UNIVERSITY OF TECHNOLOGY SYDNEY

A Study of Privacy Attacks and Defences in Deep Learning

by Guangsheng Zhang

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Bo Liu, Tianqing Zhu

University of Technology Sydney
Faculty of Engineering and Information Technology

October 2024

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, **Guangsheng Zhang**, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Computer Science, Faculty of Engineering and Information Technology, at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

SIGNATURE: Production Note: Signature removed prior to publication.

DATE: October, 2024

PLACE: Sydney, Australia

Dedication

To myself . . .

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my supervisors, Prof. Bo Liu, Prof. Tianqing Zhu, Prof. Ming Ding, and Prof. Wanlei Zhou, for their invaluable guidance, unwavering support, and profound expertise. Their mentorship has been instrumental in shaping both this research and my growth as a scholar.

I would like to extend a special thanks to Prof. Bo Liu, who served as my principal supervisor. His expertise in privacy and security and his generous commitment of time and resources were crucial in overcoming key challenges in my work. His encouragement and belief in my abilities gave me the confidence to push the boundaries of my research.

I am deeply grateful to my collaborator and close friend, Huan Tian. His contributions were pivotal to the success of my top conference and journal paper acceptances. Our stimulating discussions and his insightful feedback significantly enhanced the quality and scope of this work. Huan's unwavering support, both as a colleague and a friend, has been invaluable throughout this journey.

I would like to extend my appreciation to my laboratory colleagues for their help and support in both research and everyday life. Your assistance have made this experience richer and more rewarding.

Finally, my heartfelt thanks go to my parents for their understanding and encouragement throughout this challenging journey. Your unwavering belief in me has been a constant source of strength and motivation.

List of Publications

Related to the Thesis :

1. Visual Privacy Attacks and Defenses in Deep Learning: A Survey, **Guangsheng Zhang**, Bo Liu, Tianqing Zhu, Andi Zhou, Wanlei Zhou, *Artificial Intelligence Review*, 2022

2. Label-Only Membership Inference Attacks and Defenses In Semantic Segmentation Models, **Guangsheng Zhang**, Bo Liu, Tianqing Zhu, Ming Ding, Wanlei Zhou, *IEEE Transactions on Dependable and Secure Computing*, 2022

3. PPFed: A Privacy-Preserving and Personalized Federated Learning Framework, **Guangsheng Zhang**, Bo Liu, Tianqing Zhu, Ming Ding, Wanlei Zhou, *IEEE Internet of Things Journal*, 2024

4. How Does a Deep Learning Model Architecture Impact Its Privacy? A Comprehensive Study of Privacy Attacks on CNNs and Transformers, **Guangsheng Zhang**, Bo Liu, Huan Tian, Tianqing Zhu, Ming Ding, Wanlei Zhou, in *USENIX Security Symposium*, 2024

Others :

5. When Fairness Meets Privacy: Exploring Privacy Threats in Fair Binary Classifiers via Membership Inference Attacks, Huan Tian, **Guangsheng Zhang**, Bo Liu, Tianqing Zhu, Ming Ding, Wanlei Zhou, in *International Joint Conference on Artificial Intelligence*, 2024

Abstract

Deep learning has emerged as a transformative technology, enabling major breakthroughs across domains like computer vision and natural language processing. However, its massive data requirements raise significant privacy concerns. Understanding and mitigating these privacy vulnerabilities is essential. This thesis investigates three key areas related to the privacy risks of deep learning models. First, it proposes a label-only membership inference attack framework targeting semantic segmentation models, demonstrating higher attack performance than prior work and discussing potential defenses. Second, it introduces a unified federated learning framework to address privacy preservation and personalization challenges simultaneously. This framework outperforms existing methods in protecting privacy against gradient inversion attacks while enabling personalization across heterogeneous client data. Third, the thesis explores how model architecture impacts privacy vulnerability by studying CNNs and Transformers. It identifies specific architectural designs that make models vulnerable to attacks like membership inference, attribute inference, and gradient inversion. The findings provide valuable insights into mitigating privacy risks in deep learning through architectural design principles, defenses against specific attacks, and new privacy-preserving training paradigms. In summary, this thesis offers an understanding of deep learning privacy vulnerabilities and potential solutions.

Table of Contents

List of Figures

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Machine learning, especially deep learning, has made remarkable progress in the 21st century, leading to significant breakthroughs in various fields such as computer vision, natural language processing, medical research, and the natural sciences. These systems perform complex tasks by predicting and classifying input data, leveraging large-scale datasets for training.

However, the private information in these training datasets can be inadvertently leaked from the trained models, posing a significant threat to data privacy. This concern has motivated researchers to study techniques for extracting sensitive information from deep learning systems, known as privacy attacks, and to develop countermeasures to mitigate such attacks, termed privacy defenses.

This thesis analyzes privacy issues in deep learning, emphasizing the importance of building trustworthy machine-learning systems for real-world applications. It explores privacy attacks, which refer to techniques used by adversaries to extract sensitive information from deep learning models or their outputs, and privacy defenses, which are methods designed to mitigate such attacks and preserve the privacy of training data or model parameters.

The primary goal of the thesis is to shed light on the trustworthiness of machine learning systems, with a particular focus on model privacy in deep learning. Privacy concerns exist at different stages of the deep learning pipeline, including the training and inference stages, and in various system architectures, such as centralized and distributed systems. Privacy leakage in these domains has led to the development of attack approaches by malicious adversaries and defense mechanisms by privacy protectors.

To address these challenges, this thesis proposes several new methods and evaluations concerning privacy attacks and defenses in deep learning, highlighting real-world scenarios where privacy breaches can occur. Here are several scenarios in which privacy leakage may happen:

- Online photo sharing on social networks. Users often upload personal images without adequate protection, making it possible for adversaries to extract sensitive information from these photos, such as identities, locations, or activities.

- Public databases and datasets. Public databases and datasets commonly used for training machine learning models may inadvertently contain private or sensitive information that attackers could exploit.

- Deep learning applications in various domains. In healthcare, finance, and government domains, adversaries may attempt to extract private data from the models or their outputs, posing significant risks to individual privacy and data confidentiality.

In these scenarios, attackers leverage vulnerabilities in the deep learning pipeline, including model architectures, training procedures, or inference mechanisms, to extract personal information or sensitive data from the models or their outputs. These scenarios underscore the need for robust privacy defenses.

The thesis aims to analyze privacy leakage in various parts of the deep learning pipeline. In Chapter 3, we investigate privacy leakage in model predictions by proposing a new label-only membership inference attack pipeline for semantic segmentation models and evaluating its effectiveness against several defense mechanisms. In Chapter 4, we analyze privacy leakage in model gradients by proposing a federated learning framework to achieve privacy preservation and personalization, defending against gradient inversion attacks. In Chapter 5, we systematically analyze privacy leakage in model architectures, examine various privacy attacks against convolutional neural networks (CNNs) and Transformers, and demonstrate how certain architectural modules and micro designs can impact model privacy.

Through developing novel privacy attack and defense techniques, our work contributes to the research community by enhancing the understanding of trustworthiness in machine learning systems and addressing critical privacy concerns in deep learning applications.

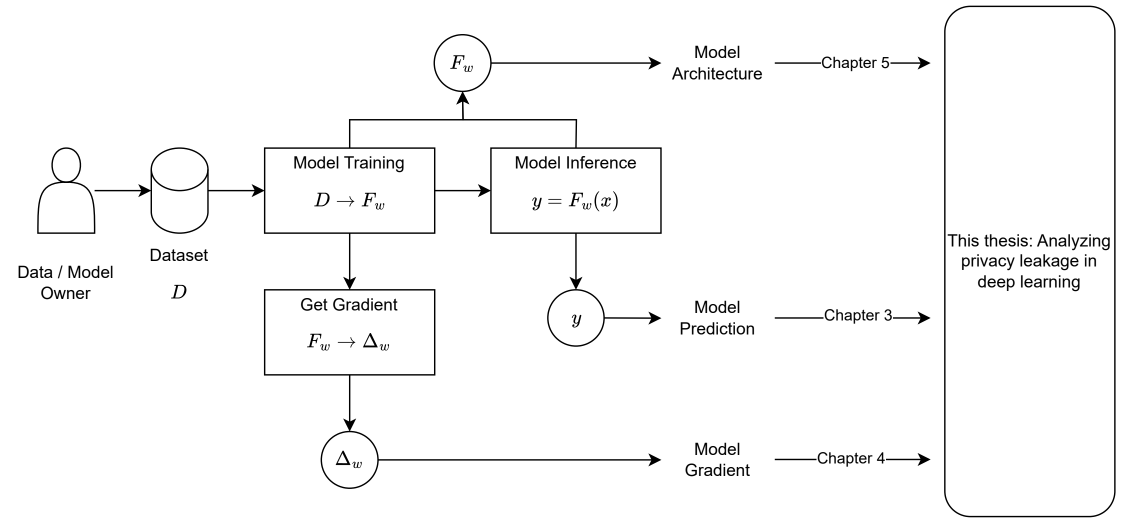


Figure 1.1: Overview of the three research directions investigated in this thesis.

1.1 Analysis of Privacy Leakage in Deep Learning

To provide a common basis for the research directions explored in this thesis, we present a comprehensive framework illustrated in Figure 1.1. Within this framework, we consider a data/model owner with a dataset D. The owner implements a deep learning pipeline: utilizing D, a model F_w is trained to perform specific tasks. This model F_w can compute gradients Δ_w during training. During model inference, an input sample x is processed by the model F_w to generate a prediction y.

Within this framework, we have identified several potential points of privacy leakage: the model's predictions y (discussed in Chapter 3), gradients Δ_w (examined in Chapter 4), and model architectures F_w (analyzed in Chapter 5). To maintain practical assumptions, we investigate the privacy leakage in model gradients, specifically within the context of federated learning in Chapter 4. This approach is taken because, typically, an adversary cannot access the model's gradients except in federated learning scenarios. In contrast, we examine the privacy leakage in the model's predictions and architectures within a centralized learning setting.

In the following subsections, we provide the essential background of the research. Then, we detail our comprehensive analysis and contributions in each chapter.

1.1.1 Analyzing Privacy Leakage in Model Predictions

In many real-world scenarios, such as machine learning as a service (MLaaS), users receive prediction results from well-trained deep learning models without direct access to the training data or model parameters. However, when these models are trained on sensitive information, there is a risk of privacy leakage through the prediction outputs. Membership inference attacks exploit this vulnerability by determining whether a specific data sample belongs to the training dataset based solely on the model's predictions.

Although attackers cannot access the training data directly, they can construct an attack model by querying the victim model and extracting information from its predictions. The pioneering work by [145] formulated a framework for membership inference attacks.

Membership inference attacks typically operate in a black-box setting, utilizing the victim model's predictions to construct the attack model. Most previous research has relied on the confidence scores of the victim model's predictions to launch the attacks [141, 145]. However, some works have relaxed the assumptions by using only the prediction labels [28, 100], demonstrating the vulnerability of various deep learning models to membership inference attacks [19, 66, 130].

Although extensive research has been conducted on membership inference attacks, we argue that unexplored domains still need investigation. This thesis addresses the problem of label-only membership inference attacks and defenses in semantic segmentation models, as presented in Chapter 3. We ask the following research questions: Given only the victim model's prediction labels, can membership inference attacks still succeed in semantic segmentation models? How do these attacks respond to current defense mechanisms?

This work has significant potential for real-world applications, as semantic segmentation is a prominent task in computer vision, drawing great attention from both research and industry. It has been widely applied in domains such as autonomous driving [31], robot navigation [211], and medical data segmentation [136]. Unprotected semantic segmentation models pose severe privacy leakage risks, and when deployed in industry, these models can often only be queried with prediction label outputs.

While the literature on membership inference attacks has grown in recent years, several challenges remain. Firstly, real-world applications typically demand minimal adversarial knowledge for the attacks to be meaningful. Secondly, semantic segmentation is more complex than image classification, and how these models react to membership inference attacks is still unknown. Thirdly, although researchers have developed some defense mechanisms against membership inference attacks, their effectiveness in semantic segmentation applications remains largely unexplored.

To address these challenges, we have developed a membership inference attack and defense framework for semantic segmentation models in Chapter 3. By leveraging various data augmentations on the input images and employing several post-processing strategies on the model's prediction outputs, we can amplify the membership signals and differentiate member samples from non-member samples. This approach demonstrates the vulnerability of semantic segmentation models to label-only membership inference attacks. It highlights the need for robust defense mechanisms to protect user privacy in real-world applications.

1.1.2 Analyzing Privacy Leakage in Model Gradients

In federated learning settings, clients send their trained models' parameters or gradients to a central server, aggregating this information and returning a global model [118]. Each client has a local training dataset that the server cannot access. However, recent research has demonstrated that private information in these local datasets can be leaked via the model's gradients if the server is malicious.

Gradient inversion attacks aim to reconstruct the original training data samples from the shared gradients in federated learning by optimizing the reconstructed gradients to match the shared gradients [52, 219]. While many studies have improved these attacks by generating high-definition data samples [192] or initiating attacks on large datasets [59], the issue of balancing privacy and utility in a federated learning paradigm remains unresolved.

In this thesis, we ask the following research question: How can we defend against gradient inversion attacks based on model gradients while preserving both privacy and personalization in federated learning settings?

Addressing this problem is essential due to the urgent necessity of both privacy preservation and personalization for deep learning models in federated learning scenarios. Model privacy has already been emphasized as a critical concern. Additionally, model personalization is in demand because the global model received from the server is not customized for the data of individual clients. A personalized model can achieve higher performance than a global model, making it crucial to balance privacy and personalization.

Constructing a deep learning model with both privacy preservation and personalization is a major challenge. For privacy preservation, existing approaches like differential privacy [53, 144], data encryption [122], and data compression [154, 219] often require high computation loads or result in lower model utility. For personalization, current methods achieve this by retraining global models [3] or dissecting models [30], but there is no consensus on how these mechanisms can guarantee a personalized model.

To tackle this challenge, we propose a federated learning framework that exploits the internal representations of deep learning models and sends only a part of the model gradients to the server. Our key insight is that layers containing private information also contain personalized information, indicating that privacy and personalization are two facets of a single challenge in deep learning models. By carefully managing the gradient-sharing process, we demonstrate that it is possible to achieve privacy preservation and personalization in federated learning settings simultaneously.

Our thesis in Chapter 4 focuses on gradient inversion attacks and proposes new defense methods for federated learning. Our framework provides both a privacy-preserving model and a personalized model for federated learning, addressing the limitations of existing approaches. This contribution advances the field by enhancing privacy-preserving and personalized deep learning in distributed settings, benefiting the research community and practical applications.

1.1.3 Analyzing Privacy Leakage in Model Architectures

Deep learning models have achieved remarkable success across various applications, but their vulnerability to privacy attacks raises significant concerns. Research has revealed that deep neural networks are susceptible to several types of privacy attacks. Membership inference attacks determine whether a given data sample was part of the model's training dataset based on the model's predictions or outputs [141, 145]. Attribute inference attacks infer sensitive attributes of training data samples from the model's representations [120, 149]. Gradient inversion attacks leverage shared model gradients in federated learning settings to reconstruct original training data samples, compromising client data privacy [52, 219].

Overfitting is a potential factor contributing to privacy leakage in deep learning models. When a model overfits training data, it may inadvertently memorize specific patterns or features unique to individual samples, leaking private information. This phenomenon facilitates privacy attacks, as overfitted models often show higher confidence or distinct behavior for training samples than unseen data [65, 145]. Interestingly, the effectiveness of privacy attacks varies across different model architectures, even when models exhibit similar levels of overfitting. This observation suggests that architectural choices may influence a model's vulnerability to privacy leakage beyond the impact of overfitting alone.

In this thesis, we investigate two key questions: Does a deep learning model architecture affect model privacy? How do architectural designs influence a model's vulnerability to privacy attacks? By addressing these questions, we aim to understand the relationship between model architectures and privacy leakage comprehensively.

Understanding the impact of model architectures on privacy leakage is crucial for real-world applications of deep learning. As these models are increasingly deployed in sensitive domains like healthcare, finance, and government organizations, ensuring data privacy and security becomes critical. Analyzing the relationship between model architectures and privacy vulnerabilities can provide insights into the root causes of privacy leakage, guide the development of more effective defense mechanisms, inform the design of inherently privacy-preserving architectures, and help create architectural modifications to mitigate privacy risks. As deep learning models grow more complex and diverse, understanding architectural factors influencing privacy leakage becomes essential for proactively identifying vulnerabilities in emerging model architectures, addressing potential privacy risks in advance, and developing robust and secure deep learning systems for the future. This knowledge is necessary to prevent severe consequences of privacy breaches in deep learning systems and ensure their responsible deployment across various domains.

Evaluating the impact of model architectures on privacy leakage requires a comprehensive analysis of various architectural choices. However, the majority of research in the field of privacy attacks focuses on developing novel attack strategies or defensive mechanisms. No established methodology exists for evaluating privacy leakage in architectural designs. Moreover, it is essential to consider a diverse range of privacy attacks, as the performance of a single attack method cannot fully represent a model architecture's vulnerability to other attacks.

In Chapter 5, we investigate whether model architectures affect model privacy by systematically analyzing two mainstream architectures: convolutional neural networks (CNNs) and Transformers. We evaluate their performance under various state-of-the-art privacy attacks, including membership inference, attribute inference, and gradient inversion attacks. By comparing the vulnerability of CNNs and Transformers to these attacks, we aim to identify architectural differences that may contribute to privacy leakage or resilience. This analysis provides insights into the relationship between model architectures and privacy leakage, contributing to developing more robust and privacy-preserving deep learning models. Our findings can guide the design of inherently resilient architectures and inform the development of architectural modifications and defense mechanisms to mitigate privacy risks.

1.2 Outline

This section outlines the contents of the thesis.

Chapter 2: Literature Review. This chapter introduces attack methods and corresponding defense mechanisms relevant to the thesis. We present an overview of membership inference attacks, attribute inference attacks, and gradient inversion attacks, followed by a discussion of multiple defense mechanisms, including differential privacy and attack-specific defenses. These attack methods and defense mechanisms are foundational to the subsequent chapters, which propose and evaluate various attacks and defenses. The content of this chapter is based on the Artificial Intelligence Review 2022 publication *Visual Privacy Attacks and Defenses in Deep Learning: A Survey* [204], with Guangsheng Zhang as the first author.

Chapter 3: Label-Only Attacks Against Semantic Segmentation. This chapter addresses privacy leakage in model predictions. We list the contributions as follows:

- We propose a new membership inference attack framework for semantic segmentation models under label-only settings.
- We design a framework for these attacks by applying different data augmentations to the data samples and employing several post-processing strategies on the victim model's prediction output.
- We evaluate our attack framework under label-only assumptions and test its effectiveness in semantic segmentation tasks.
- Additionally, we discuss several defense mechanisms and achieve competitive experimental results.
- Through this work, we demonstrate the vulnerability of semantic segmentation models to membership inference attacks and highlight the need for robust defense strategies to protect user privacy in real-world applications.

The content of this chapter is based on the IEEE TDSC 2022 publication *Label-Only Membership Inference Attacks and Defenses in Semantic Segmentation Models* [202], with Guangsheng Zhang as the first author.

Chapter 4: Privacy-Preserving and Personalized Federated Learning. This chapter focuses on privacy leakage in model gradients. We list the contributions as follows:

- We propose a privacy-preserving federated learning framework leveraging personalization mechanisms, which can address privacy leakage issues from model gradients and the need for effective model personalization methods.
- Our proposed framework ensures privacy preservation, especially against gradient inversion attacks, by splitting the model into global layers and local layers. Additionally, we inspect the threshold of splitting global layers from local layers and reveal that setting the model head as local layers is sufficient to ensure robust privacy preservation.
- From the personalization standpoint, our framework ensures model personalization by training local layers separately on each client. We also verify that employing multiple local epochs and multiple fine-tuning epochs contributes to improving the personalization performance.
- Our extensive experiments demonstrate that our framework outperforms other defense mechanisms and, at the same time, matches the personalization performance compared with other federated learning frameworks.
- Through this work, we provide a comprehensive solution that addresses the critical challenges of privacy leakage in model gradients and the need for effective personalization in federated learning settings.

The content of this chapter corresponds to the IEEE Internet of Things Journal 2024 publication *PPFed: A Privacy-Preserving and Personalized Federated Learning Framework* [203], with Guangsheng Zhang as the first author.

Chapter 5: How Does a Model Architecture Impact Its Privacy? This chapter analyzes privacy leakage in model architectures. We list the contributions as follows:

- We evaluate the privacy vulnerabilities of CNNs and Transformers using three prominent attack methods: membership inference, attribute inference, and gradient inversion attacks.
- Our analysis reveals that Transformers exhibit higher vulnerabilities to these privacy attacks than CNNs.
- We identify key factors contributing to the enhanced resilience of CNNs: the design of activation layers, stem layers, and LN layers.

- We also discover that the presence of attention modules in Transformers could make them susceptible to privacy attacks.
- We propose solutions to mitigate these vulnerabilities, including modifying model components and adding perturbations as defense mechanisms.

The content of this chapter is based on the USENIX Security Symposium 2024 publication *How Does a Deep Learning Model Architecture Impact Its Privacy? A Comprehensive Study of Privacy Attacks on CNNs and Transformers* [201], with Guangsheng Zhang as the first author.

Chapter 6: Conclusion. The final chapter summarizes the analysis and findings presented in the thesis. It concludes by highlighting the thesis's contributions to the research community.

To maintain readability, each chapter is organized in a self-contained format. Some content, such as definitions and related work, is introduced in the relevant chapters for context.

Chapter 2

Literature Review

In this chapter, we review popular attack methods and their corresponding defense mechanisms. We demonstrate that deep learning serves as both a potent privacy attack tool and a promising preservation technique. [1]

2.1 Attack Methods

The following subsections focus on membership inference, attribute inference, and gradient inversion attacks, as these methods are employed in our later chapters. As machine learning techniques continue to evolve and are applied in many applications, understanding and mitigating the privacy risks posed by these attacks becomes increasingly critical.

2.1.1 Membership Inference Attacks

Deep learning models trained on sensitive data are vulnerable to various attacks, including membership inference attacks, which aim to determine if a specific data sample was part of the model's training dataset. Even without direct access to the training data or model, attackers can query the model and use its predictions to infer membership status. Shokri *et al*. [145] pioneered the concept of membership inference attacks. Their approach treats the targeted deep learning model (the victim model) as a black box, where the

[1]The content of this chapter is based on [204]. Guangsheng Zhang was the first author of this paper.

attacker obtains only prediction confidence scores through queries. To mimic the victim model's behavior, the attacker generates multiple shadow models using synthetic data sampled from the same distribution as the victim model's training data. The key insight is that samples from the training set typically yield higher prediction confidence scores than those not present. The attacker can classify samples as members or non-members of the victim model's training data based on these confidence scores by training an attack model on the shadow models' outputs.

Membership inference attacks pose a significant privacy risk, making their mitigation crucial for maintaining user privacy and building trustworthy machine-learning systems. The following subsections explore categories of membership inference attacks, their adaptations to specific tasks, and the internal mechanisms and principles governing their effectiveness. This overview will provide insight into the nature of these attacks and lay the groundwork for developing effective countermeasures.

2.1.1.1 Types of Membership Inference Attacks

Membership inference attacks can be executed using various approaches, depending on the adversary's knowledge of the target model. The following paragraphs introduce these approaches based on the adversary's access, input, and model knowledge.

Attacks based on the adversary's access knowledge. Membership inference attacks can be classified into white-box and black-box attacks based on the adversary's level of access to the victim model. In white-box attacks, the adversary has complete access to the victim model's information, including training data distribution, samples, strategy, parameters, gradients, or prediction outputs. These attacks have been analyzed in federated learning settings [125], where such access is more realistic. In black-box attacks, the adversary can only access the victim model's prediction outputs without additional information about its internals or training data. Most research has focused on black-box attacks [141, 145, 151] due to their feasibility with limited model information. However, some research suggests that optimized black-box attacks may provide comparable information to white-box attacks [139].

Attacks based on the adversary's input knowledge. Membership inference attacks can be categorized into score-based, loss-based, and label-only attacks based on the victim model's output used as input for the attack model. Score-based attacks leverage the victim model's prediction scores, exploiting the tendency for member data to have higher scores [141, 145]. Loss-based attacks utilize prediction losses, as member data typically have smaller losses than non-member data [191]. Label-only attacks require only prediction labels, a more realistic scenario when scores and losses are unavailable. These methods [28, 100] apply data augmentations to the original input and use corresponding prediction labels as membership signals.

Attacks based on the adversary's model knowledge. Membership inference attacks can be categorized into network-based, metric-based, and likelihood-based attacks based on the attack model type. Network-based attacks train a neural network to distinguish between member and non-member samples using the victim model's prediction output [124, 141, 145]. Metric-based attacks use predefined metrics or thresholds to differentiate members from non-members based on the victim model's output. Metrics include prediction correctness [95], correct class confidence [150], or prediction entropy [145]. Likelihood-based attacks calculate likelihood ratios to determine membership status. The adversary computes the likelihood of a sample belonging to the member versus non-member set and classifies based on this ratio [15, 189].

2.1.1.2 Membership Inference Attacks for Different Machine Learning Tasks and Paradigms

Membership inference attacks have been extensively studied in classification models, where a significant portion of research has focused on exploring these attacks and their implications. However, recent research has expanded their applications to other tasks and paradigms, highlighting broader privacy risks in machine learning systems.

Attacks for other tasks. Researchers have extended membership inference attacks beyond classification models. Semantic segmentation and object detection models have shown vulnerabilities [66, 130]. Pre-trained embedding models are susceptible to attacks for infrequent training data inputs [148]. Recommender systems' privacy leakage has been quantified through user-level attacks based on recommended items [207]. Causal models' vulnerability to these attacks reveals a connection between generalization and attacks via shared causal factors [9]. Semi-supervised learning models, while generalizing well to the testing data, show vulnerability through confident predictions on training data [64]. Privacy risks of neural architecture search (NAS) architectures were measured systematically in [77], showing NAS-searched architectures were usually more robust than human-designed ones. They also identified cell patterns impacting attack performance. A study on multi-exit networks found them less vulnerable to membership inference attacks and proposed a hybrid attack exploiting exit information to improve performance [99]. An analysis of privacy risks in neural network pruning used the impact on prediction divergence to propose a self-attention membership inference attack against pruned networks [197]. Hu *et al.* [72] demonstrated membership inference attacks via a backdoored model, which behaved differently from a clean one on deliberately marked samples created by the data owner. Explainable machine learning methods could lead to increased privacy risks. A study conducted in [105] used additional membership features from changes guided by attribution maps in explanation models. Gao *et al.* [49] quantified personal information leakage risk in person re-identification models, proposing an attack method based on inter-sample similarity distribution between training and test sets. A study revealed that existing network partition solutions for On-Device models remain vulnerable to membership inference attacks [212]. Language model training data could be extracted using membership inference [17], with larger models showing greater vulnerability. Chen *et al.* [20] introduced a speaker-level membership inference attack for speaker recognition systems, using intra-similarity and inter-dissimilarity objectives to distinguish between training and non-training speakers. Hu *et al.* [75] proposed multi-modal membership inference attacks for image and text data tasks. More practical attacks on large-scale multi-modal pre-trained models were studied in [87]. These studies further illustrate the diverse landscape of privacy vulnerabilities across various machine-learning applications and modalities.

Attacks for other paradigms. Membership inference attacks have been studied beyond traditional centralized machine-learning settings. Federated learning systems, where models are trained on decentralized data from multiple sources, have been examined [120, 125, 166]. Online learning scenarios, with models continuously updated using new data, have also been explored [140]. Generative models have received attention, with studies on Generative Adversarial Networks (GANs) vulnerability to these attacks [19, 60]. Like classification models, GAN generators can implicate privacy risks by leaking training data information. Diffusion-based generative models' vulnerability was demonstrated in [38] through a query-based attack assessing forward process posterior estimation matching at each timestep. This attack showed precise membership inference on standard and text-to-image diffusion models. Transfer learning paradigms, where pre-trained models are fine-tuned on specific tasks, have been investigated for potential privacy leaks through membership inference attacks [222]. Contrastive learning models also face membership inference vulnerabilities. He *et al.* [65] found these models less vulnerable due to reduced overfitting. However, Liu *et al.* [106] showed that image encoders pre-trained by contrastive learning remained vulnerable to attacks due to overfitting. Studies in [138] explored membership inference attacks in deep ensemble learning, revealing a trade-off between accuracy and privacy. These findings underscore the persistent privacy challenges across various machine learning paradigms.

2.1.1.3 Improving the Performance of Membership Inference Attacks

Building on our discussion of various membership inference attacks across different tasks and paradigms, we now explore approaches to enhance their effectiveness. These methods leverage additional data or model information, integrate other machine learning techniques, and introduce new attack metrics to better evaluate and present attack performance.

Leveraging additional information. Long *et al.* [113] proposed an evaluation methodology for assessing privacy risk at the individual level, revealing highly vulnerable records even when aggregate attack precision was near baseline. Shafran *et al.* [142] demonstrated an increased vulnerability in models with higher dimensional inputs/outputs and proposed using predictability error to achieve high attack accuracy for image translation and semantic segmentation tasks. Hui *et al.* [79] improved attacks by extracting membership semantics via a differential comparison between the original dataset and a generated non-member set. Watson *et al.* [173] introduced difficulty calibration, adjusting predicted scores based on classification difficulty, significantly reducing false positive rates without compromising accuracy. Bertran *et al.* [11] developed attacks based on quantile regression on non-member data point confidence score distributions, achieving competitive performance with less computation by training only a single shadow model.

Integrating other techniques. Song *et al.* [152] revealed that models designed to defend against adversarial examples are vulnerable to membership inference attacks. Chen *et al.* [26] investigated data poisoning attacks to increase membership privacy risks of benign training samples, proposing dirty-label attacks for supervised learning and clean-label optimization-based attacks for transfer learning scenarios. Liu *et al.* [107] developed an attack method exploiting membership information through knowledge distillation, leveraging losses from intermediate distilled models in addition to the target model's losses, demonstrating increased effectiveness.

Discovering new attack metrics. Some researchers [137] noted that attack performance in prior studies could be misleading due to high false positive rates. Using prediction scores for attacks was found unreliable on known domains and out-of-distribution data due to high false-positive rates [68]. Carlini *et al.* [15] argued that current metrics like accuracy and AUC were insufficient for evaluating attacks. They proposed using true positive rates at low false positive rates and full ROC curves to better illustrate attack results. Their likelihood-based attacks better exposed privacy leakage for hard samples. Similarly, Ye *et al.* [189] evaluated attacks using true positive rates at low false positive rates and designed stronger membership inference attacks.

2.1.1.4 Why Membership Inference Attacks Work?

The effectiveness of membership inference attacks can be attributed to four main factors: overfitting in deep learning models, data distribution differences between member and non-member samples, the influence of data augmentations, and privacy leakage through model memorization. These factors contribute to the vulnerability of machine learning models to membership inference attacks.

Overfitting in deep learning models is a key factor enabling membership inference attacks. A model typically achieves better accuracy on its training dataset than the testing dataset, creating a vulnerability. Yeom *et al.* [191] showed that overfitting alone is sufficient for an attacker to perform an attack. Salem *et al.* [141] found that more overfitted models are more vulnerable to these attacks. Chen *et al.* [19] demonstrated that overfitting also extends attacks to generative adversarial networks. Leino *et al.* [95] observed that overfitting models leak membership signals through their personal use of features. He *et al.* [65] argued that contrastive models are less vulnerable to membership inference attacks than supervised models due to reduced overfitting.

The second key factor is data distribution. Song *et al.* [151] demonstrated a significant divergence between loss distributions of member and non-member samples. Attackers exploit this divergence to train effective attack models and launch membership inference attacks.

The third factor is data augmentation. Applying augmentations to the victim model's training dataset enhances membership signals. Choquette-Choo *et al.* [28] showed that label-only attacks could be launched using data augmentations. Further research by [86, 193] demonstrated that data augmentations could lead to more generalized attack models.

The final factor is model memorization. Carlini *et al.* [16] explored privacy leakage through memorization, introducing the "Onion Effect." They demonstrated that memorization of training data is relative, and removing the most vulnerable outlier points exposes a new layer of previously safe points to the same attack.

Beyond these specific aspects, some researchers have provided comprehensive assessments of membership inference attacks. He *et al.* [63] conducted a thorough evaluation of these attacks. Additionally, papers by [73, 74] offer literature reviews on the subject.

2.1.2 Attribute Inference Attacks

Attribute inference attacks seek to uncover sensitive attributes of data samples by analyzing the output probabilities of a victim model trained on private data. Like membership inference attacks, adversaries have limited access to the victim model's architecture and parameters. However, while membership inference determines if a data point was used in training, attribute inference aims to deduce the value of a sensitive attribute for a given sample.

Early work by [47] formalized attribute inference attacks. In these attacks, an adversary with partial information about a data sample aims to uncover an unknown sensitive attribute. The adversary generates sample variants by filling in possible values for the unknown attribute, queries the victim model with each, and receives prediction outputs. By comparing these predictions to ground truth labels, the adversary attempts to infer the true value of the sensitive attribute.

Attribute inference attacks pose a serious privacy risk with broad implications. Many machine learning models, particularly in sensitive domains like finance, healthcare, and surveillance, may train on data containing private attributes. An adversary capable of attribute inference could potentially extract and exploit this sensitive information, even if unintended by the model owner. Studying these vulnerabilities is crucial for developing robust defenses to protect user privacy and prevent misuse. The following subsections introduce different categories of attribute inference attacks and their applications to specific tasks.

2.1.2.1 Types of Attribute Inference Attacks

This subsection categorizes and describes common attribute inference attacks found in the literature. These attacks are divided into two main categories: those targeting structured data and those targeting unstructured data.

Attacks on structured data. For structured data, the victim model is trained on records containing both non-sensitive and sensitive attributes. With access to non-sensitive attributes and model predictions, the adversary aims to infer sensitive attributes.

Early work by [47] pioneered generating records with varied sensitive attribute values and querying the victim model. Yeom *et al.* [191] adapted membership inference attacks, inferring the sensitive value with the highest membership probability. Similar approaches using prediction results and confidence scores were explored in [119]. In a white-box setting, the authors of [80] monitored specific neurons' activation values to infer sensitive attributes using the model's parameters.

Attacks on unstructured data For unstructured data like images, sensitive attributes are implicitly encoded in the input. The victim model is trained on such data containing hidden sensitive attributes. With access to the model's intermediate features or outputs, the adversary aims to infer these sensitive attributes.

Research in [120, 149] demonstrated attribute inference attacks using the victim model's internal feature representations. The adversary trains an attack model on auxiliary data to recognize sensitive attributes and then uses the victim model's extracted features to predict sensitive attribute values in the original data sample. Song *et al.* [149] attributed such vulnerabilities to the overlearning behavior of deep neural networks, where models unintentionally encode aspects of training data beyond the intended task objective. A study by [117] explored a threat model where a malicious model builder intentionally trained the victim model to encode sensitive attributes by incorporating them into the loss function.

2.1.2.2 Other Research Focus of Attribute Inference Attacks

Research in [65] demonstrated that contrastive learning was more vulnerable to attribute inference attacks due to its expressive data representations. Zhao *et al.* [213] explored approximate attribute inference, a relaxed form of attack. Here, the adversary aimed to find attribute values close to the true values according to a distance metric rather than inferring exact values. These attacks performed significantly better as the target model became more overfitted to the training data. Attribute inference attacks could also extract sensitive attributes from model explanations [39]. While explanation models aimed to increase transparency, they could inadvertently leak private information, even when sensitive attributes were censored from the training data and input.

2.1.3 Gradient Inversion Attacks

Federated learning allows clients to keep training data private while sharing model parameters or gradients with a central server to build a global model. Although the central server cannot directly access local training data, an adversary can execute gradient inversion attacks to reconstruct local training data using the shared model gradients.

Zhu *et al.* [219] first discovered the possibility of extracting private data from publicly shared gradients. In these attacks, an adversary in the central server, with access only to global model gradients, generates random data samples and labels (dummy inputs). After model inference, the adversary obtains dummy gradients. The adversary then optimizes dummy inputs by minimizing the distance between the dummy and original model gradients. Through iteration, dummy inputs increasingly resemble the original training data, eventually reconstructing it.

Gradient inversion attacks pose a significant threat to data privacy in federated learning, undermining its fundamental premise of keeping local data private while collaboratively training a global model. The following subsections introduce different categories of gradient inversion attacks and their applications to specific tasks.

2.1.3.1 Types of Gradient Inversion Attacks

This subsection explores two types of gradient inversion attacks: iteration-based and recursion-based. We introduce representative literature for each category.

Iteration-based attacks. Early reconstruction attacks on federated learning, pioneered by [219], attempted to iteratively recover input data from shared parameter gradients but faced issues with model convergence and label reconstruction. iDLG [214] improved this method, making it applicable to any differentiable model trained with cross-entropy loss over one-hot labels. Geiping *et al.* [52] demonstrated high-resolution image reconstruction from model gradients alone, enhancing the attack with better optimization configurations. Wei *et al*. [176] presented an evaluation framework for assessing client data leakage risks in federated learning. The study in [82] introduced attacks using pre-trained generative models as prior knowledge of data distribution. Lam *et al*. [91] showed that adversaries could disaggregate updated model gradients using device analytics summary information to reconstruct client participant data, initiating inversion attacks. GradInversion [192] recovered large batch images from average training gradients at high resolution, succeeding even for deep networks like ResNet-50 on complex datasets like ImageNet. Li *et al*. [102] showed existing defenses using gradient degradation remained vulnerable to a new generative gradient leakage attack, using GAN latent spaces trained on public datasets to compensate for information loss. Wen *et al*. [177] presented a strategy significantly improving privacy attacks in federated learning, enabling high-fidelity data extraction on large batches without architectural modifications, posing realistic threats in cross-device and cross-silo settings. GIFD [42] disassembled GANs used as priors and optimized over intermediate feature domains, achieving superior pixel-level reconstruction quality and generalizability. Zhang *et al*. [200] demonstrated reconstructing complex, high-resolution images from large batch sizes using an overparameterized convolutional network, achieving high-fidelity image recovery. Ye *et al*. [190] proposed a two-step method using an evolutionary algorithm to reconstruct data labels, then employing a stepwise attack with a dynamically adapted objective function to reconstruct high-resolution images. Hong *et al*. [70] introduced a loss-aware vulnerability proxy replacing gradient norm for optimization, showing superiority in capturing sample vulnerabilities.

Recursion-based attacks. In contrast to iteration-based attacks, recursion-based attacks recursively calculate each layer's input by finding the optimal solution with minimized error [217]. These attacks transform the process into a recursive solving of linear equation systems based on gradient and weight constraints.

While most research follows the iteration-based approach, it's important to highlight the distinction between these two types of gradient inversion attacks.

2.1.3.2 Other Research Focus of Gradient Inversion Attacks

Balunovic *et al*. [8] formalized the gradient leakage problem in a Bayesian framework. This approach enables phrasing a Bayes optimal adversary as an optimization problem, interpreting several prior attacks as approximations of this adversary using different input and gradient distribution assumptions. Fowl *et al*. [45] introduced a new threat model where malicious modifications to the shared model architecture allow a server to directly obtain verbatim copies of user data from gradient updates. This method enables reconstruction even when gradients are aggregated over large batches.

The work in [59] demonstrated vision transformers' (ViTs) vulnerability to gradient inversion attacks, achieving high-fidelity reconstructions close to original hidden data on large datasets like ImageNet1K and MS-Celeb-1M. Lu *et al.* [114] investigated privacy risks in vision transformers, analyzing the self-attention mechanism's vulnerability. They proposed an attack method showing how vision transformers are highly susceptible to revealing private training data from shared gradients.

Some research extended these attacks to natural language processing. Balunovic *et al.* [7] successfully reconstructed the original text by leveraging auxiliary language models and alternating optimization techniques. Their approach surpassed prior methods for batch sizes larger than 1, highlighting substantial privacy risks for textual data in federated learning. Gupta *et al.* [57] recovered the text from large batch sizes by identifying words from gradients and employing beam search with a prior-based reordering strategy. Vero *et al.* [169] focused on attacks for tabular data, leveraging softmax relaxation and pooled ensembling to solve the optimization problem. They also introduced an entropy-based uncertainty quantification scheme to enable human assessment.

Gradient inversion attacks were extended to weight updates by [218]. This approach exposed privacy risks by leveraging a surrogate model method that enhanced attack effectiveness, challenging assumed privacy protections in federated learning. Zhang *et al.* [205] studied the mechanisms of privacy leakage by proposing a novel Inversion Influence Function (I2F). This function establishes a closed-form connection between recovered images and private gradients, offering insights into when and how leakage occurs.

2.2 Defense Mechanisms

Protecting data and model privacy has become a critical concern in deep learning. Various defense mechanisms have been developed to mitigate risks presented in the previous section, ensuring beneficial data use without compromising personal privacy. This section first introduces defense mechanisms based on differential privacy. Then, it covers defense mechanisms designed for specific attacks.

2.2.1 Defense based on Differential Privacy

Differential privacy is a prominent technique for defending against privacy attacks. It provides a formal framework to quantify and limit privacy risks by ensuring that including or excluding a single data point does not significantly affect analysis outcomes, thus protecting individual data points from identification. This is achieved by adding controlled random noise to data or query results, making it difficult for attackers to draw accurate conclusions about individuals, thereby providing strong privacy guarantees.

We discuss differential privacy as a defense mechanism in several topics: methods for adding differentially private noise, differential privacy in teacher-student model frameworks, application to generative models, implementation in federated learning, other aspects of differential privacy, and differential privacy for specific privacy attacks.

