Saarland University

Department of Computer Science

Privacy Risk Assessment of Emerging Machine Learning Paradigms

Dissertation
zur Erlangung des Grades
des Doktors der Ingenieurwissenschaften
der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

von
Xinlei He

Saarbrücken, 2023

Tag des Kolloquiums:	16 August 2023
Dekan:	Prof. Dr. Jürgen Steimle

Prüfungsausschuss:

Vorsitzender:	Prof. Dr. Krishna P. Gummadi
Berichterstattende:	Dr. Yang Zhang
	Prof. Dr. Neil Zhenqiang Gong
	Prof. Dr. Isabel Valera
Akademischer Mitarbeiter:	Dr. Zhengyu Zhao

Zusammenfassung

Maschinelles Lernen (ML) hat enorme Fortschritte gemacht, und Daten sind der Schlüsselfaktor, um diese Entwicklung voranzutreiben. Es gibt jedoch zwei große Herausforderungen bei der Erfassung der Daten und deren Handhabung mit ML-Modellen. Erstens kann die Erfassung qualitativ hochwertiger beschrifteter Daten aufgrund der Notwendigkeit umfangreicher menschlicher Anmerkungen schwierig und teuer sein. Zweitens wurden Graphen genutzt, um die komplexe Beziehung zwischen Entitäten, z. B. sozialen Netzwerken oder Molekülstrukturen, zu modellieren. Herkömmliche ML-Modelle können Diagrammdaten jedoch aufgrund der nichtlinearen und komplexen Natur der Beziehungen zwischen Knoten möglicherweise nicht effektiv handhaben. Um diesen Herausforderungen zu begegnen, wurden jüngste Entwicklungen im halbüberwachten Lernen und im selbstüberwachten Lernen eingeführt, um unbeschriftete Daten für ML-Aufgaben zu nutzen. Darüber hinaus wurde eine neue Familie von ML-Modellen, bekannt als Graph Neural Networks, vorgeschlagen, um die Herausforderungen im Zusammenhang mit Graphdaten zu bewältigen. Obwohl sie leistungsfähig sind, sollte auch das potenzielle Datenschutzrisiko berücksichtigt werden, das sich aus diesen Paradigmen ergibt.

In dieser Dissertation führen wir die Datenschutzrisikobewertung der aufkommenden Paradigmen des maschinellen Lernens durch. Erstens untersuchen wir die Datenschutzlecks der Mitgliedschaft, die sich aus halbüberwachtem Lernen ergeben. Konkret schlagen wir den ersten auf Datenaugmentation basierenden Mitgliedschafts-Inferenz-Angriff vor, der auf das Trainingsparadigma halbüberwachter Lernmethoden zugeschnitten ist. Zweitens quantifizieren wir das Durchsickern der Privatsphäre des selbstüberwachten Lernens durch die Linse von Mitgliedschafts-Inferenz-Angriffen und Attribut-Inferenz-Angriffen. Drittens untersuchen wir die Datenschutzauswirkungen des Trainings von GNNs auf Graphen. Insbesondere schlagen wir den ersten Angriff vor, um einen Graphen aus den Ausgaben eines GNN-Modells zu stehlen, das auf dem Graphen trainiert wird. Schließlich untersuchen wir auch mögliche Verteidigungsmechanismen, um diese Angriffe abzuschwächen.

Abstract

Machine learning (ML) has progressed tremendously, and data is the key factor to drive such development. However, there are two main challenges regarding collecting the data and handling it with ML models. First, the acquisition of high-quality labeled data can be difficult and expensive due to the need for extensive human annotation. Second, to model the complex relationship between entities, e.g., social networks or molecule structures, graphs have been leveraged. However, conventional ML models may not effectively handle graph data due to the non-linear and complex nature of the relationships between nodes. To address these challenges, recent developments in semi-supervised learning and self-supervised learning have been introduced to leverage unlabeled data for ML tasks. In addition, a new family of ML models known as graph neural networks has been proposed to tackle the challenges associated with graph data. Despite being powerful, the potential privacy risk stemming from these paradigms should also be taken into account.

In this dissertation, we perform the privacy risk assessment of the emerging machine learning paradigms. Firstly, we investigate the membership privacy leakage stemming from semi-supervised learning. Concretely, we propose the first data augmentation-based membership inference attack that is tailored to the training paradigm of semi-supervised learning methods. Secondly, we quantify the privacy leakage of self-supervised learning through the lens of membership inference attacks and attribute inference attacks. Thirdly, we study the privacy implications of training GNNs on graphs. In particular, we propose the first attack to steal a graph from the outputs of a GNN model that is trained on the graph. Finally, we also explore potential defense mechanisms to mitigate these attacks.

Background of this Dissertation

This dissertation is based on the following mentioned papers. I contributed to all papers as the main author.

The initial idea of conducting membership inference attacks against semi-supervised learning [P1] originated from a joint discussion between Xinlei He and Yang Zhang. Xinlei He proposed data augmentation-based attacks to enhance the attack performance. Xinlei He was then responsible for the implementation and evaluation. All authors participated in the discussion, writing, and reviewing of the paper.

The idea of quantifying the privacy leakage of contrastive learning [P2] was discussed by Xinlei He and Yang Zhang. Xinlei He and Yang Zhang later jointly designed the defense mechanism. The implementation and evaluation were done by Xinlei He. All authors participated in the discussion, writing, and reviewing of the paper.

The general idea of stealing links from graph neural networks [P3] was the contribution of Yang Zhang. Different attacks are designed by Xinlei He. The implementation and evaluation were carried out by Xinlei He. All authors participated in the discussion, writing, and reviewing of the paper.

[P1] He, X., Liu, H., Gong, N. Z., and Zhang, Y. Semi-Leak: Membership Inference Attacks Against Semi-supervised Learning. In: *European Conference on Computer Vision (ECCV)*. Springer, 2022, 365–381.

[P2] He, X. and Zhang, Y. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2021, 845–863.

[P3] He, X., Jia, J., Backes, M., Gong, N. Z., and Zhang, Y. Stealing Links from Graph Neural Networks. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2021, 2669–2686.

Further Contributions of the Author

The author was also able to contribute to the following:

Published Papers:

[S1] Cong, T., He, X., and Zhang, Y. SSLGuard: A Watermarking Scheme for Self-supervised Learning Pre-trained Encoders. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022, 579–593.

[S2] Li, Z., Liu, Y., He, X., Yu, N., Backes, M., and Zhang, Y. Auditing Membership Leakages of Multi-Exit Networks. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022, 1917–1931.

[S3] Liu, Y., Wen, R., He, X., Salem, A., Zhang, Z., Backes, M., Cristofaro, E. D., Fritz, M., and Zhang, Y. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2022, 4525–4542.

[S4] Ma, Y., Zhang, Z., Yu, N., He, X., Backes, M., Shen, Y., and Zhang, Y. Generated Graph Detection. In: *International Conference on Machine Learning (ICML)*. PMLR, 2023.

[S5] Qu, Y., He, X., Pierson, S., Backes, M., Zhang, Y., and Zannettou, S. On the Evolution of (Hateful) Memes by Means of Multimodal Contrastive Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2023.

[S6] Qu, Y., Shen, X., He, X., Backes, M., Zannettou, S., and Zhang, Y. Unsafe Diffusion: On the Generation of Unsafe Images and Hateful Memes From Text-To-Image Models. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2023.

[S7] Sha, Z., He, X., Yu, N., Backes, M., and Zhang, Y. Can't Steal? Cont-Steal! Contrastive Stealing Attacks Against Image Encoders. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023.

[S8] Shen, X., He, X., Backes, M., Blackburn, J., Zannettou, S., and Zhang, Y. On Xing Tian and the Perseverance of Anti-China Sentiment Online. In: *International Conference on Web and Social Media (ICWSM)*. AAAI, 2022, 944–955.

[S9] Shen, Y., He, X., Han, Y., and Zhang, Y. Model Stealing Attacks Against Inductive Graph Neural Networks. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2022, 1175–1192 (The first two authors made equal contribution).

[S10] Yang, Z., He, X., Li, Z., Backes, M., Humbert, M., Berrang, P., and Zhang, Y. Data Poisoning Attacks Against Multimodal Encoders. In: *International Conference on Machine Learning (ICML)*. PMLR, 2023.

[S11] Zhang, B., He, X., Shen, Y., Wang, T., and Zhang, Y. A Plot is Worth a Thousand Words: Model Information Stealing Attacks via Scientific Plots. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2023.

Technical Reports:

[T1] He, X., Li, Z., Xu, W., Cornelius, C., and Zhang, Y. Membership-Doctor: Comprehensive Assessment of Membership Inference Against Machine Learning Models. *CoRR abs/2208.10445* (2022).

[T2] He, X., Wen, R., Wu, Y., Backes, M., Shen, Y., and Zhang, Y. Node-Level Membership Inference Attacks Against Graph Neural Networks. *CoRR abs/2102.05429* (2021).

[T3] Sha, Z., He, X., Berrang, P., Humbert, M., and Zhang, Y. Fine-Tuning Is All You Need to Mitigate Backdoor Attacks. *CoRR abs/2212.09067* (2022).

Acknowledgments

First and foremost, I would like to express my deepest appreciation to my advisor Yang Zhang for his dedicated support and guidance. Yang continuously provides encouragement and is always willing and enthusiastic to assist in any way he could from the start of my Ph.D. till its end. Yang has a good sense of humor and he is also one of the smartest people I know. I really enjoy working with Yang and I hope that I could be as lively, enthusiastic, and energetic as him in the future.

I am also deeply indebted to my dissertation committee members: Neil Zhenqiang Gong, Isabel Valera, and Yang Zhang, for their effort and time in reviewing this thesis and the invaluable feedback.

I could not have undertaken this journey without all my collaborators and co-authors that I was lucky enough to work with. Special thanks go to Yang, Neil, Savvas, and Yun. Thank you all for the very insightful discussions and valuable suggestions.

Naturally, thanks should also go to my colleagues and friends inside and outside CISPA! Special thanks go to Yugeng, Zeyang, Tianshuo, Rui, Zheng, Minxing, and many more!

Most importantly, I would be remiss in not mentioning my family, especially my parents for their unlimited support for every decision I made. The same applies to my wife, Yuanyuan, for her love and constant support all the time. I owe you everything!

Lastly, for anyone whom I forgot to mention here due to my memory, I am deeply grateful for the support and assistance from you along this challenging journey. Thank you all from the bottom of my heart!

Contents

List of Figures

List of Tables

1 Introduction

Machine learning (ML) has progressed tremendously, and data is the key factor to drive such development. However, there are two main challenges regarding collecting the data and handling it with ML models. First, high-quality data, in particular labeled data, is often hard and expensive to collect as this relies on large-scale human annotation. Second, to model the complex relationship between entities, e.g., social networks or molecule structures, graphs have been leveraged. However, conventional ML models fall short of handling graph data properly as the relationships between the nodes are non-linear and complex.

To address the first challenge, researchers are exploring the use of unlabeled data for ML tasks, as unlabeled data is being generated at every moment. To this end, *semi-supervised learning* [98, 117, 121] and *self-supervised learning* [17, 48, 18] have been introduced. For semi-supervised learning, a small set of labeled data is used to train the ML model alongside a much larger set of unlabeled data. For self-supervised learning, it aims to learn useful features or representations from the input data itself, without any explicit labeling or supervision. By leveraging unlabeled data, the model can learn better representations, leading to improved performance.

To address the second challenge, a new family of ML models known as *graph neural networks* [61, 41, 130, 65, 128, 115] has been proposed. Unlike traditional neural networks that operate on fixed-size inputs, graph neural networks can handle graphs with different numbers of nodes and edges. A graph neural network typically works by passing messages between nodes in the graph, where each node aggregates messages from its neighboring nodes to update its own representation. By learning the structural and relational information present in the graph, graph neural networks show remarkable performance on various tasks such as node classification, link prediction, and graph classification.

Despite being powerful, ML models have been demonstrated to be vulnerable to various privacy attacks, represented by membership inference attacks [97, 94], attribute inference attacks [76, 102], and reconstruction attacks [93, 125]. For membership inference attacks, the adversary aims to infer whether a data sample is part of a target ML model's training dataset. For attribute inference attacks, the adversary leverages the overlearning property of a machine learning model to infer the sensitive attribute of a data sample. Regarding reconstruction attacks, the adversary aims to rebuild the original data by exploiting vulnerabilities in the ML model's output (e.g., prediction probability or embeddings). So far, most of the efforts on privacy attacks concentrate on models trained by supervised learning and data with conventional forms such as images and texts. However, as the emerging ML paradigms (i.e., semi-supervised learning, self-supervised learning, and graph neural networks) handle the training data in their unique ways, it is important to quantify whether such unique training paradigms would lead to more severe privacy leakage of the training data.

1.1 Our Contributions

In this dissertation, we take the first step toward quantifying the privacy risks stemming from the emerging ML paradigms. Concretely, we focus on semi-supervised learning, self-supervised learning, and graph neural networks. We first explore the possible privacy threats against each paradigm. Then, we develop defense mechanisms to mitigate the potential risks. Our work is based on the following peer-reviewed publications [P1, P2, P3], each exploring the privacy threats of an ML paradigm from a different angle.

Membership Inference Attacks Against Semi-supervised Learning: We start with [P1] by investigating the privacy risks stemming from semi-supervised learning (SSL). In this work, we focus on membership inference attacks (MIA), which is one of the most severe privacy attacks against ML models. As SSL treats labeled and unlabeled data differently during the training phase, we are curious whether this training paradigm would cause different membership leakage to the data samples. Concretely, we propose the first data augmentation-based membership inference attack that is tailored to SSL methods. Our evaluation shows that the proposed attack can consistently outperform existing membership inference attacks and achieves the best performance against the model trained by SSL. Moreover, we uncover that the reason for membership leakage in SSL is different from the commonly believed one in supervised learning, i.e., overfitting (the gap between training and testing accuracy). We observe that the SSL model is well generalized to the testing data (with almost 0 overfitting) but "memorizes" the training data by giving a more confident prediction regardless of its correctness. We also explore possible countermeasures and find that early stopping achieves the best trade-off between model utility and membership inference performance.

Quantifying and Mitigating The Privacy Risk of Self-supervised Learning: In our second work [P2], we perform the first privacy analysis of contrastive learning, one of the most representative self-supervised learning mechanisms, through the lens of membership inference and attribute inference. Our experimental results show that contrastive models are less vulnerable to membership inference attacks but more vulnerable to attribute inference attacks compared to supervised models. The former is due to the fact that contrastive models are less prone to overfitting, while the latter is caused by contrastive models' capability of representing data samples expressively. To remedy this situation, we propose the first privacy-preserving contrastive learning mechanism, namely *Talos*, relying on adversarial training. Empirical results show that *Talos* can successfully mitigate attribute inference risks for contrastive models while maintaining their membership privacy and model utility.

Link Stealing Attacks Against Graph Neural Network: In our last work [P3], we focus on graph data. In this work, we propose the first attack to steal a graph from the outputs of a GNN model that is trained on the graph. Specifically, given the black-box access to a GNN model, our attacks can infer whether there exists a link between any pair of nodes in the graph used to train the model. We call our attacks *link stealing attacks*. We propose a threat model to systematically characterize an adversary's background knowledge along three dimensions which in total leads to a comprehensive taxonomy of 8 different link stealing attacks. We propose multiple novel methods to realize these 8 attacks. Extensive experiments on 8 real-world datasets show that our attacks are effective at stealing links, e.g., AUC (area under the ROC curve) is above 0.95 in multiple cases. Our results indicate that the outputs of a GNN model reveal rich information about the structure of the graph used to train the model.

1.2 Organization

The rest of this dissertation is organized as follows. We first present the preliminaries and background in Chapter 2. In Chapter 3, we explore the potential membership leakage from the models trained with semi-supervised learning. Next, we quantify the privacy leakage of self-supervised learning through the lens of membership inference and attribute inference in Chapter 4. We then explore a unique attack, namely link stealing attacks stemming from GNN models in Chapter 5. Finally, we present the related work in Chapter 6 and conclude the dissertation in Chapter 7. Kindly be informed that the material showcased in Chapter 3 to Chapter 5 is based on the original publications [P1, P2, P3]. Therefore, there may be slight variations in the terminology and notation used in different chapters. Moreover, the introduction of different chapters may exhibit similarities in their argumentation and covered material due to the same reason.

2
Preliminaries and Background

2.1 Supervised Learning

Supervised learning, represented by classification, is one of the most common and important ML applications. We first denote a set of data samples by X and a set of labels by Y. The objective of a supervised ML model $\mathcal{M}$ is to learn a mapping function from each data sample $x \in X$ to its label/class $y \in Y$. Formally, we have

$$\mathcal{M} : x \mapsto y \tag{2.1}$$

Given a sample x, its output from $\mathcal{M}$, denoted by $p = \mathcal{M}(x)$, is a vector that represents the probability distribution of the sample belonging to a certain class. In Chapter 4, we refer to p as the prediction posteriors. To train an ML model, we need to define a loss function $\mathcal{L}(y, \mathcal{M}(x))$ which measures the distance between a sample's prediction posteriors and its label. The training process is then performed by minimizing the expectation of the loss function over a training dataset $\mathcal{D}^{train}$, i.e., the empirical loss. We formulate this as follow:

$$\arg\min_{\mathcal{M}} \frac{1}{|\mathcal{D}^{train}|} \sum_{(x,y)\in\mathcal{D}^{train}} \mathcal{L}(y, \mathcal{M}(x)) \tag{2.2}$$

Cross-entropy loss is one of the most common loss functions used for classification tasks, it is defined as the following.

$$\mathcal{L}_{CE}(y, p) = -\sum_{i=1}^{k} y^i \log p^i \tag{2.3}$$

Here, k is the total number of classes, y^i equals to 1 if the sample belongs to class i (otherwise 0), and p^i is the i-th element of the posteriors p. In this dissertation, we use cross-entropy as the loss function to train all the supervised models.

2.2 Semi-Supervised Learning

Semi-supervised learning (SSL) [64, 77, 8, 98, 117, 121] aims to train accurate models via exploiting a large amount of unlabeled data when the labeled data is scarce. In Chapter 3, we focus on the vision domain since most advanced SSL methods are designed for it. Generally speaking, state-of-the-art SSL techniques [98, 117, 121] produce "pseudo labels" for the unlabeled samples when the model's predictions are confident enough based on pre-defined threshold strategies. For example, Lee [64] first proposed to produce the class label that has the highest confidence score output by the classifier for unlabeled samples during training. After assigning pseudo labels to unlabeled samples, they can train classifiers in a supervised fashion with labeled and unlabeled samples. Recently, FixMatch [98] achieves state-of-the-art classification accuracy via assigning the strongly augmented unlabeled samples with the pseudo labels produced from the corresponding weakly augmented samples when the highest confidence score exceeds a certain threshold. While UDA [117] was proposed to treat the classifier's "sharpen" output confidence scores as the 'pseudo labels' rather than one class label. Similar to FixMatch, UDA trains strongly augmented unlabeled samples with the pseudo labels produced from the corresponding weakly augmented samples. FlexMatch [121] updates FixMatch by introducing the curriculum learning-based method to flexibly adjust the threshold for different classes during the training. Existing studies on SSL mainly focus on how to improve the performance, however, we are the first to show that state-of-the-art SSL methods are vulnerable to our tailored membership inference attacks, which exploit the strong/weak data augmentations used by state-of-the-art SSL methods.

2.3 Contrastive Learning

Supervised learning is powerful, but its success heavily depends on the labeled training dataset. In the real world, the high-quality labeled dataset is hard and expensive to obtain as it often relies on human annotation. For instance, the ILSVRC2011 dataset [92] contains more than 12 million labeled images that are all annotated by Amazon Mechanical Turk workers. Meanwhile, unlabeled data is being generated at every moment. To leverage large-scale unlabeled data, self-supervised learning is introduced.

The goal of self-supervised learning is to get labels from an unlabeled dataset for free so that one can train an unsupervised task on this unlabeled dataset in a supervised manner. Contrastive learning/loss [43, 82, 53, 17, 119, 48, 60] is one of the most successful and representative self-supervised learning paradigms in recent years and has received a lot of attention from both academia and industry. Oord et al. [82] propose contrastive predictive coding, which leverages autoregressive models to predict future observations for data samples. Wu et al. [114] utilize a memory bank to save instance representation and k-nearest neighbors to conduct prediction. He et al. [48] introduce MoCo, which relies on momentum to update the key encoder with the query encoder to maintain consistency. Chen et al. [17] propose SimCLR, which leverages data augmentation and the projection head to enhance the performance of contrastive models. SimCLR is the most prominent contrastive learning paradigm at the moment [72], thus we concentrate on it in Chapter 4. In general, contrastive learning aims to map a sample closer to its correlated views and more distant to other samples' correlated views. In this way, contrastive learning is able to learn an informative representation for each sample, which can then be leveraged to perform different downstream tasks. Contrastive learning relies on Noise Contrastive Estimation (NCE) [43] as its objective function, which can be formulated as:

$$\mathcal{L} = -\log\left(\frac{e^{sim(f(x),f(x^+))}}{e^{sim(f(x),f(x^+))} + e^{sim(f(x),f(x^-))}}\right) \tag{2.4}$$

where f is an encoder that maps a sample into its representation, x^+ is similar to x (referred to as a positive pair), x^- is dissimilar to x (referred to as a negative pair), and *sim* is a similarity function. The structure of the encoder and the similarity function can vary from different tasks. In Chapter 4, we focus on one of the most popular contrastive learning frameworks [72], namely SimCLR [17]. This framework is assembled with the following components.

Data Augmentation: SimCLR first uses a data augmentation module to transform a given data sample x to its two augmented views, denoted by $\tilde{x}_i$ and $\tilde{x}_j$, which can be considered as a positive pair for x. In our work, we follow the same data augmentation process used by SimCLR [17], i.e., first random cropping and flipping with resizing, second random color distortions, and third random Gaussian blur.

Base Encoder f: Base encoder f is used to extract representations from the augmented data samples. The base encoder can follow various neural network (NN) architectures. In Chapter 4, we apply the widely used ResNet [49] (ResNet-18 and ResNet-50) and MobileNetV2 [95] to obtain the representation $h_i = f(\tilde{x}_i)$ for $\tilde{x}_i$.

Projection Head g: Projection head g is a simple neural network that maps the representations from the base encoder to another latent space to apply the contrastive loss. The goal of the projection head is to enhance the encoder's performance. Following Chen et al. [17], we implement it with a 2-layer MLP (multilayer perceptron) to obtain the output $z_i = g(h_i)$ for h_i.

Contrastive Loss Function: The contrastive loss function is defined to guide the model to learn the general representation from the data itself. Given a set of augmented samples $\{\tilde{x}_k\}$ including a positive pair $\tilde{x}_i$ and $\tilde{x}_j$, the contrastive loss maximizes the similarity between $\tilde{x}_i$ and $\tilde{x}_j$ and minimizes the similarity between $\tilde{x}_i$ ($\tilde{x}_j$) and other samples. For each mini-batch of N samples, we have $2N$ augmented samples. The loss function for a positive pair $\tilde{x}_i$ and $\tilde{x}_j$ can be formulated as:

$$\ell(i,j) = -\log \frac{e^{sim(z_i,z_j)/\tau}}{\sum_{k=1,k\neq i}^{2N} e^{sim(z_i,z_k)/\tau}} \tag{2.5}$$

where $sim(z_i, z_j) = {z_i}^\top z_j / \|z_i\| \|z_j\|$ represents the cosine similarity between z_i and z_j and τ is a temperature parameter. The final loss is calculated over all positive pairs in a mini-batch, which can be defined as the following.

$$\mathcal{L}_{Contrastive} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)] \tag{2.6}$$

Here, $2k-1$ and $2k$ are the indices for each positive pair.

Training classifiers with SimCLR can be partitioned into two phases. In the first phase, we train a base encoder as well as a projection head by the contrastive loss using an unlabeled dataset. After training, we discard the projection head and keep the base encoder only. In the second phase, to perform classification tasks, we freeze the parameters of the encoder, add a trainable linear layer at the end of the encoder, and fine-tune the linear layer with the cross-entropy loss (see Equation 2.3) on a labeled dataset. The linear layer serves as a classifier, with its input being the representations generated by the encoder. We refer to this linear layer as the *classification layer.* In Chapter 4, we call a model trained with supervised learning as a *supervised model* and a model trained with contrastive learning as a *contrastive model.* Also, we consider contrastive models trained on image datasets, as most of the current development of contrastive learning focus on images.

Compared to supervised learning, contrastive learning can learn more informative representations for data samples. Previous work shows that supervised models are vulnerable to various privacy attacks [97, 118, 94, 76, 15, 102, 13]. However, to the best of our knowledge, privacy risks stemming from contrastive models have been left largely unexplored. In this work, we aim to fill this gap.

2.4 Graph Neural Networks

Many important real-world datasets come in the form of graphs or networks, e.g., social networks, knowledge graph, and chemical networks. Therefore, it is urgent to develop machine learning algorithms to fully utilize graph data. To this end, a new family of machine learning algorithms, i.e., graph neural networks (GNNs), has been proposed and shown superior performance in various tasks [5, 25, 61, 108].

Training a GNN Model: Given a graph, attributes for each node in the graph, and a small number of labeled nodes, GNN trains a neural network to predict labels of the remaining unlabeled nodes via analyzing the graph structure and node attributes. Formally, we define the *target dataset* as $\mathcal{D} = (\mathcal{A}, \mathcal{F})$, where $\mathcal{A}$ is the adjacency matrix of the graph and $\mathcal{F}$ contains all nodes' attributes. Specifically, $\mathcal{A}_{uv}$ is an element in $\mathcal{A}$: If there exists an edge between node u and node v, then $\mathcal{A}_{uv} = 1$, otherwise $\mathcal{A}_{uv} = 0$. Moreover, $\mathcal{F}_u$ represents the attributes of u. $\mathcal{V}$ is a set containing all nodes in the graph. Note that we consider undirected graphs in Chapter 5, i.e., $\forall u, v \in \mathcal{V}, \mathcal{A}_{uv} = \mathcal{A}_{vu}$.

A GNN method iteratively updates a node's features via aggregating its neighbors' features using a neural network, whose last layer predicts labels for nodes. Different GNN methods use slightly different aggregation rules. For instance, *graph convolutional network (GCN)*, the most representative and well-established GNN method [61], uses a multi-layer neural network whose architecture is determined by the graph structure. Specifically, each layer obeys the following propagation rule to aggregate the neighboring features:

$$H^{(k+1)} = \sigma(\tilde{\mathcal{Q}}^{-\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{Q}}^{-\frac{1}{2}}H^{(k)}W^{(k)}), \tag{2.7}$$

where $\tilde{\mathcal{A}} = \mathcal{A} + I$ is the adjacency matrix of the graph with self-connection added, i.e., I is the identity matrix. $\tilde{\mathcal{Q}}^{-\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{Q}}^{-\frac{1}{2}}$ is the symmetric normalized adjacency matrix and $\tilde{\mathcal{Q}}_{uu} = \sum_u \tilde{\mathcal{A}}_{uv}$. Moreover, $W^{(k)}$ is the trainable weight matrix of the kth layer and $\sigma(\cdot)$ is the activation function to introduce non-linearity, such as ReLU. As the input layer, we have $H^{(0)} = \mathcal{F}$. When the GCN uses a two-layer neural network, the GCN model can be described as follows:

$$\mathit{softmax}(\tilde{\mathcal{Q}}^{-\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{Q}}^{-\frac{1}{2}}\sigma(\tilde{\mathcal{Q}}^{-\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{Q}}^{-\frac{1}{2}}\mathcal{F}W^{(0)})W^{(1)}). \tag{2.8}$$

Note that in most parts of Chapter 5, we focus on two-layer GCN. Later, we show that our attack can be also performed on other types of GNNs, including GraphSAGE [45] and GAT [108] (see Section 5.4).

Prediction in a GNN Model: Since all nodes' attributes and the whole graph have been fed into the GNN model in the training phase to predict the label of a node, we

Table 2.1: List of notations.

Notation	Description
$\mathcal{D}$	Target dataset
$\mathcal{A}$	Graph of $\mathcal{D}$ represented as adjacency matrix
$\mathcal{A}^*$	Partial graph of $\mathcal{D}$
$\mathcal{F}$	Nodes' attributes of $\mathcal{D}$
$\mathcal{V}$	Set of nodes of $\mathcal{D}$
f	Target model
g	Reference model
$f(u)$	u's posteriors from the target model
$g(u)$	u's posteriors from the reference model
$\mathcal{D}'$	Shadow dataset
f'	Shadow target model
g'	Shadow reference model
$\mathcal{K}$	Adversary's knowledge
$d(\cdot,\cdot)$	Distance metric
$\Psi(\cdot,\cdot)$	Pairwise vector operations
$e(f(u))$	Entropy of $f(u)$

only need to provide the node's ID to the trained model and obtain the prediction result. We assume the prediction result is a posterior distribution (called *posteriors*) over the possible labels for the node. Our work shows that such posteriors reveal rich information about the graph structure: As mentioned before, a GNN essentially learns a node's features via aggregating its neighbors' features, if two nodes are connected, then their posteriors should be similar. We leverage this to build our attack models. We further use f to denote the target GNN model and $f(u)$ to represent the posteriors of node u. For presentation purposes, we summarize the notations introduced here and in the following sections in Table 2.1.

3
Semi-supervised Learning

3.1 Introduction

Machine learning (ML) has made tremendous progress in the past decade. One of the key reasons for the great success of ML models can be credited to the large-scale labeled data. However, such labeled datasets are often hard to collect as they rely on human annotations and expertise in the specific domain. Meanwhile, unlabeled datasets are easy to obtain. To better leverage the unlabeled data, semi-supervised learning (SSL) has been proposed. Concretely, SSL uses a small set of labeled data and a large set of unlabeled data to jointly train the ML model. In recent years, SSL shows its effectiveness on different tasks by leveraging much fewer labeled data [98, 117, 121]. For instance, by only using 250 labeled samples, FlexMatch [121] can achieve about 95% accuracy on CIFAR10.

Different from supervised learning where every data sample is treated equally in the training procedure, SSL takes different ways to handle the labeled and unlabeled data samples during the training. Concretely, the state-of-the-art SSL methods [98, 117, 121] leverage weak augmentation to the labeled samples and trains them in a supervised manner. For each unlabeled sample, it would generate a weakly-augmented view and a strongly-augmented view (by weak and strong augmentations), and the goal is to leverage the model's prediction probability (referred to as prediction or posteriors) of the weakly-augmented view to guide the training of the strongly-augmented view of the sample. Instead of directly using the posteriors as a "soft" label, those SSL methods switch the posteriors into a "sharpen" [117] or "hard" label [98, 121]. Note that the sample is not used to train the model until the highest probability of the prediction on the weakly-augmented view exceeds a pre-defined threshold τ. In this way, the model trained by SSL can gradually learn more accurate predictions.

Despite being powerful, ML models are shown to be vulnerable to various privacy attacks [33, 97, 102], represented by membership inference attacks [97, 94, 78, 103]. The goal of membership inference attack is to determine whether a data sample is used to train a target ML model. Successful membership inference attacks can raise privacy concerns as they may reveal sensitive information of people. For instance, if an ML model is trained on the data for people with a certain sensitive attribute (e.g., diseases), identifying the person in the training dataset directly reveals this individual's sensitive attribute. So far, most of the efforts on membership inference attacks concentrate on models trained by supervised learning. Also, there are some exploratory researches investigating the privacy risks in self-supervised learning [71, 52]. However, in SSL, the labeled and unlabeled samples are treated differently during the training. It is important to quantify whether this unique training paradigm would lead to different privacy risks for labeled and unlabeled samples. Also, as the different augmented views instead of the original samples are used to train the model, we are curious whether a more effective membership inference attack mechanism can be proposed against SSL. To be best of our knowledge, this is largely unexplored.

In this work, we fill the gap by proposing the first data augmentation-based membership inference attack method against SSL. A key advantage for SSL is that it only needs a small amount of labeled data and leverages the unlabeled data itself to guide the training. Concretely, for the labeled data, the model is trained in a supervised manner.

For the unlabeled data, SSL leverages the data itself as the supervision. In particular, for each unlabeled training sample, a weakly augmented and a strongly augmented views will be fed into the target model and the training objective is to minimize the distance of the model's prediction on these two views. Our proposed data augmentation-based attack is based on the intuition that the model's prediction of these two views should be more similar if the sample belongs to the model's training set.

We conduct our evaluation on three SSL methods (FixMatch, FlexMatch, and UDA) and three commonly used SSL datasets (SVHN, CIFAR10, and CIFAR100). Our empirical results show that our proposed attack can consistently outperform baseline attacks and reaches the best performance. For instance, for FixMatch trained on CIFAR10 with 500 labeled samples, our attack achieves 0.780 AUC while the best baseline attack only has 0.722 AUC. This indicates that our attack can better unleash the membership information in SSL.

Moreover, we find that, unlike supervised learning where the membership leakage can be credited to the overfitting nature of the model [97, 94] (i.e., the model predicts the training data more accurately than the testing data), models trained by SSL methods are well generalized and have almost no overfitting but still suffer high membership inference risk. Our analysis reveals that the model indeed "memorizes" the training data, but such memorization does not present as a more accurate prediction, but a more confident prediction. We show that the prediction entropy distribution of members and non-members has a large gap in models trained by SSL (measured by Jason-Shannon (JS) Distance).

Contributions: (1) We are the first to study the privacy risk of SSL through the lens of membership inference attacks and we propose a data augmentation-based attack that is tailored to SSL methods. (2) We conduct extensive experiments on SVHN, CIFAR10, and CIFAR100 datasets. Our results show that our proposed attack outperforms baseline attacks that are extended from existing works to SSL settings. (3) We show that the effectiveness of membership inference attacks against SSL is not credited to the model's overfitting level but credited to the model prediction's distinguishable entropy distributions for members and non-members (measured by Jason-Shannon Distance). (4) We study an early-stopping-based defense against our proposed attack. We show that this defense can decrease the attack AUC of our attack but sacrifice the testing accuracy of the trained models.

3.2 Conventional Membership Inference Attacks

In membership inference attacks, the adversary aims to determine whether a given data sample x belongs to the target model $\mathcal{T}$'s training dataset or not given the adversary's background knowledge $\mathcal{K}$. A data sample x is called *member* (or *non-member*) if it belongs to (or does not belong to) the training dataset of the target model $\mathcal{T}$. Formally, we define the membership inference attack as $\mathcal{A} : x, \mathcal{T}, \mathcal{K} \rightarrow \{0, 1\}$, where the attack $\mathcal{A}$ is essentially a mapping function and 1 (or 0) means the data sample x is a member (or non-member).

3.2.1 Threat Model

Given a target model $\mathcal{T}$, we first assume that the adversary only has black-box access to it, which means that the adversary can only query the target model with a data sample x and obtain the target model's prediction on it (denoted as posteriors). Note that in this work we consider the black-box attack since it is the most difficult and practical real-world scenario.

Following previous work [97, 52, 103], we assume that the adversary has a *shadow dataset* $\mathcal{D}_{shadow}$ that has the same distribution as the target model $\mathcal{T}$'s training dataset $\mathcal{D}_{target}^{train}$. The adversary can use the shadow dataset $\mathcal{D}_{shadow}$ to train a *shadow model* $\mathcal{S}$, which mimics the behavior of the target model $\mathcal{T}$ to better conduct the attacks. Also, we assume that the shadow model $\mathcal{S}$ has the same architecture as the target model. Such an assumption is realistic as: (1) The adversary can leverage the same machine learning service to train the shadow model and (2) The adversary can perform hyperparameter stealing attacks [81, 109] to obtain the target model's architecture.

3.2.2 Methodology

Generally speaking, the membership inference attack pipeline usually consists of three major components, i.e., shadow training, constructing the attack training dataset, and attack model training or performing the attack.

Shadow Training: Shadow training [97, 78, 94] aims to train shadow models to mimic the behavior of the target model based on the adversary's background knowledge. Specifically, the adversary first evenly splits the shadow dataset $\mathcal{D}_{shadow}$ into two disjoint parts, i.e., shadow training data $\mathcal{D}_{shadow}^{train}$ and shadow testing data $\mathcal{D}_{shadow}^{test}$. The adversary then uses the $\mathcal{D}_{shadow}^{train}$ to train a shadow model $\mathcal{S}$ that mimics the behavior of the target model $\mathcal{T}$.

Constructing Attack Training Dataset: To construct the training dataset for the attack model, the adversary first uses $\mathcal{D}_{shadow}^{train}$ (contains members) and $\mathcal{D}_{shadow}^{test}$ (contains non-members) to query the shadow model $\mathcal{S}$ and obtain the corresponding posteriors. Following Salem et al. [94], we leverage the descendingly sorted posteriors as the inputting features for the attack model. Finally, we assign the membership status 1/0 for members/non-members as labels.

Training Neural Network-based Attack Model: For neural network-based attacks [97, 94] (denoted as $\mathcal{A}_{NN}$), the adversary aims to train a neural network-based attack model to distinguish members and non-members given the posteriors generated by the target model $\mathcal{T}$. After constructing the attack training dataset, the adversary trains an NN-based attack model on the constructed training dataset. Following previous works [97, 94, 52, 71], we consider a multi-layer perceptron (MLP) as the neural network architecture for the attack model. Once the attack model is trained, it can be used by the adversary to predict whether a given data sample x is a member or non-member.

Metric-based Attacks: Metric-based attacks [118, 104, 66, 103] also require the adversary to train a shadow model $\mathcal{S}$. Unlike NN-based attacks that require training an attack model, metric-based attacks design a specific metric and calculate a threshold over the metrics by querying the shadow model $\mathcal{S}$ with $\mathcal{D}^{train}_{shadow}$ and $\mathcal{D}^{test}_{shadow}$. We adopt four state-of-the-art metric-based attacks following Song and Mittal. [103]: (1) Prediction correctness ($\mathcal{A}_{Corr}$) which considers a sample as a member if the label is correctly predicted by the target model; (2) Prediction confidence ($\mathcal{A}_{Conf}$) which judges a sample as a member if the prediction probability at the ground truth class is larger than a pre-defined threshold (learned from the shadow model); (3) prediction entropy ($\mathcal{A}_{Ent}$) which considers a sample as a member if the entropy of the prediction is smaller than a pre-defined threshold (learned from the shadow model); and (4) Modified prediction entropy ($\mathcal{A}_{Ment}$) which is similar to (3) but modifies the entropy function and combines the ground truth label as a new metric.

3.3 Our Method

The main difference between SSL methods and supervised learning methods is that SSL methods leverage a large amount of unlabeled samples together with a small amount of labeled samples to train the model. Recall that state-of-the-art SSL methods [116, 98, 121] leverage both weak and strong data augmentations to the unlabeled samples during the training. The key idea of these SSL methods is to train the model that maximizes the model's prediction agreement on weakly and strongly augmented views that come from the same unlabeled sample. In other words, for an unlabeled training sample, the trained model may tend to output more similar posteriors for its weakly and strongly augmented views. While for labeled training samples, the trained model may output similar posteriors for different weakly augmented views from the same sample since those posteriors result in the same predicted label. This observation may also hold for unlabeled samples since the posteriors of the same training unlabeled sample tend to produce the same "pseudo label". Intuitively speaking, the target model $\mathcal{T}$ may output similar (or dissimilar) posteriors for different weakly and/or strongly augmented views of member (or non-member).

Based on the above intuition, we propose a data augmentation-based membership inference attack (denoted as $\mathcal{A}_{DA}$) tailored to state-of-the-art SSL methods. $\mathcal{A}_{DA}$ follows the similar pipeline as NN-based attack $\mathcal{A}_{NN}$, i.e., shadow training and training an NN-based attack model.

However, our attack $\mathcal{A}_{DA}$ extracts membership features (i.e., the input for the attack model) in a different way from the attack $\mathcal{A}_{NN}$. Specifically, given a data sample x, we first generate K weakly augmented and K strongly augmented views of it, respectively. Then we use the augmented views to query the shadow model to obtain output posteriors. After that, we calculate three similarity matrices among: (1) K posteriors of weakly augmented views themselves, (2) K posteriors of strongly augmented views themselves, and (3) K posteriors of weakly augmented views and K posteriors of strongly augmented views, based on a predefined similarity metrics (e.g., JS Distance, Cosine Distance, etc.). Then we obtain three similarity matrices where each of them contains K^2 similarity values. We expand each similarity matrix into a vector and sort the values in each vector in descending order, respectively. Then we concatenate them together and finally obtain a vector with $3K^2$ values. The obtained vectors are then assigned with the membership

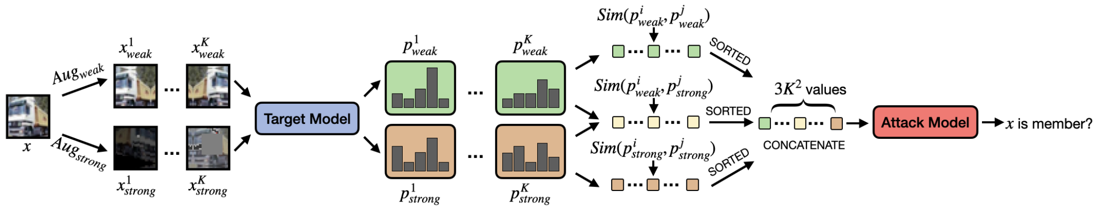


Figure 3.1: Overview of our data augmentation based attack $\mathcal{A}_{DA}$.

status as the labels. Once the attack model is trained, to determine whether a sample belongs to the target model's training dataset, we again generate K weakly and K strongly augmented views of it to query the target model, generate the attack input to query the attack model and obtain its membership prediction. Figure 3.1 shows the overview of $\mathcal{A}_{DA}$ and the detailed algorithm is shown in Algorithm algorithm 1 in the supplemental material.

3.4 Evaluation

3.4.1 Experimental Setup

Dataset Configuration: We evaluate the performance of target models and membership inference attacks on three commonly used SSL datasets, i.e., SVHN, CIFAR10, and CIFAR100. For each dataset, we first randomly split it into four equal parts, i.e., $\mathcal{D}^{train}_{target}$, $\mathcal{D}^{test}_{target}$, $\mathcal{D}^{train}_{shadow}$, and $\mathcal{D}^{test}_{shadow}$. We leverage $\mathcal{D}^{train}_{target}$ to train the target model and consider the samples from $\mathcal{D}^{train}_{target}$ as the members of the target model. Samples in $\mathcal{D}^{test}_{target}$ are considered as the non-members of the target model. $\mathcal{D}^{train}_{shadow}$ is used to build up the shadow model. Both $\mathcal{D}^{train}_{shadow}$ and $\mathcal{D}^{test}_{shadow}$ are used to train the attack model. Note that the $\mathcal{D}^{train}_{target}$ is smaller than the original training dataset (e.g., for CIFAR10, $\mathcal{D}^{train}_{target}$ contains 15,000 samples while the original training dataset contains 50,000 samples), which may lead to lower target model performance.

Metric: We follow previous work [98, 117, 121, 97, 52] and adopt testing accuracy as the evaluation metric for target model performance. Regarding the attack, we leverage AUC as the evaluation metric [112, 69] as we aim to quantify both the general membership privacy risk for members vs. non-members and the separate privacy risks for labeled/unlabeled members vs. non-members (unbalanced).

Target Model: For a fair comparison, we apply the same hyperparameters for FixMatch, UDA, and FlexMatch. Specifically, we apply SGD optimizer. The initial learning rate is set to 0.03 with a cosine learning rate decay which sets the learning rate to $\eta\ cos(\frac{\pi k}{2N})$, where η is the initial learning rate, k is the current training step, and N is the total number of training steps. We set $N = 100 \times 2^{10}$. We leverage an exponential moving average of model parameters with the momentum of 0.999. The labeled batch size (i.e., the batch size of the labeled data) is set to 64 and the ratio of unlabeled batch size to the labeled batch size is set to 7. Note that the threshold τ is set to 0.8

for UDA and 0.95 for FixMatch and FlexMatch following the original papers. We apply RandAugment [23] as the strong augmentation method in our experiments (see Section 3.7.1 in the supplementary material). Regarding the model architectures, we leverage Wide ResNet (WRN) [120] with a widen factor of 2 as the target model architecture and we also investigate different widen factors in our ablation studies (see Section 3.4.6).

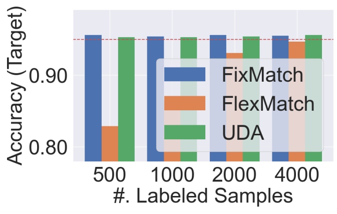
Attack Model: We apply a 3-layer MLP with 64, 32, and 2 hidden neurons for each layer as the attack model's architecture. We train the attack model for 100 epochs using Adam optimizer with the learning rate of 0.001 and the batch size of 256. For our proposed attack, we set the number of augmented views used to query the target model to 10 and leverage JS Distance as the similarity function. Note that we also evaluate different numbers of augmented views and different similarity functions in our ablation studies (see Section 3.4.5).

3.4.2 Target Model Performance

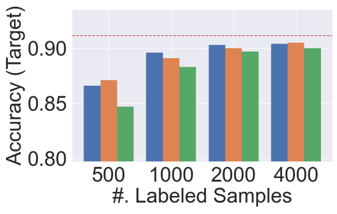
We first evaluate the performance of the supervised models and the SSL models on the original classification tasks using $\mathcal{D}_{target}^{test}$. We use the full $\mathcal{D}_{target}^{train}$ to train the supervised models, while we use a small portion of labeled samples and treat the remaining samples as unlabeled ones in $\mathcal{D}_{target}^{train}$ when training the SSL models. We observe that SSL with more labeled samples can achieve better performance on the original classification tasks. For instance, on Figure 3.2b, when the target model is FixMatch trained on CIFAR10, the classification accuracy is 0.866, 0.896, 0.903, and 0.904 with 500, 1,000, 2,000, and 4,000 labeled samples, respectively. This is expected as more labeled samples help the target model to better learn the decision boundary at the early stage. Another observation is that for a more complicated task, it may require more labeled samples to achieve comparable performance as the supervised models. We consider SVHN, CIFAR10, and CIFAR100 to have increasing difficulty levels. Take models trained by UDA as a case study (green bar in Figure 3.2), on SVHN, with only 500 labeled samples, the testing accuracy is 0.953, which is even better than the supervised model (0.951). We suspect the reason is that 500 labeled samples are enough to learn a relatively accurate decision boundary and the strong data augmentation used in SSL methods can better help the model to generalize to the unseen data. On the other hand, on CIFAR10 and CIFAR100, it may require 1,000 and 4,000 labeled samples to catch up with the performance of the supervised model. Such observation indicates that a larger portion of labeled data is still helpful for a more complicated task.

3.4.3 Membership Inference Attack Performance

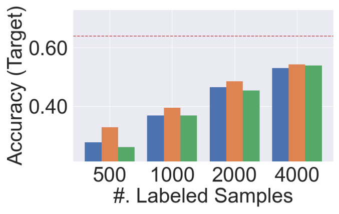
We then evaluate the performance of different membership inference attacks on SSL models. The results are summarized in Figure 3.3. Note that we leverage AUC as the attack evaluation metric to better quantify the privacy leakage of all training data (first row) as well as the separate privacy leakage of labeled (second row) and unlabeled (third row) training data. We find that for the baseline attacks (i.e., except our $\mathcal{A}_{DA}$), $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$ perform the best, while other attacks like $\mathcal{A}_{Corr}$, $\mathcal{A}_{Conf}$, and $\mathcal{A}_{Ment}$ are less



(a) SVHN

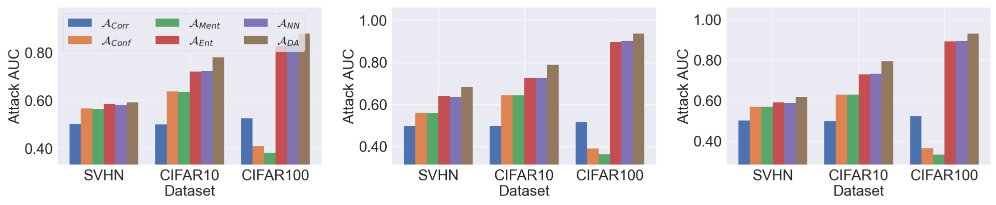


(b) CIFAR10

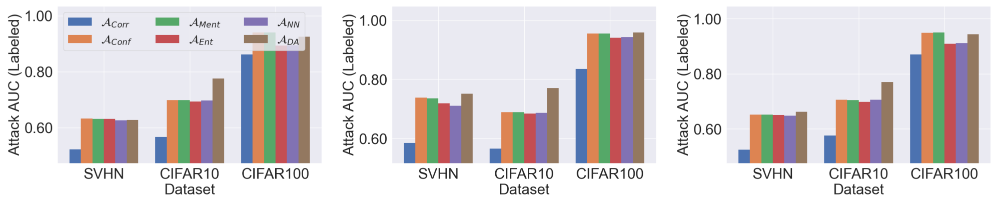


(c) CIFAR100

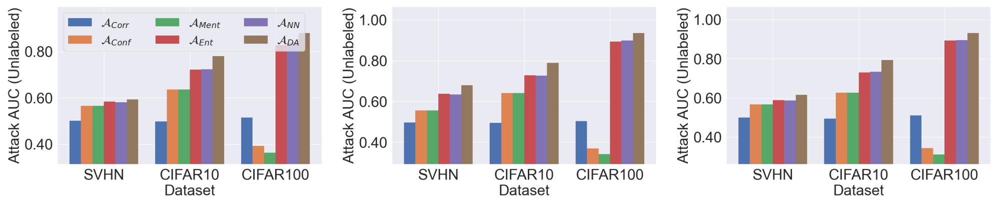
Figure 3.2: Testing accuracy on the original classification tasks. Note that the red dash line denotes the performance of supervised models.



(a) Attack AUC



(b) Attack AUC (Labeled)



(c) Attack AUC (Unlabeled)

Figure 3.3: The AUC of membership inference attacks against models trained by different SSL methods with 500 labeled samples. The first to third columns denote the models trained by FixMatch, FlexMatch, and UDA, respectively.

effective. For instance, on FlexMatch trained on CIFAR10 with 500 labeled samples (the middle one of Figure 3.3a), the attack AUC is 0.726 for both $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$, while only 0.497, 0.643, and 0.642 for $\mathcal{A}_{Corr}$, $\mathcal{A}_{Conf}$, and $\mathcal{A}_{Ment}$. To better investigate the reason behind this, we further measure the attack AUC for labeled data and unlabeled data, respectively. We find that $\mathcal{A}_{Conf}$ and $\mathcal{A}_{Ment}$ achieve even better performance on labeled training samples than $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$. For instance, for FlexMatch trained on CIFAR100, the AUC (labeled) for $\mathcal{A}_{Conf}$ and $\mathcal{A}_{Ment}$ are both 0.955, while only 0.944 and 0.941 for $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$. This is expected as the labeled sample has a higher confidence score on its ground-truth label, which facilitates the attacks that leverage such information. However, this is not the case for the unlabeled samples. As we can observe that, for FlexMatch trained on CIFAR100, the AUC (unlabeled) is only 0.370 and 0.341 for $\mathcal{A}_{Conf}$ and $\mathcal{A}_{Ment}$, but 0.899 and 0.894 for $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$. This indicates that, for the unlabeled samples, the model may give similar correctness predictions on both unlabeled training samples and testing samples, which makes it harder to differentiate them. However, the model will give more confident predictions on unlabeled training samples than on testing samples, which results in better performance for $\mathcal{A}_{NN}$ and $\mathcal{A}_{Ent}$.

On the other hand, we also observe that the data augmentation-based attack $\mathcal{A}_{DA}$ achieves consistently better overall performance on all datasets and SSL methods than those baseline attacks. Moreover, $\mathcal{A}_{DA}$ works better in determining the membership of unlabeled training samples. For instance, on FixMatch trained on CIFAR10, the unlabeled AUC is 0.780 for $\mathcal{A}_{DA}$ while only 0.722 for the best baseline attack ($\mathcal{A}_{NN}$). This is because $\mathcal{A}_{DA}$ unveils the pattern that the predictions of a sample's weak and strong augmented views should be closer if the sample is an unlabeled sample used during the training.

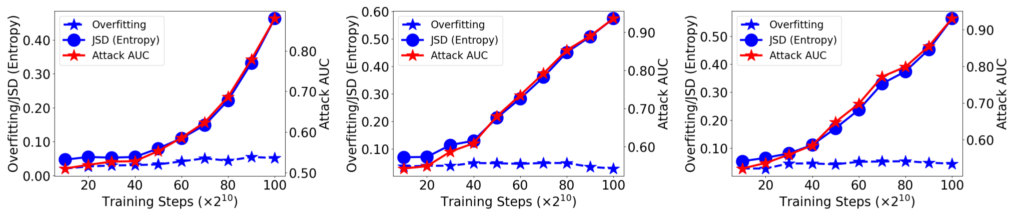
3.4.4 What Determines Membership Inference Attack in SSL

The effectiveness of membership inference attacks has been largely credited to the intrinsic overfitting phenomenon of the ML model [97, 94]. Here overfitting denotes the model's training accuracy minus its testing accuracy. Such assumption has been verified on various ML models [97, 78, 94, 52]. However, it is unclear whether such assumption still holds for SSL. If not, what is the reason for models trained by SSL to be vulnerable to membership inference attacks?

From Figure 3.3, we find that $\mathcal{A}_{Ent}$ achieves good performance in predicting the membership status of a sample, which gives us the hint that the members' and non-members' predictions may have different entropy distributions. Here we leverage the JS Distance to quantify the difference between the entropy distribution of members' and non-members' predictions (we denote this measure as JS Distance (Entropy)).

To better quantify the correlation between different factors (e.g., overfitting, JS Distance (Entropy)) and the attack performance, we measure them under different training steps of the target models. Note that here we consider the $\mathcal{A}_{DA}$ as it performs the best in membership inference. Figure 3.4 shows the results of models trained by different SSL methods on the CIFAR100 with 500 labeled samples. The results for models trained on different datasets and with different numbers of labeled samples are shown in Section 3.7.3 in supplementary materials.

In Figure 3.4, we observe that during the whole training procedure, the models trained by SSL have nearly 0 overfitting, which means that the models can always generalize well to the unseen data. However, we find that the attack AUC keeps increasing during the training. This indicates that the success of membership inference attacks is not necessarily related to the high overfitting level, which is overlooked by previous research. On the other hand, we observe that the JS Distance (Entropy) does increase during the training, which means that although the model does not predict more



(a) FixMatch **(b)** FlexMatch **(c)** UDA

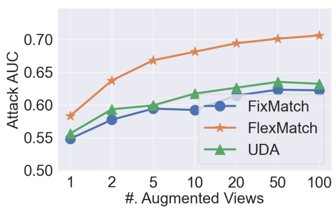
Figure 3.4: The overfitting/JS Distance (Entropy) and attack AUC with respect to different training steps. The target model is trained on CIFAR100 with 500 labeled samples. Note that we consider the attack AUC of $\mathcal{A}_{DA}$, which is the strongest attack.

accurately to the member samples (mainly unlabeled samples) than the non-member samples, the model indeed makes a more confident prediction on member samples (i.e., with lower entropy of prediction). Our observation reveals that the models trained by SSL indeed "memorize" the training data. However, such memorization does not reflect in the overfitting, i.e., the gap between training and testing accuracy. Instead, it reflects in the more confident prediction of the members than the non-members.

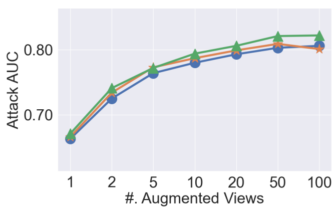
3.4.5 Ablation Study (Attack Model)

Number of Views: We first investigate how the attack performance would be affected by different numbers of views generated by the weak and strong augmentations to query the target model. To this end, for the SSL methods trained on different datasets with only 500 labeled samples, we range the number of views from 1 to 100 and the attack performance is shown in Figure 3.5. Note that we also show the results with 1,000, 2,000, and 4,000 labeled samples in Section 3.7.4 in the supplementary material. A clear trend is that more views lead to better attack performance. For instance, for FixMatch trained on CIFAR10 with 500 labeled samples (Figure 3.5b), the attack AUC is 0.780 with 10 augmented views, while 0.806 for 100 augmented views. However, we find that the attack performance increases rapidly when the number of augmented views increases from 1 to 10, but plateaus from 10 to 100. Moreover, more views lead to more queries to the target model and higher computational cost. We consider 10 as a suitable number of views since it achieves comparable performance to 100 while spending less query budget.

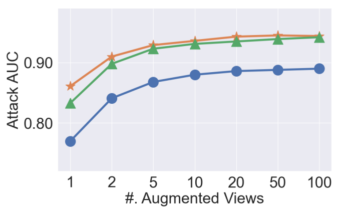
Similarity Function: Note that in our attack $\mathcal{A}_{DA}$, we can apply different similarity functions to measure the distance between the posteriors generated from different augmented views. Here we evaluate 4 distance metrics, i.e., Cosine Distance, Correlation Distance, Euclidean Distance, and JS Distance. The result for FixMatch, FlexMatch, and UDA trained on three different datasets with 500 labeled samples are summarized in Figure 3.6. Note that we also show the results with 1,000, 2,000, and 4,000 labeled samples in Section 3.7.5 in the supplementary material. We find that JS Distance consistently outperforms the other three distance metrics and achieves the best performance. For instance, FixMatch trained on CIFAR10, the attack AUC is 0.679, 0.682, 0.749, and



(a) SVHN

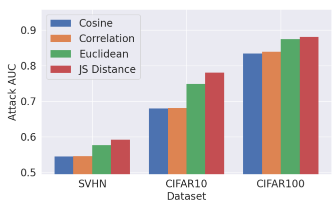


(b) CIFAR10

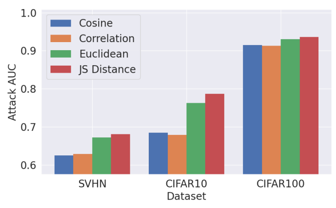


(c) CIFAR100

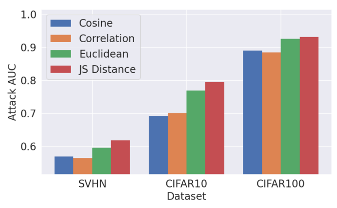
Figure 3.5: The attack AUC of $\mathcal{A}_{DA}$ with different numbers of augmented views to query the target model. The target model is trained with 500 labeled samples.



(a) FixMatch



(b) FlexMatch



(c) UDA

Figure 3.6: The attack AUC of $\mathcal{A}_{DA}$ with different similarity functions. The target model is trained with 500 labeled samples.

0.780 for Cosine Distance, Correlation Distance, Euclidean Distance, and JS Distance. We suspect the reason is that JS Distance is designed to calculate the difference between two probabilities' distributions, which may better fit our scenario as the prediction posteriors are probability as well.

Moreover, we also find that the magnitude of data augmentation and the shadow model architecture only have limited impact on the attack performance (see Section 3.7.6 in the supplementary material for more details).

3.4.6 Ablation Study (Target Model)

We also investigate whether the target model's capacity and the unlabeled ratio (i.e., $\frac{batchsize(unlabeled)}{batchsize(labeled)}$ during each training step) would affect the performance. Note that here we select FixMatch trained on CIFAR100 with 500 labeled data as a case study, since the target model's capacity and the unlabeled ratio are general to different SSL methods, and CIFAR100 with 500 labeled data is the most challenging setting to train the target model (see Figure 3.2). We consider an adaptive adversary [58] who is aware of the training details of the target model and can train the shadow model in the same way.

Model Capacity: The target model architecture we leverage in our work is WRN28-2. To better quantify the impact of model capacity on the target and attack performance, we vary the width of WRN28 from 1 to 8 and the results are shown in Table 3.1. We can observe that a larger model capacity, in general, leads to a better target model's performance on the original classification task, but also increases the membership risk

Table 3.1: The target model performance and attack performance ($\mathcal{A}_{DA}$) when the target model has different capacities. The target model is trained by FixMatch on CIFAR100 with 500 labeled samples. ($\star$) denotes the default setting.

Architecture	Test ACC	Attack AUC	Attack AUC (Labeled)	Attack AUC (Unlabeled)
WRN28-1	0.217	0.726	0.954	0.718
WRN28-2 ($\star$)	0.276	0.874	0.896	0.873
WRN28-4	0.299	0.917	0.910	0.917
WRN28-8	0.305	0.927	0.918	0.927

Table 3.2: The target model performance and attack performance ($\mathcal{A}_{DA}$) when the target model leverages different unlabeled ratios during each training step. The target model is trained by FixMatch on CIFAR100 with 500 labeled samples. ($\star$) denotes the default setting.

Ratio	Test Acc	Attack AUC	Attack AUC (Labeled)	Attack AUC (Unlabeled)
1	0.210	0.578	0.965	0.565
2	0.263	0.646	0.942	0.636
4	0.273	0.785	0.946	0.779
7 ($\star$)	0.276	0.874	0.896	0.873
8	0.269	0.886	0.924	0.884
16	0.247	0.909	0.913	0.909

(especially for unlabeled samples). For instance, when the model capacity increase from WRN28-1 to WRN-28-8, the target testing accuracy increases from 0.217 to 0.305, while the attack AUC increases from 0.726 to 0.927. One reason is that, with larger model capacity, the model can "memorize" more different views of data samples, which not only facilitate target tasks, but also raise the membership risk.

Ratio of Unlabeled Samples in Each Training Step: We then investigate whether the unlabeled ratio (URatio) during each training step affects the attack performance. Concretely, we vary the unlabeled ratio from 1 to 16 while training the target model and Table 3.2 summarizes the results. We have two findings. First, the best target model performance reaches with the default setting (7). Second, the membership inference risk, in particular for the unlabeled data, keeps increasing when the ratio increases. On the other hand, the membership inference risk for labeled data slightly decreases (but still in a high level) while increasing the ratio. Therefore, a better choice may be leveraging a relatively small unlabeled ratio to achieve good target performance while reducing the membership risk for unlabeled samples.

3.5 Discussion on Defenses

We observe that the attack performance increases sharply at the late training steps (see Figure 3.4), which indicates that early stopping may be a good strategy to mitigate membership inference attacks. We take CIFAR100 with 4,000 labeled samples as a case study and show the target/attack model performance with respect to different training steps in Figure 3.7 (in the supplementary material). We find that there is a trade-off between model utility and membership inference performance, i.e., it may reduce both the attack performance and the target model's utility. We note that previous work [71, 103] also observe such a trade-off. Besides early stopping, we also evaluate three other defenses, i.e., top-k posteriors [97], model stacking [94], and DP-SGD [4]. Our case study (see Section 3.7.7 in the supplementary material) shows that early stopping achieves the best trade-off between model utility and membership inference performance.

3.6 Conclusion

In this work, we perform the first training data privacy quantification against models trained by SSL through the lens of membership inference attack. Empirical evaluation shows that our proposed data augmentation-based attacks consistently outperform the baseline attacks, in particular for unlabeled training data. Moreover, we have an interesting finding that the reason leading to membership leakage in SSL is different from the commonly believed overfitting nature of ML models trained in supervised manners. The models trained by SSL are well generalized to the testing data (i.e., with almost 0 overfitting level). However, our attack can still successfully break the membership privacy. The reason is that the models trained by SSL "memorize" the training data by giving more confident predictions on them, regardless of the ground truth labels. We also find that early stopping can serve as a countermeasure against the attacks, but there is a trade-off between membership privacy and model utility.

3.7 Appendix

3.7.1 Data Augmentation

Models trained by FixMatch, FlexMatch, and UDA apply both weak and strong augmentation to the unlabeled samples. For the weak augmentation, we apply random cropping with a padding of 4 and random horizontal flipping to each sample following [63, 120]. For the strong augmentation, we apply random augmentation [23] to each training sample, which consists of a group of augmentation operations. Specifically, we set $N = 2$ and $M = 10$ where N is the number of transformations to a given sample and M is the magnitude of global distortion.

3.7.2 Attack Performance with Different Numbers of Labeled Samples

Figure 3.8, Figure 3.9, and Figure 3.10 show the attack performance with 1,000, 2,000, and 4,000 labeled training data. We observe that our proposed data augmentation-based attack $\mathcal{A}_{DA}$ still consistently outperforms baseline attacks.

3.7.3 What Determines Membership Inference Attack in SSL with Different Numbers of Labeled Samples.

Figure 3.11, Figure 3.12, Figure 3.13, and Figure 3.14 shows the results of models trained by different SSL methods on the three datasets with 500, 1,000, 2,000, and 4,000 labeled samples. We has the similar finding as Section 3.4.4, i.e., the models trained by SSL has almost no overfitting, but the JS Distance (Entropy) and the attack performance do increase during the training.

3.7.4 Ablation Study: Number of Views

For the SSL methods trained on different datasets with 1,000, 2,000, and 4,000 labeled samples, we range the number of views from 1 to 100 and the attack performance is shown in Figure 3.15, Figure 3.16, and Figure 3.17, respectively. We have the similar observation as Section 3.4.5 that more number of views leads to better performance.

3.7.5 Ablation Study: Similarity Function

The results for FixMatch, FlexMatch, and UDA trained on different datasets with 1,000, 2,000, and 4,000 labeled samples are shown in Figure 3.18, Figure 3.19, and Figure 3.20, respectively. We observe that the JS Distance still consistently outperforms the other three distance metrics and achieves the best performance.

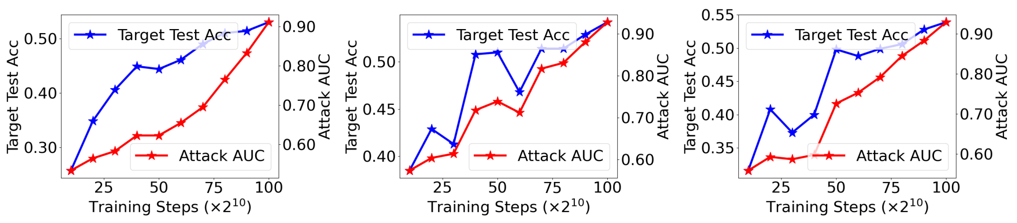
3.7.6 Ablation Study: Data Augmentation and Shadow Model Architecture

Data Augmentation: In the previous evaluation of $\mathcal{A}_{DA}$, we assume the adversary knows the data augmentation used to train the target model and can apply the same data augmentation to conduct the attack. We then relax this assumption to see whether $\mathcal{A}_{DA}$ is still effective with different levels of data augmentations. We take FixMatch trained on CIFAR10 with 500 labeled samples as a case study amd the results are shown in Table 3.3. We find that $\mathcal{A}_{DA}$ is still effective even with the weakest augmentation (Aug-Level is 0) and the attack performance can be improved with stronger augmentations added. For instance, the attack AUC is 0.876 and 0.882 when the Aug-Level is 0 and 4, respectively. This implies that a successful attack can still be launched even without the exact knowledge of the data augmentation used to train the target model.

Shadow Model Architecture: We then relax the assumption that the shadow model has the same architecture as the target model. Given the target model (WRN28-2 trained by FixMatch on CIFAR100 with 500 labeled samples), we train shadow models with WRN28-2, WRN28-1, WRN28-4, WRN28-8, and ResNet50 as the architectures, and the corresponding attack AUCs are 0.880, 0.875, 0.839, 0.835, and 0.859, respectively. Our attack achieves the highest attack AUC when the shadow and target models use the same architecture. However, our attack is still effective even if the shadow model has a different model architecture (e.g., the attack AUC is 0.859 when the shadow model architecture is ResNet50).

Table 3.3: The attack performance with respect to different augmentation levels. For Aug-Level=0, we only apply random cropping with flipping as both weak and strong augmentation. For 1-4, we gradually apply 1-4 transformation methods for the strong augmentation to each image from the general augmentation pools. The target model is trained by FixMatch on CIFAR100 with 500 labeled samples. ($\star$) denotes our default setting in the work.

Aug-Level	0	1	2 ($\star$)	3	4
Attack AUC	0.876	0.880	0.880	0.881	0.882



(a) FixMatch **(b)** FlexMatch **(c)** UDA

Figure 3.7: The target model performance and attack AUC with respect to different training steps. The target model is trained on CIFAR100 with 4,000 labeled samples, which has the highest performance on its original classification task. The attack model is $\mathcal{A}_{DA}$, which has the best attack performance.

3.7.7 Defense Evaluation

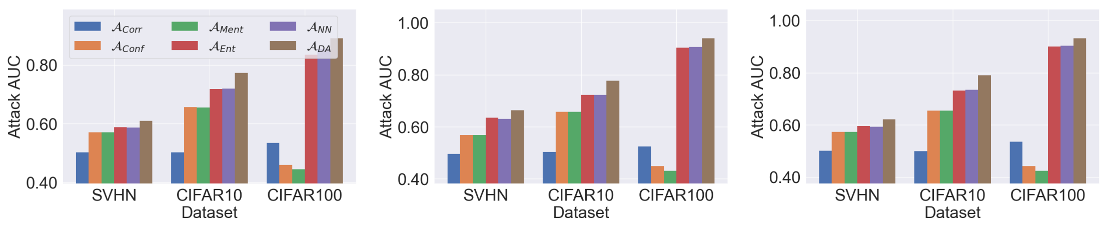
Besides early stopping (ES), we evaluate three more defenses, i.e., top-k posteriors (top-k) [97], model stacking (MS) [94], and DP-SGD [4]. We evaluate both the target model's utility (test acc) and the effectiveness of defenses (attack auc). The results are shown in Table 3.4. We observe that DP-SGD is the most effective defense since it achieves the lowest attack AUC with 0.598. However, DP-SGD suffers from unacceptable utility drop (with only 0.034 test acc). Existing work [56, 67] on DP also show that DP-SGD sacrifices the model's utility substantially in order to achieve good privacy guarantee. Therefore, we consider early stopping as the best defense since it achieves the best privacy-utility trade-off, i.e., it reduces the membership leakage to a large extent (from 0.918 to 0.695) while maintaining the utility (from 0.530 to 0.490).

Table 3.4: Target model accuracy and attack AUC for different defenses. The target model is WRN28-2 trained by FixMatch on CIFAR100 with 4,000 labeled samples (same setting as the paper). For early stopping (ES), we stop at 70 $\times 2^{10}$ training steps. For top-k, we set k=1 as it leaks the least information. For model stacking (MS), we train four models, i.e., WRN28-{1,2,4,8}, using the same dataset and average their posteriors. For DP-SGD, the noise scale is set to 10^{-5} and the gradient norm is set to 1. Note that we use the Opacus library (1) to implement DP-SGD.

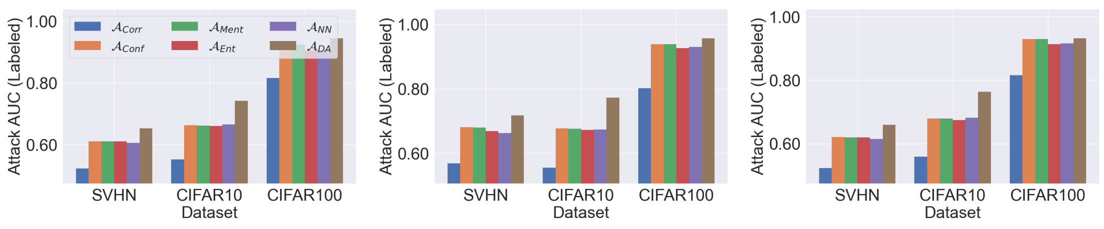
Defense	None	ES	Top-k	MS	DP-SGD
Test ACC	0.530	0.490	0.530	0.549	0.034
Attack AUC	0.918	0.695	0.906	0.905	0.598

Algorithm 1: Our Data Augmentation Based Attack $\mathcal{A}_{DA}$

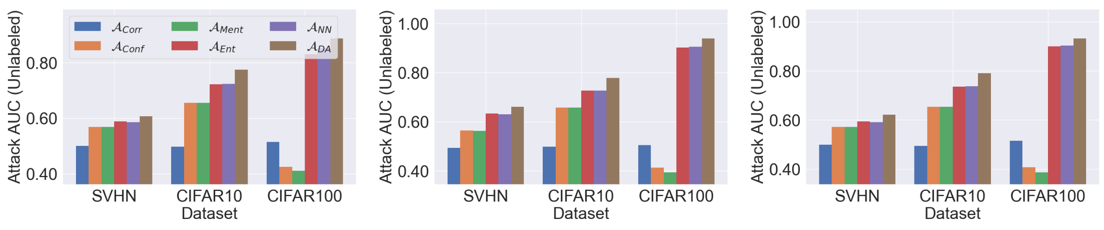
```
1  **Input:** Given sample $x$, target model $\mathcal{T}$, NN-based attack model $\mathcal{A}_{DA}$, weak data augmentations $\mathcal{A}ug_{weak}$, strong data augmentations $\mathcal{A}ug_{strong}$, similarity function $Sim$, number of augmented views $K$
2  **Output:** Member or non-member
   /* Generate augmented views. */
3  $\{x^1_{weak}, x^2_{weak}, \cdots, x^K_{weak}\} \leftarrow \mathcal{A}ug_{weak}(x, K)$
4  $\{x^1_{strong}, x^2_{strong}, \cdots, x^K_{strong}\} \leftarrow \mathcal{A}ug_{strong}(x, K)$
   /* Query the target model and obtain posteriors. */
5  $\{p^1_{weak}, p^2_{weak}, \cdots, p^K_{weak}\} \leftarrow \{\mathcal{T}(x^1_{weak}), \mathcal{T}(x^2_{weak}), \cdots, \mathcal{T}x^K_{weak}\}$
6  $\{p^1_{strong}, p^2_{strong}, \cdots, p^K_{strong}\} \leftarrow \{\mathcal{T}(x^1_{strong}), \mathcal{T}(x^2_{strong}), \cdots, \mathcal{T}x^K_{strong}\}$
   /* Obtain similarity vectors. */
7  Similarity vector $v_w(x) \leftarrow$ SORTED($\{Sim(p^i_{weak}, p^j_{weak}) | i \in [1,K], j \in [1,K]\}$)
8  Similarity vector $v_s(x) \leftarrow$ SORTED($\{Sim(p^i_{strong}, p^j_{strong}) | i \in [1,K], j \in [1,K]\}$)
9  Similarity vector
     $v_{ws}(x) \leftarrow$ SORTED($\{Sim(p^i_{weak}, p^j_{strong}) | i \in [1,K], j \in [1,K]\}$)
   /* Concatenate similarity vectors and perform the attack. */
10 Merged vector $v(x) \leftarrow$ CONCATENATE($v_w(x), v_s(x), v_{ws}(x)$)
11 **Return** $\mathcal{A}_{DA}(v(x))$
```



(a) Attack AUC

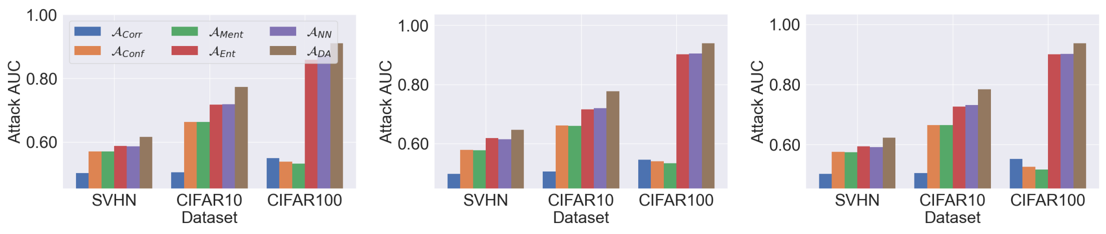


(b) Attack AUC (Labeled)

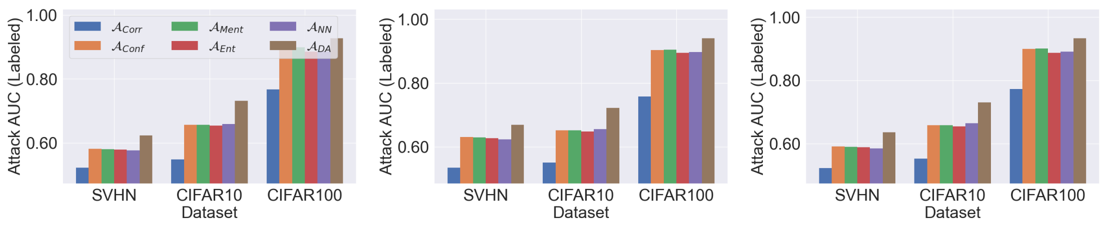


(c) Attack AUC (Unlabeled)

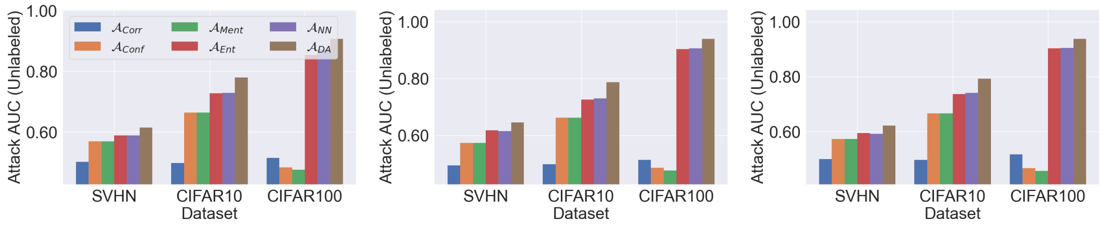
Figure 3.8: The AUC of membership inference attacks against models trained by different SSL methods with 1,000 label samples. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA, respectively.



(a) Attack AUC

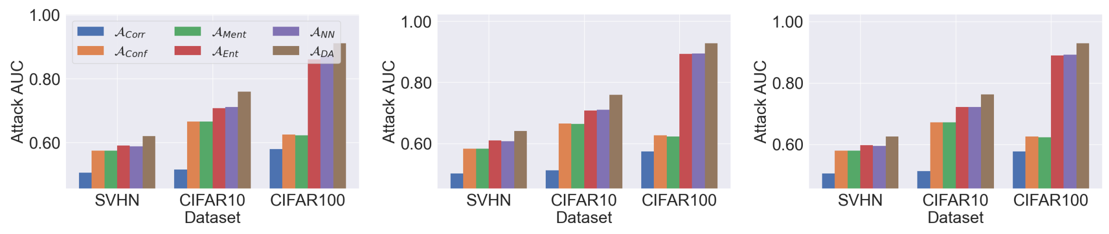


(b) Attack AUC (Labeled)

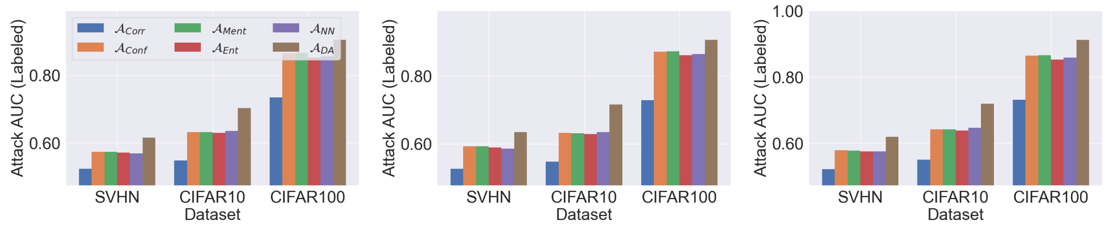


(c) Attack AUC (Unlabeled)

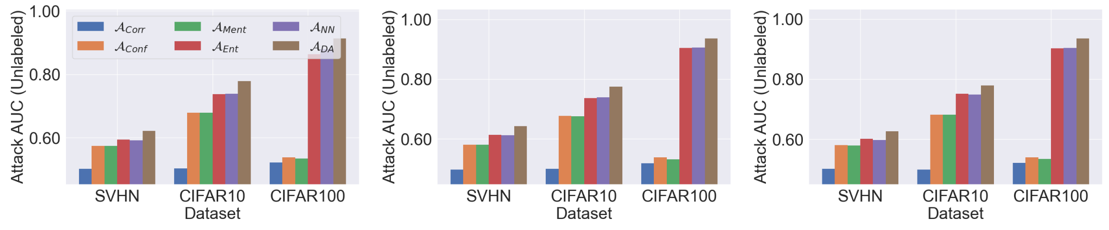
Figure 3.9: The AUC of membership inference attacks against models trained by different SSL methods with 2,000 label samples. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA, respectively.



(a) Attack AUC

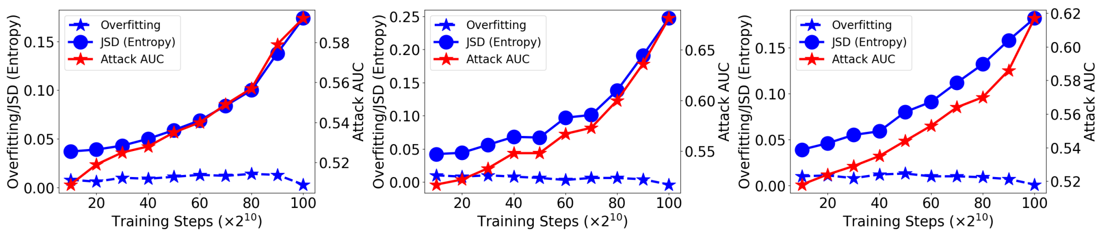


(b) Attack AUC (Labeled)

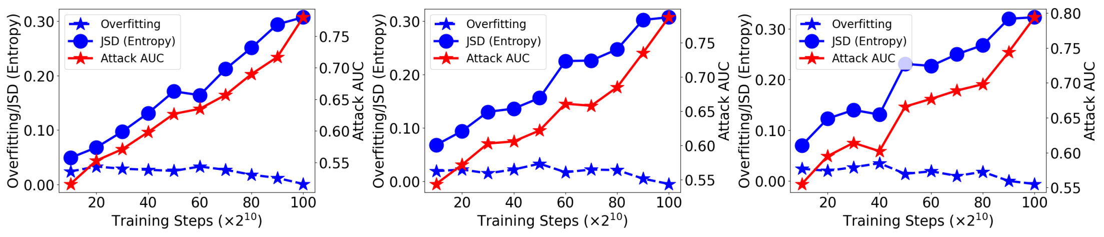


(c) Attack AUC (Unlabeled)

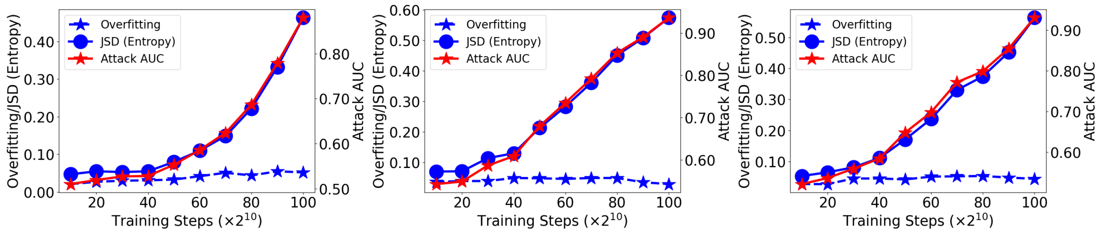
Figure 3.10: The AUC of membership inference attacks against models trained by different SSL methods with 4,000 label samples. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA, respectively.



(a) SVHN

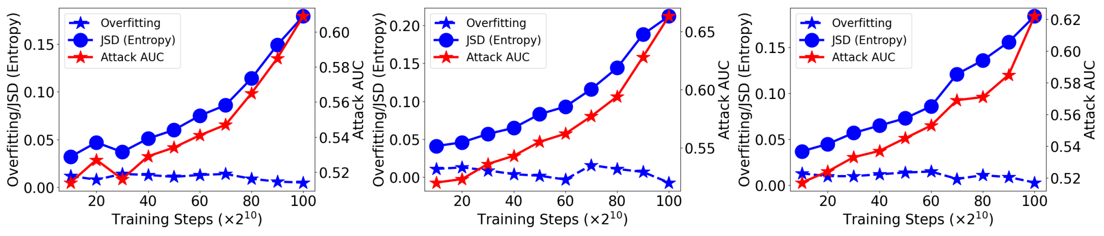


(b) CIFAR10

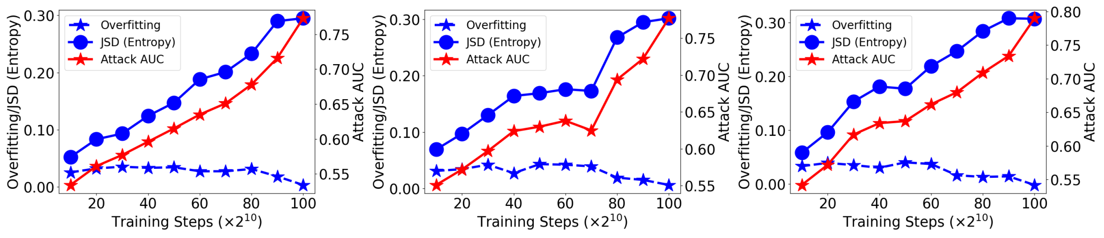


(c) CIFAR100

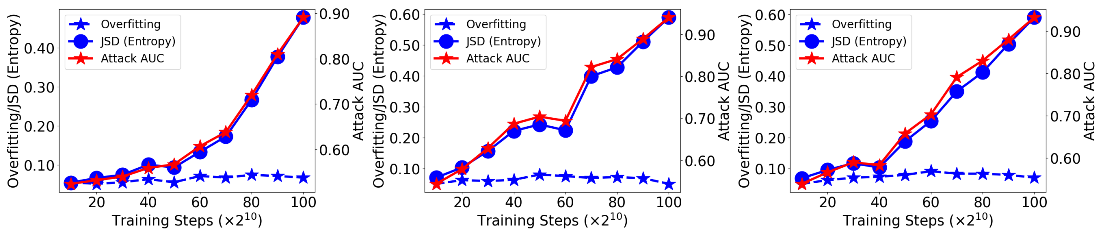
Figure 3.11: The overfitting/JS Distance (Entropy) and attack AUC with respect to different training steps. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA with 500 labeled samples, respectively. Note that we consider the attack AUC of $\mathcal{A}_{DA}$, which is the strongest attack.



(a) SVHN

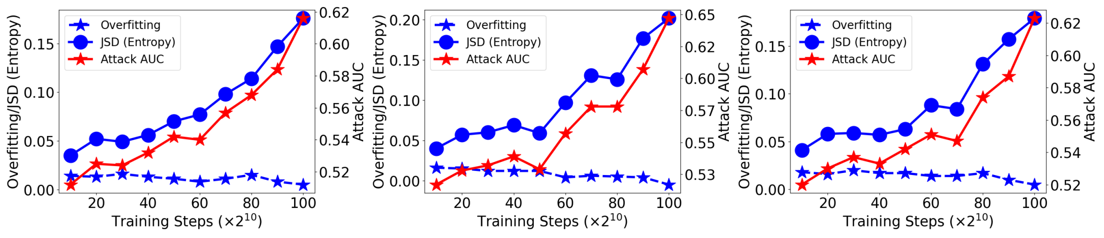


(b) CIFAR10

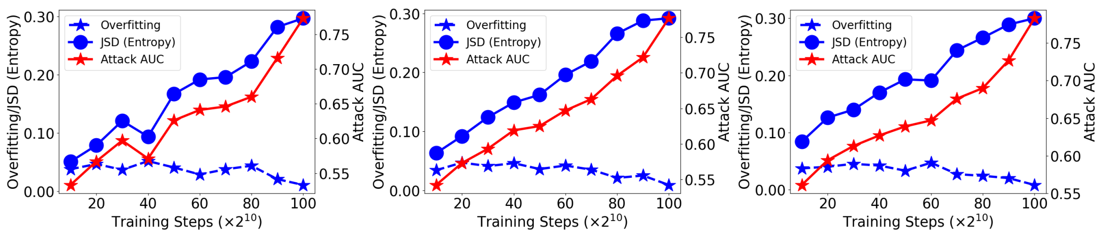


(c) CIFAR100

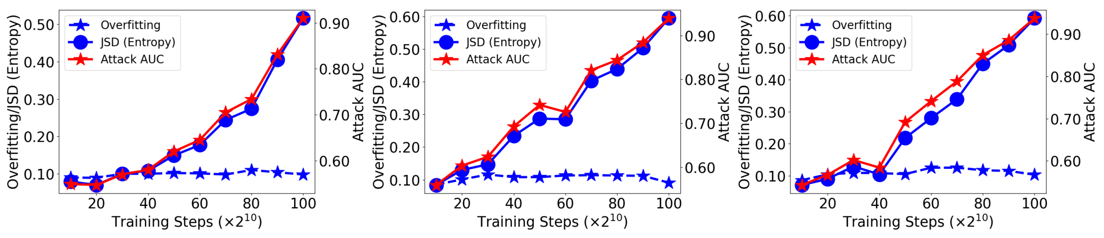
Figure 3.12: The overfitting/JS Distance (Entropy) and attack AUC with respect to different training steps. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA with 1,000 labeled samples, respectively. Note that we consider the attack AUC of $\mathcal{A}_{DA}$, which is the strongest attack.



(a) SVHN

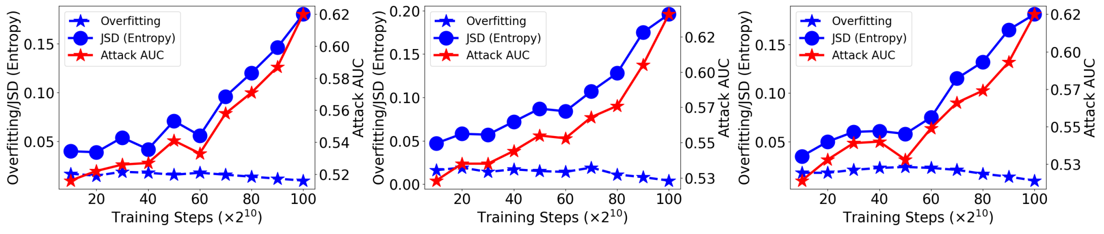


(b) CIFAR10

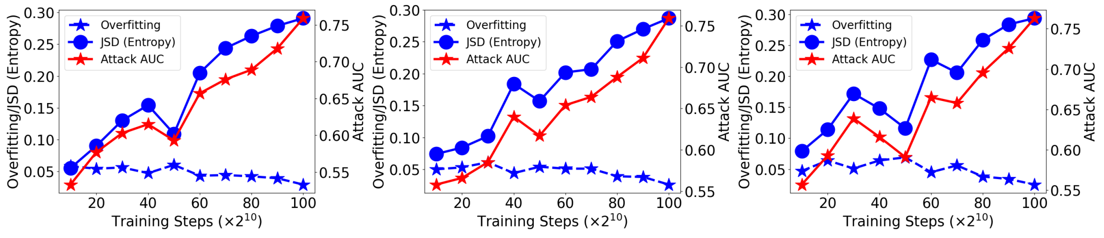


(c) CIFAR100

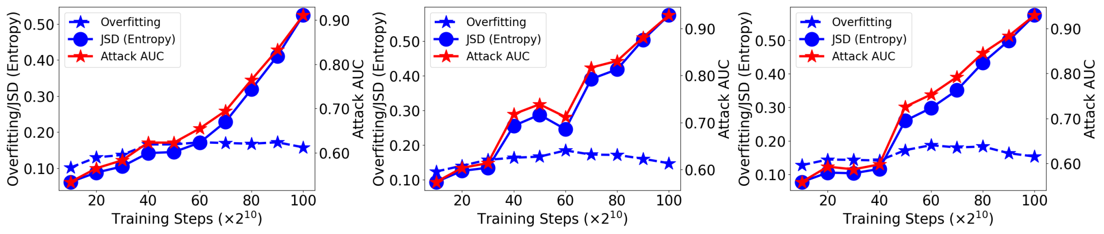
Figure 3.13: The overfitting/JS Distance (Entropy) and attack AUC with respect to different training steps. The first to third columns denotes the model trained by FixMatch, FlexMatch, and UDA with 2,000 labeled samples, respectively. Note that we consider the attack AUC of $\mathcal{A}_{DA}$, which is the strongest attack.



(a) SVHN

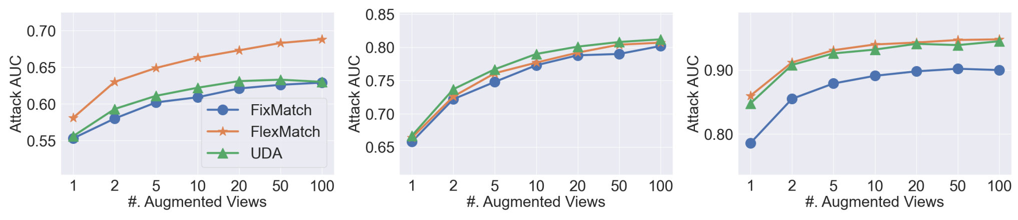


(b) CIFAR10



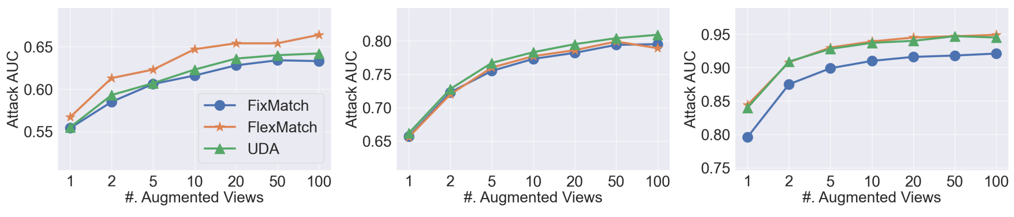
(c) CIFAR100

Figure 3.14: The overfitting/JS Distance (Entropy) and attack AUC with respect to different training steps. The first to third columns denote the model trained by FixMatch, FlexMatch, and UDA with 4,000 labeled samples, respectively. Note that we consider the attack AUC of $\mathcal{A}_{DA}$, which is the strongest attack.



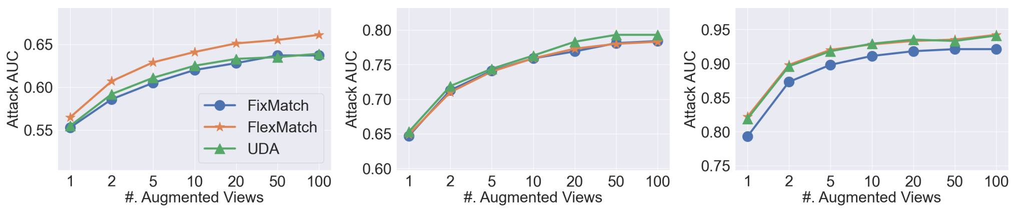
(a) SVHN **(b)** CIFAR10 **(c)** CIFAR100

Figure 3.15: The attack AUC of $\mathcal{A}_{DA}$ with different numbers of augmented views to query the target model. The target model is trained with 1,000 labeled samples.



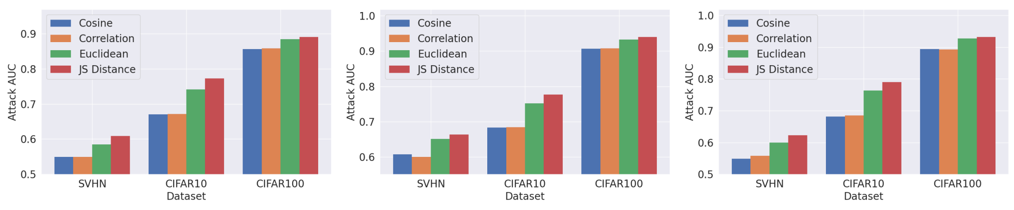
(a) SVHN **(b)** CIFAR10 **(c)** CIFAR100

Figure 3.16: The attack AUC of $\mathcal{A}_{DA}$ with different numbers of augmented views to query the target model. The target model is trained with 2,000 labeled samples.



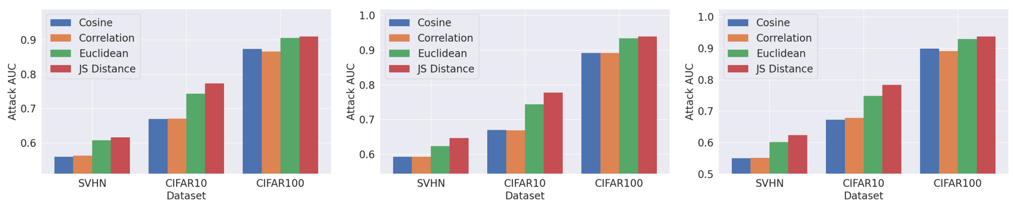
(a) SVHN **(b)** CIFAR10 **(c)** CIFAR100

Figure 3.17: The attack AUC of $\mathcal{A}_{DA}$ with different numbers of augmented views to query the target model. The target model is trained with 4,000 labeled samples.



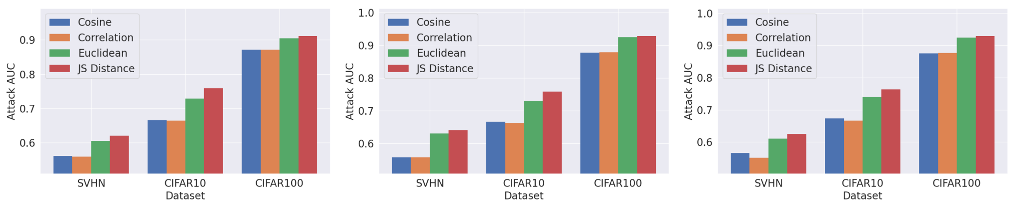
(a) FixMatch **(b)** FlexMatch **(c)** UDA

Figure 3.18: The attack AUC of $\mathcal{A}_{DA}$ with different similarity functions. The target model is trained with 1,000 labeled samples.



(a) FixMatch **(b)** FlexMatch **(c)** UDA

Figure 3.19: The attack AUC of $\mathcal{A}_{DA}$ with different similarity functions. The target model is trained with 2,000 labeled samples.



(a) FixMatch **(b)** FlexMatch **(c)** UDA

Figure 3.20: The attack AUC of $\mathcal{A}_{DA}$ with different similarity functions. The target model is trained with 4,000 labeled samples.

4 Self-supervised Learning

4.1 Introduction

Machine learning (ML) has progressed tremendously, and data is the key factor to drive such development. However, high-quality data, in particular labeled data, is often hard and expensive to collect as this relies on large-scale human annotation. Meanwhile, unlabeled data is being generated at every moment. To leverage unlabeled data for machine learning tasks, *self-supervised learning* has been introduced [72]. The goal of self-supervised learning is to derive labels from an unlabeled dataset and train an unsupervised task in a supervised manner. A trained self-supervised model serves as an encoder transforming data samples into their representations which are then used to perform supervised downstream ML tasks. One of the most prominent self-supervised learning paradigms is *contrastive learning* [43, 82, 53, 17, 119, 48, 60], with SimCLR [17] as its most representative framework [72].

Different from supervised learning which directly optimizes an ML model on a labeled training dataset, referred to as a supervised model, contrastive learning aims to train a contrastive model, which is able to generate expressive representations for data samples, and relies on such representations to perform downstream supervised ML tasks. The optimization objective for contrastive learning is to map different views derived from one training sample (e.g., through data augmentation) closer in the representation space while different views derived from different training samples more distant. By doing this, a contrastive model is capable of representing each sample in an informative way.

Recently, machine learning models have been demonstrated to be vulnerable to various privacy attacks against their training dataset [97, 101, 76, 94, 47, 102, 15, 13, 50]. The two most representative attacks in this domain are membership inference attack [97, 94] and attribute/property inference attack [76, 102]. The former aims to infer whether a data sample is part of a target ML model's training dataset. The latter leverages the overlearning property of a machine learning model to infer the sensitive attribute of a data sample. So far, most of the research on the privacy of machine learning concentrates on supervised models. Meanwhile, informative representations for data samples learned by contrastive models may cause severe privacy risks as well. To the best of our knowledge, this has been left largely unexplored.

Our Contributions: In this work, we perform the first privacy quantification of contrastive learning, the most representative self-supervised learning paradigm. More specifically, we study the privacy risks of data samples in the contrastive learning setting, with a focus on SimCLR, through the lens of membership inference and attribute inference, and we concentrate on contrastive models trained on image datasets.

We adapt the existing attack methodologies for membership inference (neural network-based, metric-based, and label-only) and attribute inference against supervised models to contrastive models. Our empirical results show that contrastive models are less vulnerable to membership inference attacks than supervised models. For instance, considering the neural network-based attacks, we achieve 0.620 membership inference accuracy on a contrastive model trained on STL10 [21] with ResNet-50 [49], while the result is 0.810 on the corresponding supervised model. The reason behind this is contrastive models are less prone to overfitting.

On the other hand, we observe that contrastive models are more vulnerable to attribute inference attacks than supervised models. For instance, on the UTKFace [127] dataset with ResNet-18, we can achieve 0.701 attribute inference attack accuracy on the contrastive model while only 0.422 on the supervised model. This is due to the fact that the representations generated by a contrastive model contain rich and expressive information about their original data samples, which can be exploited for effective attribute inference.

To mitigate the attribute inference risks stemming from contrastive models, we propose the first privacy-preserving contrastive learning mechanism, namely *Talos*, relying on adversarial training. Concretely, *Talos* introduces an adversarial classifier into the original contrastive learning framework to censor the sensitive attributes learned by a contrastive model. Our evaluation reveals that *Talos* can successfully mitigate attribute inference risks for contrastive models while maintaining their membership privacy and model utility. Our code and models will be made publicly available.

In summary, we make the following contributions:

- We take the first step towards quantifying the privacy risks of contrastive learning.
- Our empirical evaluation shows that contrastive models trained on image datasets are less vulnerable to membership inference attacks but more prone to attribute inference attacks compared to supervised models.
- We propose the first privacy-preserving contrastive learning mechanism, which is able to protect the trained contrastive models from attribute inference attacks without jeopardizing their membership privacy and model utility.

4.2 Membership Inference Attack

We first quantify the privacy risks of contrastive models through the lens of membership inference. Note that our goal here is not to propose a novel membership inference attack, instead, we aim to quantify the membership privacy of contrastive models. Therefore, we follow existing attacks and their threat models [97, 94, 103, 69, 20].

4.2.1 Attack Definition and Threat Model

Membership inference attack is one of the most popular privacy attacks against ML models [97, 94, 58, 47, 15, 66, 69, 103, 20, 16]. The goal of membership inference is to determine whether a data sample x is part of the training dataset of a target model $\mathcal{T}$. We formally define a membership inference attack model $\mathcal{A}_{MemInf} : x, \mathcal{T} \mapsto \{member, non\text{-}member\}$. Here, the target model is the contrastive model introduced in Section 2.3. A successful membership inference attack can cause severe privacy risks. For instance, if a model is trained on data samples collected from people with certain sensitive information, then successfully inferring a sample from a person being a member of the model can directly reveal the person's sensitive information.

Following previous work [97, 94, 103, 69, 20], we assume that an adversary only has black-box access to the target model $\mathcal{T}$, i.e., they can only query $\mathcal{T}$ with their data samples and obtain the outputs. In addition, the adversary also has a shadow dataset $\mathcal{D}_{shadow}$, which comes from the same distribution as the target model's training dataset. The shadow dataset $\mathcal{D}_{shadow}$ is used to train a shadow model $\mathcal{S}$, the goal of which is to obtain the necessary information to perform the attack. We further assume that the shadow model shares the same architecture as the target model [97]. This is realistic as the adversary can use the same machine learning service as the target model owner to train their shadow model. Alternatively, the adversary can also learn the target model's architecture first by applying model extraction attacks [107, 109, 81, 83].

4.2.2 Methodology

We adapt the previous membership inference attacks, which are designed for supervised models, to contrastive models [97, 94, 20, 103]. Concretely, we consider three types of membership inference attacks, i.e., NN-based attacks [97, 94], metric-based attacks [103], and label-only attacks [20].

NN-based Attacks (Neural Network-based Attacks): In NN-based attacks, the adversary aims to train an attack model to differentiate members and non-members using the posteriors generated from the target model and their predicted labels. Given a shadow dataset $\mathcal{D}_{shadow}$, the adversary first splits it into two disjoint sets, namely shadow training dataset $\mathcal{D}^{train}_{shadow}$ and shadow testing dataset $\mathcal{D}^{test}_{shadow}$. $\mathcal{D}^{train}_{shadow}$ is used to train the shadow model $\mathcal{S}$, which mimics the behavior of the target model. This means the shadow model is trained to perform the same task as the target model. Then, the adversary uses $\mathcal{D}_{shadow}$ (including both $\mathcal{D}^{train}_{shadow}$ and $\mathcal{D}^{test}_{shadow}$) to query the shadow model $\mathcal{S}$ and obtains the corresponding posteriors and prediction labels. For each data sample in $\mathcal{D}_{shadow}$, the adversary ranks its posteriors in descending order and takes the largest two posteriors (classification tasks considered in this work have at least two classes) as part of the input to the attack model. The other part is an indicator representing whether the prediction is correct or not. Thus, the dimension of the input to $\mathcal{A}_{MemInf}$ is 3. If a sample belongs to $\mathcal{D}^{train}_{shadow}$, the adversary labels its corresponding input to the attack model as a member, otherwise as a non-member. Then, this obtained dataset is used to train the attack model, which is a binary machine learning classifier. To determine whether a target data sample x is used to train the target model, the adversary first queries the target model $\mathcal{T}$ with x and obtains the input to the attack model for this sample. Then, the adversary queries this input to the attack model and gets its membership prediction.

Metric-based Attacks: Song and Mittal [103] propose several metric-based attacks. Similar to NN-based attacks, metric-based attacks need to train shadow models. However, instead of training an attack model, metric-based attacks leverage a certain metric and a predefined threshold on that metric (calculated over the shadow model) to determine a sample's membership status. Song and Mittal [103] propose four metrics, i.e., prediction correctness (metric-corr), prediction confidence (metric-conf), prediction entropy (metric-ent), and modified prediction entropy (metric-ment).

Label-only Attacks: Label-only attacks [20] consider a more restrict scenario where the target model only exposes the predicted label instead of posteriors. Similar to

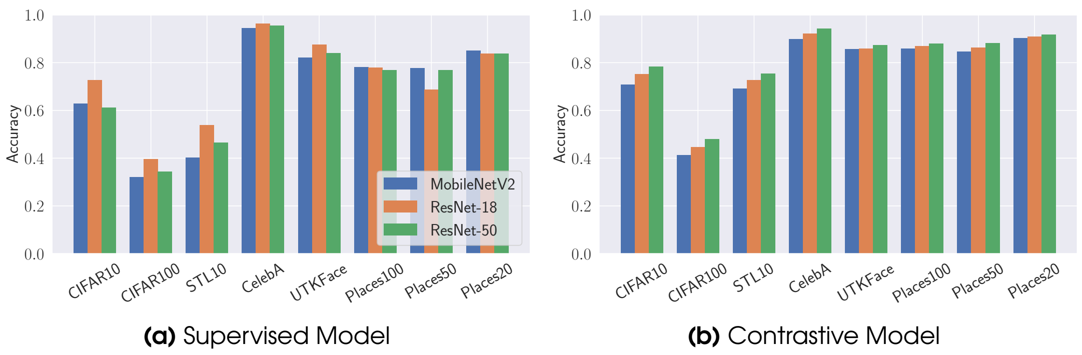


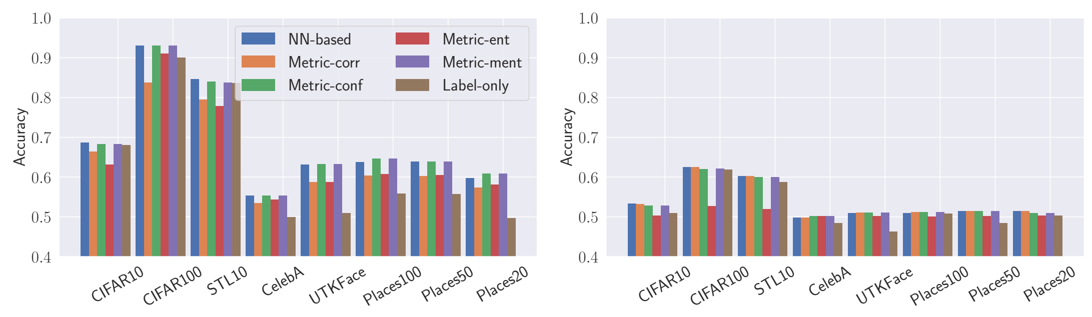
Figure 4.1: The performance of original classification tasks for both supervised models and contrastive models with MobileNetV2, ResNet-18, and ResNet-50 on 8 different datasets. The x-axis represents different datasets. The y-axis represents original classification tasks' accuracy.

previous attacks, this attack requires the adversary to train a shadow model. Label-only attacks focus more on the input samples instead of the model's outputs, relying on the adversarial example techniques. The key intuition is that the magnitude of perturbation to change the predicted label of member samples is larger than that of non-member samples. The adversary can exploit the magnitude of the perturbation to distinguish members and non-members.

4.2.3 Experimental Settings

Datasets: We utilize 8 different image datasets to conduct our experiments for membership inference.

- **CIFAR10 [2].** This dataset contains 60,000 images in 10 classes. Each class represents one object and has 6,000 images. The size of each image is 32×32.
- **CIFAR100 [2].** This dataset is similar to CIFAR10, except it has 100 classes, with each class containing 600 images. The size of each image is also 32×32.
- **STL10 [21].** This dataset is composed of 10 classes of images. Each class has 1,300 samples. The size of each image is 96×96. Besides the labeled image, STL10 also contains 100,000 unlabeled images, which we use for pretraining the encoder for the contrastive model (detailed later). These images are extracted from a broader distribution compared to those with labeled classes.
- **CelebA [74].** This dataset is composed of more than 200,000 celebrities' facial images. Note that in CelebA, we randomly select 60,000 images for our experiments. We set its target model's classification task as gender classification.
- **UTKFace [127].** This dataset consists of over 23,000 facial images labeled with gender, age, and race. We set its target model's classification task as gender classification as well.



(a) Supervised Model **(b)** Contrastive Model

Figure 4.2: The performance of different membership inference attacks against both supervised models and contrastive models with MobileNetV2 on 8 different datasets. The x-axis represents different datasets. The y-axis represents membership inference attacks' accuracy.

- **Places365 [129].** This dataset is composed of more than 1.8 million images with 365 scene categories. We randomly select 100 scene categories and randomly select 400 images per category to form the **Places100** dataset. Besides, we randomly select 50 (20) scene categories and randomly select 800 (2,000) images per category to form the **Places50** (**Places20**) dataset. Each dataset contains 40,000 images in total. We follow Song and Shmatikov [102] and set its target model's classification task as predicting whether the scene is indoor or outdoor.

All the datasets are used to evaluate membership inference attacks, while UTKFace, Places100, Places50, and Places20 are also used to evaluate attribute inference attacks since they have extra labels that can be used as sensitive attributes (see Section 4.3.3). For all the datasets, we rescale their images to the size of 96×96. Note that we concentrate on image datasets as it is the most prominent domain for applying contrastive learning at the moment [48, 43, 82, 17, 119, 60]. We leave our investigation in other data domains as future work.

Datasets Configuration: For each dataset, we first split it into four equal parts, i.e., $\mathcal{D}^{train}_{target}$, $\mathcal{D}^{test}_{target}$, $\mathcal{D}^{train}_{shadow}$, and $\mathcal{D}^{test}_{shadow}$. $\mathcal{D}^{train}_{target}$ is used to train the target model $\mathcal{T}$, the samples of which are thus considered as members of the target model. We treat $\mathcal{D}^{test}_{target}$ as non-members of the target model $\mathcal{T}$. $\mathcal{D}^{train}_{shadow}$ is used to train the shadow model $\mathcal{S}$, and $\mathcal{D}^{train}_{shadow}$ and $\mathcal{D}^{test}_{shadow}$ are used to create the attack model $\mathcal{A}_{MemInf}$.

Metric: Since the attack model's training and testing datasets are both balanced with respect to membership distribution, we adopt accuracy as our evaluation metric following previous work [97, 94].

Attack Model: For NN-based attacks, the attack model is a 3-layer MLP, and the number of neurons for each hidden layer is set to 32. We use cross-entropy as the loss function and Adam as the optimizer with a learning rate of 0.05. The attack model is trained for 100 epochs. For metric-based attacks, we follow the implementation of Song et al. [103]. For label-only attacks, we leverage the implementation of ART [3].

Target Model (Contrastive Model): We adopt three popular neural network architectures as the contrastive model's base encoder f in our experiments, including MobileNetV2 [95], ResNet-18 [49], and ResNet-50 [49]. Specifically, we discard the last classification layer of MobileNetV2, ResNet-18, and ResNet-50 and use the remaining parts as f. Then, a 2-layer MLP is added after f as the projection head g. For ResNet-18, the dimensions for the output of f, the first-layer of g, and the second-layer of g are set to 512, 512, and 256, respectively. For ResNet-50, the corresponding dimensions are 2,048, 256, and 256. For MobileNetV2, the corresponding dimensions are 1,280, 256, and 256.

After training the base encoder with the contrastive loss, we ignore the projection head g and add a new linear layer to the base encoder f as its classification layer. For all datasets, we first use the unlabeled dataset of STL10 to pretrain the base encoder f for 100 epochs. Then, we fine-tune the base encoder f with the corresponding training dataset (without label) for 100 epochs. In the end, we freeze the parameters of f and use the corresponding training dataset to only fine-tune the classification layer for 100 epochs to establish the contrastive model. In all cases, Adam is utilized as the optimizer.

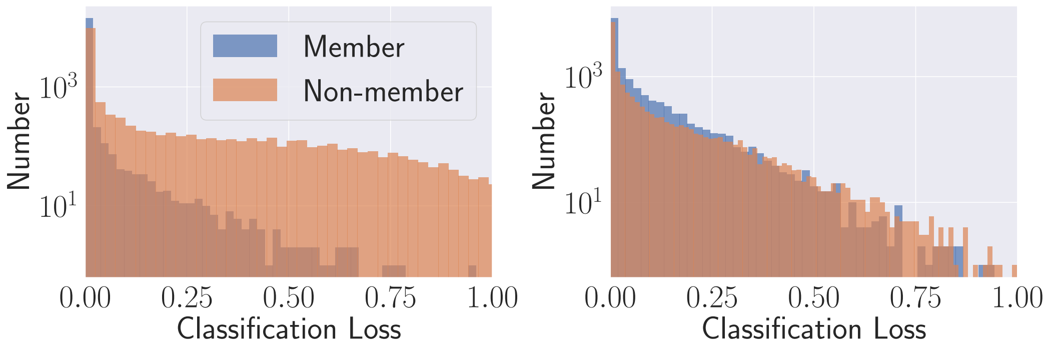
Baseline (Supervised Model): To fully understand the privacy leakage of contrastive models, we further use supervised models as the baseline. We train three models including MobileNetV2, ResNet-18, and ResNet-50 from scratch for all the datasets. The models are trained for 100 epochs. Cross-entropy is adopted as the loss function, and we again use Adam as the optimizer. Our code is currently implemented in Python 3.6 and PyTorch 1.6.0, and run on an NVIDIA DGX-A100 server with Ubuntu 18.04.

4.2.4 Results

We first show the performance of supervised models and contrastive models on their original classification tasks in Figure 4.1. We observe that contrastive models perform better than supervised models on most of the datasets. For instance, on STL10 with ResNet-18 as the base encoder, the contrastive model achieves 0.726 accuracy while the supervised model achieves 0.538 accuracy.

Regarding membership inference against supervised models and contrastive models, the results for MobileNetV2 are shown in Figure 4.2. We also summarize the results for ResNet-18 (Figure 4.15) and ResNet-50 (Figure 4.16) in Appendix. In Figure 4.2, we see that all the supervised models have higher attack accuracy than the contrastive models. E.g., when the supervised model is MobileNetV2 trained on CIFAR100, the accuracy of NN-based attack is 0.931, while the accuracy for the corresponding contrastive model is only 0.625.

We observe that NN-based, metric-conf, and metric-ment attacks achieve the best performance in all cases. The reason metric-conf and metric-ment achieving better performance than metric-corr and metric-ent is that metric-conf and metric-ment consider both prediction correctness and confidence while metric-corr (metric-ent) only considers prediction correctness (confidence). Interestingly, for supervised models, metric-corr and metric-ent perform similarly, while for contrastive models, metric-ent is worse than metric-corr. This reason is that the posteriors generated by contrastive models are more smooth compared to supervised model, which makes it harder to



(a) Supervised Model **(b)** Contrastive Model

Figure 4.3: The distribution of loss with respect to original classification tasks for member and non-member samples for both the supervised model and the contrastive model with ResNet-18 on CIFAR10. The x-axis represents each sample's classification loss. The y-axis represents the number of member and non-member samples.

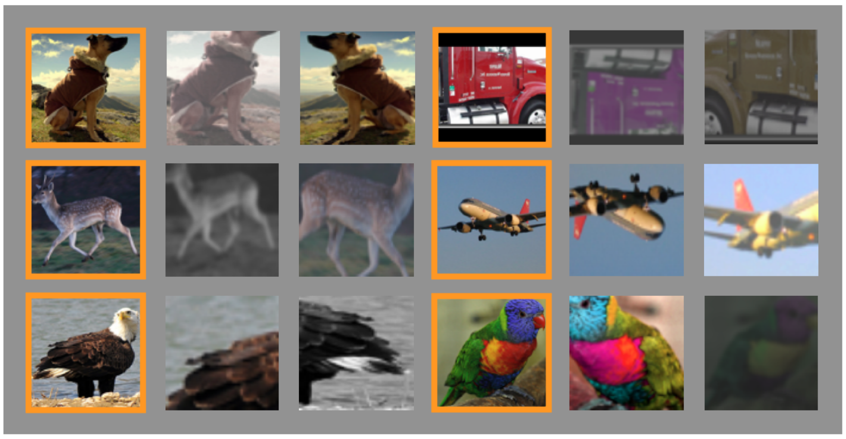
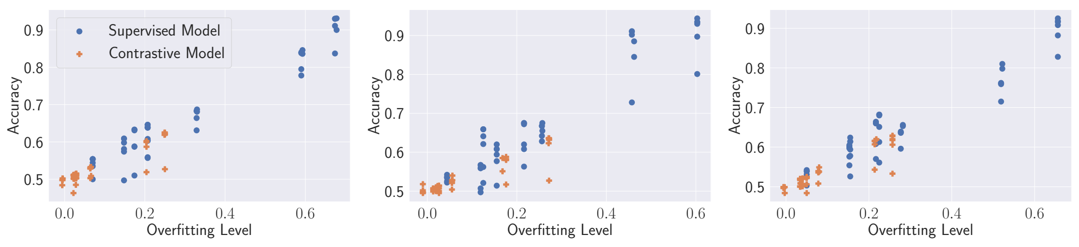


Figure 4.4: Randomly selected images from STL10 and their augmented views used during the process of contrastive learning. The first and fourth columns show the original images (bounded by orange boxes), and the rest columns show their augmented views.

distinguish members and non-members through the posterior entropy. Label-only attacks perform worse than NN-based attacks. This is expected since the adversary has less information in these cases. Note that label-only attacks do not perform well on binary classifiers, we will investigate the reason in the future.[1]

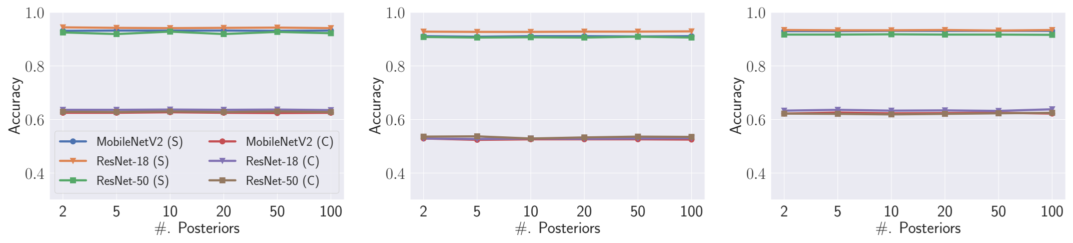
To further investigate why contrastive models are less vulnerable to membership inference, we analyze the loss distribution between members and non-members in both supervised models and contrastive models. Due to space limitations, we only show the results of ResNet-18 trained on the CIFAR10 dataset in Figure 4.3. A clear trend is

[1]Choquette-Choo et al. [20] also only perform label-only membership inference attacks against datasets with more than two classes.



(a) MobileNetV2 **(b)** ResNet-18 **(c)** ResNet-50

Figure 4.5: The performance of membership inference attacks against both supervised models and contrastive models with MobileNetV2, ResNet-18, and ResNet-50 on 5 different datasets under different overfitting levels. The x-axis represents different overfitting levels. The y-axis represents membership inference attacks' accuracy.

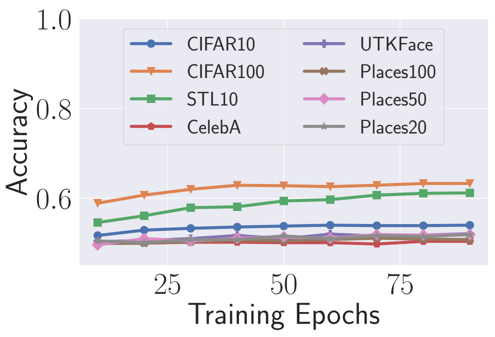


(a) NN-based **(b)** Metric-ent **(c)** Metric-ment

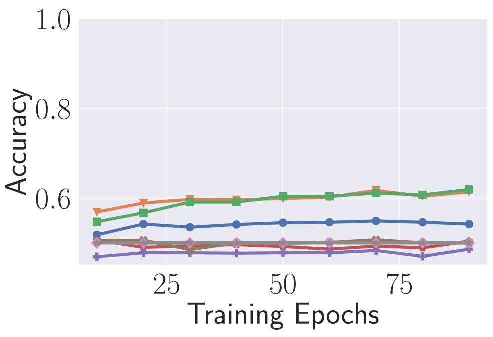
Figure 4.6: The performance of NN-based, metric-ent, and metric-ment attacks against both supervised models and contrastive models with MobileNetV2, ResNet-18, and ResNet-50 on CIFAR100 under different numbers of posteriors given by the target models. (S) and (C) denotes the supervised and contrastive models, respectively. The x-axis represents different numbers of posteriors. The y-axis represents membership inference attacks' accuracy. Note that we do not report the performance of metric-corr, metric-conf, and label-only attacks since the number of posteriors does not affect their performance.

that compared to the contrastive model, the supervised model has a larger divergence between the classification loss (cross-entropy) for members and non-members. Recall that contrastive learning uses two augmented views of each sample in each epoch to train its base encoder and the original sample to train its classification layer. This indicates that each sample is generalized to multiple views during the contrastive model training process. In this way, the contrastive model reduces its memorization of the original sample itself.

Interestingly, Song et al. [104] observe that defense mechanisms for mitigating adversarial examples [9, 88, 106, 14] increase the membership inference performance. This means such defense and contrastive learning have different effects on membership privacy. On the one hand, these defense mechanisms for adversarial examples use original samples and their visually imperceptible adversarial examples to train a model; in this way, the model learns to remember each original sample more accurately. On the other hand, the augmented samples in contrastive learning are very different from their original samples (see Figure 4.4 for some examples). Therefore, membership inference is



(a) NN-based



(b) Label-only

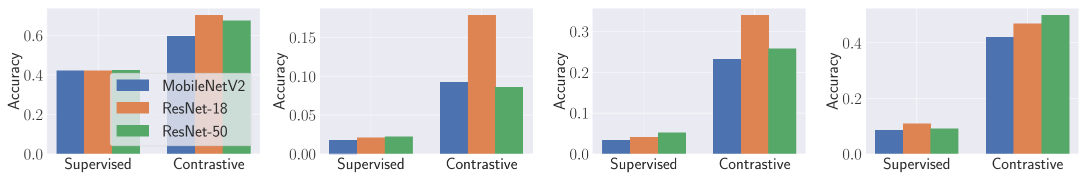
Figure 4.7: The performance of NN-based and label-only membership inference attacks against contrastive models with ResNet-50 on 8 different datasets under different numbers of epochs for classification layer training. The x-axis represents different numbers of epochs. The y-axis represents membership inference attacks' accuracy. Each line corresponds to a specific dataset.

less effective against contrastive models.

We notice that the attack performance varies on different models and different datasets. We relate this to the different overfitting levels. Similar to previous work [97, 94], we measure the overfitting level of a target model by calculating the difference between its training accuracy and testing accuracy. In Figure 4.5, we see that the overfitting level is highly correlated with the attack performance: if a model is more overfitted, it is more vulnerable to membership inference attacks. For instance, in Figure 4.5a, on CIFAR100, the contrastive model (upper right orange cross) has an overfitting level of 0.249, and the NN-based attack accuracy is 0.625, while the supervised model (upper right blue dot) has a larger overfitting level (0.678) and higher attack accuracy (0.931). Another observation is that compared to the supervised models, the overfitting levels of the contrastive models reside in a smaller range.

NN-based method as well as some of the metric-based ones (metric-ent and metric-ment) require the target model to provide posteriors to launch the attacks. We further investigate whether the number of posteriors provided by the target model can influence the attack performance. Concretely, we vary the number of posteriors from 2 to 100 on CIFAR100 for both supervised and contrastive models. Figure 4.6 shows that the number of posteriors does not have a strong influence on the attack performance. We further measure the influence of the number of epochs used for training each contrastive model's classification layer on the attack performance. Figure 4.7 shows that the attack accuracy is rather stable (the performance of metric-based attacks are summarized in Figure 4.17 in Appendix). These results show that contrastive models consistently reduce the membership threat.

In conclusion, contrastive models are less vulnerable to membership inference attacks compared to supervised models. The reason is that contrastive models are less overfitted to their training samples than supervised models due to the design of the contrastive learning paradigm.



(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.8: The performance of attribute inference attacks against both supervised models and contrastive models with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents attribute inference attacks' accuracy.

4.3 Attribute Inference Attack

In this section, we take a different angle to measure the privacy risks of contrastive learning using attribute inference attack [76, 102]. Similar to membership inference attacks, we use existing attribute inference attacks [76, 102] to measure the contrastive model's privacy risks instead of inventing new methods.

4.3.1 Attack Definition and Threat Model

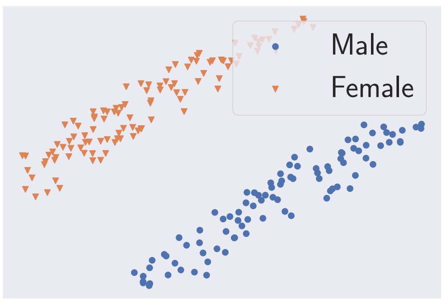
In attribute inference, the adversary's goal is to infer a specific sensitive attribute of a data sample from its representation generated by a target model. This sensitive attribute is not related to the target ML model's original classification task. For instance, a target model is designed to classify an individual's age from their social network posts, while attribute inference aims to infer their educational background.

Attribute inference attacks have been successfully performed on supervised models [76, 102]. The reason behind this is the intrinsic overlearning property of ML models. Overlearning means that an ML model trained for a certain task may also learn to represent other characteristics of data samples. Such representation capability, in some cases, can be exploited by an adversary to infer data samples' sensitive attributes.

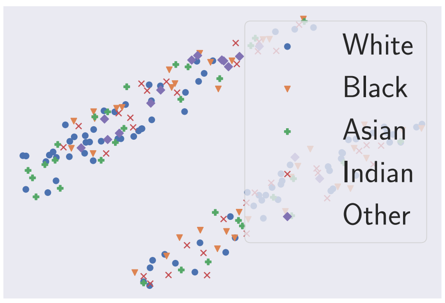
Once a contrastive model is trained, it can generate a representation for each sample with its base encoder f. For a supervised model, we consider the whole model without the classification layer as its base encoder to generate a representation for each sample. Note that the base encoder of contrastive model and supervised model has the same architecture.

For attribute inference, given a data sample's representation from a target model, denoted by h, to conduct the attribute inference attack, the adversary trains an attack model $\mathcal{A}_{AttInf} : h \mapsto s$, where s represents the sensitive attribute.

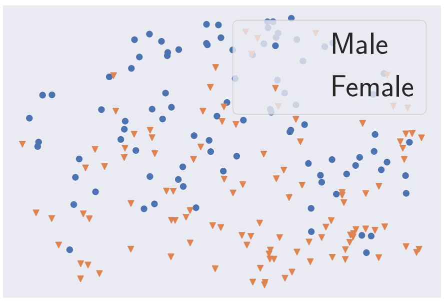
We follow the same threat model as previous work [76, 102]: the adversary only has access to the target sample's embedding (representation), but not the target sample itself. The adversary is also assumed to have a set of samples' embeddings and their sensitive attributes; this dataset is termed as an auxiliary dataset $\mathcal{D}_{aux}$. As shown by previous work, attribute inference can be applied in both federated learning [76] and model partitioning [102] settings.



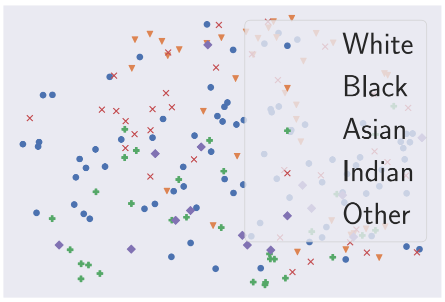
(a) Original Task (S)



(b) Sensitive Attribute (S)



(c) Original Task (C)



(d) Sensitive (C)

Figure 4.9: The representations for 200 randomly selected samples generated by both the supervised model and the contrastive model with ResNet-18 on UTKFace projected into a 2-dimension space using t-SNE. (S) and (C) denotes the supervised and contrastive models, respectively. Each point represents a sample.

4.3.2 Methodology

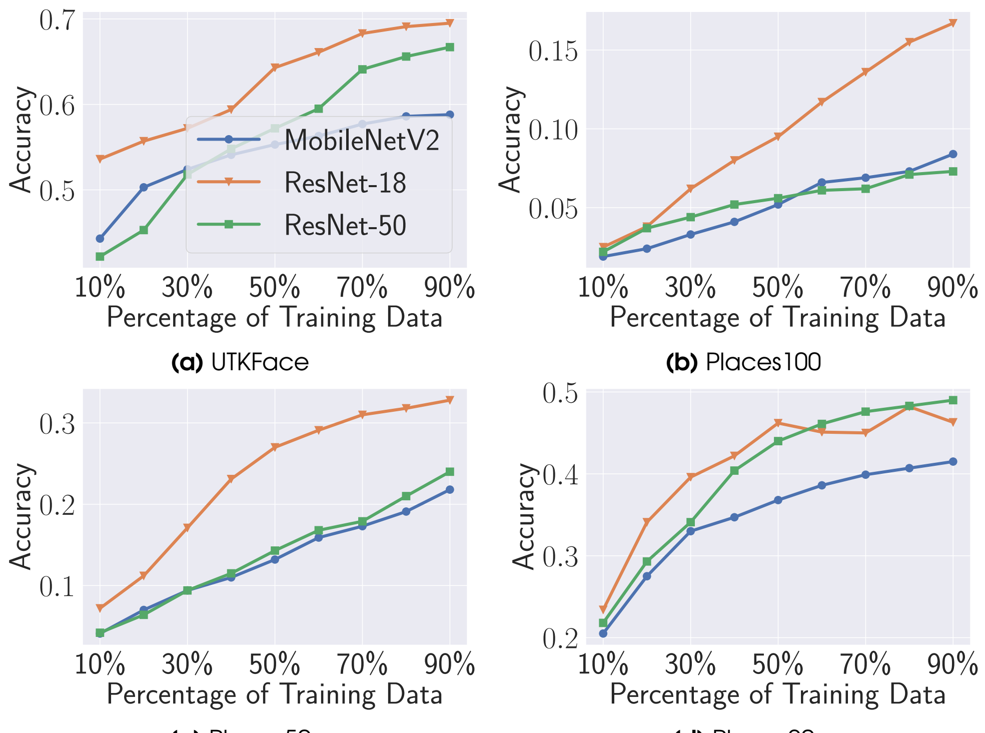
We generalize the methodology of attribute inference attacks against supervised models [76, 102] to contrastive models. The attack process can be partitioned into two stages, i.e., attack model training and attribute inference.

Attack Model Training: For each $(h, s) \in \mathcal{D}_{aux}$, the adversary takes the representation h as the input and the corresponding sensitive attribute s as the label to train the attack model.

Attribute Inference: To determine the sensitive attribute of a data sample's representation h, the adversary queries the attack model $\mathcal{A}_{AttInf}$ with h and obtains the result.

4.3.3 Experimental Setting

Datasets: We utilize UTKFace, Places100, Places50, and Places20 to evaluate attribute inference attacks as they contain extra attributes that can be considered as sensitive attributes for our experiments (see Section 4.2.3). In UTKFace, the target model's



(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.10: The performance of attribute inference attacks against contrastive models on 4 different datasets under different percentages of the attack training dataset. The x-axis represents different percentages of the attack training dataset. The y-axis represents attribute inference attacks' accuracy.

classification task is gender classification, and the sensitive attribute is race (Black, White, Asian, Indian, and Other). In Places100, Places50, and Places20, the target classification task is whether the scene is indoor or outdoor, and the sensitive attribmaxcute is scene categories. Similar to Song and Shmatikov [102], we take $\mathcal{D}_{target}^{train}$ to generate the auxiliary dataset and train the attack model, and take $\mathcal{D}_{target}^{test}$ to test the attack performance.

Metric: We adopt accuracy as the metric to evaluate attribute inference attacks following previous work [76, 102].

Models: All the target models' architectures are the same as those for membership inference attacks. For the attack model, we leverage a 3-layer MLP with the number of neurons in the hidden layer set to 128. We use cross-entropy as the loss function and SGD as the optimizer with a learning rate of 0.01. The attack model is trained for 100 epochs. The dimension of each sample's representation from the base encoder, i.e., the attack model's input, is 1,280 for MobileNetV2, 512 for ResNet-18, and 2,048 for ResNet-50.

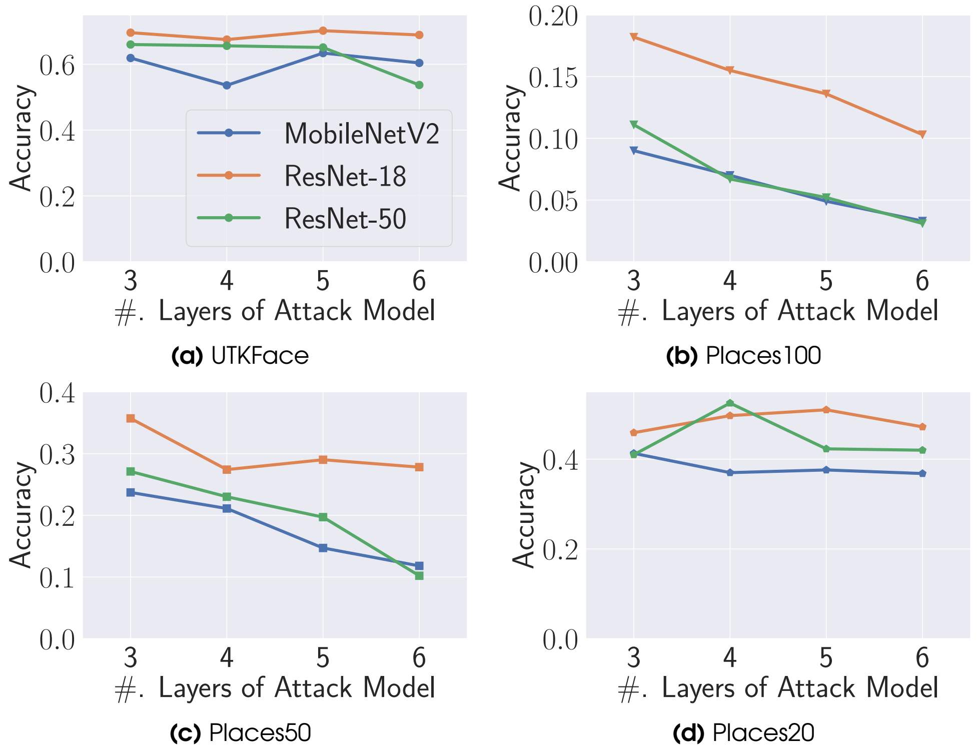


Figure 4.11: The performance of attribute inference attacks against contrastive models on 4 different datasets under attack models with different layers. The x-axis represents attack models' layers. The y-axis represents attribute inference attacks' accuracy.

Table 4.1: The baseline accuracy (random guessing based on majority class labels) of attribute inference attack on different datasets.

Dataset	#. Class	Baseline Accuracy
UTKFace	5	0.421
Places100	100	0.012
Places50	50	0.023
Places20	20	0.053

4.3.4 Results

The performance of attribute inference attacks is depicted in Figure 4.8. First, we observe that, in general, attribute inference achieves effective performance except for the supervised model trained on UTKFace dataset (close to the prior sensitive attribute distribution in the attack training dataset as shown in Table 4.1). Second, compared to the supervised models, the contrastive models are more vulnerable to attribute inference attacks. For instance, on the UTKFace dataset with ResNet-18, we can achieve an attack accuracy of 0.701 on the contrastive model while only 0.422 on the supervised model. To better understand this, we extract samples' representations (512-dimension) from ResNet-18 on UTKFace for both the supervised model and the contrastive model and project them into a 2-dimension space using t-Distributed Neighbor Embedding (t-SNE) [75]: Figure 4.9a shows the results for the supervised model on the original classification task, i.e., gender classification; Figure 4.9b shows the results for the supervised model on attribute inference, i.e., race. We see that in Figure 4.9a, male samples (blue) and female samples (orange) reside in completely different regions, which can be separated perfectly (the gender classification accuracy is 0.875 in Figure 4.1). However, for the sensitive attribute (Figure 4.9b), samples of different classes are clustered tightly, which increases the difficulty for attribute inference. Figure 4.9c and Figure 4.9d show the corresponding results for the contrastive model. We observe that different samples' representations on the contrastive model are less separable with respect to the original classification task compared to the supervised model (see Figure 4.9c and Figure 4.9a), but we can still successfully separate most of them correctly (the gender classification accuracy is 0.858 in Figure 4.1) since most of the male samples (blue) lie in the upper area while the female samples (orange) are in the lower area. On the other hand, for the sensitive attribute, compared to the supervised model (Figure 4.9b), representations generated by the contrastive model (Figure 4.9d) are more distinguishable. Our finding reveals that the representations generated by the contrastive model are more informative, which can be exploited not only for the original classification tasks but also for attribute inference.

To study the effect of training dataset size on the attack model $\mathcal{A}_{AttInf}$, we randomly select from 10% to 90% of the training dataset to train the attack model and evaluate the performance using all the testing dataset; the results for contrastive models are summarized in Figure 4.10. By jointly considering Figure 4.8 and Figure 4.10, we can observe that, in most of the cases, even using 10% of the training dataset, the contrastive models are still more vulnerable to attribute inference attack than the supervised models when the attack model is trained with its *full* training dataset. On the other hand, the attack performance on supervised models is not significantly influenced by the training dataset size (see Figure 4.18). This further shows the privacy risks of contrastive learning.

Recall our attack model is a 3-layer MLP. We further investigate whether more complex attack models would improve the attack performance. To this end, we increase the attack model's layer from 3 to 6 and summarize the corresponding attack performance for contrastive and supervised models in Figure 4.11 and Figure 4.19 (in Appendix), respectively. The results show that 3-layer attack models can achieve the best performance in most of the cases. With more layers, the attack performance may degrade or keep stable, which indicates that even simple models are enough to launch effective attacks. This further shows that informative representations learned by contrastive models can be easily exploited by the adversary to infer samples' attributes.

We also observe that attribute inference attacks over contrastive models are more effective against smaller embedding size (see Figure 4.8 and Figure 4.10). For instance, ResNet-18 (512) leak more information than MobileNetV2 (1,280) and ResNet-50 (2,048). We conjecture that a larger embedding size represents each sample in a more complex space in the contrastive setting, which is harder for the attack model to decode. However, the effect of embedding size on attribute inference attacks against the supervised models is less pronounced (see Figure 4.8 and Figure 4.18 in the Appendix). This further shows the difference between supervised models and contrastive models with respect to representing samples.

In conclusion, contrastive models are more vulnerable to attribute inference attacks compared to supervised models.

4.4 Defense

So far, we have demonstrated that compared to supervised models, contrastive models are more vulnerable to attribute inference attacks (Section 4.3) but less vulnerable to membership inference attacks (Section 4.2). In this section, we propose the first privacy-preserving contrastive learning mechanism, namely *Talos*, which aims to reduce the risks of attribute inference for contrastive models while maintaining their membership privacy and model utility.

4.4.1 Methodology

Intuition: As shown in Section 4.3, the reason for a contrastive model to be vulnerable to attribute inference attacks is that the model's base encoder f learns informative representations for data samples, which can be exploited by an adversary. To mitigate such a threat, we aim for a new training paradigm for contrastive learning which can eliminate data samples' sensitive attributes from their representations. Meanwhile, the base encoder of the contrastive model still needs to represent data samples expressively for preserving model utility. These two objectives are in conflict, and our defense mechanism should consider both simultaneously.

Methodology: Our defense mechanism, namely *Talos*, can be modeled as a mini-max game, and we rely on adversarial training [40, 29, 116, 30, 22] to realize it. Similar to the original contrastive model, *Talos* also leverages a base encoder and a projection head to learn informative representations for data samples. Besides, *Talos* introduces an adversarial classifier C, which is used to censor sensitive attributes from data samples' representations.

The adversarial classifier of *Talos* is essentially designed for attribute inference. Similar to the original contrastive learning process, *Talos* is trained with mini-batches. Given a mini-batch of $2N$ augmented data samples (generated from N original samples), we define the loss of the adversarial classifier C as follows.

$$\mathcal{L}_C = \frac{1}{2N}\sum_{k=1}^{N}[\mathcal{L}_{CE}(s_k, C(f(\tilde{x}_{2k-1}))) + \mathcal{L}_{CE}(s_k, C(f(\tilde{x}_{2k})))] \tag{4.1}$$

where $\tilde{x}_{2k-1}$ and $\tilde{x}_{2k}$ are the two augmented samples of an original sample x_k, s_k represents x_k's sensitive attribute, f is the base encoder, and $\mathcal{L}_{CE}$ is the cross-entropy loss (Equation 2.3). We consider $\tilde{x}_{2k-1}$ and $\tilde{x}_{2k}$ sharing the same sensitive attribute as x_k. Note that we take the output of the base encoder instead of the projection head as the input to the adversarial classifier. Since the projection head is discarded after the

Algorithm 2: The training process of *Talos*.

```
1  Input: Target training dataset $\mathcal{D}_{target}^{train}$ with sensitive attribute $s$, base encoder $f$,
     projection head $g$, adversarial classifier $C$, and adversarial factor $\lambda$.
2  Initialize $f$, $g$, and $C$'s parameters.
3  for each epoch do
4      for each mini-batch do
5          Sample a mini-batch with $N$ training data samples and its corresponding
             sensitive attributes $\{(x_1, s_1), (x_2, s_2), ..., (x_N, s_N)\}$ from $\mathcal{D}_{target}^{train}$
6          Generate augmented data samples: $\{(\tilde{x}_1, s_1), (\tilde{x}_2, s_1), ..., (\tilde{x}_{2N}, s_N)\}$,
             where $\tilde{x}_{2k-1}$ and $\tilde{x}_{2k}$ are the two augmented views of $x_k$
7          Feed augmented data samples into the base encoder $f$ and the projection
             head $g$ to calculate the contrastive loss:
             $\mathcal{L}_{Contrastive} = \frac{1}{2N}\sum_{k=1}^{N}[\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
8          Feed the representations generated by the base encoder $f$ into the
             adversarial classifier $C$ to calculate the adversarial classifier loss:
             $\mathcal{L}_C = \frac{1}{2N}\sum_{k=1}^{N}[\mathcal{L}_{CE}(s_k, C(f(\tilde{x}_{2k-1}))) + \mathcal{L}_{CE}(s_k, C(f(\tilde{x}_{2k})))]$
9          if epoch mod 2 $\neq 0$ then
10             Optimize adversarial classifier $C$'s parameters with the adversarial
                 classifier loss: $\mathcal{L}_C$
11         else
12             Optimize projection head $g$'s parameters with the contrastive loss:
                 $\mathcal{L}_{Contrastive}$
13             Optimize base encoder $f$'s parameters with adversarial training loss:
                 $\mathcal{L}_{Talos} = \mathcal{L}_{Contrastive} - \lambda\mathcal{L}_C$
14         end
15     end
16 end
17 Return: Base encoder $f$
```

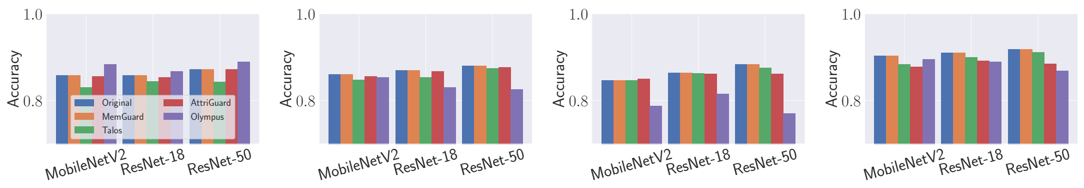
first phase of training the contrastive model, directly optimizing the base encoder with the adversarial classifier loss would maintain the effect of adversarial training.

Talos also adopts the original contrastive loss $\mathcal{L}_{Contrastive}$ (Equation 2.6). By jointly considering the adversarial classifier loss and the contrastive loss, *Talos*'s loss function is defined as follows:

$$\mathcal{L}_{Talos} = \mathcal{L}_{Contrastive} - \lambda\mathcal{L}_C \tag{4.2}$$

where λ is the *adversarial factor* to balance the two losses. We refer to a model trained with *Talos* as a *Talos* model.

algorithm 2 presents the training process of *Talos*. In each mini-batch, given N training samples, we first generate $2N$ augmented views (Line 6) and feed them into the base encoder. The generated representations are then fed into the projection head (Line 7) and the adversarial classifier (Line 8) simultaneously. Note that the adversarial classifier and contrastive model are updated alternately by epoch. We First optimize the adversarial classifier with the cross-entropy loss (Line 10). Then we optimize the



(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.12: The performance of original classification tasks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of original classification tasks.

projection head with the contrastive loss (Line 12) and the base encoder with the loss function of *Talos*, i.e., Equation 4.2 (Line 13).

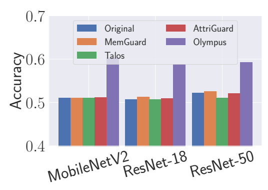
To implement this in practice, we utilize the gradient reversal layer (GRL) proposed by Ganin et al. [35]. GRL is a layer that can be added between the base encoder f and the adversarial classifier C. In the forward propagation, GRL acts as an identity transform that simply copies the input as the output. During the backpropagation, GRL takes the gradients passed through it from the adversarial classifier C, multiplies the gradients by $-\lambda$, and passes them to the base encoder f. Such operation lets the base encoder receive the opposite direction of gradients from the adversarial classifier. In this way, the base encoder f is able to learn informative representations for samples while censoring their sensitive attributes.

Note that our adversarial training is performed only on the process of training the base encoder f. The training for the classification layer of the contrastive model remains unchanged. As we show in Section 4.2, the classification layer generalizes well on the contrastive models, i.e., less overfitting. Therefore, models trained by *Talos* should be robust against membership inference attacks as well. Our evaluation shows that this is indeed the case (see Figure 4.13).

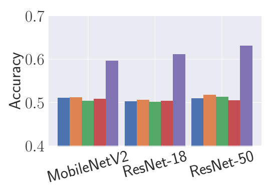
Adaptive Attacks: An adversary needs to establish a shadow model to mount membership inference attacks. To evaluate membership privacy risks of *Talos*, we consider an adaptive (and stronger) adversary [58]. Concretely, we assume that the adversary knows the training details of *Talos* and trains their shadow model in the same way. For attribute inference, the attack model is trained on embeddings generated by *Talos*, thus, our attribute inference attack considered in the evaluation of *Talos* is also adaptive.

4.4.2 Experimental Setting

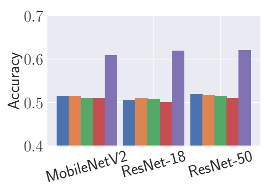
We follow the same experimental setting, including datasets, metrics, target models, and attack models (both attribute inference and membership inference), as those in Section 4.2.3 and Section 4.3.3. As mentioned before, both membership inference and attribute inference attacks are performed in an adaptive way. Regarding the adversarial classifier of *Talos*, we leverage a 3-layer MLP with 64 neurons in the hidden layer, which is smaller than the attribute inference attack model.



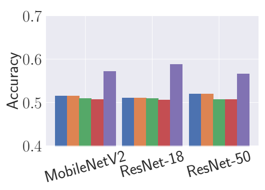
(a) UTKFace



(b) Places100

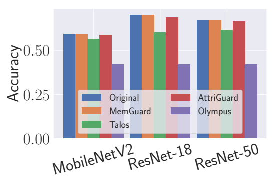


(c) Places50

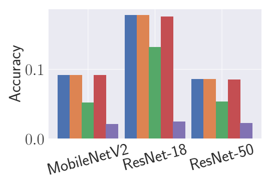


(d) Places20

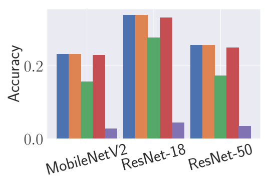
Figure 4.13: The performance of NN-based membership inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of NN-based membership inference attacks.



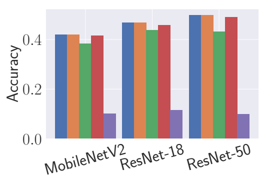
(a) UTKFace



(b) Places100



(c) Places50



(d) Places20

Figure 4.14: The performance of attribute inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of attribute inference attacks.

Baseline: We consider three state-of-the-art defenses, one for membership inference (*MemGuard* [58]) and two for attribute inference (*Olympus* [90] and *AttriGuard* [57]) as the baseline models. *MemGuard*, *Olympus*, and *AttriGuard* are originally designed for supervised models, here, we adapt them to contrastive models. Since the input to the attribute inference attack is each sample's representation, we further consider a sample's representation as the input to *Olympus* and *AttriGuard*.

MemGuard is a two-phase defense for membership inference. In phase I, the defender generates a noise vector to perturb the posteriors of a target sample, so that the adversary's membership classifier is likely to give a random guess for the perturbed posteriors. In phase II, the defender adds the noise vector to the posteriors with certain probability.

Olympus, designed for attribute inference, has three basic components: an autoencoder to transfer the original representation into the perturbed one, a classifier to perform the original task over the perturbed representation, and an adversarial classifier to infer the sensitive attribute from the perturbed representation. *Olympus* optimizes the three components using adversarial training to preserve the model utility while protecting samples' sensitive attributes. To perform *Olympus* on contrastive models, we first train a base encoder following the original contrastive learning process. Then, we add an autoencoder between the base encoder and the classification layer, and fine-tune the whole model using the original training samples with *Olympus*'s losses.

AttriGuard is a two-phase defense for attribute inference. In phase I, for each representation, the defender generates an adversarial example for each possible value of the sensitive attribute by adapting the existing evasion attack techniques. In phase II, the defender samples one sensitive attribute value based on a probability distribution and selects the corresponding adversarial example found in phase I as the new representation.

The adversarial classifier used in *AttriGuard* and *Olympus* shares the same architecutre as the one in *Talos*. For *MemGuard*, we follow Jia et al. [58] to generate the noise in Phase I. For the autoencoder of *Olympus*, we set its encoder (decoder) as a 2-layer MLP with 256 and 128 (128 and 256) neurons in the hidden layers. For *AttriGuard*, we leverage the C&W attack [14] with the L_{inf} norm in phase I.

4.4.3 Results

We compare the performance of the original classification tasks, NN-based membership inference attacks, and attribute inference attacks for the original contrastive model and the models defended by *Talos*, *MemGuard*, *Olympus*, and *AttriGuard*. The results are depicted in Figure 4.12, Figure 4.13, and Figure 4.14, respectively. Note that we also perform metric-based and label-only membership inference attacks and the results are summarized in Figure 4.20, Figure 4.21, Figure 4.22, Figure 4.23, and Figure 4.24 in Appendix.

In Figure 4.14, we find that *Talos* indeed reduces the attribute inference accuracy compared to the original contrastive learning. For instance, the attribute inference accuracy is 0.701 on the original contrastive model with ResNet-18 on the UTKFace dataset, while only 0.602 on the *Talos* model. Meanwhile, the testing accuracy of the original classification task for the *Talos* model is also preserved (Figure 4.12).

For different defense mechanisms, we find that *Olympus* reduces attribute inference attacks the most (see Figure 4.14). However, it jeopardizes the membership privacy to a large extent (see Figure 4.13). For instance, the membership inference accuracy of the *Talos* model (ResNet-50) on Place100 is 0.513 while the corresponding *Olympus* model's value is 0.631. The reason is that *Olympus*'s training process utilizes the original training samples to fine-tune the whole model, which leads to the model memorizing these samples with the model's full capacity. On the other hand, as mentioned in Section 4.4.1, *Talos* is only performed on the training process of the base encoder f which considers each sample's augmented views. The original samples are only used to fine-tune the final classification layer, the same as training a normal contrastive model. In other words, the *Talos* model memorizes its training samples with only its one-layer capacity. Therefore, *Talos* models are less prone to membership inference. In addition, *Olympus* jeopardizes the target model's utility in multiple cases (see Figure 4.12b, Figure 4.12c, and Figure 4.12d), the reason again lies in the training process of *Olympus*. More specifically, *Olympus* needs to fine-tune the whole model in a supervised way, this reduces the effect of contrastive learning in the final model. Meanwhile, *Talos* preserves the contrastive learning process to a large extent as its adversarial loss is applied together with the contrastive loss during the training of the base encoder. Since membership privacy, attribute privacy, and model utility are equally important, we believe *Talos* is a better choice than *Olympus*.

We also find that *Talos*, *MemGuard*, and *AttriGuard* models can achieve similar utility as the original contrastive models (see Figure 4.12). However, *Talos* can mitigate attribute inference attacks to a larger extent than *AttriGuard* and *MemGuard* (see Figure 4.14). For instance, the attribute inference accuracy is only 0.132 on the *Talos* model with ResNet-18 on the Places100 dataset, while 0.176 and 0.178 on the *AttriGuard* and *MemGuard* models. Also, as the contrastive learning procedure is preserved for *Talos*, *AttriGuard*, and *MemGuard*, we observe that all these defenses are robust against membership inference attacks (see Figure 4.13).

We also investigate the effect of the adversarial factor λ on the performance of original classification tasks, membership inference attacks, and attribute inference attacks. The results are summarized in Figure 4.25, Figure 4.26, and Figure 4.27. First of all, we observe that the performance of original classification tasks (Figure 4.25) and membership inference attacks (Figure 4.26) are relatively stable with respect to different adversarial factors. However, for different datasets or different model architectures, the best λ to defense attributes inference attack may vary (Figure 4.27). In general, we notice that setting λ to 2 or 3 can achieve nearly the best defense performance on most datasets and model architectures. To perform *Talos* in practice, we believe the model owner needs to tune the λ on their validation dataset. During the process, concentrating more on model utility or defense effectiveness depends on the ML model owner's purpose.

In conclusion, *Talos* can successfully defend attribute inference attacks for contrastive models without jeopardizing their membership privacy and model utility.

4.5 Discussion

Other Types of Datasets: In this work, we only focus on image datasets, as most of the current efforts on contrastive learning concentrate on the image domain. For other types of datasets like texts or graphs, the main challenge is to define a suitable augmentation method for the input sample. There indeed exist some preliminary works of contrastive learning over texts or graphs [37, 119]. However, the effectiveness of these methods still needs to be further evaluated. We believe it is straightforward to extend our analysis to contrastive models trained on other types of data.

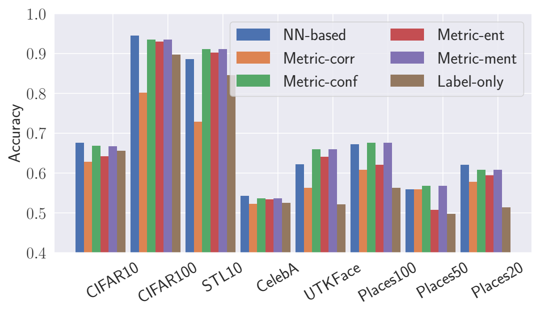
Novel Membership Inference Attacks Against Contrastive Models: Traditional membership inference attacks use the original data samples to query the model and get the corresponding posteriors to launch the attacks. However, such attacks is less effective on contrastive models as shown in the siddertation. Since the contrastive model is trained with some augmented views of each data sample, the model itself may remember these augmented views as well. This inspires us to use the augmented views of the original training sample to query the contrastive model to obtain multiple posteriors (one for each augmented version), and aggregate these posteriors as the input to the membership inference attack model. However, our initial attempt in this direction does not achieve a stronger attack. One reason might be our aggregation method is not optimal (we have tried averaging and concatenation). In the future, we plan to investigate more advanced aggregation operations to establish a membership inference attack tailored to contrastive models.

4.6 Conclusion

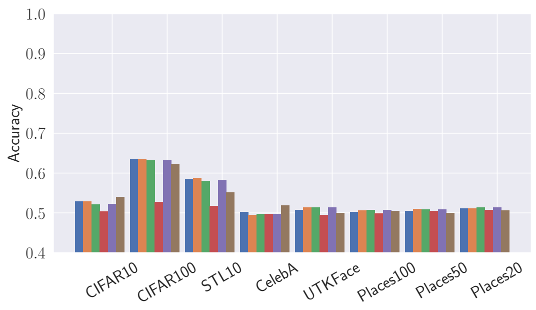
In this work, we perform the first privacy quantification of the most representative self-supervised learning paradigm, i.e., contrastive learning. Concretely, we investigate the privacy risks of contrastive models trained on image datasets through the lens of membership inference and attribute inference. Empirical evaluation shows that contrastive models are less vulnerable to membership inference attacks compared to supervised models. This is due to the fact that contrastive models are normally less overfitted. Meanwhile, contrastive models are more prone to attribute inference attacks. We posit this is because contrastive models can generate more informative representations for data samples, which can be exploited by an adversary to achieve effective attribute inference.

To reduce the risks of attribute inference stemming from contrastive models, we propose the first privacy-preserving contrastive learning mechanism, namely *Talos*. Specifically, *Talos* introduces an adversarial classifier to censor the sensitive attributes learned by the contrastive models under the adversarial training framework. Our evaluation shows that *Talos* can effectively mitigate the attribute inference risks for contrastive models while maintaining their membership privacy and model utility.

4.7 Appendix

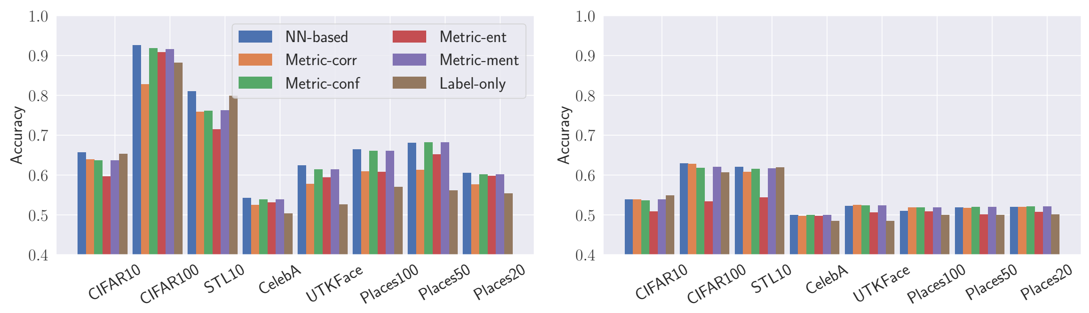


(a) Supervised Model



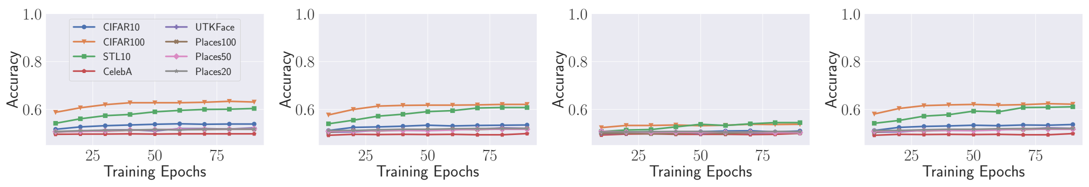
(b) Contrastive Model

Figure 4.15: The performance of different membership inference attacks against both supervised models and contrastive models with ResNet-18 on 8 different datasets. The x-axis represents different datasets. The y-axis represents membership inference attacks' accuracy.



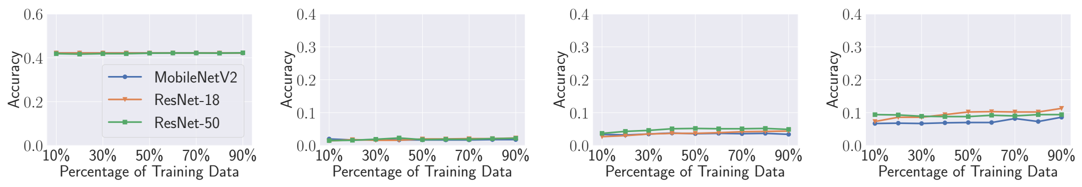
(a) Supervised Model **(b)** Contrastive Model

Figure 4.16: The performance of different membership inference attacks against both supervised models and contrastive models with ResNet-50 on 8 different datasets. The x-axis represents different datasets. The y-axis represents membership inference attacks' accuracy.



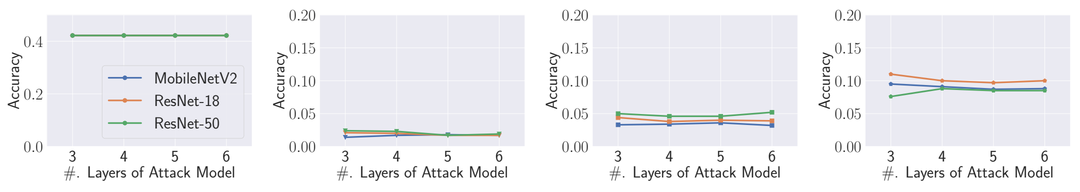
(a) Metric-corr **(b)** Metric-conf **(c)** Metric-ent **(d)** Metric-ment

Figure 4.17: The performance of metric-based membership inference attacks against contrastive models with ResNet-50 on 8 different datasets under different numbers of epochs for classification layer training. The x-axis represents different numbers of epochs. The y-axis represents membership inference attacks' accuracy. Each line corresponds to a specific dataset.



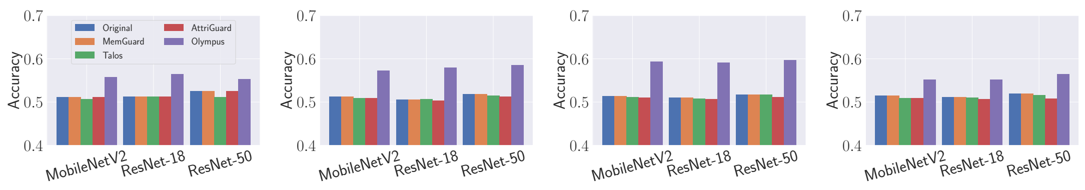
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.18: The performance of attribute inference attacks against supervised models on 4 different datasets under different percentages of the attack training dataset. The x-axis represents different percentages of the attack training dataset. The y-axis represents attribute inference attacks' accuracy.



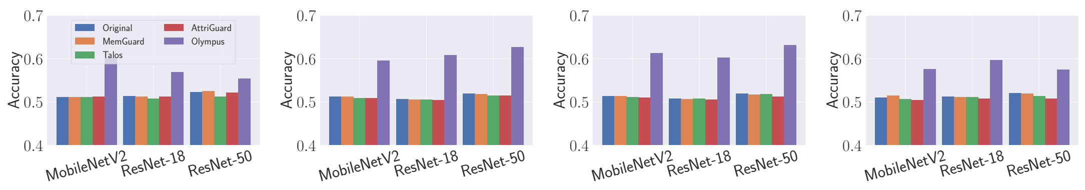
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.19: The performance of attribute inference attacks against supervised models on 4 different datasets under attack models with different layers. The x-axis represents attack models' layers. The y-axis represents attribute inference attacks' accuracy.



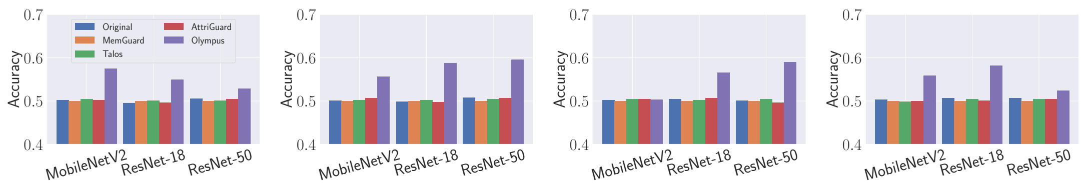
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.20: The performance of metric-corr membership inference attacks against original contrastive models,*Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of metric-corr membership inference attacks.



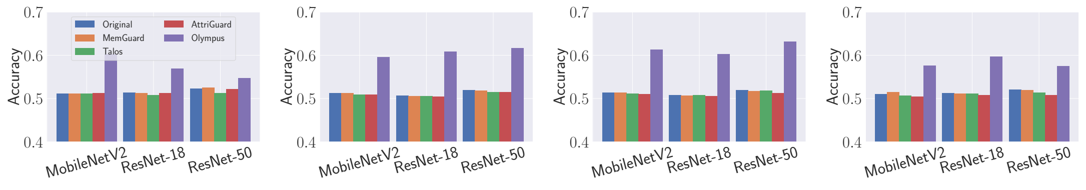
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.21: The performance of metric-conf membership inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different methods. The y-axis represents the accuracy of metric-conf membership inference attacks.



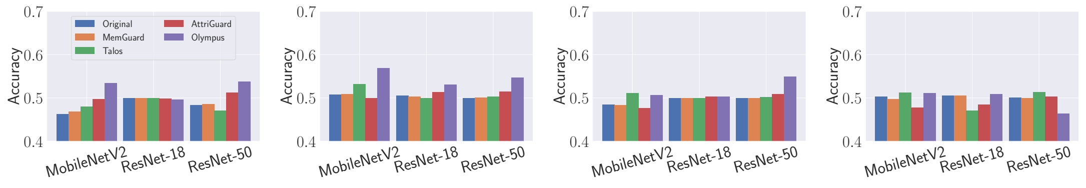
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.22: The performance of metric-ent membership inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of metric-ent membership inference attacks.



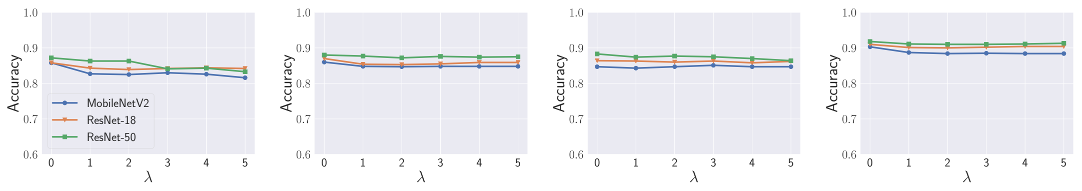
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.23: The performance of metric-ment membership inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of metric-ment membership inference attacks.



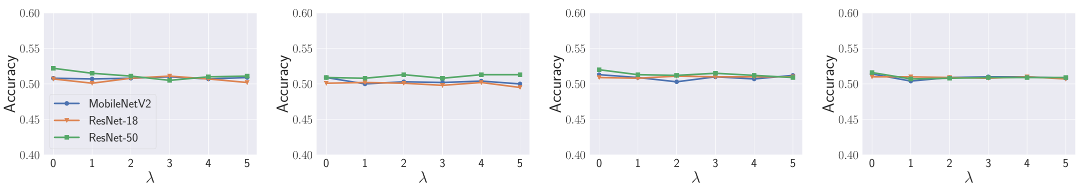
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.24: The performance of label-only membership inference attacks against original contrastive models, *Talos*, *MemGuard*, *Olympus*, and *AttriGuard* with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets. The x-axis represents different models. The y-axis represents the accuracy of label-only membership inference attacks.



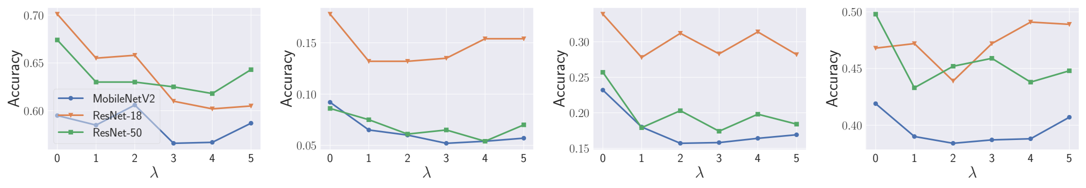
(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.25: The performance of original classification tasks for the *Talos* models with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets under different adversarial factor λ. The x-axis represents different λ. The y-axis represents the corresponding performance.



(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.26: The performance of membership inference attacks for the *Talos* models with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets under different adversarial factor λ. The x-axis represents different λ. The y-axis represents the corresponding performance.



(a) UTKFace **(b)** Places100 **(c)** Places50 **(d)** Places20

Figure 4.27: The performance of attribute inference attacks for the *Talos* models with MobileNetV2, ResNet-18, and ResNet-50 on 4 different datasets under different adversarial factor λ. The x-axis represents different λ. The y-axis represents the corresponding performance.

5 Graph Neural Networks

5.1 Introduction

Graph is a powerful tool to model the complex relationships between entities. For instance, in healthcare analytics, protein-protein interactions can be modeled as a graph (called a *chemical network*); and a social network can be modeled as a graph, where nodes are users and edges indicate certain social relationships among them. A graph may be treated as a data owner's intellectual property because the data owner may spend a lot of resources collecting the graph, e.g., collecting a chemical network often involves expensive and resource-consuming chemical experiments. Moreover, a graph may also contain sensitive user information, e.g., private social relationships among users.

Recently, a family of machine learning techniques known as *graph neural networks (GNNs)* was proposed to analyze graphs. We consider GNNs for *node classification*. Specifically, given a graph, attributes of each node in the graph, and a small number of node labels, a GNN model is trained and can predict the label of each remaining unlabeled node. Due to their superior performance, we have seen growing applications of GNNs in various domains, such as healthcare analytics [36, 31], recommender systems [32], and fraud detection [111]. However, the security and privacy implications of training GNNs on graphs are largely unexplored.

Our Contributions: In this work, we take the first step to study the security and privacy implications of training GNNs on graphs. In particular, we propose the first attacks to steal a graph from the outputs of a GNN model trained on the graph. We call our attacks *link stealing attacks*. Specifically, given a black-box access to a target GNN model, our attacks aim to predict whether there exists a link between any pair of nodes in the graph used to train the target GNN model. Our attacks reveal serious concerns on the intellectual property, confidentiality, and/or privacy of graphs when training GNNs on them. For instance, our attacks violate the intellectual property of the data owner when it spends lots of resources collecting the graph; and our attacks violate user privacy when the graph contains sensitive social relationships among users [38, 6].
Adversary's Background Knowledge: We refer to the graph and nodes' attributes used to train the target GNN model as the *target dataset*. We characterize an adversary's background knowledge along three dimensions, including the target dataset's *nodes' attributes*, the target dataset's *partial graph*, and an auxiliary dataset (called *shadow dataset*) which also contains its own graph and nodes' attributes. An adversary may or may not have access to each of the three dimensions. Therefore, we obtain a comprehensive taxonomy of a threat model, in which adversaries can have 8 different types of background knowledge.
Attack Methodology: We design an attack for each of the 8 different types of background knowledge, i.e., we propose 8 link stealing attacks in total. The key intuition of our attacks is that two nodes are more likely to be linked if they share more similar attributes and/or predictions from the target GNN model. For instance, when the adversary only has the target dataset's nodes' attributes, we design an unsupervised attack by calculating the distance between two nodes' attributes. When the target dataset's partial graph is available, we use supervised learning to train a binary classifier as our attack model with features summarized from two nodes' attributes and predictions

obtained from the black-box access to the target GNN model. When the adversary has a shadow dataset, we propose a *transferring attack* which transfers the knowledge from the shadow dataset to the target dataset to mount our attack.
Evaluation: We evaluate our 8 attacks using 8 real-world datasets. First, extensive experiments show that our attacks can effectively steal links. In particular, our attacks achieve high AUCs (area under the ROC curve). This demonstrates that the predictions of a target GNN model encode rich information about the structure of a graph that is used to train the model, and our attacks can exploit them to steal the graph structure. Second, we observe that more background knowledge leads to better attack performance in general. For instance, on the Citeseer dataset [61], when an adversary has all the three dimensions of the background knowledge, our attack achieves 0.977 AUC. On the same dataset, when the adversary only has nodes' attributes, the AUC is 0.878. Third, we find that the three dimensions of background knowledge have different impacts on our attacks. Specifically, the target dataset's partial graph has the strongest impact followed by nodes' attributes, the shadow dataset, on the other hand, has the weakest impact. Fourth, our transferring attack can achieve high AUCs. Specifically, our transferring attack achieves better performance if the shadow dataset comes from the same domain as the target dataset, e.g., both of them are chemical networks. We believe this is due to the fact that graphs from the same domain have similar structures, which leads to less information loss during transferring. Fifth, our attacks outperform conventional link prediction methods [70, 39], which aim to predict links between nodes based on a partial graph.

In summary, we make the following contributions.

- We propose the first link stealing attacks against graph neural networks.
- We propose a threat model to comprehensively characterize an adversary's background knowledge along three dimensions. Moreover, we propose 8 link stealing attacks for adversaries with different background knowledge.
- We extensively evaluate our 8 attacks on 8 real-world datasets. Our results show that our attacks can steal links from a GNN model effectively.

5.2 Problem Formulation

In this section, we first propose a threat model to characterize an adversary's background knowledge. Then, we formally define our link stealing attack.

5.2.1 Threat Model

Adversary's Goal: An adversary's goal is to infer whether a given pair of nodes u and v are connected in the target dataset. Inferring links between nodes leads to a severe privacy threat when the links represent sensitive relationship between users in the context of social networks. Moreover, links may be confidential and viewed as a model owner's intellectual property because the model owner may spend lots of resources collecting the links, e.g., it requires expensive medical/chemical experiments to determine the interaction/link between two molecules in a chemical network. Therefore, inferring links may also compromise a model owner's intellectual property.

Adversary's Background Knowledge: First, we assume an adversary has a black-box access to the target GNN model. In other words, the adversary can only obtain nodes' posteriors by querying the target model f. This is the most difficult setting for the adversary [97, 94, 93]. An adversary can have a black-box access to a GNN model when an organization uses GNN tools from another organization (viewed as an adversary) or the GNN model prediction results are shared among different departments within the same organization. For instance, suppose a social network service provider leverages another company's tool to train a GNN model for fake-account detection, the provider often needs to send the prediction results of (some) nodes to the company for debugging or refining purposes. In such a scenario, the security company essentially has a black-box access to the GNN model. Note that the graph structure is already revealed to the adversary if she has a white-box access to the target GNN model as the GNN model architecture is often based on the graph structure.

Then, we characterize an adversary's background knowledge along three dimensions:

- **Target Dataset's Nodes' Attributes, denoted by $\mathcal{F}$.** This background knowledge characterizes whether the adversary knows nodes' attributes $\mathcal{F}$ in $\mathcal{D}$. We also assume that the adversary knows labels of a small subset of nodes.
- **Target Dataset's Partial Graph, denoted by $\mathcal{A}^*$.** This dimension characterizes whether the adversary knows a subset of links in the target dataset $\mathcal{D}$. Since the goal of link stealing attack is to infer whether there exists an edge/link between a pair of nodes, the partial graph can be used as ground truth edges to train the adversary's attack model.
- **A Shadow Dataset, denoted by $\mathcal{D}'$.** This is a dataset which contains its own nodes' attributes and graph. The adversary can use this to build a GNN model, referred to as *shadow target model* (denoted by f') in order to perform a transferring attack. It is worth noting that the shadow dataset does not need to come from the same domain of the target dataset. For instance, the shadow dataset can be a chemical network, while the target dataset can be a citation network. However, results in Section 5.4 show that same-domain shadow dataset indeed leads to better transferring attack performance.

We denote the adversary's background knowledge as a triplet:

$$\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \mathcal{D}').$$

Whether the adversary has each of the three items is a binary choice, i.e., yes or no. Therefore, we have a comprehensive taxonomy with 8 different types of background knowledge, which leads to 8 different link stealing attacks. Table 5.1 summarizes our attack taxonomy.

Table 5.1: Attack taxonomy. ✓ (×) means the adversary has (does not have) the knowledge.

Attack	$\mathcal{F}$	$\mathcal{A}^*$	$\mathcal{D}'$	Attack	$\mathcal{F}$	$\mathcal{A}^*$	$\mathcal{D}'$
Attack-0	×	×	×	Attack-4	×	✓	✓
Attack-1	×	×	✓	Attack-5	✓	×	✓
Attack-2	✓	×	×	Attack-6	✓	✓	×
Attack-3	×	✓	×	Attack-7	✓	✓	✓

Table 5.2: Features adopted by our supervised attacks (Attack-3 and Attack 6) and transferring attacks (Attack-1, Attack-4, Attack-5, and Attack-7). Here, $(*)$ means the features are extracted from the shadow dataset in the training phase, and $(\star)$ means the features are extracted from both the shadow dataset and the target dataset (its partial graph) in the training phase. $d(\cdot,\cdot)$ represents distance metrics defined in Table 5.12, $\Psi(\cdot,\cdot)$ represents the pairwise vector operations defined in Table 5.13. Note that the features used in these attack models include all the distance metrics and pairwise vector operations.

Attack	$d(f(u),f(v))$	$\Psi(f(u),f(v)))$	$\Psi(e(f(u)),e(f(v)))$	$d(g(u),g(v))$	$\Psi(g(u),g(v))$	$\Psi(e(g(u)),e(g(v)))$	$d(\mathcal{F}_u,\mathcal{F}_v)$	$\Psi(\mathcal{F}_u,\mathcal{F}_v)$
Attack-1 $*$	✓	×	✓	×	×	×	×	×
Attack-3	✓	✓	✓	×	×	×	×	×
Attack-4 $\star$	✓	×	✓	×	×	×	×	×
Attack-5 $*$	✓	×	✓	✓	×	✓	✓	×
Attack-6	✓	✓	✓	✓	✓	✓	✓	✓
Attack-7 $\star$	✓	×	✓	✓	×	✓	✓	×

5.2.2 Link Stealing Attack

After describing our threat model, we can formally define our link stealing attack as follows:

Definition 1 (Link Stealing Attack)**.** *Given a black-box access to a GNN model that is trained on a target dataset, a pair of nodes u and v in the target dataset, and an adversary's background knowledge $\mathcal{K}$, link stealing attack aims to infer whether there is a link between u and v in the target dataset.*

5.3 Attack Taxonomy

In this section, we present the detailed constructions of all the 8 attacks in Table 5.1. Given different knowledge $\mathcal{K}$, the adversary can conduct their attacks in different ways. However, there are two problems that exist across different attacks.

The first problem is *node pair order.* As we consider undirected graph, when the adversary wants to predict whether there is a link between two given nodes u and v, the output should be the same regardless of the input node pair order.

The second problem is *dimension mismatch.* The shadow dataset and the target dataset normally have different dimensions with respect to attributes and posteriors (as they are collected for different classification tasks). For transferring attacks that require the adversary to transfer information from the shadow dataset to the target dataset, it is crucial to keep the attack model's input features' dimension consistent no matter which shadow dataset she has.

We will discuss how to solve these two problems during the description of different attacks. For presentation purposes, features used in our supervised attacks and transferring attacks are summarised in Table 5.2.

5.3.1 Attack Methodologies

Attack-0: $\mathcal{K} = (\times, \times, \times)$**:** We start with the most difficult setting for the adversary, that is she has no knowledge of the target dataset's nodes' attributes, partial graph, and a shadow dataset. All she has is the posteriors of nodes obtained from the target model f (see Section 2.4).

As introduced in Section 2.4, GNN essentially aggregates information for each node from its neighbors. This means if there is a link between two nodes, then their posteriors obtained from the target model should be closer. Following this intuition, we propose an unsupervised attack. More specifically, to predict whether there is a link between u and v, we calculate the distance between their posteriors, i.e., $d(f(u), f(v))$, as the predictor.

We have in total experimented with 8 common distance metrics: Cosine distance, Euclidean distance, Correlation distance, Chebyshev distance, Braycurtis distance, Canberra distance, Manhattan distance, and Square-euclidean distance. Their formal definitions are in Table 5.12 in Appendix. It is worth noting that all distance metrics we adopt are symmetric, i.e., $d(f(u), f(v)) = d(f(v), f(u))$, this naturally solves the problem of *node pair order*.

Since the attack is unsupervised, to make a concrete prediction, the adversary needs to manually select a threshold depending on application scenarios. To evaluate our attack, we mainly use AUC which considers a set of thresholds as previous works [34, 6, 44, 94, 58, 124]. In addition, we propose a threshold estimation method based on clustering (see Section 5.4 for more details).

Attack-1: $\mathcal{K} = (\times, \times, \mathcal{D}')$**:** In this attack, we broaden the adversary's knowledge with a shadow dataset, i.e., $\mathcal{D}'$. This means the adversary can train a classifier for a supervised attack, more specifically, a *transferring attack*. She first constructs a shadow target model f' with $\mathcal{D}'$. Then, she derives the training data from f' to train her attack model.

The adversary cannot directly use the posteriors obtained from the shadow target model as features to train her attack model, as the shadow dataset and the target dataset very likely have different numbers of labels, i.e., the corresponding posteriors are in different dimensions. This is the dimension mismatch problem mentioned before. To tackle this, we need to design features over posteriors.

As discussed in Attack-0, for any dataset, if two nodes are linked, then their posteriors obtained from the target model should be similar. This means if the attack model can capture the similarity of two nodes' posteriors from the shadow target model, it can transfer the information to the target model.

We take two approaches together to design features. The first approach is measuring distances between two nodes' posteriors. To this end, for each pair of nodes u' and v' from the shadow dataset $\mathcal{D}'$, we adopt the same set of 8 metrics used in Attack-0 (formal definitions are listed in Table 5.12) to measure their posteriors $f'(u')$ and

$f'(v')$'s distances, and concatenate these different distances together. This leads to an 8-dimension vector.

The second approach is to use entropy to describe each posterior inspired by previous works [78, 58]. Formally, for the posterior of node u' obtained from the shadow target model f', its entropy is defined as the following.

$$e(f'(u')) = -\sum_{i} f'_i(u') log(f'_i(u')) \tag{5.1}$$

where $f'_i(u')$ denotes the i-th element of $f'(u')$. Then, for each pair of nodes u' and v' from the shadow dataset, we obtain two entropies $e(f'(u'))$ and $e(f'(v'))$. To eliminate the node pair order problems for these entropies, we further take the approach of Grover and Leskovec [42], by applying pairwise vector operation, denoted by $\Psi(\cdot, \cdot)$. In total, we have used all the 4 operations defined in Table 5.13 (in Appendix) for our attack. Note that these operations in Table 5.13 are applied on two single numbers, i.e., scalars, in this attack. However, they can also be applied to vectors and we will adopt them again on posteriors and nodes' attributes in other attacks.

In total, the features used for training the attack model is assembled with 8 different distances between two nodes' posteriors from the shadow target model and 4 features obtained from pairwise vector operations between two nodes' posteriors' entropies. Regarding labels for the training set, the adversary uses all the links in $\mathcal{D}'$ and samples the same number of node pairs that are not linked (see Section 5.4 for more details). We adopt an MLP as our attack model.

Attack-2: $\mathcal{K} = (\mathcal{F}, \times, \times)$**:** In this attack, we assume that the adversary has the knowledge of the target dataset's nodes' attributes $\mathcal{F}$. Since the adversary has no knowledge of the partial graph and a shadow dataset, her attack here is also unsupervised (similar to Attack-0). We again rely on the distance metrics to perform our attack. For each pair of nodes u and v from the target dataset, we consider four types of information to measure distance with all the metrics listed in Table 5.12. Similar to Attack-0, we experimentally decide which is the most suitable distance metric for Attack-2.

- $d(f(u), f(v))$. The first type is the same as the method for Attack-0, i.e., distance between posteriors of u and v from the target model f, i.e., $f(u)$ and $f(v)$.
- $d(\mathcal{F}_u, \mathcal{F}_v)$. The second type is calculating the pairwise distance over u and v's attributes $\mathcal{F}_u$ and $\mathcal{F}_v$.
- $d(f(u), f(v)) - d(g(u), g(v))$. For the third type, since we have the target model's nodes' attributes (as well as a subset of their corresponding labels), we train a separate MLP model, namely *reference model* (denoted by g). Our intuition is that if two nodes are connected, the distance between their posteriors from the target model should be smaller than the corresponding distance from the reference model. Therefore, we calculate $d(f(u), f(v)) - d(g(u), g(v))$ to make prediction.
- $d(g(u), g(v))$. For the fourth type, we measure the distance over u and v's posteriors from the reference model.

Attack-3: $\mathcal{K} = (\times, \mathcal{A}^*, \times)$: In this scenario, the adversary has access to the partial graph $\mathcal{A}^*$ of the target dataset. For the attack model, we rely on links from the known partial graph as the ground truth label to train an attack model (we again adopt an MLP). Features used for Attack-3 are summarized in Table 5.2. For each pair of nodes u and v from the target dataset, we calculate the same set of features proposed for Attack-1 on their posteriors and posteriors' entropies. Besides, since we can directly train the attack model on the partial target graph (i.e., we do not face the dimension mismatch problem), we further define new features by adopting the pairwise vector operations listed in Table 5.13 to $f(u)$ and $f(v)$.

Attack-4: $\mathcal{K} = (\times, \mathcal{A}^*, \mathcal{D}')$: In this attack, the adversary has the knowledge of the partial graph $\mathcal{A}^*$ of the target dataset and a shadow dataset $\mathcal{D}'$. To take both knowledge into consideration, for each pair of nodes either from the shadow dataset or the partial graph of the target dataset, we calculate the same set of features over posteriors as proposed in Attack-1. This means the only difference between Attack-4 and Attack-1 is that the training set for Attack-4 also includes information from the target dataset's partial graph (see Table 5.2).

Different from Attack-3, Attack-4 cannot perform the pairwise vector operations to $f(u)$ and $f(v)$. This is due to the dimension mismatch problem as the adversary needs to take both $\mathcal{A}^*$ and $\mathcal{D}'$ into account for her attack.

Attack-5: $\mathcal{K} = (\mathcal{F}, \times, \mathcal{D}')$: In this attack, the adversary has the knowledge of the target model's nodes' attributes $\mathcal{F}$ and a shadow dataset $\mathcal{D}'$. As we do not have $\mathcal{A}^*$ to train the attack model, we need to rely on the graph of the shadow dataset. To this end, we first calculate the same set of features used for Attack-1. Moreover, as we have the target dataset's nodes' attributes, we further build a reference model (as in Attack-2), and also a shadow reference model in order to transfer more knowledge from the shadow dataset for the attack. For this, we build the same set of features as in Attack-1 over the posteriors obtained from the shadow reference model, i.e., the distance of posteriors (Table 5.12) and pairwise vector operations performed on posteriors' entropies (Table 5.13). In addition, we also calculate the 8 different distances over the shadow dataset's nodes' attributes.

Attack-6: $\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \times)$: In this scenario, the adversary has the access to the target dataset's nodes' attributes $\mathcal{F}$ and the partial target graph $\mathcal{A}^*$. As a supervised learning setting, we build an MLP considering links from the partial graph as the ground truth label. The adversary first adopts the same set of features defined over posteriors obtained from the target model as proposed in Attack-3. Then, the adversary builds a reference model over the target dataset's nodes' attributes, and calculate the same set of features over posteriors obtained from the reference model. In the end, we further calculate the distances of the target dataset's nodes' attributes as another set of features.

Attack-7: $\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \mathcal{D}')$: This is the last attack with the adversary having all three knowledge. The set of features for this attack is the same as the ones used in Attack-5 (Table 5.2). The only difference lies in the training phase, we can use the partial graph from the target dataset together with the graph from the shadow dataset as the ground truth. We expect this leads to better performance than the one for Attack-5. However, this attack also relies on the information of the shadow dataset, thus, the features used here are a subset of the ones for Attack-6, this is similar to the difference between Attack-4 and Attack-3. Note that if the adversary does not take the shadow dataset into consideration, this scenario is equivalent to the one for Attack-6.

5.3.2 Summary

We propose 8 attack scenarios with the combination of the knowledge that the adversary could have. They could be divided into three categories.

The first category is unsupervised attacks, i.e., Attack-0 and Attack-2, where the adversary does not have the knowledge about the partial graph from the target dataset or a shadow dataset. In these scenarios, the adversary can use distance metrics for posteriors or nodes' attributes to infer the link.

The second category is the supervised attacks, including Attack-3 and Attack-6, where the adversary has the knowledge of the partial graph from the target dataset but does not have a shadow dataset. In these scenarios, the adversary can use different distances and pairwise vector operations over nodes' posteriors (and the corresponding entropies) from the target model and their attributes to build features.

The third category is the transferring attacks (supervised), including Attack-1, Attack-4, Attack-5, and Attack-7, where the adversary has the knowledge of a shadow dataset. In these scenarios, the adversary can use distance metrics over posteriors/nodes' attributes and pairwise operations over posteriors' entropies as the bridge to transfer the knowledge from the shadow dataset to perform link stealing attacks. It is worth noting that for Attack-4 and Attack-7, if the adversary leaves the shadow dataset out of consideration, they will not have the dimension mismatch problem and can take the same attack methods as Attack-3 and Attack-6, respectively.

5.4 Evaluation

This section presents the evaluation results of our 8 attacks. We first introduce our experimental setup. Then, we present detailed results for different attacks. Finally, we summarize our experimental findings.

5.4.1 Experimental Setup

Datasets: We utilize 8 public datasets, including Citeseer [61], Cora [61], Pubmed [61], AIDS [91], COX2 [105], DHFR [105], ENZYMES [26], and PROTEINS_full [12], to conduct our experiments. These datasets are widely used as benchmark datasets for evaluating GNNs [61, 108, 28, 31]. Among them, Citeseer, Cora, and Pubmed are citation datasets with nodes representing publications and links indicating citations among these publications. The other five datasets are chemical datasets, each node is a molecule and each link represents the interaction between two molecules. All these datasets have nodes' attributes and labels.

Datasets Configuration: For each dataset, we train a target model and a reference model. In particular, we randomly sample 10% nodes and use their ground truth labels to train the target model and the reference model.[1] Recall that several attacks require the knowledge of the target dataset's partial graph. To simulate and fairly evaluate different attacks, we construct an *attack dataset* which contains node pairs and labels representing whether they are linked or not. Specifically, we first select all node pairs that are linked. Then, we randomly sample the same number of node pairs that are not linked. We note that such negative sampling approach follows the common practice in the literature of link prediction [42, 6, 123]. Furthermore, the main metric we use, i.e., AUC (introduced below), is insensitive to the class imbalance issue [34, 6, 86] contrary to accuracy. Next, we split the attack dataset randomly by half into *attack training dataset* and *attack testing dataset*.[2] We use the attack training dataset to train our attack models when the target dataset's partial graph is part of the adversary's knowledge. We use attack testing dataset to evaluate all our attacks. For the attacks that have a shadow dataset, we also construct an attack dataset on the shadow dataset to train the attack model. Note that we do not split this attack dataset because we do not use it for evaluation.

Metric: We use AUC (area under the ROC curve) as our main evaluation metric. AUC is frequently used in binary classification tasks [34, 6, 86, 85, 44, 123, 58], it is threshold independent. For convenience, we refer to node pairs that are linked as *positive node pairs* and those that are not linked as *negative node pairs*. If we rank node pairs according to the probability that there is a link between them, then AUC is the probability that a randomly selected positive node pair ranks higher than a randomly selected negative node pair. When performing random guessing, i.e., we rank all node pairs uniformly at random, the AUC value is 0.5. Note that we also calculate Precision and Recall for all supervised attacks (see Table 5.16, Table 5.17, Table 5.18, Table 5.19, Table 5.20, and Table 5.21 in Appendix).

Models: We use a graph convolutional network with 2 hidden layers for both the target model and the shadow target model, and assume they share the same architecture (see Section 5.2). Note that we also evaluate the scenario where the target model and the shadow model have different architectures later in this section and find the performances of our attacks are similar. The number of neurons in the hidden layer is set to 16. We adopt the frequently used ReLU and softmax as activation functions for the first hidden layer and the second hidden layer, respectively. Note that we append Dropout (the rate is 0.5) to the output of the hidden layer to prevent overfitting. We train 100 epochs with a learning rate of 0.01. Cross-entropy is adopted as the loss function and we use the Adam optimizer to update the model parameters. Our GNNs are implemented based on publicly available code.[3] Experimental results show that our GNNs achieve similar performance as reported in other papers. We omit them to preserve space.

[1]We do not train the reference model for attacks when $\mathcal{F}$ is unavailable.

[2]We perform additional experiments and observe that training set size does not have a strong impact on the attack performance, results are presented in Figure 5.7 in Appendix.

[3]`https://github.com/tkipf/gcn`

Table 5.3: Average AUC with standard deviation for Attack-1 on all the 8 datasets. Best results are highlighted in bold.

	Shadow Dataset							
Target Dataset	AIDS	COX2	DHFR	ENZYMES	PROTEINS_full	Citeseer	Cora	Pubmed
AIDS	-	0.720 ± 0.009	0.690 ± 0.005	**0.730 ± 0.010**	0.720 ± 0.005	0.689 ± 0.019	0.650 ± 0.025	0.667 ± 0.014
COX2	0.755 ± 0.032	-	0.831 ± 0.005	0.739 ± 0.116	**0.832 ± 0.009**	0.762 ± 0.009	0.773 ± 0.008	0.722 ± 0.024
DHFR	0.689 ± 0.004	**0.771 ± 0.004**	-	0.577 ± 0.044	0.701 ± 0.010	0.736 ± 0.005	0.740 ± 0.003	0.663 ± 0.010
ENZYMES	**0.747 ± 0.014**	0.695 ± 0.023	0.514 ± 0.041	-	0.691 ± 0.030	0.680 ± 0.012	0.663 ± 0.009	0.637 ± 0.018
PROTEINS_full	0.775 ± 0.020	0.821 ± 0.016	0.528 ± 0.038	0.822 ± 0.020	-	**0.823 ± 0.004**	0.809 ± 0.015	0.809 ± 0.013
Citeseer	0.801 ± 0.040	0.920 ± 0.006	0.842 ± 0.036	0.846 ± 0.042	0.848 ± 0.015	-	**0.965 ± 0.001**	0.942 ± 0.003
Cora	0.791 ± 0.019	0.884 ± 0.005	0.811 ± 0.024	0.804 ± 0.048	0.869 ± 0.012	**0.942 ± 0.001**	-	0.917 ± 0.002
Pubmed	0.705 ± 0.039	0.796 ± 0.007	0.704 ± 0.042	0.708 ± 0.067	0.752 ± 0.014	0.883 ± 0.006	**0.885 ± 0.005**	-

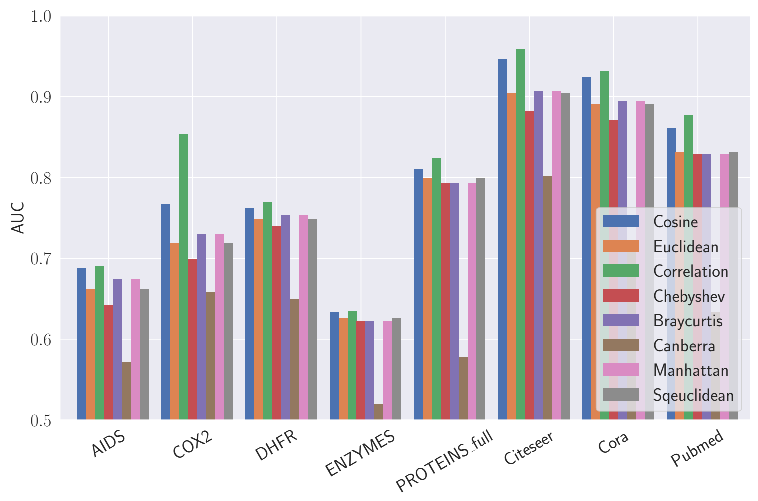


Figure 5.1: AUC for Attack-0 on all the 8 datasets with all the 8 distance metrics. The x-axis represents the dataset and the y-axis represents the AUC score.

We use an MLP with 2 hidden layers as the reference model and the shadow reference model. Hyperparameters, including the number of neurons in the hidden layer, activation functions, loss function, optimizer, epochs, and learning rate are the same as those of the target model.

We use an MLP with 3 hidden layers as our attack model. The number of neurons for all hidden layers is 32. ReLU is adopted as the activation function for hidden layers and softmax is used as the output activation function. We append Dropout (the rate is 0.5) to each hidden layer to prevent overfitting. We train 50 epochs with a learning rate of 0.001. The loss function is cross-entropy and the optimizer is Adam.

We run all experiments with this setting for 5 times and report the average value and the standard deviation of AUC scores. Note that for Attack-0 and Attack-2, the AUC scores keep the same since these two attacks are unsupervised.

5.4.2 Attack Performance

Attack-0: $\mathcal{K} = (\times, \times, \times)$**:** In this attack, the adversary only relies on measuring the distance of two nodes' posteriors obtained from the target model. We compare 8 different distance metrics and Figure 5.1 shows the results. First, we observe that

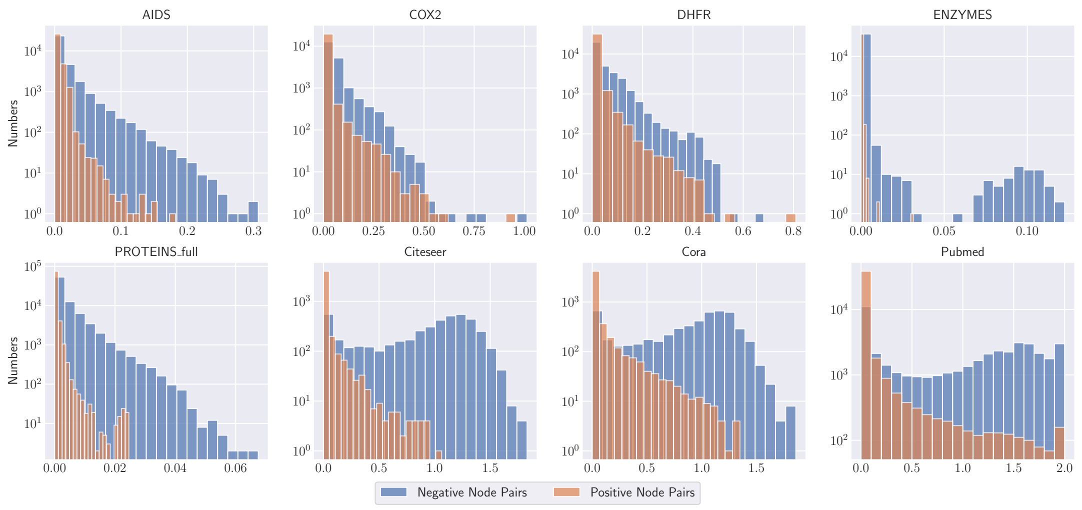


Figure 5.2: The Correlation distance distribution between nodes' posteriors for positive node pairs and negative node pairs on all the 8 datasets. The x-axis represents Correlation distance and the y-axis represents the number of node pairs.

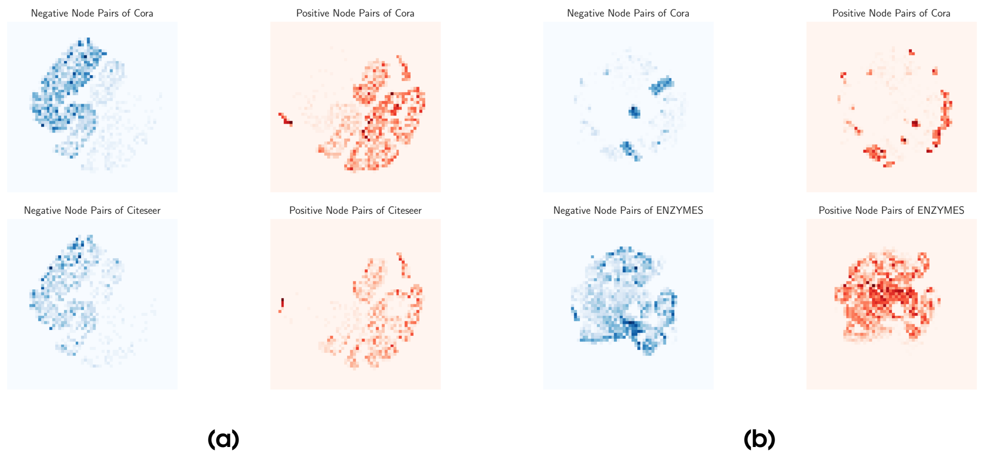


Figure 5.3: The last hidden layer's output from the attack model of Attack-1 for 200 randomly sampled positive node pairs and 200 randomly sampled negative node pairs projected into a 2-dimension space using t-SNE. (a) Cora as the shadow dataset and Citeseer as the target dataset, (b) Cora as the shadow dataset and ENZYMES as the target dataset.

Correlation distance achieves the best performance followed by Cosine distance across all datasets. In contrast, Canberra distance performs the worst. For instance, on the Citeseer dataset, the AUC scores for Correlation distance and Cosine distance are 0.959 and 0.946, respectively, while the AUC score for Canberra distance is 0.801. Note that both Correlation distance and Cosine distance measure the inner product between two vectors, or the "angle" of two vectors while other distance metrics do not. Second, we find that the performance of the same metric on different datasets is different. For

Table 5.4: AUC in different Correlation distance levels for Attack-0 on Pubmed.

Correlation Distance	AUC	Correlation Distance	AUC
0.00-0.01	0.608	0.02-0.03	0.407
0.01-0.02	0.535	0.03-0.04	0.399

instance, the AUC of Correlation distance on Citeseer is 0.959 compared to 0.635 on ENZYMES.

As mentioned in Section 5.3, unsupervised attacks could not provide a concrete prediction. To tackle this, we propose to use clustering, such as K-means. Concretely, we obtain a set of node pairs' distances, and perform K-means on these distances with K being set to 2. The cluster with lower (higher) average distance value is considered as the set of positive (negative) node pairs. Our experiments show that this method is effective. For instance, on the Citeseer dataset, we obtain 0.788 Precision, 0.991 Recall, and 0.878 F1-Score. The complete results are summarized in Table 5.14 in Appendix. Another method we could use is to assume that the adversary has a certain number of labeled edges, either from the target dataset or the shadow dataset. The former follows the same setting as our Attack-3, Attack-4, Attack-6, and Attack-7, and the latter is equivalent to Attack-1 and Attack-5. The corresponding results will be shown later.

Figure 5.2 shows the frequency of Correlation distance computed on posteriors obtained from the target model for both positive node pairs and negative node pairs in attack testing datasets. The x-axis is the value of Correlation distance and the y-axis is the number of pairs. A clear trend is that for all datasets, the Correlation distance for positive node pairs is much smaller than negative node pairs. We select the top 50% of node pairs with lowest Correlation distance, group them, and calculate the AUC for each group. Due to the space limit, we only show the result on Pubmed (Table 5.4). We can see that the AUC drops when the Correlation distance increase, which indicates that Attack-0 works better on node pairs with lower Correlation distance. In general, the posteriors for positive node pairs are "closer" than that for negative node pairs. This verifies our intuition in Section 5.3: GNN can be considered as an aggregation function over the neighborhoods, if two nodes are linked, they aggregate with each other's features and therefore become closer.

Attack-1: $\mathcal{K} = (\times, \times, \mathcal{D}')$: In this attack, the adversary can leverage a shadow dataset. In particular, for each dataset, we use one of the remaining datasets as the shadow dataset to perform the attack. Table 5.3 summarizes the results. We leave the blank in the diagonal because we do not use the target dataset itself as its shadow dataset.

As we can see from Table 5.3, the AUC scores from the best-performing shadow dataset have a consistent improvement on almost all datasets compared to Attack-0. One exception is the COX2 dataset in which the AUC score decreases by 0.02. The results indicate that the adversary can indeed transfer the knowledge from the shadow dataset to enhance her attack.

An interesting finding is that for a chemical dataset, the best shadow dataset is normally a chemical dataset as well. Similar results can be observed for citation datasets. This shows that it is more effective to transfer knowledge across datasets from the

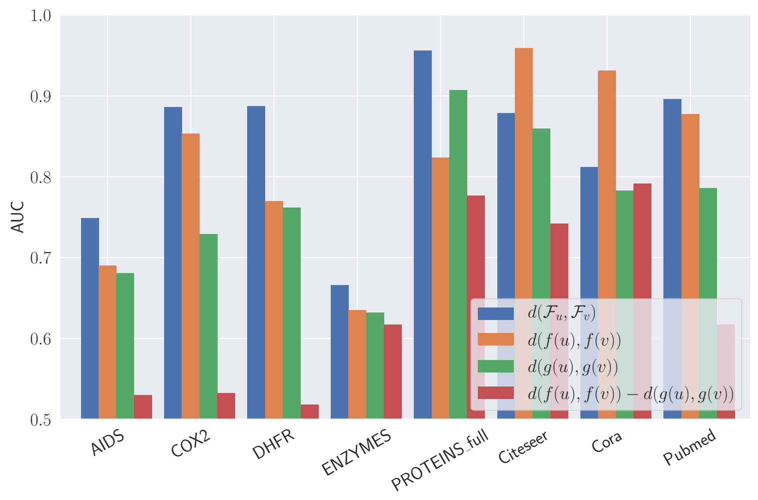


Figure 5.4: Average AUC for Attack-2 on all the 8 datasets with all the 4 types of information considered. The x-axis represents the dataset and the y-axis represents the AUC score.

same domain. To better understand this, we extract the attack model's last hidden layer's output (32-dimension) for positive node pairs and negative node pairs and project them into a 2-dimension space using t-Distributed Stochastic Neighbor Embedding (t-SNE) [75]. Figure 5.3a shows the results for Citeseer when using Cora as the shadow dataset, both of which are citation datasets. We can see that the positive (negative) node pairs from both the target dataset and the shadow dataset can be clustered into similar position, which indicates the positive (negative) node pairs from both datasets have similar distributions. This means if the attack model learns a decision boundary to separate positive nodes pairs from the negative node pairs on the shadow dataset, this decision boundary can be easily carried over to the target dataset.

In contrast, Figure 5.3b shows the results for ENZYMES (a chemical dataset) when using Cora (a citation dataset) as the shadow dataset. We see that the positive (negative) node pairs from the shadow dataset and the target dataset are distributed differently in the 2-dimension space. For example, the positive node pairs for Cora are clustered into the outer space of the circle area whereas the positive node pairs for ENZYMES are clustered into the inner space of the circle area. Therefore, it is hard for the adversary to perform an effective transferring attack. The underlying reason for this to happen is that graphs from the same domain have analogous graph structures and similar features. This leads to less information loss for our transferring attack.

Attack-2: $\mathcal{K} = (\mathcal{F}, \times, \times)$**:** In Attack-2, the adversary has the knowledge of the target dataset's nodes' attributes. As discussed in Section 5.3, she trains a reference model g by herself from $\mathcal{F}$. We compare four types of information mentioned in Section 5.3, and the results are shown in Figure 5.4. Note that we only show the results calculated with Correlation distance out of the 8 distance metrics (Table 5.12) since Correlation distance achieves the best performance in almost all settings. We can see that in all

Table 5.5: Average AUC with standard deviation for Attack-3 on all the 8 datasets.

Dataset	AUC	Dataset	AUC
AIDS	0.961 ± 0.001	PROTEINS_full	0.958 ± 0.000
COX2	0.939 ± 0.002	Citeseer	0.973 ± 0.000
DHFR	0.934 ± 0.001	Cora	0.954 ± 0.001
ENZYMES	0.882 ± 0.001	Pubmed	0.947 ± 0.001

Table 5.6: Average AUC with standard deviation for Attack-4 on all the 8 datasets. Best results are highlighted in bold.

	Shadow Dataset							
Target Dataset	AIDS	COX2	DHFR	ENZYMES	PROTEINS_full	Citeseer	Cora	Pubmed
AIDS	-	0.750 ± 0.009	**0.763 ± 0.010**	0.733 ± 0.007	0.557 ± 0.009	0.729 ± 0.015	0.702 ± 0.010	0.673 ± 0.009
COX2	0.802 ± 0.031	-	**0.866 ± 0.004**	0.782 ± 0.012	0.561 ± 0.030	0.860 ± 0.002	0.853 ± 0.004	0.767 ± 0.023
DHFR	0.758 ± 0.022	**0.812 ± 0.005**	-	0.662 ± 0.030	0.578 ± 0.067	0.799 ± 0.002	0.798 ± 0.009	0.736 ± 0.005
ENZYMES	**0.741 ± 0.010**	0.684 ± 0.024	0.670 ± 0.008	-	0.733 ± 0.019	0.624 ± 0.002	0.627 ± 0.014	0.691 ± 0.012
PROTEINS_full	0.715 ± 0.009	0.802 ± 0.025	0.725 ± 0.041	0.863 ± 0.010	-	0.784 ± 0.031	0.815 ± 0.012	**0.867 ± 0.003**
Citeseer	0.832 ± 0.078	0.940 ± 0.005	0.914 ± 0.007	0.879 ± 0.062	0.833 ± 0.088	-	**0.967 ± 0.001**	0.955 ± 0.003
Cora	0.572 ± 0.188	0.899 ± 0.003	0.887 ± 0.014	0.878 ± 0.045	0.738 ± 0.168	**0.945 ± 0.001**	-	0.924 ± 0.005
Pubmed	0.777 ± 0.056	0.893 ± 0.001	0.90 ± 0.006	0.866 ± 0.002	0.806 ± 0.042	**0.907 ± 0.004**	0.902 ± 0.001	-

chemical datasets and one citation dataset, using the distance of target dataset's nodes' attributes leads to the best performance. For the other two citation datasets, using the distance between posteriors of the target model can get better performance. Nodes' attributes' dimensions are higher in citation datasets than in chemical datasets. In other words, the node attributes for citation datasets are sparser. For instance, we observe that most attributes are 0 in citation datasets. Therefore, we conclude that the attack can get better performance using the Correlation distance between posteriors of the target model when the target dataset's nodes' attributes are in high dimension.

Attack-3: $\mathcal{K} = (\times, \mathcal{A}^*, \times)$: Table 5.5 shows the results for this attack. With the knowledge of the target dataset's partial graph, the average AUC score for all cases is over 0.9. Compared to Attack-2, the AUC scores on chemical datasets have an improvement over 10% and the AUC scores on citation datasets have an improvement over 2%.[4]

Compared to Attack-1 and Attack-2, Attack-3 achieves the best performance, this indicates the target dataset's partial graph is the most important component for an adversary for performing a link stealing attack. The reason is that the partial graph contains the ground truth links in the target dataset, which can be directly exploited by the attack model.

We further investigate the contribution of each feature set to the final prediction following the methodology of Dong et al. [27]. Concretely, when studying one feature set, we set other features' value to 0. As shown in Figure 5.5, the features extracted by applying pairwise operation over posteriors are most useful for the final prediction, followed by the features based on posteriors with different distance metrics. We note that our attack also achieves over 0.70 AUC on average when only using pairwise operation over entropy of posteriors as features. Moreover, our attack achieves the best performance when taking all the three feature sets together, which implies the

[4]Attack-2 achieves relatively high AUC scores on citation datasets.

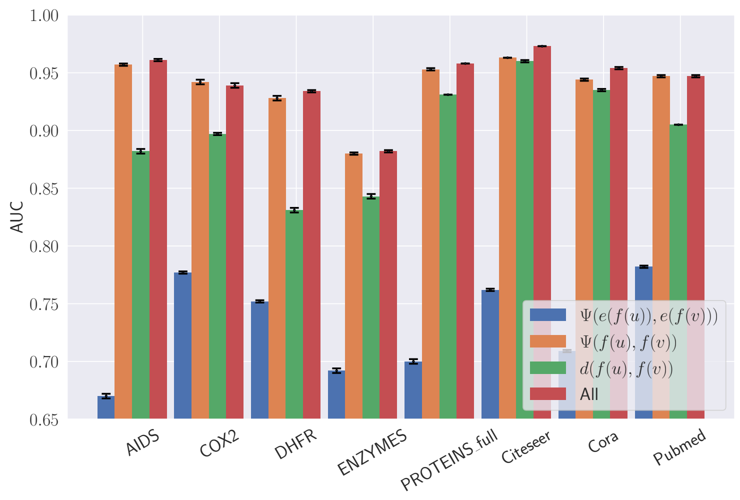


Figure 5.5: Average AUC for Attack-3 on all the 8 datasets with different set of features. The x-axis represents the dataset and the y-axis represents the AUC score.

combination of different features indeed improves the overall performance.

Attack-4: $\mathcal{K} = (\times, \mathcal{A}^*, \mathcal{D}')$**:** Table 5.6 shows the results for Attack-4. First, compared to Attack-1 ($\mathcal{K} = (\times, \times, \mathcal{D}')$), the overall performance of Attack-4 improves with the help of target dataset's partial graph $\mathcal{A}^*$. This is reasonable since the target dataset's partial graph contains some ground truth links from the target dataset. Second, we note that the performances of Attack-4 are worse than Attack-3 ($\mathcal{K} = (\times, \mathcal{A}^*, \times)$). Intuitively, the performance should be better since Attack-4 has more background knowledge. The reason for the performance degradation is that we do not take the pairwise vector operation (Table 5.13) over posteriors as the input for Attack-4 since we want to learn information from both the target dataset and the shadow dataset, and need to eliminate the dimension mismatch issue (as discussed in Section 5.3). Moreover, the results also indicate that compared to the shadow dataset, the target dataset's partial graph is more informative.

Attack-5: $\mathcal{K} = (\mathcal{F}, \times, \mathcal{D}')$**:** In Attack-5, the adversary has the knowledge of target dataset's nodes' attributes as well as a shadow dataset, evaluation results are shown in Table 5.7. We observe that Attack-5 performs better than both Attack-1 (only with $\mathcal{D}'$) and Attack-2 (only with $\mathcal{F}$). This shows the combination of $\mathcal{F}$ and $\mathcal{D}'$ can lead to a better link stealing performance. Furthermore, we observe similar trends as for Attack-1, that is the attack performs better if the shadow dataset comes from the same domain as the target dataset.

Attack-6: $\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \times)$**:** The result of Attack-6 on all datasets is shown in Table 5.9. We can see that for almost all datasets (except ENZYMES), the AUC scores are over 0.95, which means this attack achieves excellent performance. In particular, the AUC score is nearly 1 on PROTEINS_full. Moreover, Attack-6 consistently outperforms Attack-2 ($\mathcal{K} = (\mathcal{F}, \times, \times)$). This further validates the effectiveness of $\mathcal{A}^*$ in helping the

Table 5.7: Average AUC with standard deviation for Attack-5 on all the 8 datasets. Best results are highlighted in bold.

	Shadow Dataset							
Target Dataset	AIDS	COX2	DHFR	ENZYMES	PROTEINS_full	Citeseer	Cora	Pubmed
AIDS	-	0.841 ± 0.003	0.846 ± 0.009	0.795 ± 0.016	**0.875 ± 0.002**	0.839 ± 0.006	0.793 ± 0.015	0.787 ± 0.008
COX2	0.832 ± 0.036	-	**0.977 ± 0.002**	0.874 ± 0.020	0.946 ± 0.003	0.911 ± 0.004	0.908 ± 0.004	0.887 ± 0.004
DHFR	0.840 ± 0.018	**0.988 ± 0.001**	-	0.757 ± 0.032	0.970 ± 0.004	0.909 ± 0.010	0.911 ± 0.009	0.860 ± 0.004
ENZYMES	0.639 ± 0.005	0.581 ± 0.010	0.587 ± 0.005	-	0.608 ± 0.001	**0.685 ± 0.005**	0.674 ± 0.007	0.663 ± 0.002
PROTEINS_full	0.948 ± 0.007	**0.981 ± 0.004**	0.968 ± 0.014	0.818 ± 0.017	-	0.970 ± 0.002	0.876 ± 0.010	0.885 ± 0.003
Citeseer	0.773 ± 0.048	0.666 ± 0.018	0.652 ± 0.020	0.860 ± 0.049	0.794 ± 0.009	-	**0.969 ± 0.002**	0.967 ± 0.001
Cora	0.743 ± 0.017	0.587 ± 0.012	0.568 ± 0.009	0.778 ± 0.052	0.686 ± 0.018	**0.956 ± 0.001**	-	0.936 ± 0.002
Pubmed	0.777 ± 0.030	0.661 ± 0.018	0.645 ± 0.008	0.786 ± 0.041	0.741 ± 0.008	0.938 ± 0.007	**0.941 ± 0.007**	-

Table 5.8: Average AUC with standard deviation for Attack-7 on all the 8 datasets. Best results are highlighted in bold.

	Shadow Dataset							
Target Dataset	AIDS	COX2	DHFR	ENZYMES	PROTEINS_full	Citeseer	Cora	Pubmed
AIDS	-	**0.925 ± 0.001**	0.913 ± 0.005	0.784 ± 0.010	0.848 ± 0.010	0.538 ± 0.022	0.520 ± 0.011	0.849 ± 0.004
COX2	0.954 ± 0.007	-	**0.982 ± 0.001**	0.874 ± 0.010	0.898 ± 0.030	0.947 ± 0.003	0.940 ± 0.007	0.875 ± 0.034
DHFR	0.982 ± 0.002	**0.992 ± 0.00**	-	0.871 ± 0.017	0.966 ± 0.008	0.933 ± 0.008	0.947 ± 0.012	0.937 ± 0.003
ENZYMES	**0.698 ± 0.007**	0.691 ± 0.008	0.671 ± 0.003	-	0.610 ± 0.001	0.657 ± 0.009	0.662 ± 0.006	0.677 ± 0.001
PROTEINS_full	0.984 ± 0.002	0.962 ± 0.010	0.986 ± 0.002	**0.993 ± 0.001**	-	0.840 ± 0.013	0.823 ± 0.006	0.987 ± 0.005
Citeseer	0.816 ± 0.048	0.791 ± 0.033	0.702 ± 0.025	0.880 ± 0.057	0.902 ± 0.026	-	**0.977 ± 0.000**	0.964 ± 0.000
Cora	0.746 ± 0.068	0.680 ± 0.038	0.574 ± 0.038	0.888 ± 0.014	0.695 ± 0.10	**0.960 ± 0.001**	-	0.935 ± 0.001
Pubmed	0.807 ± 0.016	0.712 ± 0.025	0.710 ± 0.006	0.881 ± 0.009	0.739 ± 0.012	**0.956 ± 0.001**	0.949 ± 0.001	-

Table 5.9: Average AUC with standard deviation for Attack-6 on all the 8 datasets.

Dataset	AUC	Dataset	AUC
AIDS	0.979 ± 0.001	PROTEINS_full	0.999 ± 0.000
COX2	0.987 ± 0.001	Citeseer	0.981 ± 0.000
DHFR	0.992 ± 0.001	Cora	0.964 ± 0.000
ENZYMES	0.891 ± 0.001	Pubmed	0.970 ± 0.000

adversary to infer links. Another finding is that for chemical datasets, the information of the target dataset's partial graph brings a larger improvement than the citation datasets. One possible explanation is that the nodes' attributes in chemical datasets contain less information (they are in lower dimension), thus the target dataset's partial graph contributes more to the final prediction performance.

Attack-7: $\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \mathcal{D}')$: The results of Attack-7 are summarized in Table 5.8. Compared to Attack-5 ($\mathcal{K} = (\mathcal{F}, \times, \mathcal{D}')$), the overall performances improve with the help of $\mathcal{A}^*$. We would expect the adversary's accuracy is better than that of Attack-6 ($\mathcal{K} = (\mathcal{F}, \mathcal{A}^*, \times)$) since she has more background knowledge. However, we observe that the performance drops from Attack-6 to Attack-7. We suspect this is due to the fact that we want to learn information from both the target dataset and the shadow dataset, to avoid the dimension mismatch problem, Attack-7 uses fewer features than Attack-6 (similar to the reason that Attack-4 performs worse than Attack-3).

Comparison with Link Prediction: We further compare all our attacks with a traditional link prediction method [70]. More specifically, we build an MLP with features summarized from the target model's partial graph, including Common neighbor, Jaccard index, and Preferential attachment [70]. As we can see from Figure 5.6, most of our attacks outperforms the link prediction method. For instance, on the COX2 dataset, all

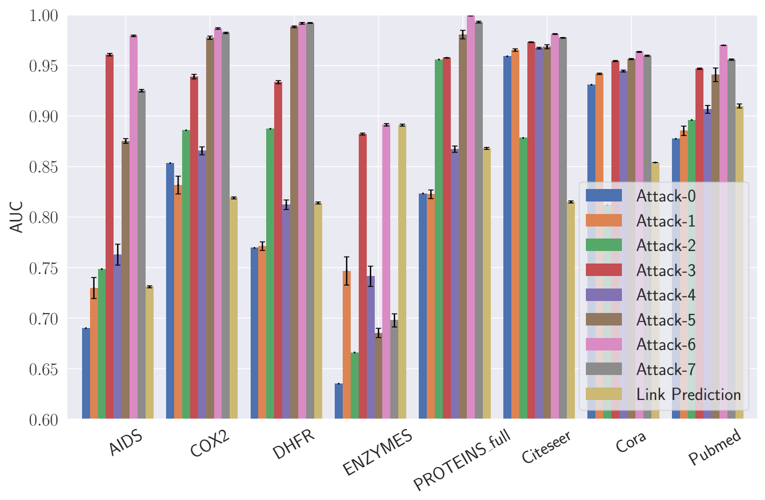


Figure 5.6: Average AUC with standard deviation for all the attacks on all the 8 datasets. For each attack, we list its best result. The x-axis represents the dataset and the y-axis represents the AUC score.

our 8 attacks outperform the link prediction model, the best attack (Attack-6) achieves more than 20% performance gain. This demonstrates that GNNs lead to more severe privacy risks than traditional link prediction.

Effect of Different GNN Structures: In our experiments, we adopt the same architecture for both the target model and the shadow target model by default for transferring attack scenarios. We further evaluate the impact of the shadow target model using different architectures. Note that for space reasons, we only report the results of Attack-1. Results for other attacks are similar. We set the number of hidden layers to 3 for the shadow target model (the target model has 2 hidden layers). The results are summarized in Table 5.15 in Appendix. We find the average AUC scores of our attack are maintained at the same level or even higher for certain datasets compared with the scenario where the shadow target model and the shadow model have the same architecture. For instance, on the Citeseer dataset, we obtain 0.924 AUC, while the original attack achieves 0.965. In other words, our attacks are still effective when the shadow target model and the target model have different architectures.

Attacks on Other GNNs: We further investigate whether our attacks are applicable to other GNN models besides GCN. Concretely, we focus on GraphSAGE [45] and GAT [108]. We implement GraphSAGE[5] and GAT[6] based on publicly available code and only report the results of Attack-6. Table 5.10 shows that our attack has similar AUC scores on GraphSAGE and GAT compared to GCN. For instance, on the COX2 dataset, our attack against GraphSAGE and GAT achieves AUC of 0.982 and 0.984, respectively (the corresponding AUC for GCN is 0.987). This further demonstrates that our attacks are generally applicable.

[5]`https://github.com/williamleif/GraphSAGE`
[6]`https://github.com/PetarV-/GAT`

Table 5.10: Average AUC with standard deviation for Attack-6 when using GraphSAGE or GAT as the target model on all the 8 datasets.

Dataset	AUC (GraphSAGE)	AUC (GAT)
AIDS	0.977 ± 0.002	0.968 ± 0.001
COX2	0.982 ± 0.001	0.984 ± 0.001
DHFR	0.990 ± 0.001	0.995 ± 0.000
ENZYMES	0.747 ± 0.001	0.766 ± 0.004
PROTEINS_full	0.999 ± 0.000	0.999 ± 0.000
Citeseer	0.938 ± 0.000	0.972 ± 0.000
Cora	0.883 ± 0.001	0.958 ± 0.000
Pubmed	0.923 ± 0.000	0.965 ± 0.000

Table 5.11: Average AUC with standard deviation for Attack-3 when only reporting top-2 posteriors on all the 8 datasets.

Dataset	AUC	Dataset	AUC
AIDS	0.855 ± 0.004	PROTEINS_full	0.954 ± 0.001
COX2	0.839 ± 0.005	Citeseer	0.958 ± 0.000
DHFR	0.851 ± 0.003	Cora	0.945 ± 0.001
ENZYMES	0.876 ± 0.002	Pubmed	0.946 ± 0.001

Possible Defense: We try to restrict the GNN model to output k largest posteriors as a defense mechanism to mitigate our attacks. The intuition is that the smaller k is, the less information the model reveals. Here, we fix $k = 2$ and report the results for Attack-3. Note that we have similar observations for other attacks. Experimental results in Table 5.11 show that this defense indeed reduces the performance of our attack. However, the performance drop is not very big, i.e., our attack still achieves relatively high AUC scores. For instance, on the Citeseer dataset, this defense reduces Attack-3's performance by less than 2%. On the AIDS dataset, the attack's performance drop is higher but AUC being 0.855 still indicates our attack is effective. We also note that the defense will impact the utility of the model. In other words, it is a trade-off between utility and privacy. In conclusion, the top-k defense is not effective enough to defend against our attacks.

We can also leverage differential privacy (DP) and adversarial examples to mitigate our attacks. In detail, we can adopt edge-DP developed for social networks [46, 122] to defend against our attacks. Borrowing the idea from previous work [57, 58], we can also add carefully crafted noise to the prediction of GNN to fool the adversary. We plan to explore both of them in the future.

Summary of Results: In summary, we have made the following observations from our experimental results.

- Our attacks can effectively steal the links from GNNs. For instance, our Attack-6 can achieve average AUC scores over 0.95 on 7 out of 8 datasets, which demonstrates that the GNNs are vulnerable to our attacks.

- Generally speaking, the performances of the attack are better if there is more background knowledge as shown in Figure 5.6. However, we find the impact of different knowledge is different. In particular, the target dataset's partial graph is the most informative. For instance, Attack-3 ($\mathcal{K} = (\times, \mathcal{A}^*, \times)$) significantly outperforms Attack-1 ($\mathcal{K} = (\times, \times, \mathcal{D}')$) and Attack-2 ($\mathcal{K} = (\mathcal{F}, \times, \times)$).

- Our transferring attack can achieve good performance. Furthermore, we find that our transferring attack achieves better performance when the shadow dataset and the target dataset are from the same domain as validated by experimental results for Attack-1 and Attack-5.

5.5 Conclusion and Future Work

In this work, we propose the first link stealing attacks against GNNs. Specifically, we show that, given a black-box access to a target GNN model, an adversary can accurately infer whether there exists a link between any pair of nodes in a graph that is used to train the GNN model. We propose a threat model to systematically characterize an adversary's background knowledge along three dimensions. By jointly considering the three dimensions, we define 8 link stealing attacks and propose novel methods to realize them. Extensive evaluation over 8 real-world datasets shows that our attacks can accurately steal links. Interesting future work includes generalizing our attacks to GNNs for graph classification and defending against our attacks.

5.6 Appendix

Table 5.12: Distance metrics, $f_i(u)$ represents the i-th component of $f(u)$. Note that these metrics can be applied to nodes' attributes as well.

Metrics	Definition
Cosine	$1 - \frac{f(u) \cdot f(v)}{\|f(u)\|_2 \, \|f(v)\|_2}$
Euclidean	$\|f(u) - f(v)\|_2$
Correlation	$1 - \frac{(f(u) - \overline{f(u)}) \cdot (f(v) - \overline{f(v)})}{\|(f(u) - \overline{f(u)})\|_2 \|(f(v) - \overline{f(v)})\|_2}$
Chebyshev	$\max_i \|f_i(u) - f_i(v)\|$
Braycurtis	$\frac{\sum \|f_i(u) - f_i(v)\|}{\sum \|f_i(u) + f_i(v)\|}$
Manhattan	$\sum_i \|f_i(u) - f_i(v)\|$
Canberra	$\sum_i \frac{\|f_i(u) - f_i(v)\|}{\|f_i(u)\| + \|f_i(v)\|}$
Sqeuclidean	$\|f(u) - f(v)\|_2^2$

Table 5.13: Pairwise vector operations, $f_i(u)$ represents the i-th component of $f(u)$. Note that these operations can be applied to nodes' attributes and entropies summarized from posteriors as well.

Operator	Definition	Operator	Definition
Average	$\frac{f_i(u)+f_i(v)}{2}$	Weighted-L1	$\lvert f_i(u)-f_i(v)\rvert$
Hadamard	$f_i(u)\cdot f_i(v)$	Weighted-L2	$\lvert f_i(u)-f_i(v)\rvert^2$

Table 5.14: Prediction results for Attack-0 on all the 8 datasets with Correlation distance.

Dataset	Precision	Recall	F1-Score	AUC
AIDS	0.524	0.996	0.687	0.691
COX2	0.523	0.987	0.684	0.867
DHFR	0.555	0.977	0.708	0.765
ENZYMES	0.501	1.000	0.667	0.630
PROTEINS_full	0.540	0.998	0.701	0.815
Citeseer	0.788	0.991	0.878	0.959
Cora	0.777	0.966	0.861	0.929
Pubmed	0.691	0.965	0.806	0.874

Table 5.15: Average AUC with standard deviation for Attack-1 with different GCN structures on all the 8 datasets. Results with respect to the best performing shadow dataset are reported.

Dataset	Shadow Dataset	AUC
AIDS	PROTEINS_full	0.729 ± 0.013
COX2	Citeseer	0.760 ± 0.026
DHFR	COX2	0.792 ± 0.005
ENZYMES	AIDS	0.732 ± 0.009
PROTEINS_full	COX2	0.808 ± 0.034
Citeseer	Cora	0.924 ± 0.006
Cora	Citeseer	0.916 ± 0.002
Pubmed	Citeseer	0.840 ± 0.001

Table 5.16: Average Precision and Recall with standard deviation for Attack-1. Results with respect to the best performing shadow dataset are reported.

Dataset	Shadow Dataset	Precision	Recall
AIDS	ENZYMES	0.725 ± 0.044	0.505 ± 0.110
COX2	PROTEINS_full	0.828 ± 0.013	0.686 ± 0.100
DHFR	COX2	0.691 ± 0.015	0.704 ± 0.022
ENZYMES	AIDS	0.639 ± 0.023	0.615 ± 0.046
PROTEINS_full	Citeseer	0.750 ± 0.022	0.800 ± 0.055
Citeseer	Cora	0.871 ± 0.005	0.958 ± 0.005
Cora	Citeseer	0.854 ± 0.003	0.883 ± 0.008
Pubmed	Cora	0.765 ± 0.009	0.897 ± 0.012

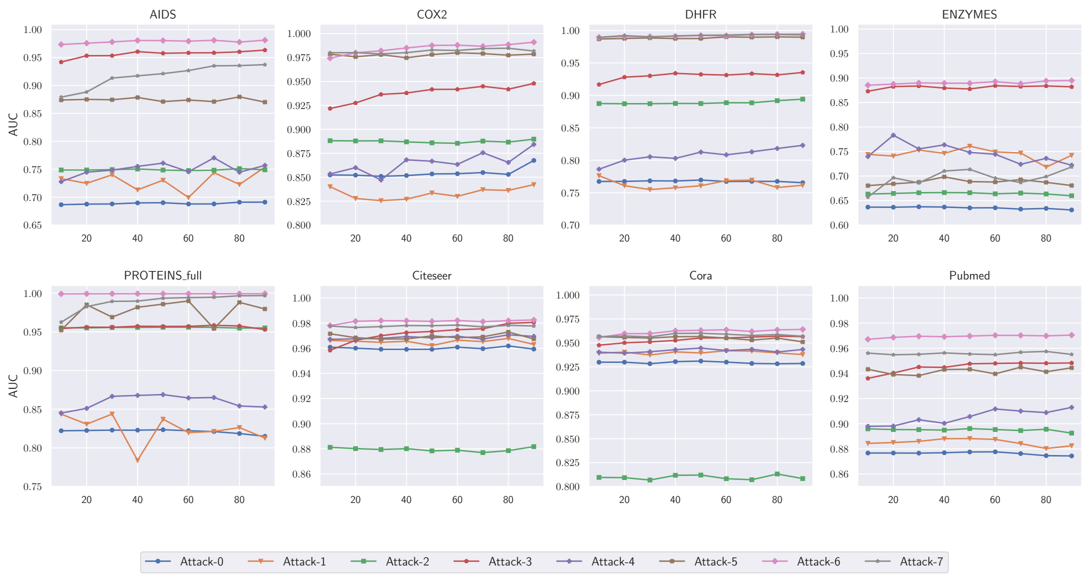


Figure 5.7: The relationship between the ratio of attack training dataset in the attack dataset and the attacks' AUC scores on all the 8 datasets. The x-axis represents the ratio and the y-axis represents the AUC score.

Table 5.17: Average Precision and Recall with standard deviation for Attack-3.

Dataset	Precision	Recall
AIDS	0.874 ± 0.006	0.966 ± 0.005
COX2	0.846 ± 0.004	0.922 ± 0.005
DHFR	0.847 ± 0.007	0.877 ± 0.009
ENZYMES	0.761 ± 0.003	0.871 ± 0.004
PROTEINS_full	0.856 ± 0.006	0.943 ± 0.004
Citeseer	0.895 ± 0.003	0.946 ± 0.005
Cora	0.858 ± 0.002	0.917 ± 0.008
Pubmed	0.869 ± 0.008	0.892 ± 0.014

Table 5.18: Average Precision and Recall with standard deviation for Attack-4. Results with respect to the best performing shadow dataset are reported.

Dataset	Shadow Dataset	Precision	Recall
AIDS	DHFR	0.688 ± 0.013	0.628 ± 0.046
COX2	DHFR	0.787 ± 0.009	0.835 ± 0.033
DHFR	COX2	0.726 ± 0.008	0.793 ± 0.015
ENZYMES	AIDS	0.637 ± 0.025	0.683 ± 0.041
PROTEINS_full	Pubmed	0.686 ± 0.045	0.955 ± 0.020
Citeseer	Cora	0.874 ± 0.004	0.956 ± 0.004
Cora	Citeseer	0.854 ± 0.002	0.896 ± 0.004
Pubmed	Citeseer	0.790 ± 0.009	0.877 ± 0.012

Table 5.19: Average Precision and Recall with standard deviation for Attack-5. Results with respect to the best performing shadow dataset are reported.

Dataset	Shadow Dataset	Precision	Recall
AIDS	PROTEINS_full	0.854 ± 0.003	0.663 ± 0.005
COX2	DHFR	0.941 ± 0.004	0.923 ± 0.022
DHFR	COX2	0.973 ± 0.004	0.942 ± 0.025
ENZYMES	Citeseer	0.608 ± 0.005	0.675 ± 0.013
PROTEINS_full	COX2	0.996 ± 0.003	0.061 ± 0.055
Citeseer	Cora	0.888 ± 0.006	0.885 ± 0.005
Cora	Citeseer	0.867 ± 0.006	0.892 ± 0.009
Pubmed	Cora	0.824 ± 0.010	0.913 ± 0.014

Table 5.20: Average Precision and Recall with standard deviation for Attack-6.

Dataset	Precision	Recall
AIDS	0.907 ± 0.002	0.986 ± 0.002
COX2	0.935 ± 0.004	0.994 ± 0.001
DHFR	0.972 ± 0.001	0.995 ± 0.002
ENZYMES	0.770 ± 0.004	0.886 ± 0.009
PROTEINS_full	0.988 ± 0.002	0.998 ± 0.001
Citeseer	0.900 ± 0.008	0.933 ± 0.006
Cora	0.878 ± 0.003	0.930 ± 0.003
Pubmed	0.903 ± 0.004	0.920 ± 0.003

Table 5.21: Average Precision and Recall with standard deviation for Attack-7. Results with respect to the best performing shadow dataset are reported.

Dataset	Shadow Dataset	Precision	Recall
AIDS	COX2	0.870 ± 0.003	0.781 ± 0.013
COX2	DHFR	0.941 ± 0.004	0.966 ± 0.009
DHFR	COX2	0.972 ± 0.002	0.994 ± 0.005
ENZYMES	AIDS	0.617 ± 0.012	0.693 ± 0.036
PROTEINS_full	ENZYMES	0.955 ± 0.004	0.974 ± 0.010
Citeseer	Cora	0.898 ± 0.003	0.913 ± 0.008
Cora	Citeseer	0.874 ± 0.004	0.911 ± 0.005
Pubmed	Citeseer	0.881 ± 0.006	0.901 ± 0.010

6 Related Work

6.1 Membership Inference Attack

In membership inference, the adversary's goal is to infer whether a given data sample is used to train a target model. Right now, membership inference is one of the major means to measure privacy risks of machine learning models [97, 118, 47, 94, 79, 104, 66, 51]. Shokri et al. [97] propose the first membership inference attack in the black-box setting. Specifically, they rely on training multiple shadow models to mimic the behavior of a target model to derive the data for training their attack models. Salem et al. [94] further relax the assumptions made by Shokri et al. [97] and propose three novel attacks. Later, Nasr et al. [79] conduct a comprehensive analysis of membership privacy under both black-box and white-box settings for centralized as well as federated learning scenarios. Song et al. [104] study the synergy between adversarial example and membership inference and show that membership privacy risks increase when a model owner applies measures to defend against adversarial example attacks. Li and Zhang [69] and Choo et al. [20] concentrated on a more restricted attack scenario (called label-only attack) where the target model only returns the predicted labels instead of posteriors when the adversary queries the target model with given samples. Roughly speaking, their proposed label-only attacks aim to infer a given sample's membership status via comparing a pre-defined threshold with the scale of adversarial perturbation that needs to be added to the given sample to change the target model's predicted label. To mitigate membership inference, many defense mechanisms have been proposed [78, 94, 58]. Nasr et al. [78] introduce an adversarial regularization term into a target model's loss function. Salem et al. [94] propose to use dropout and model stacking to reduce model overfitting, the main reason behind the success of membership inference. Jia et al. [58] rely on adversarial examples to craft noise to add to a target sample's posteriors. Also, deferentially private methods [89, 80] are introduced to mitigate membership inference.

6.2 Attribute Inference Attack

Another major type of privacy attack against ML models is attribute inference. Here, an adversary aims to infer a specific sensitive attribute of a data sample from its representation generated by a target model [76, 102]. Melis et al. [76] propose the first attribute inference attack against machine learning, in particular federated learning. Song and Shmatikov [102] later show that attribute inference attacks are also effective against another training paradigm, namely model partitioning. They further demonstrate that the success of attribute inference is due to the overlearning behavior of ML models. More recently, Song and Raghunathan [99] demonstrate that language models are also vulnerable to attribute inference.

6.3 Other Attacks Against Machine Learning Models

Besides membership inference and attribute inference, there exist a plethora of other attacks against ML models [100, 87, 55, 96, 84, 7, 93, 13, 68, 50]. One major attack is adversarial example [9, 88, 106, 14], where an adversary aims to add imperceptible noises to data samples to evade a target ML model. Another representative attack in this domain is model extraction, the goal of which is to learn a target model's parameters [107, 83, 54, 62] or hyperparameters [109, 81]. Model inversion attack is another threat to the ML models [34, 33, 87, 76, 54], whereby the adversary aims to recover the training data from target models.

6.4 Adversarial Attacks on Graph Neural Networks

Some recent studies [132, 10, 24, 133, 113, 110, 126] show that GNNs are vulnerable to adversarial attacks. In particular, the adversary can fool GNNs via manipulating the graph structure and/or node features. For instance, Zügner et al. [132] introduce adversarial attacks to attributed graphs and focus on both training and testing phase. In particular, their attacks target both node's features and graph structure and show that the node classification accuracy drops with a few perturbations. Bojchevski et al. [10] analyze the vulnerability of node embeddings to graph structure perturbation via solving a bi-level optimization problem based on eigenvalue perturbation theory. Zügner and Günnemann [133] investigate training time attacks on GNNs for node classification via treating the graph as a hyperparameter to optimize. Wang and Gong [110] propose an attack to evade the collective classification based classifier via perturbing the graph structure, which can also transfer to GNNs. Dai et al. [24] propose to fool the GNNs via manipulating the combinatorial structure of data and try to learn generalizable attack policy via reinforcement learning. Zhang et al. [126] propose a subgraph-based backdoor attack to GNN-based graph classification. In particular, a GNN classifier outputs a target label specified by an adversary when a predefined subgraph is injected to the testing graph. These studies are different from our work since we aim to steal links from GNNs.

To mitigate attacks, many defenses [11, 131, 113, 134] have been proposed. For instance, Zhu et al. [131] propose to enhance the robustness of GCNs via using Gaussian distributions in graph convolutional layers to mitigate the effects of adversarial attacks and leveraged attention mechanism to impede the propagation of attacks. Zügner and Günnemann [134] propose a learning principle that improves the robustness of the GNNs and show provable robustness guarantees against nodes' attributes perturbation. Bojchevski et al. [10] propose to certify the robustness against graph structure perturbation for a general class of models, e.g., GNNs, via exploiting connections to PageRank and Markov decision processes. These defenses are designed to improve the robustness of GNNs rather than prevent the privacy leakage of it. Note that there are also some attacks and defenses on graph that focus on non-GNN models [19, 59]. For instance, Chen et al. [19] propose attacks that mislead the behavior of the graph-cluster algorithm and show some practical defenses. Jia et al. [59] propose a certified defense that is based on randomized smoothing to defend against adversarial structural attacks to community detection.

7 Summary and Conclusion

Machine learning (ML) has achieved great success in various real-world applications. To handle the labeled data shortage and complex data form (e.g., graphs), different ML paradigms have been proposed. Despite being popular, ML models are threatened by various privacy attacks. In this dissertation, we investigate the potential privacy risks stemming from three ML paradigms, i.e., semi-supervised learning, self-supervised learning, and graph neural networks. Concretely, we perform (1) a new data augmentation-based membership inference attack against semi-supervised learning, which outperforms existing membership inference attacks, (2) a systematic privacy measurement against self-supervised learning through the lens of membership inference attacks and attribute inference attacks, and (3) a new attack against graph neural networks, namely link stealing attack, where the adversary can infer whether two nodes in the training data are linked or not. Besides introducing the attacks, we also introduce different defense mechanisms to mitigate the potential privacy leakage.

The three works presented in the dissertation are resulting in three peer-reviewed publications [P2, P1, P3]. Each work investigates the privacy risk stemming from a new ML paradigm.

Our first work [P1] investigates the privacy leakage stemming from semi-supervised learning through the lens of membership inference attacks. In Chapter 3, we propose the first data augmentation-based membership inference attack that is tailored to the training paradigm of SSL methods. Extensive evaluations show that the proposed attack outperforms existing attacks that are extended to the SSL settings. Also, we discover that the effectiveness of membership inference attacks against SSL is not credited to the commonly believed overfitting level of the model but is related to the model prediction's distinguishable entropy distributions for members and non-members. To remedy the attacks, we investigate several defense mechanisms and discover that early-stopping-based defense achieves the best trade-off between model utility and membership privacy.

Our second work [P2] quantifies the privacy leakage of contrastive learning, the most representative self-supervised learning paradigm. In Chapter 4, we consider two attacks, i.e., membership inference attacks and attribute inference attacks. We observe that, compared to supervised models, contrastive models are less vulnerable to membership inference attacks. This is because contrastive models are less prone to overfitting and the loss distribution of members and non-members is similar, which increases the membership privacy. On the other hand, we discover that contrastive models are more vulnerable to attribute inference attacks than supervised models. This is because the model trained in a contrastive way can generate more informative representations that contain rich and expressive information about the original data samples, which can be exploited for more effective attribute inference attacks. To mitigate the attribute inference risk, we propose the first privacy-preserving contrastive learning mechanism, which is able to protect the trained contrastive models from attribute inference attacks without jeopardizing their membership privacy and model utility.

Different from the first two works that focus on image data. Our third work [P3] investigates the privacy leakage from ML models (GNNs) trained with graph data. In Chapter 5, we propose the first attacks to steal a graph from the outputs of a GNN model that is trained on the graph. Concretely, given a black-box access to a GNN model, our attacks can infer whether there exists a link between any pair of nodes in the graph used to train the model. We call our attacks link stealing attacks. We propose a threat model to systematically characterize an adversary's background knowledge along three dimensions which in total leads to a comprehensive taxonomy of 8 different link stealing attacks. We propose multiple novel methods to realize these 8 attacks. Extensive experiments on 8 real-world datasets show that our attacks are effective at stealing links, e.g., AUC (area under the ROC curve) is above 0.95 in multiple cases. Our results indicate that the outputs of a GNN model reveal rich information about the structure of the graph used to train the model.

Future Research Directions: This dissertation conducts research that sheds some light on the privacy risks of emerging machine learning paradigms. And we would like to discuss some future research directions.

Firstly, in this dissertation, we mainly focus on the privacy risks of training data. However, the privacy risk of the model itself should also be taken into account as it may take a huge effort in terms of collecting the data and training the model. Model stealing attacks [107, 83, 73] are a major threat to model privacy whereby the adversary aims to reproduce the functionality of the target model with only (black-box) query access to it. A successful model stealing attack may not only compromise the intellectual property of the model owner but also serve as a stepping stone for further attacks. Our preliminary work [S9, S7] shows that pre-trained image encoders and GNNs are vulnerable to model stealing attacks. We also explore possible protection on pre-trained image encoders against model stealing attacks [S1]. For future work, it would be worth exploring more effective model stealing attacks as well as the countermeasures on emerging ML paradigms.

Secondly, there exist different types of privacy attacks such as membership inference, attribute inference, model stealing, etc. However, those attacks leverage a variety of strategies and are conducted in different models/datasets/settings. Also, different attacks are studied mainly in isolation, which lacks a holistic understanding of the risks caused by these attacks. The aforementioned limitations hinder the progress of new research in this domain and pose a challenge for practitioners to appropriately select optimal attack or defense strategies. Hence, as a future research direction, we plan to build the benchmark tool for different privacy attacks and defenses to have a more comprehensive overview of a model's privacy risk. Also, current privacy assessments mainly rely on quantifying the performance of different attacks on the target model. Therefore, another purpose for the benchmark tool is to develop a more simplified metric to represent the privacy leakage of the target model. Moreover, this metric can be used to jointly train the target model to improve its robustness against different attacks.

Thirdly, state-of-the-art ML paradigms including text-to-image generation models (e.g., DALL·E2 and Stable Diffusion) and large language models (e.g., ChatGPT) have shown great potential in generating realistic content. Besides privacy risks, these models also raise concerns about the misuse of the generated content (images or texts). For instance, people can use text-to-image generation models to generate fake but realistic images for propaganda. Also, large language models like ChatGPT can be used to generate assignments, scientific papers, or even dissertations, making a fair judgment impossible. Therefore, it is essential to develop effective detection and attribution methods against these models. One promising direction is to build robust classifiers to distinguish the generated contents from the real ones. Another solution is to integrate a watermarking module into the content generation process so that every generated content would contain a watermark that can be easily traced.

Bibliography

[P1] He, X., Liu, H., Gong, N. Z., and Zhang, Y. Semi-Leak: Membership Inference Attacks Against Semi-supervised Learning. In: *European Conference on Computer Vision (ECCV)*. Springer, 2022, 365–381.

[P2] He, X. and Zhang, Y. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2021, 845–863.

[P3] He, X., Jia, J., Backes, M., Gong, N. Z., and Zhang, Y. Stealing Links from Graph Neural Networks. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2021, 2669–2686.

Other Published Papers of the Author

[S1] Cong, T., He, X., and Zhang, Y. SSLGuard: A Watermarking Scheme for Self-supervised Learning Pre-trained Encoders. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022, 579–593.

[S2] Li, Z., Liu, Y., He, X., Yu, N., Backes, M., and Zhang, Y. Auditing Membership Leakages of Multi-Exit Networks. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022, 1917–1931.

[S3] Liu, Y., Wen, R., He, X., Salem, A., Zhang, Z., Backes, M., Cristofaro, E. D., Fritz, M., and Zhang, Y. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2022, 4525–4542.

[S4] Ma, Y., Zhang, Z., Yu, N., He, X., Backes, M., Shen, Y., and Zhang, Y. Generated Graph Detection. In: *International Conference on Machine Learning (ICML)*. PMLR, 2023.

[S5] Qu, Y., He, X., Pierson, S., Backes, M., Zhang, Y., and Zannettou, S. On the Evolution of (Hateful) Memes by Means of Multimodal Contrastive Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2023.

[S6] Qu, Y., Shen, X., He, X., Backes, M., Zannettou, S., and Zhang, Y. Unsafe Diffusion: On the Generation of Unsafe Images and Hateful Memes From Text-To-Image Models. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2023.

[S7] Sha, Z., He, X., Yu, N., Backes, M., and Zhang, Y. Can't Steal? Cont-Steal! Contrastive Stealing Attacks Against Image Encoders. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023.

[S8] Shen, X., He, X., Backes, M., Blackburn, J., Zannettou, S., and Zhang, Y. On Xing Tian and the Perseverance of Anti-China Sentiment Online. In: *International Conference on Web and Social Media (ICWSM)*. AAAI, 2022, 944–955.

[S9] Shen, Y., He, X., Han, Y., and Zhang, Y. Model Stealing Attacks Against Inductive Graph Neural Networks. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2022, 1175–1192 (The first two authors made equal contribution).

[S10] Yang, Z., He, X., Li, Z., Backes, M., Humbert, M., Berrang, P., and Zhang, Y. Data Poisoning Attacks Against Multimodal Encoders. In: *International Conference on Machine Learning (ICML)*. PMLR, 2023.

[S11] Zhang, B., He, X., Shen, Y., Wang, T., and Zhang, Y. A Plot is Worth a Thousand Words: Model Information Stealing Attacks via Scientific Plots. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2023.

Other Technical Reports of the Author

[T1] He, X., Li, Z., Xu, W., Cornelius, C., and Zhang, Y. Membership-Doctor: Comprehensive Assessment of Membership Inference Against Machine Learning Models. *CoRR abs/2208.10445* (2022).

[T2] He, X., Wen, R., Wu, Y., Backes, M., Shen, Y., and Zhang, Y. Node-Level Membership Inference Attacks Against Graph Neural Networks. *CoRR abs/2102.05429* (2021).

[T3] Sha, Z., He, X., Berrang, P., Humbert, M., and Zhang, Y. Fine-Tuning Is All You Need to Mitigate Backdoor Attacks. *CoRR abs/2212.09067* (2022).

Other references

[1] `https://github.com/pytorch/opacus`.

[2] `https://www.cs.toronto.edu/~kriz/cifar.html`.

[3] `https://github.com/Trusted-AI/adversarial-robustness-toolbox`.

[4] Abadi, M., Chu, A., Goodfellow, I., McMahan, B., Mironov, I., Talwar, K., and Zhang, L. Deep Learning with Differential Privacy. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2016, 308–318.

[5] Atwood, J. and Towsley, D. Diffusion-Convolutional Neural Networks. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2016, 1993–2001.

[6] Backes, M., Humbert, M., Pang, J., and Zhang, Y. walk2friends: Inferring Social Links from Mobility Profiles. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, 1943–1957.

[7] Béguelin, S. Z., Wutschitz, L., Tople, S., Rühle, V., Paverd, A., Ohrimenko, O., Köpf, B., and Brockschmidt, M. Analyzing Information Leakage of Updates to Natural Language Models. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2020, 363–375.

[8] Berthelot, D., Carlini, N., Goodfellow, I. J., Papernot, N., Oliver, A., and Raffel, C. MixMatch: A Holistic Approach to Semi-Supervised Learning. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2019.

[9] Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., and Roli, F. Evasion Attacks against Machine Learning at Test Time. In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*. Springer, 2013, 387–402.

[10] Bojchevski, A. and Günnemann, S. Adversarial Attacks on Node Embeddings via Graph Poisoning. In: *International Conference on Machine Learning (ICML)*. PMLR, 2019, 695–704.

[11] Bojchevski, A. and Günnemann, S. Certifiable Robustness to Graph Perturbations. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2019, 8317–8328.

[12] Borgwardt, K. M., Ong, C. S., Schönauer, S., Vishwanathan, S. V. N., Smola, A. J., and Kriegel, H.-P. Protein Function Prediction via Graph Kernels. *Bioinformatics* (2005).

[13] Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T. B., Song, D., Erlingsson, Ú., Oprea, A., and Raffel, C. Extracting Training Data from Large Language Models. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2021, 2633–2650.

[14] Carlini, N. and Wagner, D. Towards Evaluating the Robustness of Neural Networks. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017, 39–57.

[15] Chen, D., Yu, N., Zhang, Y., and Fritz, M. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2020, 343–362.

[16] Chen, M., Zhang, Z., Wang, T., Backes, M., Humbert, M., and Zhang, Y. Graph Unlearning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022.

[17] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. A Simple Framework for Contrastive Learning of Visual Representations. In: *International Conference on Machine Learning (ICML)*. PMLR, 2020, 1597–1607.

[18] Chen, X., Fan, H., Girshick, R. B., and He, K. Improved Baselines with Momentum Contrastive Learning. *CoRR abs/2003.04297* (2020).

[19] Chen, Y., Nadji, Y., Kountouras, A., Monrose, F., Perdisci, R., Antonakakis, M., and Vasiloglou, N. Practical Attacks Against Graph-based Clustering. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, 1125–1142.

[20] Choo, C. A. C., Tramèr, F., Carlini, N., and Papernot, N. Label-Only Membership Inference Attacks. In: *International Conference on Machine Learning (ICML)*. PMLR, 2021, 1964–1974.

[21] Coates, A., Ng, A. Y., and Lee, H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. JMLR, 2011, 215–223.

[22] Coavoux, M., Narayan, S., and Cohen, S. B. Privacy-preserving Neural Representations of Text. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, 2018, 1–10.

[23] Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020, 18613–18624.

[24] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J., and Song, L. Adversarial Attack on Graph Structured Data. In: *International Conference on Machine Learning (ICML)*. PMLR, 2018, 1123–1132.

[25] Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2016, 3837–3845.

[26] Dobson, P. D. and Doig, A. J. Distinguishing Enzyme Structures from Non-Enzymes without Alignments. *Journal of Molecular Biology* (2003).

[27] Dong, Y., Johnson, R. A., and Chawla, N. V. Will This Paper Increase Your *h*-index?: Scientific Impact Prediction. In: *ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2015, 149–158.

[28] Dwivedi, V. P., Joshi, C. K., Laurent, T., Bengio, Y., and Bresson, X. Benchmarking Graph Neural Networks. *CoRR abs/2003.00982* (2020).

[29] Edwards, H. and Storkey, A. J. Censoring Representations with an Adversary. In: *International Conference on Learning Representations (ICLR)*. 2016.

[30] Elazar, Y. and Goldberg, Y. Adversarial Removal of Demographic Attributes from Text Data. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, 2018, 11–21.

[31] Errica, F., Podda, M., Bacciu, D., and Micheli, A. A Fair Comparison of Graph Neural Networks for Graph Classification. In: *International Conference on Learning Representations (ICLR)*. 2020.

[32] Fan, W., Ma, Y., Li, Q., He, Y., Zhao, Y. E., Tang, J., and Yin, D. Graph Neural Networks for Social Recommendation. In: *The Web Conference (WWW)*. ACM, 2019, 417–426.

[33] Fredrikson, M., Jha, S., and Ristenpart, T. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2015, 1322–1333.

[34] Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., and Ristenpart, T. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2014, 17–32.

[35] Ganin, Y. and Lempitsky, V. S. Unsupervised Domain Adaptation by Backpropagation. In: *International Conference on Machine Learning (ICML)*. JMLR, 2015, 1180–1189.

[36] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural Message Passing for Quantum Chemistry. In: *International Conference on Machine Learning (ICML)*. PMLR, 2017, 1263–1272.

[37] Giorgi, J. M., Nitski, O., Wang, B., and Bader, G. D. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. In: *Annual Meeting of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL/IJCNLP)*. ACL, 2021, 879–895.

[38] Gong, N. Z. and Liu, B. You are Who You Know and How You Behave: Attribute Inference Attacks via Users' Social Friends and Behaviors. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2016, 979–995.

[39] Gong, N. Z., Talwalkar, A., Mackey, L. W., Huang, L., Shin, E. C. R., Stefanov, E., Shi, E., and Song, D. Joint Link Prediction and Attribute Inference Using a Social-Attribute Network. *ACM Transactions on Intelligent Systems and Technology* (2014).

[40] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2014, 2672–2680.

[41] Goyal, P. and Ferrara, E. Graph embedding techniques, applications, and performance: A survey. *Knowledge Based Systems* (2018).

[42] Grover, A. and Leskovec, J. node2vec: Scalable Feature Learning for Networks. In: *ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2016, 855–864.

[43] Gutmann, M. and Hyvärinen, A. Noise-Contrastive Estimation: A New Estimation Principle for Unnormalized Statistical Models. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. JMLR, 2010, 297–304.

[44] Hagestedt, I., Zhang, Y., Humbert, M., Berrang, P., Tang, H., Wang, X., and Backes, M. MBeacon: Privacy-Preserving Beacons for DNA Methylation Data. In: *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

[45] Hamilton, W. L., Ying, Z., and Leskovec, J. Inductive Representation Learning on Large Graphs. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2017, 1025–1035.

[46] Hay, M., Li, C., Miklau, G., and Jensen, D. D. Accurate Estimation of the Degree Distribution of Private Networks. In: *International Conference on Data Mining (ICDM)*. IEEE, 2009, 169–178.

[47] Hayes, J., Melis, L., Danezis, G., and Cristofaro, E. D. LOGAN: Evaluating Privacy Leakage of Generative Models Using Generative Adversarial Networks. *Privacy Enhancing Technologies Symposium* (2019).

[48] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. Momentum Contrast for Unsupervised Visual Representation Learning. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, 9726–9735.

[49] He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, 770–778.

[50] He, X., Jia, J., Backes, M., Gong, N. Z., and Zhang, Y. Stealing Links from Graph Neural Networks. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2021, 2669–2686.

[51] He, X., Wen, R., Wu, Y., Backes, M., Shen, Y., and Zhang, Y. Node-Level Membership Inference Attacks Against Graph Neural Networks. *CoRR abs/2102.05429* (2021).

[52] He, X. and Zhang, Y. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2021, 845–863.

[53] Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., and Bengio, Y. Learning Deep Representations by Mutual Information Estimation and Maximization. In: *International Conference on Learning Representations (ICLR)*. 2019.

[54] Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A., and Papernot, N. High Accuracy and High Fidelity Extraction of Neural Networks. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2020, 1345–1362.

[55] Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., and Li, B. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2018, 19–35.

[56] Jayaraman, B. and Evans, D. Evaluating Differentially Private Machine Learning in Practice. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2019, 1895–1912.

[57] Jia, J. and Gong, N. Z. AttriGuard: A Practical Defense Against Attribute Inference Attacks via Adversarial Machine Learning. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2018, 513–529.

[58] Jia, J., Salem, A., Backes, M., Zhang, Y., and Gong, N. Z. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2019, 259–274.

[59] Jia, J., Wang, B., Cao, X., and Gong, N. Z. Certified Robustness of Community Detection against Adversarial Structural Perturbation via Randomized Smoothing. In: *The Web Conference (WWW)*. ACM, 2020, 2718–2724.

[60] Jiao, Y., Xiong, Y., Zhang, J., Zhang, Y., Zhang, T., and Zhu, Y. Sub-graph Contrast for Scalable Self-Supervised Graph Representation Learning. *CoRR abs/2009.10273* (2020).

[61] Kipf, T. N. and Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In: *International Conference on Learning Representations (ICLR)*. 2017.

[62] Krishna, K., Tomar, G. S., Parikh, A. P., Papernot, N., and Iyyer, M. Thieves on Sesame Street! Model Extraction of BERT-based APIs. In: *International Conference on Learning Representations (ICLR)*. 2020.

[63] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2012, 1106–1114.

[64] Lee, D.-H. Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. In: *ICML Workshop on Challenges in Representation Learning (WREPL)*. ICML, 2013.

[65] Lee, J. B., Rossi, R. A., Kim, S., Ahmed, N. K., and Koh, E. Attention Models in Graphs: A Survey. *ACM Transactions on Knowledge Discovery from Data* (2019).

[66] Leino, K. and Fredrikson, M. Stolen Memories: Leveraging Model Memorization for Calibrated White-Box Membership Inference. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2020, 1605–1622.

[67] Li, J., Li, N., and Ribeiro, B. Membership Inference Attacks and Defenses in Classification Models. In: *ACM Conference on Data and Application Security and Privacy (CODASPY)*. ACM, 2021, 5–16.

[68] Li, S., Ma, S., Xue, M., and Zhao, B. Z. H. Deep Learning Backdoors. *CoRR abs/2007.08273* (2020).

[69] Li, Z. and Zhang, Y. Membership Leakage in Label-Only Exposures. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2021, 880–895.

[70] Liben-Nowell, D. and Kleinberg, J. The Link-prediction Problem for Social Networks. *Journal of the American Society for Information Science and Technology* (2007).

[71] Liu, H., Jia, J., Qu, W., and Gong, N. Z. EncoderMI: Membership Inference against Pre-trained Encoders in Contrastive Learning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2021.

[72] Liu, X., Zhang, F., Hou, Z., Wang, Z., Mian, L., Zhang, J., and Tang, J. Self-supervised Learning: Generative or Contrastive. *CoRR abs/2006.08218* (2020).

[73] Liu, Y., Jia, J., Liu, H., and Gong, N. Z. StolenEncoder: Stealing Pre-trained Encoders in Self-supervised Learning. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2022, 2115–212.

[74] Liu, Z., Luo, P., Wang, X., and Tang, X. Deep Learning Face Attributes in the Wild. In: *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015, 3730–3738.

[75] Maaten, L. van der and Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* (2008).

[76] Melis, L., Song, C., Cristofaro, E. D., and Shmatikov, V. Exploiting Unintended Feature Leakage in Collaborative Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019, 497–512.

[77] Miyato, T., Maeda, S., Koyama, M., and Ishii, S. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[78] Nasr, M., Shokri, R., and Houmansadr, A. Machine Learning with Membership Privacy using Adversarial Regularization. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2018, 634–646.

[79] Nasr, M., Shokri, R., and Houmansadr, A. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019, 1021–1035.

[80] Nasr, M., Song, S., Thakurta, A., Papernot, N., and Carlini, N. Adversary Instantiation: Lower Bounds for Differentially Private Machine Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2021.

[81] Oh, S. J., Augustin, M., Schiele, B., and Fritz, M. Towards Reverse-Engineering Black-Box Neural Networks. In: *International Conference on Learning Representations (ICLR)*. 2018.

[82] Oord, A. van den, Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *CoRR abs/1807.03748* (2018).

[83] Orekondy, T., Schiele, B., and Fritz, M. Knockoff Nets: Stealing Functionality of Black-Box Models. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, 4954–4963.

[84] Pan, X., Zhang, M., Ji, S., and Yang, M. Privacy Risks of General-Purpose Language Models. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2020, 1471–1488.

[85] Pang, J. and Zhang, Y. DeepCity: A Feature Learning Framework for Mining Location Check-Ins. In: *International Conference on Web and Social Media (ICWSM)*. AAAI, 2017, 652–655.

[86] Pang, J. and Zhang, Y. Quantifying Location Sociality. In: *ACM Conference on Hypertext and Social Media (HT)*. ACM, 2017, 145–154.

[87] Papernot, N., McDaniel, P., Sinha, A., and Wellman, M. SoK: Towards the Science of Security and Privacy in Machine Learning. In: *IEEE European Symposium on Security and Privacy (Euro S&P)*. IEEE, 2018, 399–414.

[88] Papernot, N., McDaniel, P. D., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The Limitations of Deep Learning in Adversarial Settings. In: *IEEE European Symposium on Security and Privacy (Euro S&P)*. IEEE, 2016, 372–387.

[89] Papernot, N., Song, S., Mironov, I., Raghunathan, A., Talwar, K., and Erlingsson, Ú. Scalable Private Learning with PATE. In: *International Conference on Learning Representations (ICLR)*. 2018.

[90] Raval, N., Machanavajjhala, A., and Pan, J. Olympus: Sensor Privacy through Utility Aware Obfuscation. *Privacy Enhancing Technologies Symposium* (2019).

[91] Rosvall, M. and Bergstrom, C. T. Maps of Random Walks on Complex Networks Reveal Community Structure. *Proceedings of the National Academy of Sciences* (2008).

[92] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *CoRR abs/1409.0575* (2015).

[93] Salem, A., Bhattacharya, A., Backes, M., Fritz, M., and Zhang, Y. Updates-Leak: Data Set Inference and Reconstruction Attacks in Online Learning. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2020, 1291–1308.

[94] Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., and Backes, M. ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models. In: *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

[95] Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, 4510–4520.

[96] Schuster, R., Song, C., Tromer, E., and Shmatikov, V. You Autocomplete Me: Poisoning Vulnerabilities in Neural Code Completion. *CoRR abs/2007.02220* (2020).

[97] Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership Inference Attacks Against Machine Learning Models. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017, 3–18.

[98] Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C., Cubuk, E. D., Kurakin, A., and Li, C. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

[99] Song, C. and Raghunathan, A. Information Leakage in Embedding Models. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2020, 377–390.

[100] Song, C., Ristenpart, T., and Shmatikov, V. Machine Learning Models that Remember Too Much. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2017, 587–601.

[101] Song, C. and Shmatikov, V. Auditing Data Provenance in Text-Generation Models. In: *ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2019, 196–206.

[102] Song, C. and Shmatikov, V. Overlearning Reveals Sensitive Attributes. In: *International Conference on Learning Representations (ICLR)*. 2020.

[103] Song, L. and Mittal, P. Systematic Evaluation of Privacy Risks of Machine Learning Models. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2021.

[104] Song, L., Shokri, R., and Mittal, P. Privacy Risks of Securing Machine Learning Models against Adversarial Examples. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2019, 241–257.

[105] Sutherland, J., O'Brien, L., and Weaver, D. SplineFitting with a Genetic Algorithm: A Method for Developing Classification Structure Activity Relationships. *Journal of Chemical Information and Computer Sciences* (2003).

[106] Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble Adversarial Training: Attacks and Defenses. In: *International Conference on Learning Representations (ICLR)*. 2017.

[107] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. Stealing Machine Learning Models via Prediction APIs. In: *USENIX Security Symposium (USENIX Security)*. USENIX, 2016, 601–618.

[108] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph Attention Networks. In: *International Conference on Learning Representations (ICLR)*. 2018.

[109] Wang, B. and Gong, N. Z. Stealing Hyperparameters in Machine Learning. In: *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2018, 36–52.

[110] Wang, B. and Gong, N. Z. Attacking Graph-based Classification via Manipulating the Graph Structure. In: *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2019, 2023–2040.

[111] Wang, B., Jia, J., and Gong, N. Z. Graph-based Security and Privacy Analytics via Collective Classification with Joint Weight Learning and Propagation. In: *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019.

[112] Watson, L., Guo, C., Cormode, G., and Sablayrolles, A. On the Importance of Difficulty Calibration in Membership Inference Attacks. *CoRR abs/2111.08440* (2021).

[113] Wu, H., Wang, C., Tyshetskiy, Y., Docherty, A., Lu, K., and Zhu, L. Adversarial Examples for Graph Data: Deep Insights into Attack and Defense. In: *International Joint Conferences on Artifical Intelligence (IJCAI)*. IJCAI, 2019, 4816–4823.

[114] Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, 3733–3742.

[115] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[116] Xie, Q., Dai, Z., Du, Y., Hovy, E. H., and Neubig, G. Controllable Invariance through Adversarial Feature Learning. In: *Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 2017, 585–596.

[117] Xie, Q., Dai, Z., Hovy, E. H., Luong, T., and Le, Q. Unsupervised Data Augmentation for Consistency Training. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

[118] Yeom, S., Giacomelli, I., Fredrikson, M., and Jha, S. Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting. In: *IEEE Computer Security Foundations Symposium (CSF)*. IEEE, 2018, 268–282.

[119] You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph Contrastive Learning with Augmentations. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

[120] Zagoruyko, S. and Komodakis, N. Wide Residual Networks. In: *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, 2016.

[121] Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., and Shinozaki, T. FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2021, 18408–18419.

[122] Zhang, J., Cormode, G., Procopiuc, C. M., Srivastava, D., and Xiao, X. Private Release of Graph Statistics using Ladder Functions. In: *ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, 2015, 731–745.

[123] Zhang, Y. Language in Our Time: An Empirical Analysis of Hashtags. In: *The Web Conference (WWW)*. ACM, 2019, 2378–2389.

[124] Zhang, Y., Humbert, M., Surma, B., Manoharan, P., Vreeken, J., and Backes, M. Towards Plausible Graph Anonymization. In: *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2020.

[125] Zhang, Y., Jia, R., Pei, H., Wang, W., Li, B., and Song, D. The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, 250–258.

[126] Zhang, Z., Jia, J., Wang, B., and Gong, N. Z. Backdoor Attacks to Graph Neural Networks. In: *ACM Symposium on Access Control Models and Technologies (SACMAT)*. ACM, 2021, 15–26.

[127] Zhang, Z., Song, Y., and Qi, H. Age Progression/Regression by Conditional Adversarial Autoencoder. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 2017, 4352–4360.

[128] Zhang, Z., Cui, P., and Zhu, W. Deep Learning on Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[129] Zhou, B., Lapedriza, À., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018).

[130] Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., and Sun, M. Graph Neural Networks: A Review of Methods and Applications. *CoRR abs/1812.08434* (2018).

[131] Zhu, D., Zhang, Z., Cui, P., and Zhu, W. Robust Graph Convolutional Networks Against Adversarial Attacks. In: *ACM Conference on Knowledge Discovery and Data Mining (KDD).* ACM, 2019, 1399–1407.

[132] Zügner, D., Akbarnejad, A., and Günnemann, S. Adversarial Attacks on Neural Networks for Graph Data. In: *ACM Conference on Knowledge Discovery and Data Mining (KDD).* ACM, 2018, 2847–2856.

[133] Zügner, D. and Günnemann, S. Adversarial Attacks on Graph Neural Networks via Meta Learning. In: *International Conference on Learning Representations (ICLR).* 2019.

[134] Zügner, D. and Günnemann, S. Certifiable Robustness and Robust Training for Graph Convolutional Networks. In: *ACM Conference on Knowledge Discovery and Data Mining (KDD).* ACM, 2019, 246–256.