UNIVERSITY OF CALIFORNIA SAN DIEGO

Auditing and Mitigating Safety Risks in Large Language Models

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science (Computer Engineering)

by

Fatemehsadat Mireshghallah

Committee in charge:

Professor Taylor Berg-Kirkpatrick, Chair
Professor Julian McAuley
Professor Margaret Roberts
Professor Lawrence Saul
Professor Sameer Singh

2023

The Dissertation of Fatemehsadat Mireshghallah is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This dissertation signifies the culmination of my PhD journey. It was a journey filled with obstacles to overcome, moments of pure bliss, and endless opportunities for growth and development. However, this journey would not have been possible without the unwavering encouragement, guidance, and inspiration of numerous individuals who offered their support and assistance.

First and foremost, I would like to express my immense gratitude to my advisor Taylor Berg-Kirkpatrick for his invaluable guidance, endless support, and inspiring mentorship. I have gained numerous skills and knowledge from him that will forever shape my career. Taylor has been an exceptional role model and teacher, and I am forever indebted to him.

I would also like to express my deepest appreciation to my mentor Reza Shokri whose unwavering assistance and guidance played an instrumental role in the completion of my doctoral program academic growth. His support and encouragement helped me sustain openness to new ideas and maintain a growth-oriented research agenda.

Many thanks are due to my PhD committee members, Julian McAuley, Margaret Roberts, Lawrence Saul, and Sameer Singh for their valuable insights and feedback.

I would like to express my sincere gratitude to my current and former lab-mates, whose support and friendship have been a source of immense comfort and motivation throughout my academic journey. In particular, I am deeply indebted to Nikolai Vogler and Nikita Srivatsan, who have been a constant source of academic and personal support, offering invaluable guidance and a listening ear whenever I needed it. Without their support, this journey would have been extremely difficult. Additionally, I extend my thanks to Daniel Spokoyny and Kartik Goyal, who have been wonderful collaborators and dear friends, always ready to lend a helping hand and offer their support.

I also wish to express my heartfelt appreciation to my dear friends, who have stood by my side and provided me with unwavering kindness and support throughout my PhD

and beyond. I am especially grateful to Avni Kothari, who has been a rock and a sister to me, offering invaluable support and guidance through the numerous challenges of this journey.

I would like to acknowledge my mother, whose unconditional support and sacrifices have been instrumental in my growth. Her love and encouragement have been a constant source of inspiration and motivation. Lastly, I want to offer a special thanks to my partner during my PhD years, Kazem, whose support and understanding have been the foundation of my success. His encouragement and support have helped me navigate the challenges of my PhD with patience and confidence.

The Introduction, in part, uses material from all works listed below. The dissertation author was the primary author of all the papers in the list.

Chapter 1, in full, is a reprint of the material as it appears in the 2022 Conference on Empirical Methods in Natural Language Processing. Mireshghallah, Fatemehsadat; Goyal, Kartik; Uniyal, Archit; Berg-Kirkpatrick, Taylor; Shokri, Reza and Mireshghallah, Fatemehsadat; Uniyal, Archit; Wang, Tianhao; Evans, David; Berg-Kirkpatrick, Taylor.

Chapter 2, in full, is a reprint of the material as it appears in Advances in Neural Information Processing Systems (NeurIPS) 2022 and Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT). Mireshghallah, Fatemehsadat; Backurs, Arturs; Inan, Huseyin A; Wutschitz, Lukas; Kulkarni, Janardhan and FatemehSadat Mireshghallah; Huseyin A. Inan; Marcello Hasegawa; Victor Rühle; Taylor Berg-Kirkpatrick; Robert Sim.

Chapter 3, in full, is a reprint of the material as it appears in the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP) and Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL) 2022. Mireshghallah, Fatemehsadat and Berg-Kirkpatrick, Taylor and Mireshghallah, Fatemehsadat; Goyal, Kartik; Berg-Kirkpatrick, Taylor.

VITA

2018	Bachelor of Science, Computer Engineering, Sharif University of Technology
2020	Master of Science, Computer Science (Computer Engineering), University of California San Diego
2018–2025	Research Assistant, Department of Computer Science and Engineering University of California San Diego
2023	Doctor of Philosophy, Computer Science (Computer Engineering) University of California San Diego

PUBLICATIONS

F. Mireshghallah, M. Taram, A. Jalali, D. Tullsen, and H. Esmaeilzadeh,"Shredder: Learning noise distributions to protect inference privacy", in Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Mar. 2020.

F. Mireshghallah, M. Taram, A. Jalali, A. T. Elthakeb, D. Tullsen, and H. Esmaeilzadeh, "Not all features are equal: Discovering essential features for preserving prediction privacy", in Proceedings of The Web Conference 2021 (WWW), Apr. 2021.

F. Mireshghallah, H. A. Inan, M. Hasegawa, V. Rühle, T. Berg-Kirkpatrick, and R. Sim, "Privacy regularization: Joint privacy-utility optimization in language models", in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Jun. 2021.

T. Koker, F. Mireshghallah, T. Titcombe, and G. Kaissis, "U-Noise: Learnable noise masks for interpretable image segmentation", in 2021 IEEE International Conference on Image Processing (ICIP), Sep. 2021.

F. Mireshghallah and T. Berg-Kirkpatrick, "Style pooling: Automatic text style obfuscation for improved classification fairness", in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), Selected for Oral Presentation, Nov. 2021.

F. Mireshghallah, K. Goyal, and T. Berg-Kirkpatrick, "Mix and match: Learning-free controllable text generation using energy language models", in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (ACL, Volume 1: Long Papers), May

2022.

H. Brown, K. Lee, F. Mireshghallah, R. Shokri, and F. Tramèr, "What does it mean for a language model to preserve privacy?", in Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT), Jun. 2022.

F. Mireshghallah, V. Shrivastava, M. Shokouhi, T. Berg-Kirkpatrick, R. Sim, and D. Dimitriadis, "Useridentifier: Implicit user representations for simple and effective personalized sentiment analysis", in Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Jul. 2022.

F. Mireshghallah, A. Uniyal, T. Wang, D. Evans, and T. Berg-Kirkpatrick, "Memorization in nlp fine-tuning methods", in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), Selected for Oral Presentation, Dec. 2022.

F. Mireshghallah, K. Goyal, A. Uniyal, T. Berg-Kirkpatrick, and R. Shokri, "Quantifying privacy risks of masked language models using membership inference attacks", in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), Dec. 2022.

F. Mireshghallah, A. Backurs, H. A. Inan, L. Wutschitz, and J. Kulkarni, "Differentially private model compression", Advances in Neural Information Processing Systems (NeurIPS), Dec. 2022.

ABSTRACT OF THE DISSERTATION

Auditing and Mitigating Safety Risks in Large Language Models

by

Fatemehsadat Mireshghallah

Doctor of Philosophy in Computer Science (Computer Engineering)

University of California San Diego, 2023

Professor Taylor Berg-Kirkpatrick, Chair

Large Language Models (LLMs, e.g., GPT-3, OPT, TNLG,...) show remarkably high performance on standard benchmarks, due to their high parameter count, extremely large training datasets, and significant compute. Although the high parameter count in these models leads to more expressiveness, it can also lead to higher memorization, which, coupled with large, unvetted, web-scraped datasets can cause multiple different negative societal and ethical impacts: leakage of private, sensitive information, generation of biased, hateful or stereotypical text and much more. For example, it has been shown that given the appropriate prompt, full training set instances (including people's names and phone numbers) can be extracted

from GPT-2 [26]. One might argue that since scraped data is public data, this is not a concern. However, public data is not necessarily publicly-intended data, and there could be leaked information online [16]. Further, LLMs are shown to have disparate, unfair performance on different speakers and subjects of speech based on their properties (religion, gender identity, ethnicity, etc.), and to display bias against certain subgroups [94, 112]. For instance, [183] show that dialogue models display persona biases, attributing jobs to people based on their demographics. In this dissertation, we strive to address such problems through work falling into three main categories: first we discuss *privacy auditing* – attempting to discover different ways in which language models could leak information, and analyze memorization in such models so that we can better prevent possible ramifications [134, 140] (Chapter 1). Then, we propose *mechanisms for protecting privacy* – addressing existing threats and preventing leakage of private training data from models [130, 137] (Chapter 2), and finally, we propose *attribute controlled* text revision and generation methods to address biased and stereotypical generations and to empower users by giving them control over the attributes revealed in text written by them [132, 133] (Chapter 3).

Introduction

Since natural language is the main medium of communication among humans, language models can have a significant impact on people's daily lives. Large Language Models (LLMs) have already had a profound impact on real-world applications, such as summarizing software, search engines, language translation, conversational agents, and copywriting. However, despite the demonstrated improvement in next-word prediction loss on the test set as models increase in parameter count [77,91], some tasks do not seem to benefit from this scaling. For instance, in the deductive reasoning task of modus tollens, where the model is presented with two statements and a conclusion, model performance decreases as they get bigger [81], a phenomenon named inverse scaling. Tasks whose performance does not necessarily improve with scaling are not known ahead of time [54], and this poses a risk in terms of the reliability and failure points of such large models, as they might exhibit undesired behavior. As such, apart from benchmarking performance on target downstream tasks, LLMs need to be probed for potentially harmful behavior, as they are shown to reinforce social biases [3, 83, 177], generate offensive or toxic outputs [57], leak personally identifiable information from the training data [26], generate disinformation [20], spread falsehoods [114], and more [12, 17, 55].

As LLMs continue to increase in size and are trained on more extensive datasets, the potential for harm is expected to expand as well [54]. For example, it was shown in an attack that given the appropriate prompt, full training set instances (including people's names and phone numbers) can be extracted from GPT-2 [26]. One might argue that since scraped data is public data, this is not a concern. However, public data is not necessarily publicly-intended

data, and there could be leaked information online [16], such as leaked company emails [41] or financial documents [82]. This dissertation is a step forward in the direction of both auditing privacy and bias risks, and mitigating them in LLMs. We attribute the leakage of private data and generation of biased text to the high memorization capacity of LLMs (due to their parameter count), and the large, unvetted, web-scraped datasets that these models are trained on. Based on this, we try to address the three following questions in this dissertation:

1. How can we **audit and quantify privacy risks** of language models in terms of what information they have memorized?

2. How can we **limit the privacy risks of large language models** by limiting leakage and memorization during training?

3. How can we empower users with text revision tools that **control for different attributes** of text directly, and limit bias and privacy leakage?

To answer these questions, we first introduce a memorization quantification algorithm based on membership inference attacks, which attempts to distinguish whether a given sequence was used during the training of a model, or not. Then using this algorithm, we probe pre-trained clinical masked language models for how much of the training data they memorized. We also apply the same algorithm to fine-tuning of auto-regressive models and identify different phases in the training process, based on the privacy/utility trade-off. Then, having observed high levels of memorization in language models, we propose methods for limiting the leakage of private data during training. More specifically, we propose algorithms based on differential privacy – the gold standard in providing worst-case privacy guarantees – to produce compressed private language models. We also study other notions of privacy, which do not provide worst-case guarantees and rely on adversarial learning to protect sensitive attributes of the text during training. Finally, we take a step back and focus on natural

language itself, and not the models. What this means is we try to revise text such that we enforce certain attributes in it (for instance higher levels of agency and assertiveness in speech), and hide others (for instance the writer's identity or age). As such we propose two methods for attribute-controlled generation and evaluate them on how well they can fool attribute classifiers. The rest of this section delves deeper into how we answer these three questions in the next three chapters of the dissertation and provides a road map of what will be introduced.

1 Privacy Auditing

To be able to safely deploy language models, we need to have mechanisms and tools that can quantify the privacy leakage of these models and audit them. Auditing usually involves playing the role of an adversary and trying to discover potential weaknesses and vulnerabilities of the model by coming up with different types of attacks. Membership inference attacks are widely used privacy analysis tools that test to what extent models have memorized training data by revealing whether a given data point was used to train a given model, and what the risk for individual users is [187].

Such mechanisms exist for simpler machine learning and auto-regressive language models in abundance [22, 211]. For masked language models, however, attempts at measuring memorization have been inconclusive, due to the lack of a well-defined likelihood formulation.

For enabling auditing of masked language models, we devise a likelihood ratio-based membership inference attack [134], which uses a reference model to better distinguish training-set members from non-members and relies on an energy-based formulation to derive the likelihood ratio. We demonstrated that, unlike prior belief, MLMs do memorize samples as well, with a recall of 90%. To further probe LLMs we use this auditing method to quantify and compare memorization in different fine-tuning methods, as the fine-tuning phase is most likely to deal with private and sensitive data. Surprisingly, we find that fine-tuning only

the final layer of the model demonstrates significant leakage, larger than that of full model fine-tuning, although it updates fewer parameters. This finding is quite alarming, as head fine-tuning is commonly used for domain adaptation. We also find that early stopping is crucial for limiting unnecessary memorization in LLMs.

2 Privacy Protection and Risk Mitigation

The gold standard of privacy is Differential Privacy (DP), which provides a stringent worst-case guarantee, trying to protect all aspects of the data. To train neural networks privately, DP-SGD [1], a private variant of SGD is used which requires clipping of the gradients and the addition of noise in each update, to provide worst-case guarantees that reflect the likelihood of leaking any attribute of any member of the dataset into the trained model. The worst-case guarantees of differential privacy are not customizable, in other words, they cannot be relaxed to protect only certain attributes. Therefore, DP is shown to incur huge losses to model utility [126]. DP training of models is also slower, with cumbersome hyper-parameter tuning and development [204]. Additionally, DP's utility loss is much worse for under-represented groups [5], which can have financial and societal ramifications [157].

Exciting recent works have shown that using large pre-trained models as initialization for fine-tuning via DP-SGD yields models that are as good as non-private ones for a variety of NLP and image applications [34, 109, 127, 139, 214], and improves the privacy/utility trade-off significantly. However, there hasn't been work focusing on privately compressing large models into smaller ones, which is an important problem as compression is necessary for deploying models on edge devices. As such, we set out to understand the impact of widely used model compression algorithms such as Knowledge Distillation (KD) and pruning on the private training process. We propose private frameworks for model compression, namely DP variants of knowledge distillation and pruning, and observe that DP pruning provides a more desirable

privacy/utility trade-off.

Although differential privacy provides a worst-case guarantee (as mentioned above), it has been shown that DP mechanisms have a smoothing effect, which causes disparate utility loss in minority groups [48]. It also doesn't allow for the direct limitation of memorization for certain features/aspects of the text (for instance features relating to the author's style of writing). To address this, we propose an alternative attribute-based notion of privacy, which facilitates protecting only a given set of attributes, providing better overall model utility and circumventing DP's disparate impact on underrepresented subgroups. Unlike DP, however, these protections do not come with formal worst-case guarantees and are only shown to be effective empirically. That said, we view alternative mitigations in the privacy space to be critical, as the more options available, the more likely mitigation strategies are to be used in practice.

Based on this, we introduce a training framework for neural text generation [137] which limits the memorization of sensitive strings by protecting the "authorship" attribute of text using adversarial learning and we show how this method incurs the same level of utility drop for minority and majority users.

3 Attribute Control For Fairness and Privacy

Finally, we want to take the focus away from the models, and to the text itself. As text is used heterogeneously in a variety of high-stakes tasks such as hiring decisions, it is important to make sure that the style or implicit attributes in text do not bias humans or ML models that process this text. Since we cannot always be sure that corporations use bias removal methods, it's important to empower users to take control themselves, rather than relying solely on model-based approaches for fairness. By doing so, users won't have to depend on those who deploy the models to enforce privacy and fairness measures and can be in charge of their own data.

More specifically, we study how the style of text itself can reveal sensitive attributes

like author age and race, biasing both humans and downstream classifiers [132], and how this could be mitigated through re-writing. Unlike prior work, the re-writing mechanism we propose focuses on obfuscation/mixing of styles, rather than hiding style by mimicking other styles. We found that ML algorithms, similar to humans, are heavily influenced by the part of the sentence towards which they are *most* biased and ignore other biasing factors. Therefore, it is best to remove all attribute-revealing features in text, rather than mixing multiple attributes to create confusion.

Although style transfer as proposed above can help hide sensitive attributes, it needs the training and design of new models and architectures for each attribute we want to hide (apply control for). This training is costly, and as models get larger, end-users might not have access to the appropriate tools to do it. So having inference-only tools to protect attributes is a way of democratizing privacy and fairness. There are many existing open-source pre-trained/fine-tuned classifiers and models that can be used and glued together to apply control for desired attributes. As such, we propose a framework named "Mix and Match" [133], where, unlike prior work, the main goal is to enforce attributes using existing arbitrary pre-trained models and heuristics, whether they are discrete/continuous or differentiable/non-differentiable, without the need to perform *any* training or fine-tuning. We demonstrate the application of Mix and Match on many tasks, such as bias removal and style and formality transfer, without the need to train any new models.

4 Outline

Each of the following chapters answers one of the three main questions raised above and expands on each topic. Chapter 1 presents a membership inference attack, and applies it to pre-trained and fine-tuned models to quantify memorization and equip us with better privacy auditing tools. Chapter 2 attempts at addressing the issues of high memorization and informa-

tion leakage raised in Chapter 1, and proposes two privacy risk mitigation methods, one relying on differential privacy and focusing on model compression, and the other on adversarial learning, to limit memorization of training data during model training for sensitive pre-determined attributes. Chapter 3 takes a step back and shifts the focus to text and presents attribute control methods for revising text to enforce and control for sensitive features that might appear in it. Finally, Chapter 4 concludes the dissertation and discusses possible future directions.

Chapter 1

Privacy Auditing

As discussed in the introduction, large language models are used in various real-world tasks such as classification [169, 201, 220] and generation [158, 160]. Many of the domains in which large models are deployed are privacy/safety sensitive, such as disease diagnosis, insurance analysis on financial data and sentiment analysis for improving user experience [63, 103, 209]. These models are commonly trained using the *pre-train and fine-tune paradigm*, where they are first trained (*pre-trained*) on a large, general domain dataset (on the order of hundreds of Gigabytes), and then *fine-tuned* on smaller, task-specific datasets to adapt the model to a specific domain [79, 106, 161].

Given the sensitivity of the data used to train these models, it is crucial to conceive a framework to systematically evaluate the leakage of training data from these models [24, 139, 143, 188], and limit this leakage. The conventional way to measure the leakage of training data from machine learning models is by performing membership inference attacks [148, 189], in which the attacker tries to determine whether a given sample was part of the training data of the target model or not. These attacks expose the extent of memorization by the model at the level of individual samples. Several works have demonstrated that such large models have a high capacity for memorizing training samples during pre-training and are therefore highly susceptible to membership inference and data extraction attacks [27, 146, 218].

However, prior attempts at performing membership inference and reconstruction

attacks on pre-trained masked language models have either been inconclusive [104], or have concluded that memorization of sensitive data in MLMs is very limited and these models are more private than their generative counterparts (e.g., autoregressive language models) [84, 145, 198]. Also, apart from pre-training, scant attention has been given to fine-tuning. As such, in this chapter we will first visit memorization in pre-trained models, and propose a principled framework for measuring information leakage of MLMs through likelihood ratio-based membership inference attacks and perform an extensive analysis of memorization in such models. Then we focus on different fine-tuning methods and their propensity for memorization of training samples. Fine-tuning data is actually of higher concern than pre-training data, since most pre-training datasets are large public corpora [40, 160], while fine-tuning sets are small, targeted, and potentially very private [11, 108]. For this we modify the membership inference attack from Section 1.1, and apply it to three popular fine-tuning methods (fine-tuning all model parameters, fine-tuning adapter head and fine-tuning adapters) and analyze the results.

1.1 Quantifying Privacy Risks of Masked Language Models Using Membership Inference Attacks

BERT-based encoders with Masked Language Modeling (MLM) Objectives [37, 116] have become models of choice for use as pre-trained models for various Natural Language Processing (NLP) classification tasks [169, 201, 220] and have been applied to diverse domains such as disease diagnosis, insurance analysis on financial data, sentiment analysis for improved user experience, etc [63, 103, 209]. Given the sensitivity of the data used to train these models, it is crucial to conceive a framework to systematically evaluate the leakage of training data from these models [24, 139, 143, 188], and limit the leakage. The conventional way to measure the leakage of training data from machine learning models is by performing membership inference attacks [148, 189], in which the attacker tries to determine whether a given sample

was part of the training data of the target model or not. These attacks expose the extent of memorization by the model at the level of individual samples. Prior attempts at performing membership inference and reconstruction attacks on masked language models have either been inconclusive [104], or have (wrongly) concluded that memorization of sensitive data in MLMs is very limited and these models are more private than their generative counterparts (e.g., autoregressive language models) [84, 145, 198].

We hypothesize that prior MLM attacks have been inconclusive because they rely solely on the target model's (model under attack) loss on each individual sample as a proxy for how well the model has memorized that sample. If the loss is lower than a threshold, the sample is predicted to be a member of the training set. However, the target model's loss includes confounding factors of variation like the intrinsic complexity of the sample – and thus provides a limited discriminative signal for membership prediction. This scheme has either a high false-negative rate (with a conservative threshold) – classifying many hard-to-fit samples from the training set as non-members, or a high false-positive rate (with a generous threshold) – failing to identify easy-to-fit samples that are not in the training set.

Reference-based likelihood ratio attacks, on the other hand, when applied to certain probabilistic graphical models and classifiers, have been shown to alleviate this problem and more accurately distinguish members from non-members [144, 211]. In such attacks, instead of the loss of the model under attack, we look at the ratio of the likelihood of the sample under the target model and a reference model trained on samples from the underlying population distribution that generates the training data for the target model. This ratio recalibrates the test statistic to explain away spurious variation in model's loss for different samples due to the intrinsic complexity of the samples. Unlike most other models (e.g., generative models), however, computing the likelihood of MLMs is not straightforward. Here, we propose a principled framework for measuring information leakage of MLMs through likelihood ratio-based membership inference attacks and perform an extensive analysis of memorization in such

models. To compute the likelihood ratio of the samples under the target and the reference MLMs, we view the MLMs as energy-based probabilistic models [61] over the sequences. This enables us to perform powerful inference attacks on conventionally non-probabilistic models like masked language models.

We evaluate our proposed attack on a suite of masked clinical language models, following [104]. We compare our attack with the baseline from the prior work that relies solely on the loss of the target model [84, 190, 213]. We empirically show that *our attack improves the AUC from* 0.66 *to* 0.90 on the ClinicalBERT-Base model, and achieves *a true positive rate (recall) of* 79.2% (for a false positive rate of 10%), which is a substantial improvement over the baseline with 15.6% recall. This shows that, contrary to prior results, masked language models are significantly susceptible to attacks exploiting the leakage of their training data. In low error regions (at 1% false positive rate) *our attack is* 51$\times$ *more powerful than the prior work.*

We also present analyses of the effect of the size of the model, the length of the samples, and the choice of the reference model on the success of the attack. Finally, we attempt to identify features of samples that are more exposed (attack is more successful on), and observe that samples with multiple non-alphanumeric symbols (like punctuation) are more prone to being memorized.

1.1.1 Membership Inference Attacks

In this subsection, we first formally describe the membership inference attack, how it can be conducted using likelihood ratio tests and how we apply the test for masked language models (MLMs) which do not explicitly offer an easy-to-compute probability distribution over sequences. Finally, we describe all the steps in our attack, as summarized in Figure 1.1.

1.1.1.1 Problem Formulation

Let M_θ denote a model with parameters θ that have been trained on data set D, sampled from the general population distribution p. Our goal is to quantify the privacy risks of releasing M_θ for the members of training set D.

We consider an adversary who has access to the target model M_θ. We assume this adversary can train a (reference) model M_{θ_R} with parameters θ_R on independently sampled data from the general population p. In a Membership Inference Attack (MIA), the objective of the adversary is to create a decision rule that determines whether a given sample s was used for training M_θ. To test the adversary, we perform the following experiment. We sample a datapoint s from either the general population or the training data with a 0.5 probability, and challenge the adversary to tell if s is selected from the training set (it is a member) or not (it is a non-member) [144]. The precision of the membership inference attack indicates the degree of information leakage from the target model about the members of its training set. We measure the adversary's success using two metrics: (1) the adversary's power (the true positive rate), and (2) the adversary's error (the false positive rate).

1.1.1.2 Likelihood Ratio Test

Before discussing our proposed attack for MLMs in the next subsection, we summarize the likelihood ratio test here which forms the core of our approach. A likelihood ratio test distinguishes between a null hypothesis and an alternative hypothesis via a test statistic based on the ratio of likelihoods under the two hypotheses. Prior work demonstrated an MIA attack based on the likelihood ratio to be optimal for probabilistic graphical models (Bayesian networks) [144]. Given a sample s from the training data of the target model, the adversary aims at distinguishing between two hypotheses:

1. Null hypothesis (H_{out}): The target sample s is drawn from the general population p,

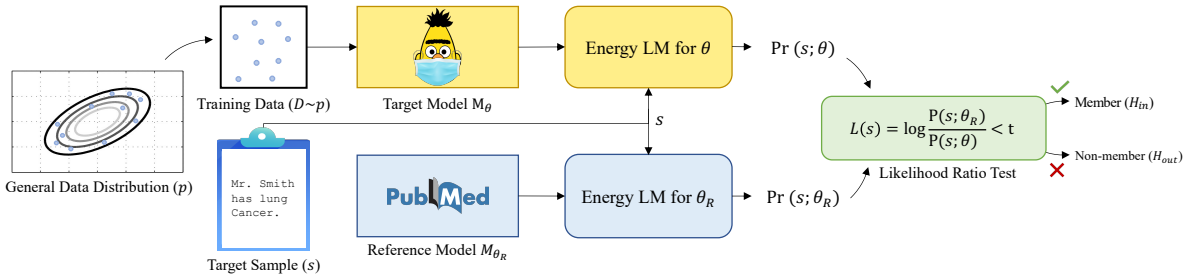


Figure 1.1. Overview of our attack: to determine whether a target sample s is a member of the training data ($D \sim p$) of the target model (M_θ), we feed it to the energy function formulation of M_θ so that we can compute $\Pr(s;M_\theta)$, the probability of s under M_θ. We do the same with a reference model M_{θ_R} which is trained on a disjoint data set from the same distribution as the training data. Then, we compute likelihood ratio $L(s)$, and based on this ratio and a given test threshold t, we decide if s is a member of D (H_{in}) or not (H_{out}).

independently from the training set D.

2. Alternative hypothesis (H_{in}): The target sample s is drawn from the target model's training set D.

The goal of hypothesis testing is to find whether there is enough evidence to reject H_{out} in favor of H_{in}. We use a likelihood ratio for this purpose which involves comparison of the likelihood of the target sample under the settings for H_{out} and H_{in} respectively. For H_{in}, we already have access to the target model, which is parameterized by θ and trained on D. For H_{out}, we require access to a model trained on the general population. As mentioned earlier, the adversary has access to a reference model parameterized by θ_R. Therefore, the likelihood ratio test is characterized by the following statistic:

$$L(s) = \log\left(\frac{p(s;\theta_R)}{p(s;\theta)}\right) \tag{1.1}$$

The Likelihood Ratio (LR) test is a comparison of the log-likelihood ratio statistic $L(s)$ with a threshold t. If $L(s) \leq t$, then the adversary rejects H_{out} (decides in favor of membership of $s \in D$); otherwise the adversary fails to reject H_{out}. We discuss the details of selecting the threshold and quantifying the attack's success in subsection 1.1.1.4.