2.2.1.1 Adding Differentially Private Noises

In deep learning models, different private noises can be added to model gradients or loss functions. Abadi *et al.* [1] proposed differentially private stochastic gradient descent (DPSGD), which became a standard for guaranteeing privacy in deep learning models. They preserved privacy at each training iteration using gradient clipping and noise addition in the SGD process. They also introduced the moments accountant (MA) mechanism to compute the overall privacy cost during training. Chen *et al.* [25] further analyzed gradient perturbations. Zhao *et al.* [216] presented a collaborative learning system using the functional mechanism to perturb the model's loss function during training to achieve differential privacy.

2.2.1.2 Differential Privacy for Teacher-Student Model

Several studies applied differential privacy in the teacher-student model framework. Papernot *et al.* [128] proposed Private Aggregation of Teacher Ensembles (PATE), a framework with multiple teacher models and a student model. Teachers were trained on sensitive data, while the student predicted output by noisy voting among teachers without accessing teacher data or parameters. This framework provided differential privacy guarantees for the student model. Their improved framework [129] introduced a new noisy aggregation mechanism with tighter differential privacy guarantees. Wang *et al.* [170] presented a private model compression framework using differential privacy and knowledge distillation. Noises were added during knowledge distillation to ensure data privacy in student models, providing a solution for deep learning models on mobile devices with limited capacity. Zhu *et al.* [221] proposed private-kNN, an algorithm with comparable or better accuracy than PATE. Using k-nearest neighbor queries to the private dataset from a random subsample scheme, their design avoided splitting the training dataset.

2.2.1.3 Differential Privacy for Generative Models

Generative models can address data scarcity by generating more samples from the same distribution. However, the learned generative distribution density can easily reflect training samples due to high model dimensions, leading to privacy leakage. Differential privacy can protect privacy during GAN training. Xie *et al.* [182] proposed DPGAN, achieving differential privacy in GANs by adding well-designed perturbations to gradients during training. Jordon *et al.* [85] presented PATE-GAN, applying the PATE framework to protect private training data in GANs. Torkzadehmahani *et al.* [161] introduced DP-CGAN, a Differentially Private Conditional GAN framework to improve model performance while guaranteeing training dataset privacy. Chen *et al.* [18] provided GS-WGAN (Gradient-sanitized Wasserstein GAN), ensuring private data in a sanitized form with privacy guarantees. The generated samples were proven to be both private and of good quality.

2.2.1.4 Differential Privacy for Federated Learning

Differential privacy in federated learning has gained increasing research attention. Shokri *et al.* [144] designed a privacy-preserving system for collaborative deep learning, allowing clients to selectively share model gradients during training. This framework, distributed selective stochastic gradient descent (DSSGD), enabled clients to benefit from others without sharing training data. Differential privacy was applied to DSSGD to prevent privacy leakage, inspiring later research on privacy preservation in federated learning. Geyer *et al.* [53] presented a client-sided differentially private federated optimization algorithm to hide clients' contributions during training while balancing privacy and utility. Truex *et al.* [165] proposed a federated learning system combining differential privacy and secure multiparty computation to protect against inference threats while maintaining high model accuracy. Wei *et al.* [174] introduced NbAFL, a novel federated learning framework employing differential privacy by adding artificial noise to parameters before aggregation, effectively preventing information leakage. Li *et al.* [101] developed SoteriaFL, a unified framework enhancing the communication efficiency of private federated learning through compression, balancing privacy, utility, and communication complexity. Sun *et al.* [155] proposed an efficient version of LoRA for privacy-preserving federated learning, stabilizing performance by fixing randomly initialized non-zero matrices and fine-tuning zero-initialized ones, addressing noise amplification and hyper-parameter sensitivity issues.

2.2.1.5 Differential Privacy for Other Aspects

Some research proposed alternative mechanisms or assumptions for differential privacy in deep learning systems. Yu *et al.* [195] introduced an approach using concentrated differential privacy (CDP) and a dynamic privacy budget allocator for model training, improving privacy loss accounting and training efficiency. Studies in [164] showed that differentially private models required either significantly more private data or features from public data to surpass traditional models. Bu *et al.* [13] proposed automatic clipping, simplifying differentially private training for deep learning models by eliminating the need to tune the clipping threshold. He *et al.* [61] introduced group-wise clipping in differentially private deep learning, demonstrating the benefits of per-layer and per-device clipping for computational efficiency and task performance. Zhang *et al.* [210] proposed error-feedback differential privacy, offering diminishing utility bounds without constant clipping bias and allowing arbitrary clipping thresholds. Wu *et al.* [179] introduced differentially private in-context learning for Large Language Models, generating private responses through noisy consensus among an ensemble of responses based on disjoint exemplar sets.

2.2.1.6 Differential Privacy for Specific Privacy Attacks

Numerous studies have shown that differentially private deep learning models protect against membership inference attacks, although at the cost of reduced model utility [19, 28, 66, 95, 130, 134, 166].

Differential privacy can limit individual sample exposure by adding noise during training, protecting sensitive attributes against attribute inference attacks [80].

It can also be employed against gradient inversion attacks [154], though this often results in decreased model effectiveness.

2.2.2 Defense Mechanisms Against Specific Privacy Attacks

Beyond differential privacy, some research focuses on countering specific privacy attacks. We introduce defenses for membership inference, attribute inference, and gradient inversion attacks.

2.2.2.1 Defenses for Membership Inference Attacks

Tang *et al.* [160] presented a privacy-preserving architecture against membership inference attacks using two key components: an ensemble architecture aggregating outputs from multiple training subsets and a self-distillation strategy to distill the training dataset. Yang *et al.* [188] proposed transforming confidence score vectors from victim models into purified confidence scores indistinguishable in shape, distribution, and prediction label between members and non-members, effectively defending against membership inference attacks. Chen *et al.* [27] introduced a defense technique enforcing less confident predictions on both training and testing samples, achieving strong membership privacy and high task accuracy without requiring extra data.

2.2.2.2 Defenses for Attribute Inference Attacks

One defense approach involves identifying and removing the most vulnerable training records before retraining the model. This minimizes their influence on the model, enhancing privacy [80].

Jia *et al.* [83] proposed AttriGuard, a defense method using adversarial examples. For each representation, the defender generates adversarial examples for each possible sensitive attribute value. They then select one value using a probability distribution and choose the corresponding adversarial example as the new representation.

2.2.2.3 Defenses for Gradient Inversion Attacks

Sun *et al.* [154] proposed Soteria, a defense that perturbs data representation to degrade reconstructed data quality while maintaining FL performance. It significantly enhances privacy by increasing the mean squared error between reconstructed and raw data. Gao *et al.* [50] leveraged data augmentation transformation policies to preprocess sensitive images, making it infeasible for adversaries to reconstruct useful information from corresponding gradients. Huang *et al.* [78] evaluated existing gradient inversion attacks, finding that relaxing some strong assumptions could substantially weaken the attacks. They proposed combining three defense mechanisms to mitigate privacy leakage with minor utility loss. Xue *et al.* [185] presented an enhanced gradient pruning defense method, slightly modifying gradient pruning to provide stronger privacy guarantees while improving communication efficiency and preserving utility. Wu *et al.* [178] introduced a method to obfuscate gradients with synthesized samples mimicking sensitive data at the gradient level, offering strong protection without compromising federated learning performance.

CHAPTER 3

Label-Only Attacks Against Semantic Segmentation

In this chapter, we propose a new membership inference attack method. Recent research has discovered that deep learning models are vulnerable to membership inference attacks, which can reveal whether a sample is in the training dataset of the victim model or not. Most membership inference attacks rely on confidence scores from the victim model for the purpose of the attack. However, a few studies indicate that prediction labels of the victim model's output are sufficient for launching successful attacks. Besides the well-studied classification models, segmentation models are also vulnerable to this type of attack. In this chapter, for the first time, we propose label-only membership inference attacks against semantic segmentation models. With a well-designed framework of the attacks, we can achieve a considerably higher successful attacking rate compared to previous work. In addition, we have discussed several possible defense mechanisms to counter such a threat. [1]

3.1 Background

Deep learning technologies have brought numerous successful applications, such as face recognition [90], image classification [62], and semantic segmentation [112]. The success of these technologies is due to the availability of large-scale datasets. These

[1]The content of this chapter is based on [202]. Guangsheng Zhang was the first author of this paper.

datasets enable a better training process and in turn, more accurate deep learning models. However, the models in deep learning applications often contain sensitive information and pose privacy leakage risks. Although the deep learning model structures are usually hidden, the attackers can still extract private information by making queries to the victim model. The work of [145] demonstrated that attackers could reconstruct some information on the model's training data by identifying the data sample's membership. This type of attack is called the membership inference attack. The attacker was assumed to have black-box access to the victim model to obtain confidence scores of the model prediction after multiple queries. Using the queried confidence scores, the attacker could infer whether a specific data sample was in the training data or not.

A large group of research works focused on studying the design or understanding the membership inference leakage in deep learning models [104, 141, 191]. Defense mechanisms against membership inference attacks have also been developed [84, 145, 166, 187]. However, obtaining confidence scores of the model prediction to launch membership inference attacks is not always practical because the deep learning models deployed in real-world applications usually do not have APIs to be queried with confidence scores, or the prediction scores have already been modified by internal defense mechanisms.

Recent work of [28] and [100] showed that obtaining confidence scores of the victim model is not mandatory to launch a membership inference attack. They made fewer assumptions about the attack by only having the victim model's prediction labels. The intuition is that deep learning models have higher confidence in predicting member samples than non-member samples. By making various data augmentations to the original training data, member and non-member samples have different performances in prediction labels. Then, with the gathered prediction labels as the attack model input data, the attacker differentiates the member samples from non-member samples.

Previous research mostly concentrated on membership inference attacks in image classification models [141], or generative models [19]. Researchers of [66] demonstrated that membership inference attacks could also target other models, such as semantic segmentation models. However, such an initial work assumed that attacks should require confidence scores.

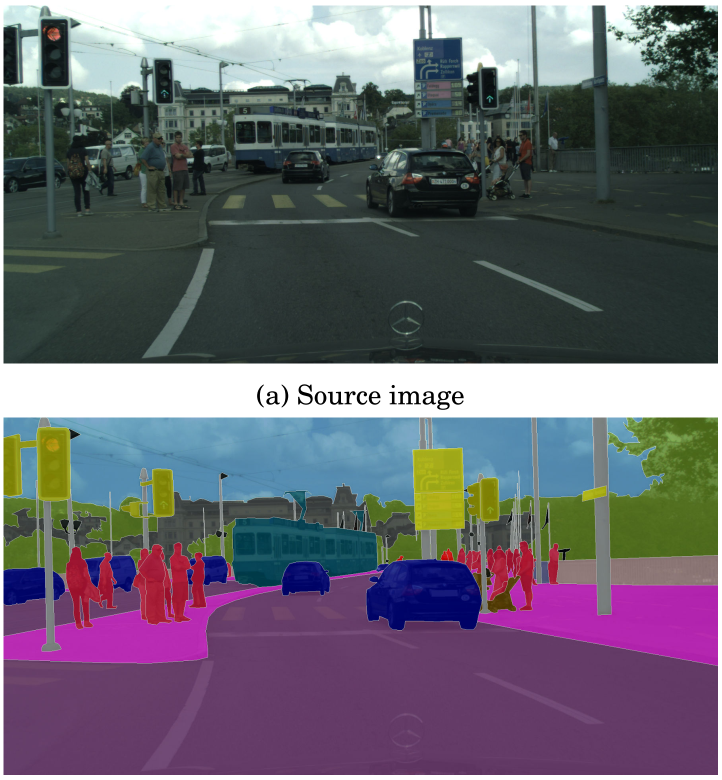
Inspired by the previous work, we study the problem of label-only membership inference attacks against semantic segmentation models. That is, given only the victim model's prediction labels, can the attacker still tell if a specific record was used in a semantic segmentation model? This research has much potential for real-world applications. Semantic segmentation tasks have already been adopted in many commercial or under-development products. Privacy leakage exists in image segmentation applications for autonomous driving [31] and robot navigation [211] because the attackers can utilize the segmentation results to figure out the users' location or other sensitive information. Applications for medical data segmentation [136] can leak the patients' diagnosis and health condition information. These applications are deployed with models with privacy risks, and the attacker can only obtain prediction labels from querying deployed applications. The label-only attacks make it possible to extract private information even from a seemingly private setting of revealing hard-value classification results. The discussed uniqueness of this work sets us apart from the existing work on membership inference attacks.

Although techniques of membership inference attacks have been developing during the past few years, a few challenges still remain: Firstly, to deploy attacks in real-world applications, the adversary knowledge needs to be as little as possible (e.g., label-only attacks). Otherwise, the attacks might not be meaningful in practice. Secondly, the extension from classification tasks to semantic segmentation tasks is not trivial. Although semantic segmentation can be considered a collection of pixel classification, the information contained in a single pixel is limited, resulting in an unreliable indicator of classification labels in the prediction output. Multiple procedures are needed to process these unreliable indicators of prediction labels efficiently. Also, the pixels are not equal in predicting the output, with some pixels carrying more information than others, which should also be considered in the data processing procedures. Further, the label-only attacks in the semantic segmentation tasks require more strategies to extract membership information from the data samples, which is a major difference from the label-only attacks in other tasks.

To tackle the above challenges, we design a new attack framework: We apply different data augmentations to the data samples to obtain more adversary knowledge. Then, we adopt several post-processing strategies (prediction-label concatenation and patch cropping) to the victim model's prediction output to apply the attacks against semantic segmentation models. Our contributions in this chapter are the following:

- We propose the first label-only membership inference attacks against semantic segmentation models.

- We design a framework for membership inference attacks by applying different data augmentations to the data samples and several post-processing strategies to the victim model's prediction output.



(a) Source image

(b) Visualization of annotated labels

Figure 3.1: An example of the semantic segmentation task.

- We discuss several defense mechanisms against membership inference attacks in semantic segmentation tasks.
- We achieve competitive experimental results of attack and defense methods compared to previous research.

3.2 Preliminaries and Related Work

3.2.1 Semantic Segmentation

Being a vital computer vision task towards complete scene understanding, semantic segmentation is a pixel-level labeling task for all image pixels, which labels all the objects, stuff, or background areas in the image to each category [121]. Figure 3.1 gives an example of what semantic segmentation looks like between a source image and its corresponding annotated labels using colorized visualization. A semantic segmentation label image is a gray-scale image where each pixel represents its class number in a real task.

With deep learning technologies widely adopted in computer vision, many semantic segmentation models have been developed based on deep learning. Several prominent deep learning based semantic segmentation models are fully convolutional networks [112], encoder-decoder based models [5], multi-scale and pyramid based models [215], dilated convolutional models [23], and attention based models [22].

One of the first deep learning models in semantic segmentation tasks applied a fully convolutional network (FCN) [112]. SegNet [5] was an encoder-decoder based model, which extracted feature maps in the encoder process and then up-sampled the lower-resolution feature maps to the original resolution. Pyramid scene parsing network (PSPNet) [215] adopted a multi-scale network to learn the global context representation and then processed the patterns from different scales with a pyramid pooling. Deeplabv3+ [23] used dilated convolutional layers to solve the decreasing resolution in the model and an atrous spatial pyramid pooling (ASPP) to extract feature maps. Although different deep learning technologies are applied in these models, the main goal of semantic segmentation is to extract either local or global features in the image.

This chapter focuses on membership inference attacks against semantic segmentation models. And we test our attacks on PSPNet [215], UperNet [180], DANet [48], and Deeplabv3+ [23].

3.2.2 Membership Inference Attacks

Here, we introduce related work on membership inference attacks **that is specifically relevant to this chapter**. Membership inference attacks aim to find whether a specific data sample has contributed to the victim model's training. The adversary does not have direct access to the training dataset or the trained model. However, based on the observations of the victim model's prediction output, the adversary can predict whether a specific data sample is in the victim model's training data or not.

Shokri *et al*. [145] pioneered the topic of membership inference attacks against image classification, leveraging multiple shadow models to generate data to train multiple attack models. Salem *et al*. [141] relaxed the assumptions of the attacks, showing that models and datasets of the victim and shadow models can be independent. They also demonstrated that one shadow model was already enough for the attacks. Yeom *et al*. [191] showed that the overfitting feature of models could lead to the models' vulnerability to membership inference attacks. Choquette-Choo *et al*. [28] and Li *et al*. [100] both proposed attacks with only access to the victim model's prediction labels.

Instead of using confident scores of the victim model's output, they applied various data augmentations to the original images and obtained corresponding prediction labels as membership information. More recent works studied the influence of data augmentations on membership inference attacks, highlighting the privacy risk in the models trained with augmented data [86, 193]. Other new works proposed novel attack methods [79] or conducted assessments on the performance of the attacks [108, 137].

While prior work has mostly studied attacks against classification models, several papers extend the scenarios to attacks against other deep learning models or learning settings. Very recent studies discovered that membership inference attacks were also possible in semantic segmentation [66].

Defense mechanisms against membership inference attacks usually fall into two categories. The first category usually tried to solve the overfitting issue of the victim model, as overfitting could lead to the exposure of individual samples [145, 191]. Dropout [141] or differential privacy [1] could be used to reduce the successful attack rate. The second category advocates perturbation to the victim model's confidence scores to break the attacks [84, 145]. Shokri *et al.* [145] proposed several possible strategies for changing the confidence scores. Jia *et al.* [84] presented MemGuard, a strategy of adding noises to the victim model's prediction to confuse the attack model. However, this kind of defense mechanism cannot be applied to label-only attacks since the confidence scores are not observable.

Our research focus is on label-only attacks against semantic segmentation models. The attacks can be deployed in real-world applications, and a deep understanding of such attacks is needed. This work provides a thorough and systematic study of this research area.

3.2.3 Architecture of the Attack Framework

There are three models involved in the task of membership inference attacks. The *target model* or *victim model* is the target of the attack. Member samples are in the model's training data, while non-member samples are not. The deep learning model usually behaves differently when meeting member and non-member samples. The model's prediction output follows a different pattern or data distribution because of the overfitting nature of the model [141, 145].

To launch a more successful attack, a *shadow model* is created to mimic the victim model's prediction output, such as the different patterns or data distribution between member and non-member samples. Then, leveraging the shadow model's prediction

Table 3.1: Notations

Notation	Description
V	Victim model, or called target model
S	Shadow model
A	Attack model
D	Dataset for the model
X	A set of images in the dataset
Y	A set of corresponding ground truth labels in the dataset
P	Prediction result of a model, e.g. $P = V(X)$

output, we can train an *attack model*, which is a classifier that differentiates member samples from non-member samples.

In this chapter, we follow the steps of [66] to study attacks in semantic segmentation models. Instead of using confidence scores of the model's predictions, we relax the membership inference attacks' assumptions to use the prediction labels only, which is inspired by [28, 100]. Table 3.1 lists the notations used in our framework.

3.3 Problem Formulation

3.3.1 Threat Model

As our goal is to show whether label-only membership inference attacks can match the performance of previous research in semantic segmentation models, we propose a threat model similar to prior work [28, 66, 145], which means the adversary only has black-box access to the model, i.e., the adversary does not have access to the model parameters and can only make queries to obtain model predictions or confidence scores. The details of the threat model are as follows.

3.3.1.1 Task Knowledge

Task knowledge refers to the type (in our case, semantic segmentation), the scenario (street view scenes), the class labels (cars, pedestrians, road, and other labels in street view scenes), the input image format (RGB images), etc. In this chapter, the task knowledge is assumed to be known to the adversary.

3.3.1.2 Model Knowledge

Model knowledge refers to any knowledge related to the victim model, including the model parameters, the size of the training dataset, the number of the training iterations and epochs, the setup of optimizers. The adversary does not have any model knowledge of the victim model.

The knowledge of the victim model's structure depends on the attack setting. This can either be known (dependent attacks) or unknown (independent attacks). Please refer to Section 3.6.1 for more information on our experimental setup.

Our study is on label-only attacks, so we do not need the confidence scores of the victim model's output. The adversary can only obtain the prediction labels of the victim model's output, which is more realistic in real-world applications because the confidence scores might not be supported by the associated API, or they might have been altered internally due to privacy protection reasons. Hence, the ground truth labels are essential for extracting the membership information.

3.3.1.3 Data Knowledge

The adversary is aware of the distribution of the training dataset of the victim model and can collect a new dataset based on the same distribution. The new dataset could be real or synthetic based on different experimental setups [145], and it should not overlap with the victim model's training dataset. We call this new dataset the shadow dataset. Besides, the adversary cannot access the victim model's training dataset directly. Please refer to Section 3.6.1 for more information on our experimental setup.

3.3.2 Design Goal

To tackle the limitations of current membership inference attacks, we design the framework of membership inference attacks in a label-only setting for semantic segmentation models. To this end, we aim to achieve the following design goals:

- The first goal is to design a pipeline framework to initiate the membership inference attacks in a label-only setting for semantic segmentation models. In Section 3.4, we describe our pipeline framework, including the training and testing of the attack model and also the details of our data representation procedures.

- The second goal is to evaluate whether our label-only attacks can achieve better attack accuracy than other attacks against semantic segmentation models. In Section 3.6, we evaluate our attack framework in various settings.

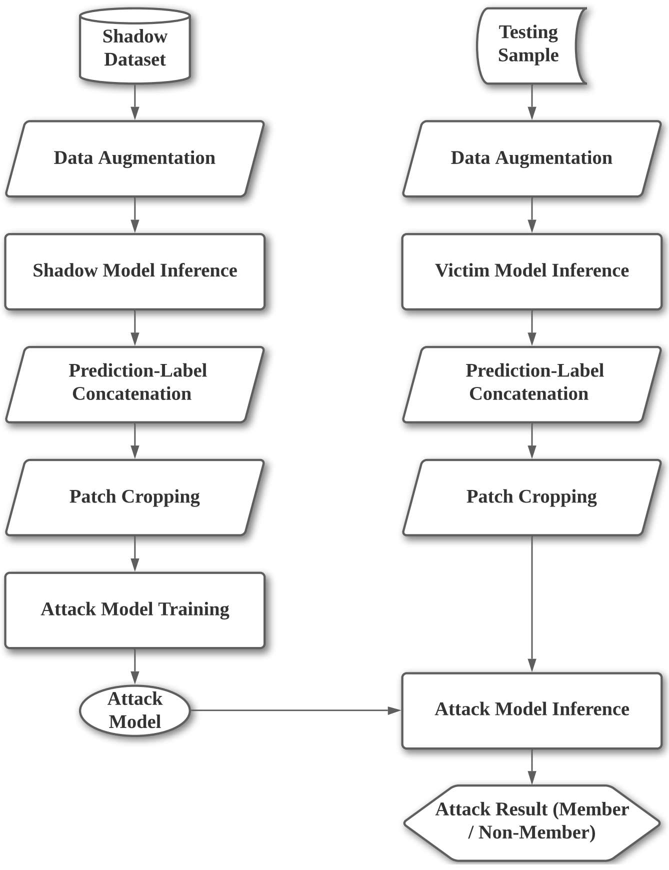


Figure 3.2: The label-only membership inference attack framework against semantic segmentation models.

- The third goal is to evaluate whether the current defense mechanisms can defend our attack settings. We introduce several defense mechanisms in Section 3.5 and test their effectiveness against our attack framework in Section 3.6.

3.4 Label-Only Attack Framework in Semantic Segmentation

In this section, the label-only membership inference attack against semantic segmentation models is introduced. Figure 3.2 shows the proposed attack framework, including the training and testing of the attack model. The following subsections will discuss the details of this framework.

3.4.1 Data Preparation

Before training the attack model, We need to prepare the datasets for the victim/shadow model for model-building. In more detail, we build a shadow model $\boldsymbol{S}$ similar to the victim model $\boldsymbol{V}$. The shadow model is expected to exhibit similar behavior in predicting outputs like the victim model when they infer member and non-member samples. Our attack model $\boldsymbol{A}$ is a classifier that differentiates member and non-member samples.

We prepare two datasets D^V and D^S for the victim model $\boldsymbol{V}$ and the shadow model $\boldsymbol{S}$, where $D^V = \{(X_i^V, Y_i^V)\}_i$, $D^S = \{(X_i^S, Y_i^S)\}_i$. Here, X_i represents a single image in the dataset, and Y_i denotes the corresponding ground truth labels, with one label for each pixel in the image. These two datasets can be dependent or independent [141], which will be discussed in detail in Section 3.6.1. We split each dataset into two sub-datasets for the victim/shadow model's training and testing. The shadow model's training data D_1^S and testing data D_0^S are considered as member/non-member samples for the attack model, where 1 and 0 denote the binary membership status. The same setting applies to the victim model's training data D_1^V and testing data D_0^V. In reality, sub-datasets D_1^S and D_0^S are prepared by the attacker, and the attacker's final goal is to determine whether the testing sample (x, y) belongs to D_1^V or D_0^V.

3.4.2 Training of the Attack Model

The training process of the attack model is discussed in this subsection. Algorithm 1 presents the pseudocode of the model training.

We prepare the dataset D^A of the attack model $\boldsymbol{A}$ based on the victim/shadow model's output. The training/testing data D_{train}^A and D_{test}^A of the attack model $\boldsymbol{A}$ are constructed by the shadow/victim model's output, respectively. D_{train}^A and D_{test}^A is formulated by the following data representation procedure, including one (data augmentations) before the victim/shadow model inference, and the other two (prediction-label concatenation and patch cropping) after the inference:

- **Data Augmentation.** Multiple data augmentation methods are applied in our algorithm, which are denoted as $aug(\cdot)$. The original images X^V and X^S are processed by different data augmentation methods, the outputs of which are denoted by $aug(X^V)$ or $aug(X^S)$. Further, they are processed by the victim/shadow model. We take the prediction label of the output, which is denoted as $P1^V = V(aug(X^V))$ or $P1^S = S(aug(X^S))$.

Algorithm 1 Training of the Attack Model

```
Input: $D^S = D_1^S \cup D_0^S = \{X^S, Y^S\}, \boldsymbol{S}, epoch\_num$
Output: $\boldsymbol{A}$
 1: $i = 0$
 2: while $i < len(D^S)$ do
 3:     $P1_i^S = S(aug(X_i^S))$
 4:     $P2_i^S = pl\_concat(P1_i^S, Y_i^S)$
 5:     $P3_i^S = crop(P2_i^S)$
 6:     $i = i + 1$
 7: end while
 8: Reshape $P3_i^S$ to $P3^S$ for each $P3_i^S$ has $k$ cropped patches
 9: $P3^S$ can be split into $P3_1^S$ (from $D_1^S$) and $P3_0^S$ (from $D_0^S$)
10: $m = 0, n = 0$
11: while $m < epoch\_num$ do
12:     while $n < len(P3^S)$ do
13:         if $P3_{(m,n)}^S \in P3_1^S$ then
14:             $A(P3_{(m,n)}^S)$ train as 1
15:         else
16:             $A(P3_{(m,n)}^S)$ train as 0
17:         end if
18:         $n = n + 1$
19:     end while
20:     $m = m + 1$
21: end while
22: return $\boldsymbol{A}$
```

- **Prediction-Label Concatenation.** We concatenate victim/shadow model predictions with labels to give a second data representation, which is denoted as $P2^V = pl\_concat(P1^V, Y^V)$ or $P2^S = pl\_concat(P1^S, Y^S)$.

- **Patch Cropping.** We crop several patches by some rules from the second data representation result as the input of the attack model denoted as $P3^V = crop(P2^V)$ or $P3^S = crop(P2^S)$.

We introduce these three steps of the data representation procedure in the following paragraphs. These three steps for the attack model's training and testing are the same. The only difference is the dataset and the model (victim or shadow) used in these steps. In this subsection, we construct D_{train}^A with D^S and $\boldsymbol{S}$.

3.4.2.1 Data Augmentation

As mentioned in [28, 146], data augmentation is a very powerful tool for building deep learning models. The goal of increasing the performance of deep learning models is to minimize the distance between the training and testing data. The augmented training data will better represent the training dataset to achieve this goal.

However, this data augmentation process makes deep learning models vulnerable to membership inference attacks, especially label-only attacks. With only the prediction labels available in the label-only attacks, the information is not enough to initiate the attack. The intuition here is to generate more information to extract membership status. If a data sample is used to train the victim/shadow model, the augmented data samples are likely to also participate in training the victim/shadow model. In this way, the attacker leverages augmented input data to obtain a better membership signal.

We apply several different data augmentation methods, two of which (translation and rotation) are similar to data augmentations in [28], while the others are new in this chapter.

- Translation. Given a translation scale s, we translate the image by $\pm s$ pixel horizontally and vertically. We output five images in total, including the original one.
- Rotation. Given a rotation scale s, we rotate the image by $\pm s^\circ$. We output three images in total, including the original one.
- Brightness, contrast, hue, or saturation. These four photometric distortions share the same strategy: we randomly apply one of them to the image and output five images, including the original one.
- Random. We randomly select the above six data augmentations and apply them to the image. We output five images in total, including the original one.

With the augmented image input, we receive the victim/shadow model's inference result, the prediction labels. As this is the training of the attack model, we receive the shadow model's prediction labels denoted as $P1^S$. In a normal membership inference attack, the inference result is the confidence score of the prediction. This is formulated as a pixel-wise matrix of size (c,h,w), where c is the number of class labels of the dataset, h and w are the height and width of the image. With a softmax layer being the last layer of the model, the matrix values range between 0 and 1, each of which denotes the probability of a class for a single pixel. In our label-only attack, the prediction label $P1^S$ is a matrix of size (n, h, w), where n means the number of data augmentations of one image. Each value in the matrix means the class ID in a single pixel.

3.4.2.2 Prediction-Label Concatenation

In this step, we leverage the prediction labels $P1^S$ and the ground truth labels Y^S to provide a better data representation for the next step. In order to differentiate member samples from non-member samples, the attack model needs input data to contain information from both the prediction labels and the ground truth labels. The ground truth labels Y^S is a matrix of size $(1, h, w)$. Here, we adopt three different strategies to form the output of the prediction-label concatenation $P2^S$.

- **Simple concatenation.** We simply perform a matrix concatenation between $P1^S$ and Y^S, leading to a matrix of size $(n+1, h, w)$ as the output $P2^S$:

  $$P2^S = matrix\_concat(P1^S, Y^S), \tag{3.1}$$

  where $maxtrix\_concat(\cdot)$ means the matrix concatenation process.

- **Mixup and one-hot concatenation.** Unlike the confidence scores of the prediction output, the prediction label of each augmented output has only one channel instead of c channels. To mimic the matrix structure of the confidence score output, we perform the one-hot encoding to each channel of the prediction label matrix and then perform Mixup [206]:

  $$M_{all} = \sum_{i=1}^{n} \lambda_i M_i, \ \sum_{i=1}^{n} \lambda_i = 1, \tag{3.2}$$

  where M_{all} is the matrix after Mixup, M_i is the ith the prediction label matrix with the size (c, h, w), and λ_i is the corresponding weight. In this way, we receive the matrix M_{all} with the size (c, h, w). The Mixup process reduces matrix dimensions from $n \times c$ layers to c layers and still contains all the information on each prediction label.

  We then concatenate the matrix M_{all} to the one-hot encoded ground truth label (a matrix of size (c, h, w)), leading to a matrix of size $(2c, h, w)$ as the output $P2^S$:

$$P2^S = matrix\_concat(M_{all}, one\_hot(Y^S)), \tag{3.3}$$

where $one\_hot(\cdot)$ means the one-hot encoding process.

- **Mixup and structured loss map (SLM).** The Mixup process is the same as Mixup and concatenation. Then we calculate the structured loss map using the Mixup output M_{all} and the one-hot encoded ground truth label [66]:

$$P2^S = -\sum_{i=1}^{c} one\_hot(Y^S)_i log(M_{(all,i)}). \tag{3.4}$$

  The structured loss map calculates cross-entropy loss values across all locations, resulting in a matrix of size $(1, h, w)$ as the output $P2^S$. This process significantly reduces the dimension of the output $P2^S$ from $2c$ to 1.

Based on different strategies, we obtain different data representations $P2^S$ with various sizes of the matrix.

3.4.2.3 Patch Cropping

Patch cropping is the final step of our data representation procedure, which is a procedure to crop one image into several patches based on a specific rule. The membership indicator information in the data representation is still weak and insufficient to support an effective attack [66]. Applying patch cropping methods to the data representation $P2^S$ can aggregate more information over patches to launch a stronger attack. For each $P2^S_i$ in $P2^S$, we make the cropping procedure to form k patches, denoted as PA^S. PA^S_k has 4 variables $(x\_idx, y\_idx, pa\_h, pa\_w)$, where x_idx and y_idx are the coordinates of the upper-left point of the patch, and pa_h and pa_w are the height and width of the patch. In this chapter, we crop patches of the prediction-label concatenation output $P2^S$ by the following rules.

- **Random.** We crop patches across the matrix $P2^S$ randomly.
- **Sliding windows.** We crop patches with a fixed step size from the upper-left to the bottom-right of the matrix. This method can guarantee that all the information in the matrix is included in the cropped patches.

Algorithm 2 Testing of the Attack Model

```
Input: Testing sample $(x, y)$ from $D^V, \boldsymbol{V}, \boldsymbol{A}$, membership threshold $\kappa$
Output: $Result$
 1: $p1 = S(aug(x))$
 2: $p2 = pl\_concat(p1, y)$
 3: $p3 = crop(p2)$
 4: $p3$ has $k$ cropped patches
 5: $m = 0, Result = 0$
 6: while $m < k$ do
 7:     $Result += A(p3_m)$
 8:     $m = m + 1$
 9: end while
10: $Result = \frac{Result}{k}$
11: if $Result >= \kappa$ then
12:     Testing sample $(x, y)$ is a member sample
13: else
14:     Testing sample $(x, y)$ is a non-member sample
15: end if
16: return $Result$
```

- **Random rejection to preserve diversity in labels.** We reject some randomly cropped patches if the patch has the most pixels of one label. We set a rejection degree η for determining when to reject the cropped patches. For example, road areas are prevalent in street scenes and may take a large portion of a single image. Here, η is set to 80%, and we reject this patch type if road labels are observed for more than 80% of the pixels.

After the patches are cropped, we obtain the final data representation $P3^S$, which forms the training dataset of the attack model D^A_{train}. $P3^S$ can be described as:

$$P3^S = \{PA^S_k\}_k = crop(P2^S). \tag{3.5}$$

Then, we construct a classification model $\boldsymbol{A}$ to differentiate member and non-member samples in the original shadow dataset $\boldsymbol{S}$.

3.4.3 Testing of the Attack Model

In this subsection, we apply the testing sample (x, y) from D^V and $\boldsymbol{V}$ to construct $P1^V$, $P2^V$, and $P3^V$, which finally forms the testing dataset of the attack model D^A_{test}. Then, we obtain the inference result of $\boldsymbol{A}$ to determine whether (x, y) is a member sample or a non-member sample. Algorithm 2 presents the pseudo-code of the attack model testing. The membership threshold κ required in the algorithm is usually set to 0.5.

There is another difference between the training and testing of the attack model. The inference result of the attack model is the classification result of the patches, not the whole image. The result of the patches in the same image should be calculated together, and the result of a single image should be as follows [66]:

$$Result = \frac{1}{N}\sum_{i=1}^{N} A(X_i^A, Y_i), \tag{3.6}$$

where X^A means the input of the attack model and (X_i^A, Y_i) means the i-th patch in the same image with the corresponding ground truth label.

3.5 Defense Mechanisms Against Label-Only Attack Framework In Semantic Segmentation

This section presents possible defense mechanisms for label-only membership inference attacks to protect private information in semantic segmentation models. Previous research proposed several defense mechanisms in the training or testing phase of the victim model to mitigate membership leakage.

Defense mechanisms in model testing usually try to add perturbations or make changes to the confidence scores of model predictions. In this way, the defender does not need to retrain the model to protect privacy. There are several developed strategies in the existing work, such as restricting the prediction scores to top k classes [145], coarsening precision of the prediction scores [145], MemGuard (a mechanism to add noises to prediction scores) [84], adding Gaussian noises to prediction scores [66], and manipulating prediction scores to decrease confidence [27, 188]. However, these defense mechanisms are not suitable for our attack algorithm since only the victim model's prediction labels are available in our scheme. The mentioned mechanisms were not meant for the prediction labels.

Therefore, defense mechanisms in model training could be a possible strategy to protect privacy from label-only membership inference attacks. We discuss Dropout [69] and DPSGD [1] in the following subsections.

3.5.1 Dropout

To reduce the overfitting of deep learning models, Hinton *et al.* [69] proposed dropout, a regularization method. The idea is to randomly drop a few units and their connections from deep neural networks during training. The dropout ratio is a hyper-parameter for setting the probability of retaining a unit in the network.

Salem *et al.* [141] demonstrated that dropout could also be used to defend against membership inference attacks because one of the requirements of successful attacks is the overfitting feature of the victim model.

3.5.2 DPSGD

Differential privacy [40] provides a standard for privacy guarantees of neighboring datasets. Differential privacy defines privacy from a mathematical perspective. The definition of differential privacy is as follows:

Definition 1: *((ϵ, δ) - Differential Privacy). For any two neighboring datasets D and D' that differ in only a single entry, a randomized mechanism $\mathscr{A}$ provides (ϵ, δ) - Differential Privacy, if for $\forall S \subseteq Range(\mathscr{A})$,*

$$Pr[\mathscr{A}(D) \in S] \leq e^{\epsilon} Pr[\mathscr{A}(D') \in S] + \delta, \tag{3.7}$$

where ϵ measures the privacy loss between the neighboring datasets and δ is related to the size of the dataset.

Adopting differential privacy in deep learning model training can prevent the model from memorizing any individual data record. Abadi *et al.* [1] proposed a differentially private stochastic gradient descent (DPSGD) to provide strict privacy bound with similar utility compared to models without defense mechanisms. Compared to a standard SGD optimizer, DPSGD introduces the following changes to guarantee privacy: adding Gaussian noises to the gradient and clipping the gradient based on the gradient norm. Therefore, two parameters (noise scale σ and gradient norm bound C) should be calculated to ensure a better privacy guarantee. In [1], they also proposed a moments accountant (MA) method to calculate these parameters more tightly.

3.6 Performance Evaluation

In this section, we present our membership inference attack's experimental setup and then demonstrate and analyze our experimental results.

Table 3.2: Dataset and Model Settings

Setting	Dataset	Model
Dependent	Cityscapes, BDD100K, Mapillary	PSPNet
Independent	Cityscapes, BDD100K, Mapillary	PSPNet, DANet, UperNet, Deeplabv3+

3.6.1 Experimental Setup

3.6.1.1 Settings

As [141] has previously reported, the dataset and model settings in the victim and shadow model can be dependent or independent. The *dependent attack* means that the attacker has a dataset from the same data distribution as the victim model's training data. The shadow model has the same architecture as the victim model. The *independent attack* is more realistic, meaning sometimes the attacker has no knowledge of the victim model and its dataset. The attacker only knows the functionality of the victim model, which is semantic segmentation in our case. The attacker also knows that the samples in the victim model's training data are street scene images, so the shadow model is trained by some other similar datasets. The shadow model is not for mimicking the victim model's behavior but for capturing the dataset's membership status.

Table 3.2 shows the dataset and model settings in each experiment. Setting Dependent is used in the dataset and model-dependent attacks while Setting Independent is used in the dataset and model-independent attacks. The dataset is usually split in a balanced setting (4 even subsets as victim member, victim non-member, shadow member, and shadow non-member) unless stated otherwise.

3.6.1.2 Datasets

We perform experiments using three well-known street scene semantic segmentation datasets:

- Cityscapes [31], a diverse set of images of street scenes from 50 different cities. As seen in Table 3.2, we split the dataset into different numbers of subsets based on different settings.

- BDD100K [194], a large-scale image dataset captured from driving videos by Berkeley AI Research.

- Mapillary Vistas [126], a fine annotated segmentation dataset of images of street scenes with various weather, season, camera, and viewpoint conditions. This dataset has the largest number of images in our experiments.

These three datasets have different image sizes. In order to have a unified model input, we resize the images of the studied datasets to the same size. We also divide the images into member samples and non-member samples.

These three datasets also have different numbers of class labels. We transform the class labels in BDD100K and Mapillary Vistas to be the same as those in Cityscapes.

3.6.1.3 Models

We evaluate our attacks using the following semantic segmentation models (victim or shadow): PSPNet [215], UperNet [180], DANet [48], and Deeplabv3+ [23]. We select Resnet-50 [62] as our attack model.

3.6.1.4 Evaluation Metrics.

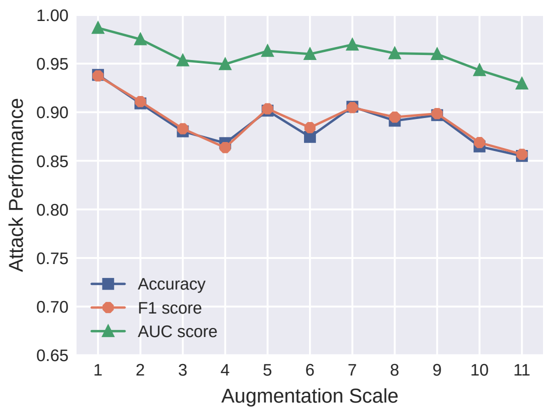
As there are only two classes (member and non-member samples) in our attack model, we denote member samples as positive samples and non-member samples as negative samples. To evaluate the effectiveness of our attack model, we count true positive (TP), false positive (FP), true negative (TN), and false negative (FN), as well as the precision and recall of our result.

To compare different attacks quantitatively, we used the above experimental results to evaluate our attack model in terms of three metrics:

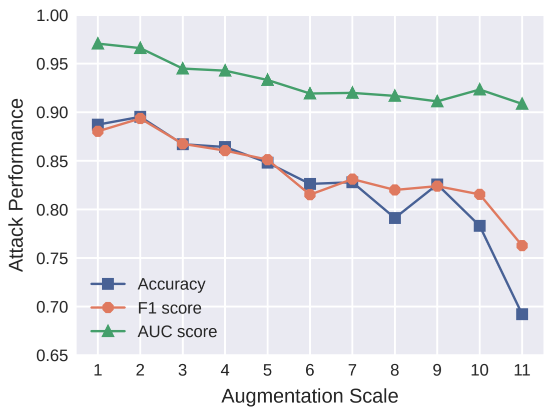
- Attack accuracy: Attack accuracy means the rate of the accurate class of the attack model;
- F1 score: F1 score measures the overall performance of precision and recall;
- AUC score: AUC score quantifies the area size under the ROC curve, which measures the trade-off between the true positive rate and the false positive rate [43].

In summary, the higher the attack accuracy, F1 score and AUC score we get, the more effective the attack model is. We measure our attack methods in different settings using the above three metrics.

The experimental setup for defense mechanisms is similar to that for attack methods. As defense mechanisms are adopted in the victim model training, we will compare the



(a) Translation



(b) Rotation

Figure 3.3: Evaluation of the attack performance with translation and rotation augmentations under different scales.

attack model performance between models with and without defense mechanisms. The privacy metrics are the same, i.e., attack accuracy, F1 score, and AUC score. We also evaluate the performance between utility and privacy. The utility metrics in semantic segmentation models is mean intersection over union (mIoU) [51]. It is a standard metric for semantic segmentation, which computes a ratio between the intersection and the union of two sets (the ground truth and the prediction) on a per-class basis. It is then averaged over the results, which can be calculated as follows:

$$mIoU = \frac{1}{k}\sum_{i=1}^{k}\frac{ground\_truth \bigcap prediction}{ground\_truth \bigcup prediction} \tag{3.8}$$

3.6.2 Attack Experiment 1: Different Data Augmentation Scales

In our first experiment, we evaluate the performance of our membership inference attack under different scales of translation and rotation augmentations (from scale 1 to 11). As this is a dependent attack, we use Cityscapes as the victim and shadow dataset. We randomly split the images in Cityscapes into four subsets as member/non-member data of the victim/shadow model. PSPNet is used as the victim/shadow model. We apply translation and rotation augmentations to the input images. We also adopted Resnet-50 as the attack model.

The attack performance is shown in Figure 3.3. For both translation and rotation augmentations, we can see that the performance decreases from around 0.9 to below 0.9 in terms of accuracy, F1, and AUC scores. The overall performance of translation augmentation is better than that of rotation augmentation, and the best performance is with a translation and rotation scale of 1.

In [28], they evaluated the performance of these two scales in image classification tasks, proving that too small or too large augmentations may harm the attack performance. We have obtained a similar result in semantic segmentation tasks: if the original victim/shadow model input images are largely augmented, the data augmentations cannot help to enhance the data sample's membership status. As a result, large data augmentations cause poor attack performance. We also discover that translation augmentation has a better attack performance than rotation augmentation on most scales. In later experiments, we adopt translation and rotation augmentations with scale 1.

3.6.3 Attack Experiment 2: Different Prediction-Label Concatenation Strategies

In our second experiment, the attack performance of three different prediction-label concatenation strategies is evaluated. In this and later experiments, the attack performance is compared among various data augmentations, which include translation (tran), rotation (rota), brightness (brig), contrast (cont), hue, saturation (satu), and random (rand). We use the same dependent attack settings, with the "sliding-windows" patch-cropping method. We evaluate the performance of simple concatenation (Simple), Mixup and one-hot concatenation (1Hot), Mixup and SLM (SLM).

Figure 3.4 shows the evaluation of the attack performance. We have discovered that the Simple strategy performed the worst, with attack accuracy, F1 and AUC scores of around 0.5. This means the attack will not work when simply concatenating the prediction labels with the ground truth labels. This is expected because the output data in this strategy has the values of class IDs, which does not give a reasonable representation of the data.

1Hot and SLM strategies achieve over 0.8 in terms of attack accuracy scores, F1 scores and AUC scores, and the 1Hot strategy has the best performance. This is different from the conclusion in [66] that the SLM strategy performs the best. This is because of the different experimental settings between both works. With limited label-only knowledge in our attack, our prediction map in the SLM strategy is different, which leads to different experimental results. In our attack framework, compared to 1Hot strategy, SLM strategy reduces the prediction matrix in the data representation matrix from $2c$ dimensions to 1, which leads to inevitable information loss. This eventually causes a lower attack

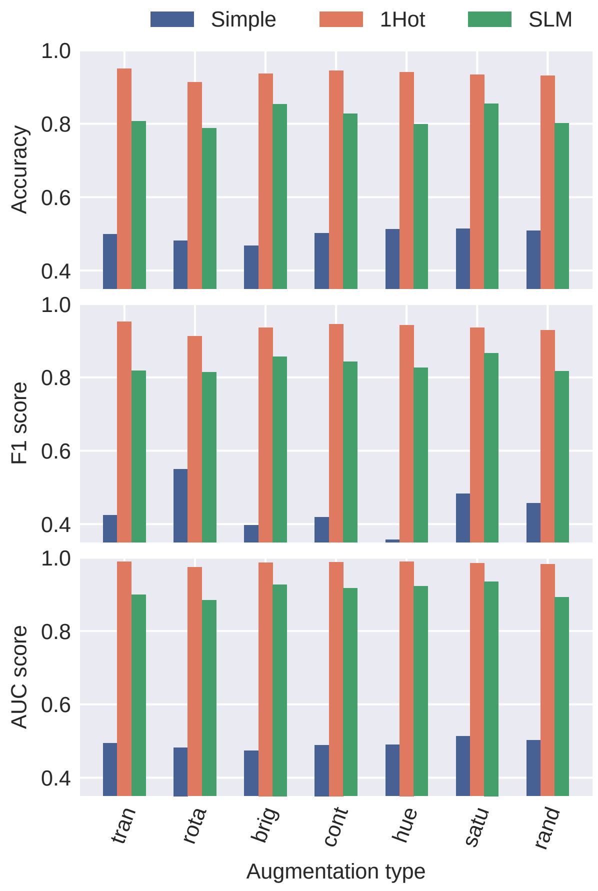


Figure 3.4: Evaluation of the attack performance under different prediction-label concatenation strategies.

accuracy.

It should be noted that the attack performance of different data augmentation types is very close. We can conclude that all types of data augmentation presented in this chapter can improve attack performance. In later experiments, we apply the 1Hot strategy as the first choice when concatenating prediction labels with ground truth labels in the data representation step.

3.6.4 Attack Experiment 3: Different Patch Cropping Methods

In our third experiment, we evaluate the attack performance with different patch-cropping methods. We compare the attack performance in all seven data augmentations with the SLM concatenation strategy. First, we set the patch number to 5 and evaluate sliding-windows (Slide), random-cropping (Random), and random-with-rejection (Reject) methods. The result in Figure 3.5 shows all three patch cropping methods can achieve

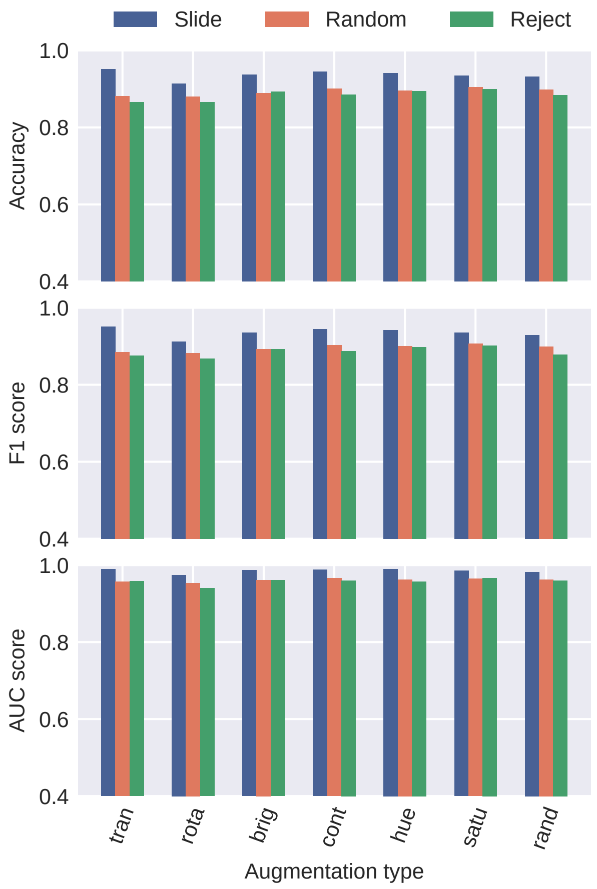
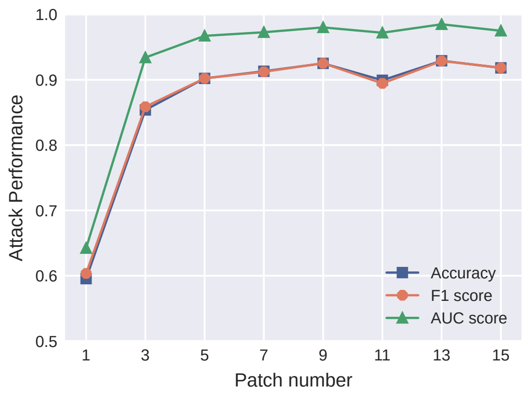


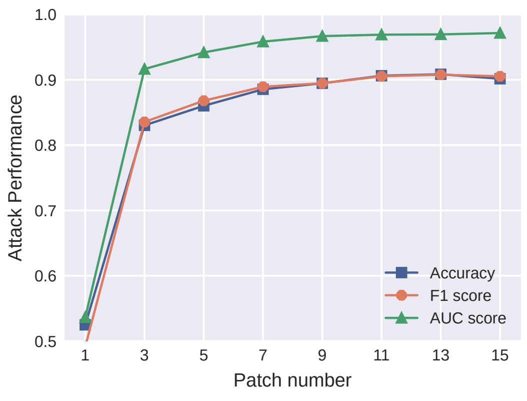
Figure 3.5: Performance under different patch cropping methods.

scores over 0.8 in terms of attack accuracy, F1, and AUC scores. The Slide method has the best results (over 0.9), but all three approaches have comparable outstanding scores overall.

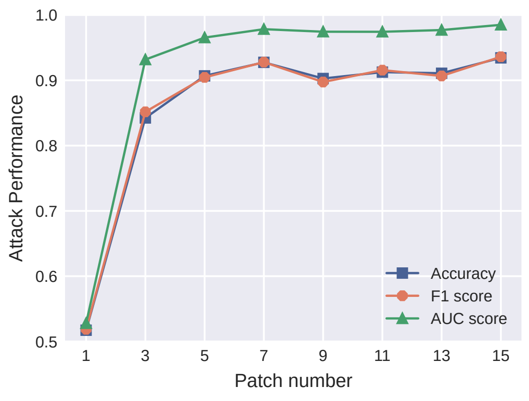
Next, we evaluate the attack performance in different patch numbers (from 1 to 15) with "Random" patch cropping method, as illustrated in Figure 3.6. The figures indicate that more patch numbers lead to better attack performance, with a peak attack accuracy of around 0.9. However, after the patch number increases to 5 or 7, the attack performance does not change much, which even begins to drop slightly in patch numbers 9 and 11 in translation, hue or saturation. The result indicates that cropped patches with 5 or 7 are enough to represent membership information in one image, which is similar to the conclusion of previous research.



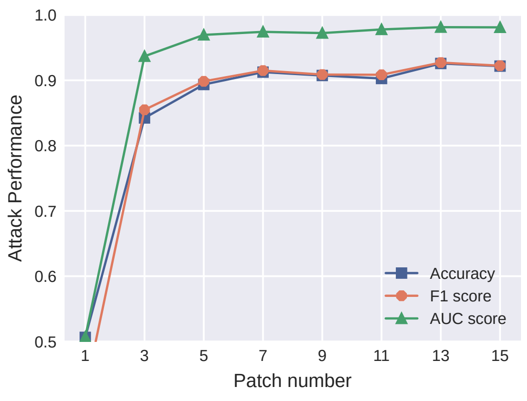
(a) Translation



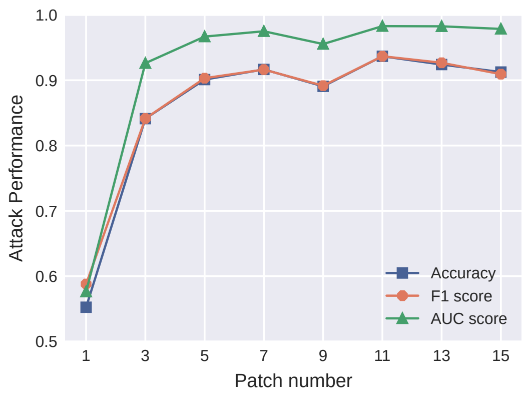
(b) Rotation



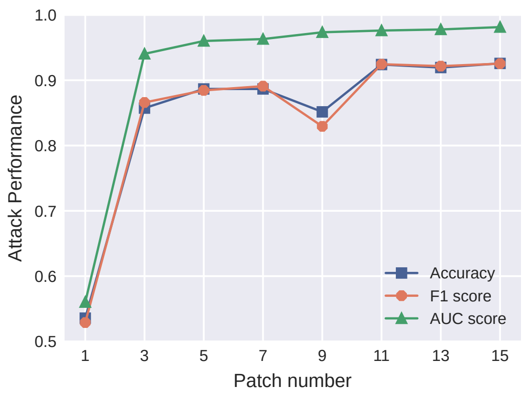
(c) Brightness



(d) Contrast



(e) Hue



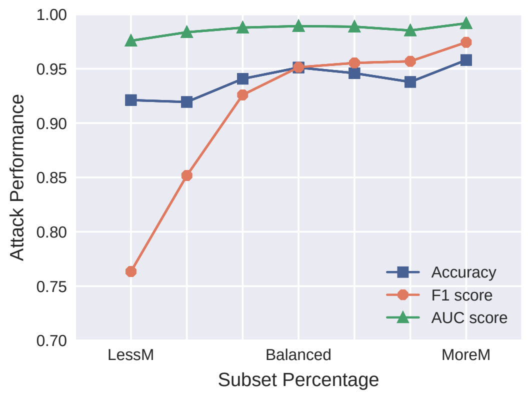
(f) Saturation

Figure 3.6: Evaluation of the attack performance under different patch numbers.

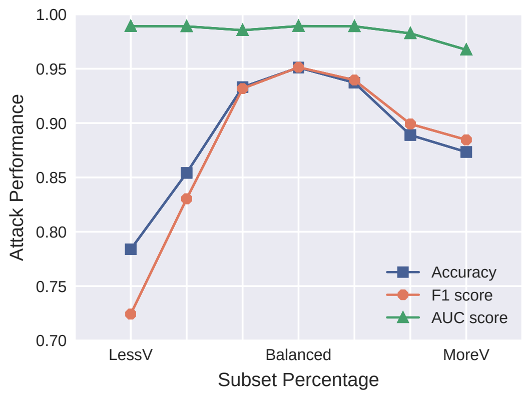
Table 3.3: Dataset size setting for Attack Experiment 4

Dataset Split Setting 1					Dataset Split Setting 2				
	Victim		Shadow			Victim		Shadow	
Type	M	NM	M	NM	Type	M	NM	M	NM
LessM	10%	40%	10%	40%	LessV	10%	10%	40%	40%
	15%	35%	15%	35%		15%	15%	35%	35%
	20%	30%	20%	30%		20%	20%	30%	30%
Balanced	25%	25%	25%	25%	Balanced	25%	25%	25%	25%
MoreM	30%	20%	30%	20%	MoreV	30%	30%	20%	20%
	35%	15%	35%	15%		35%	35%	15%	15%
	40%	10%	40%	10%		40%	40%	10%	10%

[1] M: Percentage of member samples; NM: Percentage of non-member samples.



(a) Member / Nonmember



(b) Victim / Shadow

Figure 3.7: Evaluation of the attack performance in different dataset sizes.

3.6.5 Attack Experiment 4: Different Datasets for Dependent Attacks

In this experiment, we evaluate the performance of the proposed framework under various datasets and their sizes for dependent attacks. The following tests utilize PSPNet as the victim/shadow model, and Cityscapes as the dataset. Translation augmentation, 1Hot, and Slide are chosen as the data representation procedures to collect the performance results.

First, we evenly divide the dataset into two subsets, the victim and shadow datasets, and then we divide the member and non-member subsets unevenly. Some subset percentage cases have been tested:

- Several cases with fewer member samples are denoted as LessM;
- A case with balanced member samples and non-member samples is denoted as

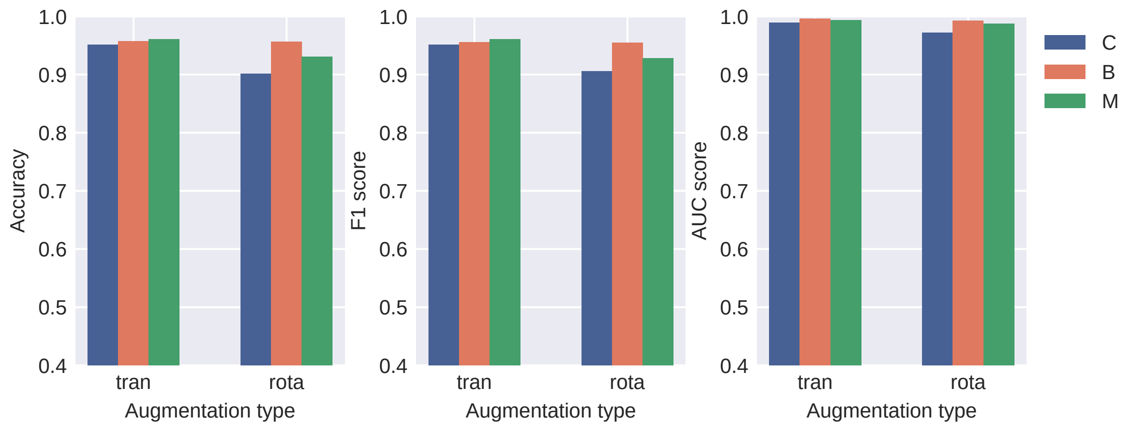


Figure 3.8: Evaluation of the attack performance in different datasets. C: Cityscapes; B: BDD100K; M: Mapillary

Balanced;

- Several cases with more member samples are denoted as MoreM.

Please refer to the dataset split setting 1 in Table 3.3 for more details. Figure 3.7a demonstrates the attack performance, indicating that most of the tests can achieve high accuracy and AUC scores over 0.9, and F1 scores increase from around 0.75 to around 0.95. This means that the membership percentage in the dataset has little impact on attack accuracy. As the F1 score is calculated using precision and recall, it is reasonable to observe a higher attack performance with more positive data (member samples).

Second, we evenly divide the dataset into two subsets, the member and non-member datasets, and then we split the victim and shadow subsets unevenly. Some subset percentage cases have been tested:

- Several cases with fewer victim subsets denoted as LessV;
- A case with balanced victim and shadow subsets denoted as Balanced;
- Several cases with more victim subsets denoted as MoreV.

Please refer to the dataset split setting 2 in Table 3.3 for more details. Figure 3.7b displays the attack performance, showing that the balanced setting yields the best attack performance with high accuracy scores around 0.95. The accuracy scores drop below 0.9 when the attacks have more victim or shadow subsets.

Next, we compare the dependent attack performance using various datasets. The experiments are conducted using Cityscapes, BDD100K, and Mapillary, and each of these datasets is divided evenly into four subsets as victim member, victim non-member, shadow member, and shadow non-member subsets. Other experiment settings are 1Hot and Slide strategies with translation and rotation data augmentations, and PSPNet is used as victim and shadow models. Figure 3.8 shows that the attack performance of these datasets all have accuracy, F1 and AUC scores of around 0.9, and the attacks with BDD100K and Mapillary can achieve higher scores. Although these tests are under various subset settings, they exhibit that the larger dataset the network is trained upon, the better generalization ability the attack model can get. The attacks with Cityscapes have already achieved relatively high accuracy, but the attacks with BDD100K and Mapillary can further increase the performance by several percentages.

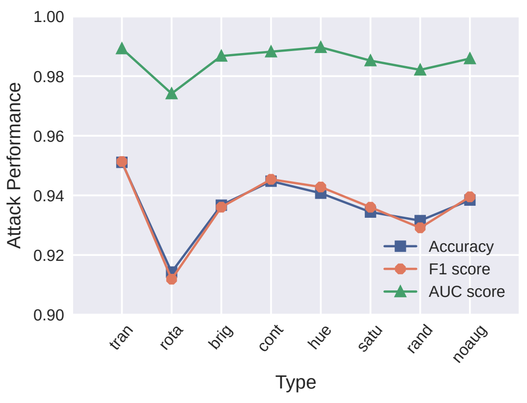
3.6.6 Attack Experiment 5: Dependent Attacks With or Without Data Representation Procedures

In our fifth experiment, we provide an evaluation of the dependent attacks with or without any one of the three data representation procedures. The experiment uses PSPNet as the victim/shadow model and Cityscapes as the dataset (divided evenly). In Figure 3.9, we show the evaluation results of the attacks with or without data augmentations, prediction-label concatenation, and patch-cropping strategies. The default strategies of the three data representation procedures are translation, 1Hot and Slide, respectively, when the procedures are not turned off.

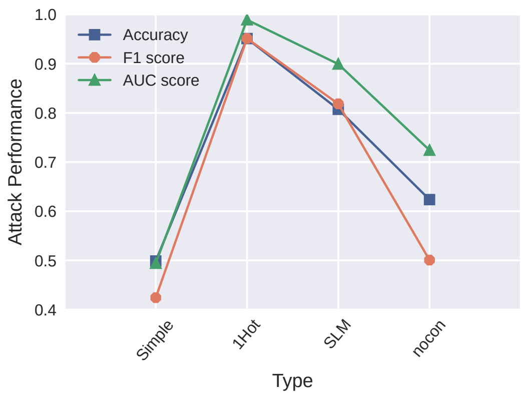
In Figure 3.9a, we evaluate the attacks with or without data augmentations. 1Hot prediction-label concatenation and Slide patch cropping are chosen as the default strategies. The attack with translation augmentations exhibits the best performance. Surprisingly, the attack without augmentations (noaug) does not have the worst performance. This indicates that some data augmentations are not very effective in the attack process. We would like to analyze this phenomenon in our future research.

In Figure 3.9b, we test the attacks with or without prediction-label concatenation. Translation augmentation and Slide patch cropping are chosen as default. The attack with 1Hot strategy achieves the highest accuracy, whereas the attack without concatenation (nocon) only gets accuracy scores of around 0.6. The results show that the attacks with 1Hot and SLM strategies significantly improve the attack performance.

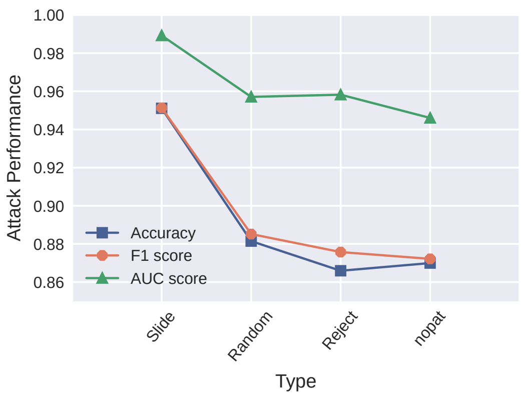
In Figure 3.9c, the attacks with or without patch cropping are tested. Translation and 1Hot are the default strategies. We discover that the attack with Slide strategy has the best performance, while the attack without patch cropping (nopa) demonstrates the worst



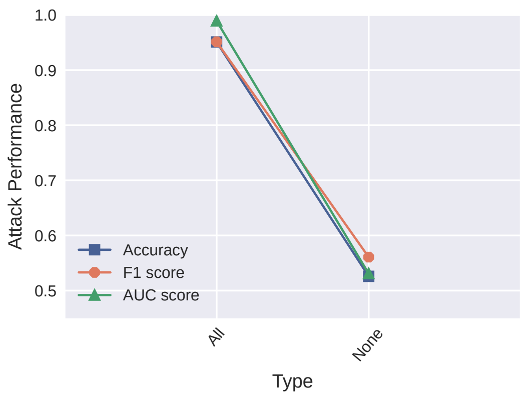
(a) With/without data augmentations



(b) With/without concatenation



(c) With/without patch cropping



(d) With/without all strategies

Figure 3.9: Evaluation of the attack performance with/without data representation procedures.

performance. The Random and Reject strategies can increase the attack performance to several percentages, but the Slide strategy is extremely effective in label-only attacks.

We also provide the results of the attacks with or without all three data representation procedures (Figure 3.9d). The attack with all three strategies (translation, 1Hot, and Slide) increases the performance from around 0.55 to 0.95, indicating that all these data representation procedures can contribute to the final result and enable the attack framework to successfully launch the label-only membership inference attacks.

3.6.7 Attack Experiment 6: Different Attack Settings

In this experiment, we evaluate our framework with independent attacks. Figure 3.10 provides an evaluation of the attacks with various shadow models, including PSPNet, DANet, and UperNet. We still use PSPNet as the victim model, Cityscapes as the dataset, and ResNet-50 as the attack model. We compare the attack performance with 1Hot and

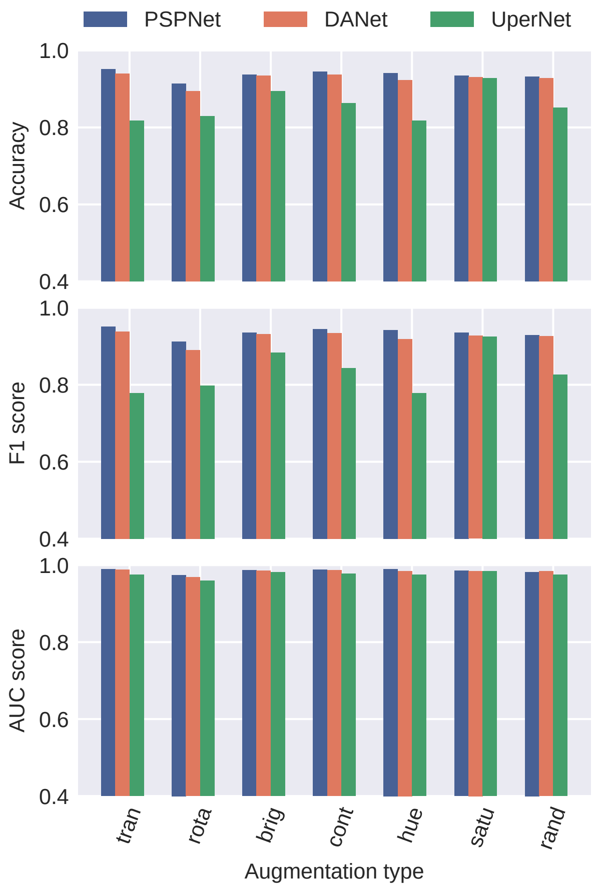


Figure 3.10: Evaluation of the attack performance in different shadow models.

Slide strategies with various data augmentations. The attacks using PSPNet as the shadow model have the best performance, all of which values are above 0.9 in terms of attack accuracy, F1 and AUC scores. The reason is that the more consistency between the shadow model and the attack model, the easier it is to mimic the attack model, resulting in a better performance.

Next, we provide an evaluation of our framework in independent attacks using various datasets as well as with different dataset sizes. Figure 3.11 shows statistics of the attack performance in three different datasets as the shadow datasets: Cityscapes, BDD100K, and Mapillary. The size of the shadow datasets varies from 1,000 to 20,000. The test of Cityscapes is a dependent attack, and the others are independent attacks. The other experiment settings include Cityscapes as the victim model, PSPNet as the victim model, and Deeplabv3+ as the shadow model. We choose translation, 1Hot and Slide for data representation procedures. Generally, we do not compare performance among different datasets. Instead, we focus on the investigation of the effect of dataset

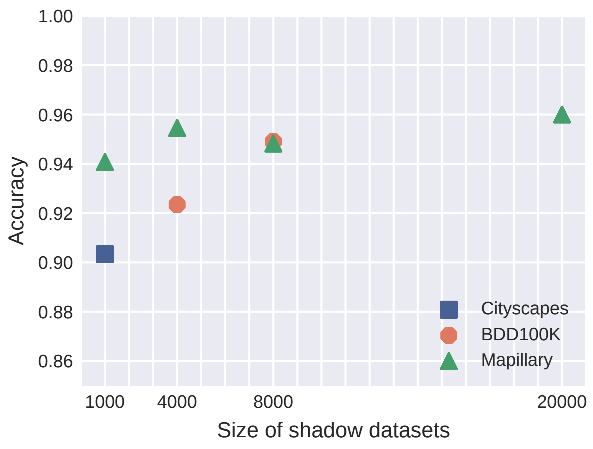


Figure 3.11: Evaluation of the attack performance in independent attacks.

Table 3.4: Attack Performance Comparison

Methods	Dependent PSP - PSP F1	Dependent PSP - PSP AUC	Independent Deep - PSP F1	Independent Deep - PSP AUC
ML-leaks (learning-based)	0.772	0.672	0.924	0.635
ML-leaks (learning-free)	0.774	0.620	0.923	0.634
Segmentations-leak (SLM, Random)	0.848	0.846	0.957	0.908
Segmentations-leak (SLM, Reject)	0.867	0.871	0.959	0.911
Ours (Tran, SLM, Random)	0.869	0.887	0.959	0.907
Ours (Tran, 1Hot, Slide)	**0.927**	**0.979**	**0.977**	**0.976**

[1] Attack performance with previous research of dependent and independent attacks. The best is marked in bold.

sizes on the independent attacks.

The results show the relationship between the dataset size and the attack accuracy. As the dataset size increases, the attack accuracy grows. The attacks using Mapillary outperform the others in most cases. When the whole Mapillary dataset is used as the shadow dataset (20,000 samples), the best attack performance can be achieved with an attack accuracy of over 0.95.

3.6.8 Comparison with Previous Research

We evaluate our experimental results (F1 scores and AUC scores) with ML-leaks [141] and Segmentations-leak [66] in Table 3.4. In dependent attacks, the best attacking scheme is our translation method, with an F1 score of 0.869 and an AUC score of 0.887. In independent attacks, we illustrate our results in Mapillary Vistas. Our translation method and Segmentations-leak method have the equally best F1 score around 0.959. Our methods in the other settings also achieve outstanding performance.

This comparison with previous research means that our membership inference attack with limited knowledge (label-only) can achieve similar performance compared with the state-of-the-art results [66]. The major difference between [66] and our work is that a stronger threat model (the attacker has access to the model's prediction scores) is needed in [66], while the assumption in our framework is relaxed to obtaining only the prediction labels. In particular, with only prediction labels (hard values) provided to the adversary, the information of classification likelihood/confidence is obscured, making it challenging to differentiate the non-member samples and the member samples used in training. In other words, it is common for a member sample and a non-member one to have exactly the same one-hot encoded vector as the neural network output, which makes them indistinguishable. Our framework can launch the attack successfully with the adoption of three data representation procedures (data augmentations, prediction-label concatenation, and patch cropping). This shows that the combination of the three data representation procedures is extremely helpful in extracting membership signals from data samples.

3.6.9 Discussion on Attack Methods

The reasons why general membership inference attacks work are two-fold: the overfitting property of deep learning models and data distribution of the model prediction results.

First, the overfitting issue of deep learning models causing membership inference attacks is widely discussed in prior research [19, 141, 191]. A deep learning model usually performs much better on the training dataset than on the testing dataset. Yeom *et al.* [191] demonstrated that overfitting was sufficient for an attacker to perform an attack. Salem *et al.* [141] discovered that a more overfitted deep learning model would be more vulnerable to membership inference attacks. Chen *et al.* [19] discussed that overfitting also caused membership inference attacks on generative adversarial networks.

Second, the model prediction results show different data distributions. Figure 3.12 shows a histogram of cross entropy loss values of member and non-member samples for the decoder loss of the victim model PSPNet in our experiment. We can see that member samples have small and concentrated loss values, while non-member samples have larger and more widely spread loss values. The deep learning model is trained by member samples, leading to two different data distributions of loss values. Membership inference attacks can leverage this difference to extract membership signals.

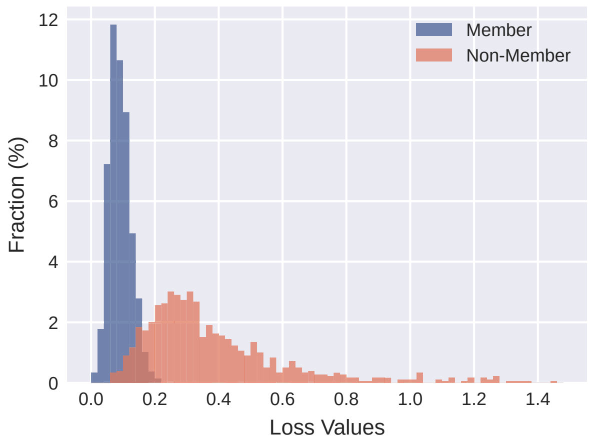


Figure 3.12: A histogram of cross entropy loss values of member and non-member samples for the victim model.

Our label-only attack framework in semantic segmentation models is a bit different. The weak membership signals in a single pixel and the limited information from prediction results make our attack framework harder than general attacks. With three data representation procedures being processed (data augmentation, prediction-label concatenation, and patch cropping), our label-only attack framework is effective in semantic segmentation. We have analyzed our attack framework in the above experiments, which shows our membership inference attacks can successfully extract private information from the victim model.

3.6.10 Defense Experiment 1: Dropout

In our first defense experiment, we evaluate the defense performance using dropout. As our task is for semantic segmentation models, we have a dropout layer before the final classification layer in the victim model. We set a dropout ratio (ranging from 0 to 1) for the dropout layer and activate it during training. In our experiment, the dropout ratio is set to 0.1 (default), 0.3, 0.5, 0.7, 0.9. The other settings are similar to prior dependent attacks. We apply translation data augmentation, SLM concatenation and Slide patch-cropping strategies when preparing the dataset of the attack model.

The experiment result is shown in Figure 3.13. We can see that as the dropout ratio increases, the utility performance only decreases a little (from 44.3% to 43.74%), and the privacy performance also decreases a little (attack accuracy from 0.81 to 0.78). And they do not decrease monotonously. As a result, we can conclude that dropout is not an effective defense mechanism against the proposed label-only membership inference

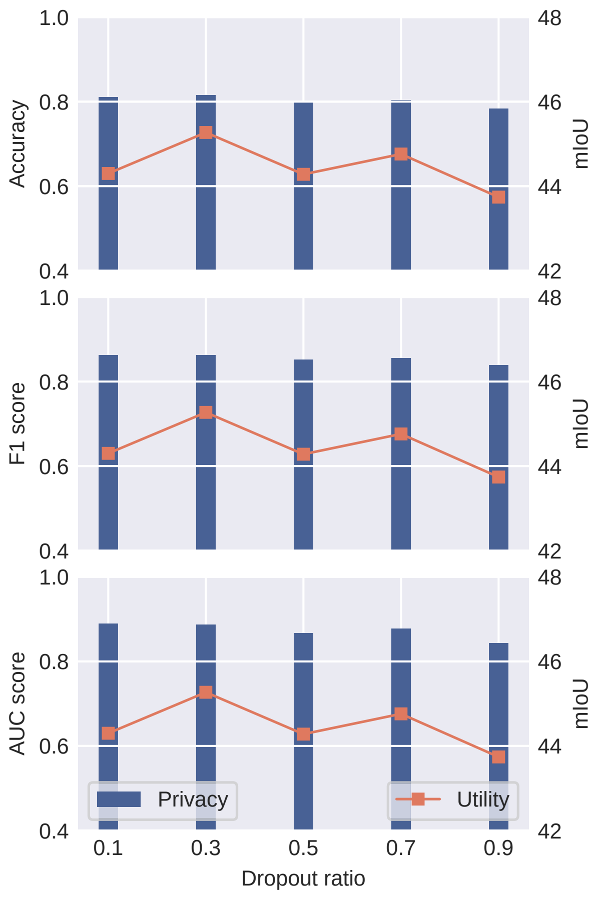


Figure 3.13: Evaluation of the defense performance using dropout.

attacks in semantic segmentation models.

3.6.11 Defense Experiment 2: DPSGD

In our second defense experiment, the defense performance using DPSGD is evaluated. We apply DPSGD in the victim model in dependent label-only attacks using the PSPNet and Cityscapes dataset. We adopt translation data augmentation, SLM concatenation and Slide patch-cropping strategies when processing data representation. We set the gradient norm bound C to 48.0 based on the gradient norm distribution and δ to 6×10^{-4} based on the size of the victim dataset. We evaluate the utility and privacy performance under different noise multipliers (the squared Gaussian noise scale σ).

From the experiment result in Figure 3.14, we can see that as the noise multiplier increases, both the utility and privacy performance decrease. When the noise multiplier ranges from 0 to 0.004, the utility performance (mIoU) decreases by over 15%, and the privacy performance (attack accuracy) decreases from above 0.8 to around 0.4. A

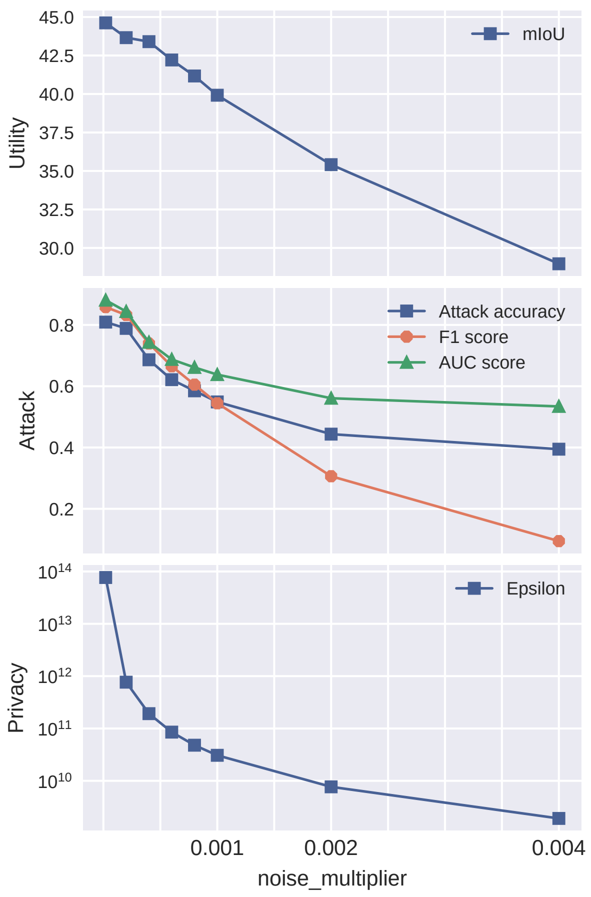


Figure 3.14: Evaluation of the defense performance using DPSGD.

relatively balanced result is when the noise multiplier is 0.0008, the mIoU drops 3%, and the attack accuracy reaches 0.58. The differential privacy budget ϵ in this experiment is tremendously large, ranging from 10^9 to 10^{13}. Theoretically, large ϵ values can not provide meaningful differential privacy guarantees. This means that when DPSGD is used in a large deep learning model (Resnet is often selected as the backbone in semantic segmentation models with large numbers of layers), a very small noise added in the model gradients can cause the membership inference attacks to fail. The work of [66, 130] draw similar conclusions when DPSGD is adopted to defend the membership inference attacks.

3.6.12 Discussion on Defense Mechanisms

We discuss and evaluate the defense performance against the proposed label-only attacks using dropout and differential privacy. The dropout strategy has little effect on our attacks, which fails to reduce the attack accuracy with any dropout ratio. Although the DPSGD strategy is more effective than the dropout strategy on our attacks, it reduces both mIoU and the attack accuracy. With the precise tuning of the DPSGD hyper-parameters, we have managed to find a balanced result with the lowered attack accuracy and a reasonable mIoU. The balanced result is not ideal, proving that the defense mechanism against our proposed attack method can only be achieved at the cost of learning degradation, which is the crucial reason for the success of our proposed attack method.

Defense mechanisms can protect the victim model from attacks, but they also jeopardize the model's performance. In this light, we should be aware of the privacy-utility trade-off when training models with defense mechanisms.

3.7 Conclusion

In this chapter, we have established a well-designed framework for label-only membership inference attacks against semantic segmentation models. We apply different data augmentations to the original data and design the data representation procedures to generate datasets for the attack model. Our ablation analysis was conducted under various experimental settings. We conclude that with seven various data augmentations, several prediction-label concatenations, and patch-cropping strategies, the label-only membership inference attacks can achieve a competitive performance compared to the previous work. We also demonstrate that the label-only attacks can be extended to other popular computer vision tasks, such as semantic segmentation. Our future work is to evaluate our attack methods with more defense mechanisms and analyze more about the effect of data augmentations on membership inference attacks.

CHAPTER 4

Private and Personalized Federated Learning

In this chapter, we propose a new defense method for federated learning. Federated learning is a distributed learning paradigm where a global model is trained using data samples from multiple clients but without the necessity of sharing raw data samples. However, it comes with several significant challenges in system designs, data quality, and communications. Recent research highlights a significant concern related to data privacy leakage through reserve-engineering model gradients at a malicious server. Moreover, a global model cannot provide good utility performance for individual clients when the local training data is heterogeneous in terms of quantity, quality, and distribution. Hence, personalized federated learning is highly desirable in practice to tailor the trained model for local usage. In this chapter, we propose PPFed, a unified federated learning framework to simultaneously address privacy preservation and personalization. The intuition of our framework is to learn part of the model gradients at the server and the rest of the gradients at the local clients. To evaluate the effectiveness of the proposed framework, we conduct extensive experiments across four image classification datasets to show that our framework yields better privacy and personalization performance compared to the existing methods. We also claim that privacy preservation and personalization are essentially two facets of deep learning models, offering a unique perspective on their intrinsic interrelation. [1]

[1]The content of this chapter is based on [203]. Guangsheng Zhang was the first author of this paper.

