MULTI-SOURCE KNOWLEDGE DISTILLATION FOR ROBUST AND EFFICIENT MACHINE LEARNING

CHI HONG

MULTI-SOURCE KNOWLEDGE DISTILLATION FOR ROBUST AND EFFICIENT MACHINE LEARNING

MULTI-SOURCE KNOWLEDGE DISTILLATION FOR ROBUST AND EFFICIENT MACHINE LEARNING

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus,
Prof. dr. ir. H. Bijl,
chair of the Board for Doctorates,
to be defended publicly on
Thursday, 18 June 2026, 12:30.

by

Chi HONG

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	Chairperson
Em. prof. dr. ir. D.H.J. Epema,	TU Delft, promotor
Prof. dr. Y. Chen,	TU Delft / University of Neuchatel, Switzerland, promotor
Independent Members:	
Prof. dr. O.E. Scharenborg,	TU Delft
Prof. dr. M. Kamp,	TU Dortmund / University Medicine Essen, Germany
Prof. dr. C. Dimitrakakis,	University of Neuchatel, Switzerland
Dr. G. Iosifidis,	TU Delft
Prof. dr. M.M. de Weerdt,	TU Delft, *reserve member*
Other Members:	
Dr. R.R.M. Birke,	University of Torino, Italy

Keywords: Knowledge distillation, Black-box attack, White-box attack, Data-free attack, Model compression, Probabilistic inference, Diffusion models.

Printed by: DelftrainPrint

Cover by: Dola

ISBN 978-94-6518-326-8

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

The way that can be told of is not an unvarying way.

Laozi

CONTENTS

SUMMARY

Knowledge distillation, the process of transferring learned knowledge from one target (data or model) to a substitute, has become essential for improving efficiency to reduce computational cost while maintaining accuracy. However, knowledge differs across data quality (noisy/clean), task types (classification/generation), and model accessibility (black-box/white-box). These variations introduce distinct challenges. Thus, this thesis systematically investigates how to distill knowledge from multiple sources—noisy crowdsourced labels, black-box classifiers, white-box generative models, and more complex diffusion models—to improve both robustness and efficiency.

To address these challenges, this thesis proposes five research questions, combining theoretical analysis with empirical validation across diverse machine learning scenarios. The first challenge considers noisy crowdsourced labels, where non-professional workers introduce errors that degrade model performance. It calls for online aggregation methods to process data incrementally rather than in one go on a whole set. The second vulnerability involves black-box model distillation without real data, where efficiently generating high-quality synthetic queries remains difficult. The third challenge extends this to incorporating semantic information from public data, aiming to reduce the number of queries typically required for effective distillation. The fourth investigates generative model distillation, asking whether dark knowledge (inference probabilities) exists beyond final outputs and how it improves generalization. The fifth examines diffusion models, whose multi-step Markov chain structure introduces unique difficulties for distillation and sampling acceleration.

Chapter 2 tackles distilling knowledge from noisy crowdsourced labels. Unlike offline aggregation methods requiring all labels at once, we propose BILA , an online framework that processes label chunks incrementally using a confusion matrix-based neural network model which can be trained by first-order stochastic optimizers. BILA achieves higher accuracy than existing offline algorithms, enabling robust real-time label cleaning.

Chapter 3 addresses black-box distillation without access to real training data. Existing methods only explore the input space inefficiently. We propose TANDEMGAN, which combines exploration, which generates diverse synthetic queries, with exploitation, which focuses on high-confidence queries. This tandem architecture enables effective substitute model training in general adversarial scenarios where only class labels are available.

Chapter 4 further improves black-box efficiency by incorporating semantic information from public data knowledge. We introduce AEDM, which leverages pre-trained diffusion models to generate semantically rich query images resembling real data. By optimizing the input noise of the diffusion model based on substitute model feedback, AEDM achieves superior distillation accuracy with significantly fewer queries and extends to federated learning settings.

Chapter 5 provides a theoretical analysis for generative model distillation. We derive a risk bound demonstrating that incorporating dark knowledge, which is the underlying conditional distributions between inputs and outputs, improves generalization. Our `DKtill` framework aligns student and teacher probabilistic relationships, outperforming methods that rely solely on final outputs across GANs and VAEs.

Chapter 6 targets diffusion models. Unlike prior work that merely mimics outputs, we propose SFDDM, which aligns the Markov chains of student and teacher models. By reparameterizing intermediate inputs and minimizing differences in both output and hidden variables, SFDDM produces high-quality samples with significantly fewer steps. SFDDM enables the distillation from the teacher to a student model with any desired step size.

Finally, we summarize the conclusions of this thesis. Two overarching findings emerge: (1) robust distillation requires identifying and extracting the most valuable information from each source, whether through elaborated inputs, probabilistic relationships, or structural alignment; (2) efficient distillation demands methods that match the constraints of each setting, including incremental processing for noisy data, semantic priors for black-box queries, and chain alignment for diffusion models. We also discuss limitations, including narrow architectural choices, dependency on specific probability approximations, and computational overhead, while outlining future directions such as exploring more powerful networks, alternative quality criteria, rigorous proofs for non-probabilistic generators, and extension to guided or latent diffusion models.

SAMENVATTING

Kennisdistillatie – het proces waarbij aangeleerde kennis van een bron (data of model) wordt overgedragen op een vervangend model – is essentieel geworden voor het verbeteren van de efficiëntie om rekenkosten te verlagen terwijl de nauwkeurigheid behouden blijft. Echter, kennis verschilt naargelang de datakwaliteit (ruisrijk/zuiver), taaktypen (classificatie/generatie) en modeltoegankelijkheid (black-box/white-box). Deze variaties brengen specifieke uitdagingen met zich mee. Daarom onderzoekt dit proefschrift systematisch hoe kennis kan worden gedistilleerd uit meerdere bronnen – ruisrijke crowdsourced labels, black-box classifiers, white-box generatieve modellen en complexere diffusiemodellen – om zowel de robuustheid als de efficiëntie te verbeteren.

Om deze uitdagingen aan te pakken, formuleert dit proefschrift vijf onderzoeksvragen, waarbij theoretische analyse wordt gecombineerd met empirische validatie in diverse machine learning-scenario's. De eerste uitdaging betreft ruisrijke crowdsourced labels, waarbij niet-professionele werknemers fouten introduceren die de modelprestaties verslechteren. Dit vereist online agglomeratiemethoden om data incrementeel te verwerken in plaats van in één keer op een volledige dataset. De tweede kwetsbaarheid betreft black-box distillatie zonder echte data, waarbij het efficiënt genereren van hoogwaardige synthetische queries moeilijk blijft. De derde uitdaging breidt dit uit door semantische informatie uit openbare data te integreren, met als doel het aantal doorgaans vereiste queries voor effectieve distillatie te verminderen. De vierde onderzoekt generatieve modeldistillatie, waarbij de vraag centraal staat of donkere kennis (inferentiekansen) bestaat naast de uiteindelijke outputs en hoe deze de generalisatie verbetert. De vijfde onderzoekt diffusiemodellen, waarvan de meerstaps Markov-ketenstructuur unieke moeilijkheden introduceert voor distillatie en samplingversnelling.

Hoofdstuk 2 behandelt de distillatie van kennis uit ruisrijke crowdsourced labels. In tegenstelling tot offline agglomeratiemethoden die alle labels in één keer vereisen, stellen we BILA voor: een online raamwerk dat labelchunks incrementeel verwerkt met behulp van een neuraal netwerkmodel gebaseerd op een verwarringsmatrix, dat kan worden getraind met eerste-orde stochastische optimalisatoren. BILA behaalt een hogere nauwkeurigheid dan bestaande offline algoritmen, waardoor robuuste real-time labelzuivering mogelijk wordt.

Hoofdstuk 3 richt zich op black-box distillatie zonder toegang tot echte trainingsdata. Bestaande methoden verkennen de invoerruimte slechts inefficiënt. Wij stellen TANDEMGAN voor, dat exploratie – het genereren van diverse synthetische queries – combineert met benutting – het zich richten op queries met hoge zekerheid. Deze tandemarchitectuur maakt effectieve training van vervangende modellen mogelijk in algemene adversaire scenario's waarin alleen classlabels beschikbaar zijn.

Hoofdstuk 4 verbetert de black-box efficiëntie verder door semantische informatie uit openbare datakennis te integreren. Wij introduceren AEDM, dat gebruikmaakt van voorgetrainde diffusiemodellen om semantisch rijke querybeelden te genereren die op

echte data lijken. Door de invoerruis van het diffusiemodel te optimaliseren op basis van feedback van het vervangende model, bereikt AEDM superieure distillatienauwkeurigheid met aanzienlijk minder queries en is het uitbreidbaar naar federated learning-settings.

Hoofdstuk 5 biedt een theoretische analyse voor generatieve modeldistillatie. Wij leiden een risicogrens af die aantoont dat het opnemen van donkere kennis – dat wil zeggen de onderliggende conditionele verdelingen tussen invoer en uitvoer – de generalisatie verbetert. Ons `DKtill`-raamwerk aligneert de probabilistische relaties van student- en teachermodel en overtreft methoden die uitsluitend vertrouwen op eindoutputs bij GAN's en VAE's.

Hoofdstuk 6 richt zich op diffusiemodellen. In tegenstelling tot eerder werk dat louter outputs nabootst, stellen wij SFDDM voor, dat de Markov-ketens van student- en teachermodellen aligneert. Door herparametrisatie van tussenliggende invoeren en het minimaliseren van verschillen in zowel output- als verborgen variabelen, produceert SFDDM hoogwaardige samples met aanzienlijk minder stappen. SFDDM maakt distillatie mogelijk van het teachermodel naar een studentmodel met elke gewenste stapgrootte.

Tot slot vatten wij de conclusies van dit proefschrift samen. Twee overkoepelende bevindingen komen naar voren: (1) robuuste distillatie vereist het identificeren en extraheren van de meest waardevolle informatie uit elke bron, hetzij via uitgewerkte invoer, probabilistische relaties of structurele aligning; (2) efficiënte distillatie vereist methoden die passen bij de beperkingen van elke setting, waaronder incrementele verwerking voor ruisrijke data, semantische priors voor black-box queries en ketenalignering voor diffusiemodellen. Wij bespreken ook beperkingen, zoals beperkte architectuurkeuzes, afhankelijkheid van specifieke kansbenaderingen en rekenkundige overhead, terwijl wij toekomstige richtingen schetsen, waaronder het verkennen van krachtigere netwerken, alternatieve kwaliteitscriteria, rigoureuze bewijzen voor niet-probabilistische generatoren en uitbreiding naar geleide of latente diffusiemodellen.

1

INTRODUCTION

Machine learning (ML) is widely used in today's research and industry to analyze data and make predictions. Its ability to learn and adapt without manual programming makes it valuable in areas like e-commerce, healthcare, finance, and technology. ML helps improve services, automate tasks, and support decision-making. As data grows rapidly, ML becomes increasingly important for turning data into useful insights. At its core, ML relies on two key components: data and models, which work together to generate accurate predictions. Data is crucial as it contains the patterns we aim to know, which help in predicting new observations. ML models serve as tools to capture these patterns by utilizing a large number of model parameters. To effectively learn from data, models undergo a training process where their model parameter values are adjusted to better represent the underlying patterns of the training data. Once the training is done, these trained models leverage their learned model parameters to make predictions on new observed data samples. As Albert Einstein said, "The only source of knowledge is experience." ML models gain experience through training, capturing "knowledge" for their prediction task. Therefore, knowledge exists both in the data used to train ML models and in the trained models themselves.

Knowledge distillation in ML emphasizes the knowledge embedded within models, with a particular focus on distilling knowledge from classifiers, which assign inputs to categories, and generative models, which learn data distributions to generate new samples. Classifiers and generative models represent two fundamental learning tasks. Understanding and distilling the knowledge in classifiers and generative models is essential for interpreting model behavior, reducing computational cost, and enhancing prediction accuracy. Knowledge in ML varies significantly depending on the model type and the specific task it is designed for. Different models, such as classification models and generative models, each capture and represent knowledge in different way. For instance, a neural network for classification encodes knowledge in the form of weighted connections across layers, capturing complex patterns in data. While generative models present knowledge by learning and representing the underlying probability distribution of the data, enabling them to produce new, diverse, and realistic data points for the pur-

pose of image generation, text generation, and audio synthesis, etc. The nature of the task also influences this knowledge diversity. In classification tasks, the model learns to assign inputs to discrete categories, such as identifying whether an email is spam or not. In contrast, generative tasks focus on predicting continuous values, enabling models to produce new data points (e.g., images) similar to the training data.

Since different ML tasks have varying probability distributions and output data formats, the way knowledge is represented in ML models differs across scenarios, leading to specific challenges and objectives for various models and tasks. Additionally, the knowledge from data includes features, patterns, and relationships within the data, even when it contains noise. Robust models can learn to distinguish between meaningful data and irrelevant noise, allowing them to perform well in further ML tasks. Distilling knowledge from models or data for improving the utilization of ML knowledge is essential for various applications. In data cleansing, distilled knowledge helps identify and remove noise, inconsistencies, or irrelevant information, leading to cleaner, more accurate datasets. In model compression, knowledge distillation is used to distill important information from large, complex models into smaller, more efficient ones without significant loss of performance, making them more suitable for deployment in resource-constrained environments like mobile devices. However, knowledge distillation can also be exploited in malicious applications, such as model stealing attacks, where adversaries distill knowledge from a trained model to replicate its behavior without access to the original training data, potentially compromising sensitive information.

This thesis explores knowledge distillation in ML from noisy data and neural network models, with a primary focus on distillation from models. Specifically, we design an algorithm to enable effective knowledge distillation from noisy data, two mechanisms to distill knowledge from black-box models where the models' exact parameter values are unknown, and two methods to distill knowledge from white-box models where model parameter values are accessible for producing smaller models with retained performance. We now discuss the background of the research objects (Section 1.1), the existing challenges (Section 1.2), our research questions (Section 1.3), our research methodology (Section 1.4), and the structure of this thesis and its scientific contributions (Section 1.5).

1.1. KNOWLEDGE DISTILLATION

Knowledge is reflected in multiple aspects of ML, existing in both the training data and the trained model parameters. Knowledge is also expressed differently across various models, such as classifiers [66, 105, 33] and generative models [23, 60, 36], reflecting the unique nature of each. Additionally, in different deployment scenarios, the presence and visibility of their knowledge vary, influencing how it can be distilled and utilized. Knowledge distillation in ML is significantly influenced by these various factors. Therefore, before introducing the methods, specific applications, and limitations of knowledge distillation, we will first illustrate the relevant concepts of target types (data and models), ML task types (classification and generation), and target accessibility (black-box and white-box). In Figure 1.1, we illustrate the relationship among target types, ML task types, and target accessibility. The different combinations of these factors form the structure of this thesis, with each chapter focusing on a specific scenario defined by a unique configuration of these factors. This organization enables a systematic exploration of knowledge

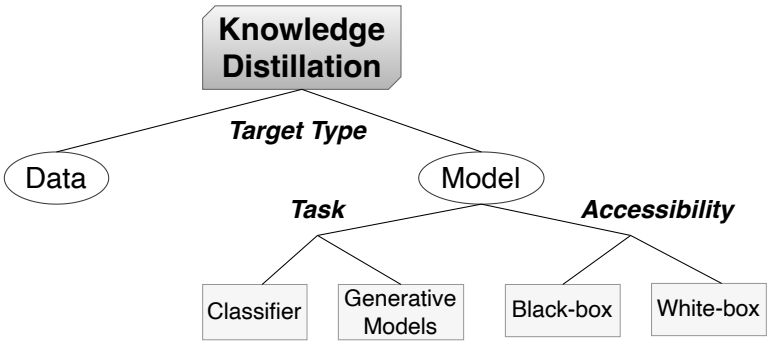


Figure 1.1: The relationship among target types, ML task types, and target accessibility for knowledge distillation.

distillation methods in various learning scenarios.

Knowledge distillation aims to transfer knowledge from targets, such as models or data, to substitutes, which can also be models or data. This process is essential because substitutes often have some better characteristics compared to the original targets. For instance, substitutes may offer higher annotation accuracy, reduced model size, or faster inference speed. By distilling and transferring knowledge, substitutes can replicate the targets while benefiting from these improved attributes. In many scenarios, substitutes can replace targets and work better. Therefore, knowledge distillation allows for more efficient and effective use of data and models, optimizing performance on various applications [100, 99, 90, 119].

In this thesis, we explored the process of distilling knowledge from data targets to data substitutes [133, 119], as well as from model targets to model substitutes [100, 26]. While both approaches are important, we focused more on model targets due to their broader range of application scenarios. For scenarios involving model targets, it is essential to consider specific ML tasks and accessibility when designing the corresponding algorithms. This thesis focuses on classification and generation tasks, with classifiers and generative models as the respective model types. These models fall into two accessibility categories: black-box and white-box.

1.1.1. BACKGROUND IN MACHINE LEARNING

LABELING

Data is the source of knowledge and it plays a crucial role in ML [14], as all model training relies on it. In supervised learning, labeled data is essential. Traditionally, experts were hired to label raw, unlabeled data, ensuring high accuracy [78]. However, as ML models grow in complexity and size, the demand for labeled data has increased, making it difficult to find enough experts to handle the large volume of data, and the cost of hiring experts can be very high. To address this, non-professional crowds have been employed to label raw data through crowdsourcing platforms like Amazon Mechanical Turk [95]. Due to the limited expertise of crowdsourcing workers, this approach, while reducing costs, also introduces unreliable labels which include many wrong labels, known as noisy la-

bels. Consequently, designing algorithms to infer and distill accurate information from these noisy labels for producing clean labels is important.

CLASSIFIERS VERSUS GENERATIVE MODELS

In ML, classification and generation are two important learning tasks. We focus on distilling knowledge from ML models of these two tasks. Classification is a fundamental task in ML that aims to assign input data to multiple predefined categories or classes. This process involves training a model on a labeled dataset, where each example is associated with a specific class label. During training, the model learns to recognize patterns in the data, allowing it to predict the class of new, unseen instances [4]. Classification is widely applied across various domains, including spam detection [11], image recognition [33, 113], and medical diagnosis [62]. The ML model used for classification is known as a classifier. Once trained, the classifier learns a mapping from input data to output classes, effectively encoding the knowledge needed to make accurate predictions. This mapping, which represents the knowledge owned by the classifier, is crucial for making correct predictions. In the following sections, we will introduce the methods used to distill this knowledge from classifiers and explore how they can be applied in different tasks, including model compression and attacks.

Generation in ML refers to the process of creating new data instances that closely resemble a given dataset. This task is typically performed using generative models, which learn the underlying distribution of the training data and use this knowledge to produce realistic synthetic outputs, such as images, text, or audio [32]. Common examples of generative models include Generative Adversarial Networks (GANs) [23], Variational Autoencoders (VAEs) [60], and diffusion models [36]. These models are widely deployed as the underlying ML models for various applications, including creative fields like art and music generation, data augmentation for improving the training of ML models, and simulation for training and testing purposes. The underlying distribution of the training data that is captured by the trained parameters of these generative models is referred to as the knowledge of the model. Understanding and distilling this knowledge is crucial for addressing several challenges in generative models, such as model compression and inference acceleration. By effectively distilling and utilizing this knowledge, researchers can improve the efficiency and performance of generative models, making them more applicable in resource-constrained scenarios.

WHITE-BOX AND BLACK-BOX MODELS

In ML, black-box and white-box models represent two different levels of target accessibility.

For black-box models [22, 63, 92], we can interact by providing an input and receiving the corresponding output, such as prediction results, generated data, or inference probabilities. However, in black-box scenarios, we lack access to the trained parameters of the model and cannot view the training dataset. This limited visibility makes understanding and distilling the target model's knowledge challenging compared to white-box scenarios. Underlying ML models deployed in online applications, such as Google Translate [25] and Google OCR [24], are usually black-box models. In these cases, service providers typically do not publish their training data, and users are unable to download or inspect the trained model parameters.

For white-box models [35, 15], we have full access to all aspects of the target models, including the trained parameters and the training data used. This comprehensive visibility facilitates knowledge distillation, allowing us to obtain more detailed and insightful information about the models. Usually, the target models in tasks like model compression and inference acceleration are white-box models.

1.1.2. KNOWLEDGE DISTILLATION UNDER VARIOUS SCENARIOS

After the background on ML in Section 1.1.1, this section will discuss knowledge distillation in various scenarios, introducing existing related works and identifying their limitations.

KNOWLEDGE DISTILLATION FROM DATA

This section discusses distilling knowledge from data targets. As previously noted, the training of large ML models often relies on labels provided by non-professional workers, which may introduce noise into the data. Directly using these noisy labels for training can negatively impact the trained model's accuracy [58, 137]. To address this issue, it is crucial to enhance the labeling accuracy. This involves designing label aggregation algorithms that distill the underlying knowledge from noisy labels to generate clean, high-accuracy labels. In this application scenario, noisy labels are the targets of knowledge distillation, while the aggregated clean labels are the substitutes. By using these clean labels instead of noisy ones during training, the performance of the ML model can be improved. An algorithm is proposed in our work to aggregate noisy labels from multiple non-professional workers and generate clean, high-quality labels to improve model training accuracy.

Existing research [58, 137, 138, 130, 125, 5] introduces various concepts to describe worker biases and behaviors, such as confusion matrices [12]. These methods often apply ML algorithms like the EM algorithm [81] and Min-Max Entropy algorithm [9] to build and train their label aggregation models. By modeling how workers label data, these algorithms can effectively infer the true label of each sample, achieving high accuracy. However, these approaches typically require collecting all noisy labels at once and then aggregating them into clean labels in an offline manner. This method is impractical for large-scale datasets. For instance, constructing a dataset like ImageNet [14], which takes years to collect, would be inefficient if it is processed offline in one go. A more efficient label aggregation manner is required. Label aggregation algorithms that work in an online way are required. An online approach allows the algorithm to handle labels continuously over subsets of data, rather than waiting to process the entire dataset at once.

BLACK-BOX KNOWLEDGE DISTILLATION

This section discusses the process of distilling knowledge from black-box target models by training a substitute model. The goal is to create a substitute model that can perform the same tasks as the target model while offering enhanced features, such as a smaller model size or faster inference. As mentioned in Section 1.1.1, the black-box scenario is challenging because the training data and the model parameters of the target are not accessible. Therefore, it is not possible to directly rely on this information to train substitute models. Instead, the only available resource for training the substitute model is

query-result pairs obtained from the target model [117, 55, 101]. By using many samples to query the target model, corresponding inference results are collected. These query-result pairs are then used to train the substitute model, allowing it to approximate the underlying input-output mapping of the target model. One application of black-box knowledge distillation is launching model-stealing attacks. Online ML services, such as OCR and image classification, often rely on black-box models accessible through APIs, which allow users to input data and receive inference results [24]. This access enables adversaries to steal knowledge from the target model by making queries and using the outputs to train a substitute model. Studying model stealing provides valuable insights into the security risks of online-deployed models.

A major challenge in black-box knowledge distillation is obtaining samples to query the target model. Since the target's training data is unknown, collecting additional samples to query the target is costly. To address this, existing methods propose to use generators to produce synthetic samples for querying target models [117, 55, 132]. These algorithms can train effective substitute models without relying on real samples. The generator in these algorithms is designed with optimization goals that are competitive to the substitute model, enabling it to efficiently explore the entire input data space. The generator aims to produce samples that the substitute model has not encountered before, maximizing the diversity of each query sample. While existing research has well studied the exploration of the input data space [117, 55, 140], further exploitation within every explored area is needed to enhance the quality of the generated query samples. Additionally, the synthetic query samples produced by current algorithms often lack human-understandable semantic information. They are more like some kind of special noise. This raises questions about whether the absence of semantic information affects the quality of black-box knowledge distillation. If semantic information is important, future algorithms should focus on generating query samples that are like real data to improve the accuracy of substitute models.

WHITE-BOX KNOWLEDGE DISTILLATION

Knowledge distillation from white-box models [35] is similar to black-box distillation in the final goal, which is to train a substitute model (student model) that captures the knowledge of the target model (teacher model). However, in the white-box scenario, more resources are accessible, including the target model's parameters, intermediate layer outputs, and the original training dataset. This additional information allows for more precise and effective training of the substitute model. The goal of white-box knowledge distillation is to create a substitute model with comparable inference performance to the target model while also providing improved characteristics. The improved characteristics of the substitute model can vary depending on the specific application. For instance, in model compression [104, 68], the aim is to produce a substitute model with a much smaller size, making it suitable for deployment in storage-constrained environments. In inference acceleration [110, 100], the focus is on creating a substitute model that offers faster and more efficient inference than the original target model.

Existing works have developed white-box distillation algorithms to distill knowledge from classifiers and classic generative models e.g., GANs and VAEs. Theoretical analysis of classifier distillation proves the significance of the target model's "dark knowledge" (the inference probabilities) in successful distillation [85]. However, similar theoretical

insights for generative models are still lacking. The key factors within the target generative model that contribute to effective distillation remain unclear. Additionally, for more complex probabilistic generative models, such as emerging diffusion models, effective and efficient distillation methods have yet to be well studied.

1.2. CHALLENGE

As discussed in Section 1.1, ML includes various target types, different levels of target accessibility, and diverse learning task types. These differences make the dynamic form of knowledge in ML, which leads to the diversity and complexity of knowledge distillation, and introduces many challenges. We summarize the challenges of knowledge distillation as follows:

[Challenge 1] Collected data from a crowd is often highly noisy. Data collected from crowds is often quite noisy. To save cost and deal with the lack of experts, much of the unlabeled raw data is labeled by non-professional workers using crowdsourcing platforms like Amazon Mechanical Turk. This approach makes it possible to quickly gather large amounts of labeled data, which is very useful for training modern machine learning models. However, the labels collected this way are often not reliable, since different workers may have different levels of skill, understanding, and attention to detail. Some may misunderstand the task, while others might make careless mistakes. As a result, the data often contains errors that can hurt the model's performance. Because of this, it becomes important to develop methods that can clean the noisy labels. These methods aim to find the true labels hidden under the noise by combining the labels from multiple workers and identifying patterns. At the same time, another problem is the size of the data. In many real-world tasks, the amount of data that needs to be labeled is very large, and the labeling process is not done all at once. It often happens over a long period, with new data coming in continuously. This means that label cleaning methods must not only be good at dealing with errors, but also be fast and efficient enough to handle large and growing datasets. To make ML systems more useful and reliable in practice, it is necessary to build simple, strong, and scalable methods that can deal with both noisy labels and large amounts of data.

[Challenge 2] Extensive and efficient searching of an input data space is difficult in knowledge distillation from black-box models without real data. To distill knowledge from black-box models, algorithms must create query samples and feed them into the target model to observe the outputs. Data-free methods, which work without access to the original training data, must rely entirely on generating synthetic samples for the queries. In this case, the input space is often very large and complex, and there is no clear direction on where useful samples may be found. Without real data as a reference, it becomes hard to tell which parts of the input space are more likely to produce informative samples for querying the model. As a result, the algorithm may spend a lot of time exploring irrelevant or unhelpful regions. This makes the search process slow and inefficient. Since black-box models do not reveal their internal structure or training data, the only feedback comes from the output, which adds to the difficulty. To make this process more effective, we need to find ways to generate samples that are not only diverse but also meaningful and likely to reveal useful information about the model's behavior.

[Challenge 3] Generating input data enriched with semantic information is nec-

essary to improve distillation efficiency compared to data-free methods. Data-free methods often produce synthetic query samples that are noisy and quite different from the actual data used to train the target model. These samples usually lack meaningful semantic content, such as recognizable objects in images. Because of this, they fail to guide the distillation process effectively. As a result, knowledge distillation becomes inefficient and slow, often requiring millions of queries to build a substitute model that can approximate the behavior of the target model. Without semantic content in the query samples, the substitute struggles to learn the true input-output patterns that the target model has captured. This makes it hard to transfer knowledge efficiently. Improving the quality and meaning of synthetic samples can help reduce the number of queries needed and make the distillation process more efficient and practical.

[Challenge 4] There is a lack of theoretical analysis when doing knowledge distillation on generative models. For classifiers, the idea of "dark knowledge" has been well studied. This refers to the full set of output probabilities produced by a classifier, not just the predicted class. Research has shown that using this richer information during the distillation process helps create better student models. These probabilities contain useful signals about class similarities and the model's internal understanding. In contrast, for generative models, such as those used for image generation, it is still unclear whether a similar form of dark knowledge exists beyond the final layer outputs. Most current approaches focus only on the final generated data, without looking into whether there is deeper information inside the generative process that could be helpful. For example, intermediate representations or sampling distributions might carry useful knowledge, but these aspects are not well studied. As a result, our understanding of how to distill and transfer knowledge from generative models remains limited, especially from a theoretical perspective.

[Challenge 5] Diffusion models introduce new difficulties for knowledge distillation due to their unique and complex design. Diffusion models have shown strong abilities in generating high-quality synthetic data, often outperforming older models like GANs and VAEs in tasks such as image generation. However, their internal structure is more complicated. While traditional generative models typically rely on a single probability distribution to describe how data is generated, diffusion models work differently. They use a sequence of steps, forming a Markov chain, to gradually transform noise into realistic data. Each step in this process is defined by its own conditional probability distribution, making the overall generation procedure more detailed and layered. Because of this, the knowledge within diffusion models is spread across many stages, rather than being concentrated in a single transformation. As a result, it becomes harder to identify which parts of the model are most useful for distillation. To distill meaningful knowledge from these models, it is important to define a clear and effective optimization goal that can guide the distillation process across all steps of the generation path. Without such an objective, it is difficult to transfer the model's capabilities into a smaller or simpler model effectively.

1.3. RESEARCH QUESTION

This thesis aims to develop algorithms for distilling knowledge from various ML targets. The overarching research question addressed is:

How can knowledge be effectively and efficiently distilled from trained neural networks and noisy data?

To address the overarching question, we propose five research sub-questions that explore various target types. These questions also cover different ML tasks. They also consider different levels of target accessibility. By exploring questions across these diverse scenarios, this thesis aims to enhance our understanding of how knowledge is represented in ML and to develop methods for knowledge distillation and its application. The following five research questions match the five challenges introduced earlier, with each question corresponding to one challenge.

[RQ1] How can we continuously aggregate noisy labels into clean labels in an online manner? Crowdsourced data contains significant noise. To address this, label aggregation methods can distill knowledge from noisy labels and transform noisy labels into clean, accurate ones. Our objective is to develop an online learning algorithm that handles labels continuously over subsets of data, rather than processing the entire dataset in one go. Additionally, we aim to ensure that this proposed algorithm converges effectively in an online learning application scenario.

[RQ2] How can we efficiently explore and exploit the input data space to generate high-quality samples for querying a black-box classifier? Data-free methods must search the input data space to generate samples that query a black-box target and obtain inference results for training a substitute model. Our goal is to enhance this process by not only exploring the input data space but also by effectively exploiting each searched area to produce higher-quality query samples. By combining exploration with exploitation, we aim to improve the knowledge distillation process, finally enabling the substitute model to achieve higher accuracy. We want to refine the quality of the generated query samples and enhance the overall effectiveness of the knowledge distillation process.

[RQ3] How can we reduce the number of queries by using semantically meaningful input data to distill knowledge from a black-box classifier? Data-free methods often require millions of queries to effectively distill knowledge from a black-box target model and train a substitute model. To address this issue, we aim to leverage the semantic information from public datasets to enhance the quality of query samples generated by the algorithm, making them more like real, understandable images. By introducing semantic information into the algorithm, we hope to improve the effectiveness of these query samples. Additionally, our goal is to successfully train a substitute model with significantly fewer queries by the enhanced algorithm, thus facilitating the knowledge distillation process and reducing costs.

[RQ4] How can we analyze the importance of inference probabilities in generative models and use them to improve knowledge distillation? Unlike classifiers, generative models typically do not provide explicit inference probabilities. We aim to explore whether these probabilities can enhance the distillation process of generative models. Our objective is to analyze their potential benefits for knowledge distillation and to propose a method for obtaining dark knowledge from generative models. By applying this method in experiments, we want to validate our analysis.

[RQ5] How can we efficiently distill a diffusion model and accelerate its sampling process? Diffusion models have a more complex architecture than generative mod-

els like GANs and VAEs due to their probability definitions, which involve two Markov chains: the reverse and forward processes. To efficiently and accurately distill these diffusion models and create student models that require fewer sampling steps, we aim to define the student model's Markov chains as a simplified version of the Markov chains of the teacher model (the target). Furthermore, our goal is to complete the distillation process in a single fold (round) to minimize the introduction of model noise and preserve the performance of the student model.

1.4. RESEARCH METHODOLOGY

Distilling knowledge in ML is essential for efficiency and robustness, especially when deploying artificial intelligence on computation-constrained and storage-constrained environments, e.g., edge devices. Besides, knowledge distillation from data can be applied for data cleaning to improve the performance of trained models. This topic has raised heated discussion among both companies and research institutions. As we search into this domain, two primary research methods emerge:

RQ	Method	Source Code Repository	Deep Learning Framework
1	BILA	[42]	TensorFlow
2	TANDEMGAN	[45]	PyTorch
3	AEDM	[38]	PyTorch
4	`DKtill`	[43]	PyTorch
5	SFDDM	[44]	PyTorch

Table 1.1: An overview of source code and frameworks used for each research question (proposed method) in this thesis.

Dataset	Data Type	Number of Samples	Number of Classes	RQ
Adult	Image classification	263	4	1
RTE	Image classification	800	4	1
Heart	Image classification	237	2	1
Age	Image classification	1002	7	1
CIFAR10	Image classification	50K	10	1, 2, 3, 5
CIFAR-100	Image classification	50K	100	3
Pedgigits	Image classification	11K	10	1
MNIST	Image classification	60K	10	3
F-MNIST	Image classification	60K	10	2
SVHN	Image classification	60K	10	2, 3
Facades	Image generation	506	N.A.	4
CelebA	Image generation	202K	N.A.	4, 5
LSUN	Image generation	120K-300K	N.A.	5

Table 1.2: An overview of public datasets used for each research question (proposed method) in this thesis.

EXPERIMENTAL APPROACH

We conduct extensive experiments using established benchmarks to evaluate the performance of the proposed methods. These studies employ a variety of datasets and distillation techniques to address challenges in real-world situations, where the distillation method may have different levels of knowledge about the model or the data. By examining how the methods perform under various conditions, such as using different target types or accessing different levels of model information, we can better understand the strengths and limitations of their approaches.

All algorithm implementations are developed entirely from scratch to ensure originality and consistency with the goals of each research task. The implementations are primarily written in Python, leveraging widely adopted deep learning libraries such as PyTorch, TensorFlow, and Keras. For reproducibility and clarity, the algorithms proposed in each chapter, along with their names and technical details, are summarized in Table 1.1. To ensure comprehensive evaluation, we adopt multiple performance metrics, including classification accuracy on substitute models, error rate of cleaned data, the number of queries required for distillation, and the Fréchet Inception Distance (FID) for evaluating generated images.

All experiments are conducted on high-performance computing infrastructure, specifically using NVIDIA RTX 3090 Ti GPUs with CUDA acceleration. This hardware configuration allows us to handle large-scale datasets and computationally intensive training in a reasonable timeframe. In cases, e.g., FID calculation, where randomness may affect the final results, each experiment is repeated at least three times with different random seeds to reduce its impact, and the reported results are given as the mean values. For evaluation across different scenarios, we rely on a wide range of publicly available benchmark datasets, which are summarized in Table 1.2 along with their key characteristics. Together, these design choices ensure that our experimental results are both reliable and representative of real-world scenarios.

THEORETICAL APPROACH

The following are the three theoretical methods we use in this thesis. First, we derive risk bounds for the proposed knowledge distillation algorithms. By showing that tighter upper bounds on the training risk lead to stronger guarantees, we provide evidence that the substitute model trained with these algorithms is more likely to generalize well in practice. In particular, we establish a risk bound for distillation methods that use dark knowledge, i.e., the conditional distribution between inputs and outputs, beyond only the generated outputs. This shows that incorporating underlying probabilistic information improves the generalization of the substitute model.

Second, stochastic optimizers are proposed for training models and prove the corresponding convergence bound. This analysis provides a formal guarantee that the optimization process will not diverge but instead move steadily toward a solution. In other words, the convergence bound ensures that the optimizer is both reliable and efficient. By connecting the practical training process with theoretical guarantees, this result strengthens the foundation of our method and shows that the improvements are not only empirical but also mathematically justified.