1.1.1.3 Likelihood Ratio Test for MLMs

Performing a likelihood ratio test with masked language models is difficult because these models do not explicitly define an easy-to-compute probability distribution over natural language sequences. Following prior work [61], we alternatively view pre-trained MLMs as energy-based probability distributions on sequences, allowing us to directly apply the likelihood ratio formalism. An energy-based sequence model defines the probability distribution over the space of possible sequences $\mathcal{S}$ as:

$$p(s;\theta) = \frac{e^{-E(s;\theta)}}{Z_\theta},$$

where $E(s;\theta)$ refers to the scalar energy of a sequence s that is parametrized by θ, and $Z_\theta = \sum_{s' \in \mathcal{S}} e^{-E(s';\theta)}$ denotes the intractable noramlization constant. Under this framework, the likelihood ratio test statistic (Eq. 1.1) is:

$$\begin{aligned}
L(s) &= \log\left(\frac{p(s;\,\theta_R)}{p(s;\,\theta)}\right) \\
&= \log\left(\frac{e^{-E(s;\,\theta_R)}}{Z_{\theta_R}}\right) - \left(\log\frac{e^{-E(s;\,\theta)}}{Z_\theta}\right) \\
&= -E(s;\,\theta_R) - \log(Z_{\theta_R}) + E(s;\,\theta) + \log(Z_\theta) \\
&= E(s;\,\theta) - E(s;\,\theta_R) + \text{constant}
\end{aligned}$$

Above, we make use of the fact that for two fixed models (i.e., target model θ, and reference model θ_R), the intractable term $\log(Z_\theta) - \log(Z_{\theta_R})$ is a global constant and can be ignored in the test. Therefore, computation of the test statistic only relies on the difference between the energy values assigned to sample s by the target model M_θ, and the reference model M_{θ_R}.

In practice, we cast a traditional MLM as an energy-based language model using a slightly different parameterization than explored by [61]. Since the training of most MLMs

(including the ones we attack in experiments) involves masking 15% of the tokens in a training sequence, we define our energy parameterization on these 15% chunks. Specifically, for a sequence of length T, and the subset size $l = \lceil 0.15 \times T \rceil$, we consider computing the energy with the set $\mathcal{C}$ consisting of all $\binom{T}{l}$ combinations of masking patterns.

$$E(s;\theta) = -\frac{1}{|\mathcal{C}|}\sum_{I\in\mathcal{C}}\sum_{i\in I}\log\Big(p_{\text{mlm}}(s_i|s_{\setminus I};\theta)\Big) \tag{1.2}$$

where $s_{\setminus I}$ is the sequence s with the l positions in I masked. Computing this energy, which involves running $|\mathcal{C}| = \binom{T}{l}$ forward passes of the MLM, is expensive. Hence, we further approximate this parametrization by summing up over K random masking patterns where $K \ll |\mathcal{C}|$.

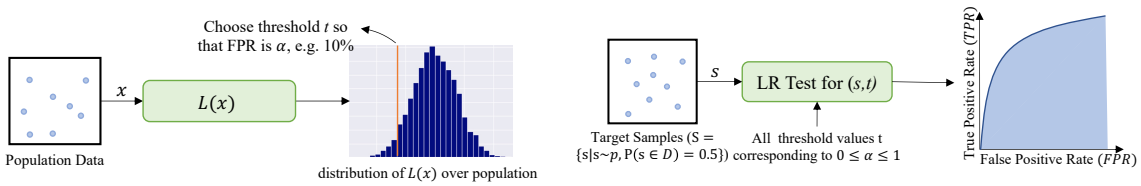
1.1.1.4 Quantifying the Privacy Risk

Given the form of the likelihood ratio test statistic (Eq. 1.2) and energy function formulation for MLM likelihood (Eq. 1.2), we conduct the attack as follows (shown in Figure 1.1):

1. Given a sample s whose membership we want to determine, we calculate its energy $E(s;\ \theta)$ under the model under attack (M_θ) using Eq. 1.2. We calculate the energy $E(s;\theta_R)$ under the reference model. Using Eq. 1.1, we compute the test statistic $L(s)$ by subtracting the two energies.

2. We compare $L(s)$ to a threshold t, and if $L(s) \leq t$, we reject the null hypothesis (H_{out}) and mark the sample as a member. Otherwise, we mark it as a non-member.

Choosing the threshold. The threshold determines the (false positive) error the adversary is willing to tolerate in the membership inference attack. Thus, for determining the threshold t, we select a false positive rate α, and empirically compute t as the corresponding percentile of the likelihood ratio statistic over random samples from the underlying distribution. This process is visualized in Figure 1.2a. We empirically estimate the distribution of the test statistic $L(x)$



(a) Selecting threshold t **(b) Plotting the ROC curve**

Figure 1.2. (a) Selecting a threshold for the attack using population data and (b) plotting the ROC curve to show the true-positive vs. false-positive rate trade-off, given different thresholds.

using all the sequences x drawn from the general population distribution. This yields the distribution of L under the null hypothesis. We then select the threshold such that the tolerance of attack's error i.e. the rate at which attack *falsely* classifies the population data as "members" is α%.

Quantifying the Privacy Risk. The attacker's success (i.e. the privacy loss of the model) can be quantified using the relation between the attack's power (the true positive rate) versus its error (the false positive rate). Higher power for lower errors indicates larger privacy loss. To compare two attack algorithms (e.g., our method versus the target model loss based methods), we can compute their power for all different error values, which can be illustrated in an ROC curve (as in Figure 1.2b and Figure 1.4). This enables a complete comparison between two attack algorithms. The Area Under the Curve (AUC) metric for each attack provides an overall threshold independent evaluation of the privacy loss under each attack.

1.1.2 Experimental Setup

We conduct our experiments using the pre-processed data, and pre-trained models provided by [104]. We use this medical-based setup as medical notes are sensitive and leakage of models trained on notes can cause privacy breaches. In this subsection, we briefly explain the details of our experimental setup. Table 1.1 provides a summary.

1.1.2.1 Datasets

We run our attack on two sets of target samples, in both of which the "members" portion is sampled from the training set (D) of our target models, which is the MIMIC-III

Table 1.1. Summary of model and baseline notations used in the results.

	Notation	Explanation
Models	Base	ClinicalBERT-base target model, trained for 300k iterations w/ sequence length 128 and 100k iterations w/ sequence length 512.
	Base++	ClinicalBERT++ target model, same as the Base model but trained for longer: trained for 1M iterations w/ a sequence length of 128.
	Large	ClinicalBERT-large target model, trained for 300k iterations w/ sequence length 128 and 100k iterations w/ sequence length 512.
	Large++	ClinicalBERT-large++ target model, same as the Large model but trained for longer: trained for 1M iterations w/ a sequence length of 128.
Methods	(A) w/ μ thresh.	Baseline with threshold set to be the mean of training sample losses (μ) (for reporting threshold-dependant metrics)
	(A) w/ Pop. thresh.	Baseline with threshold set so that there is 10% false positive rate (for reporting threshold-dependant metrics)
	(B) w/ Pop. thresh.	Our method with threshold set so that there is 10% false positive rate (for reporting threshold-dependant metrics)

dataset. The non-members, however, are different. For the results shown under "MIMIC", the non-members are a held-out subset of the MIMIC data that was not used in training. For i2b2, the non-members are from a different (but similar) dataset, i2b2. Below we elaborate on each of these datasets. Both the datasets require a license for access, so we cannot show examples of the training data.

MIMIC-III. The target models we attack are trained on the pseudo re-identified MIMIC-III notes which consist of 1,247,291 electronic health records (EHR) of 46,520 patients.

i2b2. This dataset was curated for the i2b2 de-identification of protected health information (PHI) challenge in 2014 [194]. We use this dataset as a secondary non-member dataset since it is similar in domain to MIMIC-III (both are medical notes), is larger in terms of size than the held-out MIMIC-III set, and has not been used as training data for our models.

1.1.2.2 Models

Target Models. We perform our attack on 4 different pre-trained ClinicalBERT models, that are all trained on MIMIC-III, but with different training procedures, summarized in Table 1.1 under Models.

Reference Models. We use `Pubmed-BERT` [1] trained on pre-processed PubMed texts containing around 4000M words extracted from PubMed ASCII code version [153] as our main domain-specific reference model, since its training data is similar to MIMIC-III in terms of domain, however, it does not include MIMIC-III training data. We also use the standard pre-trained `bert-base-uncased` as a general-domain reference model for ablating our attack.

[1] bionlp/bluebert_pubmed_uncased_L-12_H-768_A-12

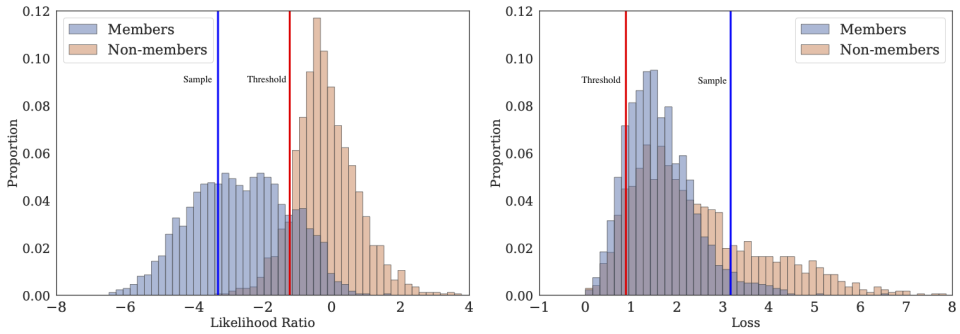
1.1.2.3 Baselines

We compare our results with a popular prior method, which uses the loss of the target model as a signal to predict membership [86, 211, 213]. We show this baseline as *Model loss* in our tables. This baseline could have two variations, based on the way its threshold is chosen: (1) μ *threshold* [84], which assumes access to the mean of the training data loss, μ and uses it as the threshold for the attack, and (2) *population threshold (pop. thresh.)* which calculates the loss on a population set of samples (samples that were not used in training but are similar to training data), and then selects the threshold that would result in a 10% false positive rate on that population.

1.1.2.4 Metrics

Area Under the ROC Curve (AUC). The ROC curve is a plot of power (true positive rate) versus error (false positive rate), measured across different thresholds t, which captures the trade-off between power and error. Thus, the area under the ROC curve (AUC) is a single, threshold-independent metric for measuring the strength of the attack. Figure 1.2b shows how we obtain the ROC curve. AUC $= 1$ implies that the attacker can correctly classify all target samples as members or non-members.

Precision and Recall. We set $\alpha = 10\%$ as the false positive rate and choose the threshold accordingly, as shown in Fig. 1.2a. For precision, we measure the percentage of samples correctly inferred as members of the training set out of the total number of target samples inferred as members by the attack. For recall, we measure the percentage of samples correctly inferred as members of the training set out of the total number of target samples that are actually members of the training set.



(a) Likelihood Ratio $L(s)$ Histogram **(b) Target Model Loss Histogram**

Figure 1.3. (a) likelihood ratio histogram for training data members and non-members. (b) loss histogram for training data members and non-members. The blue lines in the two figures correspond to the same target sample (which is a random member of training-set). The red line is is the threshold at $\alpha = 10\%$ false positive rate. The threshold from our attack (a) is correctly able to label the test sample as a training-set member but the thresholds from the baseline attack (b) fails to do so.

Table 1.2. Overview of our attack on the ClinicalBERT-Base model, using PubMed-BERT as the reference. Sample-level attack attempts to determine membership of a single sample, whereas patient-level determines membership of a patient based on all their notes. The MIMIC and i2b2 columns determine which dataset was used as non-members in the target sample pool.

		Sample-level		Patient-level	
	Non-members	MIMIC	i2b2	MIMIC	i2b2
AUC	(A) Model loss	0.662	0.812	0.915	1.000
	(B) Ours	0.900	0.881	0.992	1.000
Prec.	(A) w/ μ thresh.	61.5	77.6	87.5	100.0
	(A) w/ Pop. thresh.	61.2	79.6	87.5	92.5
	(B) w/ Pop. thresh.	88.9	87.5	93.4	92.5
Rec.	(A) w/ μ thresh	55.7	55.8	49.5	49.5
	(A) w/ Pop. thresh.	15.6	39.0	49.5	100.0
	(B) w/ Pop. thresh.	79.2	69.9	100.0	100.0

1.1.3 Results

In this subsection, we discuss our experimental results and main observations. First, we explore the overall performance improvement of our approach over baselines. Later, we analyze the effectiveness of our approach across several factors of variation that have an effect on the leakage of the model. (e.g. length of samples, model size, including names etc.) Finally, we explore correlations between samples that are deemed to be *exposed* by our approach.

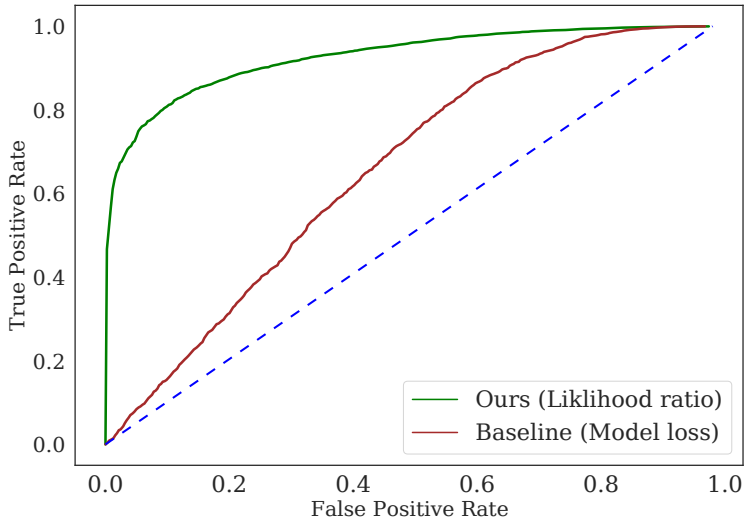


Figure 1.4. The ROC curve of sample-level attack on Clinical-BERT with MIMIC used as non-member. Green line shows our attack and the red line shows the baseline loss-based attack. The blue dashed line shows AUC=0.5 (random guess). This figure corresponds to the results presented in the first column of Table 1.2.

1.1.3.1 Comparison with Baseline

Table 1.2 shows the metrics for our attack and the baseline's on both sample and patient level, with held-out MIMIC-III and i2b2 medical notes used as non-member samples (Figure 1.4 shows the ROC curve). The table shows that our method significantly outperforms the target model loss-based baselines [84, 213], which threshold the loss of the target model based on either the mean of the training samples' loss (μ), or the population samples' loss. Our attack's improvement over the baselines is more apparent in the case where both the members and non-members are from MIMIC-III. This case is harder for the baselines since members and non-members are much more similar and harder to distinguish if we only look at the loss of the target model. Our attack, however, is successful due to the use of a reference, which helps magnify the gap in the behavior of the target model towards members and non-members, thereby teasing apart similar samples.

We can also see that in terms of precision/recall trade-off, our attack has a consistently higher recall, with an average higher precision. Population loss based thresholding ((A) w/

Table 1.3. Effect of target sample length: Sample-level attack on the ClinicalBERT-Base model, using PubMed-BERT as the reference. The MIMIC and i2b2 columns determine which dataset was used as non-members in the target sample pool. Short and long show a break down of the length of target samples.

		Short		Long	
	Non-members	MIMIC	i2b2	MIMIC	i2b2
AUC.	(A) Model loss	0.516	0.756	0.662	0.812
	(B) Ours	0.830	0.845	0.900	0.881
Prec.	(A) w/ μ thresh.	50.9	70.0	61.5	77.6
	(A) w/ Pop. thresh.	42.0	72.5	61.2	79.6
	(B) w/ Pop. thresh.	87.3	86.3	88.9	87.5
Rec.	(A) w/ μ thresh.	55.3	55.3	55.7	55.8
	(A) w/ Pop. thresh.	7.2	26.3	15.6	39.0
	(B) w/ Pop. thresh.	68.2	62.9	79.2	69.9

Pop. thresh.) has the lowest recall of 15.6%, which is due to members and non-members achieving similar losses from the target model due to their similarity. This is also shown in Figure 1.3b. In Figure 1.3a, however, we see a distinct separation between the member and non-member histogram distributions when we use the ratio statistic $L(s)$ as the test criterion for our attack. This results in the estimation of a useful threshold that correctly classifies the blue line sample as a member, as opposed to using only the target model loss (Figure 1.3b). Finally, we observe that all the metrics have higher values on the patient-level attack, compared to sample-level, for both our attack and the baselines. This is due to the higher granularity of the patient level attack, as it makes the decision based on *an aggregate* of multiple samples.

1.1.3.2 Effect of Sample Length and Model Size

Tables 1.3 and 1.9 show the metrics for our attack and the baseline broken down based on the length of the target sample, and the size and training epochs of the target model, respectively. In Table 1.3, the target model is same as that of Table 1.2, ClinicalBERT-base. Short samples are those that have between 10 to 20 tokens, and long samples have 20 to 60 tokens. We can see that both the baseline and our attacks show more leakage for long

Table 1.4. Effect of model size and training: Sample-level attack on the four different ClinicalBERT models, using PubMed-BERT as the reference and the MIMIC data as non-members. Base++ (Large++) is same as Base (Large), but trained for more epochs.

	Target Model	Base	Base++	Large	Large++
AUC.	(A) Model loss	0.662	0.656	0.679	0.700
	(B) Ours	0.900	0.894	0.904	0.905
Prec.	(A) w/ μ thresh.	61.5	61.0	62.4	64.9
	(A) w/ Pop. thresh.	61.2	61.2	63.9	69.1
	(B) w/ Pop. thresh.	88.9	88.8	88.9	88.9
Rec.	(A) w/ μ thresh.	55.7	55.8	56.4	56.1
	(A) w/ Pop. thresh.	15.6	15.6	17.6	22.2
	(B) w/ Pop. thresh.	79.2	78.5	79.2	79.3

sentences than they do for short sequences, which could be due to the longer sentences being more unique and thus being more likely to provide a discriminative signal for a sequence-level decision. Table 1.9 shows the attacks mounted on the four models from Table 1.1. We see that leakage on all the models is very similar, however, the AUC on Large++ is consistently higher than on Base, which hints at the observation made by [26] that larger models tend to have a higher capacity for memorization.

1.1.3.3 Effect of Changing the Reference Model

Table 1.5 studies how changing the reference model would affect the success of the attack. Here, *Pubmed* is the reference model that is used in the previous experiments, and *BERT-base* is Huggingface's pre-trained BERT. We observe that the attack using BERT-base performs well, but is worse than using Pubmed, especially in terms of recall (true positive rate). The main reason behind this is the domain overlap between the Pubmed reference model and the model under attack. An ideal reference model for this attack would be trained on data from a domain that is similar to that of the target model's training set so as to better characterize the intrinsic complexity of the samples. On the other hand, a reference model trained on a different data distribution (in this case Wikipedia) would give the same score to easy and

Table 1.5. Effect of reference model: Sample-level attacks on ClinicalBERT-Base model, using PubMed-BERT and standard bert-base-uncased as the reference and MIMIC data as non-member.

		Base		Large++	
	Reference Model	pubmed	bert	pubmed	bert
AUC.	(A) Model loss	0.662	0.662	0.700	0.700
	(B) Ours	0.900	0.883	0.905	0.889
Prec.	(A) w/ μ thresh.	61.5	61.5	64.9	64.9
	(A) w/ Pop. thresh.	61.2	61.2	69.1	69.1
	(B) w/ Pop. thresh.	88.9	87.8	88.9	88.0
Rec.	(A) w/ μ thresh.	55.7	55.7	56.1	56.1
	(A) w/ Pop. thresh.	15.6	15.6	22.2	22.2
	(B) w/ Pop. thresh.	79.2	71.5	79.3	72.6

difficult samples, thereby decreasing the true positive rate (recall), as shown in the table.

1.1.3.4 Effect of Inserting Names

Table 1.6 shows results for attacking the name insertion model [104], shown as Base-b, where the patient's first and last name are prepended to each training sample. We see that our attack's performance is better on the name-insertion model, compared to the base model, whereas the baseline attack performs worse (in the sample-level scenario). We hypothesize that this is due to the "difficulty" of the samples. Adding names to the beginning of each sample actually increases the entropy of the dataset overall, since in most cases they don't have a direct relation with the rest of the sentence (except for very few sentences that directly state a person's disease), therefore they might as well be random. This makes these sentences more difficult and harder to learn, as there is no easy pattern. Hence, on average, these sentences have higher loss values (2.14 for name inserted samples, vs. 1.61 for regular samples). However, for the non-members, since they don't have names attached to them, the average loss is the same (the 10% FPR threshold is 1.32), and that is why the attack performs poorly on these samples, as most of the members get classified as non-members. For our attack, since we use the reference, we are able to tease apart such hard samples as they are extremely less likely given the reference than they are given the target model.

Table 1.6. Effect of inserting names: Sample and Patient-level attacks on ClinicalBERT-Base and Base-b (name insertion) model, using PubMed-BERT as the reference and MIMIC data as non-member. We study the effect that inserting names into all training samples has on the leakage of the model.

		Sample-level		Patient-level	
	Target model	Base	Base-b	Base	Base-b
AUC.	(A) Model loss	0.662	0.561	0.915	0.953
	(B) Ours	0.900	0.960	0.992	1.000
Prec.	(A) w/ μ thresh.	61.5	53.0	87.5	100.0
	(A) w/ Pop. thresh.	61.2	44.1	87.5	91.1
	(B) w/ Pop. thresh.	88.9	90.2	93.4	92.5
Rec.	(A) w/ μ thresh.	55.7	54.0	49.5	48.5
	(A) w/ Pop. thresh.	15.6	7.8	49.5	82.8
	(B) w/ Pop. thresh.	79.2	91.3	100.0	100.0

1.1.3.5 Correlations between Memorized Samples

To evaluate whether there are correlations between samples that have high leakage based on our attack (i.e. training samples that are successfully detected as members), we conduct an experiment. In this experiment, we create a new train and test dataset, by sub-sampling the main dataset and selecting 5505 and 7461 samples, respectively. We label the training and test samples based on whether they are exposed or not, i.e. whether the attack successfully detects them as training samples or not, and get 2519 and 3283 samples labeled as "memorized", for the train and test set. Since our goal is to see if we can find correlations between the memorized samples of the training set and use those to predict memorization on our test set, we create features for each sample, and then use those features with the labels to create a simple logistic regression classifier that predicts memorization.

Table 1.7 shows these results in terms of precision and recall for predicting if a sample is "memorized" or not, with different sets of features. The first 4 rows correspond to individual handcrafted feature sets: (A) the number of digits in the sample, (B) length of a sample (in tokens), (C) the number of non-alphanumeric characters (this would be characters like '*', '-', etc.). (D) corresponds to feature sets that are obtained by encoding the tokenized sample by

Table 1.7. Analysis of correlations between samples that are leaked through our attack. We want to see what features are shared among all leaked samples by extracting a list of possible features and training a simple logistic regression model on a subset of the original training data (D), and then testing it on another subset. The logistic regression model tries to predict whether a sample would be leaked or not (based on whether our model has classified it as a member or not). The precision and recall here are those of the logistic regression model, for predicting leaked training samples.

Features	Train		Test	
	Prec.	Rec.	Prec.	Rec.
(A) #Digits	0.0	0.0	0.0	0.0
(B) Seq. Len	0.0	0.0	0.0	0.0
(C) #Non-alphanumeric	71.2	46.6	69.2	47.5
(D) 3 Least Frequent	68.9	40.5	63.8	39.2
(C) & (D)	73.9	58.8	71.1	57.8
(B) & (C) & (D)	74.3	61.3	72.1	61.3
(A) & (B) & (C) & (D)	74.3	61.3	72.1	61.3

the frequency of each of its tokens, and then taking the 3 least frequent tokens' frequencies as features (the frequency comes from a frequency dictionary built on the training set). We can see that among the hand-crafted features, (C) is most indicative, as it counts the characters that are more out-of-distribution and are possibly not determined by grammatical rules or consistent patterns. (C) and (D) concatenated together perform slightly better than (C) alone, which could hint at the effect frequency of tokens and how common they are could have on memorization. We also get a small improvement over these by concatenating (B), (C), and (D), which shows the length has a slight correlation too.

1.1.4 Related Work

Prior work on measuring memorization and leakage in machine learning models can be classified into two main categories: (1) membership inference attacks and (2) training data extraction attacks.

1.1.4.0.1 Membership inference.

Membership Inference Attacks (MIA) try to determine whether or not a target sample was used in training a target model [189,213]. These attacks can be seen as privacy risk analysis tools [87, 143, 148], which help reveal how much the model has memorized the individual samples in its training set, and what the risk of individual users is [22,117,147,174,211]. A group of these attacks rely on behavior of shadow models to determine the membership of given samples [86, 189]. [191] mounts such an attack on LSTM-based text-generation models, [123] mounts one on word embedding, [75] applies it to machine translation and more recently, [181] mounts it on transformer-based NLP classification models. Mounting such attacks is usually costly, as their success relies upon training multiple shadow models on different partitionings of shadow data, and access to adequate shadow data for training such models.

Another group of MIAs relies solely on the loss value of the target sample, under the target model, and thresholds this loss to determine membership [84, 213]. [190] mount such an attack on word embedding, where they try to infer if given samples were used in training different embedding models. [84], which is the work closest to ours, uses a thresholding loss-based attack to infer membership on MLMs. Our approach instead incorporates a reference model by using an energy-based formulation to mount a likelihood ratio based attack and achieves higher AUC as shown in the results.

1.1.4.0.2 Training data extraction.

Training data extraction quantifies the risk of extracting training data by probing a trained language model [23, 24, 26, 145, 173, 219]. One such prominent attacks on NLP models is that of [26], where they take more than half a million samples from different GPT-2 models, sift through the samples using a membership inference method to find samples that are most likely to have been memorized. [104] mount the same data extraction attack on MLMs, but their results are inconclusive as to how much MLMs memorize samples. They also mount

other types of attacks, where they try to extract a person's name given their disease, or disease given name, but in all their attacks, they only use signals from the target model and consistently find that a frequency-based baseline (i.e. one that would always guess the most frequent name/disease) is more successful.

1.1.5 Conclusion and Limitations

In Section 1.1 we introduced a principled membership inference attack based on likelihood ratio testing to measure the training data leakage of Masked Language Models (MLMs). In contrast to prior work on MLMs, we rely on signals from both the model under attack and a reference model to decide the membership of a sample. This enables performing successful membership inference attacks on data points that are hard to fit, and therefore cannot be detected using the prior work. We also perform an analysis of *why* these models leak, and which data points are more susceptible to memorization. Our attack shows that MLMs are significantly prone to memorization. This work calls for designing robust privacy mitigation algorithms for such language models.

Membership inference attacks form the foundation of privacy auditing and memorization analysis in machine learning. As we show here and as it is shown in the recent work [22, 211], these attacks are very efficient in identifying privacy vulnerabilities of models with respect to individual data records. However, for a thorough analysis of data privacy, it is not enough to rely only on membership inference attacks. We thus would need to extend our analysis to reconstruction attacks and property inference attacks.

Ethics Statement

We use two datasets here, MIMIC-III and i2b2, both of which contain sensitive data and can only be accessed by request[2] and after agreeing to the data usage and confidentiality

[2]Access can be requested through https://mimic.mit.edu/docs/gettingstarted/ and https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/

terms[3] and passing proper training for ethical and privacy-preserving use of the data.

To protect models against membership inference attacks, like the one proposed in this work, differentially private training algorithms [1, 28] can be used, as they are theoretically designed to protect the membership of each data record individually. Other methods such as adversarial training [135] and personally identifiable information scrubbing [36] can also be used, however, they do not provide the worst-case guarantees that differential privacy does [17].

Acknowledgements

The authors would like to thank the anonymous reviewers and meta-reviewers for their helpful feedback. We also thank our colleagues at the UCSD Berg Lab and NUS for their helpful comments and feedback.

[3]Data Usage Agreements (DUA) are available in https://physionet.org/content/mimiciii/view-dua/1.4/ and https://projects.iq.harvard.edu/files/n2c2/files/n2c2_data_sets_dua_preview_-_academic_user.pdf for the datasets, respectively.

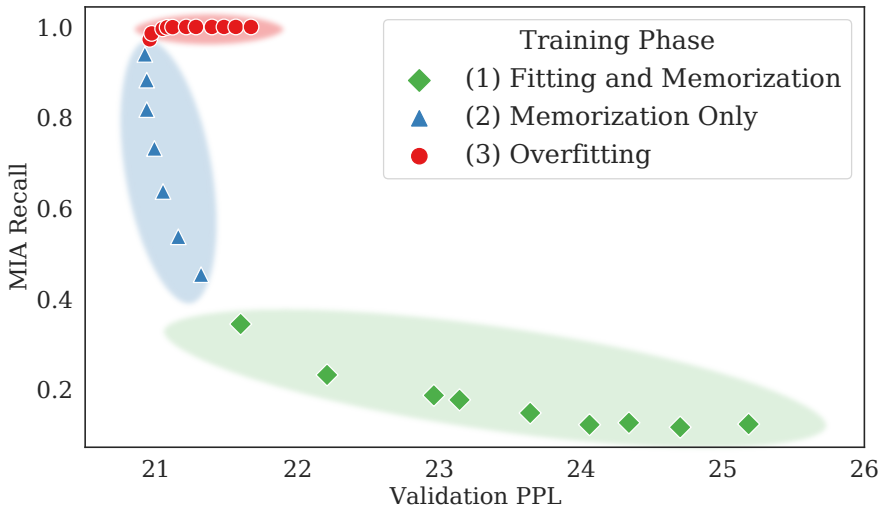


Figure 1.5. Each point in the graph shows the given metric values at the end of each training epoch. The rightmost lower points show the beginning, and as we move to left and upwards training progresses. We identify three separate phases within the learning process, distinguished by their memorization and generalization trends.

1.2 Memorization in NLP Fine-tuning Methods

Given the size of large language models, fine-tuning all the model parameters can be compute and memory-intensive [18, 50, 105]. As a result, recent works have proposed new parameter efficient fine-tuning methods that update only a subset of the model's parameters [72, 79, 106]. In this section we focus on studying memorization of three popular fine-tuning methods: (1) fine-tuning all model parameters (2) fine-tuning the head, which is commonly used by practitioners and involves updating only the last layer of the model which produces the logits, and (3) fine-tuning adapters [79], which are small bottleneck modules inserted within transformer blocks. For measuring memorization, we use two proxy metrics: (a) recall of a reference-based membership inference attack (MIA) [134] and (b) exposure [24], which measures how susceptible the model is to a sample extraction attack which tries to reconstruct samples from training data. We run our experiments on the Wikipedia [129], Penn Treebank [125] and Enron Emails [97] datasets, for the task of autoregressive language

modeling. We selected Wikipedia and Penn Treebank as they are most commonly used for fine-tuning, and Enron since it is a dataset of emails representing private tuning data.

Figure 1.5 shows how we conceptually identify three distinct phases in the fine-tuning process, based on validation perplexity (generalization) and membership inference attack recall (memorization). Each point shows these metrics at the end of a training epoch. For all fine-tuning methods, we observe that in a *memorization only* phase, the model memorizes more and more, without overfitting or generalizing better (Figure 1.6). In terms of different fine-tuning methods, we find that the common practice of fine-tuning only the head of a model has the highest memorization (by a large margin) for the same level of perplexity, among different fine-tuning methods – even full fine-tuning, which updates more parameters. This result is surprising and potentially indicates that only tuning parameters higher in the model architecture (closer to the output) exacerbates the memorization and increases the leakage based on our metrics. We also show that fine-tuning the full model and small adapters are on the Pareto-frontier in terms of the attack recall vs. validation perplexity graph.

1.2.1 Model Fine-tuning

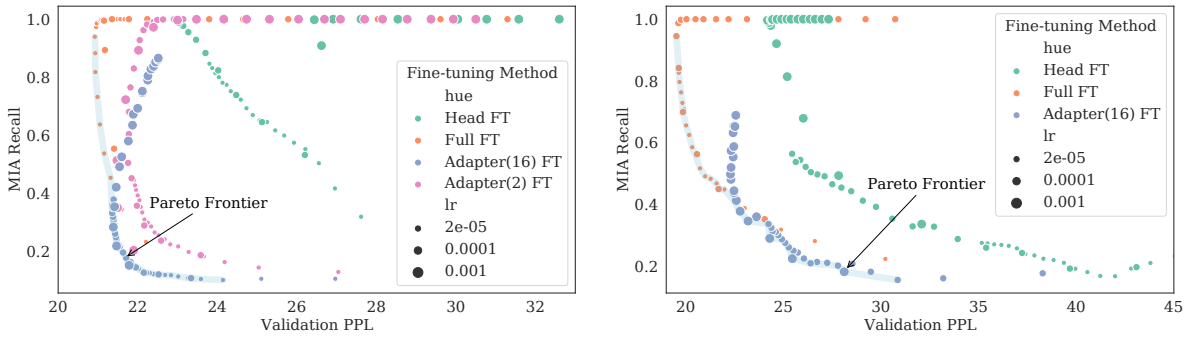
We focus on two main fine-tuning methods, for fine-tuning GPT-2 with next word prediction objective: (1) fine-tuning the model head, i.e., the prediction layer, as it is the most common method used in practice, and (2) fine-tuning adapters [79]. Adapters are small rank-restricted modules that are inserted inside transformer blocks, as added parameters and are fine-tuned for different tasks or datasets. The shape and size of the adapter module is controlled by the *reduction factor*, which determines the ratio of the size of the bottleneck to its input. During adapter tuning, the rest of the model remains frozen, therefore the number of trainable parameters is low (around 1% of the full model parameters). In our experiments, we choose reduction factors of 16 and 2, for adapters, as the former is the default used by [79,154], and the latter is the largest factor.

1.2.2 Measuring Memorization

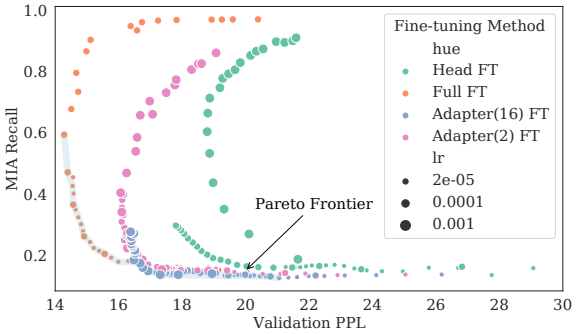
To measure memorization, we use two metrics: membership inference attack recall and exposure.

Membership Inference (MIA Recall). We use the percentage of training samples that are correctly classified as training members (out of a pool of training and validation samples) by the reference-based attack proposed in [134] and [22] as a proxy metric of memorization. For each sample x whose membership in the training set we want to determine, we feed it to the fine-tuned model, M, and get its likelihood, $\mathbf{Pr}^M(x)$. We also feed it to a reference model, R, a pre-trained model that is not fine-tuned, and get the probability $\mathbf{Pr}^R(x)$. We then use $LR(x) = \frac{\mathbf{Pr}^R(x)}{\mathbf{Pr}^M(x)}$, the likelihood ratio, to determine if x is a training sample. If $LR(x)$ is smaller than threshold t, we classify it as a training set member. Otherwise, we classify it as a non-member. We determine the threshold t by calculating $LR(s)$ for all s in the validation set, and then choose the threshold to be the highest threshold such that the false positive rate (over training and validation members) would not exceed 10%. The higher the recall of this attack is, the higher the leakage of the model.

Exposure. As a second measure of memorization, we use the exposure metric from [24] which inserts a secret (canary) of a certain format into the training data and calculates its vulnerability to extraction. Exposure is defined as the negative log-rank of the inserted secret in terms of model probability, among all other possible sequences of the same length. This quantity is then added to a constant to ensure the exposure is always positive. The lower the exposure is, the harder it is to extract the secret. In our experiments, we insert 50 copies of the phrase "the secret number is 940955" into the training data to accentuate the differences between the fine-tuning methods. For a six-digit secret, an exposure of around $\log_2(10^6) \approx 20$ means the canary can be reliably extracted from the model.



(a) Wikipedia Dataset **(b) Penn Treebank Dataset**



(c) Enron Dataset

Figure 1.6. Pareto frontier for utility (validation PPL) Vs. privacy (MIA recall). Each dot shows different checkpoints, and the colors show different fine-tuning methods. We desire models that have low PPL and low attack recall.

1.2.3 Experimental Setup

Datasets. (1) Huggingface's Wikipedia wikitext-2-raw-v1 dataset, consisting of 36718 training samples (2) Huggingface's Penn Treebank ptb_text_only, consisting of 42068 training samples and (3) a sub-sampled version of Enron email dataset consisting of 7180 emails. We use a sequence length of 1024, training batch size of 8, and fine-tune for 20 epochs.

Models. We study memorization in fine-tuning Huggingface's pre-trained GPT-2 on the datasets mentioned above. We use a pre-trained but not fine-tuned GPT-2 as the reference model for our membership inference attack. We use the adapter hub's implementation of the Pfeiffer architecture, with reduction factors 2 and 16 [154].

Metrics. We use *Validation Perplexity* as a metric for the performance of the model, where

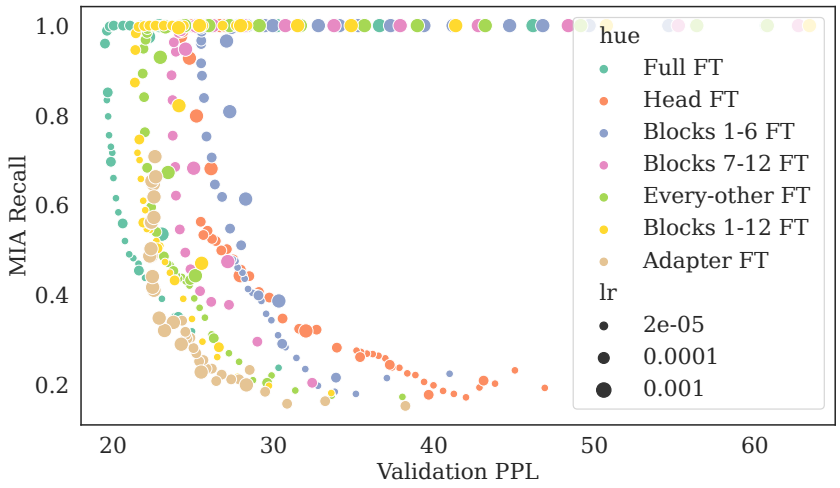


Figure 1.7. Ablating how the location and number of trainable parameters effects memorization on the Penn Treebank dataset. Each dot shows different checkpoints, and the colors show different fine-tuning methods. We desire models that have low PPL and low attack recall.

lower perplexity is better. We evaluate memorization at each epoch using the *MIA recall* and *exposure* metrics described in subsection 1.2.2. The experiments here are all repeated 3 times and we report the average values for each metric.

Hyperparameters and result presentation. We run optimization for each fine-tuning method for 20 epochs, and perform evaluation of the mentioned metrics at the end of each epoch. We experiment with the three learning rates 2×10^{-5},10^{-4},10^{-3}, and present the results for all of them. Therefore, each graph would have an overall of 20×3 points, for each fine-tuning method, unless the point is outside the plot range. For the reported exposure numbers, we selected points close to the pareto frontier to present in Table 1.8, to summarize results.

1.2.4 Results

In this subsection, we discuss our experimental results comparing the privacy-utility trends for different fine-tuning methods. We refer to the naming convention shown in Figure 1.5.

Table 1.8. Exposure metric. Higher exposure indicates more leakage, and exposure above 20 means the secrets (canaries) are reliably extractable. The perplexity numbers here are different from the ones in other experiments since the training data is diluted with the artificially inserted secrets.

		Full FT	Head FT	Adapters (2)	Adapters (16)
Parameters (Millions)		124.440	38.590	7.092	0.895
Wiki	Val PPL	**24.82**	28.76	24.41	25.26
	Exposure	1.42	10.78	14.54	**0.83**
PTB	Val PPL	29.55	31.24	29.79	**29.41**
	Exposure	7.03	12.0	12.40	**4.54**
Enron	Val PPL	**12.52**	13.51	13.03	12.81
	Exposure	1.32	10.77	2.02	**0.440**

Table 1.9. Comparison of fine-tuning different transformer blocks on the Wikipedia dataset.

	Block 1	Block 5	Block 8	Block 12	Full FT	Head FT	Adapters (2)	Adapters (16)
Validation PPL	24.39	23.35	23.36	24.05	23.05	23.93	23.62	**21.75**
MIA Recall	22.2	22.6	20.8	21.3	19.2	81.6	16.8	**15.2**
#Params (in Millions)	7.088	7.088	7.088	7.088	124.440	38.590	7.092	0.895

1.2.4.1 Memorization of Fine-tuning Methods

Figures 1.6a, 1.6b, 1.6c compare the fine-tuning methods in terms of privacy leakage, measured by MIA recall and Table 1.8 shows the exposure results for the three datasets, along with their parameter counts. The blue lines show the Pareto frontier, marking the desirable trade-off points, with low recall and PPL.

1.2.4.2 Shared Trends

The "memorization only" phase in training, where validation perplexity (generalization) is stable and the model has not yet overfit, is also observed by [195] in pre-trained BERT-based classifiers. However, it is named the "settling phase" there, and it is suggested that as *validation perplexity* is rather stable, early stopping is not important and training can stop at any point before overfitting. We, however, show that *memorization* is actually increasing during that phase. Therefore, if we are optimizing for privacy as well, it is best to stop training earlier. For all the methods, across all datasets, in the "fitting+memorization" and the "memorization

only" phases, we see an increase in memorization, without any overfitting. This shows that we can have high memorization/learning, and still not overfit. This is also observed for training large language models from scratch in [197], which focuses on analyzing the effect that text type (e.g., part of speech, numbers), data size and model size have on memorization when training from scratch.

1.2.4.3 Comparison of Fine-tuning Methods

Results for both the MIA recall and exposure metrics (Figure 1.6 and Table 1.8) are consistent, showing higher leakage for head fine-tuning and lower for full model fine-tuning and adapters. The first observation here is that head fine-tuning is an outlier, with extremely high leakage, on all three datasets. We can also see that the validation perplexity achieved by this method is consistently lower than the other methods. We hypothesize that the high leakage of fine-tuning the head is due to both the high number of parameters (38 million) and the location of the parameters, right at the last layer of the model where the next word prediction happens. While full fine-tuning actually touches more parameters than head fine-tuning, it leads to less leakage under the attacks we investigate. This result is somewhat surprising and potentially indicates that tuning parameters lower in the model architecture mitigates some of the explicit memorization performed by the head. We further study this phenomenon and ablate it in subsection 1.2.4.4.

We also observe that for a low-perplexity regime (without considering the cost), full fine-tuning is the best choice as it offers utility superior to adapters. However, if we have tolerance for higher perplexity, to get lower leakage, opting for adapters with a reduction factor of 16 appears better as it has lower MIA recall and a lower propensity for overfitting, compared to the other methods. One final observation is that full-finetuning has the shortest "fitting+memorization" phase, whereas head fine-tuning has the longest.

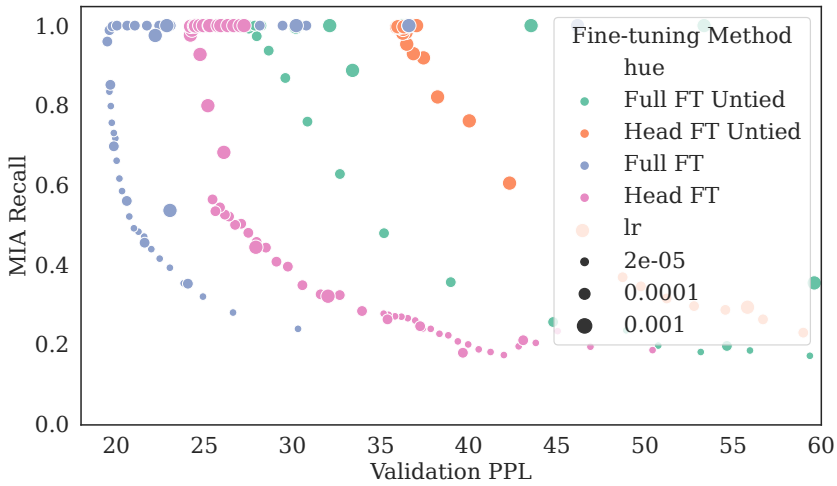


Figure 1.8. Ablating how the untying of the trainable parameters effects memorization on the Penn Treebank dataset. Each dot shows different checkpoints, and the colors show different fine-tuning methods. We desire models that have low PPL and low attack recall.

1.2.4.4 Parameter Count, Location and Tying

To further test our hypothesis that the privacy-utility trade-off has to do with *both* trainable parameter count and location/distribution within the model architecture (subsubsection 1.2.4.3), we run experiments with the following set of trainable parameters: (1) first half: blocks 1–6 of the 12 transformer blocks of the GPT2 model (42M trainable params), (2) second half: blocks 7–12 (42M), (3) every other block (42M) and (4) entire body: all the 12 blocks (84M). In all these scenarios we freeze the head and fine-tune only the blocks. As shown in Figure 1.7, we find that Full FT $>$ Adapters $>$ all 12 blocks=every other block $>$ blocks 7 to 12 $>$ blocks 1 to 6 $>$ Head FT, in terms of privacy-utility trade-off desirability. Based on this, we argue that how the trainable parameters are scattered in the network affects how well the model makes progress in the first phase (the training and fitting phase), which affects the validation perplexity when it enters the second phase (memorization-only phase). As Figure 1.6 also shows, full fine-tuning and adapter tuning make faster progress and end up in a lower perplexity.

Figure 1.8 shows an ablation study of how untying model parameters affects the privacy-utility trade-off. By untying parameters, we mean creating a separate set of parameters

for the head of the model and the input embeddings, as by default these two parameter sets are tied in GPT2, meaning the same set of 38.59 Million parameters are used for both these components. However, in the untied scenario, we first duplicate them, and then create separate trainable parameters, adding an extra set of 38.59 Million trainable parameters to the model. As the figure shows, tying the parameters improves the progress in training and puts the model at an advantage, compared to untying them, creating a better overall privacy-utility trade-off.

1.2.4.5 Fine-tuning Single Transformer Blocks

To have a full analysis of fine-tuning leakage, we also look at fine-tuning individual adapter blocks and freezing the rest of the model. The GPT-2 model has 12 blocks, and we experiment with fine-tuning the first, 5th, 8th, and 12th block, to cover different positions within the model. Table 1.9 shows the results for this experiment. We have selected the numbers such that the validation PPLs are as similar as possible. There does not seem to be any significant difference between fine-tuning different blocks, as they all manifest similar attack recalls. Block 8's recall, however, is lower than other blocks, with lower PPL, which would make it the most desirable block for fine-tuning in terms of the PPL-leakage trade-off. With respect to privacy-utility tradeoffs, fine-tuning full blocks seems less desirable than using adapters or fine-tuning the entire model.

1.2.5 Conclusion

When fine-tuning is done using sensitive training data, it is important to not just consider the cost and utility of fine-tuning methods but to also be aware that they may have different risks in terms of privacy. Our experiments show that the common practice of fine-tuning only the head of a model has the highest memorization (by a large margin). Full model fine-tuning and adapter tuning, however, are both on the Pareto-frontier in terms of attack recall vs. validation perplexity, suggesting that they are more suitable when privacy is a concern.

Acknowledgements

This project is funded in part by the NSF under grant 2200333. The authors would like to thank the anonymous reviewers and meta-reviewers for their helpful feedback. We also thank Nikolai Vogler, Nikita Srivatsan, and Kazem Taram for insightful discussions. Additionally, we thank our colleagues at the UCSD Berg Lab and UVA Security Research Group for their helpful comments and feedback.

Limitations and Ethics Statement

In our study we focus on autoregressive language models – specifically GPT-2, as it has been shown to be more prone to memorizing samples than pre-trained masked language models (MLM) [26, 104] Also, here we loosely refer to the recall of the membership inference attack on the training set as memoirzation. However, we need to keep in mind that a low attack recall does not necessarily mean low memorization, and there might be stronger attacks (of other types, such as reconstruction) that can better uncover memorization in language models.

In this work we have used publicly available datasets and have not collected any sensitive/private data. The ultimate goal of our study is to contribute to analyzing memorization under different fine-tuning paradigms, thereby advancing our intuition of how we can better deploy private, fair and safe language models.

Chapter 2

Privacy Protection and Risk Mitigations

As mentioned before, LLMs are first *pre-trained* on extremely large and diverse publicly available datasets, and are then *fine-tuned* for a specific task of interest using a much smaller *private dataset*, which may consist of sensitive information about the users. As such, protective measures need to be taken to limit the leakage of such sensitive data through the trained model parameters, as shown in the previous chapter. Over the past few years, training deep learning models guaranteeing differential privacy (DP) [43], a strong notion of data privacy, has emerged as the defacto method to mitigate such information leakage. This is achieved using the DP-SGD [1] algorithm, which is a differentially private variant of the SGD optimizer. Each optimization step in DP-SGD consists of computing *per-example* gradients, clipping them and then adding noise to them before updating the model parameters. Per-example gradients are needed for DP-SGD (as opposed to per mini-batch gradients for conventional SGD), as DP guarantees are on a "record" level, meaning the membership of every single data sample is protected, hence we need to clip gradients from each sample separately, to limit leakage. The strong guarantees of DP come at the price of huge loss of model utility [126], specially in larger models, due to the high dimension of the added noise [107]. Recent works, however, have shown that DP-SGD fine-tuning of pre-trained models, as opposed to training randomly initialized models from scratch, yields private models that are as good as non-private ones for a variety of NLP and image applications [34, 109, 127, 139, 214]. Although these large models

provide desirable privacy/utility trade-offs, they are too large to be widely deployed, specially on edge devices.

As such, in some cases, before deploying the large models are compressed (also referred to as sparsified or distilled) to reduce the parameter count [64]. There is a large body of work on model compression, however, no prior work explores private compression of large models. In this chapter we first aim to understand how private training impacts the modern deep learning pipeline of pre-train, fine-tune, and compress. The main observation is that most widely used model compression algorithms such as Knowledge Distillation (KD) and Pruning, use private datasets to produce compressed models, hence if our goal is to deploy a differentially private compressed model, we should consider their impact on the training process. We propose frameworks for private model compression in the context of NLP applications, and observe that private model pruning provides a better privacy-utility trade-off compared to private knowledge distillation.

Although DP provides stringent worst-case guarantees, it has been shown to have a smoothing effect, which causes disparate utility loss in minority groups [48]. It also doesn't allow for direct limitation of memorization for certain features or attributes of the text (for instance features relating to the author's style of writing that we might want to protect). To address this, we can adhere to an attribute-based notion of privacy, which aims at protecting only certain attributes and does not offer worst-case guarantees. It does however, have uniform impact on model utility. Based on this, we introduce a training framework for neural text generation which limits the memorization of sensitive strings by protecting the "authorship" attribute of text using adversarial learning, and show how this method incurs the same level of utility drop for minority and majority users.

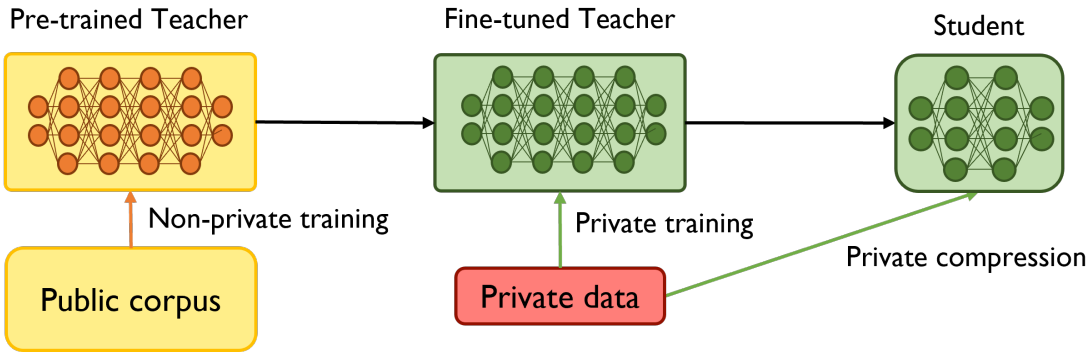


Figure 2.1. The 3-pronged modern deep learning pipeline: Pre-train on public data, fine-tune on private data, and compress the model to meet the memory and latency requirements of specific applications.

2.1 Differentially Private Model Compression

The main purpose of this section is to understand how private training impacts the modern deep learning pipeline of pre-train, fine-tune, and compress (see Figure 2.1). The main observation is that most widely used model compression algorithms such as Knowledge Distillation (KD) and Pruning, use private datasets to produce compressed models, hence if our goal is to deploy a differentially private compressed model, we should consider their impact on the training process. This leads to the question:

What algorithms should one use to produce compressed private models and how do they impact private fine-tuning via DPSGD?

The goal of this section is to investigate this question and propose frameworks for private model compression in the context of NLP applications. Although we investigate model compression techniques at the fine-tuning stage using pre-trained models, we would like to emphasize that our frameworks for private model compression are not tied to this setting. They are equally applicable to training deep learning models from scratch and to other application domains such as image classification tasks.

2.1.1 Our Contributions

- We give a framework for doing model compression using Knowledge Distillation algorithm guaranteeing differential privacy, which we call DPKD. We show that DPKD alone is not enough to transfer the knowledge from large models to compressed models. This loss in the accuracy of compressed models can be mitigated by better initialization of compressed models from the weights of the large models, which itself is a form of knowledge transfer. We propose several *zero-shot, fully private* methods for initialization compressed models using weights of the large models. Our empirical evaluation of these ideas on standard GLUE benchmarks using BERT models show that DPKD approach to the model compression loses an accuracy of 5% compared to the larger models if the compressed model has *half the size of the full BERT model*.

- To overcome the limitations of DPKD algorithm for model compression, we consider a framework for evolving the larger models to compressed models via private adaptation of Iterative Magnitude Pruning (DPIMP). We show that on standard GLUE benchmarks using BERT models, DPIMP framework produces compressed models whose performance is comparable to larger models at 50% unstructured sparsity levels.

- As a byproduct of DPIMP approach for model compression, our work also shows that pre-trained BERT models have sparse subnetworks that can be found via DPSGD that have *almost* matching performances of the private full model, similar to the Lottery Ticket Hypothesis for BERT models in non-private settings [30, 53].

To the best of our knowledge, no prior work have studied model compression techniques of LLMs in private settings. A problem broadly related to model compression is ensemble learning, where the goal is to transfer knowledge from an ensemble of teacher models to a single student model [39]. This problem was studied in the private setting

by [150, 151], who proposed the Private Aggregation of Teacher Ensembles (PATE) framework. In PATE an ensemble of teacher models is trained on *disjoint private data* and the student model is trained by noisy aggregation of teachers' answers. Two recent works combine PATE framework with KD algorithm for doing noisy aggregation of teachers' answers for mobile analytics and text generation problems [119, 196]. The PATE framework and ensemble learning techniques can be applied for fine-tuning (or training) deep learning models. Unfortunately, however, as previous works have shown, the performance of deep learning models trained via PATE are inferior to that of DPSGD for complex datasets [215]. As the performance of a fine-tuned large model on a sensitive dataset is an upper bound on the performance of a compressed model, and we do not know how to fine-tune large models using PATE to match the performance of DPSGD, we do not consider PATE framework for model compression.

2.1.2 Preliminaries

Recall the formal definition of differential privacy.

Definition 2.1.1 (Differential Privacy (DP) [42,43])**.** *A randomized algorithm $\mathcal{A}$ is (ϵ,δ)-differentially private if for any two neighboring datasets D and D', which differ in exactly the data pertaining to a single user, and for all sets $\mathcal{S}$ of possible outputs:*

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

We train all our models via DPSGD as the optimizer. We briefly describe the algorithm.

2.1.2.1 Training via DPSGD

To train a deep learning model with privacy, the most widely used algorithm is the DP stochastic gradient descent (DPSGD) [2, 10, 175, 193]. DPSGD augments the standard SGD algorithm with per-example gradient clipping and Gaussian noise addition steps. These two steps serve to limit and mask the contribution of a single example. At a high level, the privacy

analysis of DPSGD proceeds by first showing that each iteration of DPSGD is differentially private for some (ϵ,δ), then applying amplification by subsampling and composition across all the iterations. To get the tightest privacy parameters, however, one needs more sophisticated arguments such as the Moments Accountant method [2] or numerical composition algorithms [58].

2.1.3 Problem Statement

Input to our problem are privacy parameters $\epsilon > 0$, $\delta > 0$, a large model M_A with the initial model parameters $\theta_A(0)$, a private dataset D corresponding to a downstream task we want to solve, and a compression factor γ. Let $|M_A|$ denote the parameter count of M_A. Our goal is to produce a compressed model M_B satisfying two constraints: (i) $|M_B| \leq \gamma \cdot |M_A|$ and (ii) the final weights of model M_B (denoted by $\theta_B(t)$) should be (ϵ, δ)-differentially private with respect to dataset D. A compression algorithm can make use of M_A in an arbitrary way as long the final weights of model M_B ($\theta_B(t)$) are differentially private with respect to dataset D.

We measure the quality of compression algorithms by comparing the accuracy obtained by M_B satisfying (ϵ, δ)-DP on downstream task D to the accuracy obtained by M_A satisfying (ϵ, δ)-DP on downstream task D. This allows us to quantify how much performance one loses in private training due to model compression. Note that we are not comparing against the performance of non-private models. We would like to find compression algorithms where differentially private M_B has nearly the same performance as differentially private M_A.

2.1.4 Compressed Models via Knowledge Distillation

One of the most widely used algorithms for compressing models is knowledge distillation (KD) [21, 74, 170]. In this subsection, we propose a framework for implementing knowledge distillation with DP constraints and evaluate its effectiveness on standard GLUE benchmarks. We begin by briefly describing how the KD algorithm is applied for compressing models; we refer the readers to [59, 128] for more details. Adopting the naming convention

from the literature, for the rest of the section, we call large pre-trained models as *teacher models* and compressed smaller models as *student models*.

2.1.4.1 Non-Private Knowledge Distillation

Let $\mathcal{T}$ be a teacher network with the class probabilities $P_{\mathcal{T}} = \text{softmax}(a_{\mathcal{T}})$ (a.k.a. soft labels) where $a_{\mathcal{T}}$ is the output of last layer before the softmax operation. Similarly, let $\mathcal{S}$ be a student network with parameters $W_{\mathcal{S}}$ and class probabilities $P_{\mathcal{S}} = \text{softmax}(a_{\mathcal{S}})$. The main idea behind KD algorithm is to train $\mathcal{S}$ to mimic the output distribution of the teacher $P_{\mathcal{T}}$ and the true labels. The intuition is that $P_{\mathcal{T}}$ captures the knowledge learnt by the teacher, in particular probabilities assigned by the teacher to labels that are different from the true label. Hinton et al. [74] suggested to use softmax-temperature where probability for class i of the teacher is given by

$$p_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

with logits z_i where T controls the smoothness of the output distribution. Setting a higher value for the temperature parameter T produces a softer probability distribution over classes. The same relaxation is applied to the output of the student network. The student is trained to minimize the weighted combination of the distillation loss and the supervised training loss:

$$\mathcal{L}_{\text{KD}}(\mathcal{W}_{\mathcal{S}}) := \mathcal{H}(\mathbf{y}_{\text{true}}, P_{\mathcal{S}}) + \lambda \cdot \mathcal{H}(P_{\mathcal{T}}, P_{\mathcal{S}}) \tag{2.1}$$

where $\mathcal{H}$ refers to the cross-entropy and λ is a hyperparameter.

2.1.4.2 Differentially Private Knowledge Distillation (DPKD)

A natural way to generalize KD algorithm for private distillation of student model is to train the student with DPSGD to minimize the loss function given by Equation 2.1. However, such an algorithm fails to produce student models satisfying DP because of the

term $\mathcal{H}(P_{\mathcal{T}},P_{\mathcal{S}})$ in Equation 2.1. Note that $P_{\mathcal{T}}$ is a function of the entire dataset, hence clipping and adding noise alone is not enough to argue that DPSGD produces a private student. A natural solution to overcome this hurdle is to first train the teacher models with DPSGD and then apply KD. We propose our DPKD framework in Algorithm 1. In this Algorithm, if the initialization of student model weights does not incur any privacy cost (e.g. random initialization or initialization using parameters from a publicly trained model), ϵ_2 would be 0. Having said that, there could be student initialization strategies that are functions of the dataset D, in which case, we need to account for the privacy loss using non-zero ϵ_2.

Algorithm 1. Differentially Private Knowledge Distillation (DPKD)

Input: Teacher model $\mathcal{T}$, student model $\mathcal{S}$, private data D, privacy budget (ϵ,δ)

Output: Student model $\mathcal{S}$ satisfying (ϵ,δ)-DP

1: Find an allocation of (ϵ_1,δ_1), (ϵ_2,δ_2) and (ϵ_3,δ_3) from the privacy budget (ϵ,δ)

2: Train $\mathcal{T}$ on D with DPSGD using privacy budget of (ϵ_1,δ_1)

3: Initialize $\mathcal{S}$ (possibly privately with privacy budget (ϵ_2,δ_2))

4: Train $\mathcal{S}$ on D to minimize Eq. (2.1) with DPSGD using privacy budget of (ϵ_3,δ_3)

5: **return** $\mathcal{S}$

As the model parameters produced by DPSGD satisfy privacy guarantees, using the post-processing property of DP [44], one can show the following theorem. We omit the proof.

Theorem 2.1.2. *The output of DPKD algorithm is differentially private with privacy parameters obtained by the adaptive composition of privacy parameters in steps 2, 3, and 4 of Algorithm 1.*

In this work we use numerical composition of privacy mechanisms as given in [58]. Our framework for DPKD raises several interesting algorithmic and hyperparameter tuning questions. The most interesting one is whether one can have DPKD algorithm where the privacy budget is not *wasted* on training the teacher. While this is a nice theoretical question,

Table 2.1. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models with random initialization against full 12-layer BERT teacher models. The first row indicates the performance of fine-tuning the full teacher model. All our models have the same privacy budget $\epsilon = 4$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	77.8	84.7	87.8	90.5	85.2
$\frac{1}{2}$-BERT	Random	-	Finetune	55.1	74.0	59.4	69.7	64.5
$\frac{1}{2}$-BERT	Random	BERT	DPKD	53.9	73.1	59.2	65.4	62.9

we show in our experiments that DPKD framework is competitive with respect to teachers that are *not trained with DP*.

2.1.4.3 Empirical Evaluation of DPKD Algorithm

In this subsection, we perform experiments to evaluate the effectiveness of DPKD algorithm for compressing models.

Teacher and Student Architectures. Our teacher models are pre-trained BERT [1] models, which consists of 12 transformer blocks. The architecture of compressed models consist of 6 transformer blocks, which we refer to as $\frac{1}{2}$-BERT.

Tasks and datasets. Following prior work [109, 214, 216], we experiment with the following set of 4 tasks from the GLUE benchmark [201]: MNLI (Multi-Genre Natural Language Inference Corpus), QQP (Quora Question Pairs), QNLI (Stanford Question Answering Dataset) and SST-2 (Stanford Sentiment Treebank).

Training and privacy parameters. We perform experiments with two sets of privacy budgets: (i) $\epsilon = 4$ with $\delta = \frac{1}{N}$ and (ii) $\epsilon = 1$ with $\delta = \frac{1}{10N}$, where N is the number of samples in the given dataset.

We start with *random initialization* of the student models to see if DPKD algorithm can be effective in transferring the knowledge from the teacher. We compare the performance of

[1] We use Huggingface's `bert-base-uncased`.

Table 2.2. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models with random initialization against full 12-layer BERT teacher models. The first row indicates the performance of fine-tuning the full teacher model. All our models have the same privacy budget $\epsilon = 1$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	74.8	82.1	85.6	86.8	82.3
$\frac{1}{2}$-BERT	Random	-	Finetune	49.6	72.6	57.8	51	57.7
$\frac{1}{2}$-BERT	Random	BERT	DPKD	46.4	70.4	52.9	52	55.4

our student model trained with DPKD algorithm with the performance of directly fine-tuned student via DPSGD and the performance of full teacher model fine-tuned with DPSGD. Our results are summarized in Tables 2.1 and 2.2. The main takeaway from this experiment is:

- There is a large gap in the performance of students trained using DPKD algorithm compared to the teacher when students are randomly initialized. In fact, directly fine-tuning the student model using DPSGD achieves better performance compared to DPKD algorithm.

We conclude that DPKD alone is not enough to transfer the knowledge from teacher models to compressed student models.

2.1.4.4 Better Student Models via Zero-shot Initializations

In our desire to train better-performing student models, we explore different initialization strategies. We note that initialization is also a form of knowledge transfer. Here we consider two natural zero-shot initialization strategies:

- Zero-shot (PT): Here we initialize the student model using the weights of the pre-trained teacher. In particular, we simply initialize the layers of the student model with 6 layers of BERT teacher model. We follow [175] for the choice of layers.

- Zero-shot (FT): Here we initialize the student model using weights of the privately fine-tuned teacher model. As our DPKD requires the teacher to be private as well, one can

Table 2.3. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models against full 12-layer BERT teacher models and pre-trained DistilBERT, under various initialization strategies. For every student initialization method, we compare fine-tuning using DPKD algorithm vs full fine-tuning via DPSGD. All our models have the same privacy budget $\epsilon = 4$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	77.8	84.7	87.8	90.5	85.2
$\frac{1}{2}$-BERT	Zero-shot (PT)	-	Finetune	71.7	82.4	83.2	82.7	80.0
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT	DPKD	72.8	82.6	83.0	82.7	80.3
$\frac{1}{2}$-BERT	Zero-shot (FT)	-	Finetune	71.3	81.8	83.4	82.2	79.7
$\frac{1}{2}$-BERT	Zero-shot (FT)	BERT	DPKD	72.3	82.1	82.9	82.6	80.0
DistilBERT	Pretrained	-	Finetune	73.0	84.3	82.8	87.7	81.9
DistilBERT	Pretrained	BERT	DPKD	72.9	83.7	83.0	86.6	81.5

Table 2.4. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models against full 12-layer BERT teacher models and pre-trained DistilBERT, under various initialization strategies. For every student initialization method, we compare fine-tuning using DPKD algorithm vs full fine-tuning via DPSGD. All our models have the same privacy budget $\epsilon = 1$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	74.8	82.1	85.6	86.8	82.3
$\frac{1}{2}$-BERT	Zero-shot (PT)	-	Finetune	66.9	78.3	81.0	79.6	76.4
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT	DPKD	67.5	78.4	80.1	78.5	76.1
$\frac{1}{2}$-BERT	Zero-shot (FT)	-	Finetune	66.9	77.6	80.1	79.2	75.9
$\frac{1}{2}$-BERT	Zero-shot (FT)	BERT	DPKD	68.3	77.0	80.3	80.0	76.4
DistilBERT	Pretrained	-	Finetune	68.4	82.0	81.0	86.0	79.3
DistilBERT	Pretrained	BERT	DPKD	68.1	80.5	80.2	85.1	78.5

initialize the student model using the weights of the private teacher without incurring any additional privacy cost.

We compare zero-shot initialization strategies against a fully pre-trained student model given by Huggingface's compressed BERT model DistilBERT [175]. DistilBERT is trained using KD algorithm with the full BERT model on public data at the *pre-training stage*. We note

that DistilBERT has 6 transformer blocks and hence has the same architecture as $\frac{1}{2}$-BERT. We emphasize that pre-training student model is computationally expensive, and the main aim of this section is to develop algorithms where one can transfer knowledge from the teacher to students models without resorting to pre-training the student from scratch. However, DistilBERT serves as the gold standard to compare our zero-shot initialization strategies and also sets the benchmark for comparing ideas in subsection 5.

2.1.4.4.1 Results

Our results are summarized in Tables 2.3 and 2.4. For every student initialization method, we also compare fine-tuning using DPKD algorithm vs simply fine-tuning the student via DPSGD. The main takeaways from these experiments are:

- Zero-shot initialization strategies give large performance improvements to student models and come to 2-3% of the performance achieved by pre-trained DistilBERT. Somewhat surprisingly, there is not much difference between our two zero-shot initialization strategies.
- Both pre-trained DistilBERT and student models with zero-shot initialization strategies fall short of matching the performance of teacher models.
- Finally, broadly speaking, DPKD algorithm does not give a significant performance boost to the student models compared to directly fine-tuning them.

2.1.4.5 Better Models are Better Teachers?

Given the results in the previous subsection, one may wonder if better teachers can help in improving the performance of the students. In this regard, we consider two notions of *better teacher*: (1) A larger teacher model in Step 2 of Algorithm 1 (2) A teacher model that is not DP-trained. We note that the final student model in the second case is not DP; however, these experiments allow us to quantify how much performance gains one could have if one

Table 2.5. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models under different teacher models in distillation against full 12-layer BERT teacher models and pre-trained DistillBERT. All our models have the same privacy budget $\epsilon = 4$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	77.8	84.7	87.8	90.5	85.2
Student	Pretrained	-	Finetune	73.0	84.3	82.8	87.7	81.9
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT	DPKD	72.8	82.6	83.0	82.7	80.3
$\frac{1}{2}$-BERT	Zero-shot (PT)	$\text{BERT}_{\text{LARGE}}$	DPKD	72.4	81.1	83.1	81.5	79.5
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT w/out DP	DPKD	74.2	82.9	84.5	83.0	81.1

Table 2.6. Comparison between the performance of 6-layer $\frac{1}{2}$-BERT student models under different teacher models in distillation against full 12-layer BERT teacher models and pre-trained DistillBERT. All our models have the same privacy budget $\epsilon = 1$.

Model	Initialization	Teacher	Training	MNLI	QQP	QNLI	SST-2	Avg
BERT	Pretrained	-	Finetune	74.8	82.1	85.6	86.8	82.3
Student	Pretrained	-	Finetune	68.4	82.0	81.0	86.0	79.3
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT	DPKD	67.5	78.4	80.1	78.5	76.1
$\frac{1}{2}$-BERT	Zero-shot (PT)	$\text{BERT}_{\text{LARGE}}$	DPKD	67.6	78.0	80.1	78.0	75.9
$\frac{1}{2}$-BERT	Zero-shot (PT)	BERT w/out DP	DPKD	70.6	79.0	81.5	79.8	77.7

were to come up with a new framework for implementing KD in DP setting without requiring the teacher to be trained with DP.

Our results are summarized in Tables 2.5 and 2.6. These experiments show that DPKD does not benefit significantly from having better teacher models. Furthermore, our proposed framework of implementing KD in DP framework by training both the teacher and student models via DP is only within 0.8% of the doing KD with fine-tuned teacher without DP.

2.1.5 Evolving Teacher to Student Models via Pruning

As the previous subsection presents, the Knowledge Distillation approach to model compression has two main drawbacks in the private world:

- Drop in accuracy: There is a considerable drop in the accuracy between the teacher and

the student models.

- Good initialization of students is crucial: The best performance is obtained by students who already have a good initialization; in our experiments, pre-trained DistilBERT mostly achieved the best student performance.

Finding a good initialization can be challenging in practice. Often, the student architectures are chosen to suit the hardware and latency requirements of the application for which the model is being deployed, using neural architecture search [46]. Hence, finding a good initialization for every student architecture via pre-training can be expensive and in most cases impossible. Our zero-shot initialization strategies alleviate this problem to a certain degree, yet fall short of closing the gap between the teacher and the student performances. Moreover, DPKD requires that (i) the teacher is trained with DPSGD and (ii) the student is distilled via DPSGD. This two-step approach creates additional overheads in terms of training. Given these limitations, it is natural to ask: Can we evolve the teacher to a student model while fine-tuning with DPSGD? In this subsection, we explore an answer to this via *structured and unstructured pruning* with privacy, which allows us to obtain student models that are as good as the teacher models.

2.1.5.1 Model Compression via Pruning

Pruning algorithms are a broad class of model compression techniques where one drops the parameters from a model during or after the training process. Many works have shown that eliminating unnecessary parameters of neural networks via pruning can lead to sparser and compressed models that have shorter inference times without loss in performance [66, 68, 102]. For example, in *magnitude pruning*, one of the most widely used pruning techniques, we prune a fraction of parameters with the lowest magnitude. However, there are several pruning strategies, and we refer the readers to [113] for more details and references.

Pruning can be implemented in both *structured and unstructured* ways. In structured

pruning, all the pruned weights belong to a single building block of the model. For example, a 6-layer $\frac{1}{2}$-BERT can be obtained by pruning 6 layers from the full BERT model, which consists of 12 transformer blocks. On the other hand, in unstructured pruning, pruned weights may be spread across all the layers of the network. In unstructured pruning, it is possible to obtain a 50% sparse student model while still having all the 12 layers of BERT. Depending on the hardware architectures, inference latency between models with structured and unstructured sparsity could be quite different. However, in this subsection, *we use sparsity as the main measure of model compression*, which is also well accepted in the community [76, 113].

2.1.5.2 Iterative Magnitude Pruning (IMP)

Private pruning techniques we study in this subsection are based on the Iterative Magnitude Pruning (IMP) method, which is a specific pruning technique proposed in a recent work on Lottery Ticket Hypothesis [53]. The idea behind IMP is rather simple: As we train a deep learning model, after every N iterations we prune an α% of the weights with the *lowest magnitude*. We repeat this process until we achieve the desired sparsity. Here, both N and α are hyperparameters that need to be tuned. For example, to achieve 50% sparsity, one can perform $5N$ iterations where after every N iterations additional 10% of the weights with the least magnitudes are dropped. As specified IMP produces unstructured sparsity. However, we consider a simple modification of the IMP algorithm to produce structured sparsity as well.

2.1.5.3 Structured DPIMP

We first attempt to obtain a student model from the teacher model via a structured IMP technique, using the following modification: During fine-tuning the teacher model with DPSGD, we progressively drop an *appropriately chosen transformer block* from the teacher model at the end of every N iterations. We repeat this process until we obtain the student model with the required sparsity. The layer to drop is chosen using the following heuristic: Let $\alpha > 0$ be a

Algorithm 2. Structured DPIMP

Input: Teacher model $\mathcal{T}$, number of layers to prune L, hyperparams α, N and M
Output: Private student model $\mathcal{S}$ with L layers pruned from $\mathcal{T}$

Set $\mathcal{S} := \mathcal{T}$
for $j = 1$ to L **do**
    Fine-tune $\mathcal{S}$ for N iterations with DPSGD
    Set W_{min} consisting of α% of the remaining model weights with the least magnitude
    Set W_i as the weights of layer i
    Drop the layer i^* from $\mathcal{S}$ satisfying $i^* := \text{argmax}_i \{W_i \cap W_{\text{min}}\}$
Fine-tune $\mathcal{S}$ for M more iterations with DPSGD
return $\mathcal{S}$

hyperparameter. At the end of N iterations, fix bottom (by magnitude) α% of *all* model weights, and denote it by W_{min}. For the i^{th} transformer block, let W_i denote the set of model weights belonging to that block. Among all the transformers blocks we find the block i^* that has the highest number of weights from the set W_{min}; Formally, $i^* := \text{argmax}_i \{W_i \cap W_{\text{min}}\}$, and we prune the transformer layer i^*. We present this in Algorithm 2. We note that the algorithm satisfies (ϵ,δ)-DP after the pruning steps due to the post-processing property of differential privacy.

2.1.5.3.1 Empirical Evaluation

We evaluate our structured pruning algorithm with the same setup described in subsection 2.1.4.3 We split the privacy budget equally among all the iterations of the algorithm. Our goal is to produce a student model which has $\frac{1}{2}$ as many layers as the full BERT model. Table 2.7 shows the results for this setting where we compare structured DPIMP to private fine-tuning of the pre-trained DistilBERT and the full BERT model. The main takeaway from this experiment is:

- DP structured pruning algorithm produces a student model that has performance comparable to that of DistilBERT. Further, it avoids the pre-training cost associated with DistilBERT.

Table 2.7. Comparing performance of 6-layer $\frac{1}{2}$-BERT student model produced by structured DPIMP with 12-layer BERT teacher model and pre-trained DistillBERT. The results are shown with two privacy budgets $\epsilon=1$ and $\epsilon=4$.

Model	MNLI		QQP		QNLI		SST-2		Avg	
	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$
BERT	74.8	77.8	82.1	84.7	85.6	87.8	86.8	90.5	82.3	85.2
DistilBERT	68.4	73.0	82.0	84.3	81.0	82.8	86.0	87.7	79.3	81.9
$\frac{1}{2}$-BERT	68.7	72.9	80.7	83.1	80.9	82.5	83.3	85.7	78.4	81.0

2.1.5.4 Unstructured DPIMP

The structured pruning produces student models that are as good as DistilBERT; our next target is to produce student models that are as good as the full teacher BERT model. Towards that, we explore unstructured pruning techniques, in particular differentially private version of the IMP algorithm. As we fine-tune the deep learning models using DPSGD, after every N iterations we prune increments of α% of the weights with lowest magnitude. The remaining weights are then reset to the original pre-trained initialization. (We do this step to establish connections to Lottery Ticket Hypothesis, see below.) We repeat this process until we achieve the desired sparsity. We note that the initialization of the final student model's non-zero parameters at the end of the pruning step are still non-private, even though weights correspond to pre-trained model weights. This is due to the fact that pruned parameters are found during the fine-tuning process, which makes use of the private dataset. Therefore, the whole process must be performed with DPSGD to produce a private student model. Finally, we perform additional fine-tuning of the pruned student model by using DPSGD for M more iterations. We call the private variation of this procedure Unstructured DPIMP. We formally present this process in Algorithm 3.

Algorithm 3. Unstructured DPIMP

Input: Model $\mathcal{T}$ and the sparsity level S%, hyperparams α, N and M
Output: Private Student model $\mathcal{S}$ satisfying the sparsity requirements

1: Set $\mathcal{T}' := \mathcal{T}$
2: **for** $i = 1$ to $\lceil (S/\alpha) \rceil$ **do**
3: Fine-tune $\mathcal{T}'$ for N iterations with DPSGD
4: Prune $(\alpha \times i)$% of weights with the lowest magnitude from $\mathcal{T}'$
5: Reset the non-zero weights of $\mathcal{T}'$ to the original $\mathcal{T}$
6: Fine-tune $\mathcal{T}'$ for M iterations with DPSGD
7: **return** $\mathcal{S} := \mathcal{T}'$

2.1.5.4.1 Empirical Evaluation

We refer to the student model produced by Algorithm 3 as SparseBERT and indicate the sparsity level in brackets. We allocate the privacy budget equally among all iterations of Algorithm 3. Table 2.8 summarizes our experiments on unstructured pruning via DPIMP. We compare the performance of our student model to pre-trained DistilBERT (whose parameter count is the same as our final student model) and the full-sized BERT teacher model. The main takeaways are:

- DPIMP produces a student model that has better performance compared to DistilBERT.
- The average performance of DPIMP is within 2% of the full BERT model. We conclude that unstructured pruning techniques are more effective in closing the gap between the teacher and the student models in the private world. Moreover, pruning methods are computationally cheaper as student models do not require pre-training on public data.
- DPIMP algorithm as described in Algorithm 3 at the end of line 6 finds sparse subnetworks in the pre-trained BERT model that can be fine-tuned to obtain nearly matching teacher performance. This is similar to Lottery Ticket Hypothesis work for BERT networks [30], although in the private world performance of the student network is not matching that of the teacher.

Table 2.8. Comparing performance of SparseBERT student model produced by unstructured DPIMP with 12-layer BERT teacher and pre-trained DistillBERT. The results are shown with two privacy budgets $\epsilon=1$ and $\epsilon=4$.

Model	MNLI		QQP		QNLI		SST-2		Avg	
	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$	$\epsilon=1$	$\epsilon=4$
BERT	74.8	77.8	82.1	84.7	85.6	87.8	86.8	90.5	82.3	85.2
DistilBERT	68.4	73.0	82.0	84.3	81.0	82.8	86.0	87.7	79.3	81.9
SparseBERT (50%)	72.9	76.8	81.1	84.0	82.2	85.7	83.7	87.6	80.0	83.5

2.1.6 Conclusions and Future Directions

In Section 2.1 we initiated the study of differentially private model compression via Knowledge Distillation and Pruning, and gave frameworks for implementing both. Our work shows that one can obtain student models whose performance comes within 2% of the full teacher models while having 50% fewer parameters, which can lead to significant reduction in memory and improve inference time. We believe that our work takes the first step in the intersubsection of private training and model compression techniques, and opens a whole in direction with plenty of interesting and important problems. We highlight some of them here.

- *Better Algorithms For Model Compression*: While we gave natural DP adaptations of KD and pruning algorithms, we believe that there are plenty of new techniques to explore.

- *Better Accounting*: In all our results, we chose very simple strategies for allocating privacy budget across various steps of training and pruning. Theoretical innovations in the form of better accounting can further improve the performance of student models.

- *Lottery Tickets for DP-training?* Both in our experiments on KD and in pruning, we see that models that have "good initialization" lead to dramatic improvements in performance. Moreover our DPIMP algorithm finds sparse subnetworks in pretrained BERT models at 50% sparsity that can be fine-tuned to obtain *nearly* matching teacher performances. This raises an intriguing question akin to Lottery Ticket Hypothesis: Are there good initialization

of models where dynamics of DPSGD is similar to SGD? We believe that this is an exciting research direction both from a theoretical point of view and also its implications to practical private training.

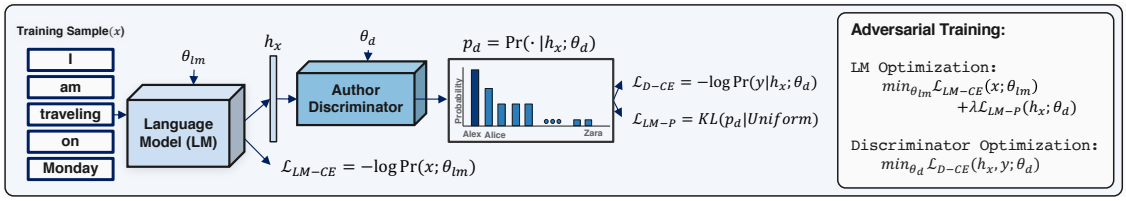


Figure 2.2. Workflow of our adversarial training regularization. The last hidden state (h_x) of the LM is fed to the discriminator to generate a distribution over the authors (p_d). p_d is used to compute $\mathcal{L}_{\text{LM-P}}$, the privacy loss.

2.2 Privacy Regularization: Joint Privacy-Utility Optimization in Language Models

As mentioned before, differentially private SGD [1] requires clipping of the gradients and addition of noise in each update, to provide worst-case guarantees that reflect the likelihood of leaking any attribute of any member of the dataset into the trained model. The worst-case guarantees of differential privacy are not customizable, in other words, they cannot be relaxed to protect only certain attributes. Therefore, DP incurs huge losses to model utility [126]. DP training of models is also much slower, with cumbersome hyper-parameter tuning and development [204]. It has also been shown that DP's utility loss is much worse for under-represented groups [5], which can have financial and societal ramifications [157].

To address these issues, we propose two privacy regularization methods, based on adversarial training and a novel privacy loss term, to jointly optimize for privacy and utility (perplexity) of recurrent language models. The main idea of our regularizers is to prevent the last hidden state representation of the language model for an input sequence x from being linked back to the sensitive attribute we are trying to protect, in our case, the identity of the author. We use the last hidden state as it corresponds to the embedding of the sequence x. We consider the linkability of the input representation to the sensitive attribute (author) as a proxy, since it is commensurate with the linked and linkable information definitions in the General Data Protection Regulation [56]. By framing privacy as an optimization problem, we

can apply the well-developed machinery of large-scale gradient-based optimization, enabling us to train models at scale while jointly tuning for an optimal privacy-utility trade-off.

To validate our approach, we develop an evaluation framework for assessing a model's privacy loss. We employ the exposure metric introduced in [25] and introduce a reconstruction (tab) attack as a realistic scenario to evaluate and compare LSTM language models trained using our regularization with those trained with differential privacy, on Avocado [149] and Reddit [199] datasets. We also empirically demonstrate that unlike DP, our technique does not have disparate impacts on under-represented groups.

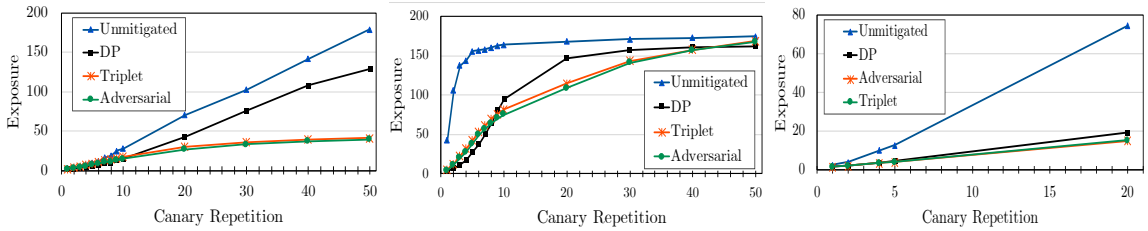
Our work is closely related to [32] and [110]. [32] consider an attacker who eavesdrops on the hidden representations of a pre-trained model during inference and tries to recover information about the input text. Adversarial training is used as a mitigation to reduce the attacker's performance. [110] use adversarial training to protect private author attributes such as age or gender, in learned text representations for part-of-speech tagging and sentiment analysis to gain better performance on out-of-domain corpora. We, on the other hand, use adversarial training and a triplet-based regularization to train private language models that do not memorize sensitive user information, which has not been explored before. We evaluate our models accordingly, by trying to extract training samples. Prior work has studied membership inference attacks against models [187, 192, 212], however, our regularizations do not target these attacks.

2.2.1 Approach

In this subsection we explain our proposed regularizers and training techniques in more detail.

2.2.1.1 Adversarial Training

Figure 2.2 shows our first proposed regularizer which is adversarial in nature. We feed an input text sequence x to the language model and extract the last hidden state representation of the model for x; denoted by h_x. h_x is then fed to a discriminator parameterized by θ_d, which



(a) Avocado - High PPL (∼100) **(b) Avocado - Low PPL (∼60)** **(c) Reddit (PPL ∼100)**

Figure 2.3. Exposure metric results for different training schemes at similar perplexities. Unmitigated denotes conventional training. Adversarial and Triplet are our regularizers. Higher exposure indicates lower privacy.

plays the role of an attacker who attempts to predict what the sensitive label (in our case, the author, y) for x is. The output probability distribution of the discriminator for the input h_x, $p_d = \Pr(\cdot|h_x;\theta_d)$ is then used to compute both the privacy loss $\mathcal{L}_{\text{LM-P}}$ of the language model and the discriminator loss $\mathcal{L}_{\text{D-CE}}$. During training, the discriminator optimizes for better linking of the last hidden state representations to the authors. Thus, the discriminator loss is $\mathcal{L}_{\text{D-CE}}(h_x,y;\theta_d) = -\log\Pr(y|h_x;\theta_d)$. Conversely, the language model optimizes θ_{lm} such that it (1) improves the utility of the language model and (2) flattens the probability distribution over authors. Thus, we devise the following loss function:

$$\mathcal{L}_{\text{LM}}(x;\theta_d,\theta_{lm}) = \mathcal{L}_{\text{LM-CE}} + \lambda\mathcal{L}_{\text{LM-P}} \tag{2.2}$$

$\mathcal{L}_{\text{LM-CE}}$ is the utility loss, for which we use conventional cross entropy loss over the next-word predictions. $\mathcal{L}_{\text{LM-P}}$ is the *privacy loss*:

$$\mathcal{L}_{\text{LM-P}}(h_x;\theta_d) = -\frac{1}{M}\sum_{c=1}^{M}\log\Pr(c|h_x;\theta_d) \tag{2.3}$$

i.e. the KL divergence between the distribution over authors and the uniform distribution where M is the number of classes (authors). The goal of this term is to drive the discriminator to predict randomly uniform outputs [164]. The reason we devised this loss as opposed to using $-\mathcal{L}_{\text{D-CE}}$ is that we do not just want the discriminator to assign zero probability to the

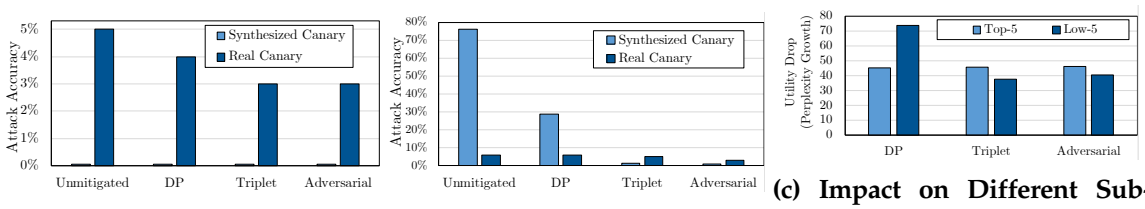
correct author, we want p_d to be uniform so that it has no information about the correct author. Hyperparameter λ allows for trading off privacy and utility.

2.2.1.2 Triplet-based Loss Function

One potential downside of the proposed adversarial regularizer is that the capacity of the discrimitor must scale with the number of authors, and thus the size of the training data. To better accommodate the larger number of authors in large datasets, we investigate another regularizer that does not require a discriminator. We build on the intuition that to obfuscate an attribute, we can increase the distance between representations of samples that have the same label for that attribute, while decreasing the distance between samples with different labels. To this end, we use the language model loss ($\mathcal{L}_{\text{LM}}$) of the previous subsection (Eq 2.2), and we set the privacy loss to be the triplet loss:

$$\mathcal{L}_{\text{LM-P}} = \|h_x - h_p\|^2 - \|h_x - h_n\|^2 \tag{2.4}$$

The triplet loss is commonly used in vision tasks for training embeddings that map images from same category to neighboring points in the embedding space [29]. We, however, invert this loss and use it for an opposite purpose: privacy regularization. During training of the language model, we select a "baseline sample", x, a positive sample p (with different sensitive label) and a negative sample n (with the same sensitive label) and feed them through the language model and extract the last hidden states h_x, h_p and h_n, respectively. We find the l_2 distance between h_x, h_p, and h_n and based on their labels, add them to or subtract them from the loss. To implement this, in practice, we sample a baseline batch and a second "auxiliary" batch during training. We feed both the baseline batch (x) and the auxiliary batch (a) through the language model, and extract the last hidden states. We then calculate the distance between last hidden states of the corresponding samples in the two batches. If the samples have different labels for the sensitive attribute (author), we add their distance to the



(a) Avocado - Low PPL ($\sim$100) **(b) Avocado - Low PPL ($\sim$60)** **(c) Impact on Different Sub-groups**

Figure 2.4. (a, b) Tab attack results for reconstructing canary sequences for two utility levels. Higher attack accuracy indicates lower privacy. (c) Effect of different mitigations on utility of well represented (Top-5) and under-represented (Low-5) users for Avocado dataset.

loss, otherwise, we subtract it. The privacy loss becomes:

$$\mathcal{L}_{\text{LM-P}} = \sum_{i:y_{x_i}=y_{a_i}} \|h_{x_i} - h_{a_i}\|^2 - \sum_{j:y_{x_j}\neq y_{a_j}} \|h_{x_j} - h_{a_j}\|^2 \tag{2.5}$$

2.2.2 Evaluation

In our experiments, we use a subset of the Avocado corporate email dataset [149] with 100 users and 60,000 samples and a subset of Reddit dataset [199] with 10,000 users and 3 million samples. We create a 80/20% training/test set split. We use a two-layer LSTM model as the language model for the next-word prediction task. We compare models trained with our proposed regularizer to differentially private (DP) ones [1]. For the privacy accounting, we use Gaussian differential privacy [19]. We use language model perplexity as a measure of utility.

Privacy measurements w/ exposure metric. To empirically compare the privacy of our methods to that of DP, we adopt the exposure metric introduced by [25]. The higher the exposure of a sequence, the more the model's memorization and the easier it is to extract the sequence from the language model.To measure exposure we insert sequences of five random words (canaries) to the training data. We insert unique canaries with different repetitions for each user, and measure the exposure of these canaries.

Figure 2.3 shows the exposure results per canary repetition. These results are averaged over all the users. In each sub-figure, the perplexities of the models are similar, hence we

can compare the privacy levels at similar utilities. Fig. 2.3a compares trained models using different techniques on the Avocado dataset, where they all have relatively high perplexities compared to a fully trained conventional model. Fig. 2.3b has the same setup, however the models have lower perplexities. Naturally, for having better utility we are trading off privacy, which can be seen by comparing the exposure values in these two figures and observing that the second one has higher exposure values (lower privacy). Finally, Fig. 2.3c shows the exposure results for Reddit.

In all cases we see that the unmitigated model has the highest exposure, as expected. We also observe that for canaries (patterns) that are repeated more than 9 times (for each user), our mitigation offers lower exposure compared to DP, especially in the high perplexity case. This is because clipping and noise addition in DP is attribute and data agnostic, meaning that noise is added to all samples regardless of whether or not they contain sensitive information. Therefore, repeated patterns are less protected. If we want to protect a pattern with n repetitions, we would need to apply noise that is $n\times$ larger, which would degrade the utility gravely and would not yield the same perplexity. For lower repetition canaries, our mitigations have comparable performance to DP. For all these experiments the Gaussian differential privacy criterion μ is extremely large (10^{20}), which practically yields $\epsilon \sim \infty$. We also experimented with lower ϵ values (e.g. $\epsilon \sim 7$), however, it yields a model with perplexity of 650, having an extremely low utility.

Privacy measurements w/ tab attack accuracy. In this experiment, we input the first token, and see if the entire sequence is reconstructed using the language model. We report the rate of correct reconstruction of canaries as the accuracy of the attack. We use the synthetic canaries from the previous experiment, and also select "real canaries" from the training corpus to create a real-world scenario. Fig. 2.4a shows that for a high perplexity model, the accuracy of the tab attack on synthesized canaries is very small, even for the unmitigated model. The unmitigated model reaches the designated perplexity in less than an epoch, and hence it

does not memorize the canaries. For the real canaries however, the memorization is higher, since they follow grammatical rules. In the lower perplexity case of Fig. 2.4b, we see that the synthesized canaries are mostly memorized by the unmitigated model. Our mitigations outperform DP, especially for the synthesized canaries. DP is not context-sensitive and applies the same amount of noise to all samples, thereby leaving correlated and higher repeated samples less-protected. Our mitigations, however, learn what sequences are link-able to their authors, and obfuscate them such that they no longer leak the "identifying" secret.

Effect on under-represented users. Differential privacy has disparate impact on the accuracy of different subgroups of the dataset [5]. Here, we want to measure the effect of our mitigations on the utility of the model among users with various data samples. For each user, we measure the average perplexity of the model for their samples in the test set, and then subtract this from the same value for an unmitigated model. This would yield the average drop in utility, per user. We compare the utility drop of well-represented users to under-represented ones by taking the top 5 users with most samples and the bottom 5 users with the fewest samples from Avocado dataset. We then measure the average utility drop over each group of 5 users on the test set. Figure 2.4c shows these results. We see that differential privacy has disparate impact, 29 points, on the two sub-groups of users (authors), whereas this gap is only 7 points for models trained with our mitigations.

2.2.3 Conclusion

This work introduces two privacy mitigation methods to jointly optimize for privacy and utility. Extensive experiments show that our approach provides comparable and in certain cases a higher level of privacy compared to differentially private model training. We further empirically demonstrate, that our methods do not exhibit disparate impacts on under-represented groups and have significantly less overhead on training performance.

Ethical Considerations

While handling sensitive email data (Avocado) we made sure to abide by the terms of its end-user license agreement (EULA) which has provisions to protect the privacy of members of the corpus. Furthermore, we took measures such as scrubbing named entities before using the data for model training. The over-arching goal of our work is to contribute to language model development that protects the privacy rights of users who contribute their data. While we rigorously evaluated our models by applying state-of-the-art attacks, deploying these models in real-world setups requires further scrutiny and regulations.

Chapter 3

Attribute Control For Fairness and Privacy

Ideally high-stakes decisions made by either humans or ML algorithms should not be influenced by irrelevant, protected attributes like ethnicity or age. In many instances, the input data used for making high-stakes decisions is text that is authored by a human candidate – for example, hiring decisions are often based on bios and personal statements. Recent work [35] shows that automatic hiring-decision models trained on bios are less likely to select female candidates for certain roles (e.g. architect, software engineer, and surgeon) even when the gender of the author is not explicitly provided to the system. Bias is, of course, not limited to algorithmic decisions, humans make biased decisions based on text, even when the protected attributes of the author are not explicitly revealed [152]. Together, these results indicate that both algorithms and humans can (1) decipher protected attributes of authors based on stylistic features of text, and (2) whether consciously or not, be biased by this information.

A large body of prior work has attempted to address *algorithmic bias* by modifying different stages of the natural language processing (NLP) pipeline. For example, [165] attempt to de-bias word embeddings used by NLP systems, while [45] address the bias in learned model representations and encodings. All such methods need to be applied by the deployer of such models (could be different service providers or hiring companies), taking the control away from the users/authors of text. They also only address the bias in the NLP models, and not implict or explicit bias of the humans who make decisions. As such, we first propose a

revision method that makes edits to text to try and obfuscate sensitive stylistic information. Unlike prior work, the re-writing mechanism we propose focuses on obfuscation/mixing of styles, rather than hiding style by mimicking other styles. For example, for protecting the sensitive attribute "age" of the author, the approach that prior work would take is to *transfer* the style of text from teenage-style to adult-style, by re-writing it such that it mimics text written by adults. We, however, propose to remove any age revealing stylistic feature entirely, as opposed to hiding age by mimicing different age-groups. Using our method, the teenage-written sentence "grr ... now i get cold quicker" would become "hmmm ... now i get cold . ", hiding age-revealing features entirely. We do this by transferring style to an unseen style, which is an intersection of the styles that the model has seen before.

Although text revision using style transfer as proposed above can help improve fairness of the decisions made by the text through obfuscating attributes, it necessitates training new models and architectures for each attribute we want to obfuscate (apply control for), making it non-trivial to customize and modify protected attributes. This raises the question *can we use existing trained models as building blocks for controlling attributes?* This would alleviate the need for training models from scratch, as there are many existing open-source pre-trained/fine-tuned classifiers and models that can be used and glued together to apply control for desired attributes. As such, in the second part of this chapter we propose a framework named "Mix and Match", where, unlike prior work, the main goal is to enforce attributes using existing arbitrary pre-trained models and heuristics, whether they are discrete/continuous or differentiable/non-differentiable, using an energy-based model. We demonstrate the application of Mix and Match on many tasks, such as bias removal and style and formality transfer, without the need to train any new models.

3.1 Style Pooling: Automatic Text Style Obfuscation for Improved Classification Fairness

We introduce a novel framework that enables 'style pooling': the automatic transduction of user-generated text to a central, obfuscated style. Notions of 'centrality' can themselves introduce bias – for example, a system might learn to obfuscate by mapping all text to the dominant style seen in its training corpus. This might 'white-wash' text, ignoring stylistic features of underrepresented groups in the learned notion of central style. Our framework operationalizes the notion of centrality in a more flexible way: our probabilistic approach allows us to choose between two distinct notions of centrality. First, we define a variant of our model which is incentivized to learn a minimal notion of central style that effectively *intersects* the various styles seen in training. This is achieved through the design of this variant's probabilistic prior. We further equip this variant with a novel "de-boosting" mechanism, which amplifies the use of words that are less likely to leak sensitive attributes, and de-incentivizes the use of words whose presence might hint at a particular sensitive attribute. Second, we propose an alternative prior that instead incentivizes a maximal notion of style that seeks to obfuscate by adding stylistic features of all protected attributes to text – in effect, computing a *union* of styles. Table 3.1 shows our intersubsection and union obfuscation applied to sentences from the Blogs dataset, and highlights the differences between them.

While we propose both these obfuscations in our framework and leave it to the users to choose, it is worth noting that the cognitive process literature shows that when humans are confronted with conflicting biasing information, they tend to form an opinion about the conflicting text, based on their own implicit biases [168]. Therefore, removing sensitive stylistic features may be more effective than combining them. This is also commensurate with our findings, where we observed that intersubsection more successfully improves the fairness metric (subsection 3.1.3.2).

Table 3.1. Example Blog sentences transformed with A4NT [185] and our proposed Intersubsection and Union obfuscations. Our Intersubsection obfuscation aims at changing the style such that it does not reflect either teen or adult style. However, the union, tries to reflect both by making changes like adding "..." to the beginning of the sentence (adult style) while keeping the "grr" (teen style). Or by adding exclamation marks at the end of the sentence.

Age	Input Sentence (Original Data)	A4NT (Baseline)	Intersection	Union
Teen	**grr** ... now i get cold quicker .	**grr** now i get cold **lol** .	**hmmm** ... now i get cold .	**...** **grr** ... now i get cold quicker .
Teen	it was so **fricken** hilarious .	it was so **boring** hilarious .	it was so **utterly** hilarious .	it was so **totally** hilarious
Adult	well i 've just been **too busy** .	well i 've just been **kinda fun** .	well i 've just been **too busy** .	well i 've just been **too busy .**
Adult	these were **common phrases** .	these were **common teacher** .	these were **common** .	these were **common ! !**

We extensively evaluate our proposed framework on a wide range of tasks. First, we compare and contrast our "intersubsection" and "union" obfuscations on a modified version of the Yelp dataset [182] where we have created three stylistic domains by deliberately misspelling three disjoint sets of words. We show that our intersubsection obfuscation successfully removes these misspellings and replaces them by the dominant spelling of the word 99.20% of the time, while our union obfuscation spreads the misspellings into the other two domains 46.40% of the time. Then, we evaluate our framework on the Blogs data [180], where the sensitive attribute is age, and we measure the impact our obfuscations have on the fairness of a job classifier, using the the TPR-gap measure from [35]. We also evaluate the removal of sensitive attributes, fluency of the generated text, and the uncertainty of a sensitive attribute classifier for our framework, in both two and three domain setups.

3.1.1 Proposed Method

In this subsection, we first introduce our model structure, then describe our style-pooling priors and the unsupervised learning and inference techniques we leverage for this model class. Finally, we introduce our style de-boosting mechanism.

3.1.1.1 Model Structure

Consider a training corpus consisting of utterances produced by authors with various protected attributes. In Figure 3.1, we depict a grouping of authors by age into three domains. We let x_i represent an individual observed text utterance in the corpus, and assume M

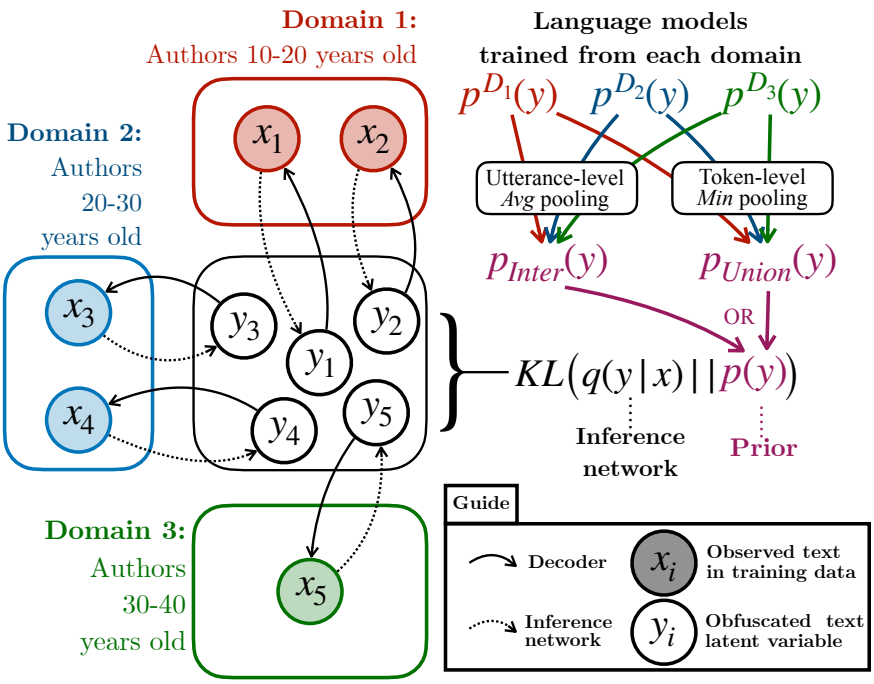


Figure 3.1. Proposed unsupervised framework for *style pooling*: inducing a centralized obfuscated style. x_i represent observed text which are clustered by their sensitive attribute (age). y_i are corresponding latent variables representing the induced obfuscated text. Training leverages an amortized inference setup similar to a VAE-style training, but, critically the prior is produced by pooling language models from each domain using two different strategies targeting (1) intersected style and (2) the union of all styles in the corpus.

domains (sensitive attribute classes) in the dataset. y_i is a latent variable that represents the obfuscated version of x_i. Hence, y_i is a text valued latent, while x_i is a text valued observation. We let $d(i)$ denote the domain of the i^{th} sample in the dataset. With this definition, our generative process assumes each sentence x_i, with corresponding domain $d(i)$, is generated as follows: First, a latent sentence y_i is sampled from a central prior, $p_{prior}(y_i)$, which is domain agnostic. Then, x_i is sampled conditioned on y_i from a transduction model, $p(x_i|y_i;\theta^{d(i)}_{y\to x})$. We let $\theta^{D_j}_{y\to x}$ represent the parameters of the transduction model for the jth domain. We extensively discuss p_{prior} in the next subsection. For now, we assume the prior distributions are pretrained on the observed data and therefore omit their parameters for simplicity of notation. Together, this gives the following joint likelihood:

$$
\begin{aligned}
&p(X^{D_1},...,X^{D_M},Y;\theta_{y\to x}^{D_1},...,\theta_{y\to x}^{D_M}) \\
&=\prod_{i=1}^{N} p\left(x_i|y_i;\theta_{y\to x}^{d(i)}\right) p_{prior}\left(y_i\right)
\end{aligned}
\tag{3.1}
$$

The log marginal likelihood of the observed data, which we approximate during training, can be written as:

$$
\begin{aligned}
&\log\, p(X^{D_1},...,X^{D_M};\theta_{y\to x}^{D_1},...,\theta_{y\to x}^{D_M}) \\
&=\log\textstyle\sum_Y p(X^{D_1},...,X^{D_M};\theta_{y\to x}^{D_1},...,\theta_{y\to x}^{D_M})
\end{aligned}
\tag{3.2}
$$

Neural Architectures. We select a parameterization for our transduction distributions that makes no independence assumptions. We use an encoder-decoder architecture based on the standard attentional Seq2Seq model which has been shown to be successful across various tasks [6, 171]. Our prior distributions for each domain are built using recurrent language models which also make no independence assumptions.

3.1.1.2 Prior Distributions

The critical component of our framework that incentivizes obfuscation are our specialized priors, as depicted in Figure 3.1. We introduce two prior variants, $p_{\text{Inter}}(y)$ and $p_{\text{Union}}(y)$, which incentivize induction of intersected styles and the union of all styles, respectively. Each prior is assembled out of M (here $M=3$) separate language models – p^{D_1}, p^{D_2}, ..., p^{D_M} – each trained on the corresponding domain of observed utterances in the training data. The intersubsection prior, $p_{\text{Inter}}(y)$, is computed by taking the sum of the likelihoods of an entire utterance across the language models from all M domains (and then re-normalizing to ensure that resulting prior is a valid distribution). This utterance-level average pooling approach incentivizes a "majority-voting" effect, in which the model is pressured to remove any words and stylistic features that are characteristic of one domain, but not the others, and converge to features that are shared by the majority of the domains. Therefore the prior for intersubsection becomes:

$$p_{Inter}(y_i) = \frac{1}{M}\sum_j^M p^{D_j}(y_i) \tag{3.3}$$

In contrast, the union prior, $p_{\text{Union}}(y)$, computes the likelihood of an utterance according to the minimum likelihood across each domain's language model *at each token position*, t.[1] Through experimentation (Sec. 3.1.3.1) we empirically observed that this prior rewards the model for inserting as many stylistic features as possible that are unique to each domain.

$$p_{Union}(y_i) \propto \prod_t^T \min(p^{D_1}(y_{i,t}|y_{i,<t}),...,p^{D_M}(y_{i,t}|y_{i,<t})) \tag{3.4}$$

3.1.1.3 Learning and Inference

Training is accomplished using an approach from [70]: We employ seq2seq inference networks and use an amortized inference scheme similar to that used in a conventional VAE, but for sequential discrete latents.

Ideally, learning should directly optimize the log data likelihood, which is the marginal shown in Eq. 3.2. However, due to our model's neural parameterization, the marginal is intractable. To overcome the intractability of computing the true data likelihood, we adopt amortized variational inference [96] to derive a surrogate objective for learning the evidence lower bound (ELBO) on log marginal likelihood:

[1]The token-wise min of the language models is not, itself, a normalized distribution. However, we can treat it as implicitly normalized in our training objective (discussed in the next subsection) because the absence of normalization only contributes an additive constant to our objective.

$$
\begin{aligned}
&\log p(X^{D_1},\ldots,X^{D_M};\theta^{D_1}_{y\to x},\ldots,\theta^{D_M}_{y\to x}) \\
&\geq \mathcal{L}_{\text{ELBO}} \quad (X^{D_1},\ldots,X^{D_M};\theta^{D_1}_{y\to x},\ldots,\theta^{D_M}_{y\to x},\phi_{x\to y}) \\
&= \sum_i^N \quad \Big[\underbrace{\mathbb{E}_{q(y_i|x_i;\phi_{x\to y})}[\log p(x_i|y_i;\theta^{d(i)}_{y\to x})]}_{\text{Reconstruction likelihood}} \\
&\qquad\qquad \underbrace{- D_{\text{KL}}\big(q(y_i|x_i;\phi_{y\to x}))||p_{prior}(y_i)\big)}_{\text{KL regularizer}}\Big]
\end{aligned} \tag{3.5}
$$

This new objective introduces $q(y|x;\phi_{x\to y})$, which represents the inference network distribution that approximates the model's true posterior, $p(y|x;\theta_{x\to y})$. Learning operates by optimizing the lower bound over both variational and model parameters. Once training is over, the posterior distribution can be used for style obfuscation.

The reconstruction and KL terms in Eq. 3.5 involve intractable expectations, which means we need to approximate their gradients. To address this, we use the Gumbel-softmax [85] straight-through estimator to backpropagate gradients from both the KL and reconstruction loss terms.

Length Control. During the training of the model, we observed that it tends to repeat the same word when it is trying to generate obfuscated text, y_i. To mitigate this, we append two floating point length tokens to the input of the inference networks decoder at each step t, one of these tokens tells the model which step it is on, and the other tells it how many steps are left [80,95]. We also experimented with positional embeddings instead of floating point tokens, but we observed that they yield worse convergence. Another measure we take to encourage shorter sentences was to hard stop the decoding during training once the re-written sentence had the same length as the original sentence. To further stabilize training we share parameters between the inference network and the transduction models, appending an embedding to the input to indicate the output domain.

3.1.1.4 Style De-boosting

To better encourage the removal of identifying stylistic features, we introduce a de-boosting mechanism, which incentivizes the use of words that are less likely to leak sensitive attributes, and de-incentivizes the use of words whose presence might hint at a particular sensitive attribute. We build on the intuition that for a given word w in the vocabulary, if the probability that it belongs to domain m is similar to the probability that it belongs to domain k, for any given m,k within the possible domains, M, then we can assume that this word does not reveal style. However, if there is a huge gap in the two probabilities, that word might hint at a certain domain if it is present in the re-written text. Therefore, we devise a normalized "style score", s, for each word w in the vocabulary [2]:

$$s_w = \frac{\max(f_w^{D_1}, f_w^{D_2}, ..., f_w^{D_M}) - \min(f_w^{D_1}, f_w^{D_2}, ..., f_w^{D_M})}{\max(f_w^{D_1}, f_w^{D_2}, ..., f_w^{D_M})} \tag{3.6}$$

Where $f_w^{D_1}$ is frequency of word w in the training corpus for domain D_1, divided by the overall number of tokens (words) in the domain corpus. Using these scores, we modify the output logits of the decoder so that the output probability distribution over the vocabulary for sample i at step t is given by:

$$p(y_{i,t}|y_{i,<t}, x_i) \propto \text{softmax}(L_{i,t} - \gamma * S) \tag{3.7}$$

Here, $L_{i,t}$ represents the logits at step t, while S is the score vector for all the words in the vocabulary. γ is a multiplier that helps tune the amount of de-boosting. Due to the nature of this de-boosting mechanism, it makes sense only to use it with the intersubsection obfuscation and not the union.

[2] While this style score may also highlight *content* that is characteristic of a domain in addition to stylistic word choices, we find in experiments that our use of de-boosting does not substantially harm the utility of downstream classifiers – indicating that content is largely preserved, even with de-boosting.

3.1.2 Experimental Setup

Here, we provide a brief description of our experimental setup.

3.1.2.1 Model Configurations

We used a single layer attentional LSTM-based Seq2Seq encoder-decoder for all the experiments, with hidden layer size of 512 for both encoder and decoder, and word embedding size of 128. For the attribute classifiers and language models, we also use LSTM models with the same architecture, with a final projection layer of the size of sensitive classes/vocabulary.

3.1.2.2 Datasets

Synthetic Yelp dataset [182]. We shuffle all the sentences in the Yelp reviews dataset and divide them into three groups (domains). We then randomly choose 15 words from the top 20 highest frequency words in the dataset, and allocate the set of top 5 words (W_1) to D_1 (domain 1), next 5 to D_2 and the least frequent 5 words to D_3. We misspell all occurrences of W_1 in D_1, by changing "word" to "11word11". We then add "11word11" to the vocabulary, and do this for all the 5 words in all 3 domains (15 words total). After this transformation, we have 3 domains with disjoint stylistic markers, which can help us more concretely analyze our obfuscation mechanism.
Blogs dataset [180]. The blogs dataset is a collection of micro blogs containing over 3.3 million sentences along with annotation of author's age and occupation. We use this data in both two and three domain style pooling, where we treat age as the sensitive attribute and balance the data so each domain has the same number of sentences. In the two domain setup, we divide the data in two groups of teenagers and adults. In the three style setup, we have three groups of teenagers, young adults (20s) and adults (people in their 30s and 40s). We use this dataset for multiple evaluations including fairness. We compare our obfuscation to that of [185] in all evaluations with this data.
Twitter dataset [162]. We use data from the PAN16 dataset, which contains manually anno-

tated (from LinkedIn) age and gender of 436 Twitter users, along with up to 1000 tweets from each user. We use this data for the purpose of sensitive attribute (age) removal comparison with [45] in subsection 3.1.3.5, and have therefore used the exact same preprocessing and handling of the data as done by them.

DIAL dataset [14]. This is a Twitter dataset which has binary dialect annotations of African American English (AAE) and Standard American English (SAE)[3], setting "author's race" as the sensitive attribute. We use this dataset for comparison with the work [207].

3.1.2.3 Baselines

One language model prior (One-LM). This model is an instance of our framework which uses the output distribution of a single language model as the prior. For the Yelp Synthetic data this single LM is trained on the original data which does not have our modifications and would provide the ideal "intersubsection", since the original data itself does not have misspellings from any of our synthetic domains and can be considered as central. In the case of the Blogs data where we don't have any ideal central data which is void of style, we train an age classifier and then choose the sentences from the training set that the classifier missclassifies. We create a new training set with these samples and train a single LM on them, and use it for the prior. The intuition is that if the classifier could not guess the domain, these samples are probably close to the notion of centrality we are looking for.

A4NT [185]. "A4NT: Author Attribute Anonymity by Adversarial Training of Neural Machine Translation" is the most closely related past work that also attempts to obfuscate text style through automatic re-writing. However, their adversarial approach uses a discriminator network to hide protected attributes simply by mapping the style of one protected category to that of another.

PATR [207]. Privacy Aware Text Rewriting (PATR) is another work close to ours, which

[3]Using standard for non-AAE might not be the most suitable naming, but we use it hereon given the lack of a better substitute.

removes sensitive attributes through text re-writing using translation and adversarial learning. Unlike style pooling, PATR, targets privacy and is therefore not concerned with the union vs. intersubsection of sensitive attributes.

Original. We include an "original" baseline in our measurements, which shows the value of a given metric if the original un-obfuscated data was used.

3.1.2.4 Evaluation Metrics

Below we discuss our evaluation metrics, all of which are measured on the test data.

Fairness TPR-gap. We first define a classifier whose main task is to determine if the occupation of an author is student or not, given text from their blog. We set the age of the author as a sensitive attribute, and want to measure the bias in the classifier, given age. We follow [35] and use the "True Positive Rate gap in age" metric. This measure quantifies the bias in a classifier by finding the gap between the true positive rate for each sensitive attribute group (teen vs. adult). For a binary sensitive attribute a (age) and a true class (for the classifier's main task) y, we define:

$$TPR_{a,y} = P(\widehat{Y} = y | A = a, Y = y) \tag{3.8}$$

$$GAP^{TPR}_{a,y} = TPR_{a,y} - TPR_{a',y} \tag{3.9}$$

where A is the random variable denoting binary sensitive attribute with values a and a'. Y, $\widehat{Y}$ are random variables denoting the correct class and the predicted class, respectively. The lower the gap is, the more fair the classifier. We report $GAP_{Teen,Student}$, which reflects how biased the classifier is towards classifying teens as students. **Linguistic Back-Translation (BT) accuracy.** We translate the obfuscated samples back to their original domain using the model, and then for each token see if it has been correctly back-translated to its origin or not. We

use this metric to see whether the obfuscated version contains sufficient information about content to reconstruct the original.

GPT-2 PPL. We feed our obfuscated test sentences to a huggingface [158] pre-trained GPT-2 medium model, and report its perplexity (PPL), as an automatic measure of fluency. Lower PPL hints at more fluent text.

BLEU Score. In the Yelp Synthetic data experiments, since we have the original (not misspelled) text, we can calculate and report the BLEU score.

GLEU Score. We use GLEU [205] score as another metric for evaluating the fluency of the generated sentences.

Lexical Diversity (Lex. Div.) To better quantify the differences between different obfuscations, we calculate the lexical diversity as a ratio where the size of the vocabulary of the model's output text is the numerator, and the denominator is the overall size of the model's output text (number of all the tokens in the output).

Sensitive-Attribute Classification

Sensitive-attribute Classifier (Clsf.) accuracy. To evaluate the removal of sensitive attributes, we train a sensitive-attribute classifier, and use its accuracy as a metric. The closer the accuracy is to chance level (random guess), the more successful is the removal. However, there is a caveat to this metric: it is not always clear how the classifier is making its decision, if it is based on content, or style. Therefore, this metric alone is **not conclusive**.

Entropy. To better measure how uncertain the classifier becomes, we also compute its average Entropy across all test samples. Entropy is always between [0.0,1.0] for two domain classification and [0.0,1.59] for three domain classification. The higher it is, the more uncertain the classifier is (more desirable for our purpose).

Confident Response (CR) percentage. We calculate the percentage of the responses from the classifier for which it was more than 75% sure.

3.1.3 Experimental Results

3.1.3.1 Synthetic Yelp Data

Table 3.2 shows the experimental results for the Synthetic Yelp dataset experiment, where we trained our proposed framework using the three synthetic domains with misspellings, as explained in subsection 3.1.2.2. The *Corrected*, *Remaining* and *Removed* percentages refer to the average ratio of the misspellings corrected, remaining and removed for each domain. These should all sum up to 100%. The *Spread* is the average ratio of the number of words from one domain that have been changed to misspellings from another domain. For instance, if there are 100 occurrences of "word" outside D_1 before obfuscation, if 40 of them are converted to "11word11" after obfuscation, then the spread would be 40%. The *One-LM* can be considered as an "oracle baseline" in this case, since it was trained on original (no misspellings) data.

The main goal of this controlled experiment is to compare and contrast our intersubsection and union obfuscations. From the Table we can see that both our obfuscations lead to high fidelity (back-translation accuracy) and semantic consistency (BLEU score). They also both render the domain classifier very close to chance level (33.33%). The main differences between these two methods becomes more clear when we look at the corrected, remaining, and spread numbers. The intersubsection obfuscation with its average pooling, demonstrates a majority voting behavior which incentivizes correcting the misspellings since 2 out of the 3 language models advocate for the correct spelling. Therefore 99.20% of the misspellings are corrected using intersubsection, very close to the oracle baseline. The Union prior, on the other hand, corrects only 45.17% of the misspellings, and lets 54.37% of them to remain as they are. It also converts 46.40% of the correctly spelled words in other domains to misspellings. This shows that the union is in fact mixing the styles, creating sentences that might have more than one misspelling in them.

Table 3.2. Results for the Synthetic Yelp dataset with 3 domains. *Corrected* shows what % of modified words in a domain were corrected back to their original format. *Spread* shows the reverse.

	Intersection	Union	One-LM
BT Accuracy (%)	92.47	94.52	95.58
Corrected (%)	99.20	45.17	99.87
Remaining (%)	0.61	54.37	0.00
Removed (%)	0.18	0.46	0.12
Spread (%)	0.18	46.40	0.00
Cls Accuracy (%)	33.48	34.99	33.35
BLEU	81.74	70.86	93.01

3.1.3.2 Blogs Data

Tables 3.3, 3.5 and 3.6 summarize the experimental results for the Blogs dataset. Below, we will explain each experiment in more detail.

Fairness Results Table 3.3 shows the results for the fairness metric measurements on text generated using different obfuscations, for "Occupation" classifiers. We have selected a subset of the Blogs data for this experiment, where author occupation is either student or arts, and the age is either teen or adult (two domain obfuscation). we have taken an approach similar to that of [165], where we create 4 different levels of imbalance. In all cases, the dataset is balanced with respect to both occupation and age. We change only the proportion of each age within each occupation class (e.g., in the 0.8 ratio, the student occupation class is composed of 80% teens and 20% adults, while the arts class is composed of 20% teens and 80% adults). For each imbalance ratio we train the classifier on the original imbalanced data, and then test it with original and autmotically generated data from different baselines.

Based on Table 3.3, we can see that our Intersubsection obfuscation can improve fairness (TPR-gap) with little harm to the classifier accuracy (Occupation), in comparison to the original data and A4NT. We can trade-off classifier accuracy and fairness, by increasing the de-boosting (DB) multiplier. In the Table, *Intersubsection* shows Intersubsection obfuscation with different DB levels. In the case of $DB = 40$, we lose slightly more utility, but observe much better fairness.

A4NT's performance in terms of the fairness metric (TPR-Gap) is comparable to our

Table 3.3. Fairness results for the Blogs data. The main task is classifying if the author occupation is student or not. Higher occupation accuracy and lower TPR-gap are better. DB denotes our style de-boosting technique, and the number next to it shows its multiplier. Larger multiplier means stronger style obfuscation.

Ratio	Occupation Accuracy (Utility)						TPR-gap (Fairness)					
	Original	A4NT	Intersection			Union	Original	A4NT	Intersection			Union
			No DB	DB= 25	DB= 40				No DB	DB= 25	DB= 40	
0.95	74.56	59.35	74.77	71.12	69.73	73.22	0.54	0.29	0.23	0.23	0.21	0.51
0.80	65.55	54.74	65.31	65.12	59.60	65.43	0.35	0.21	0.35	0.18	0.18	0.36
0.65	59.01	52.73	58.41	56.68	54.45	57.19	0.12	0.05	0.11	0.11	0.11	0.15
0.50	58.09	53.47	56.21	53.6	53.18	55.49	0.04	0.08	0.05	0.05	0.03	0.05

Intersubsection obfuscation (even without de-boosting), however, in maintaining occupation accuracy (utility), A4NT performs much more poorly. We presume this is because A4NT removes sensitive attributes solely based on hints from a discriminator, and the low occupation accuracy suggests the discriminator captures the content more than it captures style, therefore it changes the meaning and structure of the sentences as well. Our human judgments for semantic consistency and fluency in subsection 3.1.3.4 support this hypothesis. Our *Union* obfuscation, however, does not improve the fairness. We hypothesize this could be caused by keeping/adding biasing words, which can perpetuate the existing impartialities in the classifier, similar to how human cognition works [168].

Linguistic and Sensitive-attribute Classification Results The top subsection of Tables 3.5 and 3.6 show the linguistic and sensitive-attribute classification metrics for the two and three domain obfuscations, respectively. Since A4NT cannot be applied to non-binary style obfuscations as is, there are no results for it in three domains. We can see that for both two and three domains the de-boosting (denoted as DB) offers a trade-off between the linguistic quality of the generated text and the obfuscation of sensitive attributes. Compared to the One-LM baseline, for corresponding levels of de-boosting, our *Intersubsection* obfuscation is almost always superior, in both text quality and obfuscation. The *Intersubsection* obfuscation with de-boosting multiplier of 25 outperforms A4NT, with lower classifier accuracy, higher entropy and much lower Confident Response (CR) rate from the classifier. In general, the *Intersubsection*

Table 3.4. Comparison with PATR [207], on the Twitter DIAL dataset, where the author's race is the sensitive attribute.

Metric	PATR		Intersection		Union
	$\alpha=1$	$\alpha=5$	No DB	DB = 20	
GLEU	24.77	9.67	26.32	17.21	26.25
Clsf. Acc (%)	74.85	65.75	74.05	62.12	73.27

Table 3.5. Linguistic and sensitive-attribute classifier results for Blogs data, considering *two* sensitive age domains of teens and adults. For BT accuracy and entropy higher is better, for PPL and Confident Response (CR) lower is better.

	Metric	Original	A4NT	One-LM			Intersection			Union
				No DB	DB = 25	DB = 40	No DB	DB = 25	DB = 40	
Ling.	BT Accuracy (%)	100.00	66.49	94.47	92.88	90.60	95.41	87.39	88.63	96.88
	GPT-2 PPL	41.71	44.85	39.51	53.65	66.21	41.6	42.80	58.15	42.07
	Lex. Div. (%)	3.22	2.28	2.52	1.82	1.09	2.50	1.47	0.97	2.71
Clsf.	Clsf. Accuracy (%)	64.73	61.31	62.07	61.69	59.52	64.23	60.90	59.81	64.02
	Entropy	0.87	0.86	0.87	0.91	0.95	0.87	0.93	0.95	0.87
	CR (%)	14.21	15.72	13.44	6.49	2.80	13.95	4.78	2.47	14.22

obfuscation, even without de-boosting does well on *Entropy* and *CR*, which shows that our method is doing well at creating doubt in terms of what the age of the author is. One caveat however, across both two and three domain obfuscations is the classifier accuracy, which does not decrease much. We hypothesize that one reason for this could be the dependency between style and content, and that the sensitive-attribute classifier could be basing its decisions on content, therefore changing the style would not hide the sensitive attribute.

Our *Union* obfuscation is behaving differently from the *Intersubsection*, and is inferior in terms of obfuscating the text, with higher classifier accuracy and lower entropy. However, it has higher lexical diversity, which could hint at it trying to keep sentences diverse and "adding styles", whereas the *Intersubsection* is only keeping the common words and is therefore decreasing the lexical diversity.

Table 3.6. Linguistic and sensitive-attribute classifier results for Blogs data, considering *three* sensitive age domains of teens and adults. For BT accuracy and entropy higher is better, for PPL and Confident Response (CR) lower is better.

	Metric	Original	A4NT	One-LM			Intersection			Union
				No DB	DB = 25	DB = 40	No DB	DB = 25	DB = 40	
Ling.	BT Accuracy (%)	100.00	–	93.84	93.64	87.83	89.09	89.25	82.47	93.30
	GPT-2 PPL	41.70	–	43.49	48.99	84.61	48.15	49.70	69.08	42.66
	Lex. Div. (%)	3.41	–	2.46	1.81	0.94	1.97	1.02	0.77	2.86
Clsf.	Clsf. Accuracy (%)	49.78	–	49.16	47.64	45.41	48.12	47.13	45.81	48.81
	Entropy	1.38	–	1.38	1.43	1.47	1.44	1.44	1.49	1.38
	CR (%)	43.00	–	43.33	38.89	30.75	38.76	35.37	28.02	45.88

3.1.3.3 Comparison with PATR

Table 3.4 provides a comparison between our style pooling method, and PATR [207]. α is knob used by PATR to tune the intensity of attribute removal, and the classifier accuracy on non-modified text is 86.3%. We can see that without de-boosting, our intersubsection method drops the classifier accuracy to 74.05% with a GLEU score of 26.32. PATR drops the classifier accuracy to 74.85%, but with a worse level of GLEU. With de-boosting, however, we can achieve a classifier accuracy of 62.12% with GLEU of 17.2, whereas PATR reports accuracy of 65.75% for a much lower GLEU of 9.67 when α is increased. This shows that our de-boosting mechanism can provide an advantage by giving a lower probability to attribute revealing components, while maintaining the sentence structure. Our union method also achieves 73.27% accuracy with 26.25 GLEU, making it most suitable for cases where the semantic consistency of the sentences is most important.

3.1.3.4 Evaluation with Human Judgments

We design two crowd-sourcing tasks on Amazon Mechanical Turk. (1) Fluency: We provide workers with a pair of obfuscated sentences, and ask them which sentence is more fluent. (2) Semantic Consistency: We provide the original (un-obfuscated) sentences, and ask workers which of the obfuscated sentences is closer in meaning to the original sentence. The model checkpoints used for human evaluations here are those whose fairness and linguistic

metrics are reported in Tables 3.3, 3.5. We use our intersubsection obfuscation, with no de-boosting. We randomly select 188 sentences from the test set, and used the model outputs for human judgment. For consistency, each pair of sentences is rated by three workers and we take the majority vote. In terms of fluency, the workers preferred our obfuscations over those of A4NT for 60.38% of the sentences. In terms of semantic consistency, for 72.13% sentences they found our obfuscations to be closer in meaning to the original ones.

3.1.3.5 Comparison with [45]

[45] aims at creating representations for text that could be used for a specific classification task, while hiding sensitive attributes. Although our approach deals with the text as opposed to representations and and can be applied for a wider range of downstream tasks, we offer a brief comparison to this method. [45] use the Twitter dataset [162], set the sensitive attribute to be age, and try to produce representations that would perform well on the main task of "conversation detection" (mention detection) on Tweets. On the original data, they report an accuracy of 77.5% and 64.8% for a classifier that tries to classify conversations and age, respectively, which drop to 72.5% and 57.3%, after applying their adversarial learning scheme.

We cloned their repository and used their code to process the dataset.We then created and trained the conversation and age classifiers, and reached an accuracy of 75.8% and 64.63% for them, respectively. These dropped to 73.28% and 54.2%, after applying applying our intersection method. This shows that for this particular task, our re-written text can out-perform prior work.

3.1.4 Related Work

A large body of prior work has attempted to address *algorithmic bias* by modifying different stages of the natural language processing (NLP) pipeline. [15], [8], [49], [136] and [184] propose and analyze benchmarks for evaluating fairness in different applications. [165], [89], [186] and [90] attempt to de-bias word embeddings used by NLP systems,

while [9,45,202] attempt to de-bias model representations and encodings.

There is also a large body of work on modifying learning algorithms and inference procedures to produce more fair outcomes [4,67,122,138,217]. While effective in many cases, such approaches do nothing to mitigate *human bias* in decisions based on text. Fundamentally, our framework is concerned with stylistic features of human-generated text. Thus, a large body of prior work on methods for unsupervised style transfer are related to our approach [70,118,176,210]. There is also a vast body of work on style obfuscation [13,47,166,185].

Our work is most closely related to [185] and [207]. A4NT [185] attempts to obfuscate text style through automatic re-writing. However, their approach attempts to hide protected attributes simply by mapping the style of one protected category to that of another. In contrast, we seek not to map the author's text to another author's style, but to a central obfuscated style. [207] propose Privacy Aware Text Re-writing (PATR), which takes a similar adversarial learning translation based approach to address this problem and re-write text. One fundamental difference between our style-pooling method and PATR is that we provide the choice of union vs. intersubsection of styles, which is concerned with the societal aspects of removing sensitive attributes, since we are targeting removal of bias. PATR, however, targets privacy and is therefore not concerned with the union vs. intersubsection of sensitive attributes.

Finally, there is a body of work on re-writing text to mitigate the potential biases within the content of the text itself. [120] propose PowerTransformer, which rewrites text to correct the implicit and potentially undesirable bias in character portrayals. [156] propose a framework that addresses subjective bias in text and [52] and [222] introduce approaches to identifying gender bias against women at a comment level and dialect bias in text, respectively. These works focus on the text content, and not on the stylistic features of the author.

3.1.5 Conclusion

We proposed a probabilistic VAE framework for automatically re-writing text in order to obfuscate stylistic features that might reveal sensitive attributes of the author. We demon-

strated in experiments that our proposed framework can indeed reduce bias in downstream text classification. Finally, our model poses two ways of defining a central style. Future work might consider further explorations of alternative notions of stylistic centrality.

Acknowledgments

The authors would like to thank the anonymous reviewers and meta-reviewers for their helpful feedback. We also thank Junxian He for insightful discussions. Additionally, we thank our colleagues at the UCSD Berg Lab for their helpful comments and feedback.

Ethical Considerations

Our proposed model is intended to be used to address a real-world fairness issue. However, this is an extremely complicated topic, and it should be treated with caution, especially upon deploying possible mitigations such as ours. One potential issue we see is the chance that systems like this might obfuscate text by converging towards the majority and erasing styles of marginalized communities. We have tried to address this concern, and raise discussion around it in our introduction and model design, by allowing for multiple operationalizations of a "central" style, and introducing the union and intersection obfuscations. Defining a true notion of centrality that would effectively protect sensitive attributes without erasing any specific styles of writing requires further study.

3.2 Mix and Match: Learning-free Controllable Text Generation using Energy Language Models

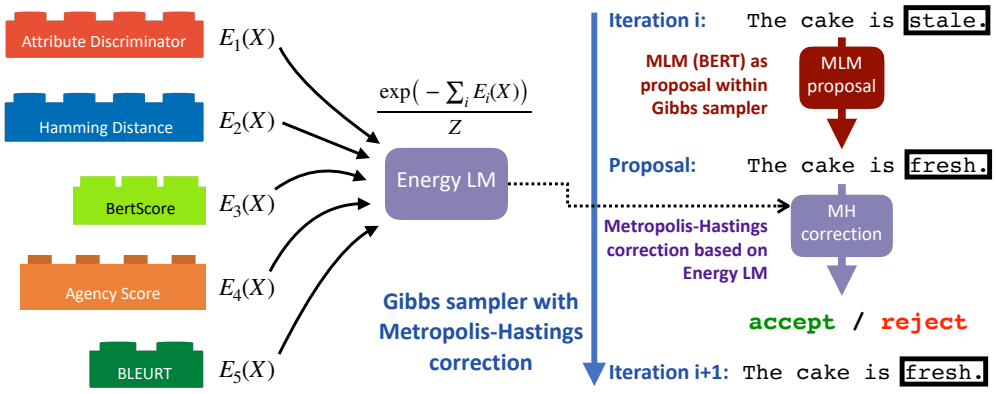


Figure 3.2. Overview of Mix and Match LM. The Lego pieces show different experts that can be used to form the energy LM and help control different features in the generated text. The right side shows the *i*th step in the the Gibbs sampling chain, where a proposal is made by the MLM, and then it is accepted/rejected based on the energy score.

While large transformer-based autoregressive language models trained on massive amounts of data found on the internet exhibit exceptional capabilities to generate natural language text, effective methods for generating text that satisfy global constraints and possess holistic desired attributes remains an active area of research. These mechanisms for controlling the generation of language have the potential to mitigate undesirable biases encoded by the large language models and prevent the generation of hate speech and toxic language [7,57,131,179,206]. Much of the prior work has approached controlled generation via either training domain-conditioned neural language models [51,71,93,99,101,155,167,182] or finetuning/modifying an underlying large pre-trained base model for generation on domain-specific data for attribute sensitive generation [31,65,92,124,223]. Not only do these approaches involve computational overhead and estimation errors associated with the training of language models, but they are also dependent on access to a large amount of attribute-specific language data which can be impractical in many scenarios and exacerbate privacy concerns [17,88,136].

Our approach eschews training and focuses on generation-time control from pre-trained modules. Recent work in this space has used attribute discriminators [33, 78, 98, 208] to steer the generation from a large autoregressive language model. These discriminators need to be separately trained on partial generations in order to be operationalized with step-wise autoregressive models. As a result, this approach also requires availability of data to train step-wise discriminators for attributes that are essentially global (at the sequence-level) in nature. Therefore, we focus on drawing samples from a test-time combination of *pretrained blackbox* experts that each score a desired property of output text – for example, fluency, attribute sensitivity, or faithfulness to the context. Specifically, we view the product of these black-box experts as a probabilistic energy model [73] – i.e., a *non-autoregressive*, globally normalized language model – and then sample (without further training or fine-tuning) using a specialized Gibbs sampler with a Metropolis-Hastings correction step [60].

Our full framework, which we entitle Mix and Match LM (depicted in Figure 3.2), enables the generation of high-quality attribute-controlled samples by mixing and matching black-box models like off-the-shelf pre-trained attribute-sensitive discriminators (e.g., sentiment classifiers), large bidirectional pre-trained language models like BERT [38], and other modules specializing in capturing desirable features pertaining to faithfulness to any additional context, like hamming distance, or BertScore distance [221] between the sample and the conditioning context. We generate samples from the energy language model assembled from these component experts by using the recently proposed Gibbs-Metropolis-Hastings scheme [60] for sampling from energy models using a masked language model as a proposal distribution. In this scheme, an expressive bidirectional language model like BERT is used to make a proposal at each transition step in the Gibbs chain to jump to a sequence $\bar{\mathbf{x}}$ from the current sequence $\mathbf{x}$. This proposal's fitness is judged by the change in the energy language model's score, with the sampler accepting proposals with larger energy reductions at a higher rate. While the MCMC nature of our sampler negatively impacts the runtime during decoding

compared to autoregressive approaches with ancestral sampling, we find our approach to still be practical and yield high-quality diverse samples that respect the distribution induced by the product of expert black-box models.

We demonstrate the flexibility of our approach by performing a variety of controlled generation tasks, such as aspect-based text revision, style transfer, and attribute grounded generation and compare it to recently proposed controlled generation approaches that are more resource/data intensive. We observe that our approach, which does not require any gradient optimization and is able to combine arbitrary heterogeneous black-box models, outperforms other approaches according to various automated metrics of fluency, quality, and control, as well as human evaluations.

3.2.1 Related Work

The approaches closest in spirit to our work involve steering generation from a base language model with external attribute-sensitive control mechanisms. Plug-and-Play LM [33] uses discriminators learned from an autoregressive LM's top-level hidden layer to modify the LM's states toward increasing the probability of the desired attribute via gradient ascent at each step. GeDi [98] and FUDGE [208] take a similar approach but train custom step-wise attribute-sensitive discriminators that decide whether the desired attribute is likely to be satisfied by the current generation path. GeDi trains class-conditional language models for these discriminators and hence additionally relies on access to attribute sensitive language data. [100] formulate the task of controlled generation as optimizing the base LM's likelihood subject to global differentiable attribute-based constraints by gradient descent over the position-wise simplexes over the vocabulary. DExperts [115] is another decoding-time controllable generation approach that modifies the step-wise softmax logits of an autoregressive pre-trained LM with softmax logits of separately trained domain-specific *expert* autoregressive language models. These approaches require training of custom modules and do not readily enjoy the benefits

of incorporating global attribute-based features into the generation mechanism in a simple probabilistic manner. In contrast, following the findings related to implicit energy-based models trained via non-probabilistic objectives [60, 62], our energy-based formulation is not only optimization-free but also fully modular and able to easily incorporate global features, allowing for heterogeneous black-box experts to be combined with each other.

3.2.2 Mix-and-match Language Models

In this subsection, we describe our approach and motivation behind our method. Specifically, we frame the problem of performing controlled generation as a problem of sampling from a specialized energy-based (or globally normalized) sequence model that defines a probability distribution that satisfies the desired constraints we wish to impose in the controlled generation setting. As described below, this energy-based model is composed of pre-trained components and does not require any further optimization. An energy-based sequence model defines the probability distribution over the space of possible sequences $\mathcal{X}$ as:[4] $p(X;\theta) = \frac{e^{-E(X;\theta)}}{\sum_{X' \in \mathcal{X}} e^{-E(X';\theta)}}$, where $E(X;\theta)$ refers to the scalar energy of a sequence X that is parametrized by θ. Lower energy corresponds to the higher likelihood of X. In contrast to the common autoregressive sequence models, exact likelihood computation and efficient sampling from these models is challenging. Despite these challenges, we focus on this paradigm of sequence modeling because energy-based models offer increased flexibility via sequence-level features and constraints. As we discuss next, this capability lets us easily define expressive functions for controlled generation of sequences which is not readily offered by the autoregressive modeling paradigm.

[4]For simplicity, we are concerned with a finite set of sequences limited by some maximum length.

3.2.2.1 Product of Experts Energy-based Models and Controlled Generation

Our approach is motivated by the perspective that the task of controlled generation requires concentrating probability mass over a small subspace of sequences in $\mathcal{X}$ that satisfies various constraints pertaining to fluency, target attributes, and other control variables. Consider the task of generating positive sentiment sentences. This requires satisfaction of two major constraints: (1) The sequence X should be well-formed, (2) The sequence X should express positive sentiment. If we have access to two separate probability distributions over $\mathcal{X}$, one for modeling well-formedness ($p_1(X)$) and another for modeling positivity ($p_2(X)$), then a natural solution for controlled generation in this setting would be to draw samples from a probability distribution that is a product of these two distributions i.e. $p_{\texttt{desire}}(X) \propto p_1(X) \cdot p_2(X)$. In our approach, we further relax this requirement by assuming access to *expert blackboxes* that yield scalar non-probabilistic energy scores E_1 and E_2 indicating fitness of a sequence w.r.t. well-formedness and positivity respectively. Under the product of experts framework above the desired probability distribution would take the form: $\log\ p_{\texttt{desire}}(X) = -(E_1(X) + E_2(X))\ -\ \log Z$. This expression shows that when working with scalar scores for the expert black-boxes, the product of expert models yields an energy model whose energy is simply the sum of the scalar energy values obtained from the expert models. Inspired by this, we propose a framework for controlled generation that involves linear combinations of various black-box experts in order to obtain a distribution whose samples satisfy the requirements of a desired controlled generation task: $E_{\texttt{M\&M}}(X) = \sum_{i=1}^{k} \alpha_i E_i(X)$, where our proposed *mix-and-match* energy is composed of k expert energy components, which are weighted by scalar hyperparameters α.

3.2.2.2 Expert Factors in Mix-and-Match LM

As shown in Fig. 3.2, we use the following black-box experts in our experiments as modules that we can add or remove to produce desired behavior:

$\mathbf{E}_{\texttt{mlm}}(\mathbf{X})$: Recent work has shown that large masked language models (MLM) like BERT

can discriminate between well-formed and ill-formed sentences [221] and induce an implicit energy function over the sequences [60]. Hence, we use BERT-base as a black-box to model the form and fluency of sentences. Specifically, we use an energy parametrization introduced in [60] which is negative of the sum of unnormalized logits iteratively computed at each position obtained via the forward pass of the MLM after masking the corresponding position.
$\mathbf{E}_{\texttt{disc}}(\mathbf{X})$: This particular expert module refers to the energy obtained via the discriminator for the attributes of interest. What this module returns is the raw logits of the discriminator, for the target attribute. For instance, if we have a sentiment classifier, and want to produce positive sentiment, then $E_{\texttt{disc}}(X) = -\log\, p(+|X)$.
$\mathbf{E}_{\texttt{hamm}}(\mathbf{X};\mathbf{X}')$: For a given sequence X', this quantity refers to the hamming distance between the sequence X and X'. This penalizes token level deviation from X' which is useful if we are interested in only making minor edits to X' as described later.
$\mathbf{E}_{\texttt{fuzzy}}(\mathbf{X};\mathbf{X}')$: Similar to the hamming distance, this quantity refers to the BertScore [221] computed between X and X' which can be viewed as a *fuzzy* hamming distance that takes semantic similarity into account.

3.2.2.3 Sampling scheme

To sample from the energy parametrizations described in the previous subsection, we follow the Metropolis-Hastings [69] MCMC scheme for sampling from the masked language models introduced by [60]. While the proposal distribution we use is the same as [60] i.e. masked language model's (BERT's) conditionals, the energy parametrizations we use are more suitably designed for controlled generation.

We briefly explain the sampling procedure, which involves forming long Markov chains of sequences starting with a random sequence, and following the MH scheme which uses a proposal distribution to propose a new sequence at each step in a chain which is either accepted or rejected based on its fitness to the energy function. The sequences at the end

of these chains correspond to samples from the desired energy-based model. Operationally, at each MCMC step, we mask out a token at a random position in the current sequence X in the chain and propose a new sequence $\bar{X}$ to transition to by sampling a token from the MLM conditional softmax at the masked position. This proposed sequence is evaluated by its ability to reduce the energy from the current sequence in the chain and is accepted with the probability $p(\bar{X};X) = \min\left(1, \frac{e^{-E_{\texttt{M\&M}}(\bar{X})}\ p_{\text{mlm}}(X_i|X_{\setminus i})}{e^{-E_{\texttt{M\&M}}(X)}\ p_{\text{mlm}}(\bar{X}_i|X_{\setminus i})}\right)$. $E_{M\&M}(X)$ refers to the product of experts energy, i refers to the position chosen for masking, p_{mlm} refers to the MLM's conditional distribution at the `[MASK]` position. Intuitively, this acceptance probability indicates that the proposed sequence $\bar{X}$ is more acceptable if it has lower energy than the current sequence X in the chain and is rare or less likely to be proposed by the proposal distribution again.

3.2.2.4 Controlled generation Tasks

We use the expert black-box factors and the sampling scheme described above in our framework to perform two kinds of controlled generation tasks.

Prompted generation: This task focuses on generating well-formed sentences that start with a specified prompt and also satisfy a target attribute for which we have access to a discriminator. An example task would be to generate positive sentiment sequences starting with `This movie`. The energy function takes the form:

$$E_{\texttt{gen}}(X) = E_{\texttt{mlm}}(X) + \alpha\ E_{\texttt{disc}}(X) \tag{3.10}$$

α is a hyperparameter that controls the tradeoff between the MLM score and the discriminator's influence. For MH-based sampling for this task, we initialize the sequence with the starting prompt and the rest of the tokens masked out, which creates a seed text of shape `the movie [MASK] [MASK] ... [MASK]`, for the prompt example of `the movie`. The number of mask tokens depends on the target generation length, and we constrain the sampler to only

produce proposals and revise non-prompt tokens, and mark the prompt tokens as "frozen".

Controlled text revision: This task involves editing a source sequence X' in order to satisfy the desired target attributes exhibited by the generated sequence X. The energy function for this task is:

$$E_{\texttt{rev}}(X) = E_{\texttt{gen}}(X) + \beta\ E_{\texttt{hamm}}(X, X') + \gamma\ E_{\texttt{fuzzy}}(X, X') \tag{3.11}$$

This energy function in addition to valuing well-formedness and satisfying target attribute requirements also focuses on maintaining faithfulness to the source sequence X'. For sampling with this energy, we initialize the sequence with the sequence X' to be edited. This sets the length of the target sequence to be the same as the source. In this setup, the sampler can revise all tokens and is not constrained.

For both these tasks, we run a separate MCMC chain for each generated sentence for 8 to 15 epochs, depending on the task. An epoch refers to one masking cycle over all the non-frozen positions (selected randomly) of the sequence.

3.2.3 Experimental Setup

Here we provide a brief overview of the tasks, datasets, baselines, and metrics used in the experiments.

3.2.3.1 Tasks and Datasets

Controllable debiasing (ROC story corpus): We use the subset of the ROC story corpus [142] test-set that is used by PowerTransformer [121] for their evaluations. We use this data for controllable debiasing, a text revision task which aims to correct the implicit and potentially undesirable agency biases in character portrayals, by replacing verbs such as "wish" and "dream", with "pursue" and "achieve".

Sentiment transfer (Yelp): We use Yelp [182] dataset's test-set for the task of sentiment transfer.

The test set comprises 1000 sentences, half with positive and half with negative sentiment. We also have a reference set of handwritten sentiment transferred sentences, provided by [71] that we use for reporting evaluation metrics.

Formality transfer (GYAFC): We use 1051 sentences from the entertainment and music domain subset of the GYAFC [163] dataset, which contains formal and informal sentences for the task of formality transfer (both directions of formal to informal and informal to formal).

Prompted generation: We evaluate our approach on two forms of prompted generation: 1) sentiment controlled generation and 2) topic controlled generation. For sentiment controlled generation, we set Mix and Match LM to generate text with positive or negative sentiment given prompts, by using a Yelp sentiment classifier as discriminator and compare against PPLM [33] which is a popular sentiment controlled generation method. For topic controlled generation, we compare against FUDGE [208], and follow their experimental setup consisting of 7 distinct topics and 20 prompts.

3.2.3.2 Expert Component Configurations

We use a Huggingface pre-trained `bert-base-uncased` model as our MLM for yielding $E_{\texttt{mlm}}$ and also providing the proposal distribution in our MH MCMC sampler. For obtaining $E_{\texttt{disc}}$, we train BERT-based classifiers on the training-set of our datasets to use as our attribute discriminators. We could have used any pre-trained attribute classifier from Huggingface for $E_{\texttt{disc}}$, but we keep those aside to use as external attribute classifiers for fair evaluation against baselines. For experiments in which we add the BertScore [221] component to the energy, we use the pre-trained `roberta-large_L17` model. Finally, for agency score, we use the lexicon provided by [178] and check each generated sequence and count the number of target agency verbs that exist there. The count becomes the agency score.

3.2.3.3 Baselines

PowerTransformer. For the task of controllable debiasing (agency revision), we compare our work with PowerTransformer [121], an approach that uses paraphrasing and self-supervision based on a reconstruction loss, building on pre-trained language models, to re-write text and control agency level of sentences.

[71] For style transfer on sentiment an formality, we compare with [71], a generative style transfer framework which uses a variational autoencoder (VAE) built using a sequence-to-sequence LSTM-based model to do unsupervised style transfer. This framework needs to be trained from scratch for each style transfer task.

UNMT. As a second baseline for style transfer, we use UNMT [101], an unsupervised machine translation framework that demonstrates high performance for sentiment transfer.

PPLM. For the task of sentiment controlled generation, we compare to Plug-and-Play LM (PPLM) [33], which does attribute controlled generation using the flow of gradients from discriminators trained on the last hidden layer representations of the generator, to guide generation.

FUDGE. This approach [208] trains step-wise discriminators on partial generations from GPT-2 to determine whether the constraints related to desired attributes will be satisfied by the future completion of the sequence or not. We compare against this on topic controlled generation as this approach was shown to be superior to PPLM on this task.

3.2.3.4 Evaluation Metrics

We use a variety of evaluation metrics to compare our approach's performance on two major facets: (1) Quality of generated text, and (2) success on matching the target attribute used for control.

3.2.3.4.1 Text Quality and Semantic Similarity

GPT-2 PPL. We feed our generated test sentences to a Huggingface [159] pre-trained GPT-2 xl model, and report its perplexity (PPL), as an automatic measure of fluency. Although this measure is not a perfect indicator of fluency, we find it to be a useful metric alongside human judgements. [5]

BLEU. For sentiment (Yelp) and formality (GYAFC) transfer where we have reference text, we report the BLEU score. For controlled debiasing, we report BLEU between generated text and source and show it as BLEU (src).

BertScore. As a measure of meaning preservation, we use the F1 BertScore metric [221] to compare the semantic similarity of the provided reference sentence with the generated output.

Hamming Distance. We also report the hamming distance between the source text and generated text, to measure the extent of the change.

3.2.3.4.2 Attribute Quality

Internal Classifier Accuracy. We report the accuracy of the internal classifier (the discriminator used for generation) on the generated text, assuming the target attribute is the correct label. The higher this accuracy is, the better.

External Classifier Accuracy. It is natural to get high accuracy on the internal classifier, since we are sampling from it. To have a fair comparison, we report accuracy using external classifiers from Huggingface (`textattack/bert-base-uncased-yelp-polarity` [141] for sentiment and `cointegrated/roberta-base-formality` for formality).

Agency Lexicon Accuracy. For controlled debiasing, we measure the accuracy of the change in agency by comparing the target agency level with that of the generated text, extracted using the connotation frames lexicon, and following the setup from [121].

[5] Due to the high variance in the PPL scores generated across sentences by GPT-2, we report the median score for each system under comparison.

Table 3.7. Original and style transferred sample sentences, using Mix & Match LM. Sentiment shows the task of sentiment transfer, from negative to positive and positive to negative, on Yelp. Agency shows the controllable agency de-biaisng task [121]. In the examples, we are transferring negative agency to positive.

	Original	Transferred
Sentiment	the food 's ok , the service is among the worst i have encountered .	the food 's wonderful , the service is among the finest i have encountered .
	we will not be using this location again .	we will definitely be seeking this location again .
	good selection of parts and accessories and reasonable prices .	poor selection of parts and accessories and high prices .
	it is a cool place , with lots to see and try .	it is a stupid place , with nothing to see and try .
Agency	mary needed new shoes .	mary got new shoes .
	she followed the instructions as best as she could .	she executed the instructions as best as she could .
	pam wanted to have a special cake for her son 's birthday .	pam decides to have a special cake for her son 's birthday .
	whitney is going to fail her test .	whitney is set to get her test .

Table 3.8. Controllable debiasing/ sentence agency revision on ROC-story corpus. The *(src)* next to the metrics denotes measurement with respect to the source text. *Int. Clsf.* is the accuracy of the discriminator used in the energy. *Hamm.* shows the Hamming distance. *Agency Acc.* is the accuracy of agency revision based on the agency lexicon (Sec 3.2.3.4.1).

	Method	BLEU(src)	GPT-2	BertScore(src)	Hamm.(src)	Int. Clsf.	Agency Acc.
	Source Text	100.00	153.9	1.00	0.00	7.47	9.81
Basel.	**PowerTransformer (No Boost)**	60.30	210.8	**0.94**	1.11	64.84	**69.17**
	PowerTransformer (+Boost)	57.46	247.2	**0.95**	1.28	77.23	**85.03**
Ours	M&M LM Verb Replace (Disc)	60.53	238.7	0.95	1.04	81.05	70.80
	M&M LM Verb Replace (Agency Score)	63.34	193.3	0.96	0.89	32.42	64.75
	M&M LM Verb Replace (Disc+Agency Score)	54.52	248.8	0.95	1.05	77.23	77.27
	M&M LM (Hamming +Disc)	56.26	211.2	**0.95**	1.37	96.52	**69.00**
	M&M LM (Hamming+Agency Score)	35.26	231.6	0.95	1.56	23.13	86.01
	M&M LM (Hamming+Disc+Agency score)	39.82	261.6	**0.93**	2.45	90.16	**89.42**

3.2.4 Results

3.2.4.1 Controllable Debiasing

Tables 3.7 and 3.8 show our results for the task of text revision for controlling agency bias which is introduced by PowerTransformer [121], our Baseline for this task. PowerTransformer has a vanilla (no boost) variant and a variant with vocab boosting, which up-weights the logits of verbs that belong to the target agency lexicon so as to increase their probability and incentivize generation in that direction. We also measure our metrics on the original

test-set, without revision, to provide a better sense of the changes made.

We offer different variants of our framework, to provide a fair comparison and to better ablate our proposed method. "Disc" denotes our framework where we add the discriminator expert ($E_{\texttt{disc}}$) which is trained to predict the agency level of a sentence, to the energy along with $E_{\texttt{mlm}}$, and $E_{\texttt{hamm}}$ (Eq. 3.2.2.4). Hamming distance is computed between the generated proposals and the source sentence. The "Agency Score" variant adds an alternative term to $E_{\texttt{M\&M}}$ instead of $E_{\texttt{disc}}$, which is the number of target agency verbs according to the connotation frames lexicon [178] in the sentence. The "Disc+Agency" variant has both energy components. We also apply our method in two ways: "Verb Replace" which allows the sampler to propose revisions for only one pre-determined verb (provided in the dataset). In this setup, all tokens remain frozen, except for the given verb. The conventional mode (M&M LM), however, proposes revisions for all tokens in the sentence and is not constrained.

Table 3.8 shows that in the conventional setup, Mix and Match LM (Disc only) has performance similar to that of PowerTransformer, without boosting. With the Agency Score component, our method outperforms PowerTransformer in terms of accuracy of revision as per the agency lexicon accuracy metric, with negligible loss in meaning (BertScore). The reason behind this better performance in terms of applying target agency accuracy is that our method's sampling is guided by the energy that is directly built on the metrics we care about, as opposed to trying to apply them through paraphrasing and proxies such as vocab boosting, which are employed in the PowerTransformer method.

Another important observation here is the difference between "Verb Replace" and conventional modes. This ablation shows that although our method makes few changes (the average Hamming distance between source and output sentences are between 1.37 and 2.45), it still outperforms a "static" method that has extra knowledge of the offending verb and focuses on changing only that verb, by a significant margin.

Table 3.9. Sentiment transfer on Yelp. *(ref)*/*(src)* means the metric measured is measured with respect to reference/source text. *Int.*/*Ext. Clsf.* show internal/external attribute classifier accuracy. *Hamm.* shows Hamming distance.

	Method	BLEU(ref)	GPT-2	BertScore(src)	Hamm.(src)	Int. Clsf.	Ext. Clsf.
	Reference Text	100.00	169.5	1.00	5.80	83.70	85.60
Basel.	[71]	18.67	200.6	0.93	4.23	84.87	79.82
Basel.	**UNMT**	17.00	171.8	**0.94**	3.67	84.87	**80.22**
Ours	M&M LM (Discriminator ↑)	15.75	163.5	0.93	2.84	97.53	90.00
Ours	**M&M LM (Hamming↑)**	19.71	191.5	**0.95**	1.83	94.72	**82.85**

Table 3.10. Formality transfer on GYAFC dataset. The *(ref)*/*(src)* next to the metrics denotes that they are measured with respect to the reference/source text. *Int. Clsf.* shows the accuracy of the discriminator used in the energy, and *→Informal*/*Form.* shows the breakdown of the external classifier accuracy. *Hamm.* shows the Hamming distance.

	Method	BLEU(ref)	GPT-2	BertScore(src)	Hamm.(src)	Int. Clsf.	→Informal	→Form.
	Reference Text	100.00	118.1	0.92	7.72	82.97	100.00	9.41
Basel.	**[71]**	15.83	122.8	**0.90**	10.03	64.79	**100.00**	**3.33**
Basel.	UNMT	14.17	143.8	0.90	11.92	56.04	99.81	7.64
Ours	M&M LM (Discriminator ↑)	17.78	206.3	0.89	5.22	91.15	96.67	23.13
Ours	**M&M LM (BertScore↑)**	27.71	194.4	**0.93**	2.50	72.12	**94.26**	**19.01**

3.2.4.2 Style Transfer

In this subsection, we experiment with sentiment and formality transfer, where Sentiment transfer needs fewer changes and formality transfer needs more structural change to the original sentence. We show sample sentences and transfers in Table 3.7 (we cannot show samples for formality as the dataset is not public).

3.2.4.2.1 Sentiment Transfer

For this task, we include two components in our energy model, the attribute discriminator ($E_{\texttt{disc}}$), to induce the target style, and the hamming distance ($E_{\texttt{hamm}}$), to maintain the meaning of the sentence. We don't include the more complex semantic similarity-related component like $E_{\texttt{fuzzy}}$, since sentiment transfer can normally be done by making only a few changes to the sentence. We report results with two different variants, one where the

discriminator component has a higher coefficient in the energy (Discriminator↑) and one where the hamming distance has a higher coefficient (Hamming↑). In effect, these two show the trade-off between transfer quality and faithfulness to the source sentence.

We see in Table 3.9 that our method, with the hamming component up-weighted, outperforms both the generative baselines in terms of transfer accuracy (Ext. Clsf.) and semantic similarity (BertScore). We can also see Mix and Match LM has higher BLEU score, with respect to the provided hand-written reference sentences. We hypothesize that this superiority is due to the tendency of our model to make minimal revisions that satisfy the product of experts energy model. Therefore, our model can successfully change the style without changing the meaning of the sentence. The generative baselines, however, regenerate the sentence which imposes more change, as can be observed from the hamming distance column (Hamm.(src)) in Table 3.9.

3.2.4.2.2 Formality Transfer

For this task, we include the formality classifier ($E_{\texttt{disc}}$), Hamming distance ($E_{\texttt{hamm}}$), and BertScore ($E_{\texttt{fuzzy}}$) components in the energy formulation, to permit the transfer of style and also maintain the meaning of the sentence. $E_{\texttt{fuzzy}}$ helps with imposing semantic similarity between the source and generated sentences, since Hamming alone isn't sufficient for judging comparable formal and informal sentences. We show results for two setups of our framework, one where the discriminator coefficient is higher (Discriminator↑) and another where the BertScore coefficient is higher (BertScore↑).

In Table 3.10 we have broken down the external classifier accuracy for the different transfer directions of formal to informal ($\rightarrow$ Inf.) and vice versa. We do this because the $\rightarrow$ Form. task is generally harder and therefore has lower accuracy. We observe that our method outperforms the baselines in terms of BertScore and BLEU, for similar levels of external classifier accuracy. However, we can see that the GPT-2 PPL of our method is higher than the baselines. The reason behind this is the format and noise in the data. The samples for

Table 3.11. Samples of prompted sentiment controlled generations, using our Mix and Match LM and PPLM.

	Ours (Mix and Match LM)	PPLM
Pos Sent.	the country is noted for attracting a quarter-million tourists.	the country's top cycling event is right behind the olympics, and the
	the lake we come across can be said to be beautiful.	the lake is a great spot for swimming, diving and snorke
	the chicken and all the other ingredients produced a delicious meal.	the chicken wing is one of the best foods you can eat and it
	the movie was family-friendly and a success in japan.	the movie, which is currently only the third the the the the the
Neg Sent.	the country was unstable and was not ready to modernize.	the country's top animal welfare agency, the ministry of agriculture and food
	the lake was not supposed to be navigable under any circumstances.	the lake, a large, and the most massive and most terrible of
	the chicken was growling and beginning to feel a little sick.	the chicken noodles are the most horrible food i have ever had.
	the movie received only two nominations and earned no grand prix.	the movie is not in the , a, a, a

Table 3.12. Prompted sentiment controlled generation results and human evaluations.*BERT* denotes the BERT MLM energy score (equivalent of GPT-2 perplexity), and lower score is better. *Int./Ext. Clsf.* show the accuracy of the discriminator used in the energy/external discriminator from Huggingface.

Length	GPT-2 (↓)		BERT (↓)		Int. Clsf. (↑)		Ext. Clsf. (↑)		Human Preference (%)	
	Ours	PPLM	Ours	PPLM	Ours	PPLM	Ours	PPLM	Ours	PPLM
12	264.1	113.1	−160.4	−137.1	94.3	71.7	65.1	58.0	71.1	29.9
20	167.2	61.1	−271.0	−237.1	96.3	74.5	65.9	57.6	62.9	37.1
50	122.3	29.0	−692.3	−606.1	93.8	73.6	68.6	60.7	46.7	53.3

this dataset are taken from the music and entertainment industry domain and contain some symbols and characters similar to emojis (e.g. ":)" and "***"). This is where the tendency of our approach toward minimal revisions is hurtful–our revisions of text, often do not get rid of all of these symbols, while the baselines' generative methods successfully remove all the superfluous characters because they rewrite sentences from scratch.

3.2.4.3 Prompted Controlled Generation

3.2.4.3.1 Sentiment Controlled Generation

We generate 560 sequences of different lengths (12, 20 and 50 tokens), given 14 prompts, 2 sentiments, and 20 sequences per sentiment, taken from [33]'s experimental setup.

Table 3.12 shows our results for this experiment. Here, we have an additional metric, the MLM energy (lower is better), which, like GPT-2, indicates the quality of generated sentences [172] according to BERT. We report this extra metric here since PPLM uses a GPT model for generation, and it is natural that it would measure better on this metric. The table shows that for all lengths of generated sentences, our method is much better at inducing

the target sentiment. However, we observe that PPLM performs better in terms of GPT-2 while our method performs better on the MLM energy metric. This suggests the tendency of model-based fluency metrics to be biased toward the corresponding models as the PPLM uses GPT-2 for generation and M&M LM uses BERT. To enable a more conclusive comparison of the text quality, we report results with human evaluations. For these evaluations, we randomly select 10 generated outputs for each prompt, per sentiment (240 overall), and asked three Amazon Turkers per sample pair, which sample they find more fluent. We report the majority vote of the Turkers in the table. The results show that for sequences with lengths 12 and 20, they found our generations more fluent. However, for length 50, the preference rate for M&M drops to 46.7%, which shows that our method is superior to PPLM for short/medium length generation, however, PPLM does better at generating longer sequences.

3.2.4.3.2 Topic Controlled Generation

We follow FUDGE's [208] experimental setup which covers 7 topics, given 20 prompts and generate 7×20 sequences of length 20. To enforce topicality on our generations, we add a topic-based energy, $E_{\texttt{topic}}$. This energy is essentially the negative count of the number of topic-related words (using the list provided by FUDGE).

Table 3.13 shows the results of this experiment. Topic-score (↑) is the usage rate of topic-related words that were used for training and evaluation of topic controlled generation by [208] in their paper. Grammaticality (↑) is the score of grammaticality given by a Roberta-based CoLA grammaticality model averaged over all outputs [203]. The "Div" (↑) metrics show the diversity of generated text, over unigrams, bigrams and trigrams. Finally, the human evaluations show human preference, in terms of fluency of the sentences. As shown by the table, the fluency of our method is comparable to that of FUDGE, even better in terms of human preference and grammaticality judgment. FUDGE has a slightly higher topic score, which is expected since it trains a custom step-wise discriminator for each topic that is op-

Table 3.13. Prompted topic controlled generation results and human evaluations.

Metrics	FUDGE	**M&M LM**
Topic-score (↑)	1.45	1.21
Grammaticality (↑)	0.61	0.74
GPT-2 PPL (↓)	104.8	110.2
Diversity over Unigrams (↑)	0.54	0.57
Diversity over Bigrams (↑)	0.86	0.89
Diversity over Trigrams (↑)	0.87	0.88
Human Preference(%) (↑)	36.5	63.5

timized for the task. But our approach shows competitive faithfulness to the topics especially considering the fact that prompted GPT-2 generations without the FUDGE discriminators only achieve a topic-score of 0.23.

3.2.4.4 Inference Speed

Given that our model's inference procedure involves MCMC sampling, it's reasonable to expect its run-time to be slower than more traditional baselines. For sequences of length 20, we find that our un-optimized implementation requires 8 seconds per generation and 3 seconds per revision – while, in contrast, baseline system PPLM requires 16 seconds and FUDGE requires 0.4 seconds per generation. This is a substantial slowdown compared to FUDGE, but not one that renders the proposed approach impractical in offline settings. Further, faster sampling schemes are beyond the scope of this dissertation but might be explored in future work to speed up models like M&M LM.

3.2.5 Conclusion

We present Mix and Match Language Models (M&M LM), a training-free framework for controlled text generation that can easily mix heterogeneous expert modules. We show that our framework outperforms prior methods on a suite of text revision and attribute-controlled generation tasks. Further, our results indicate that probabilistic energy language models, typically considered intractable, can be used for practical text generation tasks when combined

with an appropriate sampling scheme.

Acknowledgments

The authors would like to thank the anonymous reviewers and meta-reviewers for their helpful feedback. We also thank our colleagues at the UCSD/CMU Berg Lab for their helpful comments and feedback.

Ethical Considerations

The proposed approach takes steps towards a novel paradigm that might partially mitigate the need for energy-intensive GPU training – potentially leading to positive environmental impact down the line. The approach may also have positive impacts on accessibility as strong computational resources are not required when setting up a new controlled text generation system. We do however acknowledge that strong controlled generation methods that rely on discriminators have the potential to regurgitate sensitive training data and produce harmful outputs and toxic language [57, 200, 206]. However, if used properly and for good, we anticipate a positive impact on debiasing and safe generation.

Chapter 4

Conclusion

This dissertation seeks to probe and analyze the privacy leakage and fairness of large language models and offers practical, low-overhead, and accessible methods to improve these aspects of large models. The over-arching goal of this thesis is to take a step towards providing principled privacy analysis tools and mechanisms, which can be employed by users and deployers to probe the leakage of models and quantify their risks. We only focused on membership inference attacks against textual models here, however, probing privacy leakage for deploying models in real-world cases needs to go beyond that, considering different data modalities and more sophisticated extraction attacks. As multi-modal and vision-grounded models are becoming more ubiquitous [111], it is imperative that the community pays as much attention to analyzing the safety risks of these models, as they do to improving the state-of-the-art performance on a limited suite of target tasks.

We also discussed and introduced privacy mitigation methods that limit the memorization of language models and rely on differential privacy and adversarial learning. Differential privacy provides worst-case guarantees and protects the membership of any given "record" which is a well-defined interpretable guarantee for some data modalities, such as tabular data. For language, however, what a record should be and what protecting it using DP really means is not clear. Recent work (and work proposed in this dissertation) uses DP-SGD to limit leakage of language models while choosing each sentence to be a single record. This definition

of a record, however, is not very accurate as the amount of information in each sentence can vary widely, and the correlations between different records is unaccounted for. Apart from this issue, the research community still does not have a proper policy or protocol for how the privacy parameters should be set, and what the provided protection achieves. As such, we need to take a step back and re-think how we perceive privacy for language, and come up with new frameworks and definitions which rely more on how humans reason about privacy.

Finally, we proposed two methods for attribute-controlled generation of text to enforce certain attributes while hiding others and evaluate them on how well they can fool attribute classifiers. We use existing pre-trained building blocks to do controlled text generation, without the need to train or fine-tune any new models. This approach aims to increase the accessibility of NLP methods by promoting model reuse, as the size of large language models makes their training and fine-tuning increasingly challenging for researchers with limited resources. Given that the community is shifting towards using larger models, it's critical to ensure that all researchers can access these models and utilize them for their own research. By doing so, we can encourage diverse groups to conduct extensive evaluations that could ultimately enhance the current state of NLP.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, 2016. ACM.

[3] Abubakar Abid, Maheen Farooqi, and James Zou. Large language models associate muslims with violence. *Nature Machine Intelligence*, 3(6):461–463, 2021.

[4] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69. PMLR, 2018.

[5] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. In *Advances in Neural Information Processing Systems*, volume 32, pages 15479–15488, 2019.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.

[7] Ashutosh Baheti, Maarten Sap, Alan Ritter, and Mark Riedl. Just say no: Analyzing the stance of neural dialogue generation in offensive contexts. *arXiv preprint arXiv:2108.11830*, 2021.

[8] Soumya Barikeri, Anne Lauscher, Ivan Vulić, and Goran Glavaš. RedditBias: A real-world resource for bias evaluation and debiasing of conversational language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1941–1955, Online, August 2021. Association for Computational Linguistics.

[9] Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. Adversarial removal of demographic attributes revisited. In *EMNLP/IJCNLP*, 2019.

[10] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 464–473, Washington, DC, USA, 2014. IEEE Computer Society.

[11] Priyam Basu, Tiasa Singha Roy, Rakshit Naidu, Zumrut Muftuoglu, Sahib Singh, and Fatemehsadat Mireshghallah. Benchmarking differential privacy and federated learning for BERT models. In *Machine Learning for Data Workshop at ICML 2021*, June 2021.

[12] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

[13] Janek Bevendorff, Martin Potthast, Matthias Hagen, and Benno Stein. Heuristic authorship obfuscation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1108, 2019.

[14] Su Lin Blodgett, Lisa Green, and Brendan O'Connor. Demographic dialectal variation in social media: A case study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130, Austin, Texas, November 2016. Association for Computational Linguistics.

[15] Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1004–1015, Online, August 2021. Association for Computational Linguistics.

[16] Hannah Brown, Katherine Lee, FatemehSadat Mireshghallah, R. Shokri, and Florian Tramèr. What does it mean for a language model to preserve privacy? In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, June 2022.

[17] Hannah Brown, Katherine Lee, Fatemehsadat Mireshghallah, Reza Shokri, and Florian Tramèr. What does it mean for a language model to preserve privacy? *arXiv preprint arXiv:2202.05520*, 2022.

[18] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[19] Zhiqi Bu, Jinshuo Dong, Qi Long, and Weijie J. Su. Deep learning with gaussian differential privacy. *arXiv preprint*, abs/1911.11607, 2019.

[20] Ben Buchanan, Andrew Lohn, Micah Musser, and Katerina Sedova. Truth, lies, and automation. *Center for Security and Emerging Technology*, 1(1):2, 2021.

[21] C Bucilua, R Caruana, and A Niculescu-Mizil. Model compression, in proceedings of the 12 th acm sigkdd international conference on knowledge discovery and data mining. *New York, NY, USA*, 2006.

[22] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles, 2021.

[23] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models, 2022.

[24] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The Secret Sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security Symposium*, 2019.

[25] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium*, pages 267–284, Santa Clara, CA, 2019. USENIX Association.

[26] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models, 2021.

[27] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security Symposium*, 2021.

[28] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

[29] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, pages 1109–1135, 2010.

[30] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

[31] Alexandra Chronopoulou, Matthew E Peters, and Jesse Dodge. Efficient hierarchical domain adaptation for pretrained language models. *arXiv preprint arXiv:2112.08786*, 2021.

[32] Maximin Coavoux, Shashi Narayan, and Shay B. Cohen. Privacy-preserving neural representations of text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10. Association for Computational Linguistics, 2018.

[33] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.

[34] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

[35] Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128, 2019.

[36] Franck Dernoncourt, Ji Young Lee, Ozlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606, 2017.

[37] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[39] Thomas G. Dietterich. Ensemble methods in machine learning. In *MULTIPLE CLASSIFIER SYSTEMS, LBCS-1857*, pages 1–15. Springer, 2000.

[40] Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, and Matt Gardner. Documenting the english colossal clean crawled corpus. *ArXiv*, abs/2104.08758, 2021.

[41] Brian Dolan. Enron executives: What happened, and where are they now? https://www.investopedia.com/enron-executives-6831970.

[42] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '06, pages 486–503, Berlin, Heidelberg, 2006. Springer.

[43] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography*, TCC '06, pages 265–284, Berlin, Heidelberg, 2006. Springer.

[44] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[45] Yanai Elazar and Yoav Goldberg. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, 2018.

[46] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

[47] Chris Emmery, Enrique Manjavacas, and Grzegorz Chrupała. Style obfuscation by invariance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 984–996, 2018.

[48] Tom Farrand, FatemehSadat Mireshghallah, Sahib Singh, and Andrew Trask. Neither private nor fair: Impact of data imbalance on utility and fairness in differential privacy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS 2020), Privacy-Preserving Machine Learning in Practice workshop (PPMLP)*, November 2020.

[49] Tom Farrand, Fatemehsadat Mireshghallah, Sahib Singh, and Andrew Trask. Neither private nor fair: Impact of data imbalance on utility and fairness in differential privacy. In *Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice*, pages 15–19, 2020.

[50] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

[51] Jessica Ficler and Yoav Goldberg. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[52] Anjalie Field and Yulia Tsvetkov. Unsupervised discovery of implicit gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 596–608, 2020.

[53] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[54] Deep Ganguli, Danny Hernandez, Liane Lovitt, Nova DasSarma, T. J. Henighan, Andy Jones, Nicholas Joseph, John Kernion, Benjamin Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Nelson Elhage, Sheer El Showk, Stanislav Fort, Zac Hatfield-Dodds, Scott Johnston, Shauna Kravec, Neel Nanda, Kamal Ndousse, Catherine Olsson, Daniela Amodei, Dario Amodei, Tom B. Brown, Jared Kaplan, Sam

McCandlish, Christopher Olah, and Jack Clark. Predictability and surprise in large generative models. *2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022.

[55] Deep Ganguli, Liane Lovitt, John Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Benjamin Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislav Fort, Zachary Dodds, T. J. Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olsson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom B. Brown, Nicholas Joseph, Sam McCandlish, Christopher Olah, Jared Kaplan, and Jack Clark. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *ArXiv*, abs/2209.07858, 2022.

[56] GDPR Article 29 Working Party. Opinion 05/2014 on "anonymisation techniques". https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf, 2014.

[57] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.

[58] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. Numerical composition of differential privacy. *arXiv preprint arXiv:2106.02848*, 2021.

[59] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[60] Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Exposing the implicit energy networks behind masked language models via metropolis-hastings. *ArXiv*, abs/2106.02736, 2021.

[61] Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Exposing the implicit energy networks behind masked language models via metropolis–hastings. In *International Conference on Learning Representations*, 2022.

[62] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[63] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.

[64] Manish Gupta, Vasudeva Varma, Sonam Damani, and Kedhar Nath Narahari. Compression of deep learning models for nlp. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3507–3508, 2020.

[65] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online, July 2020. Association for Computational Linguistics.

[66] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[67] Xudong Han, Timothy Baldwin, and Trevor Cohn. Decoupling adversarial training for fair NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 471–477, Online, August 2021. Association for Computational Linguistics.

[68] Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5, 1992.

[69] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[70] Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. A probabilistic formulation of unsupervised text style transfer. *arXiv preprint arXiv:2002.03912*, 2020.

[71] Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. A probabilistic formulation of unsupervised text style transfer. In *International Conference on Learning Representations*, 2020.

[72] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022.

[73] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[74] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015.

[75] Sorami Hisamoto, Matt Post, and Kevin Duh. Membership Inference Attacks on Sequence-to-Sequence Models: Is My Data In Your Machine Translation System? *Transactions of the Association for Computational Linguistics*, 8:49–63, 01 2020.

[76] Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124, 2021.

[77] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and L. Sifre. Training compute-optimal large language models. *ArXiv*, abs/2203.15556, 2022.

[78] Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[79] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[80] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *ICML*, 2017.

[81] Sicong Huang, Daniel Wurgaft, Alisa Liu, Jiacheng (Gary) Liu, Max Weiss, Alexis Ross, Tomek Korbak, Gabriel Recchia, Tom Tseng, Joe Cavanagh, Andrew Gritsevskiy, Derik Kauffman, Aaron Kirtland, Ian McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Ameya Prabhu, Aaron Mueller, Najoung Kim, Sam Bowman, and Ethan Perez. Inverse scaling prize: Second round winners. https://irmckenzie.co.uk/round2.

[82] Micheal Hudson. The panama papers: Exposing the rogue offshore finance industry. https://www.icij.org/investigations/panama-papers/.

[83] Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. Social biases in nlp models as barriers for persons with disabilities. *arXiv preprint arXiv:2005.00813*, 2020.

[84] Abhyuday Jagannatha, Bhanu Pratap Singh Rawat, and Hong Yu. Membership inference attack susceptibility of clinical language models, 2021.

[85] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proceedings of ICLR*, 2017.

[86] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions, 2021.

[87] Nikhil Kandpal, Eric Wallace, and Colin Raffel. Deduplicating training data mitigates privacy risks in language models. *arXiv preprint arXiv:2202.06539*, 2022.

[88] Nikhil Kandpal, Eric Wallace, and Colin Raffel. Deduplicating training data mitigates privacy risks in language models. *ArXiv*, abs/2202.06539, 2022.

[89] Masahiro Kaneko and Danushka Bollegala. Gender-preserving debiasing for pre-trained word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, 2019.

[90] Masahiro Kaneko and Danushka Bollegala. Debiasing pre-trained contextualised embeddings. In *Proceedings of the European Chapter of the Association for Computational Linguistics*, 2021.

[91] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[92] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[93] Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. A distributional approach to controlled text generation. In *International Conference on Learning Representations*, 2021.

[94] Fereshte Khani and Percy Liang. Removing spurious features can hurt accuracy and affect groups disproportionately. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 196–205, 2021.

[95] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, 2016.

[96] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[97] Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *European conference on machine learning*, pages 217–226. Springer, 2004.

[98] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. GeDi: Generative Discriminator Guided Sequence Generation. *arXiv preprint arXiv:2009.06367*, 2020.

[99] Kalpesh Krishna, John Wieting, and Mohit Iyyer. Reformulating unsupervised style transfer as paraphrase generation. *ArXiv*, abs/2010.05700, 2020.

[100] Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. Controlled text generation as continuous optimization with multiple constraints. *Advances in Neural Information Processing Systems*, 34, 2021.

[101] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, 2018.

[102] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[103] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[104] Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron Wallace. Does BERT pretrained on clinical notes reveal sensitive data? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 946–959, Online, June 2021. Association for Computational Linguistics.

[105] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[106] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.

[107] Xuechen Li, Daogao Liu, Tatsunori B Hashimoto, Huseyin A Inan, Janardhan Kulkarni, Yin-Tat Lee, and Abhradeep Guha Thakurta. When does differentially private learning not suffer in high dimensions? *Advances in Neural Information Processing Systems*, 35:28616–28630, 2022.

[108] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[109] Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *Proceedings of the 10th International Conference on Learning Representations*, ICLR '22, 2022.

[110] Yitong Li, Timothy Baldwin, and Trevor Cohn. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30, Melbourne, Australia, 2018. Association for Computational Linguistics.

[111] Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. Foundations and recent trends in multimodal machine learning: Principles, challenges, and open questions. *arXiv preprint arXiv:2209.03430*, 2022.

[112] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher R'e, Diana Acosta-Navas, Drew A. Hudson, E. Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel J. Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan S. Kim, Neel Guha, Niladri S. Chatterji, O. Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas F. Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *ArXiv*, abs/2211.09110, 2022.

[113] Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and quantization for deep neural network acceleration: A survey. *arXiv preprint arXiv:2101.09671*, 2021.

[114] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

[115] Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online, August 2021. Association for Computational Linguistics.

[116] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[117] Yunhui Long, Vincent Bindschaedler, and Carl A. Gunter. Towards measuring membership privacy. *ArXiv*, abs/1712.09136, 2017.

[118] Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. A dual reinforcement learning framework for unsupervised text style transfer. *arXiv preprint arXiv:1905.10060*, 2019.

[119] Lingjuan Lyu and Chi-Hua Chen. Differentially private knowledge distillation for mobile analytics. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1809–1812, 2020.

[120] Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. Powertransformer: Unsupervised controllable revision for biased language correction. In *EMNLP*, 2020.

[121] Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proceedings of*

the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7426–7441, Online, November 2020. Association for Computational Linguistics.

[122] David Madras, Toni Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. *Advances in Neural Information Processing Systems*, 31:6147–6157, 2018.

[123] Saeed Mahloujifar, Huseyin A Inan, Melissa Chase, Esha Ghosh, and Marcello Hasegawa. Membership inference on word embedding and beyond. *arXiv preprint arXiv:2106.11384*, 2021.

[124] Florian Mai, Nikolaos Pappas, Ivan Montero, Noah A. Smith, and James Henderson. Plug and play autoencoders for conditional text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6076–6092, Online, November 2020. Association for Computational Linguistics.

[125] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[126] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[127] Harsh Mehta, Abhradeep Thakurta, Alexey Kurakin, and Ashok Cutkosky. Large scale transfer learning for differentially private image classification. *arXiv preprint arXiv:2203.00324*, 2022.

[128] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *arXiv preprint arXiv:2106.08962*, 2021.

[129] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[130] Fatemehsadat Mireshghallah, Arturs Backurs, Huseyin A Inan, Lukas Wutschitz, and Janardhan Kulkarni. Differentially private model compression. *Advances in Neural Information Processing Systems (NeurIPS)*, Dec 2022.

[131] Fatemehsadat Mireshghallah and Taylor Berg-Kirkpatrick. Style pooling: Automatic text style obfuscation for improved classification fairness. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2009–2022, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[132] Fatemehsadat Mireshghallah and Taylor Berg-Kirkpatrick. Style pooling: Automatic text style obfuscation for improvedclassification fairness. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), Selected for* ***Oral Presentation***, November 2021.

[133] Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. Mix and match: Learning-free controllable text generation using energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (ACL, Volume 1: Long Papers)*, May 2022.

[134] Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks, 2022.

[135] Fatemehsadat Mireshghallah, Huseyin Inan, Marcello Hasegawa, Victor Rühle, Taylor Berg-Kirkpatrick, and Robert Sim. Privacy regularization: Joint privacy-utility optimization in languagemodels. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3799–3807, 2021.

[136] Fatemehsadat Mireshghallah, Huseyin Inan, Marcello Hasegawa, Victor Rühle, Taylor Berg-Kirkpatrick, and Robert Sim. Privacy regularization: Joint privacy-utility optimization in LanguageModels. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3799–3807, Online, June 2021. Association for Computational Linguistics.

[137] FatemehSadat Mireshghallah, Huseyin A. Inan, Marcello Hasegawa, Victor Rühle, Taylor Berg-Kirkpatrick, and Robert Sim. Privacy regularization: Joint privacy-utility optimization in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, June 2021.

[138] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Ali Jalali, Ahmed Taha Taha Elthakeb, Dean Tullsen, and Hadi Esmaeilzadeh. Not all features are equal: Discovering essential features for preserving prediction privacy. In *Proceedings of the Web Conference 2021*, pages 669–680, 2021.

[139] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh. Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254*, 2020.

[140] Fatemehsadat Mireshghallah, Archit Uniyal, Tianhao Wang, David Evans, and Taylor Berg-Kirkpatrick. Memorization in nlp fine-tuning methods. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), Selected for* ***Oral Presentation***, December 2022.

[141] John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, 2020.

[142] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California, June 2016. Association for Computational Linguistics.

[143] Sasi Kumar Murakonda and Reza Shokri. Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2020.

[144] Sasi Kumar Murakonda, Reza Shokri, and George Theodorakopoulos. Quantifying the privacy risks of learning high-dimensional graphical models. In *International Conference on Artificial Intelligence and Statistics*, pages 2287–2295. PMLR, 2021.

[145] Yuta Nakamura, Shouhei Hanaoka, Yukihiro Nomura, Naoto Hayashi, Osamu Abe, Shuntaro Yada, Shoko Wakamiya, and Eiji Aramaki. Kart: Privacy leakage framework of language models pre-trained with clinical records, 2020.

[146] Yuta Nakamura, Shouhei Hanaoka, Yukihiro Nomura, Naoto Hayashi, Osamu Abe, Shuntaro Yada, Shoko Wakamiya, and Eiji Aramaki. KART: Parameterization of privacy leakage scenarios from pre-trained language models, 2021.

[147] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019.

[148] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papemoti, and Nicholas Carlin. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 866–882. IEEE, 2021.

[149] Douglas Oard, William Webber, David Kirsch, and Sergey Golitsynskiy. Avocado research email collection. https://catalog.ldc.upenn.edu/LDC2015T03, 2015.

[150] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, 2017.

[151] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with PATE. In *Proceedings of the 6th International Conference on Learning Representations*, ICLR '18, 2018.

[152] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 560–568, 2008.

[153] Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 2019 Workshop on Biomedical Natural Language Processing (BioNLP 2019)*, pages 58–65, 2019.

[154] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *Conference on Empirical Methods in Natural Language Processing (Systems Demonstrations)*, 2020.

[155] Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. Exploring controllable text generation techniques. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1–14, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[156] Reid Pryzant, Richard Diehl Martinez, Nathan Dass, S. Kurohashi, Dan Jurafsky, and Diyi Yang. Automatically neutralizing subjective bias in text. In *AAAI*, 2020.

[157] David Pujol, Ryan McKenna, Satya Kuppam, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Fair decision making using privacy-protected data. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 189–199, New York, NY, USA, 2020. Association for Computing Machinery.

[158] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[159] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[160] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[161] Alan Ramponi and Barbara Plank. Neural unsupervised domain adaptation in NLP—a survey. *arXiv preprint arXiv:2006.00632*, 2020.

[162] Francisco Rangel, Paolo Rosso, Ben Verhoeven, Walter Daelemans, Martin Potthast, and Benno Stein. Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. *Working Notes Papers of the CLEF*, 2016:750–784, 2016.

[163] Sudha Rao and Joel R. Tetreault. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In *NAACL*, 2018.

[164] Nisarg Raval, Ashwin Machanavajjhala, and Jerry Pan. Olympus: sensor privacy through utility aware obfuscation. In *Proceedings on Privacy Enhancing Technologies*, pages 5–25. Sciendo, 2019.

[165] Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*, 2020.

[166] Sravana Reddy and Kevin Knight. Obfuscating gender in social media writing. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 17–26, 2016.

[167] Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. A recipe for arbitrary text style transfer with large language models. *arXiv preprint arXiv:2109.03910*, 2021.

[168] Tobias Richter and Johanna Maier. Comprehension of multiple documents with conflicting information: A two-step model of validation. *Educational psychologist*, 52(3):148–166, 2017.

[169] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

[170] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[171] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of EMNLP*, 2015.

[172] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online, July 2020. Association for Computational Linguistics.

[173] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-Leak: Data set inference and reconstruction attacks in online learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1291–1308. USENIX Association, August 2020.

[174] Ahmed Salem, Yang Zhang, Mathias Humbert, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *ArXiv*, abs/1806.01246, 2018.

[175] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[176] Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. Fighting offensive language on social media with unsupervised text style transfer. *arXiv preprint arXiv:1805.07685*, 2018.

[177] Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A Smith, and Yejin Choi. Social bias frames: Reasoning about social and power implications of language. *arXiv preprint arXiv:1911.03891*, 2019.

[178] