. The principal benefit of our strategy lies in its application: by creating a teacher model inherently inclined to generate altered outputs, we effectively protect our model without the need for a detection mechanism or failing the model after detection. AST not only safeguards the teacher network's integrity but also strengthens its defenses against stealing attacks, offering a more robust and feasible option for deployment in real scenarios.

AST's loss function is displayed in equation (5)

$$\begin{aligned} L_{AST} = \omega\tau^2 L_{EPD}(\sigma_\tau(f(x^{(i)};\theta), \sigma_\tau(S(f(x^{(i)}_{adv};\theta),\beta))) \\ +(1-\omega)L_{CE}(\sigma(f(x^{(i)};\theta), y^{(i)}) \end{aligned} \tag{5}$$

In the context of our neural network model, several parameters play crucial roles in defining and optimizing the loss function given in equation (5).

- ω: Weight parameter ω serves to calibrate the contributions of the (L_{CE}) and the proposed *Exponential Predictive Divergence* (EPD) loss (L_{EPD}) Adjusting ω allows us to calibrate the impact of each loss component on the overall objective function.
- f: The function f, represents our neural network, which takes input images (x) and produces corresponding predictions. The subsequent step involves feeding these predictions into the softmax function to derive probability distributions across the classes.
- θ: The network's parameters are indicated by the symbol θ.

- σ: The symbol σ represents the softmax function, which is implemented on the distribution generated by our neural network (f). It creates a probability distribution across potential classes from the raw output scores. In our formulation, σ is often used in conjunction with temperature scaling σ_τ to manage the distribution's entropy.

- σ_τ: The softmax function with temperature parameter τ, denoted as σ_τ is a variant of the standard softmax that adjusts the sharpness of the probability distribution. By manipulating τ, we can regulate the model's confidence in its predictions. When τ is set to 1 this function behaves like the regular softmax function while increased τ leads to a distribution with higher entropy, distributing probabilities more evenly across classes.

- $x^{(i)}$: $x^{(i)}$ represents a clean image from our dataset, while $y^{(i)}$ represents the correct label associated with that image. These pairs $(x^{(i)}, y^{(i)})$ constitute our clean training examples.

- $x_{adv}^{(i)}$: $x_{adv}^{(i)}$ refers to the adversarially perturbed version of a clean image $x^{(i)}$. Adversarial examples are crafted for the non-defensive version of network f.

- β: The parameter β denotes the sparsity ratio of the function S.

- S: The symbol S denotes the sparsity function implemented on the output generated by our model. for $x_{adv}^{(i)}$. The function S, takes a probability distribution and retains the most confident classes in proportion to β while setting the remaining classes to negative infinity.

- L_{CE} and L_{EPD}: The cross-entropy and the introduced EPD cost functions are indicated by these phrases, respectively. L_{CE} measures the discrepancy between predicted and true class probabilities, while L_{EPD} captures the divergence between predicted and sparse adversarial distributions.

To sum up, AST is a technique which utilizes an adversarial dataset that is identical to the original dataset. Generating these examples necessitates a baseline network with an identical

structure to AST, trained on the original dataset. Subsequently, AST utilizes both datasets at the same time, and iterates through them simultaneously. While maintaining high accuracy by diminishing the cross-entropy loss in the latter portion of its combined objective function, it concurrently mitigates the divergence between predicted class probabilities of clean and adversarial images through the EPD loss in the former portion of its combined loss function.

The primary objective is to inherit the characteristics of adversarial examples. Similar to adversarial examples, which are unnoticeable by humans yet frequently misclassified by neural networks, the aim is to imbue the output distribution with a similarly deceptive nature. Since adversarial examples caries the adversary information to the output distribution, the goal is to create an output distribution that mimics the perplexing qualities of adversarial examples, challenging neural networks and potentially deceiving attempts to learn from them [1].

4.2. Exponential Predictive Divergence Loss

During our investigation, we found that the use of KL divergence loss limited our capacity to manipulate the logits in the manner we aimed for. While applying sparse logits, KL divergence also poisons the teacher itself. This led us to look for other ways to calibrate the distance between the predicted and target probability distributions. This search resulted in the creation of a new divergence function, which we describe in equation (6).

$$D_{EPD}(P, Q) = \sum_{i} e^{P(i)} \cdot (P(i) - Q(i)) \tag{6}$$

In this equation, the divergence function receives two probability distributions as input. The distribution Q represents the predicted distribution which is going to be updated, while the distribution P corresponds to the target distribution. $P(i)$ and $Q(i)$ are the logits of the corresponding distributions. The exponential components $e^{P(i)}$ is included to allow these differences to be adjusted based on the value of $P(i)$. This function is specifically designed to incorporate adversarial logits into the output distribution more smoothly. The EPD function

applies a weight proportionate to the target value to modify the model's predictions. In particular, it quickly raises the model's predictions when there is a significant deviation from a high target probability. When predictions are nearly zero, the exponential term ensures that the remaining classes are not cut off. For these lower target probabilities, it boosts predictions more slowly and progressively. By adding ambiguity and increasing prediction entropy, this approach makes the model's responses less predictable, discouraging replication or "stealing". As a result, the EPD function is not only a traditional "divergence" function. Rather, it makes adjustments to predictive divergence in a manner that guards against illegal duplication of the model.

KL divergence, with its logarithmic nature, helps to mitigate the impact of large differences between two probability distributions P and Q. This characteristic contributes to a more stable training experience, reducing the risk of overfitting by controlling the model's learning pace. By using KL divergence as a loss term, we can steer the predicted distribution towards the target distribution. Using KL divergence in our approach makes the outputs of the model directly sparse. In our approach, we frequently feed the model with its own predictions. However, when the model is fed sparse inputs, it tends to fail [14]. This challenge led us to develop a new function that addresses this issue. Unlike KL divergence, our function does not directly force the output distribution to become sparse. Instead, it gently incorporates adversarial sparse probabilities into the overall distribution.

Our function utilizes the term $e^{P(i)}$ to exponentially increase the weights of sparse classes (those with non-zero probabilities). Conversely, in classes with probabilities nearly as low as zero, the term approaches to 1, ensuring these probabilities are not entirely eliminated but increased at a slower rate than the adversarial ones. This approach maintains a non-sparse overall output, preventing the model from undermining its own predictions, while simultaneously increasing entropy and incorporating adversarial outputs with a greater impact. Additionally, EPD function exhibits resilience against numerical problems frequently posed by small probabilities in standard distance measures. By exponentially increasing the divergence, it diminishes the effects of tiny probabilities. This is particularly beneficial as small probabilities often present computational difficulties in KL divergence

assessments. In Figure 6.4, distinctions in output responses of an AST trained with EPD and trained with KL divergence can be observed. KL divergence tends to induce sparsity in the model's output, leading to more pronounced peaks for certain classes and almost zero probabilities for others. On the other hand, EPD maintains a smoother distribution among non-peaky classes, which can increase entropy. This smoothness often contributes to higher model accuracy while maintaining security.

Employing this divergence function in our experiments led to promising results, closely aligning with our expectations of discrepancies within this domain. It provides a clear and insightful metric, especially in situations where traditional measures fail to fully capture the nuances of model predictions [1].

5. EXPERIMENTAL SETUP

The AST model is trained using a synthesized adversarial dataset generated from the base model. Base model is trained with same clean dataset as AST. The generation process employs the PGD technique with specific parameters: an epsilon value of 0.3, a step length of 0.01, and 40 iterations.

Our research utilized three well-regarded datasets within the domain: CIFAR10, CIFAR100 [48] and Tiny-Imagenet [49]. In the experiments conducted on CIFAR10, the teacher network utilized the Resnet18 architecture. We examined several student model structures. These architectures involves a Convolutional Neural Network (CNN) with five layers, another Resnet18, and a couple of modified Resnet architectures designed for CIFAR10. The designed architectures, ResnetC-20 and ResnetC-32, were developed by He and colleagues [50]. For experiments conducted on CIFAR100, the influence of network capacity and dataset complexity is investigated. As teacher model structures Resnet18 and Resnet50 are utilized. The student architectures for these tests included ShufflenetV2 [51] and Resnet18. Additionally, the respective teacher networks themselves were employed. In Tiny-Imagenet experiments, a Resnet50 teacher network and a Resnet18 student network is utilized for examining the effect of the dataset's size.

The CNN was trained for 100 epochs and learning rate fixed at 0.001. In CNN training Adam optimizer is employed [52]. In other experiments SGD optimizer is used. The SGD optimizer is configured with the following settings: momentum is set to 0.9, weight decay is set to 0.0005, and the learning rate is set to 0.1 in initiation. During the training period of 160 epochs for CIFAR10, the learning rate was decreased by multiplying it by 0.1 at the 80th and 120th epochs. The training for CIFAR100 and Tiny-Imagenet continued for 200 epochs. Learning rate reduction were applied by multiplying it by 0.2 at specific epochs which are the 60th, 120th, and 160th epochs. A thorough comparison was made possible by the experiments' and parameters' strict adherence to the *Nasty Teacher* investigation's guidelines [13].

All experiments were performed using five distinct algorithms. Traditional distillation was employed as a baseline for comparison [3]. Two defensive algorithms from the literature, ST and NT, were applied [13, 14]. Additionally, a trained version of ST, which operates similarly to NT but utilizes sparse probabilities, was employed and named *Stingy Sparse Teacher* (STT). STT is excluded in the Tiny-Imagenet experiments. Finally, all experiments were conducted using our introduced method AST [1].

In the training of the *Nasty Teacher*, the temperature parameter, represented by τ, was set to 4 for CIFAR10 and 20 for both CIFAR100 and Tiny-Imagenet. Furthermore, the weighting parameter, denoted as ω, was chosen as 0.04 for CIFAR10, 0.005 for CIFAR100 and 0.01 for Tiny-Imagenet [13].

In the *Stingy Teacher* experiments, the outputs of the base model were adjusted to be sparse, and knowledge distillation (KD) was subsequently applied using these sparse outputs. The KD settings adhered to the recommendations provided in the related work [14].

During the training phase of the *Stingy Trained Teacher*, the sparsity ratio was designated as 0.2 for CIFAR10 and 0.1 for both CIFAR100 and Tiny-Imagenet, aligning with the guidelines outlined in the related work [14]. The parameter τ was specified as 4 for CIFAR10 and 20 for CIFAR100 and Tiny-Imagenet. Furthermore, the weight parameter was established at 0.04 for CIFAR10, 0.005 for CIFAR100 and 0.01 for Tiny-Imagenet, consistent with the settings proposed in [14].

In the *Adversarial Sparse Teacher* training, the parameter τ was configured to 6, while ω was set to 0.05, and the sparse ratio was established at 0.2 for CIFAR10 experiments. Conversely, in CIFAR100 experiments, the sparse ratio was adjusted to 0.02 and 0.03 for Resnet18 and Resnet50 teachers, respectively. Temperature parameter τ was standardized to 20 and the weight parameter ω was set to 0.03 for both teacher networks. During the Tiny-Imagenet experiments, τ was setted to 30, ω was setted to 0.0175 and sparsity ratio was setted to 0.015 which is 3 out of 200 classes [1].

The configuration of distillation experiments remained consistent across all methods. For CIFAR10, the temperature parameter was assigned a value of 4, and the weight parameter was set to 0.9. In the cases of CIFAR100 and Tiny-Imagenet, the temperature parameter was set to 20, with the weight parameter also being set to 0.9.

6. EXPERIMENTAL RESULTS

6.1. CIFAR10 Results

The findings of the tests for the CIFAR10 dataset are summarized in Table 6.1. The Student Base row illustrates student models' base performance depending on the label below it. The 'Teacher Type' column's abbreviations, explained in the table caption, include various models such as ST [14], KD [3], NT [13], STT, and AST, which is introduced teacher model.

The 'Teacher Performance' column aims to showcase results as close as possible to the baseline model. Experiments reveal that the AST teacher model performs comparably to other teacher models like ST, STT, and AST. Notably, student models distilled from AST exhibit significantly improved performance, showing relative reductions in accuracy, which are detailed upon alongside each corresponding score in the table.

ResnetC-20 student model experience a slight accuracy decline of about 0.1% after distillation, while ResnetC-32 models exhibit a decline of approximately 0.47%. There are performance gains of up to 1% for remaining student structures. Comparing the effectiveness of knowledge distillation between NT and AST—methods that train undistillable teacher models to inhibit knowledge transfer—it's observed that both methods result in diminished student accuracy, especially in models with less complexity. However, AST consistently outperforms NT across all student models. STT, which is the trained version of the Stingy Teacher, shows marginally better teacher performance than AST but is less effective in preventing knowledge distillation.

Interestingly, the Stingy Teacher (ST) model, which reveals just a small portion of the base teacher's output responses to the students without specifically creating inherintly defensive model, significantly influences simpler Resnet architectures, outperforming AST. Surprisingly, for the simple CNN student model, AST manages to surpass ST's performance, highlighting the nuanced effectiveness of these diverse training methodologies.

Teacher Type	Network Architecture	Teacher Perf% ↑	Student Performance ↓			
			CNN	ResnetC-20	ResnetC-32	Resnet18
–	Student Base	–	86.64	92.37	93.41	95.03
ST	Resnet18	95.03	83.11(-3.53)	67.98(-24.39)	74.08(-19.33)	92.47(-2.69)
BASE	Resnet18	95.03	87.76(+1.12)	92.27(-0.1)	92.94(-0.47)	95.39(+0.23)
NT	Resnet18	94.37(-0.67)	82.98(-6.62)	88.54(-3.73)	90.07(-2.87)	93.76(-1.63)
STT	Resnet18	**94.94**(-0.09)	86.70(+0.06)	91.30(-1.07)	91.85(-1.56)	94.48(-0.68)
AST	Resnet18	94.61(-0.42)	**79.82**(-7.94)	**87.08**(-5.19)	**88.70**(-4.24)	**93.66**(-1.73)

Table 6.1 Accuracy of Resnet18 networks trained on the CIFAR10 using different methods and the accuracy of student networks distilled from these models [1].

6.2. CIFAR100 Results

Table 6.2 presents the outcomes of the tests carried out using the CIFAR100 dataset. In these experiments, we utilized two distinct teacher architectures: Resnet18 and Resnet50. The choice of these architectures allowed us to evaluate the effect of increased network capacity on KD performance due to the larger number of categories in this dataset. The results indicate that the AST teacher, while utilizing the Resnet18 architecture, performs slightly less effectively than the NT and STT methods. However, with the more sophisticated Resnet50 structure, AST teacher model demonstrates superior performance compared to all other teachers, including the base model.

The newly introduced AST teacher shows a significant drop in accuracy and exhibits defensiveness. Notably, in Resnet18 experiments, AST outperforms all existing methods, including ST, which employs a different training scheme than AST. When distilling our robust teacher model to ShufflenetV2, AST reduces the student model's accuracy by 70.01%, and when distilling to Resnet18, it reduces the accuracy by 44.01%. These results are notably superior to all other methods. In the Resnet50 teacher setting, except for the Resnet18 student experiments, our method is also superior. Although ST shows slightly better performance with the Resnet18 student, it relies on partially revealing the outputs. Therefore, our method is the best among scenarios with fully revealed outputs. Additionaly,

since the only change in the training of the STT was the modification of the teacher's input, it suggests that adversarialy altered distributions exert a more significant influence than solely sparse distributions in creating a defensive teacher.

Teacher	**Network**	**Teacher**	**Student Performance ↓**		
Type	**Arhchitecture**	**Perf.% ↑**	**ShufflenetV2**	**Resnet18**	**Teacher Arch.**
–	Student Base	–	72.10	78.28	–
ST	Resnet18	78.28	50.49(-21.61)	55.30(-22.98)	55.30(-22.98)
ST	Resnet50	77.55	46.46(-25.64)	54.22(-24.06)	54.14(-23.41)
BASE	Resnet18	78.28	74.38(+2.28)	79.12(0.84)	79.12(0.84)
NT	Resnet18	77.80(-0.48)	65.01(-7.09)	74.68(-3.60)	74.68(-3.60)
STT	Resnet18	**77.92**(-0.36)	68.47(-3.63)	77.42(-0.86)	77.42(-0.86)
AST	Resnet18	77.02(-1.26)	**4.37**(-70.01)	**44.01**(-34.27)	**44.01**(-34.27)
BASE	Resnet50	77.55	74.00(+1.90)	79.27(+0.99)	80.03(+2.48)
NT	Resnet50	76.88(-0.67)	67.14(-4.96)	73.87(-4.41)	75.99(-1.56)
STT	Resnet50	77.25(-0.3)	70.28(-1.82)	76.16(-2.12)	77.50(-0.05)
AST	Resnet50	**77.69**(0.14)	**26.32**(-45.78)	**58.63**(-19.65)	**46.62**(-30.93)

Table 6.2 Accuracy of Resnet18 and Resnet50 networks trained on the CIFAR100 using different methods and the accuracy of student networks distilled from these models *AST*.

6.3. Tiny-Imagenet Results

A quantitative analysis with Resnet50 teacher architecture and Resnet18 student architecture utilizing Tiny-Imagenet is conducted. In this analysis, traditional KD using baseline teacher, NT, ST and our AST is applied. Table 6.3 displays the corresponding results. NT struggled to maintain security in settings provided by the related paper [13]. The student model exhibited approximately a 2% higher performance than NT but had a 2% lower accuracy compared to the baseline teacher model. Conversely, the ST method showed the expected accuracy drop in the student network. Notably, the method significantly reduced the adversary's performance by around 24%. However, as we already emphasized in CIFAR10 and CIFAR100 experiments, creation of ST is completely different that NT and AST and

Teacher Type	**Teacher Architecture**	**Teacher Perf.% ↑**	**Student Architecture**	**Student Perf.% ↓**
–	Student Base	–	—	63.44
ST	Resnet50	66.11	Resnet18	36.73(-24.06)
BASE	Resnet50	66.11	Resnet18	67.12(+1.01)
NT	Resnet50	64.08(-2.03)	Resnet18	66.33(+2.25)
AST	Resnet50	**66.65**(+0.54)	Resnet18	**66.06**(-0.59)

Table 6.3 Tiny-Imagenet results [1].

results are provided because sparseness is included in both AST and ST methods. In AST, the teacher model slightly outperformed the baseline model and reduced the student model's performance by 0.59%. However, this reduction is significantly less compared to experiments on CIFAR10 and CIFAR100 datasets. Yet, still we observed that AST is superior to the NT method, but further research is required to generalize this technique across different datasets.

6.4. Qualitative and Quantitative Analysis

6.4.1. CIFAR100 Analysis

This part enhances the experimental findings in sections 6.2. and 6.1. by providing supplementary qualitative and qualitative analysis. CIFAR100 is a more fine grained dataset that CIFAR10. Considering this fact, particular attention directed towards examining the Resnet18 architecture and CIFAR100 dataset. The output distributions of various networks, which are the base teacher, NT, STT, and AST, are depicted in Figure 6.1. In alignment with the approach in [14], a softmax temperature has been applied to the output distributions for the purpose of more understandable representation. In the third column the probability distributions generated by the baseline teacher for corresponding adversarial samples are provided. Entropy of the probabilities is displayed below them. The baseline model usually produces a distribution that is almost uniform across classes with one class showing elevated

confidence. This distribution can be seen for both adversarial and clean samples. In clean samples, peaks often appears for the true category, while in adversarial samples, they commonly appear for an incorrect category.

When compared to responses for clean samples, the output responses for adversarial samples show higher levels of confidence and lower entropy. In contrast, outputs of AST has peaks in multiple classes, which typically remains below the true class slightly. This scenario arises due to the implemented sparsity during training and the utilization of the EPD loss. For the classification scenarios with a larger number of classes, smoother distribution combined with the extra peaks increases the entropy of the logit distributions and make the stealing process more difficult .

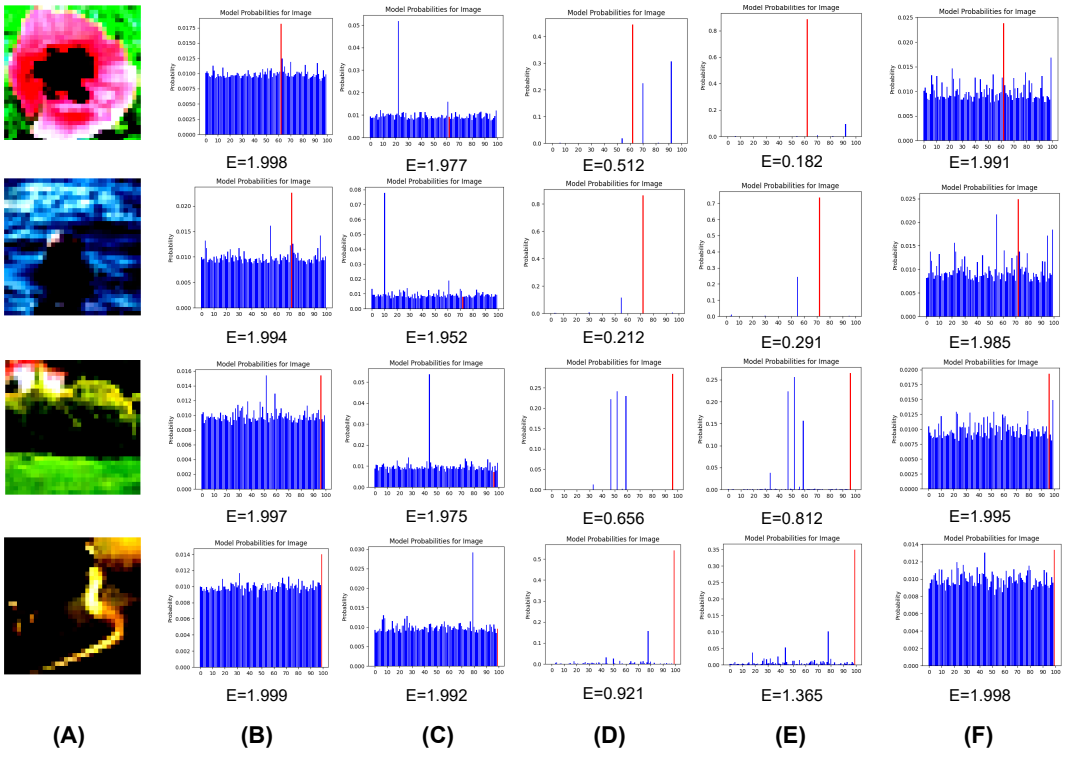

Figure 6.1 Shown is a visualization of output probability distributions post the application of softmax temperature. The columns represent different aspects: (A) clean image examples, (B) baseline network output probability distributions to these clean samples, (C) baseline network output probability distribution to adversarially perturbed images, (D) NT model responses to clean samples, (E) STT model responses to clean images, and (F) Introduced teacher (AST) responses to clean samples. Entropy values for each distribution are displayed below. The figure provides insights into the models' behaviors under various conditions [1].

Additionally, for every teacher network, the entropy values of the output distributions have been obtained for the whole test dataset. The Table 6.4 furnishes statistical data on the entropy values for two specific classes. The numerical outcomes corroborate the qualitative observations and maintain coherence across various classes. Notably, both the baseline network and AST exhibit higher entropy levels in their outputs compared to the NT and STT networks. Nevertheless, the AST demonstrates a greater variance in entropy compared to the baseline teacher network. This disparity stems from the sparsity imposed during training, resulting in output probability distributions with more than one highly confident classes. An additional set of measurements has been formulated, involving the random selection of a reference example from each of two classes. Subsequently, with respect to the results of other samples in the same category, the KL divergence of these reference cases is calculated. The results, outlined in Table 6.4, reveal substantial diversity across the models.

As anticipated, the baseline teacher network reliably displays the lowest average KL divergence for both classes. This indicates a strong alignment with the expected distribution, reflecting accurate model outputs with high confidence. This trait is favorable for recognition tasks but may not be as advantageous for protective measures. On the flip side, the NT network exhibits a higher degree of variability in KL divergence, especially concerning class 10. This disparity from the standard distribution implies a broader deviation and may suggest unstable predictions aimed at deceive the student network. The STT network shares resemblance with the NT network. However, calculated highest KL divergence value among the classes is slightly lower in the in the STT network, indicating a little narrower probability distribution. In the same metric of highest calculated KL divergence, the AST network attains a significantly lower value compared to these models, albeit greater than the baseline teacher model. This emphasizes the stability of the output signals of the AST model.

Furthermore, compared to the baseline teacher network, it exhibits higher variance in divergence values. It is possible to conclude that the intended outputs are being produced in order to deceive the student networks depending on the entropy and KL divergence values.

		KL Divergence				Entropy			
Model	**Label**	**Mean**	**Variance**	**Min**	**Max**	**Mean**	**Variance**	**Min**	**Max**
BASE	0	0.00082	8.56e-8	0	0.00215	1.99831	4.63e-8	1.99626	1.99958
NT	0	0.49432	0.26566	0	3.63332	0.25185	0.14789	0.00895	1.68001
STT	0	0.20675	0.09579	0	2.37950	0.23811	0.09251	0.02666	1.58349
AST	0	0.00240	3.68e-6	0	0.01016	1.98986	1.55e-5	1.98219	1.99746
BASE	10	0.00161	2.61e-7	0	0.00286	1.99841	8.11e-7	1.99361	1.99954
NT	10	1.06778	1.20049	0	4.21843	0.70504	0.21716	0.00968	1.74953
STT	10	0.96356	0.73454	0	4.06417	0.67207	0.24490	0.06806	1.81497
AST	10	0.01149	4.42e-5	0	0.04374	1.98177	0.00058	1.87968	1.99830

Table 6.4 KL Divergences and Entropies Within Class for Resnet18 Model Outputs on CIFAR100 Across Two Categories. The table is extracted from our paper [1].

6.4.2. CIFAR10 Analysis

A comprehensive qualitative analysis utilizing the Resnet18 architecture with the CIFAR10 dataset is presented. This analysis examines the responses of several models to clean images: the baseline teacher, Nasty Teacher (NT [13]), Stingy Teacher (ST [14]), Stingy Trained Teacher (STT)—the operational version of the Stingy Teacher—, and Adversarial Sparse Teacher (AST), as indicated in Figure 6.2. Additionally, the baseline teacher's reaction to altered images is analyzed in a separate column, echoing similar experiments with the CIFAR100 dataset discussed in the previous section.

The results reveal that AST's responses are considerably more sparse than those observed in the CIFAR100 studies, with a tendency to incorrectly favor the same class across different samples from varied categories, aligning with prior model behaviors noted in CIFAR100 analyses.

It's important to note that AST's strategy involves a sparsity ratio of 0.2, limiting it to generate strong responses for only two classes. This restriction often results in NT exhibiting higher entropy in its responses than AST. While this sparse setting modifies the entropy levels in CIFAR10, it is crucial for AST's approach to thwart potential theft of the model.

Despite generating incorrect predictions consistently, AST effectively deters stealing attacks. This consistency in response is also seen in CIFAR100, but the impact of sparsity is more noticeable in CIFAR10 due to its smaller number of classes and a higher sparsity ratio, demonstrating the significant influence of model settings across different dataset conditions.

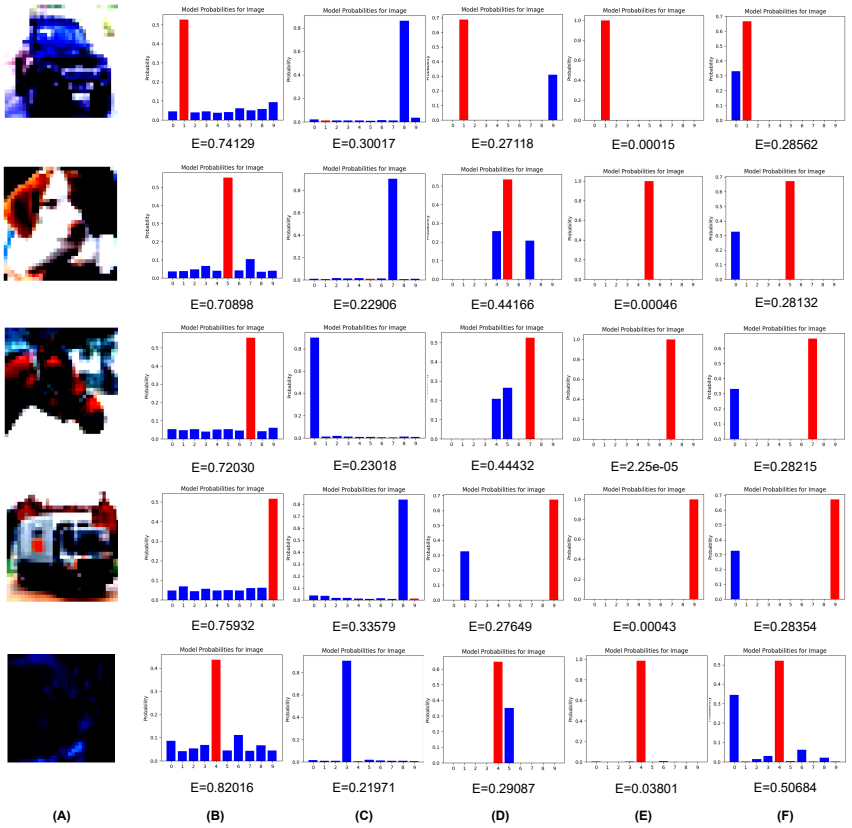

Figure 6.2 The figure illustrates the logit responses after applying softmax with temperature. Each column provides different information: (A) samples of clean images, (B) responses of the baseline model to these clean samples, (C) responses of the baseline model to the adversarial counterparts of the clean images, (D) responses of the NT model to clean samples, (E) responses of the STT model to clean images, and (F) responses of our proposed method AST to clean images. Under each distribution, the entropy of the distributions is provided. All models utilize the Resnet18 architecture and are trained using the CIFAR10 dataset [1].

The analysis provided herein utilizes the Resnet18 architecture with the CIFAR10 dataset to measure the response characteristics of various models by computing the entropy values of the prediction distributions for specified classes within the related categories of the test set. The entropy values are organized into two groups and are detailed in Table 6.5. Both quantitative and qualitative assessments demonstrate consistent results across these

categories. When compared to CIFAR100 experiments, the entropy levels of the AST outputs are lower than those of the NT models but higher than those from the STT model. This variation is attributed to AST's very low sparsity ratio, akin to that of the STT model, leading it to frequently peak at a single incorrect class. In this configuration, the entropy for the STT model is at least ten times lower than that of AST.

Moreover, a KL divergence analysis for the CIFAR10 dataset was conducted. A reference image was picked at random from the selected categories, and its KL divergence was calculated against the outputs of other samples in the same category. The results, which are presented in Table 6.5, show that there are significant variations amongst the models. Similar to the CIFAR100 experiments, the mean relative entropy of AST is higher than that of the base and NT models, which suggests more confusing outputs; however, it is lower compared to the STT model, indicating a consistency in the generated outputs, which also proves to be effective during stealing attacks.

		KL Divergence				**Entropy**			
Model	**Label**	**Mean**	**Variance**	**Min**	**Max**	**Mean**	**Variance**	**Min**	**Max**
BASE	5	0.06159	0.00277	0	0.58339	0.68850	0.01113	0.33726	0.96591
NT[13]	5	0.07565	0.03233	0	2.41124	0.46232	0.00259	0.28000	0.81502
STT[14]	5	0.41480	1.49405	0	6.70777	0.03091	0.00675	4.66e-7	0.61720
AST	5	0.15954	0.18883	0	2.54734	0.32618	0.00810	0.02961	0.76299
BASE	7	0.02263	0.00283	0	0.46163	0.70362	0.00674	0.45502	0.97086
NT[13]	7	0.03958	0.06340	0	3.83768	0.44704	0.00114	0.30123	0.76110
STT[14]	7	0.23204	0.93246	0	5.95962	0.02352	0.00654	4.56e-7	0.61105
AST	7	0.08092	0.10757	0	2.42702	0.31025	0.00552	0.00775	0.73662

Table 6.5 KL divergences and entropies within class for Resnet18 model outputs on CIFAR10 across two categories. The table is obtained from our paper [1].

6.4.3. Tiny-Imagenet Analysis

The output probability distributions of AST to random images of Tiny-Imagenet is illustrated in Figure 6.3. Three or more peaky class other than the correct class (illustrated in red) can

be observed in the figure. However, while entropy is high and distributions showes similar behavior with other datasets, peakyness of the sparse classes in this distributions is reduced relatively. This causes lower security compared to the CIFAR10 and CIFAR100 datasets but still degrading the student performance compared to the teacher. The selection of parameters were more tricky with Tiny-imagenet due to the higher class and image size. Higher value of τ resulted in uniform predicted distribution and too strong weight parameter for EPD loss. So, the parameters of Tiny-ImageNet AST were chosen by considering the selection of $\omega\tau^2$ to be higher than the weight of cross-entropy, but not too dominant. This is because when we increase the τ predicted distribution becomes flat and cross-entropy loss remains relatively weak. When we higher the ω it degrades the weight of cross-entropy and result in lower teacher accuracy. Therefore, a smaller SR was selected to strengthen the signal from adversarial logits and boost the security of the model while maintaining performance.

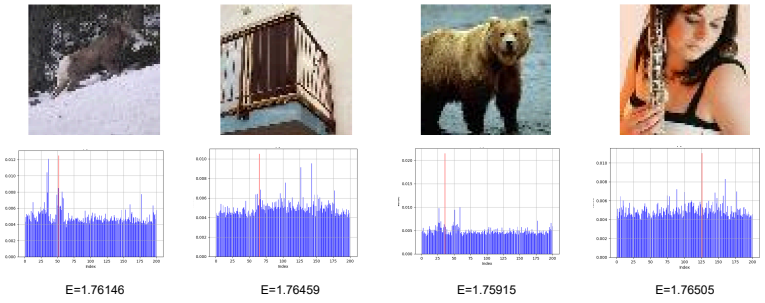

Figure 6.3 Output responses of AST to Tiny-Imagenet test images. Entropy value of each distribution is provided below distributions.

6.5. Ablation Studies

To investigate various parameter values, experiments were conducted using the CIFAR100 dataset and the Resnet18 architecture. In ω and τ experiments sparsity ratio parameter is set to two different values: 0.02 and 0.03. The results for the 0.02 sparsity ratio are provided in Table 6.6, while the results for the 0.03 sparsity ratio are provided in Table 6.7. Generally, a lower weight parameter (ω) resulted in higher teacher performance for both sparsity ratio settings. However, reducing this parameter did not always lead to improved defensiveness.

The impact of the τ and sparsity ratio parameters plays a significant role in achieving optimal teacher and defense performance.

Regarding the temperature parameter (τ), setting it between 20 and 30 typically enhanced defensiveness and caused a significant drop in student accuracy, while maintaining high teacher accuracy. However, higher values of τ led to a decrease in teacher performance and defensiveness, while lower values of τ generally increased teacher performance but decreased defensiveness. We achieved optimal performance with a sparsity ratio of 0.03 and a temperature (τ) value of 20, resulting in a student accuracy drop of 33.01%. The ω and τ parameters are highly dependent since the total loss is calculated using the term $\omega\tau^2$. Therefore, these parameters were selected with consideration for both optimal teacher and student performance.

ω	**0.03**				**0.035**				**0.04**			
τ	**10**	**20**	**30**	**40**	**10**	**20**	**30**	**40**	**10**	**20**	**30**	**40**
Teacher	**77.87**	77.02	76.80	75.69	77.21	76.53	76.13	75.27	76.62	76.67	75.28	74.57
Student	78.55	**44.01**	57.24	74.53	78.40	52.46	60.87	72.65	50.91	64.05	71.73	76.30

Table 6.6 The table displays how various values of the ω and τ parameters affect a Resnet18 *AST* network trained with the CIFAR100 dataset utilizing sparsity ratio of 0.02.

ω	**0.03**				**0.035**				**0.04**			
τ	**10**	**20**	**30**	**40**	**10**	**20**	**30**	**40**	**10**	**20**	**30**	**40**
Teacher	**77.60**	76.65	76.75	75.83	77.16	76.38	75.99	75.43	77.20	76.38	75.45	74.88
Student	78.84	72.50	56.47	63.60	78.32	**54.96**	59.15	73.09	75.87	56.46	71.71	69.51

Table 6.7 The table displays how various values of the ω and τ parameters affect a Resnet18 *AST* network trained with the CIFAR100 dataset utilizing sparsity ratio of 0.03.

Additionally, the effects of the sparsity ratio were analyzed in three different τ and ω settings. The findings of $\omega = 0.035$ and $\tau = 30$ are documented in Table 6.8. The findings of $\omega = 0.035$ and $\tau = 20$ are presented in Table 6.9. The findings of $\omega = 0.03$ and $\tau = 20$ are displayed in Table 6.10. In all scenarios, the best teacher performance is observed without using sparsity.

Teacher Type	SR	Teacher Perf.% ↑	Student Perf.% ↓	Teacher-Student Difference(D)	Teacher Difference(P)	D/P↑
BASE	—	78.28	—	—	—	—
AST	0.01	75.34	72.75	2.59	2.94	0.88
AST	0.02	76.13	60.87	15.26	2.15	7.10
AST	0.03	75.99	**59.15**	16.84	2.29	**7.35**
AST	0.05	76.27	63.30	12.97	2.01	6.45
AST	0.07	76.61	65.90	10.71	1.67	6.41
AST	0.1	76.46	66.77	9.69	1.82	5.32
AST	1	**77.87**	79.69	-1.82	0.41	-4.44

Table 6.8 The table illustrates the effects of varying sparsity ratios on a Resnet18 *AST* model trained with the CIFAR100 dataset with the ω value of 0.035 and τ value of 30.

Teacher Type	SR	Teacher Perf.% ↑	Student Perf.% ↓	Teacher-Student Difference(D)	Teacher-Base Difference(P)	D/P↑
BASE	—	78.28	—	—	—	—
AST	0.01	76.71	70.99	5.72	1.57	3.64
AST	0.02	76.53	**52.46**	24.07	1.75	**13.75**
AST	0.03	76.38	54.96	21.42	1.90	11.27
AST	0.05	76.64	62.32	14.32	1.64	8.73
AST	0.07	76.94	69.53	7.41	1.34	5.53
AST	0.1	76.75	78.25	-1.50	1.53	-0.98
AST	1	**78.24**	79.36	-1.12	0.04	-28.00

Table 6.9 The table illustrates the effects of varying sparsity ratios on a Resnet18 *AST* model trained with the CIFAR100 dataset with the ω value of 0.035 and τ value of 20.

However, this also means the elimination of the defense mechanism. Lower sparsity ratios generally lead to increased defense.

Setting the sparsity ratio to lower values implies assigning higher values to these classes compared to higher values due to the softmax function applied to these logits. This increased

Teacher Type	**SR**	**Teacher Perf.% ↑**	**Student Perf.% ↓**	**Teacher-Student Difference(D)**	**Teacher-Base Difference(P)**	**D/P↑**
BASE	—	78.28	—	—	—	—
AST	0.01	77.07	75.56	1.51	1.21	1.25
AST	0.02	77.02	**44.01**	33.01	1.26	**26.20**
AST	0.03	76.65	72.50	4.15	1.63	2.55
AST	0.05	77.40	77.08	0.32	0.88	0.36
AST	0.07	77.29	77.86	-0.57	0.99	-0.58
AST	0.1	77.12	78.26	-1.14	1.16	-0.98
AST	1	**77.61**	79.05	-1.44	0.67	-2.15

Table 6.10 The table illustrates the effects of varying sparsity ratios on a Resnet18 *AST* model trained with the CIFAR100 dataset with the ω value of 0.03 and τ value of 20.

value enhances the impact of these logits, particularly due to the exponential term in the EPD function, resulting in better defensiveness.

To select the AST model with optimal performance, we defined a new metric considering defensiveness (D) and performance degradation (P) of the trained model utilizing teacher and student accuracies. In this approach, we first calculated the defensiveness of the model by subtracting the accuracy of the student model from that of the teacher model. This indicates the degree of defensiveness of the trained model. Additionally, to assess the performance degradation of the teacher model, we calculated the difference between the baseline teacher and the AST teacher models. Our goal is to increase defensiveness of the ATS model without sacrificing much from its performance. Therefore, we calculated the metric by taking the ratio of model's defensiveness (D) to its performance degradation (P). We selected the model with the highest D/P value. Considering this metric, the best performance is achieved, with a D/P value of 26.20, when ω is set to 0.03, τ is set to 20, and a sparsity ratio is set to 0.02.

6.6. Comparative Analysis of EPD and KL Divergence Metrics

Experiments were conducted on CIFAR100 dataset using Resnet18 architecture to examine the behaviour of the EPD function. Figure 6.4 displays the results of these experiments. Four distinct classes were selected and for each class, four images along with their corresponding output distributions are presented.

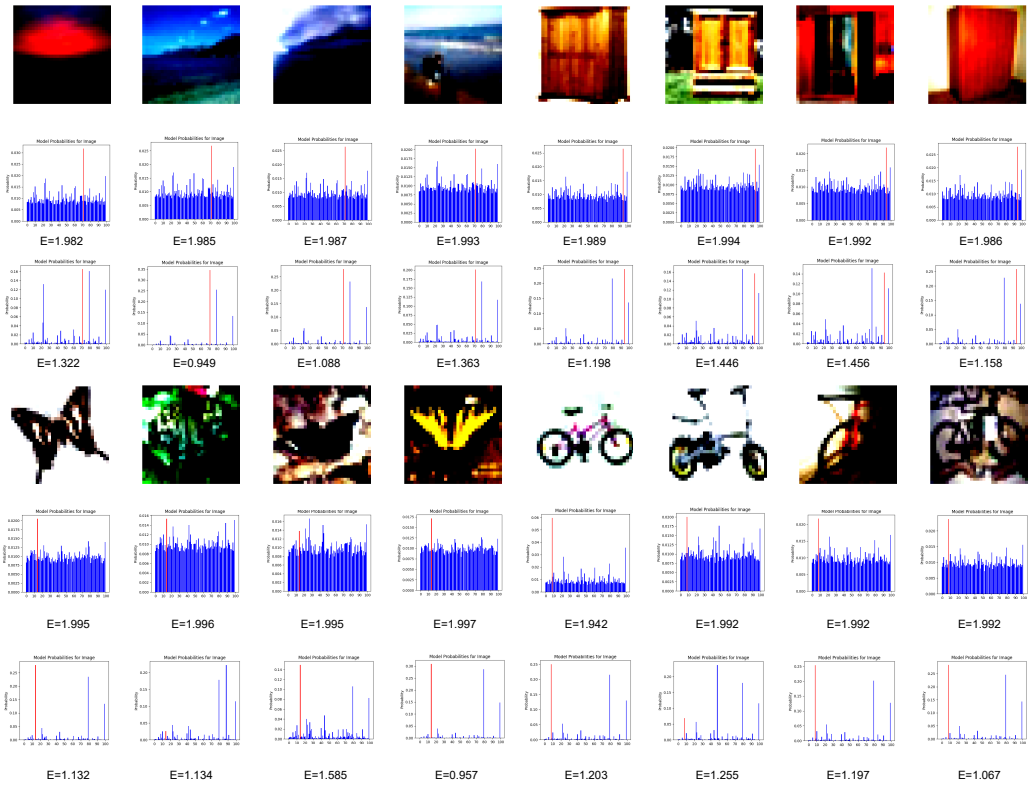

Figure 6.4 The responses from the Resnet18 AST utilizing EPD and the AST utilizing KL divergence to original images from four identical class sets are displayed. The first and fourth rows features input images, the second and fifth rows shows responses from the AST trained with EPD loss, and the third and sixth row presents responses from the AST trained with KL divergence loss. Samples from the same classes are found in the first four columns on the left and the last four columns on the right for both picture rows. The entropy values for each distribution are provided below them, and all results are after softmax temperature [1].

The AST model incorporating EPD exhibits a response with high entropy, leading to an almost uniform distribution across most categories. Within this distribution, there are noticeable peaks for a few classes, which emerge due to adversarial sparse logits. These

peaks are positioned between the highest peak, representing the correct class, and the lower, nearly uniform values of the remaining classes. The correct class, highlighted in red for emphasis, stands out as the most prominent peak in the distribution.

The AST model with KL divergence demonstrates responses characterized by increased sparsity and low entropy. Due to the incorporation of adversarial sparse logits, it also shows peaks in some classes other than the correct class.

Loss Function	**Teacher**	**Student**
	Perf.%$\uparrow$	**Perf.%**$\downarrow$
KL Divergence	67.25	62.34
EPD	**77.02**	**44.01**

Table 6.11 The table shows the impact of cost function on a Resnet18 *AST* network trained with CIFAR100.

The accuracies of the AST models trained with KL divergence and EPD loss, along with the corresponding student accuracies, are given in Table 6.11. For these experiments, two different teachers were utilized: one trained using KL divergence and the other with EPD loss. This setup allows for a comparison of the impact of the different divergence metrics, with both teachers configured with $\omega = 0.03$, $\tau = 20$, and a sparsity ratio of 0.02. The EPD method resulted in better teacher performance with the EPD-trained teacher exhibiting approximately 10% higher accuracy than the teacher trained with KL divergence. This improvement highlights the effectiveness of the EPD method in enhancing teacher performance. The lower performance of the teacher trained with KL divergence can be attributed to the highly confident and sharply sparse outputs it produces. These sparse outputs make it challenging for the student model to effectively distill knowledge, ultimately reducing the student's accuracy [14]. KL divergence as a loss function tends to lead to such sparse outputs. In the AST framework, where the model's own outputs are used during training, this sparsity diminishes teacher performance. On the other hand, the EPD method produces non-sparse outputs, which contributes to the higher performance of

the EPD-trained teacher. EPD incorporates adversarial sparse logits while maintaining high entropy by quickly increasing the adversarial sparse logits and more gradually increasing the other logits thus resulting in a non-sparse output. Moreover, EPD loss also outperforms the KL divergence in the terms of defensiveness. The teacher trained with KL divergence causes a 4.9% drop in student accuracy, while the EPD teacher caused a 33.0% drop. These results indicate that maintaining high entropy while incorporating adversarial sparse logits has a greater impact on the defense mechanism.

Following the CIFAR100 analysis, further experiments were conducted on CIFAR10 to examine the effect of the number of classes. The output responses of the AST model, trained using EPD loss and KL divergence loss, are showcased for specific classes using the Resnet18 architecture on the CIFAR10 dataset. In these experiments with AST, all parameters were kept identical except for the divergence part of the loss. Specifically, ω was set to 0.05, τ to 6, and the sparsity ratio to 0.2 for both teachers. Figure 6.5 illustrates the models' responses to randomly chosen samples from two separate categories.

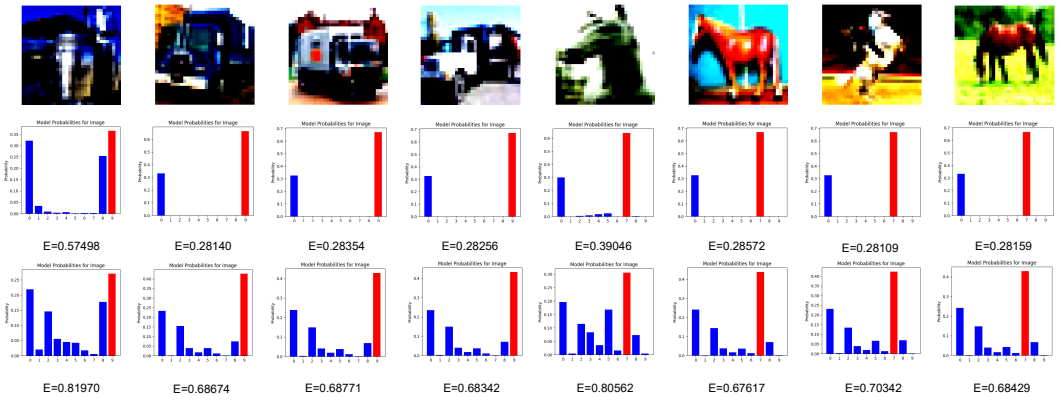

Figure 6.5 On the CIFAR10 dataset, the outcomes of a Resnet18 AST model utilizing EPD and KL divergence losses are shown. The first row displays the input examples. In the middle row, the responses generated by the AST utilizing EPD loss can be observed, while the bottom row displays the outputs produced by the AST utilizing KL divergence loss. Images in the first four columns and the last four columns are categorized similarly. All logits have been subjected to the softmax temperature; the entropies of each distribution is displayed below [1].

When comparing the AST models trained with datasets containing 10, 100, and 200 classes, we observed that the number of classes influenced the entropy of the models trained

with EPD. Specifically, the model trained with the 10-class dataset (CIFAR10) exhibited lower entropy compared to both the 100-class(CIFAR100) and 200-class (Tiny-Imagenet) datasets, which had higher entropy. Both versions of the AST models' outputs in CIFAR10 experiments repeatedly had peaks in the same incorrect classes across all samples. The outputs of the model trained with EPD was more sparse and confident than the model with KL divergence. Due to the chosen sparsity ratio, where 2 out of the 10 classes are selected, the output distribution of the model trained with EPD mostly displayed two peaks. In contrast to the CIFAR100 experiments, the entropy of the model trained with KL divergence in the CIFAR10 experiments was higher than that of the EPD model. However, despite the increased sparsity, the EPD model did not experience a degradation in its own performance. This can be attributed to the reduced number of classes: After applying the softmax function to a distribution with a smaller number of classes, the confidence and impact of the correct class in the EPD model's outputs are enhanced, allowing the model to preserve its performance.

7. CONCLUSION

This thesis presents a novel training method tailored to neural networks engaged in classification tasks.Termed the *Adversarial Sparse Teacher,* this approach is strategically devised to mitigate knowledge theft through Knowledge Distillation, thereby enhancing the network's resilience against such threats. A dedicated objective function was crafted to reduce the divergence in output probability distribution between adversarial and original images. Consequently, AST deliberately furnishes deceptive responses, ensuring a consistent stream of misinformation to confound potential adversaries.

The success of this approach has been demonstrated by several tests conducted on a variety of datasets and teacher-student structures. AST dramatically lowers opponents' accuracy in situations when they have full information, including access to training data. When applied to more complicated teacher structures and datasets, our method outperforms existing approaches in completely revealed model settings. Additionally, a divergence metric which used as an objective function in this study named *Exponential Divergence Loss* is proposed. This function enhances the defensive capabilities and accuracy of AST compared to KL divergence. Nonetheless, additional investigation is necessary to enhance this methodology and examine its broader impacts, specifically for its computational efficiency and potential generalizability across diverse network structures. Additionally, there is room for further exploration to decrease the computational time required for generating adversarial examples, thereby expediting the overall AST training process. One potential avenue involves leveraging the transferable characteristics of adversarial examples, particularly those synthesized from smaller networks. It would be worthwhile to investigate the impact of training AST with adversarial examples generated from a subset of the original dataset and involving a limited set of adversarial images.

Additionally, the training of AST is highly dependent on the hyperparameters of the loss function, and the number of these hyperparameters is considerable. The optimal performance of AST relies on both accuracy and defensiveness. To find the best settings,

we created a new measure using the accuracies of both the teacher and student models. We calculated defensiveness (D) by subtracting the student's accuracy from the teacher's accuracy. Performance deterioration (P) was found by comparing the baseline teacher model's accuracy with the AST teacher model's accuracy. Our measure is the ratio of D to P, and we chose the model with the highest D/P value.

However, searching through these parameters depends on the accuracy of both the teacher and the student models, so careful calibration requires significant time and computational resources. Adjusting these parameters necessitates extensive research. In future work, the number of parameters could be reduced, and the selection process for these parameters could be thoroughly examined. Moreover, the generalizability of the proposed model, especially on datasets with a high number of classes, is another area for further research.

In summary, this study presents a network that is resilient to model stealing via knowledge distillation. It explores the influence of adversarial examples on defensive strategies, contributing to a more comprehensive understanding of neural networks.

REFERENCES

[1] Eda Yilmaz and Hacer Yalim Keles. Adversarial sparse teacher: Defense against distillation-based model stealing attacks using adversarial examples. arXiv preprint arXiv:2403.05181, **2024**.

[2] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. International Journal of Computer Vision, 129:1789–1819, **2021**.

[3] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In Conference on Neural Information Processing Systems (NIPS). **2015**.

[4] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 3712–3721. **2019**. doi:10.1109/ICCV.2019.00381.

[5] Nikos Komodakis and Sergey Zagoruyko. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In ICLR. **2017**.

[6] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. arXiv preprint arXiv:1810.00069, **2018**.

[7] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, **2013**.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In International Conference on Learning Representations (ICLR). **2015**.

[9] Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial logit pairing. CoRR, abs/1803.06373, **2018**.

[10] Marius Mosbach, Maksym Andriushchenko, Thomas Trost, Matthias Hein, and Dietrich Klakow. Logit pairing methods can fool gradient-based attacks. arXiv preprint arXiv:1810.12042, **2018**.

[11] Daryna Oliynyk, Rudolf Mayer, and Andreas Rauber. I know what you trained last summer: A survey on stealing machine learning models and defences. ACM Computing Surveys, 55(14s):1–41, **2023**.

[12] Smitha Milli, Ludwig Schmidt, Anca D Dragan, and Moritz Hardt. Model reconstruction from model explanations. In Proceedings of the Conference on Fairness, Accountability, and Transparency, pages 1–9. **2019**.

[13] Haoyu Ma, Tianlong Chen, Ting-Kuei Hu, Chenyu You, Xiaohui Xie, and Zhangyang Wang. Undistillable: Making a nasty teacher that cannot teach students. In International Conference on Learning Representations (ICLR). **2021**.

[14] Haoyu Ma, Yifan Huang, Hao Tang, Chenyu You, Deying Kong, and Xiaohui Xie. Sparse logits suffice to fail knowledge distillation. In International Conference on Learning Representations (ICLR). **2022**.

[15] Zi Wang, Chengcheng Li, and Husheng Li. Adversarial training of anti-distilled neural network with semantic regulation of class confidence. In 2022 IEEE International Conference on Image Processing (ICIP), pages 3576–3580. IEEE, **2022**.

[16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, **2016**.

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations (ICLR). **2018**.

[18] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 535–541. **2006**.

[19] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 5191–5198. **2020**.

[20] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. **2015**.

[21] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embedding. IEEE Trans. Neural Networks Learn. Syst., 32(1):25–35, **2021**. doi:10.1109/TNNLS.2020.2970494.

[22] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 3967–3976. **2019**.

[23] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Heterogeneous knowledge distillation using information flow modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2339–2348. **2020**.

[24] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge

distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 5007–5016. **2019**.

[25] Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. Knowledge distillation from internal representations. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 7350–7357. **2020**.

[26] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. IEEE transactions on pattern analysis and machine intelligence, 44(6):3048–3068, **2021**.

[27] Mengya Gao, Yujun Shen, Quanquan Li, and Chen Change Loy. Residual knowledge distillation. arXiv preprint arXiv:2002.09168, **2020**.

[28] Shu Changyong, Li Peng, Xie Yuan, Qu Yanyun, Dai Longquan, and Ma Lizhuang. Knowledge squeezed adversarial network compression. arXiv preprint arXiv:1904.05100, **2019**.

[29] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4933–4942. **2019**.

[30] Linfeng Zhang, Xin Chen, Xiaobing Tu, Pengfei Wan, Ning Xu, and Kaisheng Ma. Wavelet knowledge distillation: Towards efficient image-to-image translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12464–12474. **2022**.

[31] Xiaoyu Yang, Qiujia Li, Chao Zhang, and Philip C Woodland. Knowledge distillation from multiple foundation models for end-to-end speech recognition. arXiv preprint arXiv:2303.10917, **2023**.

[32] Sungho Shin, Joosoon Lee, Junseok Lee, Yeonguk Yu, and Kyoobin Lee. Teaching where to look: Attention similarity knowledge distillation for low resolution face recognition. In European Conference on Computer Vision, pages 631–647. Springer, **2022**.

[33] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In 25th USENIX security symposium (USENIX Security 16), pages 601–618. **2016**.

[34] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 4954–4963. **2019**.

[35] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. arXiv preprint arXiv:1910.12366, **2019**.

[36] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. arXiv preprint arXiv:1710.07535, **2017**.

[37] Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In Proceedings of the IEEE/CVF international conference on computer vision, pages 3514–3522. **2019**.

[38] Mingfu Xue, Yushu Zhang, Jian Wang, and Weiqiang Liu. Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations. IEEE Trans. Artif. Intell., 3(6):908–923, **2022**. doi:10.1109/TAI.2021.3133824.

[39] Yuki Nagai, Yusuke Uchida, Shigeyuki Sakazawa, and Shin'ichi Satoh. Digital watermarking for deep neural networks. International Journal of Multimedia Information Retrieval, 7:3–16, **2018**.

[40] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In Proceedings of the 2017 ACM on international conference on multimedia retrieval, pages 269–277. **2017**.

[41] Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. Advances in neural information processing systems, 32, **2019**.

[42] Jie Zhang, Dongdong Chen, Jing Liao, Weiming Zhang, Gang Hua, and Nenghai Yu. Passport-aware normalization for deep model protection. Advances in Neural Information Processing Systems, 33:22619–22628, **2020**.

[43] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526, **2017**.

[44] Sanjay Kariyappa and Moinuddin K Qureshi. Defending against model stealing attacks with adaptive misinformation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 770–778. **2020**.

[45] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In 2019 IEEE European Symposium on Security and Privacy (EuroS&P), pages 512–527. IEEE, **2019**.

[46] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In Artificial intelligence safety and security, pages 99–112. Chapman and Hall/CRC, **2018**.

[47] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. International Conference on Learning Representations (ICLR), **2016**.

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

[49] Lucas Hansen. Tiny imagenet challenge submission. CS 231N, 5, **2015**.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pages 770–778. **2016**.

[51] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on computer vision (ECCV), pages 116–131. **2018**.

[52] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, **2014**.