Additionally, this thesis also tries to analyze whether the substitute model and the teacher model have probabilistic model similarities under a certain distillation algo-

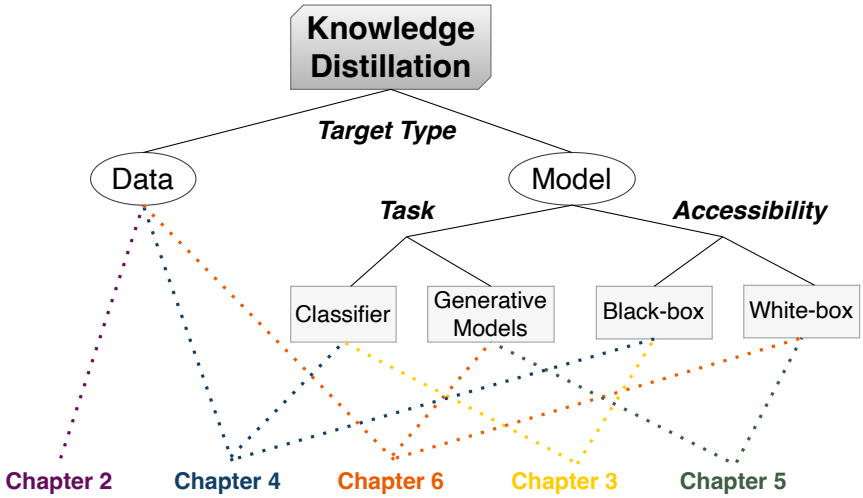


Figure 1.2: The relationship between the five technical chapters of this thesis according to target type, task, and accessibility.

rithm. High similarity means that the substitute model can more accurately reproduce the knowledge and capabilities of the target model, leading to more effective knowledge distillation. For example, we propose to align the Markov chains of the teacher and student models so that specific steps in their processes correspond to each other. This alignment ensures not only that their outputs are consistent, but also that the conditional probabilities over training samples are preserved. As a result, the student model can better mimic the teacher's behavior while simplifying the teacher's model structure.

1.5. THESIS OUTLINE AND CONTRIBUTION

In the following five chapters (Chapters 2–6), we systematically address the five research questions (RQ 1–5 from Section 1.3) in the order presented. The relationship between the five technical chapters is illustrated in Figure 1.2. In the following, we present the problems addressed in each technical chapter, describe the methods proposed to solve them, and illustrate the contributions achieved in each chapter.

1.5.1. [CHAPTER 2] DISTILLING KNOWLEDGE FROM NOISY LABELS VIA ONLINE AGGREGATION

In this chapter, we address RQ1 by proposing an online label aggregation method for distilling knowledge and producing high-quality labels based on the data of noisy labels. We propose a novel online label aggregation method, the Variational Bayesian Inference Label Aggregation (BILA) framework, which employs variational Bayesian inference method and designs a novel stochastic optimization scheme for incremental training. As shown in Figure 1.3, workers from the crowdsourcing platform continuously label the raw, unlabeled data and pass the labeled noisy data in chunks. The knowledge distilla-

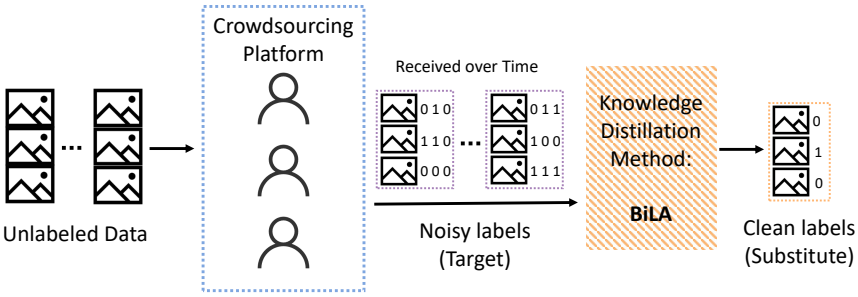


Figure 1.3: An overview of the Variational Bayesian Inference Label Aggregation (BILA) framework proposed in Chapter 2 for distilling knowledge from noisy crowdsourced labels to address RQ1.

tion algorithm is required to process this data chunk by chunk. We introduce a confusion matrix-based aggregation model, BILA-CM , which surpasses the performance of existing algorithms, including those based on EM algorithms [12], Minimax Entropy[138], and neural networks. Another key contribution is that we develop a stochastic optimizer and establish its convergence bound. Additionally, we conduct a comprehensive comparison of BILA with prominent label aggregation methods across various online crowdsourcing scenarios. This chapter is based on the following publication:

Chi Hong, Amirmasoud Ghiassi, Yichi Zhou, Robert Birke, and Lydia Y. Chen. "Online label aggregation: A variational bayesian approach." In *the Web Conference*, pp. 1904-1915. 2021.

The contributions of the authors are as follows: Chi Hong designed the framework of the algorithm, implemented the details, wrote the code, carried out the experiments, and wrote the paper. Amirmasoud Ghiassi contributed to the design of the experimental setup and conducted part of the experiments. Yichi Zhou contributed to the algorithm design, in particular the use of the confusion matrix.

1.5.2. [CHAPTER 3] EXPLORING AND EXPLOITING THE INPUT SPACE FOR BLACK-BOX KNOWLEDGE DISTILLATION.

In this chapter, we address RQ2 by exploring and exploiting the input data space of a given classifier to generate high-quality synthetic samples for querying the target. In the black-box knowledge distillation task, the primary objective is to generate synthetic query samples to obtain inference results and train the substitute model. This is achieved by efficiently and effectively searching the input data space of the target model using a generator. We consider a general adversarial setting and propose an effective data-free stealing algorithm called TANDEMGAN. The name comes from the design of its GAN, which works in tandem with two parts: one for exploration to generate diverse synthetic queries, and the other for exploitation to focus on high-quality queries. In Figure 1.4, we illustrate the workflow of TANDEMGAN. Besides the tandem exploration and

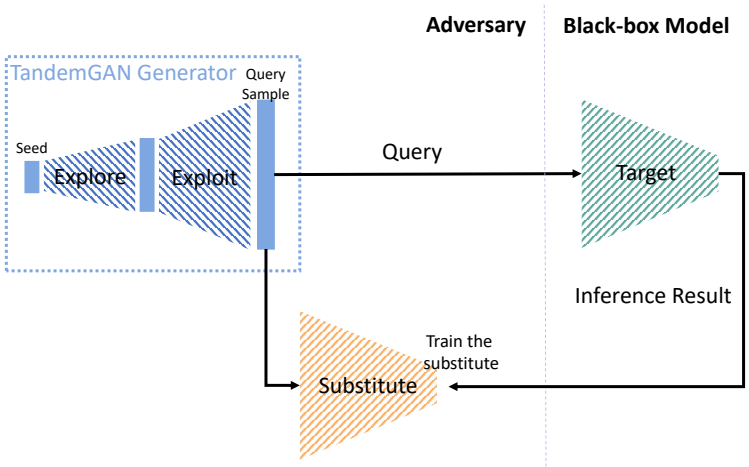


Figure 1.4: An overview of TANDEMGAN proposed in Chapter 3 for distilling knowledge from black-box classifiers by generating high-quality synthetic queries to address RQ2.

exploitation architecture, another contribution worthy of mentioning is that TANDEMGAN works for more general adversarial scenarios: only class labels are available to the adversary. The evaluation and comparison have shown the effectiveness against existing state-of-the-art data-free model stealing approaches with remarkable accuracy of the trained substitute models. This chapter is based on the following publication:

Chi Hong, Jiyue Huang, Robert Birke, and Lydia Y. Chen. "Exploring and Exploiting Data-Free Model Stealing." In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 20-35., 2023.

The contributions of the authors are as follows: Chi Hong designed the framework of the algorithm, developed the neural network for both exploration and exploitation, implemented the code, conducted the experiments, and wrote the paper. Jiyue Huang contributed to the design of the exploitation objective, assisted in running part of the experiments, and participated in writing the experimental section.

1.5.3. [CHAPTER 4] SEMANTIC QUERY GENERATION FOR EFFICIENT BLACK-BOX KNOWLEDGE DISTILLATION.

In this chapter, we address RQ3 by using public data knowledge to facilitate the knowledge distillation on the target classifiers. We introduce the Adversarial Knowledge Extraction via Steering Diffusion Models (AEDM) framework, which is designed to utilize pre-trained diffusion models to create adversarial query images. As illustrated in Figure 1.5, the public data knowledge is captured by a pre-trained diffusion model, which generates samples with rich semantic information that closely resemble real data rather than noisy data. To generate high-quality query samples, we optimize the input seed

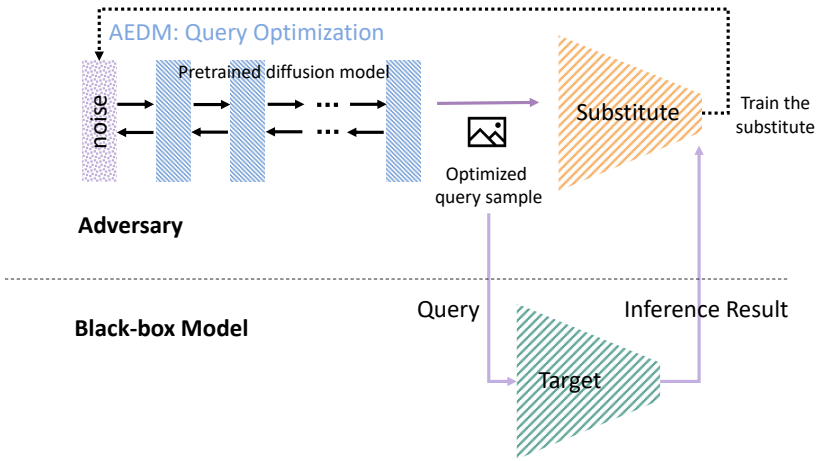


Figure 1.5: An overview of the Adversarial Knowledge Distillation via Steering Diffusion Models (AEDM) framework proposed in Chapter 4 for distilling knowledge from black-box classifiers using semantically rich queries generated by pre-trained diffusion models to address RQ3.

noise of the diffusion model based on the inference feedback from the substitute model. After this optimization process, the optimized query samples are input to the target model, obtaining corresponding inference results. These query-result pairs are then applied to train the substitute model effectively, ensuring it captures the knowledge (underlying input-output mapping) of the target model. We perform thorough comparisons between AEDM and leading data-free model stealing techniques. Our empirical results show that AEDM efficiently and effectively distills models with very few queries while achieving superior stealing accuracy. Additionally, AEDM can be deployed under Federated Learning settings. This chapter is based on the following publication:

Chi Hong, Jiyue Huang, Lydia Y. Chen, and Robert Birke. "Adversarial Knowledge Extraction via Steering Diffusion Models." In *the 31st International Conference on Neural Information Processing*, 2024.

The contributions of the authors are as follows: Chi Hong designed the overall framework of the algorithm, integrated pre-trained diffusion models to generate query samples, proposed the query optimization loss, implemented the code, conducted the experiments, and wrote the paper. Jiyue Huang contributed to selecting and tuning suitable pre-trained diffusion models, assisted in running part of the experiments, and participated in writing the experimental section and creating the figures.

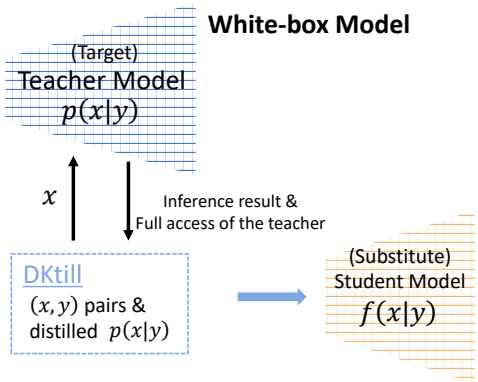


Figure 1.6: An overview of the Dark Knowledge Based Distillation (`DKtill`) framework proposed in Chapter 5 for improving generative model distillation by aligning the student and teacher models' underlying input-output conditional distributions to address RQ4.

1.5.4. [CHAPTER 5] ANALYZING INFERENCE PROBABILITIES FOR EFFECTIVE DISTILLATION OF GENERATIVE MODELS.

In this chapter, we address RQ4 by deriving a risk bound for the distillation method that leverages dark knowledge, which refers to the underlying conditional distribution between the input and the output. We demonstrate that incorporating this dark knowledge into the distillation process helps improve the generalization of the distilled substitute model. By analyzing the risk bound, we provide theoretical evidence that using underlying probabilistic information, beyond just the inference outputs, e.g., generated images, enhances the student (substitute) model's generalization, ultimately leading to better performance. We propose the Dark Knowledge Based Distillation (`DKtill`) framework, as shown in Figure 1.6, which trains the student generator by using the (approximate) dark knowledge of the target model. The key idea behind this approach is to make the underlying conditional distribution between the input and output of the student model similar to that of the teacher (target) model. By doing so, `DKtill` ensures that the student model captures the same probabilistic relationships as the teacher, allowing for a more accurate reproduction of the teacher model. We derive proposition as the approximate distillation empirical risk analysis to capture the impact of approximating dark knowledge in distilling the `GANs`. `DKtill` achieves higher distillation quality of `DKtill` on three different generative models and three different datasets, compared to distillation algorithms that do not use the proposed (approximated) dark knowledge. This chapter is based on the following publication:

Chi Hong, Robert Birke, Pin-Yu Chen, and Lydia Y. Chen. "On Dark Knowledge for Distilling Generators." In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 235-247., 2024.

The contributions of the authors are as follows: Chi Hong demonstrated the benefits

of dark knowledge for generative model distillation, derived the risk bounds, proposed and implemented the algorithm to validate the theory, wrote the code, conducted the experiments, and wrote the paper. Pin-Yu Chen suggested some possible directions for the proof.

1.5.5. [CHAPTER 6] EFFICIENT DISTILLATION OF DIFFUSION MODELS FOR ACCELERATED SAMPLING.

In this chapter, we address RQ5 by making a simplified copy of the teacher diffusion model's Markov chains rather than only mimicking the teacher's outputs. We propose the Single-fold Distillation (SFDDM) framework that flexibly compresses a teacher diffusion model into a student model with any desired step size, by reparameterizing the intermediate inputs from the teacher model. As shown in Figure 1.7, our basic idea is to align the Markov chains of the student and teacher models so that specific steps in their processes correspond to each other. By doing this, we ensure not only that their outputs match, but also that the conditional probabilities associated with the training samples are identical. This alignment allows the student model to mimic the teacher model's behavior while simplifying the structure of the teacher's Markov chain. The effective training for student diffusion is designed by minimizing the difference of output and hidden variables with respect to the teacher diffusion. We experimentally demonstrate that SFDDM succeeds in producing high-quality sampling data with a significantly reduced number of steps. This chapter is based on the following publication:

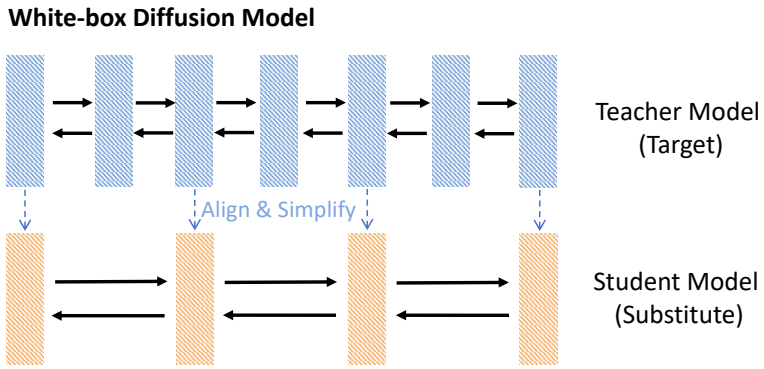


Figure 1.7: An overview of the Single-fold Distillation (SFDDM) framework proposed in Chapter 6 for simplifying teacher diffusion models by aligning the Markov chains of student and teacher models to address RQ5.

Chi Hong, Jiyue Huang, Robert Birke, Dick Epema, Stefanie Roos, and Lydia Y. Chen. "Single-fold Distillation for Diffusion models." In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2025.

The contributions of the authors are as follows: Chi Hong proposed the single-fold

distillation framework, derived the optimization goal, designed the network structure, implemented the code, conducted the experiments, and wrote the paper. Jiyue Huang contributed to designing flexible sub-sequence selection on the teacher's diffusion steps to make the alignment of the student and teacher Markov chains more adaptable, and assisted with figure preparation and writing of the experimental section.

2

DISTILLING KNOWLEDGE FROM NOISY LABELS VIA ONLINE AGGREGATION

Noisy labeled data is more a norm than a rarity for crowd sourced contents. It is effective to distill noise and infer correct labels through aggregation results from crowd workers. To ensure the time relevance and overcome slow responses of workers, online label aggregation is increasingly requested, calling for solutions that can incrementally infer true label distribution via subsets of data items. In this chapter, we propose a novel online label aggregation framework, BILA *, which employs variational Bayesian inference method and designs a novel stochastic optimization scheme for incremental training.* BILA *is flexible to accommodate any generating distribution of labels by the exact computation of its posterior distribution. We also derive the convergence bound of the proposed optimizer. We compare* BILA *with the state of the art based on minimax entropy, neural networks and expectation maximization algorithms, on synthetic and real-world data sets. Our evaluation results on various online scenarios show that* BILA *can effectively infer the true labels, with an error rate reduction of at least 10 to 1.5 percent points for synthetic and real-world datasets, respectively.*

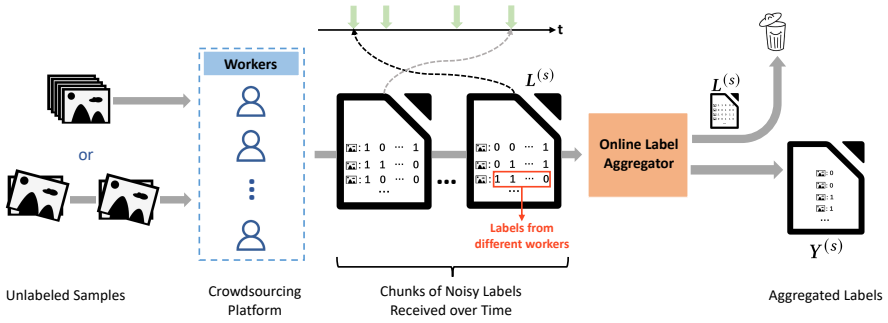


Figure 2.1: The online label aggregation scenario illustrates how raw data are labeled by non-professional workers and subsequently aggregated by aggregators.

2.1. INTRODUCTION

Crowd sourcing platforms provide economic and efficient means to curate datasets which are deemed the new oil for today's artificial intelligence [49]. One of commonly seen crowd tasks is to classify contents, e.g., web pages [107], and images [124], and to provide labels of their respective classes. However, due to differences in the crowd workers' background and experience, the resulting labels of the same content often vary across workers, including missing labels – so called noisy labels. The state of the practise [126, 130] to distill the quality of crowd sourced labels is to aggregate them across all workers and reach consensus for every content. Such a curated dataset can then conveniently power up a wide range of supervised ML models for further analysis, e.g., object detection, search engine [107], and disease diagnoses [136].

The velocity of knowledge discovery indeed hinges on the speed of data curation [19]. Faster the data is aggregated via crowd sourcing, the more insights can be extracted through ML models. For example [77], via instantaneous information from Amazon Mechanical Turk, i.e., in 2200 ms, the accuracy of predicting urban emergencies can be improved by 40%. Moreover, labelling massive datasets come as a daunting tasks requiring months or years of effort. The estimated effort to label ImageNet for a single person working 24/7 is 19 years, but even with crowd sourcing involving 25K workers it still took 21 months [18]. It becomes increasingly imperative that label curation and aggregation can be conducted in online manner, i.e., labels can be continuously aggregated over a subset of content, instead of the entire content at once. More, recent privacy and governmental policies [17] regulate the data storage time, asking for prompt action of aggregation.

The key challenge behind online label aggregation is how to utilize a partial label set from workers that only includes a small chunk of content. Existing aggregation methods [130, 12, 137] focus on the quality issues across workers but implicitly overlook the temporal aspect, i.e., timely and accurately label aggregation from online data. In other words, the prior art tailors for offline scenarios, which assumes the availability of all contents at once. As a result, in the online scenario, such approaches end up greedily

optimizing for only the available subset, without the global optimization for the entire dataset. The need of online label aggregation thus calls for a novel stochastic optimization scheme which can handle batches of observable data.

Probabilistic graphic models [61] are commonly adopted to aggregate noisy labels from crowd workers without the label ground truths. Their objective is to maximize likelihood of the observed data by capturing the dependency on latent variables, e.g., the true labels and confusion matrix that specifies the generation process of label noise. Variational Bayesian inference methods [121, 67, 115] can effectively infer the latent features by maximizing the evidence lower bound [4] of the log data likelihood of the observed data. The other popular approach to infer latent variables is Expectation-Maximization (EM) algorithm [12] that has different objectives in expectation and maximization steps - an additional hurdle for stochastic optimization. While variational inference methods have an advantage of single objective for stochastic optimization, the challenges lies in deriving a tracktable posterior distribution of the generation process of label noise.

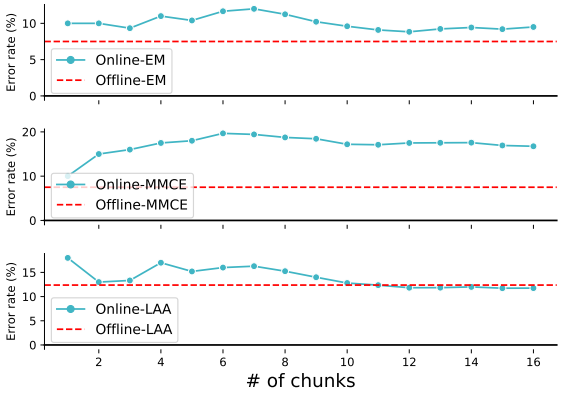
In this chapter, we propose a novel online label aggregation framework, BILA , based on incremental variational Bayesian Inference method. BILA aggregates noisy labels from crowd workers incrementally upon receiving a subset of labeled items via a novel stochastic optimization scheme. To maximize the log likelihood of observed items, BILA minimizes the Kullback-Leibler (K-L) divergence between (i) the noisy label generative distribution p, and (ii) the approximate distribution q. The unique features of BILA are (i) flexibility and extendibility for generative distribution, (ii) exact computation of posterior distribution bypassing the need of the closed from expression, and (iii) the proposed objective function has the exact expression of the expectation term of K-L divergence, avoiding the approximation variance. Using the framework of BILA , we define a label aggregation model for multiple classes, abbreviated as BILA-CM , based on confusion matrix. We employ multi-layer perceptron neural networks for approximate distribution q.

As the data chunks are received in an online fashion, BILA-CM is incrementally trained by data chunks. To such an end, we propose a stochastic optimization scheme - a variant of RMSProp [116]. It enhances RMSProp with a dynamic clip operator, bias-corrected second raw moment estimate and decaying learning rate.

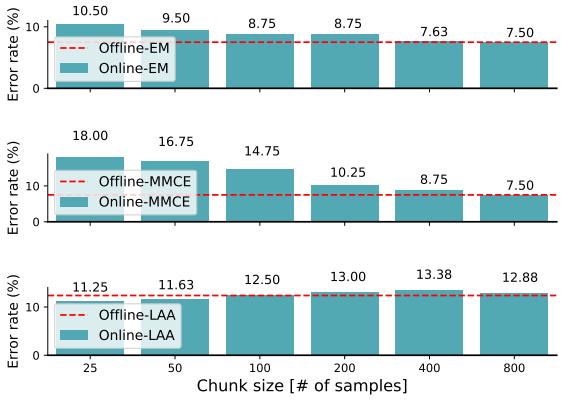
We evaluate BILA on both real-world and synthetic datasets. We compare its aggregation error rates with the state of the art label aggregation algorithms, i.e., Majority Voting, E-M based approaches, neural network based approaches and Minimax Entropy based approaches. BILA is able to achieve significant error reduction in various online scenarios, i.e., different data chunk sizes. Our results also show that BILA is robust against different crowd sourcing scenarios, i.e., different number of workers, noise ratios, and label sparsity. In terms of effectiveness of proposed optimization scheme, we are able to achieve faster convergence than RMSProp, and in par with ADAM [59] but without risks of divergence.

The contributions of this chapter are summarized as follows.

- We design a flexible online label aggregation framework, BILA , based on variational Bayesian inference framework (Section 2.3). BILA uses neural networks for the approximate distribution guided by the generating distribution.



(a) Error rate after aggregating each chunk (chunk size:50 samples)



(b) The effect of the chunk size

Figure 2.2: Motivation experiments to show the difference between online methods and offline methods on the RTE dataset.

- We provide a confusion matrix based aggregation model, BILA-CM , which outperforms existing algorithms based on EM algorithms, Minimax Entropy and neural networks (Section 2.5).
- We design a stochastic optimizer and derive its convergence bound (Section 2.4). Last, we extensively compare BILA against representative label aggregation methods on different online crowd sourcing scenarios (Section 2.5)

2.2. SYSTEM SCENARIOS

To overcome the data labelling challenges it is common practise to label datasets via crowd sourcing by non-experts. We can assign unlabeled instances to the workers in two ways: offline and online. In offline mode we publish all unlabeled instances once on a crowdsourcing platform and wait for all workers to complete their assignments before

training the label aggregation algorithm. This works well if the data does not change over time and is all available at once. In online mode, shown in Figure 2.1, we continuously publish single or multiple instances of unlabelled data on the crowdsourcing platform. Then we collect the labelling results and organize them in small chunks of redundant noisy labels. These chunks are fed one-by-one over time to the label aggregation algorithm to update the label aggregator. The processed redundant noisy labels are discarded and only the aggregated, i.e. inferred true, labels are kept. This enables continuous learning but requires the label aggregation algorithm to be be able to (incrementally) learn from small sets of data. This is challenging. Most state-of-the-art label aggregation techniques do not cope well with such a requirement.

We demonstrate this via a motivation example. We run three baseline aggregation algorithms in both online and offline mode and compare the achieved error rates. We consider Expectation-Maximization (EM) [12], Min-Max Conditional Entropy (MMCE) [137] and Label Aware Autoencoders (LAA) [130] on the RTE dataset (details given in §2.5.1). In online mode, we feed each aggregation algorithm with small chunks of 50 redundant noisy labels at a time. Each chunk is used to update the aggregator. We stop at 16 chunks (800 samples). At each step we evaluate the achieved error rate. After each update we use the aggregator to infer the aggregated label for each sample and compute the percentage of samples for which the aggregated label differs from the ground truth label. Note that label aggregation is an unsupervised learning task. The ground truth labels are used only to compute the error rate, not to train the aggregator. Figure 2.2a shows the stepwise error rate for the three methods. Instead, Figure 2.2b shows the sensibility of each method to the chunk size. Each plot reports the achieved error rate when processing 800 samples in chunks of different size. For reference we report the offline performance, i.e. processing all 800 samples at once, as a horizontal line.

EM is commonly used to estimate a confusion matrix for each worker. MMCE is designed to discern the confusion matrices across the workers as well as the instances. Both can not be readily adapted to learn incrementally from small sets of data. EM uses majority voting results to determine a good starting point for the parameter search. Hence the best starting point is different for each chunk. MMCE assigns model parameters to each sample. Since each chunk has different samples, we can not keep the learned parameters. We use this two methods in a sliding window style where each window is a new chunk of data. As a result, EM and MMCE maximize the data likelihood of the current chunk not the full data. Hence, the error rates of these two methods do not converge with time to the offline error rate, i.e. processing the whole data at once. More in detail, the performance of CE (see top plot Figure 2.2a) initially oscillates but then flattens out. After 16 chunks, i.e. all 800 samples, the error rate is still 2 percent points higher. MMCE is worse (see middel plot Figure 2.2a). The error rate first diverges before flattening out leaving a gap of 9.25 percent points after the last chunk. This is because MMCE is a generative model which needs to train a larger number of parameters compared to EM. This makes MMCE more sensible to the chunk size. This is clearly shown in Figure 2.2b. The performance of both EM and MMCE (top and middle plot) benefit from processing larger chunks sizes. With chunks of 25 samples, EM is 3 percent point worse than offline, but starting at chunk size 400 EM is able to equal the offline performance. Instead, MMCE is more sensible. At chunk size 25 the gap is 11.5 percent points. The gap dimin-

ishes with increasing chunk sizes, but it never reaches the same performance as offline. Note that chunk size 800 is equivalent to offline.

LAA is a neural network based method inspired from autoencoders. This method can be used incrementally so that its optimization goal maximizes the data likelihood of the full dataset, not only the chunk. Consequently, online LAA nicely converges to the offline results over time (bottom plot Figure 2.2a). The small difference between the two is due to the stochasticity of the training. For the same reason the sensibility of this algorithm against different chunk sizes is low (bottom plot Figure 2.2b). After processing 800 samples with different chunk sizes LAA achieves final error rates within ± 1 percent points of the offline performance. However this method leads generally to worse results. The best result achieved by LAA is 11.75% error rate compared to 7.5% for EM and MMCE. LAA does not have a probabilistic model to describe the generative process of the observed noisy labels. This harms its performance. Besides LAA needs approximate approaches to calculate the expectation terms in the loss function. Our proposed label aggregation model BILA-CM directly addresses these issues achieving superior performance in both offline and online mode.

2.3. ONLINE LABEL AGGREGATION

We consider the online learning scenario shown in Figure 2.1. Each data instance $\boldsymbol{l}_i = \{l_{i0}, ..., l_{iK}\}$ contains redundant noisy labels of sample i. These noisy labels are provided by K workers. $l_{ik} \in \boldsymbol{C}$ denotes the label of item i given by worker $k \in \{1, ..., K\}$, where $\boldsymbol{C} = \{1, ..., C\}$ is the set of the possible classes. Not every worker might label all samples. If sample i is not labeled by worker k, then the value of l_{ik} is -1. We use $y_i \in \boldsymbol{C}$ to represent the unknown true label of a sample. Data instances stream into the label aggregation model at different times in small sets $L^{(s)}$. We call the small sets as chunks. In our online learning setting, our task is to continuously infer the values of $\{y_i | i \in L^{(s)}\}$ for the current chunk $L^{(s)}$ in real time before receiving the next chunk.

In order to define our optimization goal, we need some additional notations. We use L to represent the collection of all observed noisy labels with N instances, i.e. $\boldsymbol{L} = \{\boldsymbol{l}_1, ..., \boldsymbol{l}_N\}$, and $\boldsymbol{Y} = \{y_1, ..., y_N\}$ for the collection of all the corresponding unknown true labels[1].

2.3.1. VARIATIONAL BAYESIAN INFERENCE FRAMEWORK (BILA)

In this section, we introduce our label aggregation framework (BILA) and define our optimization goal. The framework aims to predict the unknown true label y_i of each instance i with the sole knowledge of the instance's redundant noisy labels $\boldsymbol{l}_i$. The framework includes two components: a neural network q and a generative model p trained using an optimization goal defined based on the principle of variational inference. From the perspective of variational Bayesian inference, q is an approximate distribution. The choice of q and p is very flexible. q can be a multilayer perceptron (MLP), a convolutional neural network (CNN), or any other neural network. p is the model to define how to generate the observed noisy labels $\boldsymbol{L}$. Since we use a neural network as approximate distribution to learn the label aggregation model, we need stochastic optimization to

[1] To simplify notation we drop the subscript i when referring to a generic sample.

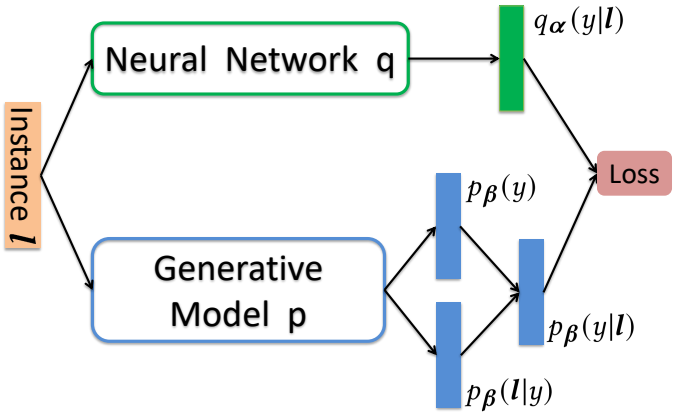


Figure 2.3: The relationship between the approximate distribution q, and the generative model p and the loss function.

train the parameters in q and p. This requires that the loss function is differentiable respect to the model parameters in p. This is the only constraint of the definition of p. Besides, the close form of the posterior of model p is needless because we rewrite the expression of the Kullback-Leibler divergence to avoid using the posterior directly. The relationship between q, p and the loss function in BILA is shown in Figure 2.3.

DEFINITION OF q AND p

The set of noisy labels $\boldsymbol{L} = \{\boldsymbol{l}_1, ..., \boldsymbol{l}_N\}$ only contains the observed instances $\boldsymbol{l}_i$. The corresponding true labels Y are unknown. The label aggregation task in this chapter is to predict the unknown true labels given $\boldsymbol{L}$. So it is an unsupervised learning task.

Given an instance $\boldsymbol{l}_i$ of noisy redundant labels, we use a neural network q with softmax activation on the last layer to predict the corresponding unknown true label y_i. q can be represented as a probability distribution $q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})$, where $\boldsymbol{\alpha}$ denotes the neural network parameters. The output of the network is a C-dimensional vector $[q_{\boldsymbol{\alpha}}(y = c|\boldsymbol{l})]_{c=1}^{C}$, where the c-th element $q_{\boldsymbol{\alpha}}(y = c|\boldsymbol{l})$ is the probability that the true label of the input instance is class c. The predicted label is given by the element with the highest probability.

In order to train q we need an optimization goal. Therefore, we define a generative model p to describe the generative process behind the observed noisy labels $\boldsymbol{L}$. This way we can define a loss function to guide the training according to variational inference rules [4, 121]. p assumes that an instance $\boldsymbol{l}$ is generated from some conditional distributions $p_{\boldsymbol{\beta}}(\boldsymbol{l}|y)$, where y denotes the unknown true label and $\boldsymbol{\beta}$ the parameters of model p. It further assumes that y is generated from a prior distribution $p_{\boldsymbol{\beta}}(y)$. The generative model p potentially defines a posterior distribution:

$$p_{\boldsymbol{\beta}}(y|\boldsymbol{l}) = \frac{p_{\boldsymbol{\beta}}(\boldsymbol{l}|y)p_{\boldsymbol{\beta}}(y)}{p_{\boldsymbol{\beta}}(\boldsymbol{l})}. \tag{2.1}$$

We do not make many simplifying assumptions about p except that the loss function is differentiable with respect to $\boldsymbol{\beta}$.

OPTIMIZATION GOAL

To solve the unsupervised learning task where we only have the observed noisy labels $\boldsymbol{L}$ a reasonable optimization goal is to maximize the data likelihood of $\boldsymbol{L}$. In particular, we maximize the data log likelihood $\log p_{\boldsymbol{\beta}}(\boldsymbol{L})$ according to the evidence lower bound $\log p_{\boldsymbol{\beta}}(\boldsymbol{L}) = KL(q_{\boldsymbol{\alpha}}(\boldsymbol{Y}|\boldsymbol{L})||p_{\boldsymbol{\beta}}(\boldsymbol{Y}|\boldsymbol{L})) + L(q) \geq L(q)$, where $KL(\cdot)$ denotes the Kullback-Leibler divergence and $L(q) = \mathbb{E}_{q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)}\left[\log(p_{\boldsymbol{\beta}}(\boldsymbol{L},\boldsymbol{Y})/q_{\boldsymbol{\beta}}(\boldsymbol{Y}))\right]$. We can see that we can maximize the lower bound of $\log p_{\boldsymbol{\beta}}(\boldsymbol{L})$ as we minimize $KL(q_{\boldsymbol{\alpha}}(\boldsymbol{Y}|\boldsymbol{L})||p_{\boldsymbol{\beta}}(\boldsymbol{Y}|\boldsymbol{L}))$. So, we need to find the consensus between the predictions of q and p. Thus, we use $KL(q_{\boldsymbol{\alpha}}(\boldsymbol{Y}|\boldsymbol{L})||p_{\boldsymbol{\beta}}(\boldsymbol{Y}|\boldsymbol{L}))$ as our loss function and minimize it during the training process.

We assume that each collected label is generated independently, i.e. instances in $\boldsymbol{L}$ are independent from each other. Plugging $q_{\boldsymbol{\alpha}}(\boldsymbol{Y}|\boldsymbol{L}) = \prod_i q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)$ and $p_{\boldsymbol{\beta}}(\boldsymbol{Y}|\boldsymbol{L}) = \prod_i p_{\boldsymbol{\beta}}(y_i|\boldsymbol{l}_i)$ into the loss function, we have:

$$\begin{aligned} KL(q_{\boldsymbol{\alpha}}(\boldsymbol{Y}|\boldsymbol{L})||p_{\boldsymbol{\beta}}(\boldsymbol{Y}|\boldsymbol{L})) &= \sum_{i=1}^{N} -\mathbb{E}_{q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)}\left[\log \frac{p_{\boldsymbol{\beta}}(y_i|\boldsymbol{l}_i)}{q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)}\right] \\ &= \sum_{i=1}^{N} KL(q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)||p_{\boldsymbol{\beta}}(y_i|\boldsymbol{l}_i)) \end{aligned} \tag{2.2}$$

Equation (2.2) cannot be directly used to train q and p, because the expression of $p_{\boldsymbol{\beta}}(y|\boldsymbol{l})$ is unknown. The exact expression of the posterior $p_{\boldsymbol{\beta}}(y|\boldsymbol{l})$ may be intractable. So we have to further rewrite the loss function. According to (2.1), we have

$$\begin{aligned} &KL(q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})||p_{\boldsymbol{\beta}}(y|\boldsymbol{l})) = -\mathbb{E}_{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})}\left[\log \frac{p_{\boldsymbol{\beta}}(y|\boldsymbol{l})}{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})}\right] \\ &= -\mathbb{E}_{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})}\left[\log \frac{p_{\boldsymbol{\beta}}(y)}{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})} + \log p_{\boldsymbol{\beta}}(\boldsymbol{l}|y)\right] + const \\ &= KL(q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})||p_{\boldsymbol{\beta}}(y)) - \mathbb{E}_{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})}\left[\log p_{\boldsymbol{\beta}}(\boldsymbol{l}|y)\right] + const \end{aligned} \tag{2.3}$$

To simplify the notations, we use $\theta = \{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$ to represent the parameters of both models in our framework. According to (2.2) and (2.3) the loss function is rewritten as

$$f(\theta;\boldsymbol{L}) = \frac{1}{N}\sum_{i=1}^{N}\{KL(q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)||p_{\boldsymbol{\beta}}(y_i)) - \mathbb{E}_{q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)}\left[\log p_{\boldsymbol{\beta}}(\boldsymbol{l}_i|y_i)\right]\} \tag{2.4}$$

where we ignore the constant term and the loss function is rescaled by $1/N$. This does not affect the optimization result.

TRAINING

During training we solve the following optimization problem

$$\hat{\theta} = \arg\min_{\theta} f(\theta;\boldsymbol{L}).$$

This optimization problem is solved by stochastic first-order optimization. The update rule of the model parameters is shown in Algorithm 2.1. Following our discussions in

§ 2.2, in order to continuously aggregate small sets of noisy labels, we apply mini-batch training to update the parameters θ. So the loss function for training is

$$f(\theta; \boldsymbol{L}^{(M)}) = \frac{1}{M} \sum_{i=1}^{M} \left\{ KL(q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)||p_{\boldsymbol{\beta}}(y_i)) - \mathbb{E}_{q_{\boldsymbol{\alpha}}(y_i|\boldsymbol{l}_i)} \left[\log p_{\boldsymbol{\beta}}(\boldsymbol{l}_i|y_i) \right] \right\}, \tag{2.5}$$

where $\boldsymbol{L}^{(M)}$ is a mini-batch sampled from current dataset $\boldsymbol{L}^{(s)}$, and M denotes the mini-batch size. The gradient of $f(\theta; \boldsymbol{L}^{(M)})$ is required to update the model parameters. Before calculating the gradient, we need to define the expression of the KL divergence and the expectation term. The unobserved variables y_i are discrete variables that take values from 1 to C. Therefore, we have the following expressions

$$KL(q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})||p_{\boldsymbol{\beta}}(y)) = -\sum_{c=1}^{C} q_{\boldsymbol{\alpha}}(c|\boldsymbol{l}) \log \frac{p_{\boldsymbol{\beta}}(c)}{q_{\boldsymbol{\alpha}}(c|\boldsymbol{l})}, \tag{2.6}$$

$$\mathbb{E}_{q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})} \left[\log p_{\boldsymbol{\beta}}(\boldsymbol{l}|y) \right] = \sum_{c=1}^{C} q_{\boldsymbol{\alpha}}(c|\boldsymbol{l}) \log p_{\boldsymbol{\beta}}(\boldsymbol{l}|c), \tag{2.7}$$

where $q_{\boldsymbol{\alpha}}(c|\boldsymbol{l})$ is the c-th element of the neural network output. Note that we do not require any approximation for calculating the expectation terms in our loss function. As such we avoid the problem of high variance of the loss function in stochastic Bayesian inference [91]. According to (2.6) and (2.7), the values of $f(\theta; \boldsymbol{L}^{(M)})$ and corresponding stochastic gradient $\nabla_\theta f(\theta; \boldsymbol{L}^{(M)})$ can be easily computed. This completes all necessary blocks in the framework to construct online label aggregation models.

2.3.2. LABEL AGGREGATION MODEL

In this subsection, we introduce our online label aggregation model, BILA-CM , based on the BILA framework. This model can be applied to aggregate discrete labels with noise.

MODEL DEFINITION

In order to define BILA-CM and exact loss function f, we need to decide the concrete forms of q and p. We set q to be a fully connected neural network with a softmax activation function on the last layer. q takes an instance $\boldsymbol{l}$ as input and outputs a distribution $q_{\boldsymbol{\alpha}}(y|\boldsymbol{l})$, where $\boldsymbol{\alpha}$ denotes the neural network parameters.

p is a generative model describing the observed noisy labels $\boldsymbol{l}$. From (2.6) and (2.7), in order to compute the loss function and its gradient, we need to define the expressions of $p_{\boldsymbol{\beta}}(\boldsymbol{l}|y)$ and $p_{\boldsymbol{\beta}}(y)$. Since every element in an instance is collected independently from different workers, we assume that the k-th element in an instance is generated from an independent distribution $\boldsymbol{\psi}_{ck}$ when the true label of the instance is c. This distribution is defined as

$$\boldsymbol{\psi}_{ck} = softmax(\boldsymbol{\omega}_{ck}), \tag{2.8}$$

where $\boldsymbol{\omega}_{ck}$ is a C-dimensional vector. Then $p_{\boldsymbol{\beta}}(\boldsymbol{l}|y = c)$ can be defined as

$$p_{\boldsymbol{\beta}}(\boldsymbol{l}_i|y_i = c) = \prod_{k \in \boldsymbol{S}_i} \psi_{ck,l_{ik}}, c \in [C], \tag{2.9}$$

where $\psi_{ck,l_{ik}}$ is the l_{ik}-th element of $\boldsymbol{\psi}_{ck}$. Since the softmax function is derivable, $\boldsymbol{\omega}_{ck}$ can be updated by stochastic optimization. In this model, the prior distribution $p_{\boldsymbol{\beta}}(y)$ is a multinomial distribution estimated by

$$\hat{p}_{\boldsymbol{\beta}}(y=c) = \frac{\sum_i \sum_k \mathbb{I}(l_{ik}=c)}{\sum_i \sum_k \mathbb{I}(l_{ik}\neq -1)}, c \in [C], \tag{2.10}$$

where the values of the estimators can be calculated by counting the observed labels. Since $p_{\boldsymbol{\beta}}(y)$ is fixed, we introduce a hyperparameter ζ to constrain the Kullback-Leibler divergence term in the loss function (2.5). We regard this constrained term as a regularizer. Then, using (2.6) and (2.7) the mini-batch loss function used is

$$f(\theta; \boldsymbol{L}^{(M)}) = -\frac{1}{M}\sum_{i=1}^{M}\Big\{\zeta \sum_{c=1}^{C} q_{\boldsymbol{\alpha}}(c|\boldsymbol{l}_i)\log\frac{p_{\boldsymbol{\beta}}(c)}{q_{\boldsymbol{\alpha}}(c|\boldsymbol{l}_i)} + \sum_{c=1}^{C} q_{\boldsymbol{\alpha}}(c|\boldsymbol{l}_i)\log p_{\boldsymbol{\beta}}(\boldsymbol{l}_i|c)\Big\}, \tag{2.11}$$

ONLINE MODEL TRAINING

The details of the complete online label aggregation model BILA-CM are illustrated in Algorithm 2.1. BILA-CM continuously receives a new noisy labels chunk $\boldsymbol{L}^{(s)}$ containing multiple redundant noisy label instances $\boldsymbol{l}$. Note that we do not require the size of each set to be equal. This increases the practicality of our algorithm. At the beginning, we accumulate few noisy label sets to construct an initial set $\boldsymbol{L}^*$. This initial set is used to initialize the model parameters $\boldsymbol{\beta} = \{\boldsymbol{\omega}_{ck}\}$ and the prior estimator $\hat{p}_{\boldsymbol{\beta}}(y)$. $\boldsymbol{\beta}$ can be initialized by its definition and majority voting on the noisy redundant labels from the initial set to predict the true labels. After initialization, we start the online aggregation. For each arriving chunk $\boldsymbol{L}^{(s)}$ at time step t, we update the model parameter θ. Then we aggregate each $\boldsymbol{l} \in \boldsymbol{L}^{(s)}$ using the updated θ. The BILA-CM update and aggregation process is illustrated in function `UpdateAndAggregate` (lines 10-30). First we retrain the model by computing the terms of the loss function f from Equation (2.11) on each sampled mini batch (lines 14-21) before updating the model (lines 22-27).

INFERRING THE AGGREGATED LABELS

Before introducing how to infer the aggregated label y, i.e. the predicted true label, for each sample $\boldsymbol{l}$, we discuss the connection between the generative model p of BILA-CM and the confusion matrices of the workers. The confusion matrix $\boldsymbol{\pi}_{c,z}^{(k)}$ of worker k is a matrix for describing the worker's labeling behavior [12]. The matrix element $\pi_{c,z}^{(k)} = p(l_{ik}=z|y_i=c)$ is the probability that worker k assigns the label z to the instance i when the true label y_i is c. According to the definition of p, we have that $\psi_{ck,z} = p_{\boldsymbol{\beta}}(l_{ik}=z|y_i=c)$ which corresponds to $\pi_{c,z}^{(k)}$. Therefore, we can easily construct the confusion matrices of the workers after learning the parameters $\boldsymbol{\beta} = \{\boldsymbol{\omega_{ck}}\}$. Note that this provides insight on the noise process which other methods lack, e.g. LAA. With the confusion matrices the inference problem becomes trivial. After obtaining the values of the confusion matrices, we can infer the aggregated label of an instance by maximizing the data likelihood of the corresponding observed noisy labels, where $p(\boldsymbol{l}_i|y_i=c,\boldsymbol{\pi}) = \prod_{k=1}^{K}\prod_{z=1}^{C}(\pi_{c,z}^{(k)})^{\mathbb{I}(L_{i,k}=z)}$. $\mathbb{I}(\cdot)$ is an indicator function taking the value 1 when the predicate is true, and 0 otherwise,

Algorithm 2.1: Online Label Aggregation Model BILA-CM . The model parameters are $\theta = \{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$, where $\boldsymbol{\alpha} = \{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2\}$ and $\boldsymbol{\beta} = \{\boldsymbol{\omega}_{ck}\}$.

Set: learning rate $\mu > 0$, exponential decay rate $\gamma \in [0, 1)$, time step $t = 1$
Input: Continuously receive new noisy labels set $\boldsymbol{L}^{(s)} = \{\boldsymbol{l}\}$
Accumulate few sets to construct the initial set $\boldsymbol{L}^*$
Initialize θ using $\boldsymbol{L}^*$
$\boldsymbol{Y}^*$ = UpdateAndAggregate($\boldsymbol{L}^*$)
Output: The aggregated labels $\boldsymbol{Y}^*$
for *each arriving set* $\boldsymbol{L}^{(s)}$ **do**
    $\boldsymbol{Y}^{(s)}$ = UpdateAndAggregate($\boldsymbol{L}^{(s)}$)
    Output: The aggregated labels $\boldsymbol{Y}^{(s)}$
Function `UpdateAndAggregate(`$\boldsymbol{L}^{(s)}$`):`
    for *number of training epochs* **do**
        for *number of minibatchs* **do**
            Sample a batch $\boldsymbol{L}^{(M)} = \{\boldsymbol{l}_1, ..., \boldsymbol{l}_M\}$ from $\boldsymbol{L}^{(s)}$
            `/* Calculate each term in` f`, Eq(2.11) */`
            for $c = 1, ..., C$ **do**
                for $k = 1, ..., K$ **do**
                    $\boldsymbol{\psi}_{ck} = softmax(\boldsymbol{\omega}_{ck})$
            for *instance* $i = 1, ..., M$ **do**
                $\boldsymbol{h} = \boldsymbol{W}_2 \, tanh(\boldsymbol{W}_1 \boldsymbol{l}_i + \boldsymbol{b}_1) + \boldsymbol{b}_2$
                $[q_{\boldsymbol{\alpha}}(y = c|\boldsymbol{l}_i)]_{c=1}^{C} = softmax(\boldsymbol{h})$
                for $c = 1, ..., C$ **do**
                    $\log g_{\boldsymbol{\beta}}(\boldsymbol{l}_i | y = c) = \sum_{k \in \boldsymbol{s}_i} \log \psi_{ck, l_{ik}}$
            `/* update the model parameters` θ `*/`
            $g_t \leftarrow \nabla_\theta f_t(\theta_t)$
            $v_t \leftarrow \gamma \cdot v_{t-1} + (1 - \gamma) \cdot (g_t \odot g_t)$
            $\mu_t = \mu \cdot \sqrt{1 - \gamma^t}$
            $\eta_t = Clip(\mu_t / \sqrt{v_t}, \eta_l(t), \eta_u(t)) / \sqrt{t}$
            $\theta_{t+1} \leftarrow \theta_t - \eta_t \odot g_t$
            $t \leftarrow t + 1$ `// count the time step`
    Get new confusion matrices $\boldsymbol{\pi}$ by the updated $\boldsymbol{\beta}$
    Infer the aggregated labels $\boldsymbol{Y}^{(s)}$ by $\boldsymbol{\pi}$
    return $\boldsymbol{Y}^{(s)}$

2.4. OPTIMIZER AND CONVERGENCE ANALYSIS

We propose a stochastic optimizer to train BILA-CM and summarize key steps in line 22-27 of Algorithm 2.1. It's a variant of RMSProp [116]. The update of the model parameter θ (line 26) is based on gradient rather than the momentum. Similar to RMSProp, we utilize a second raw moment estimate of the gradient (line 23) to obtain the element-wise adaptive learning rates for every element of θ (line 24-25). The element-wise adaptive learning rates are important because of the observation that in a multilayer neural network, the appropriate learning rates can vary widely between weights [116]. Furthermore, we apply a clip operator applied to avoid gradient explosion (line 25). In order to avoid an abrupt stop in training, we employ decayed learning rate approach. The upper and lower bound of the clip operator is then divided by $\sqrt{t}$ to obtain decayed element-wise learning rates.

Furthermore, we also analyze the convergence property of our stochastic optimization approach in the online convex framework [141]. According to the framework setting, we use an unknown sequence of convex loss functions $f_1(\theta), f_2(\theta), ..., f_T(\theta)$ to represent the loss functions at each iteration time step t. In each time step the training data (mini-batches) are different, so we need different notations to represent the stochasticity of the loss functions. The regret is applied to evaluate the convergence of our label aggregation algorithm. The regret is defined as $R(T) = \sum_{t=1}^{T}[f_t(\theta_t) - f_t(\theta^*)]$ where $\theta^* = \arg\min_{\theta\in\chi}\sum_{t=1}^{T} f_t(\theta)$. Actually $R(T)$ represents the sum of the difference between the online prediction θ_t and the best fixed parameter θ^*. We will show that our algorithm has a $O(\sqrt{T})$ regret bound. To show the bound, we define the notation $\eta_{t,i}$ to be the i^{th} element of η_t.

The following Theorem 2.1 is derived under the online convex framework [141] which is a well known setting to do convergence analysis on a stochastic optimization approach. For example, this setting has been applied on follow-the-regularized-leader (FTRL) algorithms [82] and AdaBound [102]. As the primary contribution of this chapter lies in the online learning framework, the proof of the optimization method used in our experiments is included in the appendix of the corresponding technical paper [41]. The proof follows the ideas of deriving a bound of an optimizer with a clip operator introduced by AdaBound [102].

Theorem 2.1 Let $\{\theta_t\}$ be the parameter sequence obtained from our optimizer where $\theta \in R^d$. Suppose $\eta_u(t) \le R_\infty$ and $\frac{t}{\eta_l(t)} - \frac{t-1}{\eta_u(t-1)} \le B$ for all $t \in [T]$. Assume that $\|\theta_n - \theta_m\|_\infty \le D_\infty$ for all $\theta_n, \theta_m \in \chi$ and $\left\|\nabla f_t(\theta)\right\|_2 \le G$ for all $t \in [T]$ and $\theta \in \chi$. Then our optimizer have the following guarantee of the regret

$$R(T) \le \frac{1}{2}D_\infty^2\left[2dB(\sqrt{T}-1) + \sum_{i=1}^{d}\eta_{1,i}^{-1}\right] + (\sqrt{T}-\frac{1}{2})R_\infty G^2.$$

According to Theorem 2.1, if we choose the lower and the upper bounds of the clip operator which satisfy $\eta_u(t) \le R_\infty$ and $\frac{t}{\eta_l(t)} - \frac{t-1}{\eta_u(t-1)} \le B$ for all $t \in [T]$, the regret bound will be $O(\sqrt{T})$. The lower bound and upper bound with constant values definitely satisfy these conditions. Dynamic clip bounds as shown in the experimental part of the paper [102], also meet the requirement, and they have good performance in practice. We

Table 2.1: Datasets overview to show the characteristics of six datasets.

Dataset	Workers	Items	Labels	Classes
Adult	17	263	1370	4
RTE	164	800	8000	4
Heart	12	237	952	2
Age	165	1002	10020	7
CIFAR10^S	10	50K	45K	10
Pendigits^S	10	11K	9.9K	10

^{S} uses synthetic redundant noisy labels.

choose these bounds to conduct our experiments. Note that because of the clip operator, $\sum_{i=1}^{d} \eta_{1,i}^{-1}$ takes a limited value. Then we have the corresponding convergence rate $R(T)/T = O(1/\sqrt{T})$ where $\lim_{T\to\infty} R(T)/T = 0$. That shows the average of the difference between the online prediction and the best fixed parameter tend to 0 during the iterations. Thus, the regret bound guarantees the convergence of our algorithm.

2.5. EVALUATION

2.5.1. EXPERIMENTAL SETUP

DATASETS

We consider six different datasets in our experiments to evaluate the performance of BILA-CM comparing to competitors. In our experiments, CIFAR-10 and Pendigits are the only synthetic dataset, while the rest of them are real-world datasets.

- **Adult** [95]: It contains data labeled by Amazon Mechanical Turk workers. The labels are categorized into four classes based on the amount of adult content on each web page.
- **RTE** [107]: It includes 164 workers for assigning labels of 800 items into 2 classes of textual entailment.
- **Heart** [136]: It is a dataset provided by 12 medical students categorizing the patients into 2 groups of heart and non-heart diseases based on physical examination. It has 12 workers for 237 samples.
- **Age** [31]: This dataset is the 1001 faces of different people who have been labeled with their age. In our experiments, the labels are discretized into 7 age groups: [0,9], [10,19], [20,29], [30,39], [40,49], [50,59], [60,100].
- **CIFAR-10** [65]: It is a vision dataset including 50K 32×32-pixels training images classified into 10 classes. Here we create synthetic noisy labels. For each image we generate $[6, 8, 10]$ redundant labels, i.e. workers, drawn from a bimodal noise distribution with $[0.4, 0.6, 0.8]$ mislabelling probability and $[0.1, 0.2, 0.3]$ missing label probability. We center the bimodal distribution around classes $\mu_1 = 3.0$, $\mu_2 = 7.0$ with variance $\sigma_1 = 1.0$, $\sigma_2 = 0.5$.

Table 2.2: Online label aggregation: error-rates (%). The online version of baseline algorithms are appended with prefix of "o".

Dataset	Small chunk size					Big chunk size				
	MV	**oEM**	**oMMCE**	**oLAA**	**BILA-CM**	**MV**	**oEM**	**oMMCE**	**oLAA**	**BILA-CM**
CIFAR10 (200, 500)	24.74	22.58	43.00	29.20	**13.29**	24.74	17.68	14.25	30.29	**13.69**
Pendigits (200, 500)	25.27	23.11	44.77	28.84	**13.34**	25.27	18.15	14.77	30.54	**13.06**
Age (25, 50)	34.73	35.03	41.52	35.53	**33.73**	34.73	34.43	41.92	36.33	**33.63**
RTE (25, 50)	9.88	10.5	18.0	11.25	**7.75**	9.88	9.5	16.75	11.75	**7.5**

- **Pendigits** [16]: This dataset targets the recognition of 10992 handwritten digits from 44 writers. We create synthetic redundant noisy labels using the same noise model as for CIFAR10.

Table 2.1 summarizes the characteristics of all datasets.

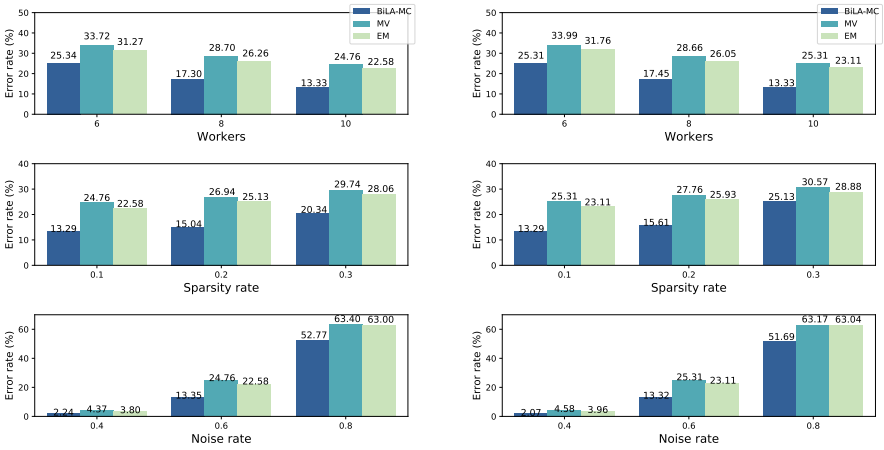
BASELINES

We consider five different baselines to compare BILA-CM against. The baselines cover both state of the art as well as state of the practise. All algorithms are programmed in Python programming language using Keras version version 2.2.4 and TensorFlow version 1.12.

- **Majority Voting (MV)**: is a basic method which selects from the set of redundant noisy labels $\boldsymbol{l}$ the label with the highest consensus.
- **Expectation Maximization (EM)** [12]: is an iterative method used to estimate each worker's confusion matrix by maximizing the likelihood of observed labels. The off-diagonal elements represent the probability of mislabeling, the diagonal elements of correct labeling.
- **Bayesian Classifier Combination (BCC)** [58]: is an extension of EM. It solves the label aggregation problem by modelling the relationship between the output of multiple classifiers (workers) and the true label.
- **Minimax Entropy (ME)** [138]: assigns a confusion matrix to workers, encoding their labeling ability, and a vector to items, encoding their labeling difficulty. The matrix and vector are estimated jointly using a minimax entropy approach.
- **MiniMax Conditional Entropy (MMCE)** [137]: extends ME by assigning confusion matrices also to items intead of a vector. It uses a minimax conditional entropy approach to jointly estimate both worker and item matrices.
- **Label Aware Autoencoders (LAA)** [130]: represents the labelling problem via an autoencoder model where the encoder acts as classifier inferring the true label, the decoder reconstructs the input and the inferred labels represent the latent space.

BILA-CM PARAMETERS

As neural network q in BILA-CM we use a multi layer perceptron with two hidden layers of size 64 and 32, respectively. The sizes of the input and output layers are given by the number of workers and the number of classes of each dataset, respectively. We train the network until convergence using marginal loss as stopping criteria.



(a) CIFAR-10: chunk size 200, initial set size 1000

(b) Pendigits: chunk size 200, initial set size 1000

Figure 2.4: The effect of the number of workers, label sparsity and noise ratio on BILA-CM . The default values for the number of workers, sparsity ratio, and noise ratio are 10, 0.1, and 0.6, respectively.

PERFORMANCE MEASURE

We use error rate as performance metric in all our experiments. We define the error rate as the percentage of inferred labels which differ from the true label. Note that the true label is only used to compute the error rate but not to train the label aggregators.

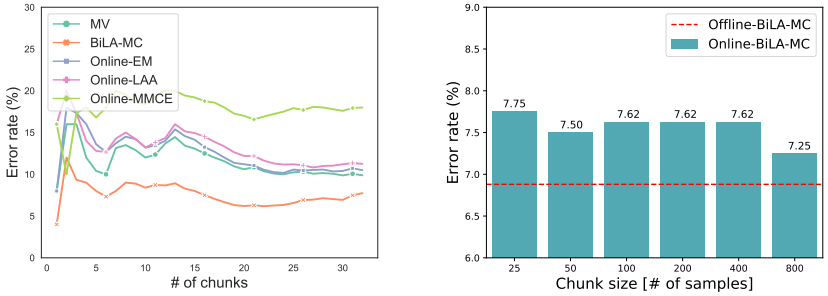
2.5.2. RESULTS

We first provide comparative results for online label aggregation, data processed in chunks, showing the superior performance of BILA-CM and its robustness to varying chunk sixes. Following we perform a sensitivity analysis of BILA-CM on dataset parameters, i.e. number of workers, noise rate and label sparsity, and analyze the optimality of BILA . Finally we conclude with results on offline label aggregation, data processed all at once.

ONLINE LABEL AGGREGATION

We summarize error rates of BILA and different baselines, across different combinations of datasets and chunk sizes in Table 2.2. The small and big chunk size are 200/25 and 500/50 for synthetic/real-world data respectively. Their initial datasets are 1000 and 500 samples for CIFAR-10/Pendigits and Age/RTE, respectively. We initialize all model based approaches by majority voting. Due to the limited number of samples in Adult and Heart dataset, we opt them out from the online evaluation.

When the chunk size is small, labels from workers are received more fluidly. One can see that BILA-CM achieved the lowest error rate. For CIFAR10 and Pendigits, the error rate of BILA-CM is 10 percent points lower than the second best algorithm, i.e., online EM. For the Age and RTE datasets, majority voting is the second best algorithm but still has at least 1.5 percent points higher error rates. As MMCE and LAA both have larger number of parameters than EM, their error rates are remarkably high due to insufficient number of samples per chunk for parameterization, especially for MMCE. We further



(a) RTE: small chunk size.

(b) BILA-CM : effect of chunk size.

Figure 2.5: Online label aggregation on the dataset RTE: error rates on BILA-CM and the baselines.

note that smaller chunks not only affect the error rate for EM and MMCE, but also the convergence speed. Due to the small number of samples, it takes MMCE more iterations to converge, compared to big chunk size.

When the chunk size is larger, more items can be aggregated at once, i.e., closer to the offline scenario. BILA-CM is still the best algorithm and MMCE comes second, except for Age. The difference from the small chunk size is that now there are sufficient number of samples in a chunk to parametrize the MMCE model. MMCE captures the confusion matrix at the levels of classes, workers and data items. Regarding EM, the error rate drops significantly for bigger chunk size.

Another observation worth mentioning is the comparison of computational overhead. Due to its simplicity, MV incurs almost no computational overhead. EM algorithm is known to have fast convergence. This is the case observed here. As LAA and BILA-CM both employ neural networks, their computational overheads are in the same order.

We further zoom into the error rates over the online aggregation process of the RTE dataset for small chunks (see Figure 2.5a). When the number of aggregated samples increases, the error rate first increases and then drops because of the large difference between the size of initial set (i.e., 500 samples) and chunk size (i.e., 25 samples). Overall, BILA-CM is able to learn the confusion matrix efficiently from only small chunks of samples and incrementally update the inference model. This is supported by visibly lower error rates across any number of samples processed by the aggregator.

We also demonstrate the robustness of BILA-CM against different chunk sizes or online velocity in Figure 2.5b. Recalling the motivation examples in Figure 2.2, existing label aggregation methods are sensitive to the online velocity, i.e., drastic error rate changes between very big and small chunk sizes. Thanks to incremental updates and stochastic optimization, BILA-CM can keep relatively low and constant error rates when encountering different online velocities.

SENSITIVITY (ROBUSTNESS) ANALYSIS OF BILA

We focus on evaluating the robustness of BILA-CM via synthetic redundant noisy labels on two datasets, i.e., CIFAR-10 and Pendigits. Specifically, we evaluate how BILA-CM performs against different types of crowd sourcing scenarios, i.e., number of crowd

Table 2.3: Offline label aggregation experiments on four datasets, Adult, RTE, Heart, and Age: error-rates (%) of label aggregation models

Dataset	MV	EM [12]	BCC [58]	ME[138]	MMCE [137]	LAA [130]	BILA-CM
Adult	26.43	25.48	22.81	24.33	24.33	25.86	**21.60**
RTE	9.88	7.50	7.15	7.25	7.50	12.38	**6.88**
Heart	22.36	18.99	18.82	16.03	16.03	13.50	**12.66**
Age	34.73	35.03	33.53	32.63	32.63	34.13	**30.18**

workers, sparsity of labels, and noise rates. The sparsity of labels defines the percentage of missing labels across all items and workers. The noise rate indicates the percentage of wrong labels of all labels collected.

Figure 2.4 summarizes such a sensitivity analysis for CIFAR-10 and Pendigits, respectively. We vary one parameter and fix the other two. The default values are 10 workers, sparsity rate of 0.1 and noise rate of 0.6. For the purpose of comparison, we choose the best performing label aggregation methods, i.e., EM and majority voting.

Across all three methods, we can make the following general observations. The error rates decrease with increasing number of workers, and increase with the sparsity and noise rate. In all cases considered, BILA-CM always achieves the lowest error rate, followed by EM and then MV.

Taking a closer look of number of workers, we observe that BILA-CM is able to achieve similar error rates, i.e., 25.34%, as MV but using only 6 instead of 10 workers. As for the robustness against the sparsity rate, EM and MV can better cope with increasing missing labels than BILA-CM . Specifically, when the sparsity increases from 0.1 to 0.3, the error rate of BILA-CM almost doubles, whereas the error rate of EM only increases by less than 30%. It appears that BILA-CM can be more sensitive to the sparsity than other methods, but the absolute performance is still better.

Regarding the impact of noise rate, all methods deteriorate drastically when the noise rate is up to 0.8. Actually none of the methods can reach accuracy above 50%. This is a bottleneck of how label aggregation methods can combat crowd's mistakes. To overcome high percentage of label noise, different solutions may be needed, e.g., a small fraction of ground-truth. When noise ratio is 0.4 and 0.6, we can observe that BILA-CM can achieve half of the error rate of the other two methods.

OPTIMALITY OF BILA

Next we investigate the optimality of the optimizer used in BILA-CM . We compare our optimizer against RMSProp [116] and ADAM [59]. Our optimizer is a enhanced version of RMSProp, while ADAM is another common choice. Figure 2.6 shows the evolution of the error rate across training epochs. We use the Age dataset with chunk size 25. We see that our optimizer is faster to converge than RMSProp. Hence we can obtain a higher model performance for the same training effort, i.e. number of epochs. ADAM is initially slightly faster to converge, but starting at epoch 120 the two optimizers achieve similar model performance. RMSProp in our implementation and our optimizer both use a clip operator to avoid the issue of gradient explosion which can lead to divergence. ADAM instead is a momentum-based optimizer. Momentum optimizers can be faster to con-

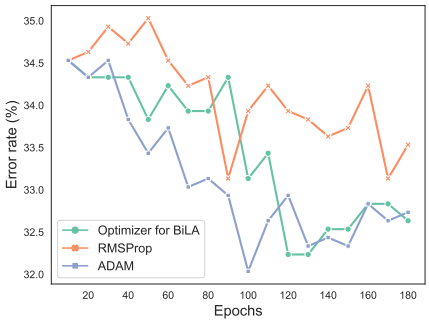


Figure 2.6: The error rates versus the number of epochs for different optimizers on the dataset Age.

verge than clip-based optimizers but pose the risk of gradient explosion and divergence. These observations are clearly shown in the results.

OFFLINE LABEL AGGREGATION

Finally, we present offline aggregation results in Table 2.3. We include two additional baselines, i.e., BCC [58] and ME [138]. BCC combines the probabilistic models and confusion matrix to infer true labels. ME is the predecessor of MMCE [137]. Both jointly estimate worker and item latent variables. Similar to the online results, BILA-CM is able to achieve the lowest error rate in all four datasets. The second best policy depends on the dataset. For Adult and RTE, the second best method is BCC that combines probabilistic models and confusion matrix. As for Heart and Age, the second best method is LAA and ME respectively. Though both LAA and BILA-CM both use neural networks, BILA-CM has a more stable performance due to the guidance of the generating distribution.

When contrasting the results of small chunk size in Table 2.2 with results of Table 2.3, we can gauge the impact of online data feeding to different aggregation methods. On the one hand BILA-CM has little variation across different online scenarios as it can incrementally update the models chunk by chunk of data. On the other hand, EM and MMCE are observed to have high variability across datasets and online velocity, weakening their applicability for online label aggregation.

2.6. RELATED WORK

Label aggregation is a well-studied subject in crowd sourcing, especially for offline scenarios. Most of existing label aggregation solutions are unsupervised, execpt [20]. We summarize the related work in accordance with our contributions: (i) the probabilistic inference framework, (ii) confusion matrix aggregation model, and (iii) stochastic models.

Probabilistic inference models. Probabilistic models are effective to capture how the latent variables, e.g., confusion matrix, affect the likelihood of observed noise labels.

BCC [58] is the very first the probabilistic graphical model for label aggregation. It uses confusion matrix to evaluate workers, and uses Gibbs sampling to perform the parameter estimation. CommunityBCC [119] and BCCWords [106] are an extension of BCC. Specifically CommunityBCC divides the workers into worker communities. The workers in the same community have similar confusion matrices. In terms of variational methods, Liu et al.[72] propose a model which uses variational inference to approximate the posterior. Recently, [130] develops LAA, a label-aware autoencoder. LAA is an unsupervised model composed of a classifier and a reconstructor, both of which are neural networks. [70] proposes an offline probabilistic graphical model for label aggregation, including an enhanced variational Bayesian classifier combination with inference based on a mean-field variational method. Orthogonally, Yang et al. [127, 126] apply probabilistic inference models applied to jointly distill noisy labels via experts and learning tasks.

Aforementioned studies tailors for offline scenarios where lables of all items is collected at once. And, these methods require to derive a close-form of the generative model's posterior.

Confusion matrix. Confusion matrix specifies how labels are corrupted from their true class to noisy ones. It can be based on the entre dataset, each worker, and even each content, with increasing model complexity. Dawid and Skene [12] uses the confusion matrix to describe the expertise and the bias of a worker. They then design an EM algorithm for label aggregation. Raykar et al. [97] uses noisy labels to train their classification model. Their two-coin model is a variation of the confusion matrix. GLAD [123] is a model that can infer the true labels, the expertise of workers, and the difficulty of items at the same time. However, GLAD is applicable for binary labeling tasks. Furthermore, Zhou et al. [138, 137] propose the minimax entropy estimator and its extensions. In these model, the authors set a separate probabilistic distribution for each worker-item pair.

Due to the iterative nature of EM algorithms and minmax entropy, it is not straightforward to extend those methods to construct stochastic optimizer needed for online label aggregation.

Different from aformentinoed label aggregation methods, DeepAgg [20] is a supervised model based on a deep neural network. The model is trained by a seed dataset which contains noisy labels and the corresponding ground truth labels. DeepAgg can not aggregate incomplete data, where many annotators only labeled a few items.

Stochastic Optimizer First order stochastic optimization is applied to a wide range of learning problems. RMSProp [116] and Adam [59] are the state-of-the art optimizers. RMSProp updates the model parameters based on the current gradient. It achieves more robust results than stochastic gradient decent because it utilizes element-wise adaptive learning rates to update the model parameters. Adam is a variant of RMSProp. It uses a moving average to estimate the first moment of the gradients and applies the moment to update the model parameters. Often, a clipping operator on the learning rates is used to limit their values during the training and avoid gradient explosion [83]. McMahan et al. [83] provides the theoretical base for deriving the regret bound of optimizers that use clip operator.

2.7. CONCLUSION

Motivated by the need of timely and accurately data curation and the avoidance of slow response from crowd workers, we design online label aggregation framework, BILA , maximizing the likelihood of noise labels and inferring unobservable true labels. The core components of BILA are variational Bayesian inference model and a stochastic optimizer for incrementally training on online data subset. The general design of BILA is able to model any generating distribution of labels via exact computation of posterior probability distribution and neural networks based approximate distribution. We design a stochastic optimizer that can incrementally minimize the loss function of the variational inference model based on the evidence lower bound. We theoretically prove the convergence bound of the proposed optimizer in terms of parameters of gradient update. We evaluate BILA on both synthetic and real world datasets on various online scenarios. Compared to the state of the art label aggregation algorithms that adopt sliding window update, BILA shows significant and robust error reduction, especially for challenging scenarios with small chunk data set.

3

EXPLORING AND EXPLOITING THE INPUT SPACE FOR BLACK-BOX KNOWLEDGE DISTILLATION

Deep machine learning models, e.g., image classifiers, are increasingly deployed in the wild to provide services to users. Adversaries are shown capable of stealing the knowledge of these models by sending inference queries and then training substitute models based on query results. The availability and quality of adversarial query inputs are undoubtedly crucial in the stealing process. The recent prior art demonstrates the feasibility of replacing real data by exploring the synthetic adversarial queries, so-called data-free attacks, under strong adversarial assumptions, i.e., the deployed classier returns not only class labels but also class probabilities. In this chapter, we consider a general adversarial model and propose an effective data-free stealing algorithm, TANDEMGAN, *which not only explores synthetic queries but also explicitly exploits the high-quality ones. The core of* TANDEMGAN *is composed of (i) a substitute model that imitates the target model through synthetic queries and their inferred labels; and (ii) a tandem generator consisting of two networks, $\mathcal{G}_x$ and $\mathcal{G}_e$, which first explores the synthetic data space via $\mathcal{G}_x$ and then exploits high-quality examples via $\mathcal{G}_e$ to maximize the knowledge transfer from the target to the substitute model. Our results on four datasets show that the accuracy of our trained substitute model ranges between 96–67% of the target model and outperforms the existing state-of-the-art data-free model stealing approach by up to 2.5X.*

3.1. INTRODUCTION

Emerging intelligent services, such as Google translate and optical character recognition [48], are increasingly powered by deep models. Users can access these services by sending queries via APIs to get outputs, for instance, the class labels of queried images. While an open access to deployed models greatly eases users' experience, it opens up vulnerability issues related to model stealing [117, 140, 55]. Adversaries may use such an access to steal the knowledge of the deployed model and create a copy of it, named substitute model, which can then be used to do malicious actions, e.g., adversarial attacks [30, 71, 117, 120, 27] or membership attacks [128]. Several studies design defenses against model stealing [90, 56, 57] , but [6] provides theoretical evidence that model extraction is inevitable.

To launch a model stealing attack, the adversary first needs to query the target model and get the corresponding inference results. For instance, to steal an image classifier, the adversary sends query images to the deployed classifier and then receives the predicted class labels. Recognizing the issue of limited availability of real data, recent studies [117, 55] introduce data-free model stealing methods, i.e., only using synthetic query images. Generative adversarial networks (GANs), composed of generator and substitute models, are key to synthesize queries. Synthetic query quality is paramount to extract knowledge from the target model. Low quality queries provide little information to train the substitute model, e.g. low confidence results from the target model leading to small feedback losses.

Though the existing studies demonstrate the feasibility to steal models in a (real) data-free way, they are limited in the adversarial assumptions and quality of synthesised queries. The common adversarial assumption of the prior art [117, 55] is that both predicted class labels and confidence are provided by the target model. The additional information beyond class labels is crucial for existing methods to carry out GANs training which can not backpropagate the gradients simply from the class labels.

Furthermore, the competition between generator and substitute models pushes the two networks to continuously **explore** the synthetic data space, but the classic minmax loss used in training GANs eschews to **exploit** synthetic examples [23]. As a result the

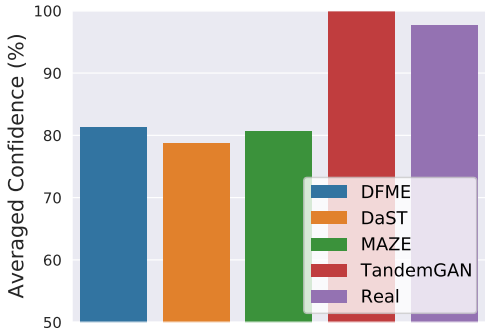


Figure 3.1: Average target model confidence on 1K real data and 1K synthetic data generated by the methods, DFME, DaST, MAZE and TANDEMGAN respectively.

average quality of synthetic queries is low, i.e., target model has low confidence in classifying synthetic data, and further limits the knowledge extraction performance in model stealing. Figure 3.1 illustrates the average prediction confidence of the target model on real and synthetic data generated by the state of the art data-free stealing methods, namely DFME [117], DaST [140] and MAZE [55]. In this example, the target model is RestNet34 trained on CIFAR 10 and the settings of adversaries can be found in the evaluation section. Shown in the figure, the existing methods can only generate synthetic queries reaching an average confidence level of 80% that is 20% lower than the real data and limits its capacity to extract knowledge.

In this chapter, we propose an effective data-free model stealing framework, TANDEMGAN, which does not require any real data and considers general adversarial assumptions where queries return label predictions, termed *label-only*, or label probabilities, termed *probability-only*. The core components of TANDEMGAN are (i) the substitute model and (ii) the tandem generator consisting of two networks to explore and exploit the synthesizing space. The substitute model minimizes the distance loss between its and the target model's predictions while the two tandem generator networks jointly synthesize diverse and high confidence query examples which maximize the stealing potential. The first network, $\mathscr{G}_x$, explores the synthetic data space, whereas the second network $\mathscr{G}_e$ exploits and refines the synthesizing space to produce high-quality synthetic examples with high target model confidence. In the example of Figure 3.4, we demonstrate more systematically the effectiveness of incorporating exploitation evaluating the performance of TANDEMGAN on stealing four deployed classifiers under both scenarios of label-only and probability-only against SOTA prior art.

The contributions of this chapter are: (i) an effective data-free model stealing framework, TANDEMGAN, which uniquely features joint exploration and exploitation of synthetic data space and examples; (ii) more general adversarial scenarios: only class labels are available to the adversary; (iii) extensive evaluation and comparison against existing SOTA data-free model stealing approaches; and (iv) remarkable accuracy of the trained substitute models, i.e., reaching 67% up to 96% accuracy of the target classifiers.

3.2. RELATED WORK

Model stealing aims to distill the knowledge from a deployed (target) model, specifically, to train a highly similar substitute model [64, 6, 140, 117, 53, 90, 101]. A successful substitute model is able to obtain the implicit mapping function (or knowledge, at high level) of the target model via different (simpler) network structures [55, 90]. Two types of model stealing methods exist depending on whether the attackers have (partially) access to real training data or not. When real data is available, knowledge distilling [35, 89] extracts the knowledge of the target model through its class probabilities and transfers it to a lightweight substitute model. Without real data attackers can only query the target model through synthetic examples [86, 64, 140, 55] –a data-free approach.

The core of existing data-free model stealing methods [55, 117, 140] follows the design principle of GANs –competing generator-substitute networks. A generator produces synthetic examples to query the target model, $\mathscr{T}$, whereas the substitute model, $\mathscr{S}$, tries to imitate/steal $\mathscr{T}$ via the synthetic query results. The target model parameters and architecture are unknown. MAZE [55] and DFME [117] rely on a gradient approxi-

Table 3.1: The properties of four existing data-free model stealing methods.

Method	Probability-only	Label-only	Exploration	Exploitation
MAZE [55]	✓	✗	✓	✗
DFME [117]	✓	✗	✓	✗
DaST [140]	✓	✓	✓	✗
TANDEMGAN (ours)	✓	✓	✓	✓

mation [94] to train their generator. DFME and MAZE can not be applied to scenarios where the target model provides only inference labels, Furthermore, DaST [140] regards the output of the target model as a constant vector forgoing the need for gradient approximation. Aforementioned studies explore the general synthesizing space, overlooking the option of exploitation. The features of different data-free model stealing methods are summarized in Table 3.1.

3.3. METHODOLOGY

In this section, we introduce TANDEMGAN, a data-free model stealing framework that explores and exploits synthetic queries. Prior to introducing the design of TANDEMGAN, we first introduce the adversarial assumptions.

Adversarial assumptions. We consider a realistic deployment setting where a target classifier[1] $\mathscr{T}$ is deployed and its parameters and architecture of $\mathscr{T}$ are unknown. The only way to interact with the target model is by sending queries, e.g., images, and getting the inferred results, for both benign and malicious adversaries. Furthermore, due to the limitation and difficulty of obtaining real data, we further assume adversaries have no access to the real data. According to the format of the inference results, we consider two types of adversarial scenarios: (i) *label-only* scenario: $\mathscr{T}$ only provides a label prediction for each query without any additional information, and (ii) *probability-only* scenario: $\mathscr{T}$ returns the class probabilities instead.

3.3.1. TANDEMGAN FRAMEWORK

We propose TANDEMGAN to steal the knowledge from $\mathscr{T}$ and train an accurate substitute model $\mathscr{S}$, shown in Figure 3.2. $\mathscr{S}$ can be regarded as a clone of $\mathscr{T}$ but with a different network architecture. Different from related work TANDEMGAN leverages a tandem generator which generates synthetic queries [2] with high classification confidence. Specifically, the generating process includes two networks, one network for exploring the new areas of the synthetic data space and one network for exploiting a particular space to generate high-quality synthetic queries. In Figure 3.3, we illustrate the projected synthesizing space of TANDEMGAN on CIFAR10, the exploration points randomly scatter, whereas the exploitation points center around some exploration points. Note that we apply UMAP [80] to reduce the data dimension for visualization.

Preliminary. In the data-free model stealing, a generator $\mathscr{G}$ is fed a noise vector z as

[1] TANDEMGAN can be extended to other model types but here we only discuss classification tasks.

[2] We refer the data sample sent to the target model as a query. In case of querying via synthetic data sample, we abbreviate it as a synthetic query.

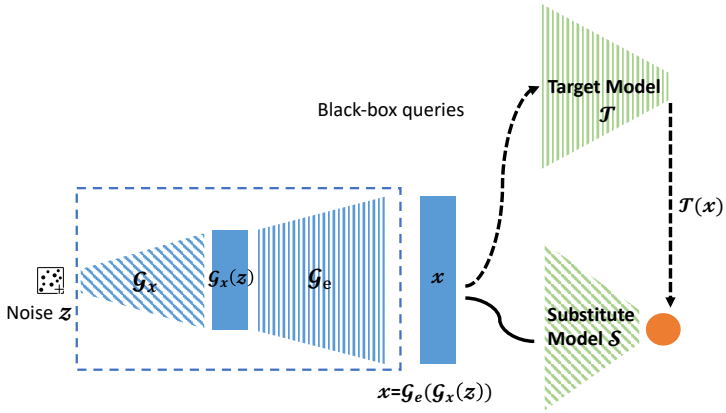


Figure 3.2: TANDEMGAN framework: data-free model stealing process.

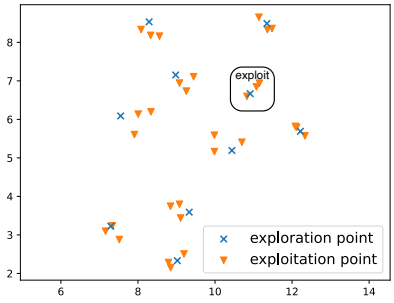


Figure 3.3: Exploration and exploitation of TANDEMGAN in projected 2D space, where the horizontal axis is the value of the first dimension of the 2D space and the vertical axis is the value of the second dimension of the 2D space.

seed to generate a synthetic example $x = \mathcal{G}(z)$ to query $\mathcal{T}$ and obtain $\mathcal{T}(x)$. We assume $\mathcal{T}$ to be a model for classification task with N classes. Depending on the context, $\mathcal{T}(x)$ is either the predicted probability vector (*probability-only*) or predicted one-hot encoded label (*label-only*) for input x and $\mathcal{T}_i(x)$ denotes the i-th ($i = 1, ..., N$) class output. Similarly, we use $\mathcal{S}(x)$ to denote the probability output of the substitute model $\mathcal{S}$ for input x.

Algorithm architecture. Figure 3.2 shows the architecture of our proposed model stealing framework, TANDEMGAN. The key components are the substitute model $\mathcal{S}$, and the tandem generator, $\mathcal{G}(z) = \mathcal{G}_e(\mathcal{G}_x(z))$, comprising $\mathcal{G}_x$ and $\mathcal{G}_e$. $\mathcal{G}$ generates synthetic queries that explore the synthetic data space via $\mathcal{G}_x$ and exploit a particular generation space via $\mathcal{G}_e$. We use $\theta_{\mathcal{S}}$, $\theta_{\mathcal{G}_x}$ and $\theta_{\mathcal{G}_e}$ to denote the model parameters of $\mathcal{S}$, $\mathcal{G}_x$ and $\mathcal{G}_e$, respectively.

During model stealing, a noise vector z is fed into $\mathcal{G}$ to produce a synthetic example x via $\mathcal{G}_x$ and $\mathcal{G}_e$. $\mathcal{G}_x$ transforms z into a latent code $\mathcal{G}_x(z)$ while $\mathcal{G}_e$ generates the query, e.g. image, from the latent code $x = \mathcal{G}_e(\mathcal{G}_x(z))$. x is used to query $\mathcal{T}$ and get the prediction $\mathcal{T}(x)$. Next, x associated with $\mathcal{T}(x)$ is used to train $\mathcal{S}$. When training $\mathcal{S}$, we minimize the distance measure to maximize the agreement between $\mathcal{S}$ and $\mathcal{T}$. Besides training $\mathcal{S}$, we separately train $\mathcal{G}_x$ and $\mathcal{G}_e$ according to the outputs $\mathcal{T}(x)$ and $\mathcal{S}(x)$. Since $\mathcal{G}_x$ is designed to explore new areas of the data generating space which can provide new knowledge to train $\mathcal{S}$, we employ the design idea of GANs – a minmax optimization to train $\mathcal{G}_x$ and $\mathcal{S}$ jointly.

Hence to train $\mathcal{G}_x$, we optimize the model parameters $\theta_{\mathcal{G}_x}$ to maximize the distance measure between $\mathcal{T}(x)$ and $\mathcal{S}(x)$. $\mathcal{G}_e$ is responsible to refine the new area searched by $\mathcal{G}_x$ to generate high-quality synthetic queries. The objective of $\mathcal{G}_e$ thus needs to be aligned with the definition of query quality. Following the risk analysis of knowledge distillation [85], we define the optimization objective of $\mathcal{G}_e$ according to effectiveness of knowledge extraction, i.e., maximizing the confidence of the target model in predicting synthetic queries. Other possible definitions by active learning [1] are discussed in Sec. 3.5. As a result $\mathcal{G}$ simultaneously explores and exploits the synthetic data space to find diverse and highly informative examples which maximize the knowledge transfer to $\mathcal{S}$. The optimization objectives of $\mathcal{S}$, $\mathcal{G}_x$ and $\mathcal{G}_e$ are detailed in the following.

3.3.2. OPTIMIZATION OBJECTIVES AND TRAINING PROCEDURE

Optimization objective of $\mathcal{S}$. Since $\mathcal{S}$ is a substitute of $\mathcal{T}$, their outputs are expected to be as similar as possible. Inspired by knowledge distillation [35], $\mathcal{S}$ imitates the outputs of $\mathcal{T}$ through the loss of distance measure. The loss function of $\mathcal{S}$ over a query example x is as follows:

$$\mathcal{L}_{\mathcal{S}} = \mathrm{D}(\mathcal{T}(x), \mathcal{S}(x)), \tag{3.1}$$

where $\mathrm{D}(\cdot)$ denotes the distance measure for loss, e.g., L1-Norm or cross-entropy. L1-Norm provides a stronger feedback than cross-entropy. This fits well the *probability-only* scenario where the loss inputs are the class probabilities. When working with the more limited scenario of *label-only*, the L1-Norm would require the output of the two models to be identical which is an aggressive condition given that only a one-hot output is provided rather than the full distribution on all classes. Thus, cross-entropy is applied

for *label-only*. After training $\mathscr{S}$, we can steal the knowledge of $\mathscr{T}$ because $\mathscr{S}$ learns the mapping of $\mathscr{T}$.

Optimization objective of $\mathscr{G}_x$. We incorporate $\mathscr{G}_x$ to diversify the latent codes fed to $\mathscr{G}_e$ and generate queries in new areas of the data space. $\mathscr{G}_x$ aims at making $\mathscr{S}(x)$ as different as possible from $\mathscr{T}(x)$. This is the opposite training objective of $\mathscr{S}$. Thus, we formulate the loss of $\mathscr{G}_x$ as:

$$\mathscr{L}_{\mathscr{G}_x} = -\mathscr{L}_{\mathscr{S}} = -\mathrm{D}(\mathscr{T}(x), \mathscr{S}(x)), \quad where\ x = \mathscr{G}_e(\mathscr{G}_x(z)). \tag{3.2}$$

By this means, $\mathscr{G}_x$ ensures that $\mathscr{S}$ is trained by a broad spectrum of diverse synthetic queries in data space to prevent model collapse or degenerated cases. It should be noted that although $\mathscr{T}(x)$ can not be differentiated directly via backpropagation (since its network parameters are unavailable), it is possible to apply gradient approximation (details below) under *probability-only* scenario. On the other hand, under *label-only* scenario, the output of the target model is the class label and it is a non-differentiable constant.

Optimization objective of $\mathscr{G}_e$. We incorporate $\mathscr{G}_e$ to generate high quality queries around a latent space explored by $\mathscr{G}_x$. We derive the loss function of $\mathscr{G}_e$ inspired by the risk analysis of knowledge distillation [85]. We aim to achieve high quality $\mathscr{T}$ so as to better teach $\mathscr{S}$. Specifically, the quality of the probability estimate of $\mathscr{T}$ can be measured by log-loss and calibration error [28], the lower the better. Inspired by this observation, for minimizing the log-loss (or calibration error) on the outputs $\mathscr{T}(x)$, in our model stealing process, the inference confidence of x, i.e., the biggest element of $\mathscr{T}(x)$, is expected to be high. Consequently, the objective of $\mathscr{G}_e$ is to generate a synthetic query x that maximizes the confidence over model $\mathscr{T}$. We thus define high-quality queries as ones with high inference confidence on $\mathscr{T}$. Then, we define the loss function of $\mathscr{G}_e$ as:

$$\mathscr{L}_{\mathscr{G}_e} = -\left\{\log \mathscr{T}_k(x) \mid \forall_j : \mathscr{T}_j(x) \leq \mathscr{T}_k(x)\right\}, \quad where\ x = \mathscr{G}_e(\mathscr{G}_x(z)). \tag{3.3}$$

With this loss, for an input example $x = \mathscr{G}_e(\mathscr{G}_x(z))$, we maximize the value of the k-th element of $\mathscr{T}(x)$ where k is the index of the biggest element. For *probability-only*, calculating Eq. (3.3) relies on gradient approximation (details below). For *label-only*, $\mathscr{T}(x)$ is a constant (one-hot label) and gradient approximation is not applicable because one can not obtain its directional derivative. Thus, we use $\mathscr{S}(x)$ to approximate $\mathscr{T}(x)$ since $\mathscr{S}(x)$ gradually approaches $\mathscr{T}(x)$ during training. Hence, updating $\mathscr{G}_e$ only needs the gradient of $\mathscr{S}$ which fits the *label-only* scenario.

Gradient approximation. Training the generator $\mathscr{G}$ requires the gradient $\nabla_{\theta_g}\mathscr{L}$ where $\mathscr{L}$ has two arguments $\mathscr{T}(x)$ and $\mathscr{S}(x)$. However, $\mathscr{T}$ is not differentiable in our black-box setting as its model parameters are unknown. Therefore we cannot obtain $\nabla_{\theta_g}\mathscr{L}$ without $\nabla_{\theta_g}\mathscr{T}(\mathscr{G}(z))$. To address this challenge, we apply gradient approximation [117] to approximate $\nabla_{\theta_g}\mathscr{L}$. Given

$$\nabla_{\theta_g}\mathscr{L} = \frac{\partial \mathscr{L}}{\partial \theta_g} = \frac{\partial \mathscr{L}}{\partial x} \times \frac{\partial x}{\partial \theta_g},$$

the second term can be computed because $\mathscr{G}$ is differentiable with known parameters. Thus, we only need to approximate $\frac{\partial \mathscr{L}}{\partial x}$. This can be done by the forward differences

method [94]. It approximates the gradient of a function at a point by computing directional derivatives along some random directions. For a synthetic example $x \in \mathbb{R}^d$, the approximate gradient is:

$$\hat{\nabla}_x \mathscr{L}(x) = \frac{1}{M} \sum_{j=1}^{M} \frac{\mathscr{L}\left(x + \epsilon u_j\right) - \mathscr{L}(x)}{\epsilon} u_j$$

where u_j is a random direction (a d-dimensional unit vector) and M is the number of directions used for the approximation. ϵ is a small step size in the direction of u_j. The approximate value becomes more precise when M increases. For TANDEMGAN, gradient approximation is used to optimize both $\mathscr{G}_x$ and $\mathscr{G}_e$ under *probability-only* scenario.

Algorithm 3.1: TANDEMGAN

1 **Input:** n_x, n_e, n_s, μ_x, μ_e and μ_s
2 In the following, $z \sim \mathscr{N}(0,1)$
3 **for** *number of rounds* **do**
4 **for** $i = 1, ..., n_x$ **do** `// explore the data space`
5 Get $\mathscr{T}(x)$ and $\mathscr{S}(x)$, $x = \mathscr{G}_e(\mathscr{G}_x(z))$
6 **if** *probability-only* **then**
7 $\mathscr{L}_{\mathscr{G}_x}(x) = -\|\mathscr{T}(x) - \mathscr{S}(x)\|_1$
8 **else**
9 $\mathscr{L}_{\mathscr{G}_x}(x) = -CE(\mathscr{T}(x), \mathscr{S}(x))$
10 $\theta_{\mathscr{G}_x} \leftarrow \theta_{\mathscr{G}_x} - \mu_x \nabla_{\theta_{\mathscr{G}_x}} \mathscr{L}_{\mathscr{G}_x}(x)$
11 **for** $j = 1, ..., n_e$ **do** `// exploit the data space`
12 Get $\mathscr{T}(x)$ and $\mathscr{S}(x)$, $x = \mathscr{G}_e(\mathscr{G}_x(z))$
13 Compute $\mathscr{L}_{\mathscr{G}_e}(x)$ `// see Eq. (3.3) for the loss calculation`
14 $\theta_{\mathscr{G}_e} \leftarrow \theta_{\mathscr{G}_e} - \mu_e \nabla_{\theta_{\mathscr{G}_e}} \mathscr{L}_{\mathscr{G}_e}(x)$
15 **for** $j = 1, ..., n_s$ **do**
16 Get $\mathscr{T}(x)$ and $\mathscr{S}(x)$, $x = \mathscr{G}_e(\mathscr{G}_x(z))$
17 **if** *probability-only* **then**
18 $\mathscr{L}_{\mathscr{S}}(x) = \|\mathscr{T}(x) - \mathscr{S}(x)\|_1$
19 **else**
20 $\mathscr{L}_{\mathscr{S}}(x) = CE(\mathscr{T}(x), \mathscr{S}(x))$
21 $\theta_{\mathscr{S}} \leftarrow \theta_{\mathscr{S}} - \mu_s \nabla_{\theta_{\mathscr{S}}} \mathscr{L}_{\mathscr{S}}(x)$
22 **Result:** The trained $\mathscr{S}$

Stealing algorithm Algorithm 3.1 shows the training process. In each round, there are two stages, exploration (line 1-5) and exploitation (line 6-7). In exploration, $\mathscr{G}$ and $\mathscr{S}$ are playing a min-max game just like GANs. By the game, $\mathscr{G}_x$ is updated to explore a new area in the latent space of $\mathscr{G}_e$. In exploitation, $\mathscr{G}_e$ exploits the area and produces synthetic examples to train $\mathscr{S}$. After the exploitation, we sample multiple examples for updating the substitute model $\mathscr{S}$ (line 8-13).

To fine-tune the balance between exploration and exploitation in each round, each training stage of the tandem generator is repeated n_x and n_e times, respectively. The optimization goal of $\mathscr{G}_x$ is opposite to $\mathscr{S}$. If the exploration is too aggressive, the training

Table 3.2: The $\mathscr{S}$ubstitute model Acc(uracy) (%) under the methods, DaST, MAZE, DFME, and TANDEMGAN. Arch. stands for model architecture.

	$\mathscr{T}$ arget		$\mathscr{S}$	Label-only		Probability-only			
Dataset	Acc.	Arch.	Arch.	DaST	TANDEMGAN	MAZE	DFME	DaST	TANDEMGAN
MNIST	99.51	VGG16	VGG11	83.98	91.30	95.13	90.22	90.16	95.95
F-MNIST	93.09	VGG16	VGG11	43.00	72.15	41.63	48.43	44.43	79.96
SVHN	94.96	ResNet34	VGG11	58.39	62.41	52.60	50.62	55.69	68.80
CIFAR10	90.71	ResNet34	VGG11	21.28	29.58	58.67	54.79	29.81	75.81

of $\mathscr{S}$ diverges. If the exploration is too conservative, $\mathscr{S}$ can collapse to a bad local optima because of the limited area searched in the latent space during training. After training $\mathscr{G}_e$, $\mathscr{S}$ needs enough updates to extract knowledge from $\mathscr{T}$.

3.4. EVALUATION

In this section, we comprehensively evaluate the model stealing performance via the accuracy of the substitute model. We compare TANDEMGAN with state-of-the-art data-free model stealing approaches and conduct an ablation study to verify the effectiveness of exploration and exploitation, and the impact of different substitute model architectures.

Datasets and model structures. We evaluate our proposed method on four benchmark datasets: MNIST, Fashion-MNIST (F-MNIST), SVHN and CIFAR10. For MNIST and F-MNIST, we use VGG16 for the target model and ResNet34 for SVHN and CIFAR10. For the substitute model, we apply VGG11 for every dataset so that comparisons on the same/different network architecture(s) family between $\mathscr{S}$ and $\mathscr{T}$ are possible. However, for each baseline approach and TANDEMGAN, we use the same target and substitute models to guarantee a fair comparison. Finally, TANDEMGAN uses a two convolutional layers network for $\mathscr{G}_x$ and a one convolutional layer network for $\mathscr{G}_e$.

Evaluation criteria. The goal of our data-free model stealing is to achieve high classification **accuracy** on substitute model. We also compare the convergence process of TANDEMGAN with the baseline models to show the learning performance. For evaluating the attack efficiency, we also show the accuracy of the substitute model on different **number of queries** during model stealing.

Experiment settings. The networks $\mathscr{S}$, $\mathscr{G}_x$ and $\mathscr{G}_e$ are trained with a batch size of 256. We apply RMSprop as the optimizer for all the networks. The recommended learning rates for $\mathscr{S}$, $\mathscr{G}_x$ and $\mathscr{G}_e$ are 0.001, 10^{-5} and 10^{-6} respectively. $\mathscr{G}_x$ has two convolutional layers and $\mathscr{G}_e$ has one convolutional layer. We apply batch normalization after each convolutional layer on the generator. For gradient approximation, we set the number of random directions M to be 1 and choose the step size $\epsilon = 0.001$. To balance the training of $\mathscr{S}$, $\mathscr{G}_x$ and $\mathscr{G}_e$, we let $n_s = 5$, $n_x \in \{1,3,5\}$ and $n_e \in \{1,3,5\}$ for all experiments. We implement our method using pytorch. Intel Xeon E3-1200 CPUs and Nvidia GeForce RTX 2080 Ti GPUs are utilized to run the experiments.

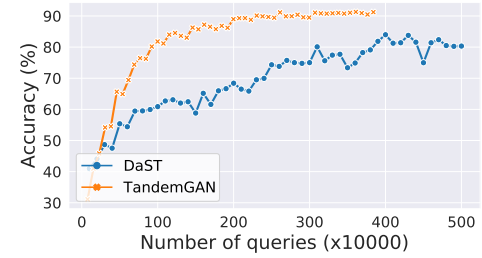
3.4.1. MODEL STEALING PERFORMANCE

Model stealing accuracy. We evaluate the accuracy of model stealing results compared to the DaST, MAZE and DFME, for both *label-only* and *probability-only* scenarios, and the original target model. Table 3.2 and Figure 3.4 summarize the results. Here we use VGG11 as substitute model. This is more challenging for SVHN and CIFAR10 since the target model is ResNet34, differing significantly in neural network architecture family. The accuracy of the target model represents the upper bound accuracy for the substitute model. It should be noted that MAZE and DFME are only used on *probability-only* scenario since they require the additional information on class probabilities. We further note that DaST, MAZE, DFME and TANDEMGAN generate synthetic data per training iteration for generator or substitute model in different ways and different order. For a fair comparison, we report the accuracy under same number of queries.

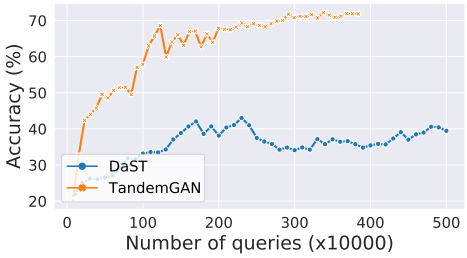
Table 3.2 shows that TANDEMGAN achieves the highest accuracy among all datasets and inference scenarios. The accuracy of TANDEMGAN's substitute models trails behind the target models by margins of 4 to 32%, except for CIFAR10 under the label-only case. In other words, TANDEMGAN can achieve roughly 96 to 67% of the accuracy of the target model for a given dataset and inference scenario. The substitute model accuracy of TANDEMGAN is consistently and significantly higher than DaST, MAZE and DFME, showing an accuracy improvement up to 67% for the challenging label-only scenario, and up to 250% for the probability-only scenario.

Comparing probability-only to label-only, the accuracy of substitute models is higher for any given stealing method that is applicable to both scenarios. This is due to the fact that probability-only provides more inference information about the target model and we do not need to use $\mathscr{S}$ to approximate $\mathscr{T}$. For label-only scenarios, the accuracy of the substitute model trained by TANDEMGAN on MNIST surpasses 90% with a less than 10 percent points gap to the target model. This strongly demonstrates the effectiveness of TANDEMGAN. Owing to the increasing task difficulty, for F-MNIST, SVHN and CIFAR10, the stealing performance of TANDEMGAN drops. Even so the accuracy of the trained substitute models is still more than half of the target models (except CIFAR10), and 1.67x, 1.06x and 1.39x the accuracy achieved by DaST. For probability-only scenarios, the additional information provided by the class probabilities improves the accuracy of the substitute models. DFME outperforms MAZE and DaST on F-MNIST. MAZE outperforms DFME and DaST on MNIST and CIFAR10. DaST is better on SVHN. However, TANDEMGAN outperforms all. More impressive, TANDEMGAN achieves results close to the target model. The gap is limited from 3 to 26 percent points.

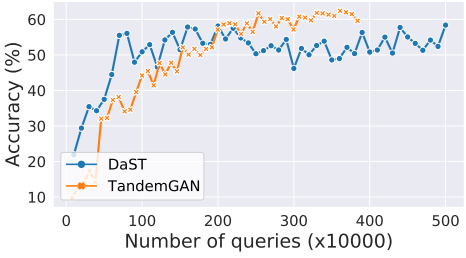
Model stealing convergence. Figure 3.4 shows the evolution of the substitute model accuracy across the number of queries to the target model. For DaST, we can see that for all cases the accuracy fluctuates and does not converge at a good local optima during the entire training.Sometimes the accuracy does not even show an increasing trend, for instance Figure 3.4b, 3.4f and 3.4h. Due to the unstable convergence, another issue with DaST is to choose an appropriate stopping criteria for training. Further, DaST saves the substitute model at each iteration, and chooses the one with the highest accuracy to be the final result. We argue that this is normally infeasible because attackers do not have real data to evaluate their saved substitute models. It also requires additional resources and efforts to store and select the best substitute model once the model stealing process



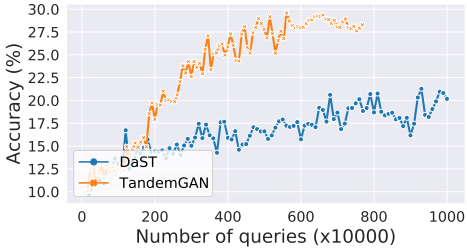
(a) MNIST, L-only



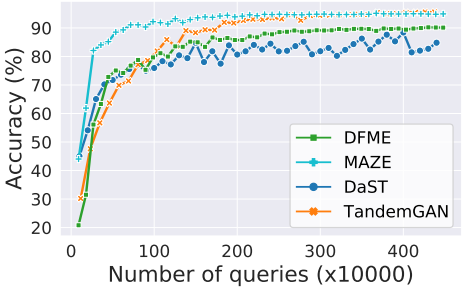
(b) F-MNIST, L-only



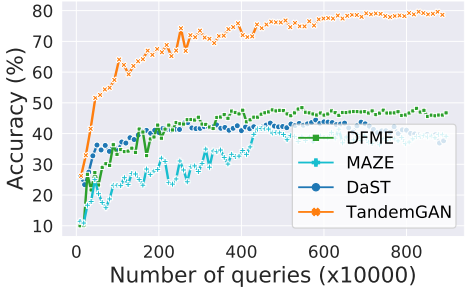
(c) SVHN, L-only



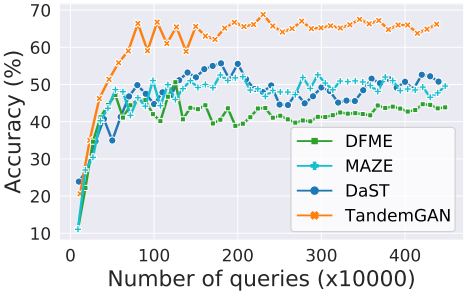
(d) CIFAR10, L-only



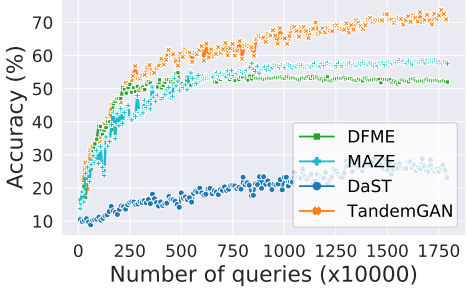
(e) MNIST, P-only



(f) F-MNIST, P-only



(g) SVHN, P-only



(h) CIFAR10, P-only

Figure 3.4: The convergence of accuracy of substitute models versus the number of queries during training. For the label-only scenario, there are the results on the datasets, MNIST (a), F-MNIST (b), SVHN (c) and CIFAR10 (d). For the probability-only scenario, the experiments are conducted also on MNIST (e), F-MNIST (f), SVHN (g) and CIFAR10 (h).

ends. This further leads to the unstable training process of DaST. For MAZE and DFME, the convergence has no significant oscillation. The convergence of TANDEMGAN is almost monotonic. The accuracy increases smoothly during the whole training, and converges to local optima at around 2M queries for MNIST and SVHN, and 4M for F-MNIST respectively. For CIFAR10, label-only scenario requires 6M queries while it takes around 13M for probability-only scenario. The reason behind the different number of queries is that label-only scenario provides limited information which cannot improve the accuracy even with increased queries for this relatively difficult task. This observation also implies that less information provided by the target model may serve as the short board for challenging tasks to apply model stealing attack.

Inherited from GANs, the generator design of DaST, MAZE and DFME only contains exploration to train the generator and substitute model in a min-max game. In TANDEMGAN, besides $\mathscr{G}_x$ to search for new space areas which generate diverse data examples, the tandem generator also contains $\mathscr{G}_e$ to exploit the latent code generated by $\mathscr{G}_x$, fine-tuning the task-specific data properties to train $\mathscr{S}$ for model stealing. Benefiting from both exploration and exploitation, our synthetic data captures the real data training scenarios better, resulting in stable convergence towards a good local optima.

3.4.2. ABLATION STUDY

Table 3.3: Ablation study for exploration and exploitation:the $\mathscr{S}$ubstitute model Acc(uracy) (%)

Ablation	TANDEMGAN
main results	75.81
w/o exploration	60.74
w/o exploitation	67.23

Table 3.4: Analysis for different substitute model architectures:the $\mathscr{S}$ubstitute model Acc(uracy) (%)

Architecture	TANDEMGAN
VGG11	75.81
ResNet18	84.70
AlexNet	23.25

Importance of both exploring and exploiting. In previous sections we highlight the advantages to additionally exploit, and not only explore, the synthetic examples. To quantify the benefits of either phase, we present an ablation study using VGG11 as $\mathscr{S}$ for CIFAR10 where TANDEMGAN forgoes either $\mathscr{G}_x$ (w/o exploration) or $\mathscr{G}_e$ (w/o exploitation). This is achieved by skipping the corresponding training phase, i.e. keeping either $\theta_{\mathscr{G}_x}$ or $\theta_{\mathscr{G}_e}$ fixed. Table 3.3 shows the results. One clearly sees that both exploring the synthetic data space and exploiting known examples holds the best results. Without exploitation the accuracy drops by 8 percent. This is in line with the result obtained by DFME which also performs only exploration (see Table 3.2). Without exploration the accuracy is reduced by 15 percent. This shows that exploration and exploitation are both important for training a high accuracy substitute model.

Impact of architecture choice. As the neural network architecture is unknown by the attacker, we evaluate the impact of choosing different architectures for $\mathscr{S}$ Besides VGG11, we try ResNet-18, which is of the same neural network family of the target model (using ResNet34), as well as AlexNet which is a simpler CNN [76, 103]. The accuracy of the substitute AlexNet is low, i.e., 23.25% (see Table 3.4). The reason is that AlexNet contains 5 convolutional layers and 3 fully connected layers which is too shallow to train on CIFAR10. Choosing a suitable task-specific substitute model architecture is cru-

Table 3.5: Comparison of the $\mathscr{S}$ubstitute model Acc(uracy) (%) under the proposed method, TANDEMGAN, and the baseline, stealing the target using publicly available data samples. Arch. stands for model architecture.

Dataset	$\mathscr{T}$ arget		$\mathscr{S}$	Label-only		Probability-only	
	Acc.	Arch.	Arch.	Public Samples	TANDEMGAN	Public Samples	TANDEMGAN
F-MNIST	93.09	VGG16	VGG11	37.08	72.15	32.06	79.96
CIFAR10	90.71	ResNet34	VGG11	15.45	29.58	14.01	75.81

cial.ResNet18 achieves the highest accuracy. While ResNet18 is able to achieve slightly better accuracy than VGG11 on CIFAR10, i.e. 92.36% against 91.6%, we impute the performance gap of 9 percent points mainly to the sharing of the same neural network family between $\mathscr{S}$ and $\mathscr{T}$. Since the exact architecture of $\mathscr{T}$ is unknown to attackers we choose VGG11 for our main results. Using VGG11 we can better verify broadly the effectiveness and generality of TANDEMGAN.

Stealing the target using publicly available data samples. It is possible to query a target model and steal it using publicly available data samples (E.g., MNIST samples etc), when the real training dataset of the target model is unknown. To study the effectiveness of directly using publicly available data samples for model stealing, in the following, we apply MNIST samples to steal a F-MNIST target model and use SVHN samples to steal a CIFAR10 target model. The results are shown in Table 3.5. We can see that TANDEMGAN significantly outperforms the baseline which directly uses public samples for querying. Besides, comparing to Table 3.2, DaST, MAZE and DFME also outperform the baseline. Directly using public data samples to query cannot effectively and efficiently search the input data space of the target model, especially when the model and the task are complicated e.g., the CIFAR10 target. It also cannot utilize the inference feedback from the target to adjust the stealing process. That's why it is much worse than DaST, MAZE, DFME and TANDEMGAN which apply data sample generators for data space searching. In a data-free scenario, it is important to design a good data space searching strategy to perform model stealing attacks.

3.5. POSSIBLE EXTENSION

In this section, we discuss the possible extension of the proposed algorithm. In the following, we show some other perspectives of exploitation.

In the methodology part, we derive the optimization objective of $\mathscr{G}_e$ from a statistical observation [85] that the quality of a target (or a teacher) $\mathscr{T}$ can be measured by log-loss and calibration error (the lower the better). Since we regard the predictions from $\mathscr{T}$ as the ground truth labels in model stealing, in order to minimize the log-loss (or calibration error) on $\mathscr{T}$ we aim to increase the inference confidence of each data example x. Then we define the loss (see Eq. (3.3)) according to this motivation. From the statistical perspective, the high-quality examples produced by exploitation are defined as examples with high inference confidence on $\mathscr{T}$. Actually, it is possible to extend TANDEMGAN by designing the optimization objective of $\mathscr{G}_e$ from other perspectives.

From active learning [98] perspectives, $\mathscr{T}$ can be seen as an oracle or a expert who can provide ground truth labels for unlabeled examples. In order to train $\mathscr{S}$, we need examples to query $\mathscr{T}$ and get the predictions. For efficiently and effectively querying $\mathscr{T}$,

we need query strategies to select informative examples (high-quality examples). Therefore the query strategies, e.g., variance reduction, entropy sampling and margin sampling etc, of active learning can be utilized to define high-quality examples and design the optimization objective of $\mathscr{G}_e$.

If attackers know some prior information about the training data space of $\mathscr{T}$, e.g., data distribution, we can also consider the information when designing the loss of $\mathscr{G}_e$. In this case, $\mathscr{G}_e$ can be utilized to ensure that the data examples are sampled from the prior distribution. A special case is to consider the class balance of the generated examples when training $\mathscr{G}_e$.

3.6. CONCLUSION

It is challenging to design adversarial attacks without knowing the target model parameters nor having access to real-world data. In this chapter, we propose a novel and effective data-free model stealing framework, TANDEMGAN, consisting of substitute model and a tandem generator networks, which aims to steal the knowledge of the target model by synthetic queries. Beyond the state of the art, we not only consider a general adversarial scenario with only the availability of predicted class labels only but also design a steal optimization algorithm to explore and exploit synthetic queries generation. We empirically demonstrate that TANDEMGAN effectively steals the target model using a small number of queries for four datasets. Under various adversarial scenarios, we show that the model stolen through TANDEMGAN achieves up to 2.5 times higher accuracy than state-of-the-art data-free model stealing attacks, and its accuracy is as high as the 96 – 67% of target model.

4

SEMANTIC QUERY GENERATION FOR EFFICIENT BLACK-BOX KNOWLEDGE DISTILLATION

Model stealing allows to extract the knowledge of a deployed target machine learning model by sending query images and training a substitute model via the inference results. However, accessing the same data used to train the target model is often impractical. Recent data-free model stealing methods overcome this limit using specifically crafted noise as queries. However, two major flaws limit their effectiveness. First, data-free methods suffer from low query efficiency, requiring high query budgets, such as over 20 million queries to train a substitute for CIFAR-10. Second, crafted queries lack any perceptual semantics. Hence, they can easily be filtered out from legitimate requests. To address these issues, we propose AEDM*, a framework for* ***A****dversarial knowledge* ***E****xtraction via steering* ***D****iffusion* ***M****odels.* AEDM *leverages publicly available pre-trained diffusion models to craft adversarial query images, controlled through the latent variable fed into the diffusion models, to maximize the knowledge transferred to the substitute model. These queries not only resemble in all aspects legitimate requests of images with a semantic meaning, but also significantly lower the number of required queries. Results on three datasets demonstrate that, given a budget as limited as 1/200 queries of the baselines, the accuracy of our trained substitute model can outperform that of state-of-the-art data-free stealing methods.*

Table 4.1: The required number of queries to steal a CIFAR-10 classifier under different methods, DFME, MAZE, TandemGAN, DS and AEDM.

Method	DFME [117]	MAZE [55]	TandemGAN [46]	DS [3]	AEDM (ours)
Queries (x10K)	800	900	2000	600	**10**

4.1. INTRODUCTION

A rich variety of online services such as Google OCR [10] and GPTZero [114] rely on deep learning models trained at high costs to provide their functionalities. Users interact with these services through APIs, which allow them to send queries to the model and receive the inference results. Most APIs provide a per-class probability distribution as inference results. While this open accessibility greatly enhances user experience, it also creates opportunities for malicious parties to steal knowledge from the deployed *target* models [117, 140, 55, 129]. Adversaries extract knowledge by training a *substitute* model, which aims to replicate the mapping of the target model. Such substitute models can then be exploited for profit by hosting inference services [64] or to launch further attacks, such as membership inference attacks [128] and adversarial attacks [7, 29].

Ideally, adversaries use queries semantically similar to those used to train the target model. However, in many cases, adversaries face challenges in obtaining such real data, particularly from privacy- and business-sensitive domains like banks and medical institutions. Taking into account the limited accessibility to real training data, recent studies [117, 55, 46, 3] propose data-free model stealing methods. Such methods use a generator designed to synthesise query examples out of random noise as inputs to query the target model, where real data is no longer necessary. Impressively, the performance of substitute models trained with such data-free methods has achieved levels similar to query by real data. However, data-free stealing has two significant drawbacks that limit its practical efficacy. First, it requires massive numbers of queries. For example, stealing a benchmark CIFAR-10 image (32×32 pixels) classifier based on ResNet-34 [33] requires between 6 and 20 million synthetic queries (see Table 4.1). Querying the online service multiple times makes it easy for the target to detect the adversarial behaviour, and for the attacker to exceed the cost budget. Second, the key difference among diverse data-free model stealing baselines is how they generate query examples, but overall, all craft specifically designed noise with no semantic meaning. We report some example queries generated by data-free stealing baselines in Figure 4.2. The lack of semantic meaning allows the target to easily single out adversarial queries from legitimate queries. For example, a simple defence that compares the entropy of the training data against the data from adversarial queries is already able to drop the defence pass rate below 21% for all baselines (see Table 4.5 for details).

To address both challenges we propose to enhance the query quality. Our observation study highlights that semantic information has a significant impact on both knowledge extraction and hiding adversarial queries from the target model. Leveraging the capabilities of emerging diffusion probabilistic models [13, 36, 21, 47], which generate high-quality realistic images from latent noise, we introduce a novel framework called **A**dversarial knowledge **E**xtraction via steering **D**iffusion **M**odels (AEDM). This frame-

Figure 4.1: Visualization of some synthetic samples generated by our method AEDM, where the samples are used to query the target model.

work relies on a diffusion model pre-trained on public data to infuse the adversarial queries with semantic meaning so that they seem realistic and legitimate while being optimised to maximise knowledge extraction. The diffusion model serves as a semantic knowledge prior. During query generation, AEDM tunes the latent noise of the diffusion model so that the synthesised images maximise the entropy of the output of the substitute model. Then the tuned query is sent to the target model. To efficiently and effectively steal the target model, the substitute model minimizes the distance between its predictions and those returned by the target model. Figure 4.1 shows examples of optimised queries when stealing a classifier trained on CIFAR10. One can note how the queries perceptually resemble images from the target model training dataset, though the diffusion model is pre-trained on a different dataset, i.e. ImageNet. This demonstrates the effectiveness of capturing knowledge from the target model.

This process is repeated until the performance of the substitute model resembles that of the target model. This approach enables the sampling of diverse and high-quality queries which maximise the learning capability of the substitute model and drastically reduces the number of required queries. Evaluation on three datasets has shown that using semantic-infused queries, AEDM is able to successfully steal the target model with 1/200 – 1/100 of the number of queries needed by state-of-the-art data-free baselines, achieving up to 65.88% higher stealing accuracy than baselines with more queries. At the same time, the stealthiness of the knowledge extraction is increased. Due to the closing of the semantic gap between adversarial and legitimate queries, AEDM overcomes simple defences effective against the other baselines.

In summary, the contributions of this chapter are as follows. In Section 4.3, we propose a framework for knowledge extraction, AEDM that significantly reduces the number of queries required for successful stealing. The generation of high-quality queries is facilitated by a novel diffusion model prior. In Section 4.4, we conduct extensive comparisons between AEDM and state-of-the-art data-free model stealing approaches. Our empirical results demonstrate that AEDM can efficiently and effectively steal models using very few queries while achieving higher stealing accuracy. Additionally, AEDM is effective under federated extraction settings to further reduce the number of queries per attacker with only a slight accuracy loss. The existing defence against data-free attacks also fail to detect most of AEDM real-like queries with semantic meaning, arising the need for more advanced defences for such adversarial knowledge extraction methods.

4.2. RELATED WORK

Model stealing aims at extracting the knowledge from a target model to a closely resembling substitute model, which is usually smaller in neural network size [64, 51, 6, 140].

Compared with white-box knowledge distillation [35, 129], where the layer-wise parameters of the target model are used to train the substitute model, adversarial model stealing can only leverage the inference information obtained from the target model through numerous queries [117, 53, 90]. Specifically, given the same input, the substitute model aims to approximate the output of the target model by updating its network parameters, thus, expecting to copy the implicit mapping function of the target model. Early model stealing studies assume (partial) access to the training data used to query the target model [35]. Their application is limited when such training data is not available.

Data-free stealing is proven effective without the necessity of using real training data [64, 55, 140, 46]. To accomplish this, the adversary continuously optimizes a generator to synthesize high-quality noise to query the target model. For output approximation, DFME [117] relies on the L1 norm loss when quantifying the output difference between the target and substitute model, whereas DaST [140] uses the cross entropy loss, and MAZE [55] applies the Kullback–Leibler divergence to measure the distance. Stealing by synthetic data benefits from diverse query examples [46, 3]. To thoroughly investigate the input space, TandemGAN [46] brings a dual generator which explores the synthetic data space and then exploits high-quality examples to maximize the knowledge transfer. Differently, DS [3] explores a more diverse input space via two symmetrically trained substitute models. However, data-free model stealing typically requires a significant number of queries due to the substantial difference between the training inputs of the target and substitute models. In contrast AEDM leverages a public pre-trained distillation model to close this gap utterly reducing the number of queries to achieve successful stealing.

4.3. METHODOLOGY

In this section, we first present the fundamental concept and problem statement of classic data-free model stealing attacks. Then, we motivate the benefits of using adversarial queries which seem realistic to the target but are optimized to maximize knowledge extraction. Thus, we propose our AEDM framework, where we leverage a pre-trained diffusion model to generate semantic meaningful queries. Finally, the local training objective of the substitute model is further elaborated to closely mimic the target model.

4.3.1. PROBLEM STATEMENT

Data-free Model Stealing. Given a trained target N-class classification neural network $\mathcal{T}$, where its network architecture, training data and model parameters are unknown to adversaries. The attacker attempts to copy the knowledge of $\mathcal{T}$ by training a substitute model $\mathcal{S}$. It should be noted that under our setting with no access to the training data, the query sample is synthetic image data, denoted as $x \in \mathbb{R}^{H\times W\times C}$, where H, W, and C are image height, width and the number of channels, respectively. Given the same input x, the substitute output $\mathcal{S}(x)$ is expected to be similar with the target output $\mathcal{T}(x)$. To achieve this, the target model $\mathcal{T}$ allows restricted access for querying, which means the attackers are able to obtain $\mathcal{T}(x)$ for a limited number of queries. Here the inference output $\mathcal{T}(x)$ is defined by probability vector $[p_1, p_2, ..., p_N]$ among different class labels where $\mathcal{T}_j(x)$ denotes the j-th ($j = 1, ..., N$) element of the output vector $\mathcal{T}(x)$. Hence,

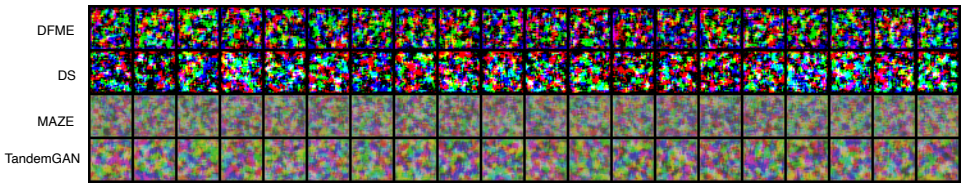


Figure 4.2: Visualization of some synthetic samples generated by the baseline methods, DFME, DS, MAZE and TandemGAN, where the samples are used to query the target model.

Table 4.2: Comparison of the (Acc)uracy (%) of the substitute models over 10000 queries of different type of query data.

Dataset	Target Acc.	Substitute Acc. CIFAR-10	Noise	ImageNet
CIFAR-10	96.59	59.56	10.00	18.81

data-free model stealing leverage the input x and the inference $\mathscr{T}(x)$ pairs to train the substitute model $\mathscr{S}$. Then, the important problem is designing methods to craft synthetic query samples x.

4.3.2. MOTIVATION STUDY

Current state-of-the-art data-free model stealing baselines seek to produce high-quality queries x by optimizing a untrained generators, which generate queries from random latent variables. The generator parameters are tuned during the stealing process so as to effectively search the data space of x. Figure 4.2 illustrates the synthetic queries generated by baseline data-free methods of DFME, DS, MAZE, and TandemGAN. It is evident that they are all noisy images. According to the visualization, none of them exhibits any perceptual meaning.

We compare the model stealing performance of $\mathscr{S}$ using different types of query data: *i)* the same data used for training the target model, *ii)* Gaussian noise data sampled from $N(0, I)$ and *iii)* realistic images sourced from different public dataset, as shown in Table 4.2. Here, we employ the *CIFAR-10* dataset as an example and report the model classification accuracy of the substitute model over 10K queries. The accuracy of the target model $\mathscr{T}$ is 95.59%. The comparison in Table 4.2 reveals that the quality of queries significantly influences the training of substitute models. Random noise without any generator design targeting specific tasks fails in extracting the knowledge. Moreover, the substitute model's performance benefits from leveraging query data infused with semantic information. For instance, when querying with images from a different public datasets even with no semantic overlap, the accuracy is notably higher compared to using noise.

The above results motivated us to reconsider the power semantic information can have in knowledge extraction. Inspired by active learning, which proactively selects the subset of high quality examples to be labeled next by an expert from the pool of unla-

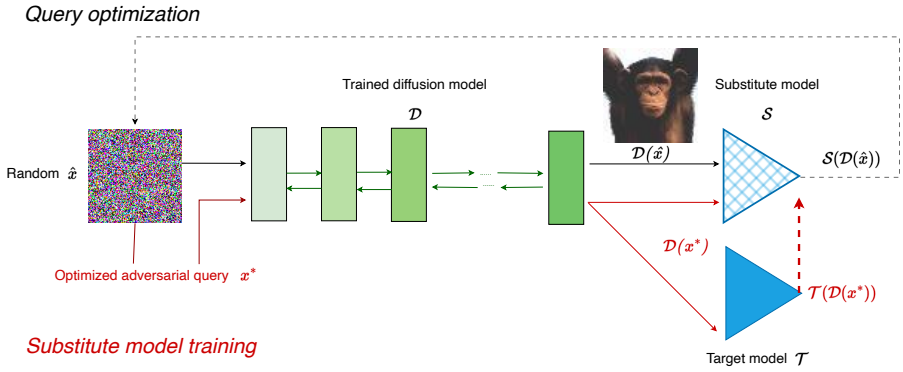


Figure 4.3: AEDM framework on a single client: data-free model stealing process.

beled data, we propose to train $\mathcal{S}$ by "selected" semantic-infused queries x^* as follows.

4.3.3. AEDM FRAMEWORK

We propose our model stealing algorithm based on Adversarial Knowledge Extraction via Steering Diffusion Models (AEDM), which can steal the knowledge from N-class image classifier $\mathcal{T}$ and train an accurate substitute model $\mathcal{S}$ with low query budget. The model parameters for $\mathcal{S}$ are represented as θ_s. To optimize θ_s so that $\mathcal{S} \sim \mathcal{T}$, we first generate realistic high-quality queries infused with semantic information, which leverages a pre-trained diffusion models as prior. Then, such diverse queries are used to minimize the inference difference between the target model and the substitute model. Algorithm 4.1 summarizes the attack steps.

Architecture The architecture of AEDM, is shown in Figure 4.3. We first generate random Gaussian noise $\hat{x}$ that is of the same dimension of the realistic synthetic images $\mathcal{D}(\hat{x})$. The Gaussian noise will be used to sample a realistic image containing semantic information form a pre-trained diffusion model to query the current (or initialized, for the first iteration) $\mathcal{S}$ and get a high-quality query x^* based on $\hat{x}$. Given x^*, AEDM feds x^* into $\mathcal{T}$ and $\mathcal{S}$ at the same time to get outputs for updating the neural network parameters of $\mathcal{S}$. Note that this is an iterative process. Each query will use the substitute model updated based on to the previous query. The algorithm procedure alternates query generation and substitute model optimization.

Query generation. Our objective is to generate queries that exhibit diversity, exploring a wide data space, while also containing semantic information. To achieve this, we apply a pre-trained diffusion model to provide semantic information in the produced queries while design a query optimization method to search the input data space of the diffusion model.

Diffusion models are a class of latent variable generative models designed to synthesize high-quality data. A well-trained diffusion model possesses the capability to generate realistic and diverse data following a training procedure that includes both a forward and reverse process. During the forward process, Gaussian noise is progressively added to the original training data across multiple steps. Each step introduces a controlled

Algorithm 4.1: AEDM

Require: A trained target model $\mathscr{T}$, a pre-trained diffusion model $\mathscr{D}$
Input: n_r, n_d, μ_d and μ_s
Initializing the substitute parameters θ_s
Function `AdversarialExtraction`$(\theta_s, \mathscr{D})$**:**
    Initialize $\mathscr{S}$ by θ_s
    for $t = 1, ..., n_r$ **do**
        $\hat{x} \sim \mathcal{N}(0, I)$
        for $i = 1, ..., n_d$ **do** `// Query generation`
            Get the synthetic output $\mathscr{D}(\hat{x})$
            Get the substitute output $\mathscr{S}(\mathscr{D}(\hat{x}))$
            $\mathscr{L}_{\mathscr{D}} = \sum_{j=1}^{N} p(\mathscr{S}_j(\mathscr{D}(\hat{x}))) \log p(\mathscr{S}_j(\mathscr{D}(\hat{x})))$
            $\hat{x} \leftarrow \hat{x} - \mu_d \nabla_{\hat{x}} \mathscr{L}_{\mathscr{D}}$
        end
        Let x^* be the optimized $\hat{x}$
        Get $\mathscr{T}(\mathscr{D}(x^*))$ and $\mathscr{S}(\mathscr{D}(x^*))$
        $\mathscr{L}_{\mathscr{S}}(\mathscr{D}(x^*)) = Distance(\mathscr{T}(\mathscr{D}(x^*)), \mathscr{S}(\mathscr{D}(x^*)))$, compute $\nabla_{\theta_s} \mathscr{L}_{\mathscr{S}}(\mathscr{D}(x^*))$
        $\theta_s \leftarrow \theta_s - \mu_s \nabla_{\theta_s} \mathscr{L}_{\mathscr{S}}(\mathscr{D}(x^*))$ `// Training the substitute model` $\mathscr{S}$
    end
Return θ_s
Result: The trained θ_s

level of structured complexity. Conversely, the reverse denoising process is responsible of learning to identify and remove the specific noise patterns introduced at each step, thus restoring the original data. Consequently, the input and output dimensions remain the same across all steps, and a well-trained diffusion model can effectively sample data resembling real-world data with semantic information from Gaussian noise.

Given a pre-trained diffusion model $\mathscr{D}(\cdot)$ learned from a public dataset and random Gaussian noise input $\hat{x} \in \mathbb{R}^{H \times W \times C}$, an intuitive method to obtain semantic-enhance queries with diversity is:

$$x^* = \arg\max_{\hat{x}} (-\sum_{j=1}^{N} p(\mathscr{S}_j(\mathscr{D}(\hat{x}))) \log p(\mathscr{S}_j(\mathscr{D}(\hat{x})))), \tag{4.1}$$

where $x^* \in \mathbb{R}^{H \times W \times C}$ is the optimized input for producing semantic-enhanced queries $\mathscr{D}(x^*)$, obtained by maximizing the entropy of the output from the substitute model. In Eq. 4.1, we optimize $\hat{x}$ while $\mathscr{D}$ is a static pre-trained diffusion model on public datasets. Remarkably, it is not necessary that $\mathscr{D}(\cdot)$ generates very similar images as the training data of $\mathscr{T}$. This makes our method fit for general applications. In the evaluation, we will present the results of different datasets for the target model and diffusion model to demonstrate the versatility.

A query sample $\mathscr{D}(x^*)$ is crafted to maximize the entropy of the substitute output. The high entropy on the produced query sample means that it is a informative sample

Algorithm 4.2: Federated adversarial extraction

Input: n, K
Initializing the substitute parameters $\theta_{s,0}$
for $t = 1, ..., n$ **do** `// n rounds of Federated training`
  for $k = 1, ..., K$ **do**
    $\theta^k_{s,t} =$ `AdversarialExtraction` $(\theta_{s,t-1}, \mathscr{D})$
  end
  $\theta_{s,t} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \theta^k_{s,t}$
end
Result: The global $\theta_{s,n}$

for $\mathscr{S}$. The substitute $\mathscr{S}$ rarely encountered similar samples in previous training. Thus, it is reasonable that we use such a sample, $\mathscr{D}(x^*)$, to query the target model $\mathscr{T}$ and get the inference result for training the substitute model.

Substitute model optimization. Since $\mathscr{S}$ is a substitute of $\mathscr{T}$, their outputs on the same given input are expected to be as similar as possible. In our method, $\mathscr{S}$ imitates the outputs of $\mathscr{T}$ through minimizing their output distance on the same input $\mathscr{D}(x^*)$ obtained from the previous step. The loss function of $\mathscr{S}$ over $\mathscr{D}(x^*)$ is:

$$\mathscr{L}_{\mathscr{S}}(\mathscr{D}(x^*)) = Distance(\mathscr{T}(\mathscr{D}(x^*)), \mathscr{S}(\mathscr{D}(x^*)))$$

In knowledge extraction, the commonly used distance metrics are cross entropy and l_1 norm. Experimentally, we find that using l_1 norm can make $\mathscr{S}$ converge with less iteration. Thus, we choose l_1 norm in our implementation. Then, we have the following loss

$$\mathscr{L}_{\mathscr{S}}(\mathscr{D}(x^*)) = \|\mathscr{T}(\mathscr{D}(x^*)) - \mathscr{S}(\mathscr{D}(x^*))\|_1. \tag{4.2}$$

Accordingly, the optimization objective of training the substitute model is:

$$\arg\min_{\theta_s} \|\mathscr{T}(\mathscr{D}(x^*)) - \mathscr{S}(\mathscr{D}(x^*))\|_1 \tag{4.3}$$

After training $\mathscr{S}$ by many rounds of queries, the mapping of $\mathscr{T}$ from a given image to the classification output can be captured by $\mathscr{S}$. Thus, the knowledge of the target model $\mathscr{T}$ is extracted by the algorithm via building the substitute $\mathscr{S}$.

4.3.4. FEDERATED EXTRACTION

To further reduce the number of queries from a single adversarial party and lower the possibility of detection by the target model, AEDM can also be applied in a federated context for model stealing. In our federated extraction approach, multiple adversarial parties, i.e., clients or attackers, collude to collaboratively train over n global rounds the substitute model without directly sharing queries. Consequently, the number of queries for each adversary decreases to $1/K$ of the original, where K represents the number of clients involved.

In particular, a single adversarial server coordinates multiple client entities engaged in stealing. In each global training round t, every client employs the stealing algorithm

shown in Algorithm 4.1 to train their local substitute model θ_s^k, based on the publicly available diffusion model $\mathscr{D}$. The parameters of these local substitute models at the end of each round t are then aggregated through averaging:

$$\theta_{s,t} = \frac{1}{K}\sum_{k=1}^{K}\theta_{s,t}^k. \tag{4.4}$$

The aggregated substitute model parameters will be distributed to every clients for next round of training. The federated adversarial extraction process by multiple attackers is shown in Algorithm 4.1.

4.4. EVALUATION

In this section, we conduct a thorough evaluation of the performance of our adversarial knowledge extraction via steering diffusion models, measured by the accuracy of the substitute model and the number of queries. We start with introducing our experimental setups and then provide a comprehensive comparison between our AEDM and state-of-the-art baseline stealing methods. Furthermore, we expand the scope of our evaluation by demonstrating its performance under Federated knowledge extraction settings, where multiple adversarial parties collaboratively launch the stealing attack. We also apply a commonly-adopted defence mechanism against knowledge extraction to validate the effectiveness of our attack. In addition, we investigate the sensitivity of our framework across different neural network architectures of substitute models, providing insights into real-world applications of stealing.

4.4.1. EXPERIMENTAL SETUP

Dataset and Model Structure: Our proposed method is evaluated on three 3-channel image classification datasets: CIFAR-10 [65], CIFAR-100 [65], and SVHN [88]. It is important to note that different network structures are used for the target and substitute models since attackers lack prior knowledge of the target structure. Specifically, for the three data sets, we use the commonly adopted ResNet-34 architecture for the target model $\mathscr{T}$, while we use VGG-16 for the substitute model $\mathscr{S}$ in the experiments.

Pre-trained diffusion models: In the context of AEDM, it is not required for the adversary to have in-distribution similar data as the training set used by the target model. Instead, access to publicly available pre-trained diffusion models is sufficient and practical. Therefore, for all three data sets, we use diffusion models trained on ImageNet [14].

Federated Adversarial Setting: We evaluate Federated collaborative stealing with 5, 10 and 20 collusive attackers. The total number of queries remains the same as for a single attacker, with each attacker using an equal split, i.e. $1/K$, of the total number of queries. The pre-trained diffusion models held by each client are trained by ImageNet, which is different from our evaluation datasets to validate the versatility of the approach.

Evaluation Metrics: Our aim in adversarial knowledge extraction is to attain high classification **accuracy** on the substitute model with a minimal **number of queries**. When encountering defences, we also evaluate the **defence pass rate** (DPR $\in [0,1]$) of adversarial queries, which is defined by the rate of undetected attacker queries by the target model.

Table 4.3: The $\mathscr{S}$ubstitute model Acc(uracy) (%) under MAZE, DFME, DS, TandemGAN and AEDM on *SVHN*, *CIFAR-10* and *CIFAR-100*. The network architectures is ResNet-34 for all target models and VGG-16 for all substitute models.

	$\mathscr{T}$ **arget**		**Baselines**				**Ours**	
Dataset	Acc.	Querie	DFME	MAZE	DS	TandemGAN	Querie	AEDM
SVHN	95.18	2M	91.50	89.84	89.68	93.45	**20K**	**93.47**
CIFAR-10	96.59	20M	60.18	55.33	58.84	77.55	**0.1M**	**88.45**
CIFAR-100	78.71	20M	25.91	14.12	22.00	37.17	**0.2M**	**61.66**

Baseline Methods: We compare our adversarial knowledge extraction with four state-of-the-art baseline methods under the same settings: DFME [117], MAZE [55], DS [3] and TandemGAN [46].

4

4.4.2. ADVERSARIAL KNOWLEDGE EXTRACTION PERFORMANCE

We evaluate AEDM against the state-of-the-art baselines based on the accuracy of substitute models and the number of queries required to achieve such accuracy. The results are detailed in Table 4.3. The accuracy of the target ResNet-34 models trained on three different datasets is provided as reference for comparison. For each baseline method, the substitute model is trained by VGG-16 due to limited knowledge of the target network by attackers. Since AEDM is designed to notably minimize the required query number, we also present the number of queries for each dataset accordingly.

As shown in Table 4.3, baseline data-free extraction methods, which employ tuned synthetic noisy data to query the target model, typically need a large number of queries ranging from 2 million (notated as 2M in the table) to 20M. This aligns with our expectations, as illustrated by the motivation results in Table 4.2, and indicates the inefficiency of using synthetic noisy data without semantic meanings. Although such noise is well-designed and generated to explore diverse random data spaces, its resemblance to the actual training data of the target model is limited. Consequently, certain important features containing semantic information (such as smooth pixel-wise edges and typical luminance contrast) from the target model's training images are difficult to capture with noisy data. In contrast, leveraging the semantic information provided by the diffusion model, alongside our optimized latent variables, our AEDM requires merely 20K to 0.2M queries on the same target model to achieve even higher accuracy than the baselines. We visualized queries generated by our AEDM in Figure 4.1. Note that the diffusion model is pre-trained by public data, i.e. ImageNet [14], and we optimize the latent variable to generate query data which maximises the knowledge extraction from the target model.

Zooming into the impact across different datasets, we can clearly observe that baseline methods employing synthetic noisy data perform effectively on the *SVHN* dataset, which is the relatively simple task of classifying street house numbers. The target model for this dataset is originally trained on monotonous data with shallow features, that makes it easily extractable by substitute models. However, when confronted with a more complex task such as the 100-class classification of *CIAFR-100*, synthetic noisy queries struggle to train high accuracy substitute models, even with query numbers reaching tens of millions. However, our AEDM maintains good performance with only 1/100

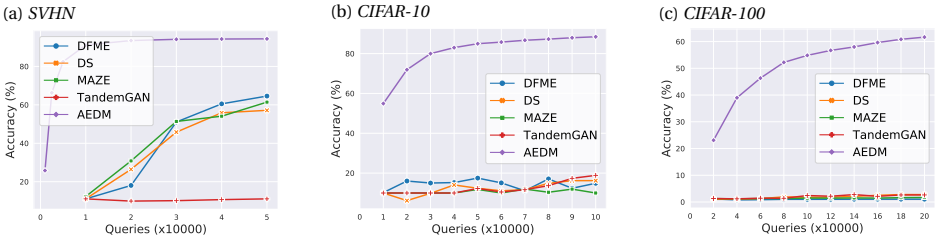


Figure 4.4: Substitute model accuracy of AEDM over the increasing number of queries used on *SVHN*, *CIFAR-10* and *CIFAR-100* datasets.

Table 4.4: The (Acc)uracy (%) of the substitute models over different number of clients (attackers) in Federated learning.

Dataset	Centralized Acc.	Federated Model Stealing 5 Clients	10 Clients	20 Clients
SVHN (20K Queries)	93.47	91.11	90.76	91.49
CIFAR-10 (0.1M Queries)	88.45	83.35	85.56	84.43

number of the queries, underlining the essential need of using semantically informative queries.

To compare by the number of queries explicitly, we maintain a consistent number of queries and track the accuracy of the substitute models as the queries increase in Figure 4.4. Specifically, we set track the accuracy up to 50K, 0.1M, and 0.2M queries for *SVHN*, *CIFAR-10*, and *CIFAR-100*, respectively. Figure 4.4 illustrates the superiority of AEDM in knowledge extraction through adversarial queries, demonstrating both effectiveness and efficiency. For simple tasks like *SVHN*, AEDM converges with approximately 30K queries, whereas for *CIFAR-10* and *CIFAR-100*, AEDM continues to improve, though it already significantly outperforms the baselines. This observation is in line with our findings in Table 4.3 that more complex tasks require a greater number of queries to thoroughly explore and extract knowledge from the target model.

It is also interesting to notice that the performance of the baseline methods varies depending on the task at hand. The primary distinction among these baseline methods lies in how they generate the noisy query data to mimic the training data space. For instance, DFME outperforms other baselines on *SVHN* but exhibits significant fluctuations on *CIFAR-10* and falls behind TandemGAN on *CIFAR-100*. This can be explained by the fact that single-generator data-free frameworks lack diversity in the query data space, which aligns with the findings of TandemGAM [46]. Thanks to the exploration and exploitation process facilitated by dual generators, TandemGAN performs well and consistently, especially on more complex classification tasks, while suffering from inefficiencies, particularly in simple tasks.

4.4.3. EXTRACTION UNDER FEDERATED SETTING

Given the fact that extremely large number of queries by a single adversary will cause easy detection by the target model, her we evaluate our AEDM stealing method under a federated settings where multiple federated attackers collusively launch the attack. The results on *SVHN* with 20K queries in total and *CIFAR-10* with 0.1M queries in total are summarized in Table 4.4. Note that the number of queries is equally split over all participating attackers (clients) and every attacker holds a diffusion model pre-trained by ImageNet. From Table 4.4, we see that on both datasets stealing under federated setting reaches similar but slightly lower substitute model accuracy than stealing by a single adversary. This result validates that distributing the stealing process to multiple clients can significantly reduce the number of per-client queries while maintaining similar knowledge extraction effectiveness. One interesting note from Table 4.4 is the substitute model accuracy over different number clients. The accuracy does not monotonically increasing or decreasing with more clients but the same total number of queries. Yet, among 5 to 20, 20-clients achieves the highest substitute model accuracy on *SVHN* while 10-clients works better on *CIFAR-10*. This finding indicates that more query partitions for stealing does not necessarily decrease the accuracy and the best number of clients for collaborative stealing can be specified on different task and data.

Table 4.5: The defence pass rate (DPR) with baseline extraction methods on the dataset Cifar-10 with the presence of the entropy-based defence.

Methods	DFME	MAZE	DS	TandemGAN	AEDM
DPR	0.060	0.021	0.210	0.164	**0.853**

4.4.4. EFFECTIVENESS AGAINST DEFENCE

The target model may employ defence mechanisms to detect the abnormal queries. This part we analyse our extraction effectiveness in the presence of stealing defences. One lightweight but effective approach is to detect the quality of received queries, in which low quality queries are determined as outliers and possible attackers [118]. The context of image quality in this chapter is the entropy of the query. We present the defence pass rate of each knowledge extraction method in Table 4.4. From the results, it is obvious that the baseline data-free stealing methods which generate noise query images, are easily identified as outliers and stopped, e.g., the highest DPR across all baselines is 21% by DS followed by TandemGAN 16%. Thanks to the diffusion model infusing semantics into the queries, more than 85% of the queries of AEDM bypass the defence.

As our AEDM generates adversarial knowledge extraction queries similar to real legitimate queries, current simple defence mechanisms are no longer effective against our attack. To enhance the defence one possible way left for future work is to leverage the image similarity between the query data and the training data, i.e., only allow in-distribution data for queries. However, this comes at a trade-off with additional computational overhead at the target model.

Table 4.6: The (Acc)uracy (%) of the substitute models over different network (Arc)hitecture for our method AEDM.

Dataset	Target Acc.	Substitute Arc. VGG-16	Substitute Arc. ResNet-18
CIFAR-10	96.59	88.45	**90.77**
CIFAR-100	78.71	61.66	**65.19**

4.4.5. SENSITIVITY OVER DIFFERENT SUBSTITUTE ARCHITECTURE

Given that the adversary is assumed to be unaware of the target model architecture, the substitute model can be selected from various neural network architectures. Therefore, in this section, we investigate the impact of choosing different neural network architectures on *CIFAR-10* and *CIFAR-100*. The findings are presented in Table 4.6. Here, the target model is ResNet-34, and we compare the knowledge extraction effectiveness of substitute models using VGG-16 and ResNet-18, with the same number of queries, i.e., 0.1M and 0.2M for two datasets respectively. According to the results, ResNet-18 demonstrates superior extraction accuracy, likely due to the fact that it belongs to the same neural network family as the target model. This outcome aligns with our expectations that similar neural network architectures exhibit better ability in knowledge transfer/extraction, and is in line with the findings from [46]. However, in this chapter, we employ VGG-16 for other experiments to show better versatility without prior knowledge of the target model, where VGG-16 achieves only slightly lower accuracy.

4.5. CONCLUSION

Existing data-free model stealing attacks benefit from broader applications due to not requiring real data, but suffer from extremely large number of queries to achieve high substitute model accuracy. In order to reduce the number of queries, this chapter proposes a novel adversarial knowledge extraction via steering diffusion models AEDM. Different from generator-based data-free approaches, AEDM generates synthetic data by diffusion model pretrained on publicly available data to generate semantically meaningful queries for the target model. To ensure diverse and realistic data queries, AEDM iteratively optimizes the diffusion model input latent variable to explore the large image space. We empirically demonstrated the effectiveness and efficiency of stealing by AEDM, in terms of higher substitute model accuracy with as little as 1/100 to 1/200 of the number of queries required by state-of-the-art baseline methods. Our extensive experiments show that AEDM also works against data-free query defence, as well as under federated setting, indicating the possibility of further reducing the per-client query number by collaborative stealing.

5

Analyzing Inference Probabilities for Effective Distillation of Generative Models

*Knowledge distillation has been applied on generative models, such as Variational Autoencoder (VAE) and Generative Adversarial Networks (GANs). To distill the knowledge, the synthetic outputs of a teacher generator are used to train a student model. While the dark knowledge, i.e., the probabilistic output, is well explored in distilling classifiers, little is known about the existence of an equivalent dark knowledge for generative models and its extractability. In this chapter, we derive the first kind of empirical risk bound for distilling generative models from a Bayesian perspective. Through our analysis, we show the existence of the dark knowledge for generative models, i.e., Bayes probability distribution of a synthetic output from a given input, which achieves lower empirical risk bound than merely using the synthetic output of the generators. Furthermore, we propose a **D**ark **K**nowledge based **D**istillation , DKtill, which trains the student generator based on the (approximate) dark knowledge. Our extensive evaluation on distilling VAE, conditional GANs, and translation GANs on Facades and CelebA datasets show that the FID of student generators trained by DKtill combining dark knowledge are lower than student generators trained only by the synthetic outputs by up to 42.66%, and 78.99%, respectively.*

5.1. INTRODUCTION

Generative models, such as Variational autoencoders (VAE) and generative adversarial networks (GANs), are increasingly applied to synthesize images. To practically deploy those models on edge devices, namely the generators, it is critical to distill/compress the models first due to the memory and computation constraints. Knowledge distillation via teacher and student models can effectively compress the machine learning models, especially for classification models [35, 6, 51]. The student model tries to imitate the (non-compressed) teacher model through the input and output pairs from the teacher model such that the same learning efficacy can be achieved via a smaller model. Dark knowledge [139] is shown particularly critical for the distillation quality of classifiers. Recently, some related studies [2, 122, 8, 122] focus on distilling the generator of GANs. This prior art empirically distills the teacher generator by directly leveraging the synthetic outputs. While the studies in [75, 134, 93] provide theoretical analysis for distilling classifiers, e.g., distillation bound, little is on the theoretical understanding of distilling generative models, for instance, the empirical risk bound and the existence of dark knowledge of the generators.

In this chapter, we rethink the generator distillation from a theoretical Bayesian perspective. We hypothesize the existence the "dark knowledge", which is embedded in the synthetic output, e.g., image, but not directly observable. To such an end, we model the synthetic outputs of a generator as a conditional Bayes probability distribution, $\mathbb{P}(y|x)$, where y is the synthetic output and x is the input of the generator. We derive two types of empirical risks for distillation: regular empirical risk using only synthetic inputs, and Bayes empirical risk using the proposed conditional probability distribution. We show that the variance of the Bayes empirical risk is lower than the variance of regular empirical risk, demonstrating better generalization capability. We thus refer to the knowledge of $\mathbb{P}(y|x)$ as dark knowledge. Our analysis can be applied on both probabilistic generative models, e.g., VAE, and non-probabilistic models, e.g., GANs. To derive such Bayes empirical risk bound for GANs, we approximate the conditional probability distribution and quantify its impact in the distillation bound.

Motivated by the effectiveness of such dark knowledge, we further propose the **D**ark **K**nowledge **D**istillation, DKtill, algorithm to train a student generator model through the (approximate) dark knowledge. Specifically, we try to minimizing the difference between teacher and student outputs by controlling the tensor of the second last layer of their networks, which is an approximated conditional Bayes probability. We extensively evaluate DKtill on distilling VAE, conditional GANs and translational GANs for Facades, and CelebA datasets. Our results show that the student generators from DKtill achieve lower FID than the ones trained on only synthetic images by up to 78.99% and a slightly higher FID than the ones from the teacher model by 17.77%.

Our key contributions for this work can be summarized as follows. *i)* Having Theorem 5.1 and Proposition 5.1 as the distillation empirical risk analysis to show the effectiveness of dark knowledge in distilling generative models. *ii)* Deriving Proposition 5.2 as the approximate distillation empirical risk analysis to capture the impact of approximating dark knowledge in distilling the GANs. *i)* Proposing DKtill which is a dark knowledge based distillation algorithm for training student generative models. *iv)* Achieving higher distillation quality of DKtill on three different generative models and 3 different

datasets, compared to distillation algorithms which do not use the proposed (approximated) dark knowledge.

5.2. PRELIMINARY

We consider general generative models for synthesizing images, including probabilistic generators and non-probabilistic generators. In existing generator distillation works [2, 122, 8] the basic idea is as follows. First, they provide the same inputs x into a trained teacher model $\mathscr{T}$ and a student model f where x is the generator input. x has different formats in different generative models. For vanilla `GANs` or `VAE`, x is a noise vector. For conditional GANs, x is a noise and a conditional label. x can also be a picture in image translation `GANs`. Second, the distance between the two model outputs $\mathscr{T}(x)$ and $f(x)$ is minimized. In this way, the student f can mimic the input-output mapping of $\mathscr{T}$ and extract the knowledge from the teacher's synthetic image outputs $\{\mathscr{T}(x)\}$.

In classifier distillation [35], the final predicted labels of a teacher are not sufficient to train a student. The logits or the softmax outputs of the teacher model are needed to train the student so as to capture the underlying "dark knowledge", which is the probability the teacher assigns to "wrong" labels. Some basic settings and terms of generator distillation used in this chapter are introduced in the following.

Synthetic images. In generator distillation, we input x into a trained teacher[1] $\mathscr{T}$ and obtain the corresponding synthetic output $y = \mathscr{T}(x)$. After N inferences, a training dataset $D = \{(x_n, y_n)\}_{n=1}^{N} \sim \mathbb{P}^N$ is obtained to train the student model f, where x_n is an input sample and $y_n = \mathscr{T}(x_n)$ is the corresponding synthetic image. To simplify the notation, without loss of generality, y_n is a single channel image and $y_n^{ij} \in \{0, ..., C\}$ is a pixel of the image, where C indicates the number of possible colors[2]. Let $i \in [1, ..., I]$ and $j \in [1, ..., J]$ be the pixel indexes of an image of width I and height J.

Distillation optimization. To distill the knowledge from the teacher model $\mathscr{T}$, a student generator which can be defined as $f : \mathscr{X} \to \mathbb{R}^{I \times J \times C}$ is trained to minimize the following risk:

$$R(f) = \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{P}} [\ell(y, f(x))]. \tag{5.1}$$

According to the definition of f, the mapping result $f(x)$ is an $I \times J \times C$ tensor, and $f^{ij}(x) \in \mathbb{R}^C$ corresponds to a pixel. The vector $f^{ij}(x)$ represents the weights of taking different colors in the pixel. For each training pair (x, y) collected from $\mathscr{T}$, the loss of $f(x)$ takes the form $\ell(y, f(x)) = \sum_i \sum_j \ell^{ij}(y^{ij}, f^{ij}(x))$. The term $\ell^{ij}(y^{ij}, f^{ij}(x))$ is the loss value of predicting $f^{ij}(x)$ when the true pixel color is y^{ij}. To do distillation, the student needs to mimic the teacher's outputs. Therefore $\ell^{ij}(y^{ij}, f^{ij}(x))$ should be a loss that minimizes the distance between $y = \mathscr{T}(x)$ and $f(x)$, e.g., the softmax cross-entropy loss.

Distillation using only synthetic images. Having different definitions of $\ell(y, f(x))$ in $R(f)$ leads to different distillation methods. A common way in most existing works is to define $\ell(y, f(x))$ directly using the synthetic output images $\{y\}$ from the teacher. Given $y = \mathscr{T}(x)$, the basic idea for training the student f is to maximize the element $f^{ij,y^{ij}}(x)$ in the output probabilities $f^{ij}(x)$ of the student for each pixel i, j.

[1] We interchangeably use teacher or target model/generator.
[2] The analysis of this chapter can be straightforwardly extended to three channel images.

5.3. Theoretical Analysis of Dark Knowledge in Distilling the Generator

This section introduces the novel concept of dark knowledge in distilling generators and demonstrates that harvesting this dark knowledge can train a student generator f which generalizes better. In generator distillation, the goal is to transfer the knowledge of the teacher $\mathcal{T}$ as much as possible to a student f. We use the risk (generalization error) of the student f shown in Eq. 5.1 to evaluate the quality of the distilled student model. The risk describes the generalization ability of f on unseen new data sampled from $\mathbb{P}$ other than the observed training dataset $D = \{(x_n, y_n)\}_{n=1}^N \sim \mathbb{P}^N$. A lower risk value means having a better distilled f. In the following, we first introduce the concept of dark knowledge in generative models and then derive the distillation empirical risks.

5.3.1. Dark Knowledge of Generators

From a Bayesian viewpoint, each synthetic image y from the teacher is generated by a conditional distribution $\mathbb{P}(y|x)$ where the distribution $\mathbb{P}$ is potentially determined by the trained teacher generator $\mathcal{T}$. Obviously, the knowledge of $\mathbb{P}(y|x)$ cannot be fully extracted from the teacher's synthetic image output. A synthetic image y is only a sample drawn from the distribution $\mathbb{P}(y|x)$. Thus, in generator distillation we call the underlying distribution $\mathbb{P}(y|x)$ the "dark knowledge" of the teacher $\mathcal{T}$. The next section demonstrates that using the dark knowledge $\mathbb{P}(y|x)$ to define $\ell(y, f(x))$ so as to have a precise empirical estimation of $R(f)$ can facilitate the training of a student in generator distillation.

We consider two ways of distilling teacher generator: i) by solely the synthetic image outputs of $\mathcal{T}$, and ii) by using the underlying dark knowledge $\mathbb{P}(y|x)$ of $\mathcal{T}$. We analyze the generalization error of the student f under both distillation ways by comparing their (empirical) risks. According to the introduction in Section 5.2, in generator distillation, $R(f)$ should be minimized by the algorithm where Eq. 5.1 shows the general definition of $R(f)$. However, calculating the exact value of $R(f)$ is intractable because $\mathbb{P}$ is unknown or it has no explicit expression. Hence empirical risk definitions are required to approximate $R(f)$. The aforementioned two distillation ways leverage two different corresponding empirical risk definitions. These will be illustrated in detail in the following.

5.3.2. Distillation Empirical Risk

Distillation with sole synthetic images. First, we introduce the distillation empirical risk of using $\mathcal{T}$'s synthetic image outputs. In this way of distillation, the dataset $D = \{(x_n, y_n)\}_{n=1}^N$ collected from the teacher $\mathcal{T}$ is used to train the student f. We have the following distillation empirical risk definition to approximate $R(f)$:

$$\hat{R}(f; D) = \frac{1}{N} \sum_{n\in[N]} \sum_i \sum_j \ell^{ij}(y_n^{ij}, f^{ij}(x_n)) = \frac{1}{N} \sum_{n\in[N]} \sum_i \sum_j e_{y_n^{ij}}^\top \ell^{ij}(f^{ij}(x_n)), \tag{5.2}$$

where $e_{y_n^{ij}}$ is the one-hot vector encoding of the color value of y_n^{ij}, and $\ell^{ij}(f^{ij}(x)) = [\ell^{ij}(1, f^{ij}(x)), ..., \ell^{ij}(C, f^{ij}(x))] \in \mathbb{R}^C$ is a vector of loss values for each possible color of the pixel with index i, j. As shown in Eq. 5.2, this empirical risk definition only needs the synthetic image y_n from $\mathcal{T}$ to decide the value of $e_{y_n^{ij}}$. Then we can calculate $\hat{R}(f; D)$ with no additional information from $\mathcal{T}$.

Distillation with dark knowledge. Here we assume that besides the final synthetic image output y we also get the conditional probability $\mathbb{P}(y|x)$ (the dark knowledge) of a given input x from $\mathscr{T}$. Consequently, we can define another distillation empirical risk $\hat{R}_\alpha(f,D)$ to approximate $R(f)$:

$$\hat{R}_\alpha(f,D) = \frac{1}{N}\sum_{n\in[N]}\sum_i\sum_j p^{ij}(x_n)^\top \ell^{ij}((f^{ij}(x_n)) \tag{5.3}$$

where the conditional probability of pixel y^{ij} color is denoted by

$$p^{ij}(x) = [\mathbb{P}(y^{ij}|x)]_{y_{ij}\in[C]} \tag{5.4}$$

Connection to the empirical risk. $\hat{R}(f;D)$ (Eq. 5.2) can be seen as an approximation of $\hat{R}_\alpha(f,D)$ (Eq. 5.3). Considering that: $R(f) = \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{P}}[\ell(y,f(x))] = \mathop{\mathbb{E}}_{x}[\mathop{\mathbb{E}}_{y|x}[\ell(y,f(x))]]$, we know that $\hat{R}_\alpha(f,D)$ is an empirical estimate of $\mathop{\mathbb{E}}_{x}[\mathop{\mathbb{E}}_{y|x}[\ell(y,f(x))]]$ over the random variable x. If we further use the one-hot encoding $e^\top_{y_n^{ij}}$ to approximate $p^{ij}(x_n)$ in Eq. 5.3, we have Eq. 5.2. To distinguish these two empirical risks, we call $\hat{R}(f;D)$ (Eq. 5.2) as the *distillation empirical risk with synthetic images* and $\hat{R}_\alpha(f,D)$ (Eq. 5.3) as the *distillation empirical risk with dark knowledge*.

5

5.3.3. GENERALIZATION OF THE STUDENT GENERATOR

Given a teacher $\mathscr{T}$, a student model f can be trained by optimizing the distillation empirical risk with synthetic images (Eq. 5.2) or the distillation empirical risk with dark knowledge (Eq. 5.3). In this section, we analyze the generalization ability of the student f under the two different distillation empirical risk bounds. In the following, we show that a student generator trained with Eq. 5.3 is expected to generalise better than trained under Eq. 5.2, which means using the dark knowledge can facilitate generator distillation. Although both Eq. 5.2 and Eq. 5.3 are unbiased estimates of $R(f)$, the variances of Eq. 5.2 and Eq. 5.3 over the observed training dataset D are different. This is formally demonstrated by Theorem 5.1. The proof of Theorem 5.1 is shown in the appendix [39] of the corresponding technical paper.

Theorem 5.1 *Let D be a training dataset sampled from $\mathbb{P}^N$. $\mathbb{V}$ represents the variance of a random variable. For any fixed hypothesis $f:\mathscr{X}\to\mathbb{R}^{I\times J\times C}$,*

$$\mathbb{V}_{D\sim\mathbb{P}^N}\left[\hat{R}_\alpha(f;D)\right] \le \mathbb{V}_{D\sim\mathbb{P}^N}[\hat{R}(f;D)]$$

The two variances in Theorem 5.1 equal to each other when $p^{ij}(x)$ is concentrated on one single color and the probability on all other colors is zero. However, this case rarely happens. The benefit of having lower variance is that the student f generalises better as shown in the following. Applying Theorem 6 of [79], we have the following bound for the distillation empirical risk with dark knowledge.

Proposition 5.1 *Let $D\sim\mathbb{P}^N$ and $\mathscr{F}$ be a class of hypotheses $f:\mathscr{X}\to\mathbb{R}^{I\times J\times C}$ with induced class $\mathscr{H}\subset[0,1]^{\mathscr{X}}$ of $h(x)=\sum_i\sum_j p^{ij}(x)^\top\ell^{ij}((f^{ij}(x))$. Suppose $\mathscr{H}$ has uniform covering*

number $\mathcal{N}_\infty$. *For any* $\delta \in (0,1)$ *and set* $\mathcal{M}_N = \mathcal{N}_\infty(1/N, \mathcal{H}, 2N)$. *Then with probability at least* $1-\delta$ *over* D *we have:*

$$R(f) \leq \hat{R}_\alpha(f;D) + \mathcal{O}\left(\sqrt{\mathbb{V}_N(f)/N} \cdot \sqrt{\log(\mathcal{M}_N/\delta)} + \log(\mathcal{M}_N/\delta)/N\right)$$

where $\mathbb{V}_N(f)$ *is the empirical variance of* $\{h(x_n)\}_{n=1}^N$.

Note that using the same procedure we can derive a similar bound for the distillation empirical risk with synthetic images (Eq. 5.2). Considering Theorem 5.1 and the two bounds, we know that the bound for Eq. 5.3 has lower variance penalty. Increasing the training dataset size N (number of queries on $\mathcal{T}$), Eq. 5.3 has a risk bound with a better rate of convergence. Thus, a student model trained by minimizing the distillation empirical risk with dark knowledge generalises better compared to one using the distillation empirical risk with synthetic images. We conclude that a student generator f trained by the dark knowledge $\mathbb{P}(y|x)$ is better than using sole synthetic image outputs $\{y\}$ from $\mathcal{T}$.

5

5.3.4. IMPACT OF PROBABILITY APPROXIMATION

In the aforementioned analysis, we showed that having the dark knowledge $\mathbb{P}(y|x)$ from the teacher $\mathcal{T}$ allows for a more precise empirical approximation of $R(f)$. Thus it benefits the training of the student f in generator distillation. However, as mentioned before, doing distillation with dark knowledge requires the conditional probability $\mathbb{P}(y|x)$ for any given input x. In ideal cases, the intermediate layer output of $\mathcal{T}$, e.g., the second last layer output in `VAE`, can provide $\mathbb{P}(y|x)$. Then, it can be used to train the student model. Unfortunately, for some teacher models $\mathcal{T}$, e.g., `GANs`, the intermediate layer output cannot directly show $\mathbb{P}(y|x)$. We refer to the generators that can provide $\mathbb{P}(y|x)$ in the intermediate layer output as *probabilistic generators* and the generators that cannot show the probability as *non-probabilistic generators*. To distill the dark knowledge from non-probabilistic generators, it is required to approximate $\mathbb{P}(y|x)$. Let $\tilde{\mathbb{P}}(y|x)$ be an approximated probability of $\mathbb{P}(y|x)$. In this section, we study the impact on the student generalization error using such an approximated distribution to do distillation. Using the approximated probability to train the student f, referring to Eq. 5.3 we have the following distillation empirical risk:

$$\tilde{R}_\alpha(f,D) = \frac{1}{N} \sum_{n\in[N]} \sum_i \sum_j \tilde{p}^{ij}(x_n)^\top \ell^{ij}((f^{ij}(x_n)), \tag{5.5}$$

where $\tilde{p}^{ij}(x_n)$ is the approximation of $p^{ij}(x_n)$. According to Eq. 5.1 and Eq. 5.4 we can rewrite the population risk $R(f)$ as:

$$R(f) = \mathop{\mathbb{E}}_x[\mathop{\mathbb{E}}_{y|x}[\ell(y,f(x))]] = \mathop{\mathbb{E}}_x[\sum_i \sum_j p^{ij}(x)^\top \ell^{ij}(f^{ij}(x))]. \tag{5.6}$$

The following Proposition 5.2 reveals the connection between the approximation of $\mathbb{P}(y|x)$ and the generalization error of f. The proof of Proposition 5.2 is shown in the appendix [39] of the corresponding technical paper of this chapter.

Proposition 5.2 *If the loss ℓ is bounded, we train the student model by minimizing the empirical risk shown in Eq. 5.5. For any hypothesis $f : \mathcal{X} \to \mathbb{R}^{I\times J\times C}$,*

$$\mathbb{E}\left[(\tilde{R}_\alpha(f;D) - R(f))^2\right] \leq \frac{1}{N}\cdot\mathbb{V}[\sum_i\sum_j \tilde{p}^{ij}(x)^\top \ell^{ij}((f^{ij}(x))] + \mathcal{O}(\mathbb{E}[\sum_i\sum_j \left\|\tilde{p}^{ij}(x) - p^{ij}(x)\right\|_2])^2. \tag{5.7}$$

On the right side of Eq. (5.7), when N is big, the second term dominates the upper bound of the gap $\tilde{R}(f;D) - R(f)$. That means minimizing the distance between the approximated probability $\tilde{\mathbb{P}}(y|x)$ and the ground truth probability $\mathbb{P}(y|x)$ yields a tighter upper bound of the risk gap $\tilde{R}(f;D) - R(f)$. Hence, a tighter approximation leads to a better student model.

5.4. DKtill: Extracting Dark Knowledge for Training Student Generator

In the previous section, we theoretically demonstrate that using the underlying dark knowledge of a teacher $\mathcal{T}$ can improve the generator distillation and train a student f that generalizes better. However, to make use of the underlying dark knowledge in distillation, we first need to know how to extract and use the dark knowledge from a teacher generator $\mathcal{T}$. In this section, we propose two methods to extract the dark knowledge based on the class of the generator: one for probabilistic generators and one for non-probabilistic generators. For verifying the effectiveness of the extracted dark knowledge, we propose a generator distillation algorithm `DKtill`. When distilling $\mathcal{T}$, `DKtill` makes the student f to mimic the extracted (approximated) $\mathbb{P}(y|x)$ besides the synthetic images $y = \mathcal{T}(x)$ (see the supplementary for more implementation details).

5.4.1. Extracting from probabilistic generators

Probabilistic generators, e.g., variational auto-encoder (`VAE`) [60], commonly assume the existence of latent variables. More in detail, they assume that a synthetic image y is generated by some random process involving some latent random variables. In such generators, to do distillation with dark knowledge, the conditional probability $\mathbb{P}(y|x)$ can be calculated by some middle layer outputs of the teacher generator. In the following, we take `VAE` as an example for dark knowledge extraction in probabilistic generators. The training method of `VAE` is a kind of variational inference that uses $q(x|y)$ to approximate the intractable true posterior $p(x|y)$ and maximize the evidence lower bound. For `VAE` distillation, we focus on its trained decoder. In the decoder, the synthetic image is produced by the conditional distribution $\mathbb{P}(y|x) = \mathcal{N}(y;\mu,\sigma^2 I)$, where the distribution parameters μ and σ are the middle layer outputs of the decoder neural network. Thus, given any input x, we can obtain $\mathbb{P}(y|x)$ by getting the output value of μ and σ from the middle layers.

5.4.2. Extracting from non-probabilistic generators

Non-probabilistic generators, e.g., `GANs`, do not assume any probability relationship between output image y and latent random variables. y is directly produced by a neural network mapping $\mathcal{T}(x)$, where $\mathcal{T}$ is the teacher generator. Different from probabilistic generators, in non-probabilistic generators we cannot derive the knowledge of $\mathbb{P}(y|x)$

by some middle layer outputs of $\mathcal{T}$. In the following, we take GANs as an example to show extracting dark knowledge $\mathbb{P}(y|x)$ from non-probabilistic generators. Given a GANs teacher generator $\mathcal{T}$, we let g_α be the second last layer and g_β be the last layer of $\mathcal{T}$. Given an input x into the teacher generator, the intermediate output of g_α is represented by α. Then the corresponding synthetic image can be represented as $y = g_\beta(\alpha)$. To get the dark knowledge, we can apply a differentiable probabilistic network (e.g., MLP) g_γ to replace the original last layer g_β. The input of g_γ is g_α's output, α and the output of g_γ is a tensor that takes the shape corresponding to the underlying Bayes probabilities. The shape of $g_\gamma(\alpha)$ is decided by I, J and C where I and J are decided by the synthetic image size and C depends on the color range. The final synthetic image is now sampled from the tensor $g_\gamma(\alpha)$. If $\mathcal{T}$ is untrained, we can train it from scratch using this new architecture. Thus, after the training, we can easily get $\mathbb{P}(y|x)$ from $g_\gamma(\alpha)$. If the given $\mathcal{T}$ is pre-trained, we can just fine-tune the parameters of g_γ using the optimization objective, $\text{argmin}_{g_\gamma} ||g_\beta(\alpha) - \hat{y}||_1$, where $\hat{y} \sim Discrete(g_\gamma(\alpha))$ is the synthetic image sampled from $g_\gamma(\alpha)$.

5

5.5. EMPIRICAL ILLUSTRATION

5.5.1. SETTING

Datasets: we consider the datasets, Facades [50] and CelebA [74]. In the case of conditional generation, we focus on the gender class, i.e., male and female, in the CelebA dataset. We downsampled the CelebA images from originally 178 × 218 to 64 × 64 pixels. The input images size of Facades is 256 × 256.

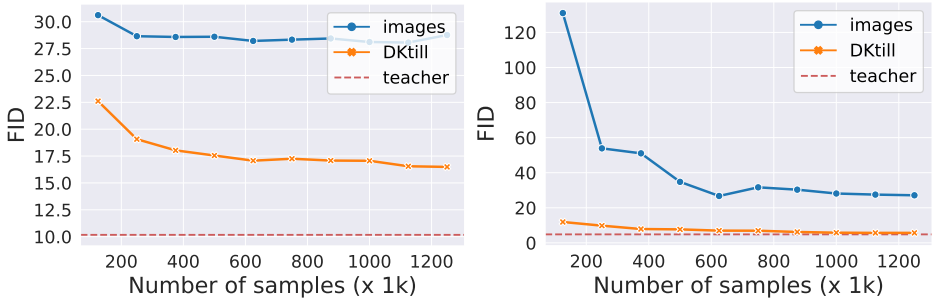
Networks: the network structures of all generators are based on convolutional neural networks [60, 96]. For VAE, we compress a 27.2MB teacher into a 2.4MB student. As for conditional GANs, a 2.8MB student is distilled from a 14.4MB teacher. In the image translation experiment, Pix2pix is compressed from 217.8MB into 14.0MB.

Distillation process and baseline: for both DKtill and the baseline (abbreviated as "image" [8]), we train the student generators, by minimizing the distance of the generated images to the teacher from the same random inputs. However, instead of only using generated images (baseline), DKtill also uses in parallel the information of dark knowledge (underlying distribution) for loss minimization. Note that although the baseline distillation is originally designed for Image Translation, we adopt it for VAE and conditional GANs.

Evaluation metrics: we use Fréchet Inception Distance (FID) to evaluate the quality of the images produced by the distilled student model. Lower values of FID indicate higher quality of generated data.

5.5.2. DISTILLING PROBABILISTIC GENERATORS

Here, we distill VAE (the teacher generator) using the baseline and DKtill. Figure 5.1 shows the comparison results based on FID for VAE and conditional GANs (Figure 5.1a). We also illustrate the FID of images generated by the teacher VAE, i.e., the horizontal line in Figure 5.1a. The FID value decreases during the training, from 30.61 to 28.76, and 22.61 to 16.49, for baseline and DKtill respectively, demonstrating lower distance to ground truth images. When looking at the final FID (after 128K examples), DKtill



(a) VAE (b) conditional GANs

Figure 5.1: The Fréchet Inception Distance (FID) value versus the number of samples of distilling the generative models, VAE (a) and conditional GANs (b), on the dataset CelebA.

is 42.66% lower than the baseline image and 62.14% higher than the teacher generator. Here FID for the teacher VAE is 10.17, which is much better than both students. This is within our expectation as the teacher model is 10X the size of the students. In general, we can see that DKtill, using the dark knowledge provided by the intermediate output of teacher VAE, can significantly improve the quality of distillation. As the number of examples increases, the student learns the teacher's mapping and distribution better. Using dark knowledge can achieve lower FID and thus it can distill better.

5.5.3. DISTILLING NON-PROBABILISTIC GENERATORS

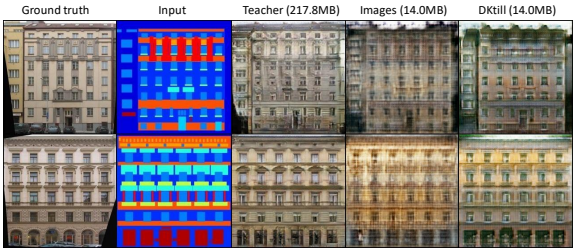


Figure 5.2: Knowledge distillation of Pix2pix image translation GANs on the dataset Facades using our method DKtill and the baseline.

Here we implement two tasks for non-probabilistic generators: the conditional GANs and image translation as the baseline [8]. Figure 5.1b evaluates the distillation on conditional GANs.We can see that FID shares the same trend as distilling the probabilistic generator VAE. Thus, our proposed method DKtill is able to effectively extract knowledge from non-probabilistic teacher generators too, given the approximate nature of dark knowledge. The reason why FID of DKtill gets much closer to the teacher than Figure 5.1a is that the teacher model is only 5X bigger in parameter size than the students. When looking at the final FID (after 128K examples), DKtill is 78.99% lower than the baseline image and only 17.77% higher than the teacher generator.

Let us zoom into the distillation results of the image translation task in Figure 5.2. The goal is to train the translation GANs so that given input (as the "input" in visualization), the Pix2pix network approximately maps it into the ground truth image. From the results, we observe that even with 6% size of the teacher model (FID: 35), DKtill is able to distill the image translation generator at high quality (FID: 35) with dark knowledge, especially compared with the baseline (FID: 37.88). By distillation, the floating point operations per second (FLOPs) on one input image is reduced from 6.06G (Teacher) to 0.83G (Student).

5.5.4. SMALL GENERATORS THROUGH DKTILL

In addition to achieving FID similar to the teacher model, we show another advantage of exploring dark knowledge, i.e., smaller student generator. We first solicit the synthetic images generated by the following model in Figure 5.3: the ground truth, teacher VAE, 27.2MB student VAE trained by baseline, 27.2MB student VAE from DKtill, and 2.4MB student VAE from DKtill. Without any surprise, DKtill achieves better distillation quality when using bigger networks, comparing Figure 5.3d and Figure 5.3e. Another observation is that, smaller student (2.4MB) VAE trained by DKtill, achieve better image quality than baseline with 27.2MB, comparing Figure 5.3c and Figure 5.3e.

The conditional GANs result is also presented in Figure 5.4 (on gender class, i.e., male and female for the left 3 and right 3 pictures). Note that here we do not show the corresponding real images of the synthetic ones since conditional GANs do not have a latent code encoder as VAE. Thus, having the latent code of a real image for reproducing some corresponding synthetic ones is difficult. Given the same small network with 2.8MB for DKtill and the baseline, DKtill shows better generated image quality than baseline. These results again prove the existence of dark knowledge for generators and its benefit for distilling generators.

5.6. RELATED WORK

Knowledge distillation [35, 51, 73, 54], which transfers the knowledge from a big network to a small one, enables the light-weight deep learning. Generally, knowledge distillation is studied on classifiers for model compression, e.g, [35] compresses the knowledge of an ensemble into a single model which is much easier to deploy. Such techniques are used to train surrogate models [117, 64, 140], even without the necessity of knowing the target model parameters.

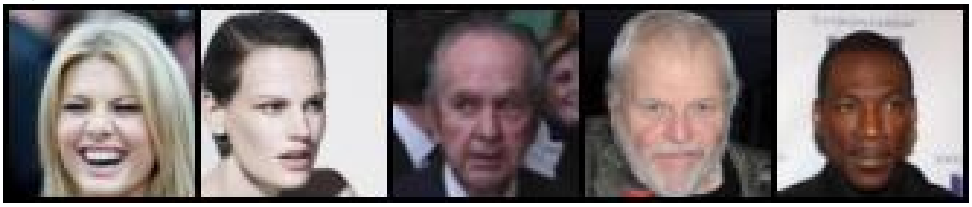
Besides, knowledge distillation has also been applied on GANs [8, 122, 8, 122]. KDGAN simultaneously optimizes the distillation and adversarial losses between a classifier, a teacher, and a discriminator, to learn the true data distribution at the equilibrium [122]. To further improve the performance, [8] include a student discriminator to measure the distances between real data, and the synthetic data generated by student and teacher generators. Recently, there are theoretical analyses on knowledge distillation for classifiers [75, 134, 93, 87, 139, 52]. On the one hand, [87] first provides the theoretical analysis of self-distillation, fitting a nonlinear function on Hilbert space and L2 regularization. This analysis sheds light on the relation between self-distillation rounds and (under-)over- fitting. On the other hand, based on neural tangent kernel, [52] provides a transfer

risk bound for the linearized model of the wide neural networks, revealing the impact of soft (hard) labels for (im)perfect teachers according to the designed data inefficiency metric. Furthermore, [139] explores the bias-variance trade-off brought by distillation with soft labels. According to their analysis, novel weighted soft labels are inspired to help the network adaptively handle the trade-off. However, none of the existing work provides a generalization analysis on generators.

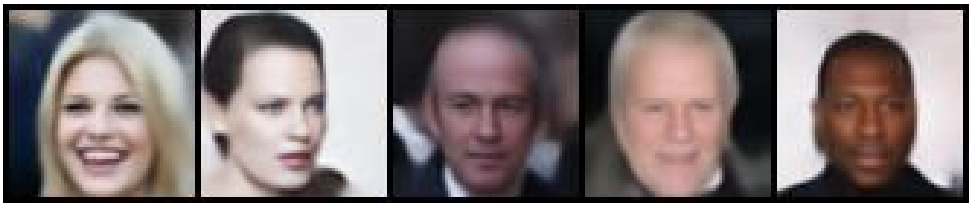
5.7. CONCLUSION

In this chapter, we model the knowledge distillation for generative models from a Bayesian perspective, identifying dark knowledge and its influence on the generalization ability of student models, i.e., lower empirical risk. Furthermore, we propose a dark knowledge based distillation optimization, `DKtill`, to train student generators on both non-probability-based and probability-based generative models. Evaluation results on three datasets across different scenarios show that synthetic images from `DKtill` achieve lower FID by up to 78.99%, in contrast to using images only. `DKtill` also generates images of similar quality as the teacher model, using smaller and more compact generator networks.

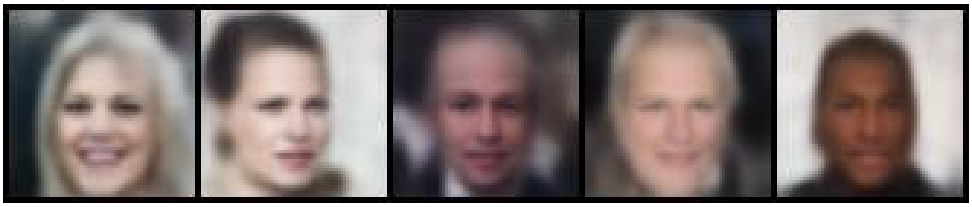
(a) ground truth



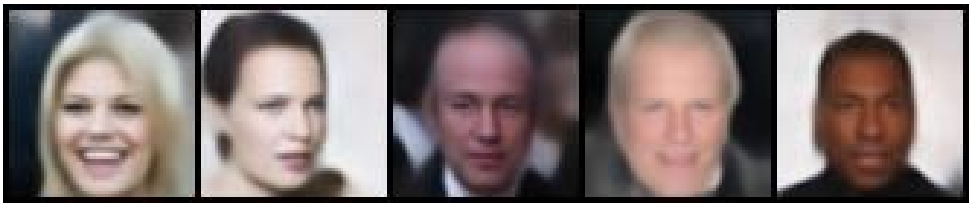
(b) teacher (27.2MB)



(c) images (27.2MB)



(d) `DKtill` (27.2MB)



(e) `DKtill` (2.4MB)

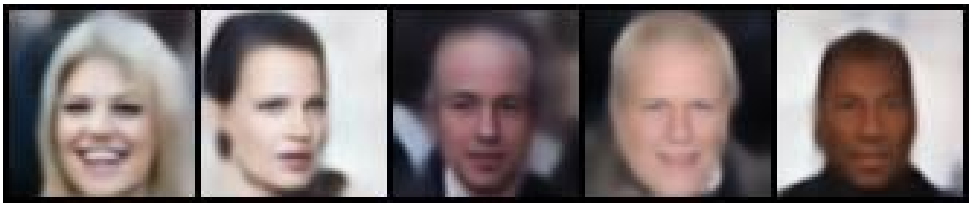


Figure 5.3: Distilling the generative model, `VAE`, on the dataset `CelebA` using our method `DKtill` (d & e) and the baseline (c) which trains the student model only based on the output images from the teacher (b).

(a) teacher (14.4MB)

(b) images (2.8MB)

(c) DKtill (2.8MB)

Figure 5.4: Distilling the generative model, Conditional GANs, on the dataset CelebA using our method DKtill (c) and the baseline (b) which trains the student model only based on the output images from the teacher (a).

6

EFFICIENT DISTILLATION OF DIFFUSION MODELS FOR ACCELERATED SAMPLING

While diffusion models effectively generate remarkable synthetic images, a key limitation is the inference inefficiency, requiring numerous sampling steps. To accelerate inference and maintain high-quality synthesis, teacher-student distillation is applied to compress the diffusion models in a progressive and binary manner by retraining, e.g., reducing the 1024-step model to a 128-step model in 3 folds. In this chapter, we propose a single-fold distillation algorithm, BiLA *, which can flexibly compress the teacher diffusion model into a student model of any desired step, based on reparameterization of the intermediate inputs from the teacher model. To train the student diffusion, we minimize not only the output distance but also the distribution of the hidden variables between the teacher and student model. Extensive experiments on four datasets demonstrate that our student model trained by the proposed* BiLA *is able to sample high-quality data with steps reduced to less than 1%, thus, trading off inference time. Our remarkable performance highlights that* BiLA *effectively transfers knowledge in single-fold distillation, achieving semantic consistency and meaningful image interpolation.*

6.1. INTRODUCTION

Diffusion models [111, 37, 36] have emerged as generative models for images of exceptionally high quality without the necessity of conducting adversarial training.A diffusion model constitutes a Markov chain of forward steps of slowly adding random noise to data, followed by a reverse denoising process that gradually reconstructs the data from the noise via trained neural networks. These underlying networks typically use the UNet architecture to better connect the forward steps to the corresponding denoising step. However, such models require large numbers of sampling steps, e.g., 1000 in DDPM [36], which leads to high sampling/inference[1] times, limiting their applications in latency-sensitive applications.

Prior art explored diverse directions to reduce the sampling time of diffusion models and maintain the image synthesis quality. One approach is to reduce the computing complexity by compressing the UNet [69, 135] and leverage the acceleration technologies of modern GPUs. Another approach is to reduce the number of sampling steps, i.e., the required UNet inferences. [109] skips intermediate steps by generalizing the original Markovian process via a class of non-Markovian diffusion steps. These two approaches do not change the original training procedure. Differently, progressive distillation [100, 84] introduces a teacher-student framework to reduce a trained teacher diffusion model of T steps into a student diffusion of T' steps, where $T' \ll T$, via multiple binary foldings by retraining. For instance, distilling a 1024-step diffusion model into a 128-step model needs first to train a 512-step intermediate model, then a 256-step, to finally arrive to the 128-step model. The distillation objective is to minimize the output differences between the teacher and student models. Progressive distillation better maintains the synthesis quality than step-skipping, but it incurs high distillation time and must comply with specific values of T and T' due to progressive halvings.

Our objective is to design an effective distillation method that achieves high quality in (any) small sampling step and concurrently incurs low distillation time. We propose BILA , a single-fold distillation framework able to reduce a T-step teacher diffusion model into a T'-step student diffusion model in a single fold. Thanks to its single-fold nature, BILA offers the flexibility to distill the teacher model into a student with any number of steps. To such an end, we first define a new student model, which can extract knowledge of the teacher diffusion model, by an arbitrary steps sub-sequence. Our forward process definition solves the challenge of aligning the variables of teacher and student models, enabling flexible single-fold distillation. Secondly, when training the reverse denoise process of the student model, we minimize not only the difference in the model outputs but also in the hidden variables at each step to better preserve the image synthesis quality.

To demonstrate the effectiveness, we evaluate BILA against sampling-skip [109] and progressive distillation [100] on CIFAR-10, CelebA, LSUN-Church, LSUN- Bedroom image datasets and 2D Swiss Roll tabular dataset. We compare their synthesis quality in terms of Fréchet Inception Distance (FID) [34] of distilling a 1024-step diffusion model into 128-step and 16-step models. BILA achieves the lowest FID as well as the best perceptual quality, also with flexibility on the numerical relation between the teacher and

[1] We interchangly use sampling and inference time.

the student, i.e., works for both 1024 to 128 or 100 steps. Further, the distillation effectiveness is validated on semantic input-output consistency and image interpolation.

We summarize the contributions of this chapter as follows:

- We propose a novel single-fold distillation algorithm, BILA , which can agilely compress teacher diffusion into a student diffusion model of any step in one fold.
- We define a new forward process for student diffusion, which aligns and approximates student and teacher Markovian variables, enabling flexible single-fold distillation.
- We design effective training for student diffusion by minimizing the difference of output and hidden variables with respect to the teacher diffusion.
- We experimentally demonstrate superiority of BILA in achieving high-quality sampling data by less than 1% number of steps.

6.2. SINGLE-FOLD DISTILLATION

We rethink knowledge distillation of diffusion models to reduce the number of sampling steps by proposing single-fold distillation. Instead of progressive multiple folds [100], which introduces distortion and costs extra training effort at each fold, BILA directly distills the teacher model to a student model with a given number of steps in a single fold. One crucial challenge is the alignment from the teacher's to the student's hidden variables, as the Markov chain is defined by every two consecutive variables but the student has much fewer steps.

We first introduce the preliminaries of a DDPM model used as a teacher model, including the definition of the forward/reverse process and the training objective. Then we present BILA which defines the forward and reverse process of the student by aligning and matching the hidden variables of the teacher. Finally, we design the distillation algorithm for deployment, which shows the training procedure of the student accordingly. For ease of presentation, we set the number of steps in the student model as a divisor of the number of steps in the teacher model, but the method is valid for an arbitrary number of student steps, i.e. fractional teacher/student step ratios.

6.2.1. PRELIMINARY

DDPM is composed of a forward (noising) and a reverse (denoising) process, through T steps. Its objective is to learn the denoising process via a given forward process.

Reverse process: The optimization objective of diffusion models [108] is derived by variational inference. Given the observed data $\boldsymbol{x}_0$, the diffusion model is a probabilistic model which specifies the joint distribution $p_\theta(\boldsymbol{x}_{0:T})$, where $\boldsymbol{x}_1, ..., \boldsymbol{x}_T$ are latent variables with the same dimensions as $\boldsymbol{x}_0$, and θ are learnable model parameters. $p_\theta(\boldsymbol{x}_{0:T})$ is a Markov chain that samples from $\boldsymbol{x}_T$ to $\boldsymbol{x}_0$, $p_\theta(\boldsymbol{x}_{0:T}) := p_\theta(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$. DDPM assumes that $p_\theta(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; \boldsymbol{0}, \boldsymbol{I})$ and

$$p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \sigma_t^2 \boldsymbol{I}\right),$$

where $\sigma_t \in \mathbb{R}_{\geq 0}$. $p_\theta(\boldsymbol{x}_{0:T})$ is called the *reverse process*. It gradually denoises a noise $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

Forward process: To derive a lower bound on the log likelihood of the observed data, diffusion models introduce the approximate posterior $q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)$ which is a Markov chain, $q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0) := \prod_{t=1}^{T} q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, that samples from $\boldsymbol{x}_1$ to $\boldsymbol{x}_T$. Then the log data likelihood can be decomposed as

$$\mathbb{E}\left[\log p_\theta(\boldsymbol{x}_0)\right] = \mathbb{E}_q\left[\log \frac{p_\theta(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\right] + D_{\mathrm{KL}}\left(q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)\right).$$

Thus, we have the lower bound $\mathbb{E}\left[\log p_\theta(\boldsymbol{x}_0)\right] \geq \mathbb{E}_q\left[\log \frac{p_\theta(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\right]$. When training the diffusion model, to maximize $\mathbb{E}\left[\log p_\theta(\boldsymbol{x}_0)\right]$, the parameters θ are learned to minimize the negative evidence lower bound:

$$\arg\min_\theta \mathbb{E}_q\left[\log q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0) - \log p_\theta(\boldsymbol{x}_{0:T})\right]. \tag{6.1}$$

The Markov chain $q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)$ is called the *forward process*. It progressively turns the data x_0 into noise. The conditional distribution in each forward step is defined as:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) := \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\frac{\alpha_t}{\alpha_{t-1}}} x_{t-1}, \left(1 - \frac{\alpha_t}{\alpha_{t-1}}\right)\boldsymbol{I}\right), \tag{6.2}$$

where $\alpha_t \in (0, 1]$ and $\alpha_1, ..., \alpha_T$ is a decreasing sequence, which ensures that the values on the diagonal of the covariance matrix are positive. Reparameterizing using the definition in Eq. (6.2), an important property of the forward process is that:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_0, (1 - \alpha_t)\boldsymbol{I}\right). \tag{6.3}$$

Training and sampling: According to the definition of the reverse p and forward q processes, α_t and σ_t for all t are not learnable parameters. Thus, DDPM simplifies Eq. (6.1) as:

$$\arg\min_\theta \sum_t \mathbb{E}_q[D_{\mathrm{KL}}\left(q(x_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)\right)]. \tag{6.4}$$

DDPM further chooses the form of $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)$ to be:

$$\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\frac{\alpha_t}{\alpha_{t-1}}}}\left(\boldsymbol{x}_t - \frac{1 - \frac{\alpha_t}{\alpha_{t-1}}}{\sqrt{1 - \alpha_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right), \tag{6.5}$$

where $\boldsymbol{\epsilon}_\theta$ is a function with trainable parameters θ. According to the above definitions of the forward and reverse processes and applying the parameterization shown in Eq. (6.5), Eq. (6.4) can be simplified as $\arg\min_\theta L(\theta)$, where:

$$L(\theta) := \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}_t}\left[\gamma_t \left\| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_t, t\right)\right\|^2\right] \tag{6.6}$$

with $\gamma_t = \frac{(\alpha_{t-1} - \alpha_t)^2}{2\sigma_t^2 \alpha_t \alpha_{t-1}(1 - \alpha_t)}$ and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. It is important to note that when deriving this loss, DDPM reparameterize $\boldsymbol{x}_t$ as

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_t, \tag{6.7}$$

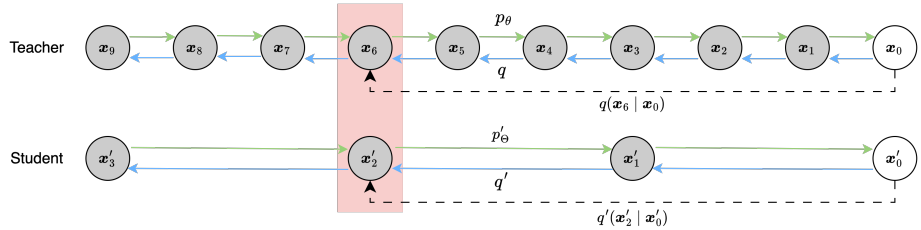


Figure 6.1: Single-Fold Distillation of Diffusion Model (BILA). The student accelerates the inference by a small number of steps T' instead of a large T. We use $T = 9$ and $T' = 3$ in the figure for readability. To align the teacher and student Markov chains, we propose to match the intermediate hidden variables to make, e.g., $q'(\boldsymbol{x}'_2 = \boldsymbol{x}_6 | \boldsymbol{x}'_0 = \boldsymbol{x}_0)$ equal to $q(\boldsymbol{x}_6 | \boldsymbol{x}_0)$.

using the property in Eq. (6.3). DDPM further simplifies L by setting $\gamma_t = 1$ independent of $\alpha_{1:T}$.

The number of sampling steps T decides the generalization capability and sampling cost of diffusion models. A big T leads to a reverse process with high generalization capability that better captures the pattern of x_0. However, it also increases the sampling time and makes the sampling from DDPMs significantly slower than other generative models, e.g, GANs. Such inefficiency promotes our design of BILA to reduce the number of sampling steps via knowledge distillation.

6.2.2. SINGLE-FOLD DISTILLED DIFFUSION (BILA)

Instead of multiple folds like progressive distillation, which introduces distortion at every fold by retraining, we want to extract knowledge from the teacher model through a single fold. The first consideration is aligning steps from a T-step teacher to steps of a given smaller T'-step student. Here, our goal is to distill the knowledge of the teacher model by mimicking its hidden variables $(\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ by a compressed student model[2] with hidden variables $(\boldsymbol{x}'_1, ..., \boldsymbol{x}'_{T'})$, where $T' \ll T$.

A crucial challenge stems from the need to map a subset of multiple consecutive steps at the teacher into one single step at the student (see Figure 6.1). For example, given an index[3] $c \cdot t$, where $c = T/T'$ and $c \in \mathbb{Z}$, according to Sec. 6.2.1, the distributions we can explicitly obtain from the teacher are $q(\boldsymbol{x}_{c \cdot t} \mid \boldsymbol{x}_{c \cdot t-1})$ (see Eq. 6.2). However, mapping $\boldsymbol{x}'_t$ with $\boldsymbol{x}_{c \cdot t}$ from the student to the teacher does not hold for the next step, i.e., $\boldsymbol{x}'_{t-1}$ does not correspond to $\boldsymbol{x}_{c \cdot t-1}$, which is supposed to map $\boldsymbol{x}_{c(t-1)}$. Thus, when distilling the DDPM-like Markov chains, it is not reasonable to simply and straightforwardly simulate $q'(\boldsymbol{x}'_t | \boldsymbol{x}'_{t-1})$ as $q(\boldsymbol{x}_{c \cdot t} | \boldsymbol{x}_{c \cdot t-1})$ while correspondingly estimating $p'_\Theta(\boldsymbol{x}'_t | \boldsymbol{x}'_{t-1})$ as $p_\theta(\boldsymbol{x}_{c \cdot t} | \boldsymbol{x}_{c \cdot t-1})$, where the notations q' and p'_Θ are used to represent the forward process and reverse process of the student model, respectively.

To overcome this challenge, we define a novel student model as follows. Note that the key definition for diffusion models is the forward process, as it determines the variational inference and dominates the training of the model. Therefore, to better distill the knowledge from the teacher, we design the forward process of the student $q'(\boldsymbol{x}'_{1:T'} \mid \boldsymbol{x}'_0)$

[2] Hereon we use $'$ in symbols to indicate the corresponding student variables.
[3] For simplicity, we assume that T is divisible by T' but this is not necessary (see Sec. 6.2.6).

by extracting the teacher's forward process $q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)$. Specifically, to ensure correspondence between the latent variables in the two diffusion models, given any $t \in [1, T']$, the proposed distillation algorithm **aims** to make $q'(\boldsymbol{x}'_t = \boldsymbol{x}_{c\cdot t} \mid \boldsymbol{x}'_0 = \boldsymbol{x}_0)$ equal to $q(\boldsymbol{x}_{c\cdot t}|\boldsymbol{x}_0)$, which has a close-form solution (see Eq. 6.3). After fixing q' by such an approximation in the forward process, the reverse process of the student is constructed accordingly. In the following, we show in detail how to define and train a student model.

6.2.3. THE FORWARD PROCESS OF THE STUDENT MODEL

To distill the DDPM teacher, we also assume that the student model has a Markovian forward process. Thus $q'\left(\boldsymbol{x}'_{1:T'} \mid \boldsymbol{x}'_0\right)$ is a Markov chain that can be factorized as

$$q'\left(\boldsymbol{x}'_{1:T'} \mid \boldsymbol{x}'_0\right) := \prod_{t=1}^{T'} q'\left(\boldsymbol{x}'_t \mid \boldsymbol{x}'_{t-1}\right).$$

As shown in Eq. (6.2), the forward process of the teacher is defined by a decreasing sequence $\alpha_1, ..., \alpha_T$. As for the student, different from the Gaussian distribution shown in Eq. (6.2), we set the student's forward process to the following form:

$$q'\left(\boldsymbol{x}'_t \mid \boldsymbol{x}'_{t-1}\right) := \mathcal{N}\left(\boldsymbol{x}'_t; \sqrt{\frac{\alpha_{c\cdot t}}{\alpha_{c\cdot t-c}}}\boldsymbol{x}'_{t-1}, \left(1 - \frac{\alpha_{c\cdot t}}{\alpha_{c\cdot t-c}}\right)\boldsymbol{I}\right), \tag{6.8}$$

for all $t \in [1, T']$ based on the elements of the sequence $\{\alpha_t\}_{t=1}^{T}$ (the hyper-parameters of the given teacher). Then, according to this forward process definition of the student, we have the property:

$$q'(\boldsymbol{x}'_t \mid \boldsymbol{x}'_0) = \mathcal{N}\left(\boldsymbol{x}'_t; \sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0, (1 - \alpha_{c\cdot t})\,\boldsymbol{I}\right). \tag{6.9}$$

This **ensures** that $q'(\boldsymbol{x}'_t = \boldsymbol{x}_{c\cdot t} \mid \boldsymbol{x}'_0 = \boldsymbol{x}_0) = q(\boldsymbol{x}_{c\cdot t}|\boldsymbol{x}_0)$, where $\boldsymbol{x}'_t$ corresponds to $\boldsymbol{x}_{c\cdot t}$. Thus, the student's Markov chain $q'\left(\boldsymbol{x}'_{1:T'} \mid \boldsymbol{x}'_0\right)$ can be regarded as a simplified copy of the teacher's forward process $q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)$.

Before introducing the reverse process, we derive some important forward process posteriors. Using Bayes' rule $q'\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t, \boldsymbol{x}'_0\right) = q'\left(\boldsymbol{x}'_t \mid \boldsymbol{x}'_{t-1}, \boldsymbol{x}'_0\right)\frac{q'(\boldsymbol{x}'_{t-1}|\boldsymbol{x}'_0)}{q'(\boldsymbol{x}'_t|\boldsymbol{x}'_0)}$, and the Markov chain property that $q'\left(\boldsymbol{x}'_t \mid \boldsymbol{x}'_{t-1}, \boldsymbol{x}'_0\right) = q'\left(\boldsymbol{x}'_t \mid \boldsymbol{x}'_{t-1}\right)$, we have the posteriors:

$$q'\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t, \boldsymbol{x}'_0\right) = \mathcal{N}(\boldsymbol{x}'_{t-1}; \frac{(1-\alpha_{c\cdot t-c})\sqrt{\alpha_{c\cdot t}}}{(1-\alpha_{c\cdot t})\sqrt{\alpha_{c\cdot t-c}}}\boldsymbol{x}'_t + \frac{\alpha_{c\cdot t-c}-\alpha_{c\cdot t}}{(1-\alpha_{c\cdot t})\sqrt{\alpha_{c\cdot t-c}}}\boldsymbol{x}'_0, \sigma'_t\boldsymbol{I}), \tag{6.10}$$

where $\sigma'_t = \frac{(1-\alpha_{c\cdot t-c})(\alpha_{c\cdot t-c}-\alpha_{c\cdot t})}{(1-\alpha_{c\cdot t})\alpha_{c\cdot t-c}}$ (see the appendix [40] for the derivation).

6.2.4. THE REVERSE PROCESS OF THE STUDENT MODEL

In the following, we define the reverse process of the student model according to its forward process. Similarly, it is also a Markov chain represented as

$$p'_\Theta(\boldsymbol{x}'_{0:T'}) := p'_\Theta\left(\boldsymbol{x}'_T\right)\prod_{t=1}^{T'} p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right),$$

where $p'_\Theta\left(\boldsymbol{x}'_T\right) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and Θ represents the learnable parameters of the student.

Then, we decide the form of $p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right)$. Referring to Eq. (6.9), we can reparameterize $\boldsymbol{x}'_t$ as a linear combination of $\boldsymbol{x}'_0$ and $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ that is $\boldsymbol{x}'_t = \sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0 + \sqrt{1-\alpha_{c\cdot t}}\epsilon_t$. Let the model $\epsilon'_\Theta(\boldsymbol{x}'_t, t)$ predict ϵ_t, we have a prediction of $\boldsymbol{x}'_0$ given $\boldsymbol{x}'_t$:

$$f_\Theta^{(t)}\left(\boldsymbol{x}'_t\right) := \left(\boldsymbol{x}'_t - \sqrt{1-\alpha_{c\cdot t}}\cdot\epsilon'_\Theta\left(\boldsymbol{x}'_t, t\right)\right)/\sqrt{\alpha_{c\cdot t}} \tag{6.11}$$

According to Eq. (6.4), we know that in the student diffusion model, $p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right)$ is a distribution that predicts $q'(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t, \boldsymbol{x}'_0)$. Thus, we define that for all $t \in [1, T']$,

$$p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right) = q'(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t, f_\Theta^{(t)}\left(\boldsymbol{x}'_t\right)). \tag{6.12}$$

Then, based on Eq. (6.10) and Eq. (6.12), by replacing $\boldsymbol{x}'_0$ as the predictor $f_\Theta^{(t)}\left(\boldsymbol{x}'_t\right)$, we have:

$$p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right) = \mathcal{N}\left(\boldsymbol{x}'_{t-1}; \boldsymbol{\mu}'_\Theta\left(\boldsymbol{x}'_t, t\right), \sigma'_t\boldsymbol{I}\right), \tag{6.13}$$

where the mean is:

$$\boldsymbol{\mu}'_\Theta\left(\boldsymbol{x}'_t, t\right) = \frac{(1-\alpha_{ct-c})\sqrt{\alpha_{ct}}}{(1-\alpha_{ct})\sqrt{\alpha_{ct-c}}}\boldsymbol{x}'_t + \frac{\alpha_{ct-c}-\alpha_{ct}}{(1-\alpha_{ct})\sqrt{\alpha_{ct-c}}}f_\Theta^{(t)}\left(\boldsymbol{x}'_t\right). \tag{6.14}$$

6.2.5. DISTILLATION PROCEDURE

Having defined the forward and reverse processes, here we design the algorithm for training the student model. The reparameterization of $\boldsymbol{\mu}'_\Theta\left(\boldsymbol{x}'_t, t\right)$ shown in Eq. (6.14) can be applied to derive the training loss of the student. For maximizing the log data likelihood of the observed data $\{\boldsymbol{x}'_0\}$ on the student[4], referring to Eq. (6.4), it is equivalent to minimize the Kullback-Leibler divergence between Eq. (6.10) and Eq. (6.13). Then by using Eq. (6.10), (6.13), (6.14) and (6.11), we have the following loss for training the student:

$$L(\Theta) := \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}'_0, \boldsymbol{\epsilon}'_t}\left[\gamma'_t \left\|\boldsymbol{\epsilon}'_t - \boldsymbol{\epsilon}_\Theta\left(\sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}'_t, t\right)\right\|^2\right], \tag{6.15}$$

where $\gamma'_t = \frac{(\alpha_{c\cdot t-c}-\alpha_{c\cdot t})^2}{2\sigma'^2_t\alpha_{c\cdot t}\alpha_{c\cdot t-c}(1-\alpha_{c\cdot t})}$. By the loss (Eq. 6.15), we can train the defined student model from scratch using the observed data $\{\boldsymbol{x}'_0\}$. However, in order to extract the knowledge from the trained teacher, in the following we connect the training of the student with the trained teacher.

In the derivation of the loss $L(\Theta)$ (Eq. 6.15), we use the following reparameterization (see the appendix [40] of the technical paper of this chapter for the derivation):

$$\boldsymbol{x}'_t = \sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}'_t. \tag{6.16}$$

According to Eq. (6.7), we know that $\boldsymbol{x}_{c\cdot t} = \sqrt{\alpha_{c\cdot t}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}_{c\cdot t}$. In our distillation, we want to make the hidden variable $\boldsymbol{x}'_t$ of the student equal to its corresponding hidden variable $\boldsymbol{x}_{c\cdot t}$ of the teacher. As the student and the teacher use the same observed data, $\boldsymbol{x}'_0 = \boldsymbol{x}_0$, by Eq. (6.16), if we assume $\boldsymbol{\epsilon}'_t = \boldsymbol{\epsilon}_{c\cdot t}$, then we can have $\boldsymbol{x}'_t = \boldsymbol{x}_{c\cdot t}$. By the training loss of the teacher $L(\theta)$ (Eq. 6.6), we know that the output $\boldsymbol{\epsilon}_\theta\left(\sqrt{\alpha_{c\cdot t}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}_{c\cdot t}, c\cdot t\right)$

[4] In distillation, student and teacher observe the same training samples. Thus, $\boldsymbol{x}'_0$ always equals $\boldsymbol{x}_0$ and $\{x'_0\}$ is equivalent to $\{x_0\}$. To avoid confusion, we use $\{x'_0\}$ to represent the observed data when training the student.

from the trained function $\boldsymbol{\epsilon}_\theta$ of the teacher is a good predictor for $\boldsymbol{\epsilon}_{c\cdot t}$ (also for $\boldsymbol{\epsilon}'_t$). Thus, we naturally rewrite the student loss (Eq. 6.15) as

$$L(\Theta) := \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}'_0,\boldsymbol{\epsilon}'_t}[\gamma'_t \| \boldsymbol{\epsilon}_\theta\left(\sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}'_t, c\cdot t\right) - \boldsymbol{\epsilon}_\Theta\left(\sqrt{\alpha_{c\cdot t}}\boldsymbol{x}'_0 + \sqrt{1-\alpha_{c\cdot t}}\boldsymbol{\epsilon}'_t, t\right)\|^2]. \tag{6.17}$$

Training: For distillation, we train the student according to the loss (Eq. 6.17). During the implementation in Sec. 6.3, we simplify the loss by setting $\gamma'_t = 1$, a simpler approach shown beneficial for sample quality.

Sampling: We need an input noise $\boldsymbol{x}'_{T'}$ when sampling images from the trained student. The straightforward way is to draw the noise by $\boldsymbol{x}'_{T'} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Reducing the number of sampling steps can negatively impact a model's generalization ability. To avoid the risk of generating poor-quality images from certain input sample points when performing direct random sampling from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we propose a method to reduce the sampling space of $\boldsymbol{x}'_{T'}$. This approach ensures that the student model maintains strong performance even with fewer sampling steps (e.g., 4 steps), preserving high-quality outputs while improving efficiency. For a distilled student model, a high-quality input noise set $\{\hat{\boldsymbol{x}}'\}$ can be obtained by $\text{argmin}_{\hat{\boldsymbol{x}}'} \| StudentDiffusion(\hat{\boldsymbol{x}}') - \boldsymbol{x}'_0 \|$. A random linear weighted combination of elements from this set will be used for the random sampling of $\boldsymbol{x}'_{T'}$.

Since the reverse process of the student is defined by Eq. (6.13), given the input $\boldsymbol{x}'_{T'}$, we can steadily sample all variables from $\boldsymbol{x}'_{T'-1}$ to $\boldsymbol{x}'_0$. Note that the student model is also a DDPM model. Any sampling algorithm compatible with DDPM can be applied to the distilled student.

6.2.6. DISTILLATION ON FLEXIBLE SUB-SEQUENCE

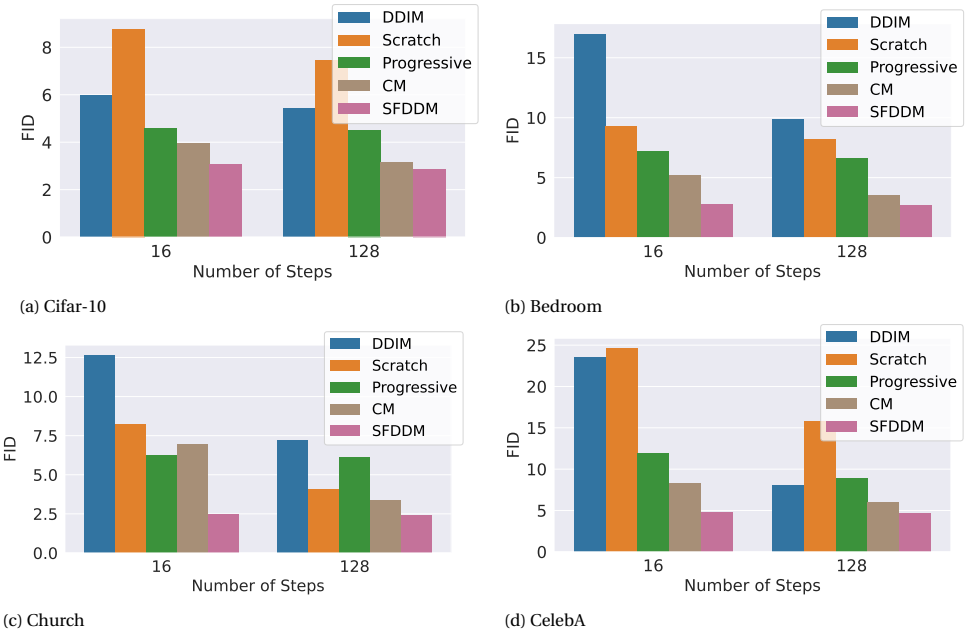
In the previous derivation, the student extracts the knowledge from a special variable subset $\{\boldsymbol{x}_{c\cdot t}\}$ of the teacher where $t \in [1, T']$. Indeed, our proposed method can be extended to a more general case where we distill the knowledge from any given subset $\{\boldsymbol{x}_{\phi_0}, ..., \boldsymbol{x}_{\phi_{T'}}\}$ of the teacher where ϕ is an increasing sub-sequence of $\{0, ..., T\}$, $\phi_0 = 0$ and $\phi_{T'} = T$. In this general case, the Gaussian mean of $p'_\Theta\left(\boldsymbol{x}'_{t-1} \mid \boldsymbol{x}'_t\right)$ (see Eq. 6.13) is

$$\boldsymbol{\mu}'_\Theta\left(\boldsymbol{x}'_t, t\right) = \frac{(1-\alpha_{\phi_{t-1}})\sqrt{\alpha_{\phi_t}}}{(1-\alpha_{\phi_t})\sqrt{\alpha_{\phi_{t-1}}}}\boldsymbol{x}'_t + \frac{\alpha_{\phi_{t-1}} - \alpha_{\phi_t}}{(1-\alpha_{\phi_t})\sqrt{\alpha_{\phi_{t-1}}}}\mathscr{F}_\Theta^{(t)}\left(\boldsymbol{x}'_t\right),$$

where $\mathscr{F}_\Theta^{(t)}\left(\boldsymbol{x}'_t\right) := \left(\boldsymbol{x}'_t - \sqrt{1-\alpha_{\phi_t}} \cdot \epsilon'_\Theta\left(\boldsymbol{x}'_t, t\right)\right) / \sqrt{\alpha_{\phi_t}}$ is a prediction of $\boldsymbol{x}'_0$ given $\boldsymbol{x}'_t$. Remarkably, T no longer needs to be divisible by T'. This extension is beneficial as it greatly expands the applicability of our distillation under various steps of teacher diffusion models.

6.3. EVALUATION

BILA distills the knowledge of any arbitrary DDPM-like teacher models with efficiency and high sampling quality. We demonstrate its effectiveness by four image benchmark datasets: CIFAR-10 [65], CelebA [74], LSUN-Church and LSUN-Bedroom [131], as well as one tabular dataset: 2D Swiss Roll following the diffusion model settings [112]. For



(a) Cifar-10

(b) Bedroom

(c) Church

(d) CelebA

Figure 6.2: Fréchet Inception Distance (FID) under different number of sampling steps from the teacher $T = 1024$, on four datasets, Cifar-10 (a), Bedroom (b), Church (c), CelebA (d).

each image dataset, we distill the same teacher diffusion model of DDPM into 16, 100, or 128 student steps. Note that although the original DDPM contains 1000 sampling steps, in this chapter, we set it as 1024 steps in order to compare with progressive distillation [100], which requires a number of steps that is a power of 2 for progressive halving. For the 2D Swiss Roll dataset, we distill a teacher model with 500 steps into a student with 50 steps by BILA . The evaluation metric applied is FID together with perceptual visualization of sampled images.

6.3.1. SAMPLING QUALITY AND EFFICIENCY

To demonstrate the quality and efficiency of BILA , we report the FID in Figure 6.2 on all four image datasets comparing against DDIM, progressive distillation ("Progressive"), Consistency model (CM) [110], and training directly on the smaller model with the same dataset as the teacher model ("From scratch").

In general, our BILA achieves the best FID over different datasets and different small T'. From the results, we observe that the sampling data quality increases with the increase of T', as more sampling steps match more hidden variables of the teacher model, better approximating the teacher distributions.

Comparing the baselines, DDIM achieves mostly the lowest quality in sampled images as shown by high FID scores. This stems from the nature that DDIM **does not** retrain a student model but focuses on improving the inference efficiency of the original model. Thus, it benefits from simplicity but falls short in terms of quality, when skip-

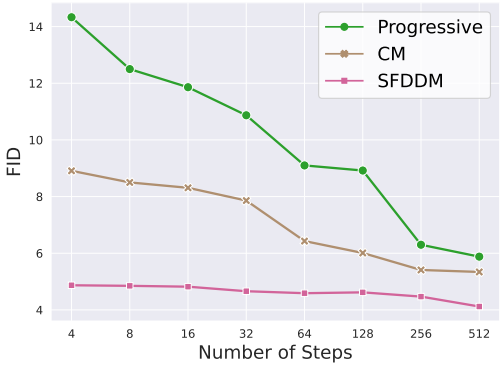


Figure 6.3: Fréchet Inception Distance (FID) of the methods with different number of sampling steps on the dataset CelebA-HQ.

Figure 6.4: Visualizied the images generated by the student models with different number of sampling steps of our method BILA on the dataset CelebA-HQ.

ping too many intermediate steps during sampling, e.g., $T' = 16$. On the other hand, training from scratch on a small diffusion model comes in as the second worst in terms of FID, due to the difficulty of capturing image features with a small number of steps. Progressive distillation yields marginal improvement as multiple folding and retraining increases distortion from teacher knowledge due to noises it introduces at each folding. An interesting finding emerges in the context of the LSUN-Church dataset with 128 sampling steps. Here, training from scratch surpasses progressive distillation in FID performance. This can be attributed to the fact that, with a relatively large number of steps, direct training exhibits superior quality compared to progressive distillation, where systematic distortion occurs fold by fold. This is consistent with the conclusion in [100]. We also compare the student FID for progressive distillation, CM, and BILA with varying sampling steps from 4 to 512 in Figure 6.3. Overall, BILA consistently outperforms the baselines. We also visualize the corresponding student model output of step 4 and 16 in Figure 6.4, demonstrating limited quality loss of small sampling steps by BILA .

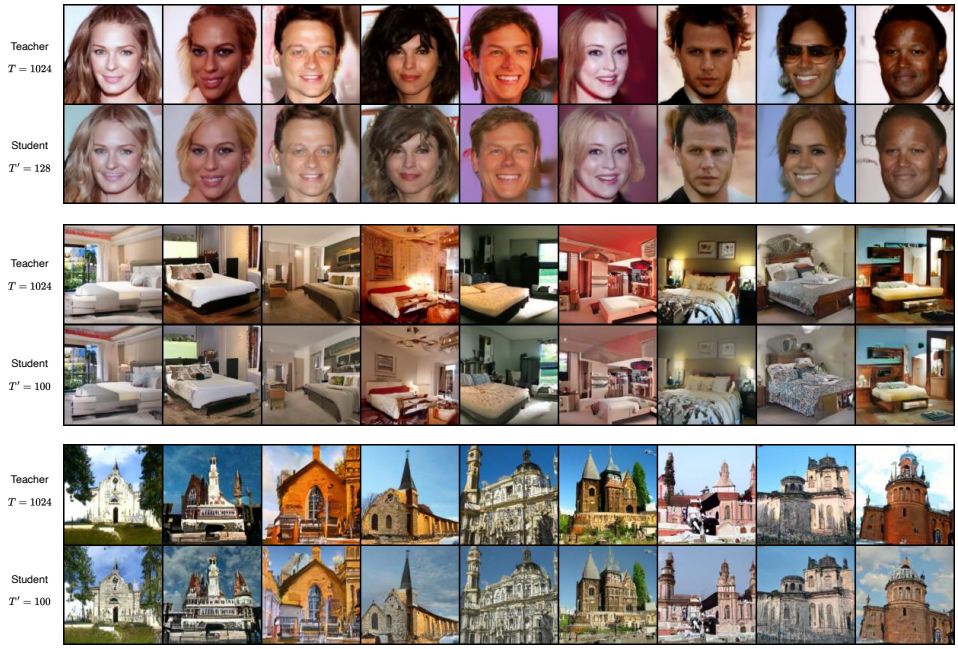


Figure 6.5: Consistency on CelebA-HQ, LSUN-Bedroom and LSUN-Church: inputing the same noise into the teacher and the student.

6.3.2. DISTILLATION WITH DIFFERENT SUB-SEQUENCES

In accordance with Sec. 6.2.6, our algorithm can be extended to accommodate flexible sub-sequences of the student model. Here, we compare FID values for different choices to demonstrate their impact. Specifically, the different cases of flexible sub-sequence are designed by various degrees of concentration of mapped steps around the midpoint element. We define it by the percentage of elements distributed uniformly within a 5% range near the midpoint element (i.e., 512) while the others are uniformly distributed in the remaining range of steps. Moreover, In our results presented in Table 6.1, we include the concentration degrees of 40%, 20%, plus Scattered. "Scattered" refers to the student model matching a sparse sub-sequence spread across the full teacher Markov chain. Scattered allows to cover more knowledge of the noising/denoising procedure form the teacher. Yet, concentrated choices, in which elements are nearby and centered in a partial part of the teacher chain, are still able to distill high-quality diffusion models, as shown by similarity in FID scores.

Table 6.1: Distilling the same teacher with different sub-sequences on CelebA-HQ with $T' = 16$.

sub-sequence	Concentrated (40%)	Concentrated (20%)	Scattered
FID	4.85	4.69	4.82

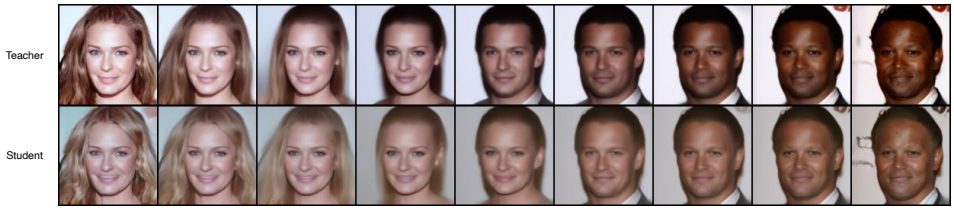
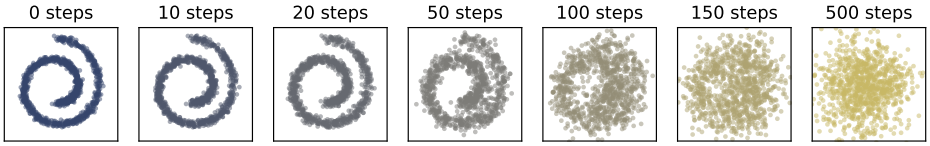


Figure 6.6: Interpolation on the distilled student and original teacher.

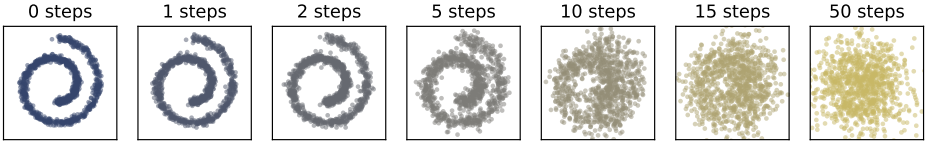
6.3.3. CONSISTENCY BETWEEN TEACHER AND STUDENT

We zoom into the consistency between images sampled by the teacher and student models when inputting identical noise. It is important to note that for a fair comparison of the distillation results, we use the DDIM sampling method (also applied in Sec. 6.3.4) for both the DDPM teacher and our BILA student. This is because the output of the original DDPM sampling is not solely determined by the input noise, owing to the introduced random factor during the stochastic generative process. In contrast, DDIM sampling ensures pair correspondence, i.e., same input, same output, facilitating a clear and accurate comparison. The outcomes across three distinct datasets are illustrated in Figure 6.5. It is noteworthy that we employ different sampling steps for the student among different datasets, thereby validating the flexibility of our approach under varying numbers of sub-sequences, where the number of steps is not a power of 2. As evidenced by the images, it is clearly observed that inputting identical noise leads to similar outputs, showing our effectiveness in transferring knowledge from the teacher to a student having only approximately 1/10 of the steps.

6.3.4. INTERPOLATION ON THE TEACHER AND THE STUDENT



(a) Teacher diffusion forward process of 500 steps on 2D Swiss Roll.



(b) Student diffusion forward process of 50 steps on 2D Swiss Roll.

Figure 6.7: Forward process of teacher and BILA student model on 2D Swiss Roll dataset.

We further assess the efficacy of knowledge transfer through measuring the similarity of semantic interpolation between the teacher and student models. We evenly interpolate between two given noises to showcase the intermediate sampled data in Figure 6.6. The results reveal a stable visual interpolation in the generated images since the input

noise encodes distinctive high-level features of the image. Consequently, the interpolation data implicitly captures the features between the two noises into perceptually similar outputs. When comparing the output of the teacher and student, we observe similarity in each interpolation, showcasing the effectiveness of distillation as the student successfully inherits a stable and consistent sampling capability from the teacher.

6.3.5. DISTILLATING TABULAR DATA

To assess the generality and applicability of BILA among different types of data, we apply BILA on the tabular 2D Swiss Roll dataset, visualized in Figure 6.7. On this tabular DDPM, the forward process turns Swiss Roll-like points into randomly distributed 2D points. Contrarily, the reverse process constructs a Swiss Roll distribution according to randomly distributed points. Figure 6.7 presents the forward process of both teacher and student models. The results show that our proposed algorithm BILA works on DDPM for distilling the generation of tabular data by demonstrating the similar capability of generating 2D Swiss Roll points with student sampling steps 10x fewer than the teacher diffusion.

6.4. CONCLUSION

In this chapter, we propose a novel and effective single-fold distillation method for diffusion models, BILA . In contrast to the prior study of progressive distillation, BILA is able to compress any T-step teacher model into any T'-step student model in single-fold distillation. The key enabling features are (i) new derivation of the forwarding process of the student model, which leverages the reparameterization of the teacher model and approximation of their Markovian states; and (ii) optimization of the denoise process of the student model by minimizing the difference of model outputs and distribution of the hidden variables. Our evaluation results on five datasets show that BILA achieves remarkable quality on FID, allowing to strike better quality-compute tradeoffs.

7

CONCLUSION

This thesis explores how knowledge can be effectively and efficiently distilled from trained neural networks and noisy data in various machine learning settings. Our study is structured around three aspects: the type of target (data or model), the accessibility to the target (white-box or black-box), and the type of task (classification or generation). These dimensions define the different scenarios where knowledge distillation is needed. To guide the research, we propose five specific research questions, each addressing a particular setting or challenge related to the overarching goal. For each question, we design and analyze algorithmic solutions, supported by experiments, to understand how knowledge can be distilled in practical and reliable ways. Below, we summarize the main findings of the thesis, highlighting the techniques that prove effective across different conditions. Then, we discuss open problems related to each research question that remain unresolved in the research of this thesis and may inspire future work.

7.1. CONCLUSIONS

The conclusions of the five research questions are as follows:

1. In Chapter 2, we have proposed an online label aggregation method (BILA) for distilling knowledge and producing high-quality labels based on the data of noisy labels collected from crowd workers. Our method enables conducting label curation and aggregation in an online manner, where labels are continuously handled over a subset of content, rather than processing the entire content all at once. We designed the aggregation method based on variational Bayesian inference and applied neural networks as the approximate distribution. We have demonstrated that it is possible to design an online aggregation method that achieves even higher accuracy compared to existing offline methods. We also made a convergence analysis of the proposed method and derived a convergence bound.

2. In Chapter 3, we have presented TANDEMGAN, a method for distilling knowledge from a trained classifier that is deployed online and offers only black-box access.

The main idea of TANDEMGAN is to explore and exploit the input data space of a given classifier to generate high-quality synthetic samples for querying the target classifier. Thus, the algorithm can distill knowledge from the trained target without real data samples. We have shown that compared with other data-free knowledge distillation algorithms that only rely on exploration, our method achieves higher accuracy on the substitute (copied) model. We conclude that introducing exploitation into the data space searching process is meaningful for data-free knowledge distillation.

3. In Chapter 4, we have introduced a black-box knowledge distillation method, AEDM, which uses public data knowledge to facilitate the knowledge distillation on target classifiers. To utilize the public data knowledge, a generative model is pre-trained by publicly available datasets, e.g., ImageNet. Then we steer the generative model by tuning its input noise to produce informative samples for querying the target model. AEDM significantly outperforms methods without public data knowledge which demonstrates that semantic knowledge from public data is important for knowledge distillation. We also proposed a federated learning framework to enable distributed collaboration among multiple participants. We conclude that public data knowledge can be applied to significantly reduce the required number of queries in black-box knowledge distillation.

4. In Chapter 5, we have demonstrated that dark knowledge, represented by underlying inference probabilities, exists in generative models like GANs and VAEs. The dark knowledge is beneficial for white-box knowledge distillation. We derived a risk bound for the distillation method with dark knowledge and proved that using dark knowledge helps the method generalize better. To empirically validate our proof, we designed a knowledge distillation algorithm called `DKtill` which leverages the dark knowledge of generative models. `DKtill` achieved higher distillation quality across three different generative models and datasets, outperforming distillation algorithms that do not utilize the proposed dark knowledge.

5. In Chapter 6, we have introduced a knowledge distillation method, SFDDM, which targets diffusion models to generate student diffusion models with fewer steps, thereby reducing inference time. A diffusion model has multiple steps in the sampling process which means to distill a diffusion model, the outputs and probability distributions on each step should be considered in the distillation algorithm. SFDDM distills a diffusion model by making a simplified copy of its Markov chains rather than only mimicking the target's outputs. Besides, SFDDM conducts knowledge distillation in one single fold rather than multiple binary folds. We concluded that preserving the features of the original Markov chains by defining a new forward process for the student diffusion model is beneficial for maintaining sampling quality in diffusion distillation.

This thesis explored how to effectively and efficiently distill knowledge from trained neural networks and noisy data. From the five research questions, we find two overall conclusions.

1. Superior student models are created not by merely replicating a teacher's outputs, but by designing intelligent processes that identify, prioritize, and extract the most valuable information, whether from elaborated inputs or pre-trained models.

2. Efficiency comes from designing learning processes that match the constraints of the distillation task setting. Methods that work incrementally, or align with model structure, often use fewer resources while keeping good performance.

While advancing the field of knowledge distillation, this thesis is naturally bounded by several methodological and conceptual constraints. The techniques developed are designed to specific architectural and algorithmic choices, such as the use of relatively simple neural networks for variational inference in Chapter 2, narrow quality metrics in Chapter 3, and the dependency on specific probability approximations for non-probabilistic generators in Chapter 5. Furthermore, the scope of empirical validation is limited, as the distillation methods for generative models in Chapters 5 and 6 were primarily evaluated on standard models and datasets, with their extension to more complex variants like guided or latent diffusion models remaining open questions. From a computational perspective, certain frameworks face efficiency constraints, including the sampling overhead from using diffusion models in Chapter 4 and the absence of optimized inference methods for the distilled models in Chapter 6. In summary, these limitations attract a clear and promising works for future research.

7.2. FUTURE DIRECTIONS

This thesis has conducted an exploratory investigation into the solution space of knowledge distillation. In addition to what is presented in this thesis, there are still numerous opportunities for further exploration. We now outline several promising directions for future work under each chapter:

1. In Chapter 2, we defined the approximate distribution of BILA as a multi-layer perceptron with two hidden layers which is a simple and easily trainable network. However, future works may try other complex neural networks, e.g., large convolutional neural networks that have high generalization ability on massive data. For the generating distribution of BILA , we designed it based on a confusion matrix to describe the behavior of workers. It is also possible to replace it with another distribution definition that can describe the generation process of the noisy labels. Thus, future works can study the effectiveness of other probability distributions on the generation part or the approximation part.

2. In Chapter 3, we used inference confidence as the criterion to evaluate the quality of a synthetic sample. Then, we further designed the exploitation stage of TANDEMGAN based on this criterion. However, this is not the only criterion for evaluating synthetic samples. Future works may use other criteria and propose another optimization goal for the exploitation stage. For example, from an active learning perspective, a high-quality sample should be an informative sample. Then, many query strategies for selecting informative samples from active learning can be applied to build the exploitation stage of the knowledge distillation algorithm.

3. In Chapter 4, we used diffusion models as the pre-trained generative model which has a high generalization ability. However, the sampling of diffusion models is time-consuming. Future works may try other generative models with lower computation overhead. We applied the entropy of the substitute output as the criterion to select informative samples. Other criteria can be tried in the future works. The federated distillation part of AEDM is realized by FedSGD. Future works may apply FedAvg to further reduce the overhead of transmitting model parameters.

4. In Chapter 5, we derived a risk bound based on probabilistic generators. For non-probabilistic generators, we made a probability approximation and then analyzed its impact. Therefore, rigorous proof of non-probabilistic generators can be done in future works. Besides, `DKtill` illustrated a method based on additional layers to do probability approximation. Future works may explore other ways of probability approximation.

5. In Chapter 6, we conducted our experiments mainly on vanilla diffusion models for images. In future works, SFDDM can be extended to guided diffusion models, tabular diffusion models, latent diffusion models, etc. These diffusion models are also widely used in the industry. Thus, studying the distillation on them is meaningful. SFDDM proposed a novel training method to train a student model based on a given teacher target. However, there are no corresponding new inference methods to further improve the sampling quality. This can be done in future work.

Looking ahead, future research can continue to improve knowledge distillation by building on several ideas explored in this thesis. One direction is to explore more powerful model architectures and richer representations to better capture the structure of noisy data and complex models. Another is to rethink how the quality of synthetic or selected samples is defined, possibly by incorporating ideas from related fields such as active learning or uncertainty estimation. Computational efficiency remains an important concern, especially when working with large models or under resource constraints. Future work may seek to reduce training or inference costs while maintaining strong performance. On the theoretical side, a deeper understanding of the assumptions and limitations behind current algorithms would help make knowledge distillation more reliable. Finally, expanding the application of distillation methods to a broader range of model types and learning settings will be important for making these techniques more widely used.

BIBLIOGRAPHY

[1] Charu C. Aggarwal et al. “Active Learning: A Survey”. In: (2014), pp. 571–606.

[2] Angeline Aguinaldo et al. “Compressing GANs using Knowledge Distillation”. In: *CoRR* abs/1902.00159 (2019).

[3] James Beetham et al. “Dual Student Networks for Data-Free Model Stealing”. In: *ICLR 2023*. 2023.

[4] Christopher M Bishop. “Pattern recognition and machine learning”. In: (2006), pp. 461–517.

[5] David Carmel et al. “Multi-objective ranking optimization for product search using stochastic label aggregation”. In: *Proceedings of The Web Conference 2020*. 2020, pp. 373–383.

[6] Varun Chandrasekaran et al. “Exploring Connections Between Active Learning and Model Extraction”. In: *USENIX Security 2020*. 2020, pp. 1309–1326.

[7] Fenghong Chen and Zhidong Shen. “A Data-Free Substitute Model Training Method for Textual Adversarial Attacks”. In: *International Conference on Neural Information Processing*. 2023, pp. 295–307.

[8] Hanting Chen et al. “Distilling Portable Generative Adversarial Networks for Image Translation”. In: *AAAI*. 2020.

[9] Altannar Chinchuluun et al. *Pareto optimality, game theory and equilibria*. Vol. 17. 2008.

[10] Cloud Team Google. *Google Cloud Vision API for OCR*. `https://cloud.google.com/vision/docs/ocr`. 2016.

[11] Michael Crawford et al. “Survey of review spam detection using machine learning techniques”. In: *Journal of Big Data* (2015), pp. 1–24.

[12] Alexander Philip Dawid and Allan M Skene. “Maximum likelihood estimation of observer error-rates using the EM algorithm”. In: *Applied statistics* (1979), pp. 20–28.

[13] Kamil Deja, Tomasz Trzcinski, and Jakub M. Tomczak. “Learning Data Representations with Joint Diffusion Models”. In: *ECML PKDD*. 2023, pp. 543–559.

[14] Jia Deng et al. “ImageNet: A large-scale hierarchical image database”. In: *(CVPR*. 2009, pp. 248–255.

[15] Lei Deng et al. “Model compression and hardware acceleration for neural networks: A comprehensive survey”. In: *Proceedings of the IEEE* (2020), pp. 485–532.

[16] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: `http://archive.ics.uci.edu/ml`.

[17] *European Union's General Data Protection Regulation*. May 25, 2018.

[18] Li Fei-Fei. "ImageNet: crowdsourcing, benchmarking & other cool things". In: *CMU VASC Seminar*. Vol. 16. 2010, pp. 18–25.

[19] André Freitas and Edward Curry. "Big Data Curation". In: *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. 2016, pp. 87–118.

[20] Alex Gaunt, Diana Borsa, and Yoram Bachrach. "Training deep neural nets to aggregate crowdsourced responses". In: *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*. 2016, p. 242251.

[21] Sandra Gilhuber et al. "DiffusAL: Coupling Active Learning with Graph Diffusion for Label-Efficient Node Classification". In: *ECML PKDD*. 2023, pp. 75–91.

[22] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*. 2015.

[23] Ian J. Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*. 2014, pp. 2672–2680.

[24] *Google OCR*. `https://cloud.google.com/use-cases/ocr`.

[25] *Google Translate*. `https://translate.google.com/`.

[26] Jianping Gou et al. "Knowledge distillation: A survey". In: *International Journal of Computer Vision* (2021), pp. 1789–1819.

[27] Federica Granese et al. "MEAD: A multi-armed approach for evaluation of adversarial examples detectors". In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022*. 2023, pp. 286–303.

[28] Chuan Guo et al. "On calibration of modern neural networks". In: 2017.

[29] Zhongyi Guo et al. "Attribution of Adversarial Attacks via Multi-task Learning". In: *International Conference on Neural Information Processing*. 2023, pp. 81–94.

[30] Viresh Gupta and Tanmoy Chakraborty. "VIKING: Adversarial Attack on Network Embeddings via Supervised Network Poisoning". In: *PAKDD*. 2021, pp. 103–115.

[31] Hu Han et al. "Demographic estimation from face images: Human vs. machine performance". In: *IEEE transactions on pattern analysis and machine intelligence* (2015), pp. 1148–1161.

[32] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. 2009.

[33] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. 2016, pp. 770–778.

[34] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*. 2017, pp. 6626–6637.

[35] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. “Distilling the Knowledge in a Neural Network”. In: *CoRR* abs/1503.02531 (2015).

[36] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[37] Jonathan Ho et al. “Cascaded diffusion models for high fidelity image generation”. In: *The Journal of Machine Learning Research* (2022), pp. 2249–2281.

[38] Chi Hong. *AEDM*. https://github.com/ChiHong-Xtautau/DiffModelStealing. 2022.

[39] Chi Hong. *Appendix for chapter five*. https://drive.google.com/file/d/1VEof_pXJkUNDnu_NqZcyCy6tEyxRZRe6/view?usp=sharing.

[40] Chi Hong. *Appendix for chapter six*. https://drive.google.com/file/d/1zly0t3b6OiCCk_YV-ZPPkZ7nRDE5pjYR/view?usp=sharing.

[41] Chi Hong. *Appendix for chapter two*. https://dl.acm.org/doi/pdf/10.1145/3442381.3449933.

[42] Chi Hong. *BILA*. https://github.com/ChiHong-Xtautau/BiLA-OnlineLabelAggregation. 2020.

[43] Chi Hong. *DKtill*. https://github.com/ChiHong-Xtautau/generator-distillation. 2023.

[44] Chi Hong. *SFDDM*. https://github.com/ChiHong-Xtautau/diffusion_distillation. 2024.

[45] Chi Hong. *TANDEMGAN*. https://github.com/ChiHong-Xtautau/MSECCV2022. 2021.

[46] Chi Hong et al. “Exploring and Exploiting Data-Free Model Stealing”. In: *ECML PKDD*. 2023.

[47] Bosong Huang et al. “Two-Stage Denoising Diffusion Model for Source Localization in Graph Inverse Problems”. In: *ECML PKDD 2023*. 2023.

[48] Yongfeng Huang, Shanshan Tu, Basharat Ahmad, et al. “Learning a semantic space for modeling images, tags and feelings in cross-media search”. In: *PAKDD*. 2019, pp. 65–76.

[49] Muhammad Imran et al. “AIDR: Artificial intelligence for disaster response”. In: *Proceedings of the 23rd International Conference on World Wide Web*. 2014, pp. 159–162.

[50] Phillip Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *CVPR*. 2017, pp. 1125–1134.

[51] Matthew Jagielski et al. “High Accuracy and High Fidelity Extraction of Neural Networks”. In: *USENIX Security*. 2020, pp. 1345–1362.

[52] Guangda Ji and Zhanxing Zhu. “Knowledge Distillation in Wide Neural Networks: Risk Bound, Data Efficiency and Imperfect Teacher”. In: *NeurIPS 2020*. 2020.

[53] Mika Juuti et al. “PRADA: Protecting Against DNN Model Stealing Attacks”. In: *IEEE EuroS&P*. 2019, pp. 512–527.

[54] Neel Kanwal et al. “Vision Transformers for Small Histological Datasets Learned Through Knowledge Distillation”. In: *PAKDD*. 2023.

[55] Sanjay Kariyappa, Atul Prakash, and Moinuddin K. Qureshi. “MAZE: Data-Free Model Stealing Attack Using Zeroth-Order Gradient Estimation”. In: *IEEE/CVF CVPR*. 2021.

[56] Sanjay Kariyappa, Atul Prakash, and Moinuddin K. Qureshi. “Protecting DNNs from Theft using an Ensemble of Diverse Models”. In: *ICLR*. 2021.

[57] Sanjay Kariyappa and Moinuddin K. Qureshi. “Defending Against Model Stealing Attacks With Adaptive Misinformation”. In: *2020 IEEE/CVF CVPR*. 2020, pp. 767–775.

[58] Hyun-Chul Kim and Zoubin Ghahramani. “Bayesian classifier combination”. In: *Artificial Intelligence and Statistics*. 2012, pp. 619–627.

[59] Diederik Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[60] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *ICLR*. 2014.

[61] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. 2009.

[62] Igor Kononenko. “Machine learning for medical diagnosis: history, state of the art and perspective”. In: *Artificial Intelligence in medicine* (2001), pp. 89–109.

[63] Josua Krause, Adam Perer, and Kenney Ng. “Interacting with predictions: Visual inspection of black-box machine learning models”. In: *Proceedings of the 2016 CHI conference on human factors in computing systems*. 2016, pp. 5686–5697.

[64] Kalpesh Krishna et al. “Thieves on Sesame Street! Model Extraction of BERT-based APIs”. In: *8th International Conference on Learning Representations, ICLR 2020*.

[65] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009).

[66] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).

[67] Kenichi Kurihara, Max Welling, and Yee Whye Teh. “Collapsed Variational Dirichlet Process Mixture Models.” In: *IJCAI*. Vol. 7. 2007, pp. 2796–2801.

[68] Muyang Li et al. “Gan compression: Efficient architectures for interactive conditional gans”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 5284–5294.

[69] Yanyu Li et al. “SnapFusion: Text-to-Image Diffusion Model on Mobile Devices within Two Seconds”. In: *arXiv preprint arXiv:2306.00980* (2023).

[70] Yuan Li, Benjamin Rubinstein, and Trevor Cohn. “Exploiting worker correlation for label aggregation in crowdsourcing”. In: *International Conference on Machine Learning*. 2019, pp. 3886–3895.

[71] Zilong Lin, Yong Shi, and Zhi Xue. “Idsgan: Generative adversarial networks for attack generation against intrusion detection”. In: *PAKDD*. 2022, pp. 79–91.

[72] Qiang Liu, Jian Peng, and Alexander T Ihler. “Variational inference for crowdsourcing”. In: *Advances in neural information processing systems*. 2012, pp. 692–700.

[73] Zhen Liu et al. “ItrievalKD: An Iterative Retrieval Framework Assisted with Knowledge Distillation for Noisy Text-to-Image Retrieval”. In: *PAKDD*. 2023.

[74] Ziwei Liu et al. “Deep Learning Face Attributes in the Wild”. In: *Proceedings of International Conference on Computer Vision (ICCV)*. 2015.

[75] David Lopez-Paz et al. “Unifying distillation and privileged information”. In: *ICLR*. 2016.

[76] Siyuan Lu, Zhihai Lu, and Yu-Dong Zhang. “Pathological brain detection based on AlexNet and transfer learning”. In: *Journal of computational science* (2019), pp. 41–47.

[77] Alan Lundgard et al. “Bolt: Instantaneous crowdsourcing via just-in-time training”. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–7.

[78] Kolby Nottingham Markelle Kelly Rachel Longjohn. *The UCI Machine Learning Repository*. `https://archive.ics.uci.edu`.

[79] Andreas Maurer and Massimiliano Pontil. “Empirical bernstein bounds and sample variance penalization”. In: *COLT 2009 - The 22nd Conference on Learning Theory* (2009).

[80] Leland McInnes and John Healy. “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction”. In: *CoRR* abs/1802.03426 (2018). arXiv: 1802.03426.

[81] Geoffrey J McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*. 2007.

[82] Brendan McMahan. “Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization”. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 2011, pp. 525–533.

[83] H Brendan McMahan and Matthew Streeter. “Adaptive bound optimization for online convex optimization”. In: *arXiv preprint arXiv:1002.4908* (2010).

[84] Chenlin Meng et al. “On distillation of guided diffusion models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 14297–14306.

[85] Aditya Krishna Menon et al. “A statistical perspective on distillation”. In: *ICML*. Vol. 139. 2021, pp. 7632–7642.

[86] Paul Micaelli and Amos J. Storkey. “Zero-shot Knowledge Transfer via Adversarial Belief Matching”. In: *NeurIPS*. 2019, pp. 9547–9557.

[87] Hossein Mobahi, Mehrdad Farajtabar, and Peter L. Bartlett. “Self-Distillation Amplifies Regularization in Hilbert Space”. In: *NeurIPS*. 2020.

[88] Yuval Netzer et al. “Reading digits in natural images with unsupervised feature learning”. In: *NIPS workshop on deep learning and unsupervised feature learning*. Vol. 2011. 5. 2011, p. 7.

[89] Dang Nguyen et al. “Knowledge distillation with distribution mismatch”. In: *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021*. 2021, pp. 250–265.

[90] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. “Knockoff Nets: Stealing Functionality of Black-Box Models”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*. 2019, pp. 4954–4963.

[91] John Paisley, David Blei, and Michael Jordan. “Variational Bayesian inference with stochastic search”. In: *arXiv preprint arXiv:1206.6430* (2012).

[92] Nicolas Papernot et al. “Practical Black-Box Attacks against Machine Learning”. In: *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017*. 2017, pp. 506–519.

[93] Mary Phuong and Christoph H. Lampert. “Towards Understanding Knowledge Distillation”. In: *CoRR* abs/2105.13093 (2021).

[94] Boris T Polyak. “Introduction to optimization. optimization software”. In: *Inc., Publications Division, New York* 1 (1987).

[95] Foster Provost, Wang Jing, and Panagiotis G. Ipeirotis. “Quality management on amazon mechanical turk”. In: *Proceedings of the ACM SIGKDD workshop on human computation*. 2010, pp. 64–67.

[96] Yunchen Pu et al. “Variational autoencoder for deep learning of images, labels and captions”. In: *NIPS* 29 (2016).

[97] Vikas C Raykar et al. “Learning from crowds”. In: *Journal of Machine Learning Research* 11.Apr (2010), pp. 1297–1322.

[98] Pengzhen Ren et al. “A survey of deep active learning”. In: *ACM Comput. Surv. (CSUR)* 54.9 (2021), pp. 1–40.

[99] Yuxi Ren et al. “Online multi-granularity distillation for gan compression”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 6793–6803.

[100] Tim Salimans and Jonathan Ho. “Progressive Distillation for Fast Sampling of Diffusion Models”. In: *The Tenth International Conference on Learning Representations, ICLR*. 2022.

[101] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu. “Towards data-free model stealing in a hard label setting”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 15284–15293.

7

[102] Pedro Savarese. “On the Convergence of AdaBound and its Connection to SGD”. In: *arXiv preprint arXiv:1908.04457* (2019).

[103] Abdulkadir Şeker. “Evaluation of Fabric Defect Detection Based on Transfer Learning with Pre-trained AlexNet”. In: *IDAP*. 2018, pp. 1–4.

[104] Han Shu et al. “Co-evolutionary compression for unpaired image translation”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 3235–3244.

[105] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[106] Edwin D Simpson et al. “Language understanding in the wild: Combining crowdsourcing and machine learning”. In: *Proceedings of the 24th international conference on world wide web*. 2015, pp. 992–1002.

[107] Rion Snow et al. “Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks”. In: *Proceedings of the conference on empirical methods in natural language processing*. 2008, pp. 254–263.

[108] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *International conference on machine learning*. 2015, pp. 2256–2265.

[109] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising Diffusion Implicit Models”. In: *9th International Conference on Learning Representations, ICLR*. 2021.

[110] Yang Song et al. “Consistency Models”. In: *International Conference on Machine Learning, ICML*. 2023.

[111] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *ICLR 2021* (2020).

[112] “Swiss Roll 2D”. In: *The link* (). URL: `https://github.com/joseph-nagel/diffusion-demo/blob/main/notebooks/swissroll.ipynb`.

[113] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[114] GPTZero Team. `https://gptzero.stoplight.io`. 2023.

[115] Yee W Teh, David Newman, and Max Welling. “A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation”. In: *Advances in neural information processing systems*. 2007, pp. 1353–1360.

[116] T. Tieleman and G. Hinton. *Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude*. COURSERA: Neural Networks for Machine Learning. 2012.

[117] Jean-Baptiste Truong et al. “Data-Free Model Extraction”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 4771–4780.

[118] Du-Yih Tsai, Yongbum Lee, and Eri Matsuyama. “Information Entropy Measure for Evaluation of Image Quality”. In: *J. Digit. Imaging* 21.3 (2008), pp. 338–347.

[119] Matteo Venanzi et al. “Community-based bayesian aggregation models for crowdsourcing”. In: *Proceedings of the 23rd international conference on World wide web*. 2014, pp. 155–164.

[120] Prashanth Vijayaraghavan and Deb Roy. “Generating black-box adversarial examples for text classifiers using a deep reinforced model”. In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD*. 2020, pp. 711–726.

[121] Martin J Wainwright, Michael I Jordan, et al. “Graphical models, exponential families, and variational inference”. In: *Foundations and Trends® in Machine Learning* 1–2 (2008), pp. 1–305.

[122] Xiaojie Wang et al. “KDGAN: Knowledge Distillation with Generative Adversarial Networks”. In: *NeurIPS*. 2018, pp. 783–794.

[123] Jacob Whitehill et al. “Whose vote should count more: Optimal integration of labels from labelers of unknown expertise”. In: *Advances in neural information processing systems*. 2009, pp. 2035–2043.

[124] Tong Xiao et al. “Learning from massive noisy labeled data for image classification”. In: *IEEE CVPR*. 2015, pp. 2691–2699.

[125] Yan Yan et al. “Robust semi-supervised learning through label aggregation”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.

[126] Jie Yang et al. “Leveraging crowdsourcing data for deep active learning an application: Learning intents in alexa”. In: *Proceedings of the 2018 World Wide Web Conference*. 2018, pp. 23–32.

[127] Jie Yang et al. “Scalpel-cd: leveraging crowdsourcing and deep probabilistic modeling for debugging noisy training data”. In: *The World Wide Web Conference*. 2019, pp. 2158–2168.

[128] Samuel Yeom et al. “Privacy risk in machine learning: Analyzing the connection to overfitting”. In: *IEEE CSF*. 2018, pp. 268–282.

[129] Hongxu Yin et al. “Dreaming to Distill: Data-Free Knowledge Transfer via DeepInversion”. In: *CVPR*. 2020.

[130] Li’ang Yin et al. “Aggregating crowd wisdoms with label-aware autoencoders”. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017, pp. 1325–1331.

[131] Fisher Yu et al. “LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop”. In: *arXiv preprint arXiv:1506.03365* (2015).

[132] Zhenrui Yue et al. “Black-box attacks on sequential recommenders via data-free model extraction”. In: *Proceedings of the 15th ACM conference on recommender systems*. 2021, pp. 44–54.

[133] Jing Zhang, Victor S Sheng, and Jian Wu. “Crowdsourced label aggregation using bilayer collaborative clustering”. In: *IEEE transactions on neural networks and learning systems* 30.10 (2019), pp. 3172–3185.

[134] Zhilu Zhang and Mert R. Sabuncu. “Self-Distillation as Instance-Specific Label Smoothing”. In: *NeurIPS*. 2020.

[135] Yang Zhao et al. “MobileDiffusion: Subsecond Text-to-Image Generation on Mobile Devices”. In: *arXiv preprint arXiv:2311.16567* (2023).

[136] KEXIN ZHENG. *HEART DISEASE DATASET*. 2025. DOI: 10.21227/49m4-zh81. URL: https://dx.doi.org/10.21227/49m4-zh81.

[137] Dengyong Zhou et al. “Aggregating ordinal labels from crowds by minimax conditional entropy”. In: *International Conference on Machine Learning*. 2014, pp. 262–270.

[138] Denny Zhou et al. “Learning from the wisdom of crowds by minimax entropy”. In: *Advances in Neural Information Processing Systems*. 2012, pp. 2195–2203.

[139] Helong Zhou et al. “Rethinking Soft Labels for Knowledge Distillation: A Bias-Variance Tradeoff Perspective”. In: *9th International Conference on Learning Representations, ICLR 2021*. 2021.

[140] Mingyi Zhou et al. “DaST: Data-Free Substitute Training for Adversarial Attacks”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*. 2020, pp. 231–240.

[141] Martin Zinkevich. “Online convex programming and generalized infinitesimal gradient ascent”. In: *Proceedings of the 20th international conference on machine learning*. 2003, pp. 928–936.

ACKNOWLEDGEMENTS

To those who helped me grow day by day,
To the ones who cheered me on,
And the ones who said “not yet”—and made me strong.
To Jiyue—
you always spark a little light in my life.

For every soul that wanders free,
For my two cats and my dog,
For every life that longs to see,
Curious about what’s still unknown—
I’m still on the road,
learning, searching,
figuring out the world,
and a little more of myself.

Curriculum Vitæ

Chi Hong

05-01-1993 Born in Hainan, China.

Education

2020–present Ph.D Computer Science
Delft University of Technology, The Netherlands

2015–2018 Msc. Computer Science and Technology
Tsinghua University, China

2010–2014 B.Sc Computer Science and Technology
Tianjin University, China

Work Experience

2014-2015 Software Engineer
BAIOO Family Interactive
Guangzhou, China

2018-2020 Senior Software Engineer
NetEase Games
Hangzhou, China

LIST OF PUBLICATIONS

1. 🖉 **C. Hong**, J. Huang, L.Y. Chen, and S. Roos. "Single-fold Distillation for Diffusion models." In *37th Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pp. 173-189, 2025.

2. J. Huang, **C. Hong**, L.Y. Chen, and S. Roos. "Gradient inversion of federated diffusion models." In *20th International Conference on Availability, Reliability and Security (ARES)*, pp. 380-401, 2025.

3. 🖉 **C. Hong**, J. Huang, L.Y. Chen, and R. Birke. "Adversarial Knowledge Extraction via Steering Diffusion Models." In *31th International Conference on Neural Information Processing (ICONIP)*, pp. 336-350, 2024.

4. 🖉 **C. Hong**, R. Birke, PY. Chen, and L.Y. Chen. "On Dark Knowledge for Distilling Generators" In *28th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pp. 235-247, 2024.

5. 🖉 **C. Hong**, J. Huang, R. Birke, and L.Y. Chen. "Exploring and Exploiting Data-Free Model Stealing." In *35th Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pp. 20-35, 2023.

6. J. Huang, **C. Hong**, Y. Liu, L.Y. Chen, and S. Roos. "Maverick matters: Client contribution and selection in federated learning." In *27th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pp. 269-282, 2023.

7. J. Xu, **C. Hong**, J. Huang, L.Y. Chen, and J. Decouchant. "Agic: Approximate gradient inversion attack on federated learning." In *41st International Symposium on Reliable Distributed Systems (SRDS)*, pp. 12-22, 2022.

8. 🖉 **C. Hong**, A.Ghiassi, Y. Zhou, R. Birke, and L.Y. Chen. "Online Label Aggregation: A Variational Bayesian Approach." In *30th International World Wide Web Conference (WWW)*, pp. 1904-1915, 2021.

🖉 Included in this thesis.