4.1 Background

The Internet of Things (IoT) involves the integration of diverse devices and systems, allowing them to collect, store, and analyze data for various purposes. The usage of extensive IoT devices, including a wide range of cameras and sensors, generates large amounts of user data that are in need of processing. Employing machine learning and deep learning technologies is necessary to handle this data influx. Nevertheless, the escalating volume of data poses challenges for conventional machine learning and deep learning. Moreover, utilizing these generalized approaches may expose sensitive information from local IoT devices. This vulnerability underscores the significance of adopting more secure and privacy-centric strategies.

In response to these concerns, federated learning has emerged as a prominent decentralized learning paradigm tailored for large-scale IoT deployments. In federated learning, a centralized model is trained by leveraging the computational power and data from local clients (i.e., IoT devices). The idea of federated learning is to preserve the privacy of local training data, as the training samples are kept at the local clients and cannot be accessed by the centralized server [118].

One of the most crucial factors in federated learning is the issue of data heterogeneity. Data heterogeneity refers to the fact that the underlying data distribution in each client could be much different. When a centralized model is trained at the server and sent back to each client, the local model's utility will be highly affected by heterogeneous data from other clients. To solve this issue, personalized federated learning is developed because a personalized local model can help to improve the local model's utility in a heterogeneous environment. The existing approaches achieve this goal by applying personalization layers in a neural network [3, 30], utilizing both global and local models [33, 58], requiring multi-task learning [21, 96], leveraging meta learning [41], or applying additional mechanisms to guarantee personalization [21, 159]. However, these approaches establish various frameworks to solve model personalization, and the other factors in federated learning can not be appropriately addressed in these frameworks.

The issue of privacy preservation is another crucial factor. Although federated learning intuitively preserves data privacy, recent research has shown the risk of privacy leakage through reserve-engineering model gradients at a curious or malicious server [52, 192, 219]. Such attacks, known as gradient inversion attacks, involve the adversary on the server side reconstructing the training data by leveraging model gradients. The private information of the client's training data is therefore leaked to the server.

Both standard and personalized federated learning frameworks suffer the threat of these attacks. To defend against such attacks, several approaches are noteworthy, including differential privacy [53, 144], data encryption [122], and data compression [154, 219]. However, these methods either require high computation loads or lead to the lower utility of target models. This raises a fundamental question: is there an approach that preserves data privacy while achieving high model utility in federated learning?

In this light, we propose PPFed, a **p**rivacy-preserving and **p**ersonalized **fed**erated learning framework for IoT. This is a non-trivial task because it needs to achieve both privacy preservation and personalization for deep learning models. Nonetheless, several challenges persist: Firstly, privacy preservation of training data in federated learning is in high demand. Privacy leakage from model gradients has not been well solved. Many defense mechanisms struggle to strike a harmonious balance between privacy and utility. Secondly, a personalized model framework should obtain comparable performance to a single global model framework. The performance of current personalized frameworks varies in different settings. Thirdly, there have been no frameworks that can achieve both privacy preservation and personalization in federated learning.

To tackle these challenges, our framework exploits the inner representations of deep learning models and introduces perturbations to a part of model gradients. Employing a splitting strategy, we partition the model into global and local layers. Notably, our experiments reveal that layers housing private information also encompass personalized data. PPFed is a unified solution for two major factors (i.e., privacy preservation and personalization) in federated learning. In comparison to previous privacy-preserving approaches [1, 154, 219], PPFed preserves privacy by defending against gradient inversion attacks while delivering better utility performance and maintaining practical computation costs. Furthermore, compared to previous personalization frameworks [3, 30, 118, 127], PPFed achieves competitive personalization performance through the implementation of splitting layer strategy.

We summarize the contributions of our work as follows:

- We propose a privacy-preserving federated learning framework leveraging personalization mechanisms, which can address both the issues of privacy leakage from model gradients and the need for effective model personalization methods.

- Our proposed framework ensures privacy preservation, especially against gradient inversion attacks, by splitting the model into global layers (i.e., shallower layers that capture the low-level and common features) and local layers (i.e., deeper layers that capture the high-level and sensitive features). The clients meticulously train all the model parameters, introduce perturbations to local layers, and then send the gradients of all layers to the server. The server aggregates all the global layer gradients and then distributes the updated gradients back to the clients. Therefore, the malicious server only has valid gradients of the global layers and cannot launch successful attacks. In addition, we inspect the threshold of splitting global layers from local layers and reveal that setting the model head as local layers is sufficient to ensure robust privacy preservation.

- From the personalization standpoint, our framework ensures model personalization by training local layers separately on each client. We also verify that employing multiple local epochs and multiple fine-tuning epochs contributes to improving the personalization performance.
- Our extensive experiments demonstrate that our framework outperforms other defense mechanisms and, at the same time, matches the personalization performance compared with other federated learning frameworks.

4.2 Related Work

4.2.1 Federated Learning and Personalized Federated Learning

Federated learning was applied to train a centralized model with decentralized data located in large amounts of clients [88, 118]. The research aims to solve multiple issues: various data partition patterns, privacy protection, framework architectures, and data/-model heterogeneity [53, 118]. Personalized federated learning is strongly related to the last one.

A personalized federated learning framework was in demand because of the different data distributions among the local clients. A global model aggregated in the server cannot provide enough utility to local clients. There are several approaches to achieving personalization in federated learning. Among these approaches, applying personalization layers is a strategy to implement model personalization. Arivazhagan *et al.* [3] retrained the personalization layers in local clients. Liang *et al.* [103] provided a framework where local layers are shallow layers, and global layers are deeper layers. Collins *et al.* [30] dissected the model into global and local layers to achieve personalization. Oh *et al.* [127] only updated the global layers in federated learning training and then updated the local layers during fine-tuning in local clients.

There are also other methods to achieve personalized federated learning, including mixing global and local models [33, 58], utilizing multi-task learning [21, 96], adopting meta-learning [41], and some other strategies [21, 159]. Although these federated learning frameworks achieve personalized for local data, data privacy is not guaranteed. Our method follows the steps of prior research and forms a unified privacy-preserving and personalized framework.

Split learning is another related distributed machine learning approach in which the distribution target is the deep learning model rather than the training data. In split learning, the neural network is divided into a client-side network and a server-side network [56, 168]. In this protocol, shallower layers are trained on the client side, while deeper layers are trained on the server. The clients can share the network's intermediate features with the server, and the server manages the remaining training process. Although split learning provides a global model like conventional federated learning, it has no personalization capability.

4.2.2 Gradient Inversion Attacks and Corresponding Defenses

Here, we introduce related work on gradient inversion attacks **that is specifically relevant to this chapter**. The adversary aims to reconstruct training samples through model parameters in gradient inversion attacks. The server does not directly access the clients' training samples in federated learning. However, an honest-but-curious server can still recover private data through the updated model gradients [52, 59, 192, 219]. DLG attacks [219] were the first to reconstruct training samples from model gradients. IG attacks [52] also reconstructed data leveraging model gradients. Unlike DLG attacks using Euclidean distance, IG attacks used cosine similarity to optimize reconstruction cost between raw and generated gradients.

Existing privacy-preserving technologies against gradient inversion attacks for federated learning can be categorized into three groups: differential privacy, encryption-based mechanisms, and data manipulation-based mechanisms. Differential privacy provides a mathematical guarantee for the privacy preservation of client data [144, 175, 220]. However, adding noises on the client side based on differential privacy reduces the performance of the targeted applications. Encryption-based mechanisms consist of approaches of secure multi-party computation and homomorphic encryption [122, 199]. However, the major concerns of utilizing encryption-based mechanisms are the large communication cost (the expense of communication between the server and the client) and the computation cost (the time for encrypting the client data) compared with other mechanisms.

Table 4.1: Summary of notations used throughout the chapter.

Notation	**Description**
S	The server
f	Client fraction ratio
t	Communication round between server and clients
$C = \{C_1, C_2, ..., C_N\}$	A set of clients
$C^t = \{C_1, C_2, ..., C_k\}$	A set of selected clients at t
$\mathscr{F}$	A model for the task
$D = \{X, Y\}$	Dataset for the model with a set of data samples and corresponding ground truth labels
θ	Parameters of the model
$\nabla\theta$	Gradients of the parameters
$\mathscr{L}$	Loss function of the task
E	Local epoch for client training
τ	Fine-tuning epoch after client training
B	Batch size for client training
κ	Split threshold for splitting the model layers
$\mathbb{G}$	For global model
$\mathbb{L}$	For local model
ψ	Privacy preservation performance
ζ	Personalization performance

The third group, data manipulation-based mechanisms, aims to prune the gradients below a fixed threshold [219] or manipulate input labels [98]. However, this approach could result in poor utility performance. Our proposed method can match the other privacy-preserving technologies while reducing the computation cost and guaranteeing the model's utility.

4.3 Preliminaries and Problem Statement

In this section, we introduce the preliminaries of the federated learning training procedure, as well as the problem of privacy leakage in model gradients in federated learning. The notations used in this chapter are given in Table 4.1.

4.3.1 Training Procedure in Federated Learning and Personalized Federated Learning

There are multiple clients and a central server in a training process in a standard federated learning framework. A set of clients is denoted as $C = \{C_1, C_2, ..., C_N\}$ when the number of clients is N. The total communication round between the server and the clients is T. At the t-th communication round ($t \leq T$), a set of selected clients $C^t = \{C_1, C_2, ..., C_k\} (k \leq N)$ participate in the training process and send their gradients to the central server S. The number of selected clients k is determined by the client fraction ratio f, and $k = f \times N$. Then, the server S calculates a centralized model with the parameters received instead of training samples for each client. The goal of federated learning is to minimize the losses across all the clients C:

$$\min_{(\mathscr{F}_1,...,\mathscr{F}_N)} \frac{1}{N} \sum_{i=1}^{N} \mathscr{L}_i(\mathscr{F}_i), \tag{4.1}$$

where $\mathscr{L}_i$ and $\mathscr{F}_i$ are the loss function and learning model for client C_i. This step is repeated until the loss function converges or the desired accuracy is achieved.

In a supervised learning setting, the data for the client C_i is formed as $(X_i, Y_i) \sim D_i$, where X_i and Y_i are data samples and corresponding ground truth labels, and D_i is the data distribution on C_i. In real-world applications, the data distribution could be different across all local datasets in federated learning, which is called the issue of data heterogeneity. The client C_i^t at the t-th communication round is one of the selected clients, where $C_i^t \in C^t$. We also have different data distributions from D_1 to D_N for different clients from C_1 to C_N. Apparently we have $D_i \in \{D_1, ..., D_N\}$. The global data distribution for the whole federated learning procedure is $D_{\mathbb{G}}$, where $D_{\mathbb{G}} = \{D_1, ..., D_N\}$.

The gradients $\nabla\theta_i^t$ from the client C_i at the communication round t is computed as follows:

$$\nabla\theta_i^t = \frac{\partial \mathscr{L}_i(\mathscr{F}(x_i, \theta_i^t), y_i)}{\partial \theta_i^t}, \tag{4.2}$$

where x_i and y_i are the images and ground truth labels from C_i, $(x_i, y_i) \in (X_i, Y_i)$. The client C_i can either send the gradient $\nabla\theta_i^t$ or the parameters θ_i^t to the global server. If the server receives gradients, the aggregation process is $\theta_{\mathbb{G}}^{t+1} = \theta^t - \eta\frac{1}{N}\sum_{i=1}^{N}\nabla\theta_i^t$. If the server receives parameters, it is $\theta_{\mathbb{G}}^{t+1} = \frac{1}{N}\sum_{i=1}^{N}\theta_i^t$. Here, N is the number of clients, and η is the learning rate. $\theta_{\mathbb{G}}^{t+1}$ is sent back to each client as the parameters for the global model for the communication round $t+1$.

In a federated learning paradigm, the optimization objective is all about the parameters of the model θ. The loss function $\mathscr{L}_i(\mathscr{F}_i)$ is an expected risk over $\mathscr{D}_i$, and we have:

$$\mathscr{L}_i(\mathscr{F}_i) = \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i, \theta_i), Y_i]. \tag{4.3}$$

In standard federated learning, all the clients share a global model with parameters $\theta_{\mathbb{G}}$, and $\theta_{\mathbb{G}} = \theta_1 = ... = \theta_N$. Based on Equations (4.1) and (4.3), the objective of learning can be described as:

$$\min_{\theta_{\mathbb{G}}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i, \theta_{\mathbb{G}}), Y_i]. \tag{4.4}$$

In contrast, in personalized federated learning, $\theta_G \neq \theta_1 \neq ... \neq \theta_N$, and the objective is to optimize all the parameters from θ_1 to θ_N. With Equations (4.1) and (4.3), we have the objective as:

$$\min_{(\theta_1,...,\theta_N)} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i,\theta_i),Y_i]. \tag{4.5}$$

4.3.2 Threat Model

The adversary's objectives. Our threat model follows the work of gradient inversion attacks [52, 219]. In a federated learning paradigm, the adversary hides in the server and aims to extract private information from a specific client.

The adversary's knowledge. The adversary has access to the parameters or gradients sent from each client. In most personalized federated learning approaches, the model's all parameters or gradients are sent to the server, to which the adversary has access. In other approaches, the model splits into global and local layers, and only the model's global layer parameters or gradients are sent to the server. In this situation, the adversary can only access these parameters or gradients.

The adversary's capability. The adversary has no intention of compromising the training procedure on the server. The adversary can take further actions to the received parameters or gradients to try to reconstruct data samples from clients.

4.3.3 Privacy Leakage

Now, we explain how privacy is leaked from the server. The adversary in the server aims to extract the private information from the client Ci given $\nabla\theta_i^t$. Following Equation (4.2), a dummy image (randomly initialized image) $\hat{x}$ and its corresponding ground truth $\hat{y}$ is initialized to calculate the gradients $\nabla\hat{\theta}$:

$$\nabla\hat{\theta} = \frac{\partial\mathscr{L}(\mathscr{F}(\hat{x},\theta_i^t),\hat{y})}{\partial\theta_i^t}. \tag{4.6}$$

The attack is an optimization procedure by minimizing the difference between the given gradients $\nabla\theta_i^t$ and the generated gradients $\nabla\hat{\theta}$:

$$x^* = \arg\min_{x^*} \mathscr{L}_{grad}(\nabla\theta_i^t - \nabla\hat{\theta}), \tag{4.7}$$

where x^* is the reconstructed image. If x^* is recognizable and the difference between x and x^* is below a reconstruction threshold, the privacy of x is leaked.

4.3.4 Problem Statement

4.3.4.1 Privacy Preservation

Our primary target is to overcome the privacy leakage issues in federated learning, i.e., when the adversary tries to initiate the attack, we would like to maximize the difference between the data samples and the reconstructed data samples. Here, we have the privacy preservation performance ψ defined as:

$$\psi = dist(x, x^*). \tag{4.8}$$

In practice, it can be achieved by maximizing the mean square error (MSE) or minimizing the peak signal-to-noise ratio (PSNR).

4.3.4.2 Performance Constraints

Apart from privacy preservation, we want to achieve good performance at the same time. Therefore, there are two other aspects considered in our proposed federated learning framework: personalized performance and complexity.

Personalized performance (accuracy). There is a trade-off between privacy preservation and personalized performance (utility). The performance of the trained model in federated learning should be monitored while some privacy preservation mechanisms are applied in the federated learning system.

The utility of federated learning refers to the performance of the global model over all the clients, denoted as $\zeta_{\mathbb{G}}$:

$$\zeta_{\mathbb{G}} = \zeta[\mathscr{F}(\theta_{\mathbb{G}})]. \tag{4.9}$$

However, for personalized federated learning, the utility is the average performance of all the local models over all the clients, denoted as $\zeta_{\mathbb{L}}$:

$$\zeta_{\mathbb{L}} = \frac{1}{N}\sum_{i=1}^{N} \zeta[\mathscr{F}(\theta_i)]. \tag{4.10}$$

Our work tries to achieve at least comparable personalized performance to other federated learning frameworks.

Complexity. Here, the complexity performance indicates the computational overhead in the framework. The computational overhead when applying privacy preservation mechanisms is also a key factor, especially in distributed systems. The proposed framework should guarantee an affordable computational overhead both in the server and the clients. During the federated learning training procedure, it is essential to monitor how much time is needed for running the framework. The frameworks are evaluated with FedAvg set as 1 unit as a baseline. The time cost of the other frameworks is aligned with that of FedAvg. We provide the evaluation of complexity performance in our experimental results.

4.4 PPFed: Privacy-Preserving and Personalized Federated Learning

In this section, we will introduce the proposed **P**rivacy-Preserving and **P**ersonalized **Fed**erated Learning (PPFed), a unified framework that can defend against gradient inversion attacks and achieve model personalization at the same time.

Our solution is inspired by research from representation learning and interpretable machine learning. Bengio *et al.* [10] showed that common representations could be obtained from various data samples in neural networks, even though these data samples should be classified differently. Common representations can be learned by some neural network layers. Zeiler *et al.* [198] presented a deconvolutional visualization framework showing that the feature map of each layer could be visualized. They concluded that the common texture features of the image input could be extracted in the shallow layers and the detailed classification features in the deep layers. More research in [116] further demonstrated that different convolutional layers appeared to be capturing different types of structures in images, from lines and curves to instance-specific information.

Based on prior research, can the server and the clients learn different parts of the model to resolve the privacy issues? In our framework, instead of aggregating all the model gradients in the server, we allow the server to receive part of the valid model gradients to form a common representation of the data. Then, each client trains the rest of the model gradients to learn their own representation of the data. The design of the PPFed framework is introduced in detail in the next subsection.

4.4.1 Design of the Framework

As shown in Figure 4.1, we have one server and multiple clients distributed across various locations in our PPFed framework. Like standard federated learning, we have stages of client training, server updates, and communications between the server and

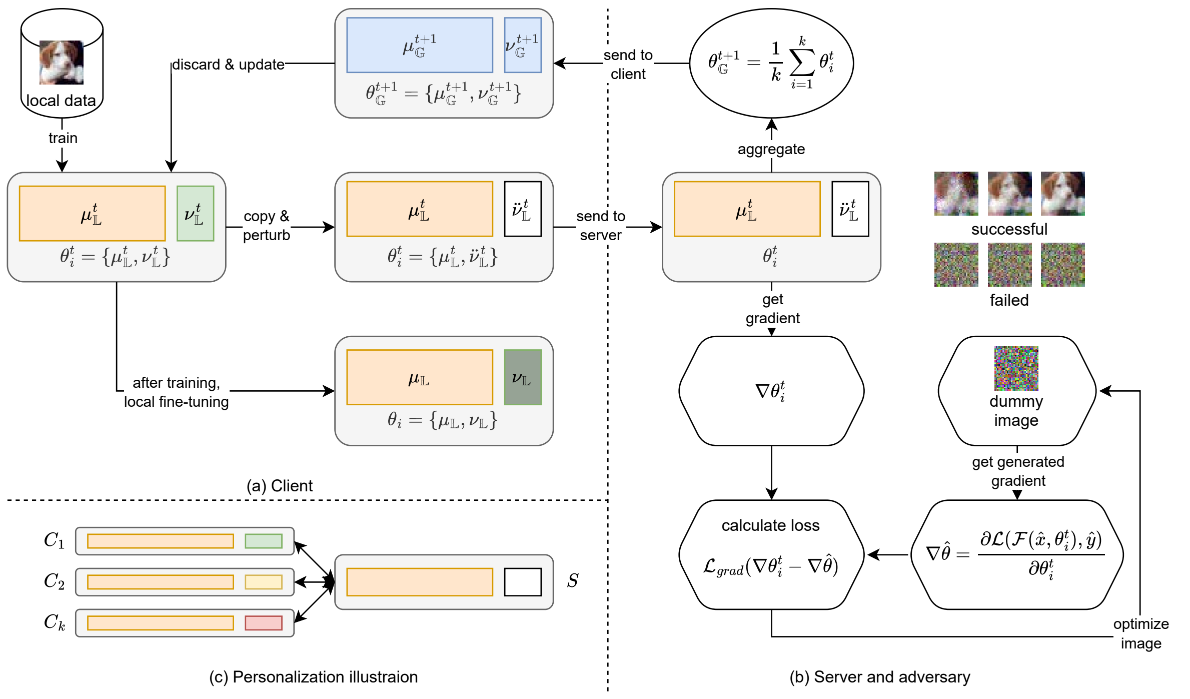


Figure 4.1: An illustration of our framework: (a) The client-side process; (b) The server-side process with the attack procedure by the adversary; (c) An illustration of personalized local layers in each client.

the clients. During the training stage, the server communicates with several clients, and their goal is to train a model $\mathscr{F}$ with global layers $\mathscr{F}_{\mathbb{G}}$ and local layers $\mathscr{F}_{\mathbb{L}}$.

4.4.1.1 Model Splitting

The Client C_i receives the parameters $\theta_{\mathbb{G}}$ from the server. We would like to split the model into global and local parts, where the clients share the same global layers and have their own local layers. μ and ν denote the parameters of global layers and local layers. We use a split threshold κ to show the index where the local layers begin in the model: $\mu = \theta_{\mathbb{G}}[0:\kappa]$, $\nu = \theta_{\mathbb{G}}[\kappa:end]$. The model can learn a common representation with μ first. Then, in each client, ν is optimized to ensure personalization.

4.4.1.2 Clients

Figure 4.1 (a) and Algorithm 3 illustrate the procedure in the local clients. It consists of three major steps:

- **Discard and update** (lines 1-2 in Alg. 1). At the communication round t, the client C_i^t receives $\theta_{\mathbb{G}}^t$ from the server. We can split $\theta_{\mathbb{G}}^t$ into parameters for global layers

Algorithm 3 PPFed: Client C_i

```
Input: communication round $t$, $\theta^t_{\mathbb{G}}$, model $\mathscr{F}$
Output: $\mu^t_{\mathbb{L}}$, $\ddot{v}^t_{\mathbb{L}}$
1: When $C_i$ is selected at $t$:
2: Split $\theta^t_{\mathbb{G}}$ into $\mu^t_{\mathbb{G}}$ and $v^t_{\mathbb{G}}$
3: Train $\mathscr{F}$ with $\mu^t_{\mathbb{G}}$ to get $v^t_{\mathbb{L}}$ (Equation (4.11))
4: Train $\mathscr{F}$ with $\mu^t_{\mathbb{G}}$ and $v^t_{\mathbb{L}}$ to get $\mu^t_{\mathbb{L}}$ and $v^t_{\mathbb{L}}$ (Equation (4.12))
5: Add perturbations to $v^t_{\mathbb{L}}$ to get $\ddot{v}^t_{\mathbb{L}}$ (Equation (4.13))
6: return $\mu^t_{\mathbb{L}}$, $\ddot{v}^t_{\mathbb{L}}$
```

$\mu^t_{\mathbb{G}}$ and for local layers $v^t_{\mathbb{G}}$, and we only need $\mu^t_{\mathbb{G}}$ for local client training. We then discard $v^t_{\mathbb{G}}$.

- **Local client training** (lines 3-4 in Alg. 1). We first train parameters for local layers (denoted as $v^t_{\mathbb{L}}$) by freezing the received parameters for global layers $\mu^t_{\mathbb{G}}$.

$$v^t_{\mathbb{L}} = \arg\min_{v} \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i,\mu^t_{\mathbb{G}},v),Y_i]. \tag{4.11}$$

  Then we update parameters for global and local layers, denoted as $\mu^t_{\mathbb{L}}$ and $v^t_{\mathbb{L}}$.

$$\mu^t_{\mathbb{L}}, v^t_{\mathbb{L}} = \arg\min_{(\mu,v)} \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i,\mu,v),Y_i]. \tag{4.12}$$

  During the local client training, we can have one or more local epochs (i.e., $E \geq 1$).

- **Copy and perturb** (lines 5-6 in Alg. 1). Now, we are preparing to send back the parameters to the server. $\mu^t_{\mathbb{L}}$ can be directly sent back without change. We keep $v^t_{\mathbb{L}}$ in local clients and send back $\ddot{v}^t_{\mathbb{L}}$, a perturbed copy of $v^t_{\mathbb{L}}$. There are several approaches to add perturbations in our framework: pruning to zero, randomizing the parameters, or adding randomized perturbations, which can be defined as follows:

$$\begin{aligned} \ddot{v}^t_{\mathbb{L}} &= v^t_{\mathbb{L}} \times 0.0, \\ or\ \ddot{v}^t_{\mathbb{L}} &= rand(0,1), \\ or\ \ddot{v}^t_{\mathbb{L}} &= v^t_{\mathbb{L}} + rand(0,1). \end{aligned} \tag{4.13}$$

4.4.1.3 Server

Figure 4.1 (b) and Algorithm 4 describe the procedure in the server. At the communication round t, the server S receives $\mu^t_{\mathbb{L}}$ and $\ddot{v}^t_{\mathbb{L}}$ from each client. We denote parameters from

Algorithm 4 PPFed: Server S

```
Input: Total communication round $T$, model $\mathscr{F}$, $t = 1$
Output: $\theta_{\mathbb{G}}$
 1: while $t < T$ do
 2:     Select $k$ clients $C^t = \{C_1, C_2, ..., C_k\}$
 3:     $i = 1$
 4:     while $i < k$ do
 5:         Receive $\theta_i^t$ from $C_i^t$
 6:         $i = i + 1$
 7:     end while
 8:     Aggregate $\theta_{\mathbb{G}} = \theta_{\mathbb{G}}^{t+1} = \frac{1}{k}\sum_{i=1}^{k}\theta_i^t$ (Equation (4.14))
 9:     Send $\theta_{\mathbb{G}}$ to each client
10:     $t = t + 1$
11: end while
```

Table 4.2: Comparison with other privacy preservation mechanisms.

Method	Pruning	Adding perturbation	Scope
Gradient compression [219]	Yes	No	All layers
Differential privacy [1]	No	Yes	All layers
Soteria [154]	No	Yes	Selected layers
PPFed (Ours)	Yes	Yes	Selected layers

the client C_i^t as θ_i^t, where $\theta_i^t = \{\mu_{\mathbb{L},i}^t, \ddot{v}_{\mathbb{L},i}^t\}$. The parameters are aggregated in the server:

$$\theta_{\mathbb{G}}^{t+1} = \frac{1}{k}\sum_{i=1}^{k}\theta_i^t. \tag{4.14}$$

After the aggregation is completed, the global parameters $\theta_{\mathbb{G}}^{t+1}$ will be sent to each client.

4.4.1.4 Fine-tuning After Federated Learning Training

After the federated learning training between the server and the clients, each client can have a fine-tuning procedure to ensure the local model's performance is well suited for the local dataset, as shown in Figure 4.1 (a). We can have one or more fine-tuning epochs (i.e., $\tau \geq 1$).

4.4.1.5 Comparisons with other methods

We provide a comparison with other privacy preservation mechanisms in Table 4.2. There are three approaches to adding perturbations in our framework: pruning, randomizing and adding random perturbation noises. These perturbations are selectively applied to specific layers (local layers in our case), setting our framework apart from gradient

Table 4.3: Comparison with other personalization frameworks.

Method	**Parameters to clients**	**Local parameters update**	**Client training steps**	**Parameters back to server**	**Fine-tuning**
FedAvg [118]	All	Yes	Single	All	No
FedPer [3]	Global	Yes	Single	Global	No
FedRep [30]	Global	Yes	Multi	Global	No
FedBABU [127]	Global	No	Single	Global	Yes
PPFed (Ours)	All	Yes	Multi	All (Partly perturbed)	Yes

compression and differential privacy. Our framework has a better utility and privacy trade-off and is suitable for personalized federated learning compared to gradient compression and differential privacy. Soteria has a masking strategy on chosen layers, which is also different from our framework. Compared to Soteria, the pruning and perturbation strategy in our framework is straightforward to implement and exhibits a better performance. These differences make our framework stand out in these privacy preservation mechanisms, and we present the experimental results in Section 4.5.3.

In addition, we summarize a comparison with other personalized federated learning frameworks in Table 4.3. This comparison highlights the unique aspects of our framework at each stage of the federated learning process. Distinctively, our framework differs from others in several key aspects: a) the clients receive all the parameters but only utilize parameters of global layers; b) we use multiple steps in client training to optimize parameters of global and local layers separately; c) after the client training, the clients send all the parameters (part of the parameters are perturbed) to the server, which can ensure the data privacy and personalization of each client. In contrast, all other frameworks have their own partial disadvantages. FedAvg, FedPer, and FedBABU employ a single training step at the client level. Furthermore, FedAvg, FedPer, and FedRep lack fine-tuning stages following their training processes. Our framework, on the other hand, incorporates these elements to significantly improve personalization performance. Additionally, the absence of perturbation procedures in other frameworks exposes them to privacy risks. In contrast, our framework integrates perturbations in local layers, maintaining privacy protection. We compare the experimental results with these personalization frameworks in Section 4.5.4.

4.4.2 Theoretical Analyses for Privacy Preservation

Our framework is effective in privacy preservation for local data samples, which can defend against gradient inversion attacks when the adversary has access to data in the server in federated learning. Here, we present theoretical analyses of privacy preservation of our framework.

Proposition 1: *Given a data sample x with its calculated gradients $\nabla\theta$, and a reconstructed sample x^* with its reconstructed gradients $\nabla\hat{\theta}$, perturbations can be added to gradients $\nabla\theta$ to improve the privacy preservation performance ψ.* $\square$

Proof 1: *We add perturbations to $\nabla\theta$ to be $\nabla\ddot{\theta}$. Because of perturbations, the reconstruction from the adversary changes to x' and $\nabla\hat{\theta}'$. And Equation (4.7) becomes:*

$$x' = \arg\min_{x'} \mathscr{L}_{grad}(\nabla\ddot{\theta} - \nabla\hat{\theta}'). \tag{4.15}$$

This would significantly raise the reconstruction error. Then the distance between x' and x^* would increase. As stated in Equation (4.8), $\psi = dist(x, x^*)$. Now we have $\psi' = dist(x, x')$. The increasing distance between x' and x^* will eventually cause ψ' to be a better privacy preservation performance than ψ. $\square$

Proposition 2: *Given a data sample x with its gradients $\nabla\theta$, and a reconstructed sample x^* with its reconstructed gradients $\nabla\hat{\theta}$, perturbations can be added to* ***part of gradients*** *$\nabla\theta$ to improve the privacy preservation performance ψ.* $\square$

Proof 2: *In our framework, the parameters θ from the client can be split into $\mu_{\mathbb{L}}$ and $v_{\mathbb{L}}$. We add perturbations to $v_{\mathbb{L}}$ to be $\ddot{v}_{\mathbb{L}}$. In this way, the gradients $\nabla\theta$ received by the adversary can be split into $\nabla\mu_{\mathbb{L}}$ and $\nabla\ddot{v}_{\mathbb{L}}$. Because of perturbations, the reconstruction from the adversary changes to x' and $\nabla\hat{\theta}'$. And Equation (4.7) becomes:*

$$x' = \arg\min_{x'} \mathscr{L}_{grad}(\nabla\mu_{\mathbb{L}} - \nabla\hat{\mu}') + \mathscr{L}_{grad}(\nabla\ddot{v}_{\mathbb{L}} - \nabla\hat{v}'), \tag{4.16}$$

where $\nabla\hat{\mu}'$ and $\nabla\hat{v}'$ are reconstructed gradients split from $\nabla\hat{\theta}'$. There is no way for the adversary in the server to know whether part of the gradients (i.e., $\nabla\ddot{v}_{\mathbb{L}}$) have been perturbed. It would be difficult to minimize $\mathscr{L}_{grad}(\nabla\ddot{v}_{\mathbb{L}} - \nabla\hat{v}')$, which would significantly raise the reconstruction error. Then the distance between x' and x^* would increase. Just like Proposition 1, the increasing distance between x' and x^* will eventually cause ψ' to be a better the privacy preservation performance than ψ. $\square$

4.4.3 Theoretical Analyses for Personalization

From the personalization perspective, our framework still achieves a comparable model utility to other personalization frameworks. Here, we give theoretical analyses of the personalization of our framework.

Proposition 3: *Given a set of personalized model $\mathscr{F}(\theta_i)$, and the utility $\zeta_{\mathbb{L}}$ can be improved by splitting θ_i into global parameters $\mu_{\mathbb{G}}$ and local parameters $\nu_{\mathbb{L},i}$.* □

Proof 3: *As stated in Equation* (4.5)*, the objective of personalized federated learning is to optimize θ_i (i.e. from θ_1 to θ_N) instead of $\theta_{\mathbb{G}}$. In our framework, θ is split into μ and ν, so we need to optimize $\mu_{\mathbb{G}}$ and $\nu_{\mathbb{L},i}(i = 1,..,N)$. Now, our objective can be described as:*

$$\min_{(\mu_{\mathbb{G}},\nu_{\mathbb{L},1},\dots,\nu_{\mathbb{L},N})} \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{(X_i,Y_i)\sim\mathscr{D}_i}[\mathscr{F}(X_i,\mu_{\mathbb{G}},\nu_{\mathbb{L},i}),Y_i]. \tag{4.17}$$

In this way, personalized performance $\zeta_{\mathbb{L}}$ can be achieved by optimizing the parameters of global layers and local layers separately. As local parameters $\nu_{\mathbb{L},i}$ are optimized with local data D_i and global parameters $\mu_{\mathbb{G}}$ are optimized with all the data $D_{\mathbb{G}}$, it is possible to improve the utility $\zeta_{\mathbb{L}}$ gradually. Setting multiple local epochs E and multiple fine-tuning epochs τ can help to optimize $\nu_{\mathbb{L},i}$, which improves $\zeta_{\mathbb{L}}$. □

4.5 Experiments

4.5.1 Experimental Setup

Our experiments evaluate the framework from three perspectives: (1) privacy preservation: we test our defense mechanism against several attack methods and compare it with several other existing defense methods; (2) personalization performance (i.e., utility): we compare our framework with both standard federated learning and state-of-the-art personalization methods; and (3) complexity: we provide the statistics of computational overhead in the framework compared with multiple baseline methods. In addition, before evaluating the framework, we determine the threshold κ for splitting the model into the global and local layers.

4.5.1.1 Datasets and Models

We evaluate our framework with four datasets (MNIST [93], CIFAR10 [89], ImageNet [32], and Caltech256 [55]) and two models (ConvNet and ResNet-18 [62]). ConvNet is a model with multiple convolutional layers, several max-pooling layers, and a fully connected layer at last. It has 17 gradient layers in total. ResNet-18 is an 18-layer ResNet with a fully connected layer at last as well. It has 41 gradient layers in total. Table 4.4 presents the model structures for ConvNet and ResNet-18. Convolutional layers (Conv), BatchNorm layers (BN) and fully connected layers (FC) are presented here. As we focus on the

Table 4.4: Model architectures for ConvNet and ResNet-18.

Layer Num	**ConvNet**	**Layer Num**	**ResNet-18**
1, 2	Conv 3-64, BN	1, 2	Stem: Conv 3-64, BN
3, 4	Conv 64-128, BN	3 - 6	Block 0.0: Conv 64-64, BN, Conv 64-64, BN
5, 6	Conv 128-128, BN	7 - 10	Block 0.1: Conv 64-64, BN, Conv 64-64, BN
7, 8	Conv 128-256, BN	11 - 14	Block 1.0: Conv 64-128, BN, Conv 128-128, BN
9, 10	Conv 256-256, BN	15, 16	Block 1.ds: Conv 64-128, BN
11, 12	Conv 256-256, BN	17 - 20	Block 1.1: Conv 128-128, BN, Conv 128-128, BN
12, 14	Conv 256-256, BN	21 - 24	Block 2.0: Conv 128-256, BN, Conv 256-256, BN
15, 16	Conv 256-256, BN	25, 26	Block 2.ds: Conv 128-256, BN
17	FC -10	27 - 30	Block 2.1: Conv 256-256, BN, Conv 256-256, BN
		31 - 34	Block 3.0: Conv 256-512, BN, Conv 512-512, BN
		35, 36	Block 3.ds: Conv 256-512, BN
		37 - 40	Block 3.1: Conv 512-512, BN, Conv 512-512, BN
		41	Head: FC -10

gradients calculated from each layer, the layers without gradients (i.e., activation layers, maxpool layers, and skip connections in ResNet-18) are hidden.

4.5.1.2 Privacy Preservation Experimental Setup

Attack Methods. We evaluate our defense mechanism against two state-of-the-art gradient inversion attack methods. DLG attacks [219] aimed to reconstruct the clients' training samples leveraging model gradients shared with the server. IG attacks [52] achieved better performance than DLG attacks utilizing cosine similarity when optimizing between updated and reconstructed gradients. In this chapter, we apply these attacks in their default settings (DLG attacks with L-BFGS optimizer for 300 iterations and IG attacks with Adam optimizer for 1000 iterations).

Defense Mechanisms. Our defense mechanism is compared with several existing approaches. Gradient compression [219] only sends a part of gradients above a specific threshold to the server. The difference between gradient compression and our method is that gradient compression selects gradients based on the values, while our method is based on the network layers. Differential privacy [1] (or DPSGD) adds Gaussian or Laplacian noises to the gradients and then sends them to the server. Soteria [154] perturbs the data representation of a single layer and then sends the gradients to the server. Both differential privacy and Soteria add perturbations to defend against the attacks. The difference is that differential privacy adds noises to all the model gradients, while Soteria applies noises to the representation of a single layer.

4.5.1.3 Personalization Experimental Setup

We compare our framework with several state-of-the-art methods to validate the personalization performance. FedAvg [118] is a baseline of federated learning that learns a single global model. FedPer [3] retrains the local head in clients to personalize the model. FedRep [30] is a framework that learns a shared representation and a unique head in local clients. FedBABU [127] only trains global layers first and then trains local layers at fine-tuning stages. Apart from these frameworks, we also compare with the "Local" framework, where local models are trained in each local client, and there is no single global model in this setting. We test our framework against these methods in various settings.

Hyperparameters for Federated Learning. Following the configurations in baseline federated learning method [118], we construct our datasets in non-IID settings for federated learning. We assume that there are 100 clients and one server. In a non-IID setting, each client is given two randomly selected classes of data samples. We train these datasets for 200 communication rounds. We set the local epoch E as 1, the batch size B as 16, and the client fraction f as 0.1 (meaning ten clients are selected at each communication round) for personalization experiments. For the sake of comparability, we set the batch size B as 1 for gradient inversion attacks, meaning that the adversary tries to reconstruct this specific data sample, and our method aims to defend against these attacks. This setting aims to examine the accurate reconstruction result of a single data sample. The stochastic gradient descent (SGD) optimizer is chosen for federated learning training with the learning rate η as 0.01.

4.5.1.4 Metrics

We evaluate privacy preservation using the mean square error (MSE) and peak signal-to-noise ratio (PSNR) between the original and reconstructed images. A higher MSE or lower PSNR value signifies the ineffectiveness of the data sample reconstruction in the attack process. Consequently, a better privacy-preserving mechanism is characterized by larger MSE values and lower PSNR values. Here we set the privacy preservation performance ψ (mentioned in Equation (4.8)) as ψ_{MSE} and ψ_{PSNR}. When $\psi_{MSE} > 1$ or $\psi_{PSNR} < 15$, the reconstruction result is unrecognizable by human eyes. In this case, local privacy is preserved.

The personalization performance is evaluated by model accuracy ζ (mentioned in Equations (4.9) and (4.10)). We compare with other methods based on the average client accuracy in the personalized federated learning framework. The global accuracy of the standard federated learning framework is provided as it trains a global model.

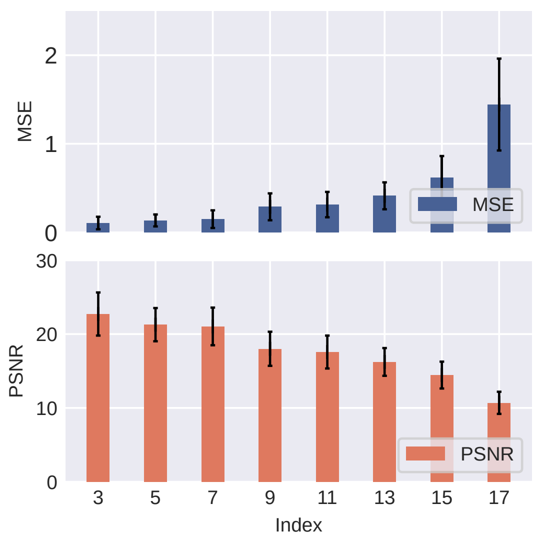
4.5.2 Finding the Optimal Split Threshold for Privacy Preservation

The split threshold κ denotes the point at which local layers begin in the model. To find the optimal κ, we start by adding perturbations to every single layer and monitor the attack performance. If the attack performance decreases when perturbations are applied to a layer indexed at i, it indicates the presence of private information within that specific layer. Subsequently, we introduce perturbations to multiple layers, starting from a layer with an index i till the last layer. We evaluate our framework in these two ways and present our analysis as follows.

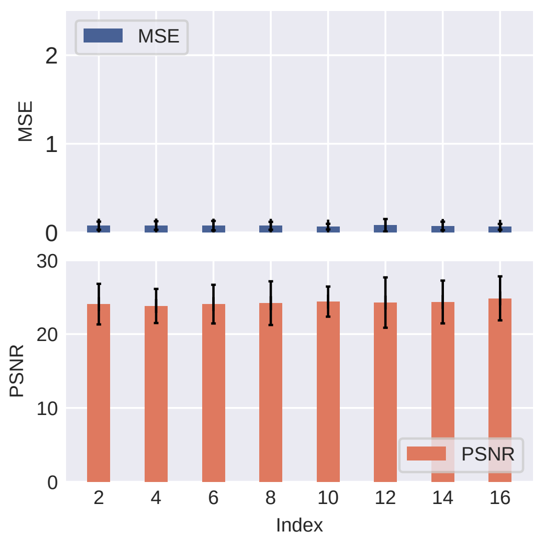
Figure 4.2 illustrates the attack performance when adding perturbations to every single layer in ConvNet and ResNet on CIFAR10. We choose the approach of pruning the gradients to zero. Convolutional layers (Conv) and BatchNorm layers (BN) are evaluated separately because they are essentially different layers. If we receive high MSE and low PSNR values, it indicates that the attack performance is not good. This suggests that our defense approach is effective, and the private information could be contained in this layer. In Figure 4.2a and Figure 4.2c, we can see that the MSE values get higher and the PSNR values get lower as the index increases. This phenomenon underscores that deeper Conv layers contain more private information, as the attacks on these deeper layers struggle to achieve satisfactory performance. In Figure 4.2b and Figure 4.2d, we observe very low MSE values and very high PSNR values. This outcome implies that the adversary can reconstruct data samples easily, meaning that there is a lack of private information in these layers.

We present the results of the attacks targeting multiple layers in Figure 4.3. We continue to employ the approach of pruning gradients to zero, and this time, the gradients of multiple layers are pruned to zero. The index denotes the starting index of the layers, and all subsequent layers are considered as local layers. The results on ConvNet and ResNet-18 reveal a consistent pattern: all the attacks have failed to provide valid reconstruction results. The MSE values of all the tests are above 1.5, and the PSNR values are below 15, making the reconstruction results unrecognizable. This shows that it is unnecessary to add perturbations to gradients from a large percentage of layers.

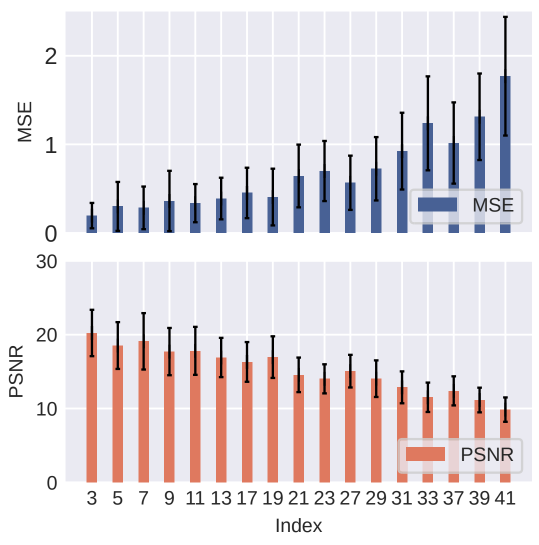
Our results confirm that private information is indeed contained within the gradients



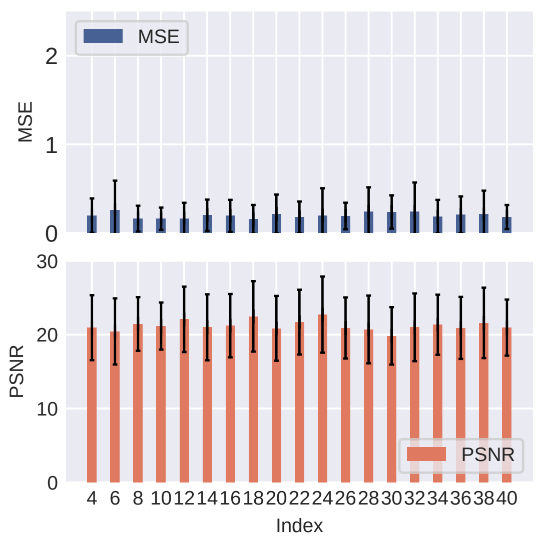
(a) Conv layers in ConvNet



(b) BN layers in ConvNet



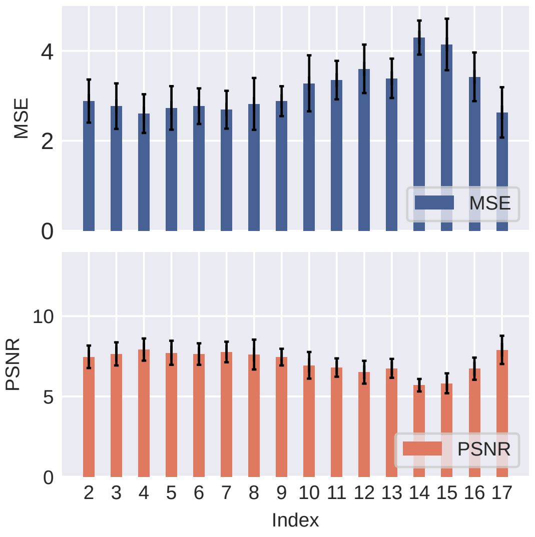
(c) Conv layers in ResNet-18



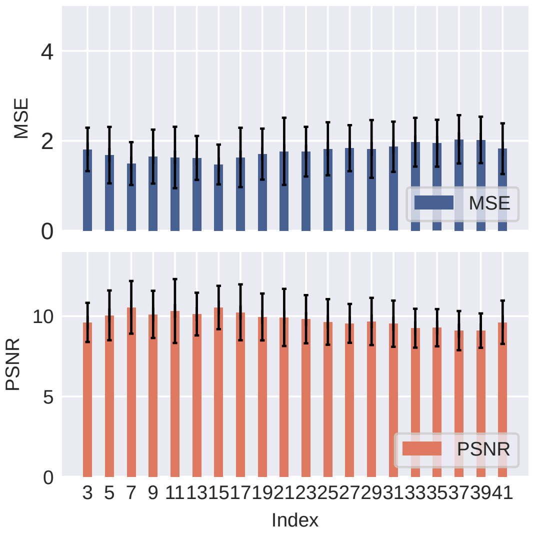
(d) BN layers in ResNet-18

Figure 4.2: The attack performance on single layers on CIFAR10.

of deep layers. **Pruning the gradients of the last few layers (usually the model head) proves to be an effective strategy against the attacks. Moreover, Conv layers contain more private information than BN layers.** In our following experiments, we set the split threshold κ to be the index of the model head in ConvNet (layer 17) and ResNet-18 (layer 41). This signifies that the local layer is only the last fully connected layer in these models. All preceding layers are backbone layers, categorized as global layers.



(a) ConvNet



(b) ResNet-18

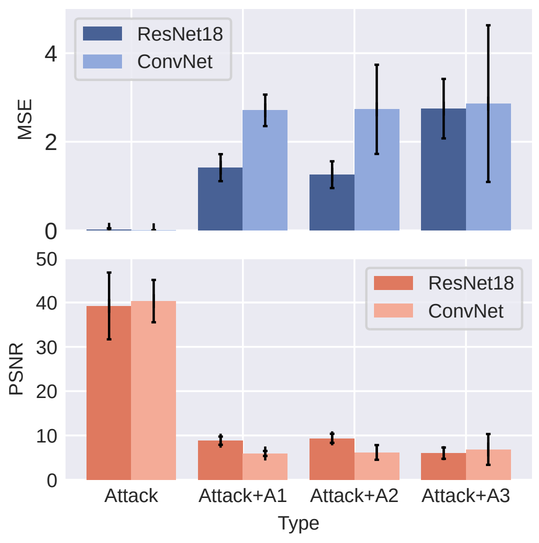
Figure 4.3: The attack performance on multiple layers on CIFAR10.

4.5.3 Privacy Preservation in Our Framework

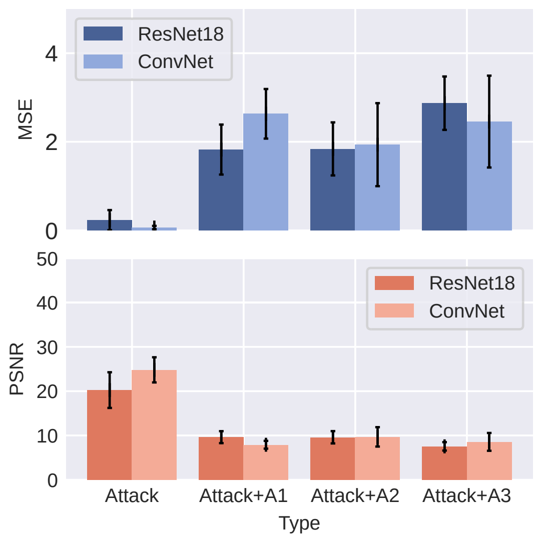
Previous experiments proved that the private information in data samples could be protected using our PPFed framework. In this subsection, we begin by evaluating our framework against IG attacks on multiple datasets. Then, we provide qualitative results from several state-of-the-art gradient inversion attacks with or without our defense mechanisms (i.e., pruning gradients to zero, randomizing gradients, and adding randomized noises to gradients). Finally, we compare our method with other defense mechanisms.

Figure 4.4 shows the original IG attack performance along with the performance with our three defense approaches on multiple datasets. The MSE and PSNR values are provided across all attack performances. We see a clear trend that all three defense approaches yield significant reductions in attack performance. These approaches can successfully defend against attacks on different models (ConvNet and ResNet-18) and on four different datasets.

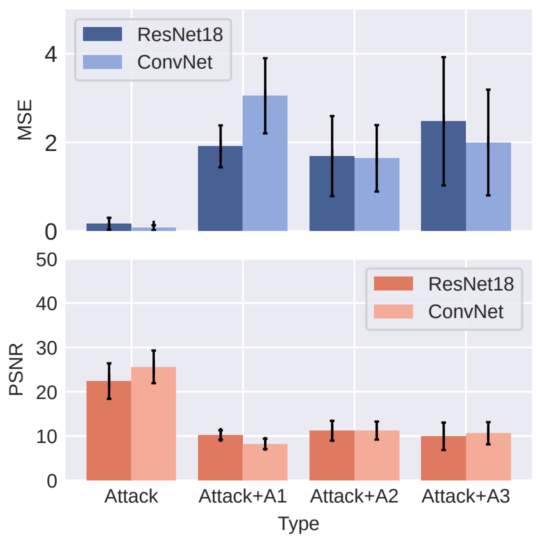
We further present the qualitative results from MNIST, CIFAR10, ImageNet1K, and Caltech256 in Figure 4.5. We select a random data sample from each of these datasets to show the results of the IG attack and the attack with our defense approaches. Reconstruction results across multiple training iterations are provided. Without any defense approaches, the attacks can produce recognizable data samples from 50 iterations on MNIST and 200 iterations on other datasets. However, when our defense approaches are employed, the attacks fail to reconstruct recognizable data samples throughout



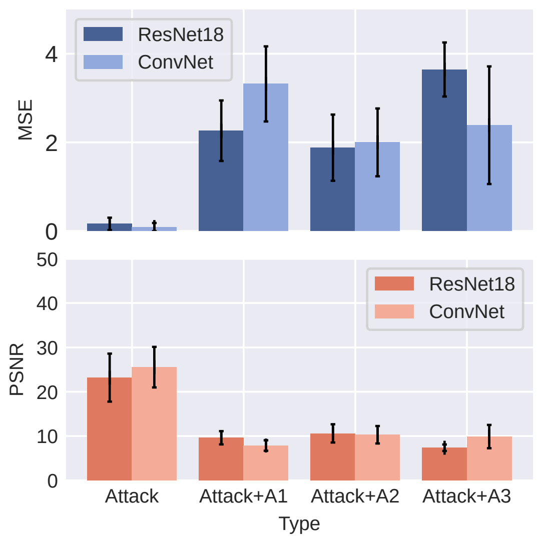
(a) MNIST



(b) CIFAR10



(c) ImageNet



(d) Caltech256

Figure 4.4: The IG attack performance with or without our defense on multiple datasets. A1: Pruning gradients to zero; A2: Randomizing gradients; A3: Adding randomized noises to gradients.

the entire training stages. All three of our defense approaches exhibit similar defense performance.

Figure 4.6 further highlights the effectiveness of our defense mechanism against the two state-of-the-art attack methods: IG and DLG. Several images are randomly selected from CIFAR10, and the model is ResNet-18. DLG attacks occasionally fail to produce a proper reconstruction result (the first image), and IG attacks can often reconstruct data samples effectively. When utilizing our defense mechanism, all the attacks have failed on

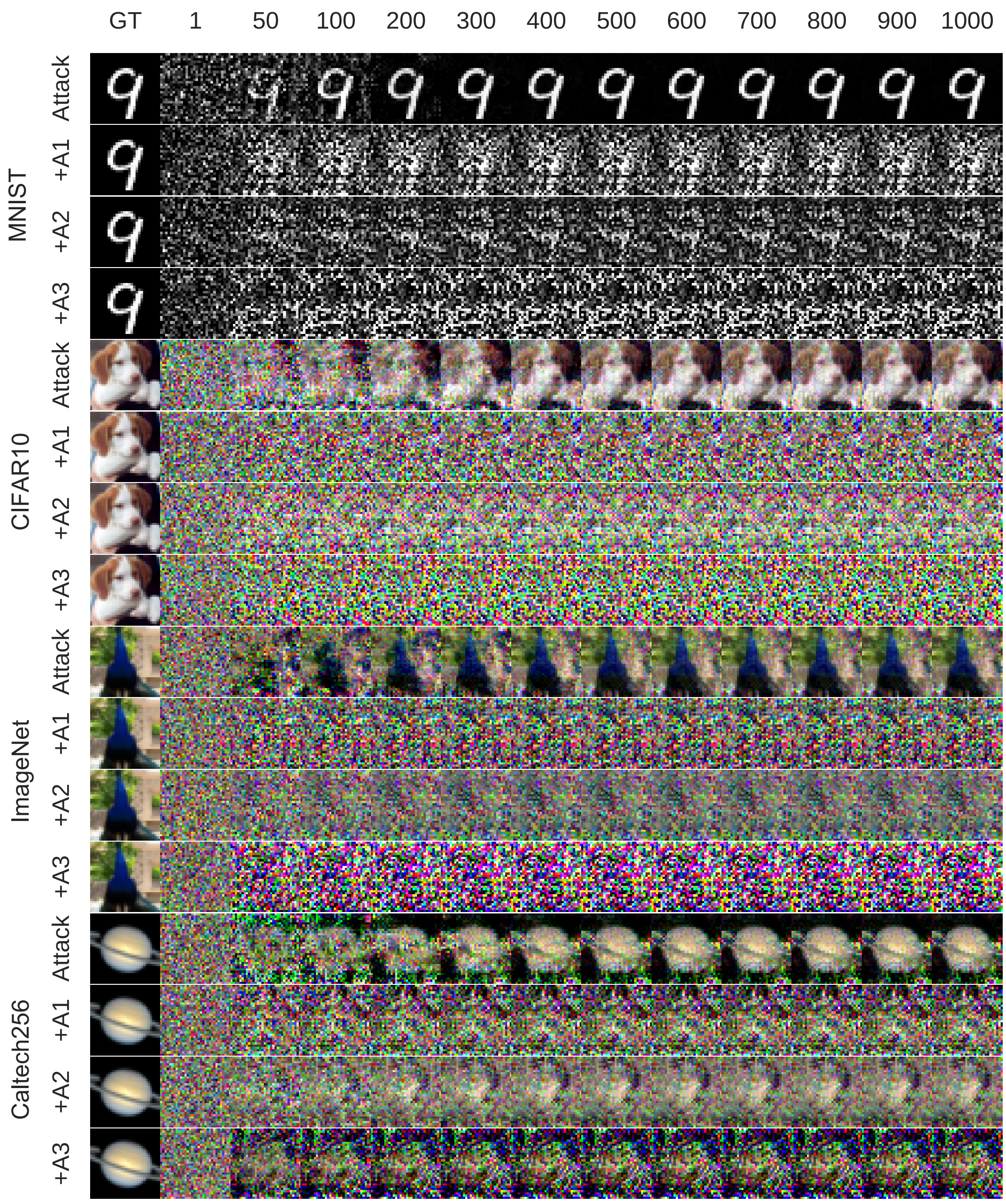


Figure 4.5: Qualitative results of the IG attack performance with or without our defense on multiple datasets.

reconstruction, receiving high MSE values (all above 0.50), making the reconstruction unrecognizable. These findings show the robustness of our defense mechanism against DLG and IG attacks, and the private information in data samples is well preserved.

In Figure 4.7, we compare our proposed method with four other defense mechanisms against IG attacks, using ConvNet on images in CIFAR10. Each X-axis in the subfigure is

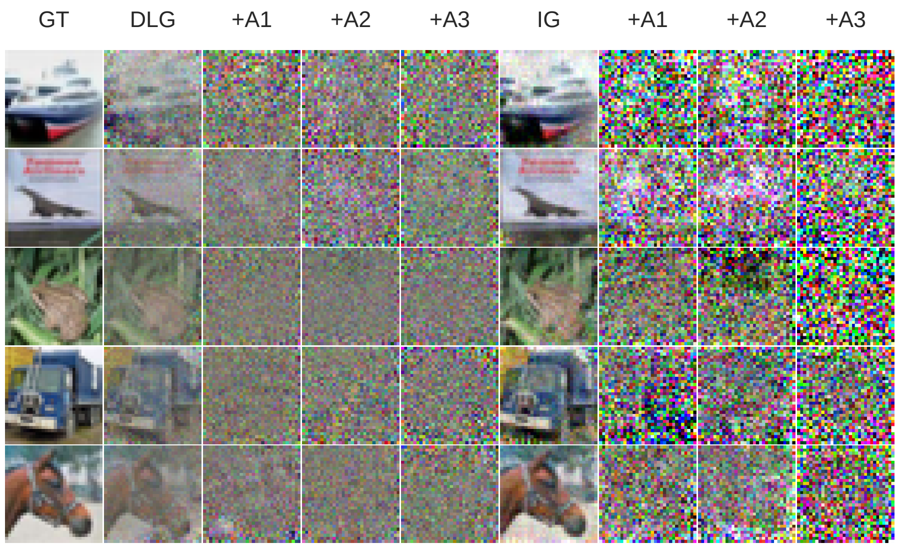
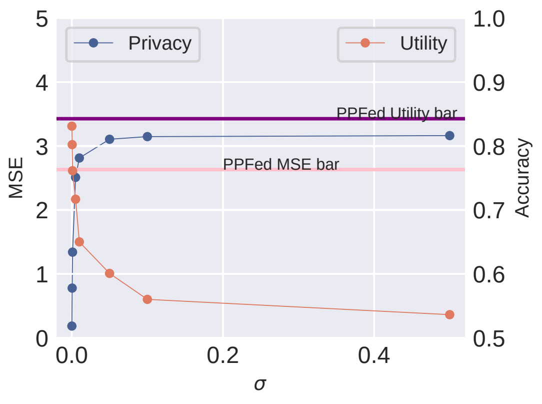


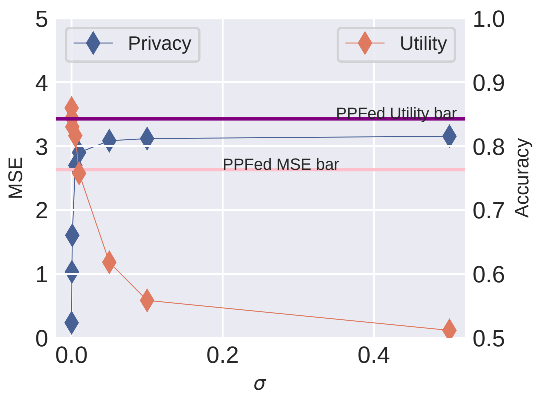
Figure 4.6: Qualitative results of the attacks with or without defenses.

its own hyperparameter, and its corresponding MSE and accuracy values are compared. We draw two bars in each subfigure to indicate the defense performance of our framework for comparison. The setting is pruning the gradients of local layers to zero as a defense approach (with an MSE score of 2.6, depicted in pink), and the federated learning part is one local epoch and five fine-tuning epochs (with an accuracy of 0.86, shown in purple). This setting causes the attacks to produce unrecognizable results of reconstruction samples, but eventually, each client can have a personalized model with high utility.

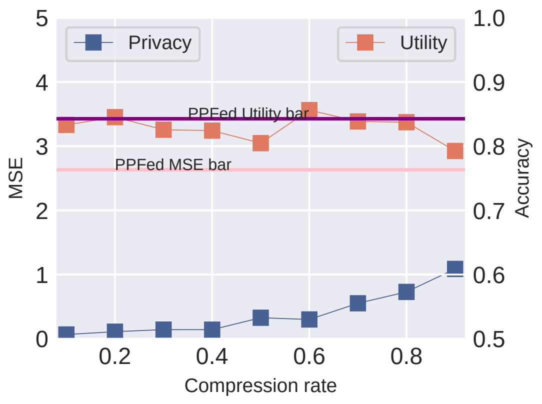
The utility performance of the four defense mechanisms is evaluated based on the defense mechanism added to FedAvg. All four subfigures in Figure 4.7 indicate that MSE values increase when their own hyperparameters rise. Differential privacy approaches are evaluated with Gaussian noises (DP-Gaussian) and Laplacian noises (DP-Laplacian). The σ in DP-Gaussian and DP-Laplacian indicates the magnitude of noise added to the model gradients. In DP-Laplacian, the σ is 0.0005 when MSE is above 1; in DP-Gaussian, it is 0.001. The main issue when applying differential privacy to the model training is that it reduces model utility significantly. Adding more differential privacy noises to the model can make it unusable. The compression rate in gradient compression measures how much percentage of gradients needs to be compressed. To receive an MSE value above 1, this method requires compressing over 90% of the gradients, making this mechanism unrealistic. The perturbation rate in Soteria calculates how much percentage of the representation in a single layer can be perturbed. When the MSE value is above 1, the perturbation rate is above 0.7. All these defense mechanisms show that changing the model gradients can lead to differences in reconstruction results. Simultaneously, the



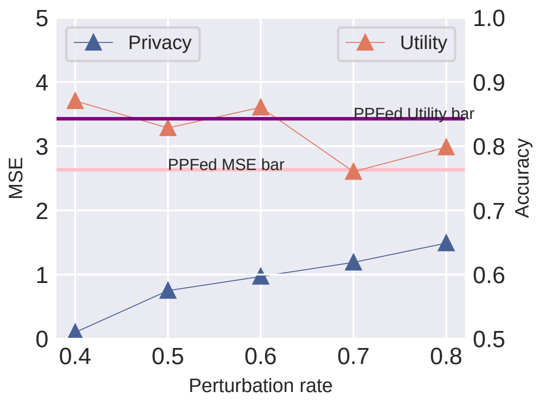
(a) DP-Gaussian



(b) DP-Laplacian



(c) Gradient Compression



(d) Soteria

Figure 4.7: The privacy and utility performance of other defenses compared to our performance bar.

Table 4.5: Accuracy of our method compared with prior research.

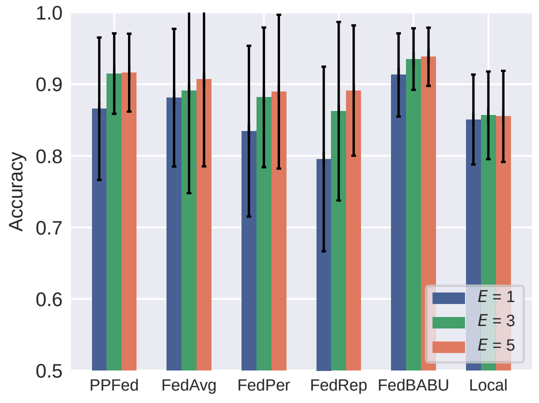
Setting	**PPFed (Ours)**	**FedAvg**	**FedPer**	**FedRep**	**FedBABU**	**Local**
CIFAR10, ConvNet	0.8657	0.8811	0.8343	0.7955	0.9129	0.8506
MNIST, ConvNet	0.9992	0.9983	0.9990	0.9985	0.9990	0.9953

federated learning model suffers varying degrees of accuracy loss.

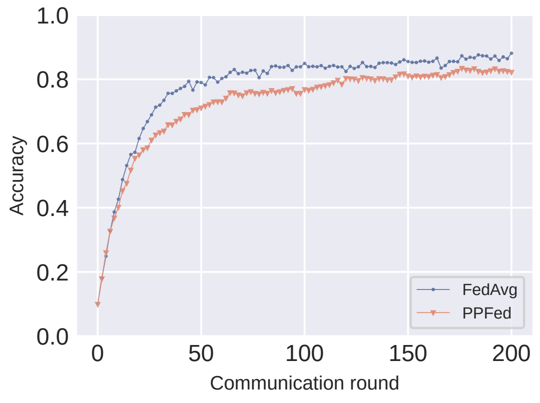
4.5.4 Personalization Performance

In this subsection, we compare the personalization performance of our PPFed framework with several other state-of-the-art methods. Then, we conduct several ablation studies on our framework.

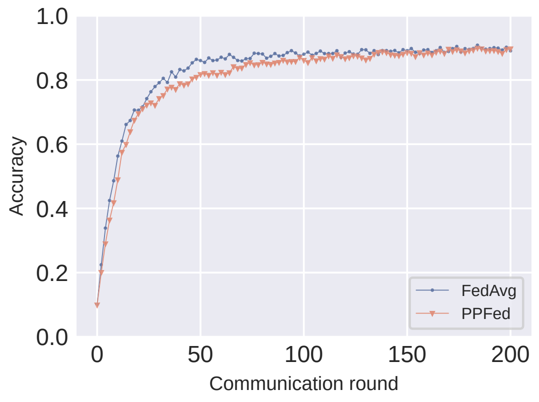
Table 4.5 compares our framework with other methods concerning personalization performance. These different frameworks possess differences in model structures: FedAvg



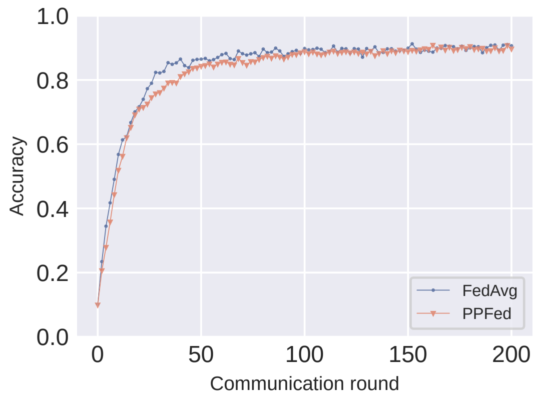
(a) Various local epoch performance



(b) Training when $E = 1$



(c) Training when $E = 3$

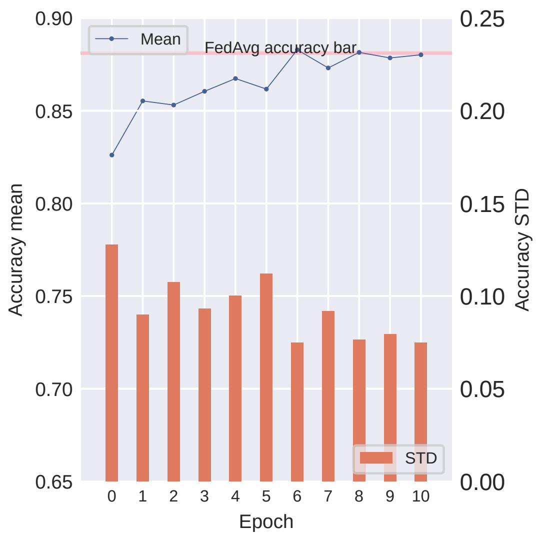


(d) Training when $E = 5$

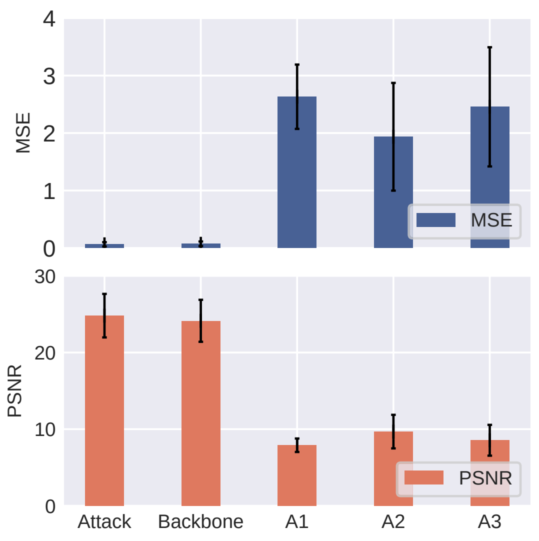
Figure 4.8: The personalization performance under multiple local epochs.

has a single global model; PPFed (Ours), FedPer, FedRep and FedBABU have both global models and local models; The "Local" framework only has local models for each client. We evaluate the personalization performance in two datasets with ConvNet. The results are obtained after 200 communication rounds, and the performance is assessed based on the average accuracy results from multiple runs. The results of all frameworks on MNIST achieve an accuracy exceeding 0.99. Hence, it would be difficult to tell the difference between these frameworks. The results on CIFAR10 show that our framework can match other frameworks.

Figure 4.8a compares the performance of different frameworks when the local epoch E is set to 1, 3 or 5. When E increases, the accuracy also rises in each framework, except for the "Local" framework. E seems to have little effect on the "Local" framework. We also observe that our PPFed framework can outperform FedAvg and have the second-highest accuracy when E is 5. The rest of the subfigures in Figure 4.8 give a comparison of FedAvg and PPFed in each communication round. When E is 1 (Figure 4.8b), the accuracy of



(a) The performance of PPFed with multiple fine-tuning epochs.



(b) The comparison of defense performance if only backbone gradients are sent to the server.

Figure 4.9: Ablation study on personalized federated learning.

PPFed is lower than that of FedAvg during the entire training process. However, as E is increased to 3 or 5 (Figure 4.8c and Figure 4.8d), the performance of PPFed matches that of FedAvg, and ultimately surpasses it in terms of accuracy. While FedAvg demonstrates a superior convergence rate with a setting of $E = 1$, PPFed effectively matches this rate of convergence when the value of E is adjusted to either 3 or 5. This adjustment in E enables PPFed to achieve comparable performance in terms of convergence rate and accuracy. These subfigures also illustrate that the accuracy performance of these frameworks becomes relatively stable after approximately 100 communication rounds.

Figure 4.9a shows the multiple fine-tuning results of PPFed compared with FedAvg. The average accuracy and the standard deviation are the result of 100 local clients. In this setting, each local client undergoes one local epoch during each communication round. The X-axis represents the number of epochs allocated for fine-tuning stages, with ConvNet as the chosen model. We can see that the accuracy increases when more epochs are allowed in fine-tuning stages. Eventually, the performance of PPFed matches that of FedAvg. Therefore, there are two ways for PPFed to match the performance of FedAvg: by increasing the local epochs (e.g., setting it to 5) or by extending the duration of fine-tuning epochs (i.e., exceeding 8 epochs). Furthermore, it is worth noting that the standard deviation decreases with an increase in fine-tuning epochs, indicating that more local clients experience an accuracy boost as the number of epochs rises.

We have demonstrated that our framework can match the utility performance of other

Table 4.6: Normalized time cost comparison across several defense and personalization methods.

FedAvg	**PPFed**	**DP-Gaussian**	**DP-Laplacian**	**Gradient compression**	**Soteria**
1	1.05	1.70	1.91	1.49	1.33
FedAvg	**PPFed**	**FedPer**	**FedRep**	**FedBABU**	**Local**
1	1.05	0.98	1.06	1.01	0.73

personalized federated learning frameworks. Meanwhile, our framework can achieve much better privacy performance thanks to the design of adding perturbations to local layers (three approaches: pruning gradients to zero, randomizing gradients, and adding randomized noises to gradients). While FedPer, FedRep and FedBABU only send the backbone layers (global layers) to the server during the training process, what would be the performance of the attack if only gradients of backbone layers are received by the server? Figure 4.9b presents the comparison result. The label with "Backbone" is the attack results with backbone-only gradients. It turns out that only sending gradients of backbone layers cannot mitigate the success of the attack. In such cases, adversaries can match the gradients and feed them into the gradient-matching process, allowing for successful data sample reconstruction. In contrast, our proposed approach, which incorporates perturbations in local layers, maintains a critical advantage. As we introduce perturbations to local layers, adversaries are unable to distinguish between original and modified gradients, making the attack ineffective and incapable of reconstructing data samples.

4.5.5 Complexity

We also evaluate the complexity (computational overhead) of our framework compared with other defense and personalization methods. This complexity is measured by the computational time required to complete a communication round between the server and the clients. The results are presented in Table 4.6. With 100 local clients and the local epoch set to 1, we set the performance of FedAvg as 1, and the performance of all the other frameworks is normalized to align with FedAvg.

In the first line, we see the complexity of PPFed and several other defense mechanisms. Our framework only costs slightly more than FedAvg while outperforming all other defense mechanisms. Notably, differential privacy-based mechanisms are the most computationally demanding, owing to the necessity of calculating and adding noise to the gradients of every layer in the model.

In the second line, the performance of several personalized federated learning frameworks is presented. It is observed that most of these frameworks have similar complexity levels. An exception is the "Local" framework, which demonstrates the lowest complexity. This is attributed to the absence of a global model, eliminating the need for server-client communication. In summary, our framework not only achieves a balance in complexity compared to other frameworks but also provides a substantial improvement in privacy.

4.5.6 Discussions

Our experiments evaluate the effectiveness of our framework in both privacy preservation and personalization in federated learning. Sections 4.5.2 and 4.5.3 demonstrate that perturbations on gradients and only parts of gradients (from local layers) can cause reconstruction errors to rise, thereby increasing ψ. These findings effectively defend against gradient inversion attacks, which are aligned with Propositions 1 and 2. Moreover, all three of our defense approaches can improve our privacy preservation performance. Section 4.5.4 show that optimizing local parameters in each local client achieves model personalization in federated learning, which is aligned with Proposition 3. Furthermore, we highlight that increasing the number of local epochs and fine-tuning epochs can enhance model personalization.

Privacy preservation and personalization are two separate challenges in federated learning. We demonstrate that these challenges can be effectively addressed through a unified framework by splitting the model into global and local layers. Our findings reveal that private information is contained in the local layers, and a personalized model can be trained by optimizing local layers on the client side. In essence, both privacy preservation and model personalization can be achieved through processing local layers. This suggests that privacy preservation and personalization, while initially seen as separate issues, are, in fact, two facets of a single challenge.

4.6 Conclusions and Future Work

In this chapter, we have established a unified federated learning framework, named PPFed, that can achieve both privacy preservation and personalization. By splitting the model into global and local layers, our framework aggregates the parameters of global layers in the server and trains the parameters of local layers in each client. We then add perturbations to local layers (i.e., pruning gradients to zero, randomizing gradients, and adding randomized noises to gradients) to ensure the privacy preservation of local data. We conclude that our framework can defend against gradient inversion attacks in federated learning when the adversary tries to reconstruct the clients' private information through model gradients, thereby protecting the privacy of the client data. Our framework also enables personalization in local clients and improves the utility of local models.

Unlike previous research, which often concentrated solely on either privacy or personalization, we have successfully found a unified solution that addresses both privacy preservation and personalization concerns from a model design perspective. Our experiments validate that our proposed PPFed framework can outperform other privacy preservation mechanisms and match the performance of other state-of-the-art personalization federated learning frameworks. Our future work includes evaluating more scenarios of our framework compared with prior research.

CHAPTER 5

HOW DOES A MODEL ARCHITECTURE IMPACT ITS PRIVACY?

In this chapter, we analyze privacy leakage from the perspective of model architectures. As a booming research area in the past decade, deep learning technologies have been driven by big data collected and processed on an unprecedented scale. However, privacy concerns arise due to the potential leakage of sensitive information from the training data. Recent research has revealed that deep learning models are vulnerable to various privacy attacks, including membership inference attacks, attribute inference attacks, and gradient inversion attacks. Notably, the efficacy of these attacks varies from model to model. In this chapter, we answer a fundamental question: *Does model architecture affect model privacy?* By investigating representative model architectures from convolutional neural networks (CNNs) to Transformers, we demonstrate that Transformers generally exhibit higher vulnerability to privacy attacks than CNNs. Additionally, we identify the micro design of activation layers, stem layers, and LN layers, as major factors contributing to the resilience of CNNs against privacy attacks, while the presence of attention modules is another main factor that exacerbates the privacy vulnerability of Transformers. Our discovery reveals valuable insights for deep learning models to defend against privacy attacks and inspires the research community to develop privacy-friendly model architectures. [1]

[1]The content of this chapter is based on [201]. Guangsheng Zhang was the first author of this paper.

5.1 Background

Deep learning has been gaining massive attention over the past several years. Training deep learning models requires collecting and processing user data, which raises significant privacy concerns. The data gathered during the training phase often contains sensitive information that malicious parties can access or retrieve. Various privacy attacks targeting deep learning models have demonstrated this vulnerability extensively. One prominent type of attack is membership inference, which focuses on determining whether a specific data sample belongs to the training data [141, 145]. Another attack is attribute inference, which aims to uncover implicit attributes learned by the model beyond the intended target attribute [120, 149]. Additionally, gradient inversion attacks pose a significant threat by attempting to reconstruct the information of the training data from the gradients of the model [46, 52]. These attacks empower adversaries to exploit deep learning models for extracting sensitive data.

Prior research has established that overfitting is one of the primary causes of privacy leakage in deep learning models [24, 65, 104, 145]. In general, overfitting occurs when models excessively learn specific details from the training data, which can lead to inadvertent privacy breaches. Surprisingly, we discover that even when models exhibit comparable levels of overfitting, the effectiveness of attacks varies across different models. This observation raises intriguing questions as to why certain deep learning models are more susceptible to privacy attacks than others, a puzzle that researchers have not fully comprehended. Consequently, we conjecture that other factors beyond overfitting might also contribute to the increased vulnerability of some deep learning models to privacy attacks. Though existing literature has explored model robustness and explainability [6, 133], the privacy leakage of the model architectures remains underexplored. Therefore, we are motivated to address this critical gap by answering the following question: *How does a model's architecture affect its privacy preservation capability?*

In this chapter, we approach this question by comprehensively analyzing different deep learning models under various state-of-the-art privacy attacks. Our investigation focuses on two widely adopted deep learning model architectures: convolutional neural networks (CNNs) and Transformers. CNN-based models have been dominant in computer vision, thanks to its sliding-window strategy, which extracts local information from images effectively. Transformers, initially introduced in natural language processing (NLP), have gained popularity in computer vision by capturing large receptive fields through attention mechanisms, resulting in comparable accuracy performance against CNNs. The tremendous achievements and wide usage of these two model architectures provide an excellent opportunity for us to make a comparative analysis regarding model privacy risks. Through our investigation, we make an intriguing discovery: *Transformers, in general, exhibit higher vulnerability to mainstream privacy attacks than CNNs.*

While Transformers and CNNs have different designs in many aspects, we investigate whether some key modules in the model architecture have a major impact on privacy risks. To this end, we evaluate the privacy leakage of several major modules in a Transformer architecture by sending only selected gradients to the gradient inversion attacks and discover that attention modules cause significant privacy leakage. Moreover, we start with a popular CNN-based model, ResNet-50 [62], and gradually morph the model to incorporate the key designs of Transformers. This leads us to the structure of ConvNeXt [110]. We evaluate the privacy leakage through this process and identify several key components that have a significant impact on privacy risks: (1) the design of the activation layers; (2) the design of stem layers; (3) the design of LN layers. We further conduct ablation studies to verify our discoveries and propose solutions to mitigate the privacy risks.

In summary, our contributions in this chapter are summarized as follows:

- For the first time, we investigate the impact of model architectures and micro designs on privacy risks.
- We evaluate the privacy vulnerabilities of two widely adopted model architectures, i.e., CNNs and Transformers, using three prominent privacy attack methods: (1) membership inference attacks, (2) attribute inference attacks, and (3) gradient inversion attacks. Our analysis reveals that Transformers exhibit higher vulnerabilities to these privacy attacks than CNNs.
- We identify three key factors: (1) the design of activation layers, (2) the design of stem layers, and (3) the design of LN layers, that significantly contribute to the enhanced resilience of CNNs in comparison to Transformers. We also discover that the presence of attention modules in Transformers could make them susceptible to privacy attacks.
- We propose solutions to mitigate the vulnerabilities of model architectures: modifying model components and adding perturbations as defense mechanisms.

5.2 Related Work

5.2.1 CNNs and Vision Transformers

Convolutional Neural Networks (CNNs) are a type of neural network that employs convolutional layers to extract features from input data. In contrast to fully connected networks, CNNs use convolutional kernels to connect small samples to neurons for feature extraction, reducing the number of model parameters and enabling the recognition of local features. Various techniques are employed to construct a CNN model, including padding, pooling, dilated convolution, group convolution, and more.

The concept of convolutional neural networks (CNNs) dates back to the 1980s [92]. However, the invention of AlexNet [90] makes CNNs the most prominent models in computer vision. Subsequent research improved the accuracy and efficiency of models [147, 156]. ResNet [62] addressed the challenge of training deep networks using skip connections. Other notable networks consist of Inception [157], MobileNet [71], ResNeXt [183], EfficientNet [158], RegNet [132], ConvNeXt [110].

Vision Transformers, originating from natural language processing, divide the input image into multiple patches, forming a one-dimensional sequence of token embeddings. Their exceptional performance can be attributed to the multi-head self-attention modules [167]. The attention mechanism has significantly contributed to the advancement of natural language processing [12, 34, 186], subsequently leading to the introduction of Transformers in the field of computer vision as Vision Transformers (ViT) [36]. Research has shown that ViTs can surpass CNNs in various downstream tasks [36, 153]. Later Transfomer models have focused on numerous improvements of ViTs, such as Tokens-to-Token ViT [196], Swin Transformers [109], DeiT [162], MViT [97], DaViT [35].

Numerous studies have compared CNNs and Transformers from the perspectives of robustness [6, 131, 171] and explainability [133]. However, our research diverges from previous works by concentrating on the privacy leakage inherent in both CNNs and Transformers.

5.2.2 Privacy Attacks on Deep Learning Models

Here, we introduce related work on privacy attacks **that is specifically relevant to this chapter**. A primary concern in deep learning privacy is that the model may reveal sensitive information from the training dataset. An adversary can exploit various approaches to compromise privacy, including predicting whether a particular sample is in the model's training dataset via membership inference attacks, or disclosing the implicit attributes of data samples via attribute inference attacks, or even recovering private data samples utilized in training a neural network through gradient inversion attacks.

Membership inference attacks were initially introduced in [145], where an attack model was employed to distinguish member samples from non-member samples in the training data. To execute these attacks, shadow models would mimic the behavior of victim models [141, 145]. Prediction results from victim models were gathered for attack model training. Usually, the confidence scores or losses were utilized [145], but more recent work (label-only attacks) applied prediction labels to launch attacks successfully [28, 100]. The attacks could also be executed by designing a metric with a threshold by querying the shadow model [150]. Other researchers relaxed the attack assumptions and improved the attacks, including discussion on white-box/black-box access for the attacks [139], providing more metrics (i.e. ROC curves and the true positive rate at a low false positive rate) to measure the attack performance more accurately [15, 81, 113, 173, 189]. We select [15, 141, 145] as our baseline methods.

Attribute inference attacks, another significant category of privacy attack methods, attempt to reveal a specific sensitive attribute of a data sample by analyzing the posteriors of the victim model trained by the victim dataset. Some early research launched the attacks by generating input samples with different sensitive attributes and observed the victim model output [47, 191]. However, these methods could only work in structured data. Later research improved the attacks with victim model representations [120, 149]. They also claimed that the overlearning feature of deep learning models caused the execution of the attacks [149]. As we aim to infer attributes from visual data, we select [120, 149] as baseline methods.

Gradient inversion attacks primarily aim to reconstruct training samples at the local clients in federated learning. Using the publicly shared gradients in the server, adversaries can execute the attacks by reconstructing the training samples using gradient matching. DLG [219] and its variant, iDLG [214], were the early attacks to employ an optimization-based technique to reconstruct the training samples. Later research like Inverting Gradients [52] and GradInversion [192] improved the attack performance by incorporating regularizations into the optimization process. The use of Generative Adversarial Networks (GANs) in some gradient inversion attack methods [102] can have a significant impact on reconstructed results, making it difficult to isolate the influence of other factors on privacy leakage. Therefore, we use conventional gradient inversion attack methods [52] that do not involve the use of GANs.

There have been several evaluations and reviews of these privacy attacks against deep learning models [65, 73, 104, 108, 150, 204, 209]. However, we aim to evaluate the model architectures leveraging these privacy attacks. To sum up, we utilize conventional privacy attacks [15, 52, 120, 141, 145, 149] as the baseline attacks in our analysis, for these attack methods have inspired many follow-up research works, and they are suitable for evaluation on various models and datasets.

5.3 Methodology of Evaluating the Impact of the Model Architecture on Privacy

In this section, we present our approach to assessing the impact of model architectures on privacy leakage. In order to organize our study in a thorough and logical manner, We aim to answer the following research questions sequentially:

- **RQ1**: How to analyze the privacy leakage in model architectures?
- **RQ2**: What model architectures of CNNs and Transformers should we choose to evaluate these attacks?
- **RQ3**: What performance aspects should we focus on when evaluating the privacy attacks on model architectures?
- **RQ4**: How should we investigate what designs in model architectures contribute to privacy leakage?

In this work, we focus on classifier or feature representation models such as CNNs and Transformers, which are subject to the investigated privacy attacks. A new line of generative AI models, such as generative adversarial networks (GANs) and diffusion models, are vulnerable to different privacy attacks and thus out of the scope of this chapter. We believe our evaluation methodology could shed light on the effect of model privacy from the perspective of model architectures.

5.3.1 Privacy Threat Models

To answer the first research question (**RQ1**), we choose three prominent privacy attack methods: membership inference attacks, attribute inference attacks, and gradient inversion attacks.

5.3.1.1 Membership Inference Attacks

Network-Based Attacks. Initiating a network-based membership inference attack [141, 145] requires three models: the victim model $\mathscr{V}$ (the target), the shadow model $\mathscr{S}$ (the model to mimic the behavior of the victim model), and the attack model $\mathscr{A}$ (the classifier to give results whether the sample belongs to the member or non-member data). The following paragraphs provide explanations of how the attacks work.

The first step is the attack preparation. Since the adversary has only black-box access to the victim model $\mathscr{V}$, they can only query the model and record prediction results. To launch a membership inference attack, the adversary needs to create a shadow model $\mathscr{S}$, which behaves similarly to the victim model $\mathscr{V}$. This involves collecting a shadow dataset $\mathscr{D}_S$, usually from the same data distribution as the victim dataset $\mathscr{D}_V$. The shadow dataset $\mathscr{D}_S$ is then divided into two subsets: $\mathscr{D}_S^{train}$ for training and $\mathscr{D}_S^{test}$ for testing.

Once the preparation is complete, the adversary trains the attack model. The shadow model $\mathscr{S}$ and shadow dataset $\mathscr{D}_S$ are used to train the attack model $\mathscr{A}$. Each prediction result of a data sample from the shadow dataset $\mathscr{D}_S$ is a vector of confidence scores for each class, which is concatenated with a binary label indicating whether the prediction is correct or not. The resulting vector, denoted as $\mathscr{P}_S^i$, is collected for all n samples, forming the input set $\mathscr{P}_S = \{\mathscr{P}_S^i, i = 1, ..., n\}$ for the attack model $\mathscr{A}$. Since $\mathscr{A}$ is a binary classifier, a three-layer MLP (multi-layer perceptron) model is employed to train it.

At last, the adversary launches the attack model inference. The adversary queries the victim model $\mathscr{V}$ with the victim dataset $\mathscr{D}_V$ and records the prediction results, which are used as the input for the attack model $\mathscr{A}$. The attack model then predicts whether a data sample is a member or non-member data sample.

Likelihood-Based Attacks. The Likelihood Ratio Attack (LiRA) [15] is a state-of-the-art attack method that employs both model posteriors and their likelihoods based on shadow models. In contrast to attacks relying on a single shadow model, LiRA requires the adversary to train multiple shadow models $\mathscr{S} = \{S_1, ..., S_n\}$. This ensures that a target sample (from the victim dataset $\mathscr{D}_V$) is included in half of the models $\mathscr{S}$ and excluded from the other half. The adversary then queries the shadow models with the target sample and calculates the logits for each model. Using these logits, the adversary calculates the probability density function to determine the likelihood ratio of the target sample, which corresponds to its membership status.

There are other kinds of membership inference attacks, including metric-based attacks and label-only attacks [28, 100, 150]. Instead of using a neural network to be the attack model, metric-based attacks [150] launch the attacks using a certain metric and threshold to separate member data from non-member data. Label-only attacks [28, 100] relax the assumptions of the threat model leveraging only prediction labels as the input of the attack model. Our study focuses on two types of membership inference attacks: network-based and likelihood-based attacks. We chose these two types of attacks because the network-based attack is commonly used as a baseline in many research papers, making it a conventional attack to consider. Additionally, the likelihood-based attack is a more recent state-of-the-art attack that has demonstrated high effectiveness, making it an important attack to evaluate as well. By considering these two types of attacks, we can effectively represent the performance of membership inference attacks against various victim models and gain insights into potential privacy risks associated with different machine learning models.

5.3.1.2 Attribute Inference Attacks

The goal of attribute inference attacks [120, 149] is to extract sensitive attributes from a victim model, which may inadvertently reveal information about the training data. For instance, suppose the victim model is trained to classify whether a person has a beard or not. In that case, an adversary may infer the person's race based on the model's learned representation.

At the attack preparation stage, the victim model $\mathscr{V}$ is trained by the victim dataset $\mathscr{D}_V$ with two subsets $\mathscr{D}_V^{train}$ and $\mathscr{D}_V^{test}$ for the training and testing.

The second step is also the attack model training. To train the attack model $\mathscr{A}$, the adversary uses an auxiliary dataset $\mathscr{D}_A^{train}$, which includes pairs of the representation h and the attribute a, i.e., $(h, a) \in \mathscr{D}_A$.

At last, the adversary launches the attack. The adversary takes a data sample's representation h as the input and uses the attack model $\mathscr{A}$ to infer the attribute result.

5.3.1.3 Gradient Inversion Attacks

Launching the gradient inversion attack [52, 214, 219] involves solving an optimization problem, which aims to minimize the difference between the calculated model gradients and the original model gradients. The optimization process continues for a certain number of iterations, after which the input data sample can be reconstructed.

The adversary operates within a federated learning scenario. In the attack preparation stage, the adversary operates from the central server, aggregating model gradients to create a centralized model. Since the adversary has access to the communication channels used during the federated learning process, they can retrieve the model gradients and prepare to extract sensitive information from the training samples. This allows the adversary to launch attacks against the federated learning system.

In the step of gradient reconstruction, the aggregated model gradients are denoted as $\nabla_\theta \mathscr{L}_\theta(x, y)$, where θ is the model parameters, x and y are the original input image and its ground truth in a local client, and $\mathscr{L}$ represents the cost function for the model. To initiate the reconstruction process, the adversary generates a dummy image x^*. The adversary tries to minimize this cost function: $\arg\min_x ||\nabla_\theta \mathscr{L}_\theta(x, y) - \nabla_\theta \mathscr{L}_\theta(x^*, y)||^2$. The dummy image x^* is reconstructed to resemble x closely.

5.3.2 CNNs vs Transformers

To answer the second research question (**RQ2**), we investigate the privacy of two mainstream architectures: CNNs and Transformers. We carefully select several popular CNNs and Transformers for the attacks to analyze the privacy leakage.

For CNNs, we choose ResNets [62] as baseline models, which are known for incorporating residual blocks and have become widely used in various computer vision tasks. We specifically select ResNet-50 (23.52 million parameters) and ResNet-101 (42.51 million parameters) to represent CNN architectures in our analysis. Regarding Transformers, we focus on Swin Transformers [109], which have gained attention for their innovative design incorporating attention modules and shifted window mechanisms. We analyze Swin-T (27.51 million parameters) and Swin-S (48.80 million parameters) as representatives of Transformer architectures.

To ensure fair comparisons, we organize the evaluation of the four models based on their parameter sizes, grouping models with similar parameter sizes together. This approach allows us to compare models that exhibit comparable task performances while considering their architectural differences. Specifically, we compare ResNet-50 with Swin-T, as they have similar parameter sizes, and we compare ResNet-101 with Swin-S for the same reason.

Apart from model architectures, training Transformers requires a modernized training procedure compared to training traditional CNNs [109, 167]. To ensure fair comparisons, we employ the same training recipe for both CNNs and Transformers in each comparison.

5.4 Evaluation on Privacy Attacks with CNNs and Transformers

This section addresses the third research question (**RQ3**) and comprehensively analyzes the experimental settings and results. We aim to compare the attack performance under various metrics to gain a deeper understanding of the findings. We also focus on the performance differences and overfitting differences between CNNs and Transformers.

5.4.1 Settings

Datasets. Our experiments evaluate the privacy leakage under these four datasets.

- **CIFAR10 [89]** consists of 60,000 color images with dimensions of 32×32 pixels. It is organized into ten classes, with each class containing 6,000 images. The dataset covers a wide range of general object categories.
- **CIFAR100 [89]** is similar to CIFAR10 but with 100 classes. Each class has 600 images and represents a specific general object category.
- **ImageNet1K [32]** is a widely used dataset in computer vision containing over 1 million labeled images, covering 1,000 different classes. The dataset encompasses a diverse range of objects and scenes.
- **CelebA [111]** contains over 200,000 face images, each with 40 binary attributes.

In membership inference attacks, we use two datasets, CIFAR10 and CIFAR100, for their popularity in prior attack research [15, 108, 141]. For network-based attacks, each dataset is evenly split into four subsets for training and testing both the victim and shadow models to ensure a fair evaluation [141]. For likelihood-based attacks, we follow the training settings in [15]. This involves dividing the training data into multiple subsets, ensuring that a specific training sample is present in only half of these subsets.

For attribute inference attacks, we utilize the CelebA dataset, which provides rich attribute labeling. This allows us to identify hidden attributes accurately. In our experiments, we focus on inferring the race attribute while using the gender attribute as the classification goal for the victim model. We randomly select 20,000 images from CelebA and evenly split them into four subsets for training and testing both the victim model and the attribute inference attack model.

In gradient inversion attacks, we employ the CIFAR10 and ImageNet1K datasets for low/high-resolution reconstruction. These attacks are conducted in a federated learning scenario to reconstruct the training batch. We randomly select a subset of images from these datasets to evaluate the attacks, ensuring a representative assessment of the attack performance.

In a nutshell, all these datasets are benchmark datasets for evaluating privacy attacks [15, 52, 65, 108, 192]. These datasets cover a wide range of objects with different numbers of classes, making them sufficient for our evaluation.

Victim Models. To ensure fair comparisons, we evaluate the performances of CNNs and Transformers with similar parameter sizes. For all attacks, we select two groups of models based on their parameter sizes. These groups include ResNet-50 and ResNet-101 as CNN-based models and Swin-T and Swin-S as Transformer-based models. We train these models to reach more than 0.99 training accuracy and output testing accuracy (i.e., task accuracy) results in the experiments.

Attack Models. For membership inference attacks, we employ a three-layer MLP model to infer membership information. Attribute inference attacks utilize a two-layer MLP model to uncover learned representations and potentially infer sensitive information. In gradient inversion attacks, we optimize input and generate gradients to reconstruct the original data, revealing private information.

Evaluation Metrics. We evaluate the performance of different attacks using specific metrics. For membership inference attacks, we consider attack accuracy (↑), ROC curve (↑), AUC (↑), and TPR at low FPR (↑). For attribute inference attacks, we assess effectiveness using attack accuracy (↑) and macro-F1 score (↑). Regarding gradient inversion attacks, we use multiple metrics to evaluate the quality of reconstruction results. These metrics include mean square error (MSE ↓), peak signal-to-noise ratio (PSNR ↑), learned perceptual image patch similarity (LPIPS ↓) [208], and structural similarity (SSIM ↑) [172]. Note that "↑" means the higher metric corresponds to the higher attack performance, while ↓ means the lower metric leads to the higher attack performance.

Training Settings for Privacy Attacks. Table 5.1 illustrates the training configurations for membership inference attacks, attribute inference attacks and gradient inversion attacks.

5.4.2 Evaluation on Membership Inference Attacks

The results presented in Table 5.2 offer the performance of network-based membership inference attacks on CIFAR10 and CIFAR100 datasets. The task accuracy scores of the victim models on CIFAR10, which are approximately 0.82 for both CNNs and Transformers, indicate that these models exhibit competitive task performance with similar

Table 5.1: Training recipes for privacy attacks.

Config	Param	Config	Param
for all models of membership inference			
optimizer	AdamW	training epochs	300
learning rate	0.001	batch size	256
weight decay	0.05	mixup	0.8
optimizer momentum	β_1, β_2 = 0.9, 0.999	cutmix	1.0
learning rate schedule	cosine annealing	label smoothing	None
for all models of attribute inference			
optimizer	SGD	batch size	256
learning rate	0.01	training epochs	100
weight decay	0.0005	random augmentation	None
optimizer momentum	0.9		
for gradient inversion			
cost function	similarity	total iteration	3000
optimizer	Adam	total variance	0.0001
learning rate	0.1		

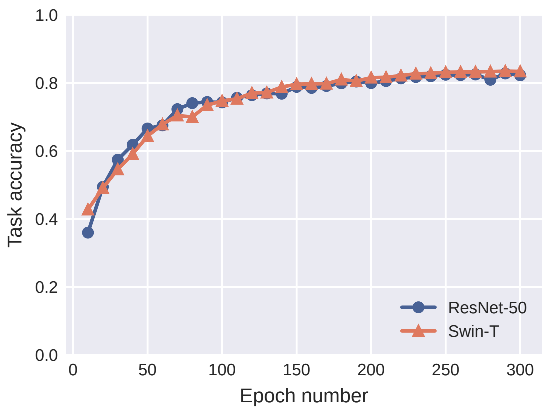
Table 5.2: Results for network-based membership inference attacks.

	CIFAR10		CIFAR100	
	Task acc ↑	Attack acc ↑	Task acc ↑	Attack acc ↑
ResNet-50	0.8220 ± 0.0023	0.6385 ± 0.0078	0.5288 ± 0.0083	0.8735 ± 0.0029
Swin-T	0.8335 ± 0.0042	0.6904 ± 0.0052	0.5632 ± 0.0056	0.9340 ± 0.0030
ResNet-101	0.8301 ± 0.0037	0.6317 ± 0.0063	0.5313 ± 0.0074	0.8607 ± 0.0034
Swin-S	0.8258 ± 0.0039	0.6405 ± 0.0075	0.5665 ± 0.0059	0.9357 ± 0.0039

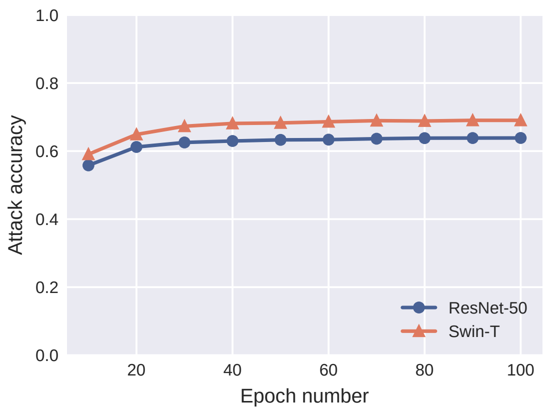
overfitting levels. Notably, the attack accuracy on CIFAR10 reveals that Transformers exhibit more privacy leakage within each group than CNN models. Similar findings are observed on CIFAR100, suggesting that Transformers consistently exhibit higher vulnerability to membership inference attacks compared to CNN models.

It is worth mentioning that the dataset used in this study is divided into four equally sized subsets for training and testing the victim and attack models. Consequently, the task accuracy of the victim models on CIFAR10 and CIFAR100 might be lower than expected in a standard CIFAR10 or CIFAR100 classification task. This does not affect the performance of the attacks, and this phenomenon has been acknowledged in prior research [15, 65, 86, 143]. This also applies to later experimental results.

Figure 5.1 illustrates the task accuracy and attack accuracy of ResNet-50 and Swin-T across multiple epochs on CIFAR10. The task accuracy of ResNet-50 starts relatively low but exhibits a rapid increase over time. Eventually, both ResNet-50 and Swin-T can achieve task accuracy scores of approximately 0.82. Regarding attack accuracy, the plot demonstrates that the attack on Swin-T consistently outperforms the attack on



(a) Performance of victim models on CIFAR10



(b) Performance of privacy attack on CIFAR10

Figure 5.1: The performance of membership inference attacks against ResNet-50 and Swin-T on CIFAR10 under different numbers of epochs.

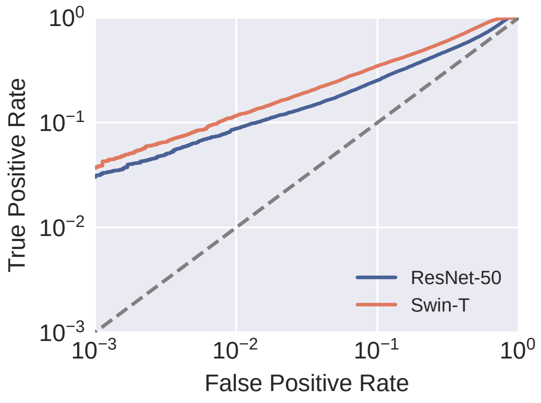
Table 5.3: Results of likelihood-based membership inference attacks.

		Task acc ↑	AUC ↑	TPR@0.1%FPR ↑	Attack acc ↑
CIFAR10	ResNet-50	0.8716 ± 0.0035	0.6446 ± 0.0276	3.15% ± 0.25%	0.6009 ± 0.0163
	Swin-T	0.8630 ± 0.0017	0.7384 ± 0.0029	3.68% ± 0.33%	0.6553 ±0.0027
	ResNet-101	0.8708 ± 0.0043	0.6671 ± 0.0107	3.33% ± 0.41%	0.6090 ± 0.0079
	Swin-S	0.8636 ± 0.0036	0.7392 ± 0.0054	3.75% ± 0.37%	0.6576 ± 0.0045
CIFAR100	ResNet-50	0.5632 ± 0.0032	0.9431 ± 0.0005	23.75% ± 2.09%	0.8524 ± 0.0009
	Swin-T	0.6001 ± 0.0033	0.9756 ± 0.0003	28.52% ± 1.54%	0.9112 ± 0.0010
	ResNet-101	0.5654 ± 0.0042	0.9379 ± 0.0021	21.25% ± 1.55%	0.8484 ± 0.0036
	Swin-S	0.5900 ± 0.0029	0.9639 ± 0.0006	31.02% ± 1.74%	0.8978 ± 0.0014

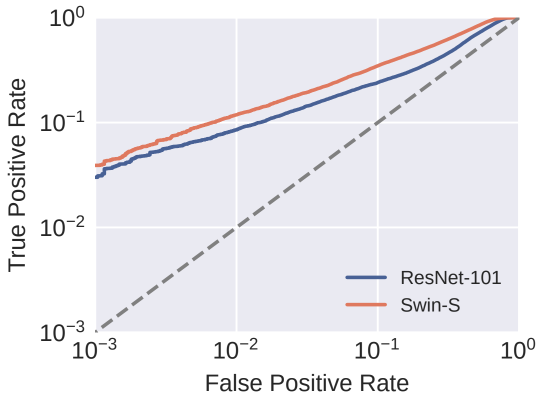
ResNet-50.

Table 5.3 displays the outcomes of likelihood-based membership inference attacks on CIFAR10 and CIFAR100. The table includes task accuracy and attack performance, the same as the evaluation for network-based attacks. To assess attack performance more comprehensively, we adopt the methodology proposed in [15] and incorporate additional metrics such as AUC, TPR@0.1%FPR, besides attack accuracy. The victim models achieve task accuracy scores of approximately 0.86 on CIFAR10 and 0.56 on CIFAR100. Consistently, the attack performance metrics highlight that Transformers are more vulnerable to membership inference attacks compared to CNNs in terms of any attack metric. We also present the attack performance through ROC curves in Figure 5.2. These curves demonstrate that Transformers yield higher ROC curves, signifying their better attack performance.

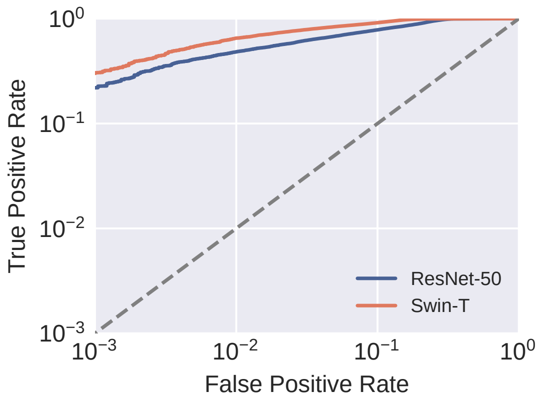
The privacy leakage varies on different models and datasets. Similar to [141, 145],



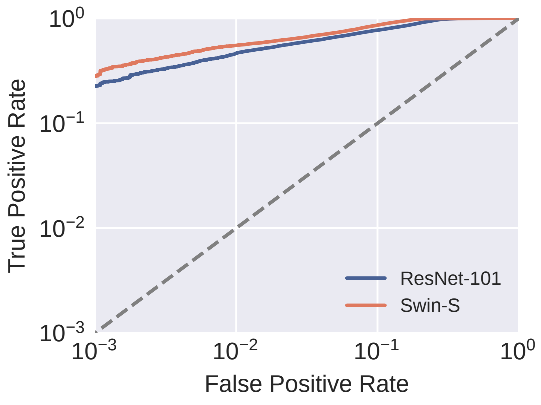
(a) ResNet-50 vs Swin-T on CIFAR10



(b) ResNet-101 vs Swin-S on CIFAR10



(c) ResNet-50 vs Swin-T on CIFAR100



(d) ResNet-101 vs Swin-S on CIFAR100

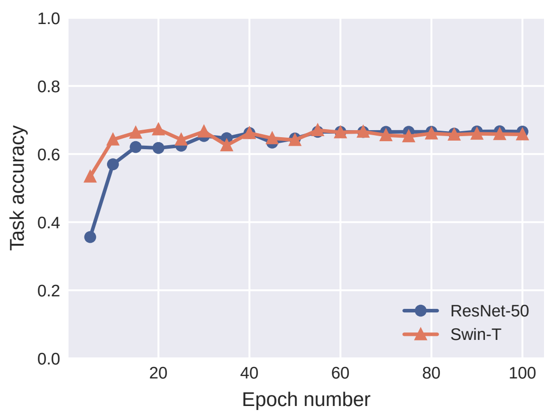
Figure 5.2: ROC curves for membership inference attacks on CIFAR10 and CIFAR100. Comparisons between CNNs and Transformers.

we analyze the overfitting levels of victims models in Figure 5.4a. The overfitting level indicates the accuracy difference of a model between its training and inference. Figure 5.4a illustrates the results of Transformers and CNNs in CIFAR10 and CIFAR100. We conclude that a more overfitted model comes with higher membership inference attack accuracy. More importantly, at the same overfitting level, Transformers always suffer from higher attack accuracy.

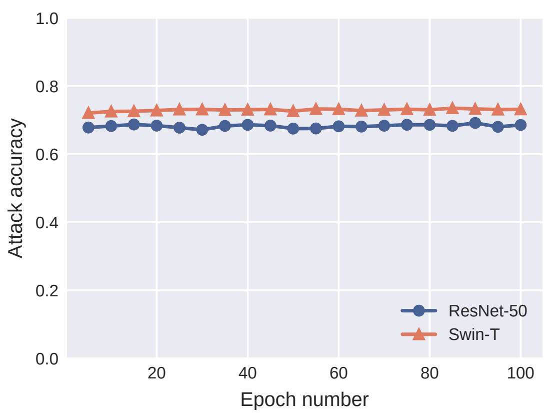
In conclusion, the results strongly suggest that Transformers are more susceptible to network-based and likelihood-based membership inference attacks compared to CNNs. Transformers consistently demonstrate higher vulnerability to membership inference attacks across various metrics.

Table 5.4: Results of attribute inference attacks on CelebA.

	Task acc ↑	Attack acc ↑	Macro F1 ↑
ResNet-50	0.6666 ± 0.0020	0.6854 ± 0.0015	0.3753 ± 0.0012
Swin-T	0.6587 ± 0.0023	0.7312 ± 0.0014	0.5530 ± 0.0019
ResNet-101	0.6431 ± 0.0029	0.6291 ± 0.0023	0.4262 ± 0.0009
Swin-S	0.6569 ± 0.0024	0.7369 ± 0.0036	0.5536 ± 0.0015



(a) Performance of victim models on CelebA



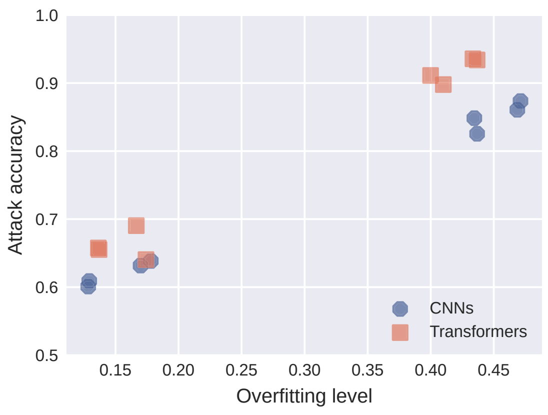
(b) Performance of privacy attack on CelebA

Figure 5.3: The performance of attribute inference attacks against ResNet-50 and Swin-T on CelebA under different numbers of epochs.

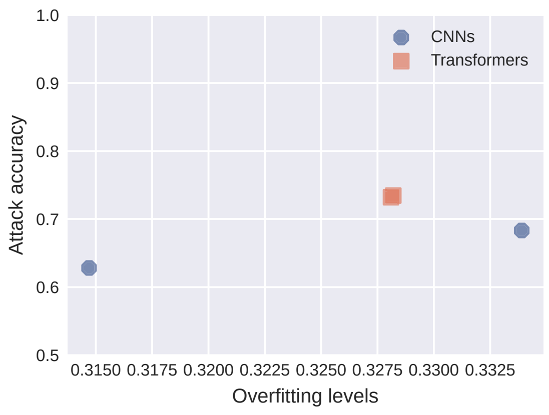
5.4.3 Evaluation on Attribute Inference Attacks

Table 5.4 presents the results of attribute inference attacks on CelebA. Similarly to membership inference attacks, we categorize CNN and Transformer models into two groups based on their parameter sizes. The table shows that within each group, CNNs and Transformers achieve similar task accuracy scores, approximately 0.65. However, when considering the attack accuracy and Macro F1 score, Transformers consistently outperform CNNs. The results from attribute inference attacks align with our previous findings from membership inference attacks, emphasizing the increased vulnerability of Transformer models to privacy attacks.

Figure 5.3 presents the performance of task accuracy and attack accuracy in attribute inference attacks using ResNet-50 and Swin-T on CelebA over 100 epochs. Figure 5.3a shows that although ResNet-50 and Swin-T models start with different task accuracy scores, both models gradually converge to similar task accuracy scores of around 0.65. However, when we examine the attack accuracy in Figure 5.3b, the attack accuracy on Swin-T consistently outperforms that on ResNet-50 throughout the 100 epochs. This reveals that Transformers like Swin-T are more vulnerable to attribute inference attacks



(a) Membership inference



(b) Attribute inference

Figure 5.4: The performance of privacy attacks against both CNNs and Transformers with various models and datasets under different overfitting levels.

Table 5.5: The results of gradient inversion attacks on CNNs and Transformers on CIFAR10.

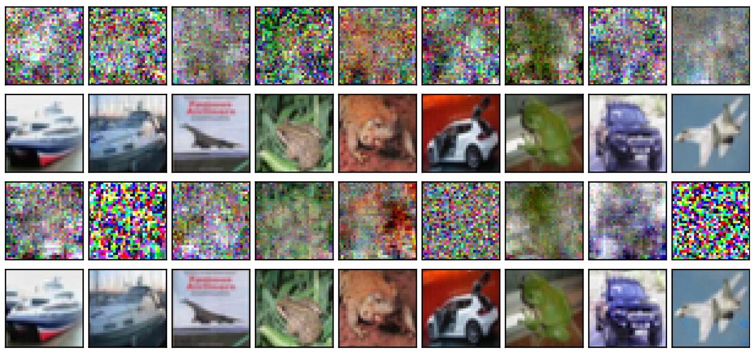
	MSE ↓	PSNR ↑	LPIPS ↓	SSIM ↑
ResNet-50	1.3308 ± 0.6507	11.30 ± 2.24	0.1143 ± 0.0403	0.0946 ± 0.0989
Swin-T	0.0069 ± 0.0071	36.24 ± 5.21	0.0012 ± 0.0016	0.9892 ± 0.0118
ResNet-101	1.2557 ± 0.6829	11.58 ± 2.16	0.1461 ± 0.1012	0.0784 ± 0.0675
Swin-S	0.0063 ± 0.0083	37.85 ± 6.15	0.0016 ± 0.0028	0.9878 ± 0.0128

than ResNet-50 from the start of the attack training to the end.

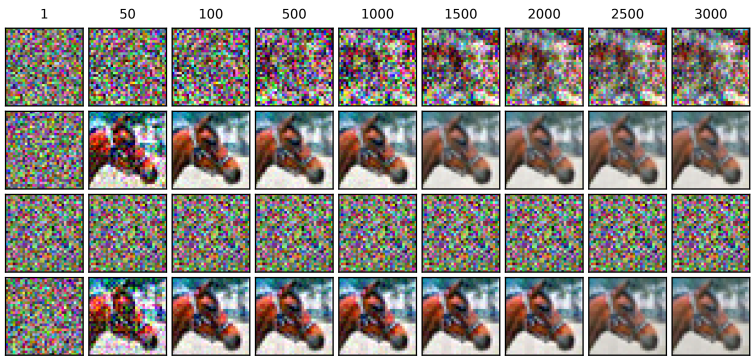
We further analyze the relationship between the attack performance and the overfitting levels of victim models in Figure 5.4b. We have made a similar discovery to the previous evaluation: Transformers suffer from higher attack accuracy than CNNs when the victim models are at the same overfitting level.

5.4.4 Evaluation on Gradient Inversion Attacks

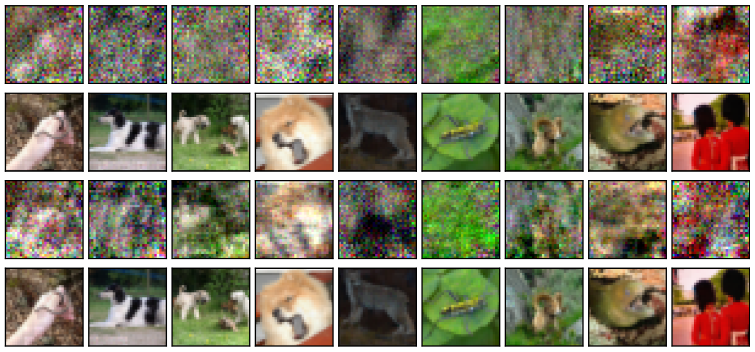
Table 5.5 presents the results of gradient inversion attacks on CNNs and Transformers using CIFAR10. Similar to our previous evaluations, we still compare these two model architectures in groups. The attacks are evaluated by multiple metrics, which measure reconstruction results between ground truth images and reconstruction images. The table clearly shows that the attacks on Transformers outperform the attacks on CNNs by a significant margin. Figure 5.5a provides examples of the reconstruction results. The attacks on CNNs fail to generate high-quality reconstruction images, whereas the attacks on Transformers produce remarkably accurate reconstructions that closely resemble the



(a) CIFAR10 with 3000 iterations.



(b) CIFAR10 with different iteration numbers (i.e. 1, 50, 100, 500, 1000, 1500, 2000, 2500, 3000).



(c) ImageNet1K with 3000 iterations.

Figure 5.5: The performance of gradient inversion attacks on CNNs and Transformers. From the top row to the bottom in each subfigure are ResNet-50, Swin-T, ResNet-101, and Swin-S.

originals.

Figure 5.5b provides the reconstruction results of the gradient inversion attacks over multiple iterations. It demonstrates the transformation of a raw dummy image towards a reconstruction that closely resembles the original image as the attack training continues. The reconstruction results reveal the varying degrees of success achieved

Table 5.6: The results of membership inference attacks on more models of CNNs and Transformers on CIFAR10.

	Task acc ↑	Attack acc ↑
ResNet-50	0.8220 ± 0.0023	0.6385 ± 0.0078
EfficientNet-B4	0.8180 ± 0.0052	0.6115 ± 0.0049
RegNetX-4.0GF	0.8326 ± 0.0018	0.6197 ± 0.0042
Swin-T	0.8335 ± 0.0042	0.6904 ± 0.0052
MViTV2-T	0.8279 ± 0.0057	0.6813 ± 0.0056
DaViT-T	0.8311 ± 0.0045	0.6752 ± 0.0039

Table 5.7: The results of attribute inference attacks on more models of CNNs and Transformers on CelebA.

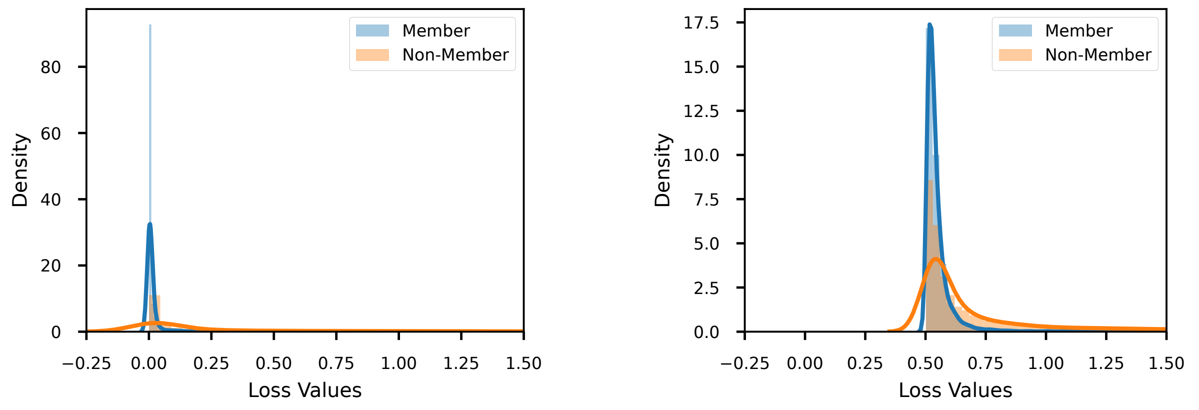
	Task acc ↑	Attack acc ↑	Macro F1 ↑
ResNet-50	0.6666 ± 0.0020	0.6854 ± 0.0015	0.3753 ± 0.0012
EfficientNet-B4	0.6192 ± 0.0038	0.6504 ± 0.0031	0.4036 ± 0.0018
RegNetX-4.0GF	0.6248 ± 0.0041	0.6701 ± 0.0016	0.3649 ± 0.0022
Swin-T	0.6587 ± 0.0023	0.7312 ± 0.0014	0.5530 ± 0.0019
MViTV2-T	0.6151 ± 0.0012	0.6908 ± 0.0025	0.4883 ± 0.0024
DaViT-T	0.6517 ± 0.0030	0.7263 ± 0.0015	0.5093 ± 0.0011

Table 5.8: The results of gradient inversion attacks on more models of CNNs and Transformers on CIFAR10.

	MSE ↓	PSNR ↑	LPIPS ↓	SSIM ↑
ResNet-50	1.3308 ± 0.6507	11.30 ± 2.24	0.1143 ± 0.0403	0.0946 ± 0.0989
EfficientNet-B4	0.8178 ± 0.7276	14.6970 ± 3.9812	0.1953 ± 0.0816	0.2698 ± 0.1413
RegNetX-4.0GF	0.7937 ± 0.4853	13.8190 ± 2.6519	0.0851 ± 0.0391	0.3197 ± 0.1282
Swin-T	0.0069 ± 0.0071	36.24 ± 5.21	0.0012 ± 0.0016	0.9892 ± 0.0118
MViTV2-T	0.0711 ± 0.0172	23.6450 ± 1.1224	0.0068 ± 0.0033	0.8639 ± 0.0463
DaViT-T	0.0615 ± 0.0425	25.9200 ± 4.7786	0.0090 ± 0.0076	0.9347 ± 0.0489

by the attacks on different models. In the case of ResNet-50, the reconstructed image shows limited resemblance to the original image. When attacking ResNet-101, the reconstruction result fails to capture any meaningful information. On the other hand, when targeting Transformers such as Swin-T and Swin-S, the attacks yield highly accurate reconstruction results since early iterations.

In the evaluation of gradient inversion attacks on ImageNet1K, we still compare the performance of ResNet-50, ResNet-101, Swin-T, and Swin-S models. Randomly selected images from ImageNet1K are used to generate reconstruction results, shown in Figure 5.5c. Similar to the previous experiments on CIFAR10, the attacks on ResNet variants have limited success in reconstructing the original images. In contrast, the attacks on Transformer models (Swin-T and Swin-S) yield significantly better reconstruction results.



(a) Loss distributions on ResNet-50 (b) Loss distributions on Swin-T

Figure 5.6: The loss distributions of membership inference attacks against ResNet-50 and Swin-T on CIFAR10.

5.4.5 Evaluation of More CNNs and Transformers on Privacy Attacks

Tables 5.6 to 5.8 show privacy attack results for more models of CNNs and Transformers. The comparison still involves models with similar parameter sizes, and our conclusion remains unchanged that Transformers exhibit higher attack performance.

We plot the loss distributions between the member and non-member data in membership inference attacks for both CNNs and Transformers in Figure 5.6. We can see that the distributions of member and non-member data for CNNs are more concentrated at 0, and the distributions for Transformers are concentrated at 0.5.

These findings highlight the higher vulnerability of Transformers to gradient inversion attacks compared to CNNs. This raises questions about specific architectural features in Transformers that contribute to their increased vulnerability to privacy attacks. We further analyze the vulnerability of architectural features in Section 5.5.

5.5 Which Architectural Features Can Lead to Higher Privacy Leakage?

In this section, we answer the fourth research question (**RQ4**). We delve into analyzing architectural features that can potentially lead to privacy leakage. Through a comprehensive analysis, we first examine the impact of partitioning a transformer model on privacy vulnerabilities. Then, we investigate the influence of various micro designs on

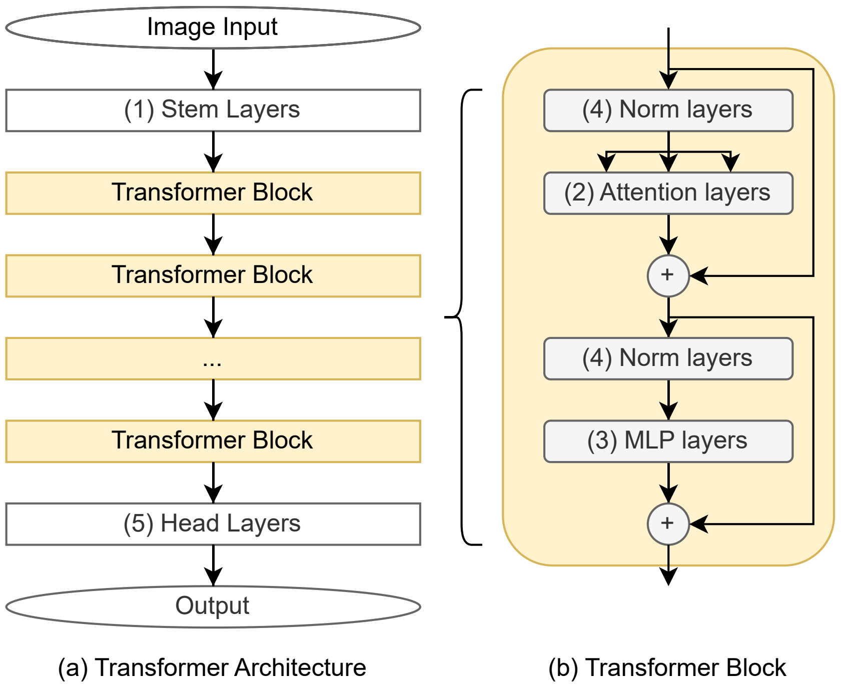


Figure 5.7: An illustration of a Vision Transformer architecture and the modules we focus on evaluating. Numbers (1) - (5) denote the modules we evaluate in gradient inversion attacks. (a): The whole model architecture. (b): Detailed architecture of each Transformer block.

model privacy and conduct an ablation study on these micro designs.

5.5.1 Segmenting a Transformer Model to Analyze the Privacy Leakage

One of the key distinctions between a CNN and a Transformer lies in the design of their respective blocks. In this section, we focus on segmenting a Transformer-based model and assessing the potential influence of specific layers on privacy leakage. As there are only prediction results and model representations from membership inference attacks and attribute inference attacks, it is difficult to evaluate some specific layers using these attacks. However, gradient inversion attacks offer a more precise means of evaluation, as they require a comprehensive list of gradients from each layer of the victim model.

In our analysis, we employ ViT-B as the victim model. This is because ViT models are one of the first Transformers in computer vision. Additionally, ViT models incorporate several common modules shared across various Transformer architectures. We utilize gradient inversion attacks as our chosen attack method. Rather than supplying all the

Table 5.9: The performance of gradient inversion attacks when segmenting ViT-B to make a selection of gradients.

Layers	Num of layers	Params	MSE ↓	PSNR ↑
All	152	85.65M	0.0007 ± 0.0003	43.70 ± 1.84
Stem	4	0.59M	0.0000 ± 0.0000	67.43 ± 5.03
Attention	48	28.34M	0.0020 ± 0.0009	39.61 ± 2.76
MLP	48	56.66M	0.0036 ± 0.0016	36.98 ± 2.59
Norm	48	0.05M	0.0040 ± 0.0018	36.57 ± 2.56
Head	4	0.01M	0.2776 ± 0.2312	19.01 ± 3.89

Table 5.10: The performance of gradient inversion attacks when segmenting Swin-T to make a selection of gradients.

Layers	Num of layers	Params	MSE ↓	PSNR ↑
All	173	27.51M	0.0069 ± 0.0071	36.24 ± 5.21
Stem	4	0.01M	0.0073 ± 0.0058	34.38 ± 2.89
Attention	48	8.64M	0.2691 ± 0.2099	17.72 ± 1.21
MLP	68	18.82M	0.3132 ± 0.1001	17.06 ± 3.03
Norm	49	0.03M	0.3526 ± 0.0826	16.65 ± 0.95
Head	4	0.01M	1.1179 ± 0.2996	11.71 ± 1.29

gradients to the attacks, we selectively provide specific gradients for evaluation purposes. The architecture of a Vision Transformer (an example of ViT-B) is depicted in Figure 5.7. To facilitate our assessment, we divide the model into five distinct modules based on their layer designs. Subsequently, we evaluate the influence of gradients obtained from each module individually. If the attack using gradients from Module A yields a higher attack accuracy compared to the attack using gradients from Module B, it suggests that Module A is more likely to reveal more information about the data samples than Module B.

- Module 1: Stem layers. This module receives the model's input and has patch embedding and position embedding layers.
- Module 2: Attention layers. They are in the Transformer block, and this module is the main reason why Transformers are different from CNNs.
- Module 3: MLP layers. They are also in the Transformer block.
- Module 4: Norm layers. They are located right before the attention layers and MLP layers. LayerNorm is often used as Norm layers in Transformers.
- Module 5: Head layers. They are the last few layers for producing the output of the model. A few fully connected layers could be used as head layers.

Table 5.9 presents the reconstruction results of gradient inversion attacks when only gradients from selected layers are utilized in the attacks. The "All" layers represent the

Table 5.11: Attack performance on ConvNeXt-T compared with ResNet-50 and Swin-T. NN MIA for Network-based membership inference, AIA for attribute inference, and GIA for gradient inversion.

		ResNet-50	ConvNeXt-T	Swin-T
NN MIA	Attack acc ↑	0.6385 ± 0.0078	0.7471 ± 0.0052	0.6904 ± 0.0052
AIA	Attack acc ↑	0.6854 ± 0.0015	0.7203 ± 0.0020	0.7312 ± 0.0014
	Macro F1 ↑	0.3753 ± 0.0012	0.5469 ± 0.0011	0.5530 ± 0.0019
GIA	MSE ↓	1.5096 ± 0.5538	0.0177 ± 0.0171	0.0069 ± 0.0071
	PSNR ↑	10.58 ± 1.87	31.88 ± 5.04	36.24 ± 5.21
	LPIPS ↓	0.1624 ± 0.0613	0.0032 ± 0.0055	0.0012 ± 0.0016
	SSIM ↑	0.0896 ± 0.0544	0.9666 ± 0.0451	0.9892 ± 0.0118

default attack scenario, where gradients from all layers are employed. The stem layers contain the patch embedding and position embedding processes. These layers exhibit minimal changes in the output compared to the original image sample. As the stem layers comprise only four layers, they can be relatively easier to attack compared to other types of layers. Consequently, the attacks on the stem layers demonstrate excellent performance, and we will further assess stem layers in the next subsection.

Among the attacks conducted with the remaining selected layers, the attack that demonstrates the best performance is the one utilizing "attention layers." This attack achieves an MSE of 0.0020 and a PSNR of 39.61. These results indicate that the attention layers are more susceptible to attacks, suggesting that they potentially leak more information about the data samples. We also show additional results using Swin-T in Table 5.10, drawing the same conclusion.

5.5.2 Impact of Other Micro Designs on Privacy

In the previous subsection, we established that attention layers within Transformers can contribute to privacy leakage. However, it is important to recognize that other micro designs within Transformers may also impact privacy vulnerabilities. One example is ConvNeXt [110], a convolutional neural network that incorporates multiple schemes from Transformers, similar to the Swin Transformer. ConvNeXt-T, ResNet-50, and Swin-T all share a similar parameter size, with ConvNeXt-T having approximately 27.83 million parameters. This allows for a direct comparison of the attack performance between ConvNeXt-T and the other two models. When ConvNeXt-T is tested on CIFAR10 using the same attack settings, it achieves a task accuracy of 0.8258. This indicates that we can compare the attack performance of ConvNeXt-T with ResNet-50 and Swin-T. The results presented in Table 5.11 further confirm the high attack performance on ConvNeXt-T.

Table 5.12: Detailed Architecture Specifications for 14 steps (Steps 1 to 6).

	ResNet-50, Step 1	Step 2	Step 3
stem	7 × 7, 64, stride 2 3 × 3, max pool, stride 2	7 × 7, **96**, stride 2 3 × 3, max pool, stride 2	7 × 7, 96, stride 2 3 × 3, max pool, stride 2
block1	[1 × 1, 64 3 × 3, 64 1 × 1, 256] × 3	[1 × 1, **96** 3 × 3, **96** 1 × 1, **384**] × 3	[1 × 1, 96 3 × 3, 96 1 × 1, 384] × **3**
block2	[1 × 1, 128 3 × 3, 128 1 × 1, 512] × 4	[1 × 1, **192** 3 × 3, **192** 1 × 1, **768**] × 4	[1 × 1, 192 3 × 3, 192 1 × 1, 768] × **3**
block3	[1 × 1, 256 3 × 3, 256 1 × 1, 1024] × 6	[1 × 1, **384** 3 × 3, **384** 1 × 1, **1536**] × 6	[1 × 1, 384 3 × 3, 384 1 × 1, 1536] × **9**
block4	[1 × 1, 512 3 × 3, 512 1 × 1, 2048] × 3	[1 × 1, **768** 3 × 3, **768** 1 × 1, **3072**] × 3	[1 × 1, 768 3 × 3, 768 1 × 1, 3072] × **3**
other specs	ReLU, BN	ReLU, BN	ReLU, BN
-	-	-	-
	Step 4	Step 5	Step 6
stem	**4** × **4**, 96, stride **4** 3 × 3, max pool, stride 2	4 × 4, 96, stride 4 3 × 3, max pool, stride 2	4 × 4, 96, stride 4 3 × 3, max pool, stride 2
block1	[1 × 1, 96 3 × 3, 96 1 × 1, 384] × 3	[1 × 1, 96 **d3** × **3**, 96 1 × 1, 384] × 3	[1 × 1, 96 d3 × 3, **384** 1 × 1, **96**] × 3
block2	[1 × 1, 192 3 × 3, 192 1 × 1, 768] × 3	[1 × 1, 192 **d3** × **3**, 192 1 × 1, 768] × 3	[1 × 1, 192 d3 × 3, **768** 1 × 1, **192**] × 3
block3	[1 × 1, 384 3 × 3, 384 1 × 1, 1536] × 9	[1 × 1, 384 **d3** × **3**, 384 1 × 1, 1536] × 9	[1 × 1, 384 d3 × 3, **1536** 1 × 1, **384**] × 9
block4	[1 × 1, 768 3 × 3, 768 1 × 1, 3072] × 3	[1 × 1, 768 **d3** × **3**, 768 1 × 1, 3072] × 3	[1 × 1, 768 d3 × 3, **3072** 1 × 1, **768**] × 3
other specs	ReLU, BN	ReLU, BN	ReLU, BN

These findings suggest that ConvNeXt-T and Swin-T exhibit vulnerability to various attacks at a similar level.

The difference between ConvNeXt and the Swin Transformer lies in the attention module. This allows us to explore and investigate the privacy implications of various micro designs in model architectures beyond just the attention layers. By studying the impact of different micro designs, we can gain deeper insights into the specific architectural features that may pose privacy risks.

We continue with utilizing gradient inversion attacks as a tool for further examination. Following the design process outlined in [110], we have made several adjustments based

Table 5.13: Detailed Architecture Specifications for 14 steps (Steps 7 to 12).

	Step 7	Step 8	Step 9
stem	4 × 4, 96, stride 4 3 × 3, max pool, stride 2	4 × 4, 96, stride 4 **(removed)**	4 × 4, 96, stride 4 **(removed)**
block1	[**d7 × 7**, 96 1 × 1, 384 1 × 1, 96] × 3	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3
block2	[**d7 × 7**, 192 1 × 1, 768 1 × 1, 192] × 3	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3
block3	[**d7 × 7**, 384 1 × 1, 1536 1 × 1, 384] × 9	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9
block4	[**d7 × 7**, 768 1 × 1, 3072 1 × 1, 768] × 3	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3
other specs	ReLU, BN	ReLU, BN	**GELU**, BN
-	-	-	-
	Step 10	Step 11	Step 12
stem	4 × 4, 96, stride 4	4 × 4, 96, stride 4	4 × 4, 96, stride 4
block1	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3
block2	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3
block3	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9
block4	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3
other specs	**Fewer GELU**, BN	Fewer GELU, **Fewer BN**	Fewer GELU, **Fewer LN** **Conv w/ True bias**

on our own analysis. Since ConvNeXt is constructed incrementally from ResNet, we meticulously scrutinized each model architecture to investigate the steps that contribute most significantly to privacy leakage. We focus on ResNet-50 and ConvNeXt-T models and conduct tests on a total of 14 model architectures using randomly selected samples from CIFAR10.

The analysis involves a 14-step process, where we begin by modifying the overall architecture (Steps 1 to 4) based on ResNet-50. Subsequently, we align the bottleneck design (Steps 5 to 7) and refine the stem layer design in Step 8. Finally, we align the micro designs (Steps 9 to 14) to achieve ConvNeXt-T. This 14-step process, from macro to micro

Table 5.14: Detailed Architecture Specifications for 14 steps (Steps 13 and 14).

	Step 13	ConvNeXt-T, Step 14
stem	4 × 4, 96, stride 4	4 × 4, 96, stride 4
block1	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3	[d7 × 7, 96 1 × 1, 384 1 × 1, 96] × 3
sep ds	**2 × 2, 192, stride 2**	2 × 2, 192, stride 2
block2	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3	[d7 × 7, 192 1 × 1, 768 1 × 1, 192] × 3
sep ds	**2 × 2, 384, stride 2**	2 × 2, 384, stride 2
block3	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9	[d7 × 7, 384 1 × 1, 1536 1 × 1, 384] × 9
sep ds	**2 × 2, 768, stride 2**	2 × 2, 768, stride 2
block4	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3	[d7 × 7, 768 1 × 1, 3072 1 × 1, 768] × 3
other specs	Fewer GELU, Fewer LN Conv w/ True bias	Fewer GELU, Fewer LN Conv w/ True bias **Stochastic depth, Layer Scale**

levels, as outlined in [110], has been established as an optimal procedure for designing ConvNeXt. Leveraging this proven methodology, we utilize it to analyze the impact on privacy. By identifying the critical steps that contribute to privacy leakage, we can assess the influence of micro designs on the privacy of model architectures. This analysis provides valuable insights into understanding the specific stages or modifications that may pose risks to privacy. The 14 steps are outlined below:

1. **ResNet-50:** We begin our process with this model.
2. **Changing channel dimensions:** Each stage in ResNet-50 uses different channel dimensions (i.e., $(64, 128, 256, 512)$). To align with ConvNeXt-T, we modify these dimensions to $(96, 192, 384, 768)$.
3. **Changing the stage compute ratio:** ResNet employs a multi-stage design that modifies channel dimensions. ResNet-50 has a stage compute ratio of $(3, 4, 6, 3)$, while ConvNeXt and Swin Transformers adopt $(3, 3, 9, 3)$. We follow this adjustment in this step.
4. **Applying "Patchify":** Vision transformers process input images by sliding them into patches. Here, we replace the stem convolutional layers with a kernel size of (4×4) and a stride of 4.

5. **Applying "ResNeXtify":** ResNeXt [183] introduced grouped convolution, reducing parameter size while maintaining performance. We utilize depth-wise convolution, which employs the same number of groups and channels.
6. **Using the inverted bottleneck:** The inverted bottleneck design is widely employed in models like MobileNet, ConvNeXt, and Swin Transformers. We incorporate this step into our process.
7. **Enlarging kernel sizes:** To align the parameters with ConvNeXt, we adopt a larger kernel size of (7×7) instead of (3×3).
8. **Forming the new stem layers:** In this step, we remove the activation and maxpool layers, originally part of ResNet.
9. **Changing ReLU to GELU:** The Gaussian Error Linear Unit (GELU) [67] is a variant of ReLU commonly used in Transformers. We introduce this change to the model.
10. **Removing some activation layers:** Transformer blocks typically have fewer activation layers. Here, we retain only one activation layer within the block.
11. **Removing some normalization layers:** We reduce the number of BatchNorm (BN) layers within the block to one.
12. **Changing BN to LN:** Inspired by the prevalent use of LayerNorm (LN) layers [4] in Transformers, we replace the BN layers in our model with LN layers. We also enable bias parameters for all convolutional layers in this step.
13. **Separating downsampling layers:** We move the downsampling layers between stages, introducing an LN layer to ensure stability during training.
14. **Final touches to reach ConvNeXt-T:** We incorporate Stochastic Depth [76] and Layer Scale [163] in the final stage to complete the ConvNeXt-T model.

We present a detailed architecture comparison between models from 14 steps (changing from ResNet-50 to ConvNeXt-T) in Tables 5.12 to 5.14. Changes are marked in bold.

Figure 5.8 and Table 5.15 show the performance of gradient inversion attacks on each model architecture. The 14-step changes from macro to micro levels exhibit fluctuations but generally show a trend of improved attack performance.

Figure 5.8 provides qualitative results of the attacks. During the initial stages (Steps 1 to 3), the attacks struggle to reconstruct proper images. In the middle stages, some information emerges in the reconstruction results, but to a limited extent. The recon-

Table 5.15: The results of gradient inversion attacks on model architectures from 14 steps. Some significant changes in results are marked in bold.

Steps	Task acc ↑	MSE ↓	PSNR ↑	LPIPS ↓	**SSIM** ↑
1. ResNet-50	0.8220 ± 0.0039	1.5096 ± 0.5538	10.58 ± 1.87	0.1624 ± 0.0613	0.0896 ± 0.0544
2. Channel dim	0.8240 ± 0.0072	1.4706 ± 0.5710	10.74 ± 1.97	0.1724 ± 0.0616	0.0826 ± 0.0405
3. Stage ratio	0.8282 ± 0.0040	1.5286 ± 0.5246	10.56 ± 2.05	0.1834 ± 0.0581	0.0731 ± 0.0613
4. Patchify	0.8293 ± 0.0061	**0.9011** ± **0.4376**	**12.97** ± **2.10**	**0.0867** ± **0.0436**	**0.1727** ± **0.0794**
5. ResNeXtify	0.8397 ± 0.0033	1.2415 ± 0.6934	11.86 ± 2.77	0.1066 ± 0.0391	0.1334 ± 0.0950
6. Inv bottleneck	0.8407 ± 0.0058	1.1123 ± 0.4994	12.06 ± 2.19	0.0989 ± 0.0290	0.1429 ± 0.0844
7. Kernel sizes	0.8432 ± 0.0052	0.8206 ± 0.3543	13.40 ± 2.30	0.0821 ± 0.0355	0.2353 ± 0.0766
8. New stem	0.8459 ± 0.0043	0.5684 ± 0.3564	15.43 ± 3.01	0.0752 ± 0.0381	0.4924 ± 0.1205
9. ReLU to GELU	0.8436 ± 0.0027	1.0540 ± 0.5075	12.42 ± 2.61	0.2422 ± 0.0904	0.1746 ± 0.1166
10. Removing Act	0.8480 ± 0.0064	**0.0215** ± **0.0150**	**29.93** ± **3.58**	**0.0049** ± **0.0026**	**0.9562** ± **0.0224**
11. Removing BN	0.8491 ± 0.0059	0.0198 ± 0.0139	30.57 ± 4.12	0.0045 ± 0.0032	0.9605 ± 0.0232
12. BN to LN	0.8501 ± 0.0031	**0.0049** ± **0.0044**	**36.86** ± **3.96**	**0.0005** ± **0.0003**	**0.9927** ± **0.0064**
13. Sep downsamp	0.8553 ± 0.0070	0.0121 ± 0.0171	33.79 ± 4.69	0.0011 ± 0.0008	0.9859 ± 0.0151
14. ConvNeXt	0.8523 ± 0.0064	0.0177 ± 0.0171	31.88 ± 5.04	0.0032 ± 0.0055	0.9666 ± 0.0451

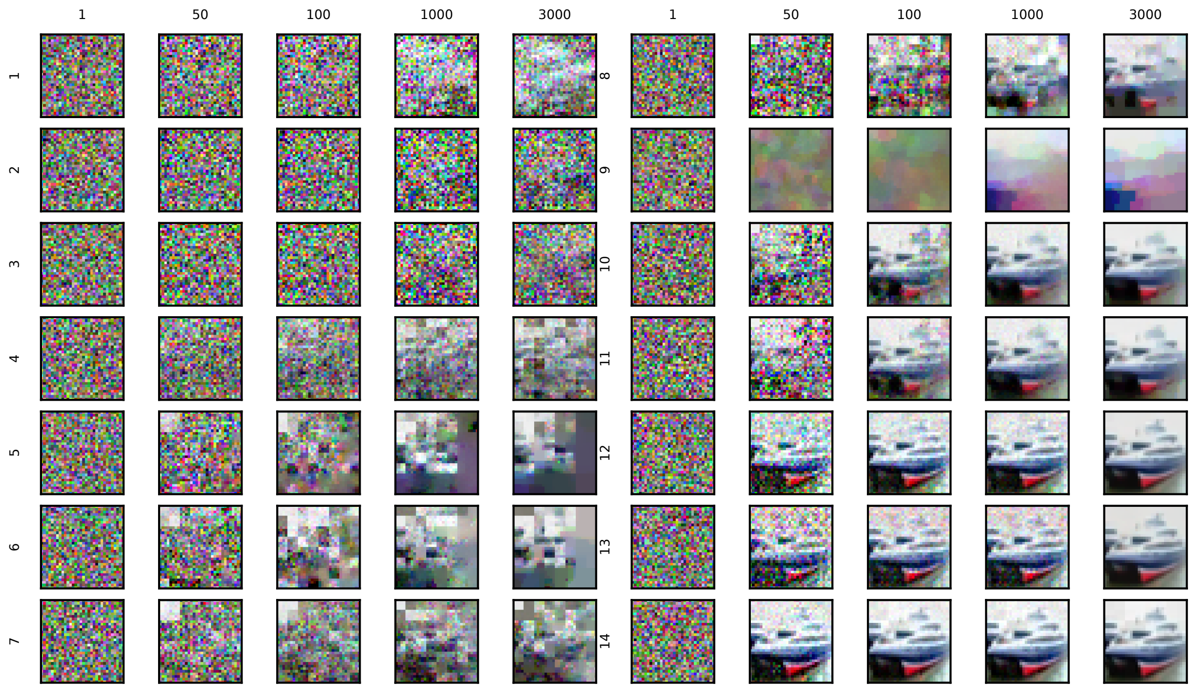


Figure 5.8: The performance of gradient inversion attacks on each architecture changing from ResNet-50 to ConvNeXt-T with several selected iterations (i.e., 1, 50, 100, 1000, 3000). Model architectures from 14 steps are shown.

struction results improve in the later stages (After Step 10). At last, using ConvNeXt-T, which is step 14, the attacks achieve a good attack performance.

Table 5.15 presents more information on gradient inversion attacks, highlighting their performance in various architectural changes. Several steps exhibit significant changes compared to other steps. We define a step as a significant change when a metric

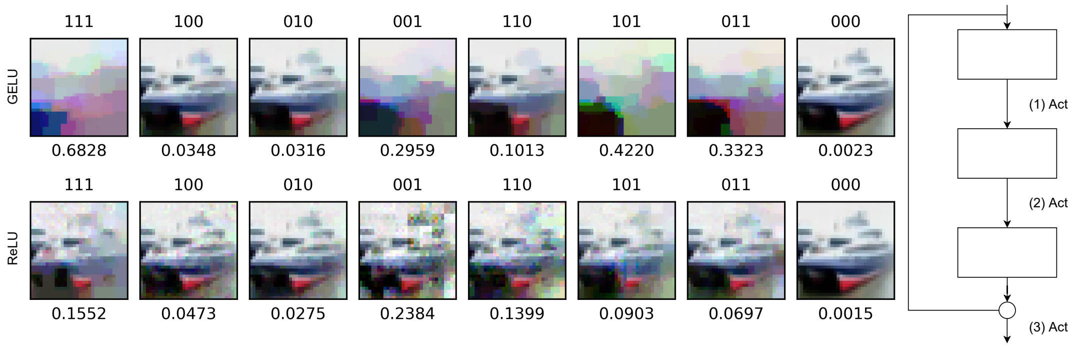
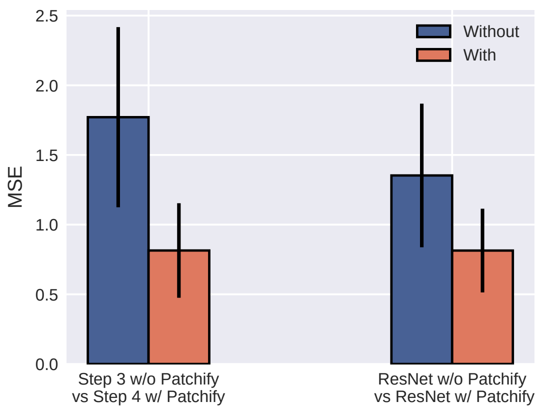


Figure 5.9: The performance of gradient inversion attacks on models with different positions of activation layers. The position of three activation layers is illustrated on the right, with two between convolutional layers and one after the addition operation. The three digits on the top of the subfigures show whether the activation layer on this position is added or not. The bottom of each subfigure provides MSE values for the attack on this model (i.e., models with GELU or ReLU).

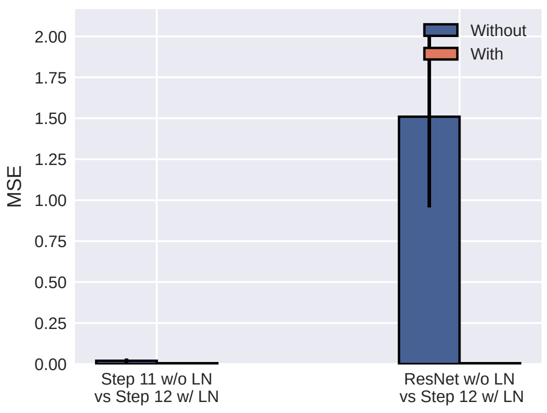
notably decreases or increases. To illustrate, we consider the MSE metric, which we define a step as a significant change if it experiences a reduction exceeding 50%. Our analysis reveals three significant increases in attack performance, each correlated with specific architectural changes: The first one occurs when applying "Patchify" to stem layers; The second one occurs with the removal of some activation layers; The third one happens when changing BN to LN. These findings underscore the pivotal role these specific architectural changes play in determining the model's vulnerability to privacy attacks.

5.5.3 Ablation Study on Micro Designs

Ablation study on the design of activation layers. One of the differences between a Transformer block and a ResNet block is that a Transformer block has fewer activation layers. Leaving fewer activation layers in the model boosts the attack performance. As illustrated in Figure 5.9, when different activation layers are removed, there is a noticeable improvement in the attack performance. Particularly, the removal of the third activation layer, located after the skip connection of the ResNet block, results in a significant enhancement in attack accuracy. This observation suggests that this activation layer introduces a non-linear process that reduces the amount of information available for the attack, thus making it harder to reconstruct the original input. By removing this layer, the attack performance is improved. The analysis further highlights that changing ReLU to GELU results in a marginal reduction in attack performance.



(a) Patchify



(b) LN

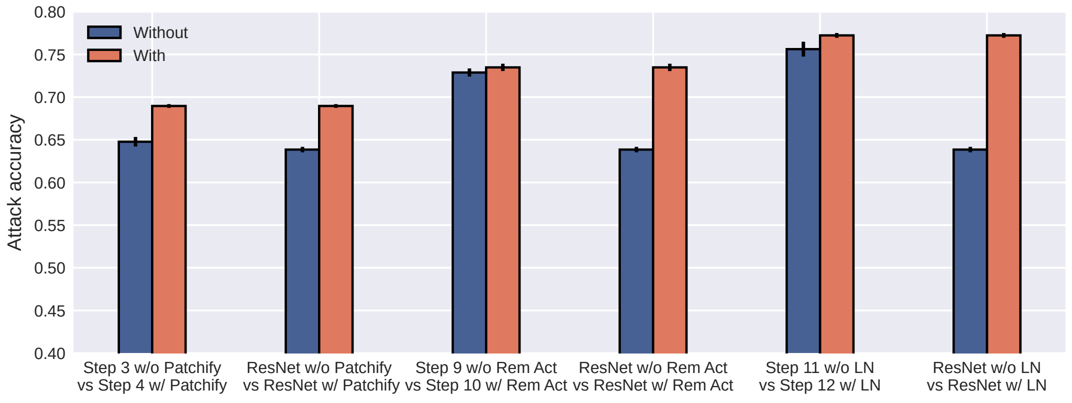
Figure 5.10: The performance of gradient inversion attacks on model architectures with or without some micro designs.

This is attributed to the fact that GELU is a smoother approximation of ReLU, which strengthens the model's robustness and generalization [6]. Using GELU makes the adversary extract less information from data samples, consequently decreasing their attack efficacy. In sum, the most significant enhancement in attack performance occurs when all activation layers are removed from the model architecture.

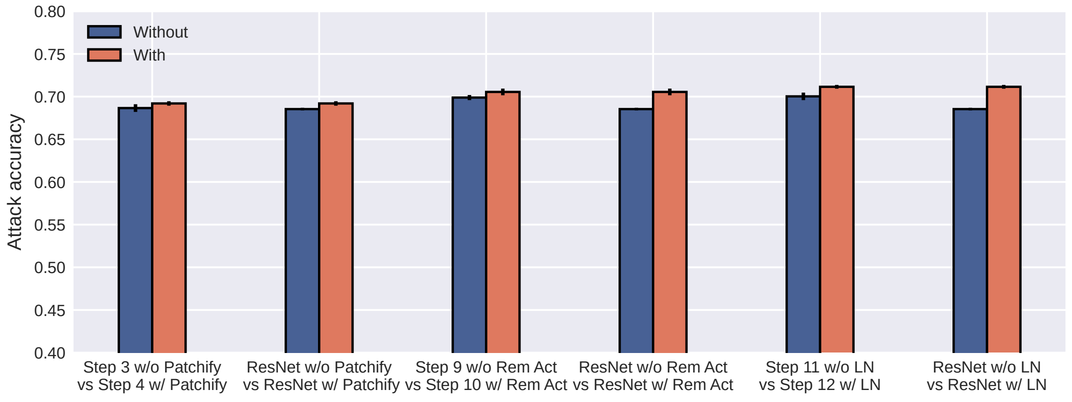
Ablation study on the design of stem layers and LN layers. The additional analysis presented in Figure 5.10 highlights two more features that appear to have an impact on model privacy leakage. Firstly, Figure 5.10a demonstrates that the utilization of "Patchify" reduces reconstruction MSE results. As this process involves modifications to the stem layers, we believe that these stem layers are crucial for privacy attacks. Secondly, Figure 5.10b illustrates that changing BN to LN results in improved reconstruction results. This suggests that incorporating LN also contributes to the model's privacy leakage. These findings emphasize the significance of stem layers and LN layers on model privacy leakage.

Ablation study on micro designs for membership inference attacks and attribute inference attacks. Figure 5.11 shows the impact of three micro designs on more attacks. Just like previous studies, adding these designs can also increase privacy leakage in these two attacks.

In summary, attention modules, as well as the design of activation layers, stem layers, and LN layers, are the key architectural features that lead to more privacy leakage. We provide more discussions in Section 5.6.



(a) Membership inference



(b) Attribute inference

Figure 5.11: The performance of privacy attacks on model architectures with or without some features.

5.6 Discussion

In the previous sections, we discovered that four design components in Transformers could result in privacy leakage: attention modules, activation layers, stem layers, and LN layers. In this section, we would like to provide a more in-depth discussion of these modules.

5.6.1 The Impact of Attention Modules

In Section 5.5.1, we show that the utilization of attention modules makes Transformers more susceptible to privacy leakage than CNNs. This susceptibility is attributed to the larger receptive field of attention modules, contributing to increased model memorization and, subsequently, heightened privacy leakage.

The receptive field of a model refers to the information received within a specified range by a neuron in a model layer. In a fully connected neural network, each neuron receives input from the elements of the entire input sample. Due to the convolution operation, the neuron in a convolutional network receives input limited to its receptive field. The range of the receptive field is defined by the convolution templates in CNNs. This design allows CNNs to capture local patterns in the input data efficiently. The receptive field has a theoretical limit. Some researchers have demonstrated that the effective receptive field (i.e., the receptive field's effective area) is smaller than the theoretical receptive field [115]. From a privacy perspective, a CNN model could only reveal part of sensitive information from the input sample due to the design of localized convolution templates.

Transformers employ the multi-head self-attention mechanism, also known as attention modules. The input sample is taken as a sequence of flattened 2D patches. The attention module receives the input sequence and generates its representation of the sequence by mapping the query and the key-value pairs to the output. Transformers tend to have much larger receptive fields than CNNs due to the fact that their attention module is computed with the entire input sequence [36, 167]. In terms of privacy, a Transformer model is able to extract more sensitive information than a CNN model because of its wide-angle receptive field. Hence, Transformers are more prone to attacks than CNNs, as demonstrated by our evaluation based on three popular privacy attacks. In the following paragraphs, let us explore this theoretical framework step by step:

Receptive Field. Given a neural network with n layers, the output of the i-th layer ($i \in \{1,...,n\}$) is the feature map f_i. We have the input image as f_1 and the final output feature map as f_n. We define $\mathscr{R}_i$ as the receptive field size of feature map f_i. In other words, $\mathscr{R}_i$ is related to the process of the i-th layer. The receptive field $\mathscr{R}_i$ can be defined as

$$\mathscr{R}_i = \zeta(\mathscr{R}_{i+1}, \alpha_1, \alpha_2, ..., \alpha_n), \tag{5.1}$$

where $\alpha_1, \alpha_2, ..., \alpha_n$ are calculation factors in $\mathscr{R}_{i+1}$, and ζ is the calculation process of $\mathscr{R}_{i+1}$.

In a CNN model, the receptive field of a neuron represents the convolution-computation region in the input space that influences the output of that neuron. It signifies the area of the input data to which the neuron reacts. According to [2], the receptive field in a CNN is determined by:

$$\mathscr{R}_i^C = s_{i+1} \cdot \mathscr{R}_{i+1}^C + (k_{i+1} - s_{i+1}), \tag{5.2}$$

where k_i and s_i are the kernel size and the stride in a convolution layer. Based on Equations (5.1) and (5.2), we can define the receptive field size of a neuron in a CNN layer $\mathscr{R}^C$ as follows (we omit i for the simplicity of notations):

$$\mathscr{R}^C = \zeta(k, s), \tag{5.3}$$

where k and s are the kernel size and the stride.

The receptive field in a Transformer model, utilizing attention mechanisms, significantly differs from CNNs. In Transformers, attention layers calculate each position in the input sequence using query (q), key (k), and value (v) matrices. The output of the attention module is computed as follows:

$$\text{Attention}(q, k, v) = \text{SoftMax}(q \cdot k^T / \sqrt{d}) \cdot v, \tag{5.4}$$

where d is the scaling factor based on query and key. Based on Equations (5.1) and (5.4), we can define the size of the receptive field of an attention layer $\mathscr{R}^T$ as:

$$\mathscr{R}^T = \zeta(q, k, v). \tag{5.5}$$

As q, k, and v in a Transformer model are derived from the entire input sequence, $\mathscr{R}^T$ tends to be much larger. With Equations (5.3) and (5.5), we have:

$$\mathscr{R}^T > \mathscr{R}^C. \tag{5.6}$$

In summary, Transformers activate a larger receptive field of input data compared to CNNs, which is also supported by empirical research [133].

Model Memorization. The concept of model memorization lacks a formal definition in the research field. Informally, it can be characterized as a label ($\mathbf{y}$) memorization for a specific sample $\mathbf{x}$ if removing this $\mathbf{x}$ from the training set would change the performance of the model predicting $\mathbf{y}$ [16, 44]. Other research refers to it as memory capacity, assessing how much information a model can store in its parameters [14, 29, 135]. In either case, model memorization is intricately linked to how the model learns from input data. The relationship between the receptive field and model memorization can be formulated by:

$$\text{increase}(\mathscr{M}) \sim \text{increase}(\mathscr{R}), \tag{5.7}$$

where $\mathscr{R}$ denotes the receptive field and $\mathscr{M}$ denotes the model memorization. A larger receptive field allows the model to reap more information from the input data, and as

Transformers inherently possess a larger receptive field, they exhibit an enhanced ability to absorb more information. With Equations (5.6) and (5.7), we have:

$$\mathscr{M}^T > \mathscr{M}^C, \tag{5.8}$$

where $\mathscr{M}^T$ is the model memorization in Transformers, and $\mathscr{M}^C$ is the model memorization in CNNs.

Privacy Leakage. Model memorization introduces potential privacy leakage. Normally, input data $\mathbf{x}$ is considered extractably memorized if an adversary can construct data that makes the model produce $\mathbf{x}$ [123]. The relationship between model memorization and privacy leakage can be expressed as:

$$\text{increase}(\mathscr{L}) \sim \text{increase}(\mathscr{M}), \tag{5.9}$$

where $\mathscr{M}$ is the model memorization and $\mathscr{L}$ denotes the privacy leakage. An increased capacity for the model to learn from input data correlates with a higher likelihood of privacy leakage. With Equations (5.8) and (5.9), we have:

$$\mathscr{L}^T > \mathscr{L}^C, \tag{5.10}$$

where $\mathscr{L}^T$ is the privacy leakage in Transformers, and $\mathscr{L}^C$ is the privacy leakage in CNNs. In summary, Transformers, employing attention modules and featuring a larger receptive field, inherently enhance model memorization, thereby contributing to increased privacy leakage.

5.6.2 The Impact of the Design of Activation Layers

In Sections 5.5.2 and 5.5.3, we show that removing activation layers could make the model more vulnerable to privacy attacks. Here we delve into more theoretical discussions.

ReLU vs GELU. ReLU and GELU are two widely adopted activation layers in CNNs and Transformers, defined as follows:

$$\text{ReLU}(\mathbf{x}) = \max(0, \mathbf{x}), \tag{5.11}$$

$$\text{GELU}(\mathbf{x}) = 0.5\mathbf{x}(1 + \tanh(\sqrt{\frac{2}{\pi}}(\mathbf{x} + 0.044715\mathbf{x}^3))), \tag{5.12}$$

where $\mathbf{x}$ is the input of the activation layer.

Both ReLU and GELU introduce non-linearity into the model, enabling the learning of complex and hierarchical representations from input data. However, non-linearity can also bring challenges during model training, such as the vanishing gradient and exploding gradient problems, making it difficult to update model weights and biases [94]. The ReLU unit may suppress half of its inputs by outputting zero values, selectively passing information to the next layer [115]. Moreover, the non-differentiability of ReLU at $\mathbf{x} = 0$ poses issues in gradient-based optimization. The GELU unit, designed as a smooth approximation to ReLU, preserves the non-linearity of the model [94]. The adoption of GELU can enhance the model's robustness and generalization, making the adversary extract less information from data samples and decrease the attack performance [6]. This aligns with our experimental results presented in Step 9 of Table 5.15 and Figure 5.9.

The removal of activation layers. As using GELU does not help with improving the privacy attack performance, our analysis shows that removing activation layers and leaving fewer activation layers is the key to increasing the model's vulnerability to privacy attacks (shown in Figure 5.9). This is because retaining fewer activation layers allows the model to preserve more information learned from the training data, resulting in increased privacy leakage compared to models with a greater number of activation layers, which can be written as $\mathscr{L}^{\text{fewer act}} > \mathscr{L}^{\text{more act}}$.

5.6.3 The Impact of the Design of Stem Layers

In Sections 5.5.2 and 5.5.3, we illustrate that the design of stem layers (the "Patchify" step) can result in increased privacy leakage. Here, we provide more theoretical discussions.

The stem layers in a standard ResNet have a 7×7 convolution layer with stride 2. The "Patchify" strategy introduced by Transformers uses a 4×4 convolution layer with stride 4, which is actually a non-overlapping convolution process to make patches of the input data. Some research [37, 54] theoretically show that non-overlapping patches can easily learn information from input data, comparable to the capabilities of overlapping patches. Empirical studies [181] provide further support for the efficacy of this kind of stem layer design with convolutional layers. This design is shown to enhance optimization stability and improve the adversary's attack performance, which is demonstrated in our experimental results in Sections 5.5.2 and 5.5.3. We can formulate this as $\mathscr{L}^{\text{patchify stem}} > \mathscr{L}^{\text{resnet stem}}$.

5.6.4 The Impact of the Design of LN Layers

In Sections 5.5.2 and 5.5.3, we show that changing BatchNorm to LayerNorm can contribute to increased privacy leakage. Here, we provide more theoretical discussions on the implications of this design choice.

Let $\mathbf{x} = (x_1, x_2, ..., x_H)$ and $\mathbf{y}$ be the input of size H and output of a normalization layer. LayerNorm re-centers and re-scales the input x as

$$\mathbf{y} = \mathbf{g} \odot \frac{\mathbf{x} - \mu}{\sigma} + \mathbf{b}, \mu = \frac{1}{H} \sum_{i=1}^{H} x_i, \sigma = \sqrt{\frac{1}{H} \sum_{i=1}^{H} (x_i - \mu)^2}, \tag{5.13}$$

where $\odot$ denotes a dot production operation, μ and σ are the mean and standard deviation of $\mathbf{x}$, bias $\mathbf{b}$ and gain $\mathbf{g}$ are learnable parameters [4, 184].

The bias and gain are designed for affine transformation on normalized vectors to enhance performance. However, these parameters are trained based on the training data and may not adequately consider the input distributions of testing data. This increases the risk of overfitting when applying LayerNorm in the model [184]. The overfitting of the model would lead to more privacy leakage of the training data. As a result, models with LayerNorm could potentially expose sensitive information during privacy attacks, which can be written as $\mathscr{L}^{\text{ln layer}} > \mathscr{L}^{\text{bn layer}}$.

5.6.5 The Impact of Overfitting

Previous work claimed that privacy attacks are caused mainly by the undesirable overfitting issue in deep learning models [108, 145]. Overfitting normally occurs when a model performs well on the training data, but poorly on the validation data. The overfitting issue tends to become severe on an over-trained model with a large number of parameters. Deep learning models are exposed to privacy threats due to the overfitting effect. In our work, we find that model architectures have impacts on the performance of privacy attacks, which can *not* be attributed solely to the overfitting effect. Indeed, our experiments validate that the variation in performance is due to the difference in model architectures. For models with the same level of parameter sizes, Transformers tend to be more vulnerable to privacy attacks than CNNs. More importantly, for models with the same overfitting level, our conclusion still holds that Transformers are more vulnerable to privacy attacks than CNNs. We have then identified some architectural features that could be responsible for privacy leakage.

5.6.6 Potential (Incorrect) Explanations for the Vulnerability of Different Models Against Privacy Attacks

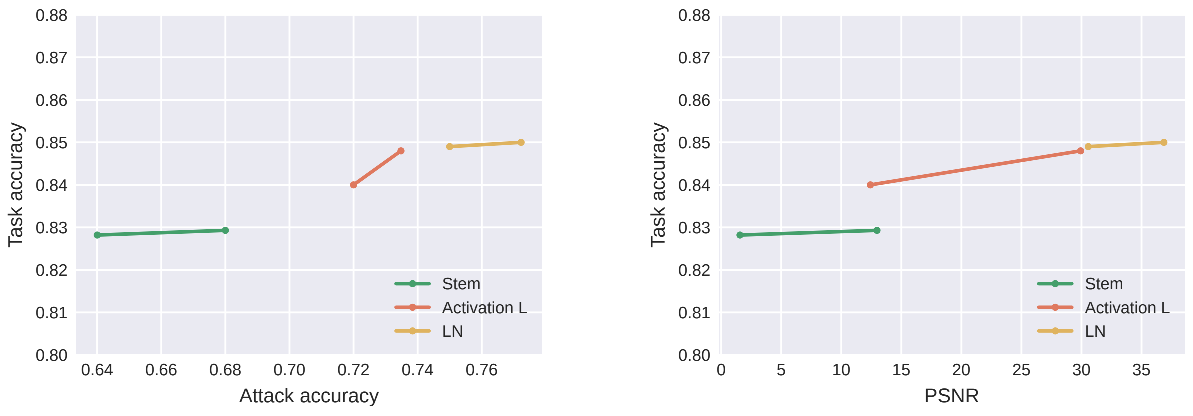
Here, we explore several potential explanations for our experimental results and shed more insights on our conclusion.

- The attacks on Transformers are more effective than those on CNNs. Is it due to random noise? To dismiss this concern, we have conducted multiple runs for each experiment setting and calculated the mean and standard deviation scores for our results. In total, We have done around 100 different experiment settings, involving 1400 experiments, 1700 training models, and 1200 training hours, which averages out the impact of random noise.
- Is it due to the overfitting of the victim models? To account for this, we meticulously compare the attacks on CNNs and Transformers when their victim models are trained on the same level of overfitting. By doing so, we minimize the factor of overfitting for a fair assessment.
- Is it due to the immature training of the victim models? In order to conduct our experiments, it was necessary to split the dataset into multiple subsets for both victim and shadow models. Consequently, it is reasonable to expect relatively low accuracy on the victim models for CIFAR10, CIFAR100, and CelebA (compared to the models trained on a full dataset). This aligns with the findings of the literature on privacy attacks, which have reported similar results [15, 65, 108].
- Is it due to other training factors? We ensure that the training recipes for each victim model remain consistent across all comparisons. Consequently, we can confidently exclude other training factors from the comparison study when drawing our conclusion.

After eliminating these potential explanations, we believe that our experimental results should be best explained by several architectural features mentioned in Section 5.5.

5.7 How to Exploit the Privacy Impact of Model Components?

Chapter 2 provides a comprehensive literature review of existing defense mechanisms against privacy attacks. However, these mechanisms do not specifically address or exploit



(a) Membership inference (b) Gradient inversion

Figure 5.12: The impact of the performance of privacy attacks when model components change. In each line, the left and right points represent the results without and with the micro design modifications, respectively.

the privacy impact of individual model components. To fill this gap, we propose two new mechanisms that leverage privacy leakages within model architectures.

5.7.1 Modifying Model Components as a Defense Mechanism

Based on our discoveries in previous sections, we propose to explore the modification of model components as a defense mechanism against privacy attacks. We aim to achieve a trade-off between utility and privacy. In Figure 5.12, we demonstrate the influence of certain micro-design changes on the efficacy of privacy attacks. Notably, we observe that the task accuracy is minimally impacted in each test, indicating that the proposed defense measures do not significantly compromise the model's overall performance. However, we can observe a decreased effectiveness of membership inference attacks and gradient inversion attacks when these model components are not applied.

These findings suggest that the proposed micro-design modifications can serve as effective countermeasures against privacy attacks, while the task accuracy remains almost intact.

5.7.2 Adding Perturbations as a Defense Mechanism

The privacy leakage issue observed in Transformers highlights the need for an enhanced privacy treatment compared to CNNs. Specifically, when employing perturbation as a defense mechanism, such as incorporating differential privacy noises into the model

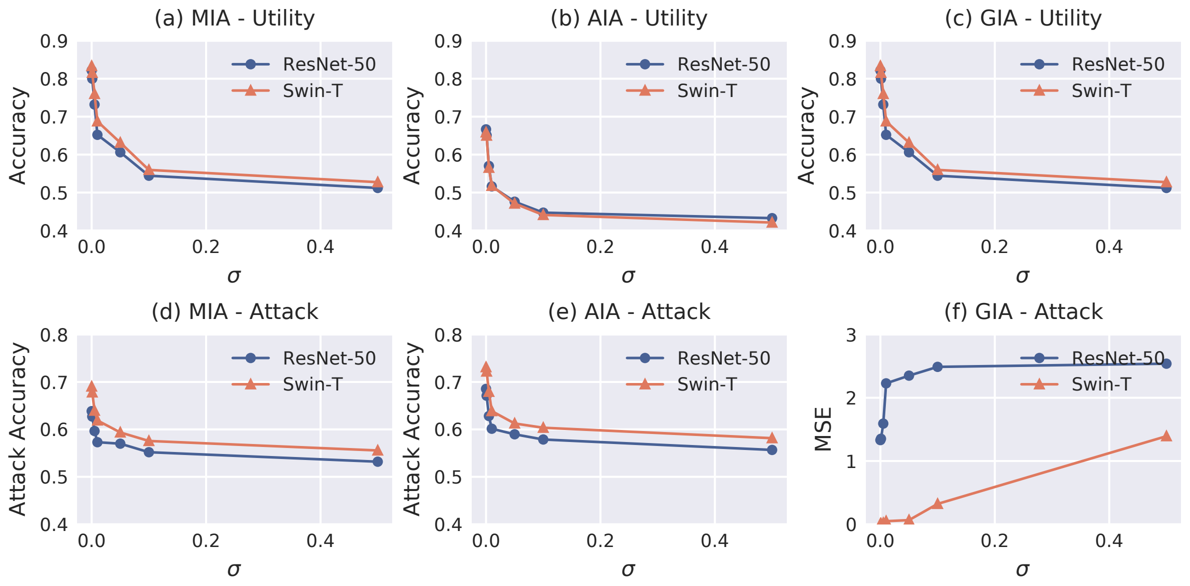


Figure 5.13: The performance of privacy attacks when DPSGD is applied. The utility performance of victim models and the attack performances are given when various DP noises are given.

parameters, it is better to increase the level of noise specifically for Transformers and Transformer-like models. Figure 5.13 illustrates the outcomes of three privacy attacks when DPSGD [1] is applied. As the differential privacy noise increases, a noticeable decline in the utility performance of victim models is observed (depicted in subfigures (a) to (c)). Concurrently, subfigures (d) to (f) reveal diminishing attack performances, with Transformers (Swin-T) consistently exhibiting superior attack performance at equivalent noise levels. From a defensive standpoint, adding more noise to Transformers is recommended.

Another approach to consider is a layer-wise perturbation defense mechanism, where noises are added to selected layers only. In this scenario, it would be interesting to introduce additional noises to the layers that are more susceptible to privacy leakages, such as activation layers, stem layers, LN layers, and attention modules. Paying special attention to these "privacy-leakage" layers could also achieve satisfactory privacy protection. Figure 5.14 demonstrates the impact of privacy attacks when DPSGD is applied to stem layers. As the differential privacy noise escalates, it influences both utility and attack performance, demonstrating the efficacy of adding noises to targeted layers.

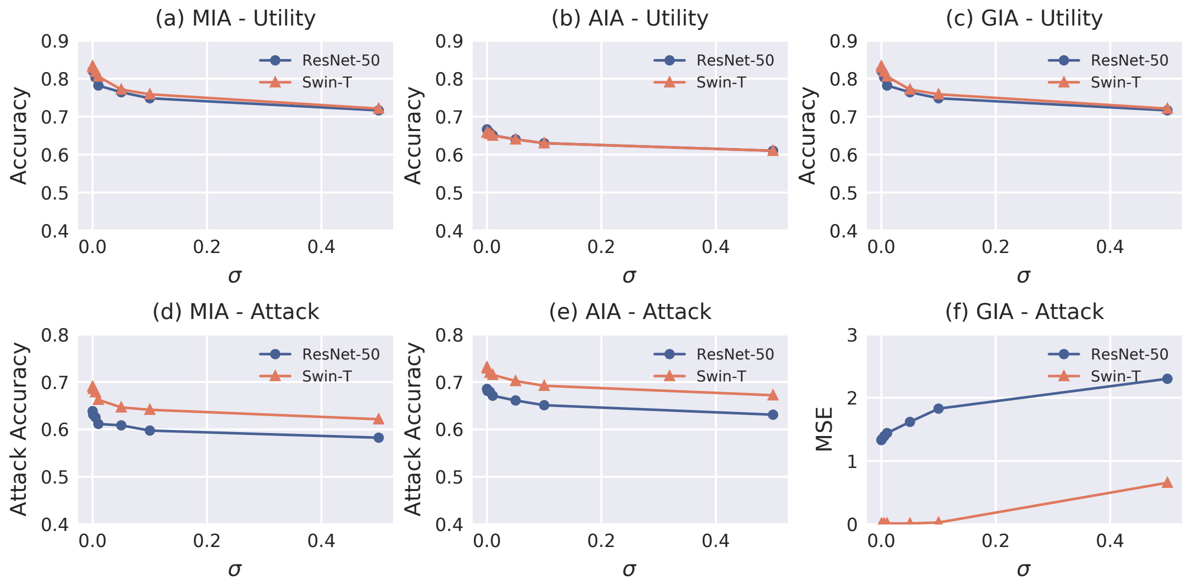


Figure 5.14: The performance of privacy attacks when DPSGD is applied in stem layers. The utility performance of victim models and the attack performances are given when various DP noises are given.

5.8 Conclusion

In this study, we pioneer the exploration of privacy risks on model architectures, especially CNNs and Transformers. We have conducted a comparison of three prominent privacy attacks, i.e., membership inference attacks, attribute inference attacks, and gradient inversion attacks. We discover that Transformers tend to be more vulnerable to privacy attacks than CNNs. As a result, many Transformers-inspired CNN designs, such as ConvNeXt, are also susceptible to privacy threats. A number of Transformers' features, including the design of activation layers, the design of stem layers, the design of LN layers, and the attention modules, are likely to incur high privacy risks.

It is still challenging to establish accurate and theoretical explanations for why certain architectural features are critical to privacy preservation. We believe that these analyses require further experimental campaigns, and we intend to study this matter in our future work.

CHAPTER 6

CONCLUSIONS

Building a trustworthy machine learning system is essential to ensure that the outcomes and decisions generated by these systems are reliable. We specifically focus on the privacy of a machine learning system because trustworthy machine learning systems should be private and secure, with measures in place to protect against potential data leakage or cyber-attacks.

In this thesis, we have analyzed privacy leakage in artificial intelligence in three parts: the model's predictions, gradients, and architectures. Our main contribution can be summarized as follows:

Chapter 3 explores privacy breaches in model predictions by introducing a novel label-only membership inference attack pipeline specifically designed for semantic segmentation models. This work extends membership inference attacks from the domain of image classification to semantic segmentation. We propose the first label-only membership inference attacks targeting these models, designing a comprehensive framework that utilizes various data augmentations and post-processing strategies on the prediction outputs of the victim model. We evaluate the effectiveness of our attack framework under label-only assumptions in semantic segmentation tasks, achieving competitive experimental results. This chapter highlights the susceptibility of semantic segmentation models to membership inference attacks and underscores the importance of robust defense strategies to protect user privacy in practical applications.

Chapter 4 delves into privacy risks associated with model gradients by presenting a privacy-preserving federated learning framework that adapts privacy protection with the need for effective model personalization, specifically countering gradient inversion attacks. Our framework is designed to split the model into global and local layers. This mechanism ensures that a potentially malicious server can only access global gradients, preventing successful attacks. We demonstrate that setting the model head as local layers provides robust privacy protection. Additionally, our framework supports model personalization by allowing each client to train the local layers independently, enhancing the personalization performance through multiple local and fine-tuning epochs. Our extensive experiments indicate that our framework surpasses other defense mechanisms in terms of privacy protection and matches the personalization performance of other federated learning frameworks.

Chapter 5 provides a comprehensive analysis of privacy vulnerabilities in model architectures, focusing on CNNs and Transformers. This chapter investigates the privacy risks associated with these architectures using three prominent attack methods: membership inference attacks, attribute inference attacks, and gradient inversion attacks. Our findings reveal that Transformers are generally more susceptible to these privacy attacks than CNNs. We identify three critical factors contributing to the greater resilience of CNNs: the design of activation layers, stem layers, and LN layers. Moreover, we find that attention modules in Transformers may increase their vulnerability to privacy attacks. We propose modifying model components and introducing perturbations as effective defense mechanisms to mitigate these risks.

Collectively, our research aims to significantly contribute to the broader research community by advancing the understanding of privacy in machine learning. We hope to inspire further investigations through our discoveries, ultimately leading to more robust, secure, and innovative solutions across various fields. Our work addresses current challenges and sets the stage for future advancements.

BIBLIOGRAPHY

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019.

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated Learning with Personalization Layers. *arXiv preprint arXiv:1912.00818*, 2019.

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:2481–2495, 2017.

[6] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are Transformers more robust than CNNs? In *Advances in Neural Information Processing Systems*, volume 34, pages 26831–26843, 2021.

[7] Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev. LAMP: Extracting Text from Gradients with Language Model Priors. *Advances in Neural Information Processing Systems*, 35:7641–7654, 2022.

[8] Mislav Balunovic, Dimitar Iliev Dimitrov, Robin Staab, and Martin Vechev. Bayesian Framework for Gradient Leakage. In *International Conference on Learning Representations*, 2022.

[9] Teodora Baluta, Shiqi Shen, S. Hitarth, Shruti Tople, and Prateek Saxena. Membership Inference Attacks and Generalization: A Causal Perspective. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 249–262, 2022.

[10] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1798–1828, 2013.

[11] Martin Andres Bertran, Shuai Tang, Aaron Roth, Michael Kearns, Jamie Heather Morgenstern, and Steven Wu. Scalable Membership Inference Attacks via Quantile Regression. In *Thirty-Seventh Conference on Neural Information Processing Systems*, 2023.

[12] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[13] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Automatic Clipping: Differentially Private Deep Learning Made Easier and Stronger. *Advances in Neural Information Processing Systems*, 36:41727–41764, 2023.

[14] Sebastien Bubeck, Ronen Eldan, Yin Tat Lee, and Dan Mikulincer. Network size and size of the weights in memorization with two-layers neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 4977–4986, 2020.

[15] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership Inference Attacks From First Principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914, 2022.

[16] Nicholas Carlini, Matthew Jagielski, Chiyuan Zhang, Nicolas Papernot, Andreas Terzis, and Florian Tramer. The Privacy Onion Effect: Memorization is Relative. In *Advances in Neural Information Processing Systems*, 2022.

[17] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models. In *30th USENIX Security Symposium*, 2021.

[18] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators. *Advances in Neural Information Processing Systems*, 33:12673–12684, 2020.

[19] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 343–362, 2020.

[20] Guangke Chen, Yedi Zhang, and Fu Song. SLMIA-SR: Speaker-Level Membership Inference Attacks against Speaker Recognition Systems. In *Proceedings 2024 Network and Distributed System Security Symposium*, 2024.

[21] Hong-You Chen and Wei-Lun Chao. On Bridging Generic and Personalized Federated Learning for Image Classification. In *International Conference on Learning Representations*, 2022.

[22] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L. Yuille. Attention to Scale: Scale-Aware Semantic Image Segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3640–3649, 2016.

[23] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, volume 11211, pages 833–851, 2018.

[24] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When Machine Unlearning Jeopardizes Privacy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 896–911, 2021.

[25] Xiangyi Chen, Steven Z. Wu, and Mingyi Hong. Understanding Gradient Clipping in Private SGD: A Geometric Perspective. *Advances in Neural Information Processing Systems*, 33:13773–13782, 2020.

[26] Yufei Chen, Chao Shen, Yun Shen, Cong Wang, and Yang Zhang. Amplifying Membership Exposure via Data Poisoning. In *Advances in Neural Information Processing Systems*, 2022.

[27] Zitao Chen and Karthik Pattabiraman. Overconfidence is a Dangerous Thing: Mitigating Membership Inference Attacks by Enforcing Less Confident Prediction. In *Proceedings 2024 Network and Distributed System Security Symposium*, 2024.

[28] Christopher A. Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-Only Membership Inference Attacks. In *International Conference on Machine Learning*, pages 1964–1974, 2021.

[29] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and Trainability in Recurrent Neural Networks. In *International Conference on Learning Representations*, 2017.

[30] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting Shared Representations for Personalized Federated Learning. In *Proceedings of the 38th International Conference on Machine Learning*, pages 2089–2099, 2021.

[31] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[32] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[33] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive Personalized Federated Learning. *arXiv preprint arXiv:2003.13461*, 2020.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[35] Mingyu Ding, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. DaViT: Dual Attention Vision Transformers. In *Computer Vision – ECCV 2022*, pages 74–92, 2022.

[36] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021.

[37] Simon S. Du, Jason D. Lee, and Yuandong Tian. When is a Convolutional Filter Easy to Learn? In *International Conference on Learning Representations*, 2018.

[38] Jinhao Duan, Fei Kong, Shiqi Wang, Xiaoshuang Shi, and Kaidi Xu. Are Diffusion Models Vulnerable to Membership Inference Attacks? 2023.

[39] Vasisht Duddu and Antoine Boutet. Inferring Sensitive Attributes from Model Explanations. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 416–425, 2022.

[40] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503, 2006.

[41] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. In *Advances in Neural Information Processing Systems*, volume 33, pages 3557–3568, 2020.

[42] Hao Fang, Bin Chen, Xuan Wang, Zhi Wang, and Shu-Tao Xia. GIFD: A Generative Gradient Inversion Method with Feature Domain Optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4967–4976, 2023.

[43] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[44] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020.

[45] Liam H. Fowl, Jonas Geiping, Wojciech Czaja, Micah Goldblum, and Tom Goldstein. Robbing the Fed: Directly Obtaining Private Data in Federated Learning with Modified Models. In *International Conference on Learning Representations*, 2022.

[46] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[47] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 17–32, 2014.

[48] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual Attention Network for Scene Segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3141–3149, 2019.

[49] Junyao Gao, Xinyang Jiang, Huishuai Zhang, Yifan Yang, Shuguang Dou, Dongsheng Li, Duoqian Miao, Cheng Deng, and Cairong Zhao. Similarity Distribution Based Membership Inference Attack on Person Re-identification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:14820–14828, 2023.

[50] Wei Gao, Shangwei Guo, Tianwei Zhang, Han Qiu, Yonggang Wen, and Yang Liu. Privacy-preserving Collaborative Learning with Automatic Transformation Search. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 114–123, 2021.

[51] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and José García Rodríguez. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, abs/1704.06857, 2017.

[52] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting Gradients - How easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33:16937–16947, 2020.

[53] Robin C. Geyer, Tassilo Klein, and Moin Nabi. Differentially Private Federated Learning: A Client Level Perspective. *arXiv preprint arXiv:1712.07557*, 2018.

[54] Surbhi Goel, Adam Klivans, and Raghu Meka. Learning One Convolutional Layer with Overlapping Patches. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1783–1791, 2018.

[55] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

[56] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[57] Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. Recovering Private Text in Federated Learning of Language Models. *Advances in Neural Information Processing Systems*, 35:8130–8143, 2022.

[58] Filip Hanzely and Peter Richtárik. Federated Learning of a Mixture of Global and Local Models. *arXiv preprint arXiv:2002.05516*, 2021.

[59] Ali Hatamizadeh, Hongxu Yin, Holger R. Roth, Wenqi Li, Jan Kautz, Daguang Xu, and Pavlo Molchanov. GradViT: Gradient Inversion of Vision Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10021–10030, 2022.

[60] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Membership Inference Attacks Against Generative Models. *Proceedings on Privacy Enhancing Technologies*, 2019:133–152, 2019.

[61] Jiyan He, Xuechen Li, Da Yu, Huishuai Zhang, Janardhan Kulkarni, Yin Tat Lee, Arturs Backurs, Nenghai Yu, and Jiang Bian. Exploring the Limits of Differentially Private Deep Learning with Group-wise Clipping. In *The Eleventh International Conference on Learning Representations*, 2023.

[62] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[63] Xinlei He, Zheng Li, Weilin Xu, Cory Cornelius, and Yang Zhang. Membership-Doctor: Comprehensive Assessment of Membership Inference Against Machine Learning Models. *arXiv preprint arXiv:2208.10445*, 2022.

[64] Xinlei He, Hongbin Liu, Neil Zhenqiang Gong, and Yang Zhang. Semi-Leak: Membership Inference Attacks Against Semi-supervised Learning. In *Computer Vision – ECCV 2022*, pages 365–381, 2022.

[65] Xinlei He and Yang Zhang. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 845–863, 2021.

[66] Yang He, Shadi Rahimian, Bernt Schiele, and Mario Fritz. Segmentations-Leak: Membership Inference Attacks and Defenses in Semantic Image Segmentation. In *Computer Vision – ECCV 2020*, pages 519–535, 2020.

[67] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

[68] Dominik Hintersdorf, Lukas Struppek, and Kristian Kersting. To Trust or Not To Trust Prediction Scores for Membership Inference Attacks. In *Thirty-First International Joint Conference on Artificial Intelligence*, volume 4, pages 3043–3049, 2022.

[69] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[70] Hyeong Gwon Hong, Yooshin Cho, Hanbyel Cho, Jaesung Ahn, and Junmo Kim. Foreseeing Reconstruction Quality of Gradient Inversion: An Optimization Perspective. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:12473–12481, 2024.

[71] Andrew Howard, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, Yukun Zhu, Ruoming Pang, Hartwig Adam, and Quoc Le. Searching for MobileNetV3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019.

[72] Hongsheng Hu, Zoran Salcic, Gillian Dobbie, Jinjun Chen, Lichao Sun, and Xuyun Zhang. Membership Inference via Backdooring. In *Thirty-First International Joint Conference on Artificial Intelligence*, volume 5, pages 3832–3838, 2022.

[73] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. Membership Inference Attacks on Machine Learning: A Survey. *ACM Computing Surveys*, 2022.

[74] Li Hu, Anli Yan, Hongyang Yan, Jin Li, Teng Huang, Yingying Zhang, Changyu Dong, and Chunsheng Yang. Defenses to Membership Inference Attacks: A Survey. *ACM Computing Surveys*, 56:92:1–92:34, 2023.

[75] Pingyi Hu, Zihan Wang, Ruoxi Sun, Hu Wang, and Minhui Xue. M^{4}I: Multi-modal Models Membership Inference. *Advances in Neural Information Processing Systems*, 35:1867–1882, 2022.

[76] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep Networks with Stochastic Depth. In *Computer Vision – ECCV 2016*, pages 646–661, 2016.

[77] Hai Huang, Zhikun Zhang, Yun Shen, Michael Backes, Qi Li, and Yang Zhang. On the Privacy Risks of Cell-Based NAS Architectures. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 1427–1441, 2022.

[78] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. Evaluating Gradient Inversion Attacks and Defenses in Federated Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 7232–7241, 2021.

[79] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. Practical blind membership inference attack via differential comparisons. In *28th Annual Network and Distributed System Security Symposium, NDSS 2021, Virtually, February 21-25, 2021*, 2021.

[80] Bargav Jayaraman and David Evans. Are Attribute Inference Attacks Just Imputation? In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 1569–1582, 2022.

[81] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting Membership Inference Under Realistic Assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021:348–368, 2021.

[82] Jinwoo Jeon, jaechang Kim, Kangwook Lee, Sewoong Oh, and Jungseul Ok. Gradient Inversion with Generative Image Prior. In *Advances in Neural Information Processing Systems*, volume 34, pages 29898–29908, 2021.

[83] Jinyuan Jia and Neil Zhenqiang Gong. {AttriGuard}: A Practical Defense Against Attribute Inference Attacks via Adversarial Machine Learning. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 513–529, 2018.

[84] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 259–274, 2019.

[85] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[86] Yigitcan Kaya and Tudor Dumitras. When Does Data Augmentation Help With Membership Inference Attacks? In *International Conference on Machine Learning*, pages 5345–5355, 2021.

[87] Myeongseob Ko, Ming Jin, Chenguang Wang, and Ruoxi Jia. Practical Membership Inference Attacks Against Large-Scale Multi-Modal Models: A Pilot Study. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4871–4881, 2023.

[88] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv preprint arXiv:1610.05492*, 2017.

[89] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[90] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[91] Maximilian Lam, Gu-Yeon Wei, David Brooks, Vijay Janapa Reddi, and Michael Mitzenmacher. Gradient Disaggregation: Breaking Privacy in Federated Learning by Reconstructing the User Participant Matrix. In *Proceedings of the 38th International Conference on Machine Learning*, pages 5959–5968, 2021.

[92] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1:541–551, 1989.

[93] Y. LeCun and Corinna Cortes. The mnist database of handwritten digits. 2005.

[94] Minhyeok Lee. GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance. *arXiv preprint arXiv:2305.12073*, 2023.

[95] Klas Leino and Matt Fredrikson. Stolen Memories: Leveraging Model Memorization for Calibrated White-Box Membership Inference. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[96] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proceedings of the 38th International Conference on Machine Learning*, pages 6357–6368, 2021.

[97] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. MViTv2: Improved Multiscale Vision Transformers for Classification and Detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4794–4804, 2022.

[98] Zhaohua Li, Le Wang, Zhaoquan Gu, Yang Lv, and Zhihong Tian. Labels are Culprits: Defending Gradient Attack On Privacy. *IEEE Internet of Things Journal*, pages 1–1, 2023.

[99] Zheng Li, Yiyong Liu, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. Auditing Membership Leakages of Multi-Exit Networks. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 1917–1931, 2022.

[100] Zheng Li and Yang Zhang. Membership Leakage in Label-Only Exposures. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2021)*, 2021.

[101] Zhize Li, Haoyu Zhao, Boyue Li, and Yuejie Chi. SoteriaFL: A Unified Framework for Private Federated Learning with Communication Compression. *Advances in Neural Information Processing Systems*, 35:4285–4300, 2022.

[102] Zhuohang Li, Jiaxin Zhang, Luyang Liu, and Jian Liu. Auditing Privacy Defenses in Federated Learning via Generative Gradient Leakage. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10132–10142, 2022.

[103] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B. Allen, Randy P. Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think Locally, Act Globally: Federated Learning with Local and Global Representations. *arXiv preprint arXiv:2001.01523*, 2020.

[104] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When Machine Learning Meets Privacy: A Survey and Outlook. *ACM Computing Surveys*, 54:31:1–31:36, 2021.

[105] Han Liu, Yuhao Wu, Zhiyuan Yu, and Ning Zhang. Please Tell Me More: Privacy Impact of Explainability through the Lens of Membership Inference Attack. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 120–120, 2024.

[106] Hongbin Liu, Jinyuan Jia, Wenjie Qu, and Neil Zhenqiang Gong. EncoderMI: Membership Inference against Pre-trained Encoders in Contrastive Learning. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2081–2095, 2021.

[107] Yiyong Liu, Zhengyu Zhao, Michael Backes, and Yang Zhang. Membership Inference Attacks by Exploiting Loss Trajectory. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2085–2098, 2022.

[108] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, Michael Backes, Emiliano De Cristofaro, Mario Fritz, and Yang Zhang. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In *31st USENIX Security Symposium (USENIX Security 22)*, 2022.

[109] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021.

[110] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *2022 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[111] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

[112] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3431–3440, 2015.

[113] Yunhui Long, Lei Wang, Diyue Bu, Vincent Bindschaedler, Xiaofeng Wang, Haixu Tang, Carl A. Gunter, and Kai Chen. A Pragmatic Approach to Membership Inferences on Machine Learning Models. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 521–534, 2020.

[114] Jiahao Lu, Xi Sheryl Zhang, Tianli Zhao, Xiangyu He, and Jian Cheng. APRIL: Finding the Achilles' Heel on Privacy for Vision Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10051–10060, 2022.

[115] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[116] Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120:233–255, 2016.

[117] Mohammad Malekzadeh, Anastasia Borovykh, and Deniz Gündüz. Honest-but-Curious Nets: Sensitive Attributes of Private Inputs Can Be Secretly Coded into the Classifiers' Outputs. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 825–844, 2021.

[118] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[119] Shagufta Mehnaz, Sayanton V. Dibbo, Ehsanul Kabir, Ninghui Li, and Elisa Bertino. Are Your Sensitive Attributes Private? Novel Model Inversion Attribute Inference Attacks on Classification Models. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4579–4596, 2022.

[120] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706, 2019.

[121] Shervin Minaee, Yuri Y. Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[122] Payman Mohassel and Yupeng Zhang. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 19–38, 2017.

[123] Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable Extraction of Training Data from (Production) Language Models. *arXiv preprint arXiv:2311.17035*, 2023.

[124] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine Learning with Membership Privacy using Adversarial Regularization. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 634–646, 2018.

[125] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 739–753, 2019.

[126] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5000–5009, 2017.

[127] Jaehoon Oh, SangMook Kim, and Se-Young Yun. FedBABU: Toward Enhanced Representation for Federated Image Classification. In *International Conference on Learning Representations*, 2022.

[128] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[129] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with PATE. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[130] Yeachan Park and Myungjoo Kang. Membership Inference Attacks Against Object Detection Models. *arXiv preprint arXiv:2001.04011*, 2020.

[131] Sayak Paul and Pin-Yu Chen. Vision Transformers Are Robust Learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:2071–2081, 2022.

[132] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollar. Designing Network Design Spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10428–10436, 2020.

[133] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do Vision Transformers See Like Convolutional Neural Networks? In *Advances in Neural Information Processing Systems*, volume 34, pages 12116–12128, 2021.

[134] Md. Atiqur Rahman, Tanzila Rahman, Robert Laganière, and Noman Mohammed. Membership inference attack against differentially private deep learning model. *Trans. Data Priv.*, 11:61–79, 2018.

[135] Shashank Rajput, Kartik Sreenivasan, Dimitris Papailiopoulos, and Amin Karbasi. An Exponential Improvement on the Memorization Capacity of Deep Threshold Networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 12674–12685, 2021.

[136] Simon Reiss, Constantin Seibold, Alexander Freytag, Erik Rodner, and Rainer Stiefelhagen. Every annotation counts: Multi-label deep supervision for medical

image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9532–9542, 2021.

[137] Shahbaz Rezaei and Xin Liu. On the Difficulty of Membership Inference Attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7892–7900, 2021.

[138] Shahbaz Rezaei, Zubair Shafiq, and Xin Liu. Accuracy-Privacy Trade-off in Deep Ensemble: A Membership Inference Perspective. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 364–381, 2023.

[139] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Herve Jegou. White-box vs Black-box: Bayes Optimal Strategies for Membership Inference. In *International Conference on Machine Learning*, pages 5558–5567, 2019.

[140] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-Leak: Data Set Inference and Reconstruction Attacks in Online Learning. In *29th USENIX Security Symposium (USENIX Security 20)*, page 1291, 2020.

[141] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. ML-Leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019*, 2019.

[142] Avital Shafran, Shmuel Peleg, and Yedid Hoshen. Membership Inference Attacks Are Easier on Difficult Problems. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14820–14829, 2021.

[143] Virat Shejwalkar and Amir Houmansadr. Membership Privacy for Machine Learning Models Through Knowledge Transfer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:9549–9557, 2021.

[144] Reza Shokri and Vitaly Shmatikov. Privacy-Preserving Deep Learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1310–1321, 2015.

[145] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2017.

[146] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *J. Big Data*, 6:60, 2019.

[147] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2015.

[148] Congzheng Song and Ananth Raghunathan. Information Leakage in Embedding Models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390, 2020.

[149] Congzheng Song and Vitaly Shmatikov. Overlearning Reveals Sensitive Attributes. In *International Conference on Learning Representations*, 2020.

[150] Liwei Song and Prateek Mittal. Systematic Evaluation of Privacy Risks of Machine Learning Models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632, 2021.

[151] Liwei Song, Reza Shokri, and Prateek Mittal. Membership Inference Attacks Against Adversarially Robust Deep Learning Models. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 50–56, 2019.

[152] Liwei Song, Reza Shokri, and Prateek Mittal. Privacy Risks of Securing Machine Learning Models against Adversarial Examples. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 241–257, 2019.

[153] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your ViT? Data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022.

[154] Jingwei Sun, Ang Li, Binghui Wang, Huanrui Yang, Hai Li, and Yiran Chen. Soteria: Provable Defense Against Privacy Leakage in Federated Learning From Representation Perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9311–9319, 2021.

[155] Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving LoRA in Privacy-preserving Federated Learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[156] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[157] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[158] Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6105–6114, 2019.

[159] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. FedProto: Federated Prototype Learning across Heterogeneous Clients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:8432–8440, 2022.

[160] Xinyu Tang, Saeed Mahloujifar, Liwei Song, Virat Shejwalkar, Milad Nasr, Amir Houmansadr, and Prateek Mittal. Mitigating Membership Inference Attacks by {Self-Distillation} Through a Novel Ensemble Architecture. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1433–1450, 2022.

[161] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. DP-CGAN: Differentially Private Synthetic Data and Label Generation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 98–104, 2019.

[162] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, pages 10347–10357, 2021.

[163] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with Image Transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 32–42, 2021.

[164] Florian Tramer and Dan Boneh. Differentially Private Learning Needs Better Features (or Much More Data). In *International Conference on Learning Representations*, 2020.

[165] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A Hybrid Approach to Privacy-Preserving Federated Learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 1–11, 2019.

[166] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying Membership Inference Attacks in Machine Learning as a Service. *IEEE Transactions on Services Computing*, pages 1–1, 2019.

[167] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[168] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[169] Mark Vero, Mislav Balunovic, Dimitar Iliev Dimitrov, and Martin Vechev. TabLeak: Tabular Data Leakage in Federated Learning. In *Proceedings of the 40th International Conference on Machine Learning*, pages 35051–35083, 2023.

[170] Ji Wang, Weidong Bao, Lichao Sun, Xiaomin Zhu, Bokai Cao, and Philip S. Yu. Private Model Compression via Knowledge Distillation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1190–1197, 2019.

[171] Zeyu Wang, Yutong Bai, Yuyin Zhou, and Cihang Xie. Can CNNs Be More Robust Than Transformers? In *The Eleventh International Conference on Learning Representations*, 2023.

[172] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004.

[173] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the Importance of Difficulty Calibration in Membership Inference Attacks. In *International Conference on Learning Representations*, 2022.

[174] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. Federated Learning With Differential Privacy: Algorithms and Performance Analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[175] Wenqi Wei and Ling Liu. Gradient Leakage Attack Resilient Deep Learning. *IEEE Transactions on Information Forensics and Security*, 17:303–316, 2022.

[176] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A Framework for Evaluating Client Privacy Leakages in Federated Learning. In *Computer Security – ESORICS 2020*, pages 545–566, 2020.

[177] Yuxin Wen, Jonas A. Geiping, Liam Fowl, Micah Goldblum, and Tom Goldstein. Fishing for User Data in Large-Batch Federated Learning via Gradient Magnification. In *Proceedings of the 39th International Conference on Machine Learning*, pages 23668–23684, 2022.

[178] Jing Wu, Munawar Hayat, Mingyi Zhou, and Mehrtash Harandi. Concealing Sensitive Samples against Gradient Leakage in Federated Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:21717–21725, 2024.

[179] Tong Wu, Ashwinee Panda, Jiachen T. Wang, and Prateek Mittal. Privacy-Preserving In-Context Learning for Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024.

[180] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified Perceptual Parsing for Scene Understanding. In *Computer Vision – ECCV 2018*, pages 432–448, 2018.

[181] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollar, and Ross Girshick. Early Convolutions Help Transformers See Better. In *Advances in Neural Information Processing Systems*, volume 34, pages 30392–30400, 2021.

[182] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially Private Generative Adversarial Network. *arXiv preprint arXiv:1802.06739*, 2018.

[183] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[184] Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and Improving Layer Normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[185] Lulu Xue, Shengshan Hu, Ruizhi Zhao, Leo Yu Zhang, Shengqing Hu, Lichao Sun, and Dezhong Yao. Revisiting Gradient Pruning: A Dual Realization for Defending against Gradient Attacks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:6404–6412, 2024.

[186] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[187] Ziqi Yang, Bin Shao, Bohan Xuan, Ee-Chien Chang, and Fan Zhang. Defending Model Inversion and Membership Inference Attacks via Prediction Purification. *arXiv preprint arXiv:2005.03915*, 2020.

[188] Ziqi Yang, Lijin Wang, Da Yang, Jie Wan, Ziming Zhao, Ee-Chien Chang, Fan Zhang, and Kui Ren. Purifier: Defending Data Inference Attacks via Transforming Confidence Scores. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:10871–10879, 2023.

[189] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced Membership Inference Attacks against Machine Learning Models. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 3093–3106, 2022.

[190] Zipeng Ye, Wenjian Luo, Qi Zhou, and Yubo Tang. High-Fidelity Gradient Inversion in Distributed Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38:19983–19991, 2024.

[191] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282, 2018.

[192] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. See Through Gradients: Image Batch Recovery via GradInversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16337–16346, 2021.

[193] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. How Does Data Augmentation Affect Privacy in Machine Learning? *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:10746–10753, 2021.

[194] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2633–2642, 2020.

[195] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially Private Model Publishing for Deep Learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 332–349, 2019.

[196] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 538–547, 2021.

[197] Xiaoyong Yuan and Lan Zhang. Membership Inference Attacks and Defenses in Neural Network Pruning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4561–4578, 2022.

[198] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *Computer Vision – ECCV 2014*, pages 818–833, 2014.

[199] Chi Zhang, Sotthiwat Ekanut, Liangli Zhen, and Zengxiang Li. Augmented Multi-Party Computation Against Gradient Leakage in Federated Learning. *IEEE Transactions on Big Data*, pages 1–10, 2022.

[200] Chi Zhang, Zhang Xiaoman, Ekanut Sotthiwat, Yanyu Xu, Ping Liu, Liangli Zhen, and Yong Liu. Generative Gradient Inversion via Over-Parameterized Networks in Federated Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5126–5135, 2023.

[201] Guangsheng Zhang, Bo Liu, Huan Tian, Tianqing Zhu, Ming Ding, and Wanlei Zhou. How Does a Deep Learning Model Architecture Impact Its Privacy? A Comprehensive Study of Privacy Attacks on CNNs and Transformers. *arXiv preprint arXiv:2210.11049*, 2024.

[202] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Ming Ding, and Wanlei Zhou. Label-Only Membership Inference Attacks and Defenses In Semantic Segmentation Models. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2022.

[203] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Ming Ding, and Wanlei Zhou. PPFed: A Privacy-Preserving and Personalized Federated Learning Framework. *IEEE Internet of Things Journal*, pages 1–1, 2024.

[204] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Andi Zhou, and Wanlei Zhou. Visual privacy attacks and defenses in deep learning: A survey. *Artificial Intelligence Review*, 55:4347–4401, 2022.

[205] Haobo Zhang, Junyuan Hong, Yuyang Deng, Mehrdad Mahdavi, and Jiayu Zhou. Understanding Deep Gradient Leakage via Inversion Influence Functions. *Advances in Neural Information Processing Systems*, 36:3921–3944, 2023.

[206] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[207] Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhunmin Chen, Pengfei Hu, and Yang Zhang. Membership Inference Attacks Against Recommender Systems. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 864–879, 2021.

[208] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[209] Rui Zhang, Song Guo, Junxiao Wang, Xin Xie, and Dacheng Tao. A Survey on Gradient Inversion: Attacks, Defenses and Future Directions. In *Thirty-First*

International Joint Conference on Artificial Intelligence, volume 6, pages 5678–5685, 2022.

[210] Xinwei Zhang, Zhiqi Bu, Steven Wu, and Mingyi Hong. Differentially Private SGD Without Clipping Bias: An Error-Feedback Approach. In *The Twelfth International Conference on Learning Representations*, 2024.

[211] Yuxiao Zhang, Haiqiang Chen, Yiran He, Mao Ye, Xi Cai, and Dan Zhang. Road segmentation for all-day outdoor robot navigation. *Neurocomputing*, 314:316–325, 2018.

[212] Ziqi Zhang, Chen Gong, Yifeng Cai, Yuanyuan Yuan, Bingyan Liu, Ding Li, Yao Guo, and Xiangqun Chen. No Privacy Left Outside: On the (In-)Security of TEE-Shielded DNN Partition for On-Device ML. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 51–51, 2023.

[213] Benjamin Zi Hao Zhao, Aviral Agrawal, Catisha Coburn, Hassan Jameel Asghar, Raghav Bhaskar, Mohamed Ali Kaafar, Darren Webb, and Peter Dickinson. On the (In)Feasibility of Attribute Inference Attacks on Machine Learning Models. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 232–251, 2021.

[214] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. iDLG: Improved Deep Leakage from Gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[215] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6230–6239, 2017.

[216] Lingchen Zhao, Qian Wang, Qin Zou, Yan Zhang, and Yanjiao Chen. Privacy-Preserving Collaborative Deep Learning With Unreliable Participants. *IEEE Transactions on Information Forensics and Security*, 15:1486–1500, 2020.

[217] Junyi Zhu and Matthew B. Blaschko. R-GAP: Recursive Gradient Attack on Privacy. In *International Conference on Learning Representations*, 2020.

[218] Junyi Zhu, Ruicong Yao, and Matthew B. Blaschko. Surrogate Model Extension (SME): A Fast and Accurate Weight Update Attack on Federated Learning. In

Proceedings of the 40th International Conference on Machine Learning, pages 43228–43257, 2023.

[219] Ligeng Zhu, Zhijian Liu, and Song Han. Deep Leakage from Gradients. *Advances in Neural Information Processing Systems*, 32, 2019.

[220] Linghui Zhu, Xinyi Liu, Yiming Li, Xue Yang, Shu-Tao Xia, and Rongxing Lu. A Fine-grained Differentially Private Federated Learning against Leakage from Gradients. *IEEE Internet of Things Journal*, pages 1–1, 2021.

[221] Yuqing Zhu, Xiang Yu, Manmohan Chandraker, and Yu-Xiang Wang. Private-kNN: Practical Differential Privacy for Computer Vision. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11851–11859, 2020.

[222] Yang Zou, Zhikun Zhang, Michael Backes, and Yang Zhang. Privacy Analysis of Deep Learning in the Wild: Membership Inference Attacks against Transfer Learning. *arXiv preprint arXiv:2009.04872*, 2020.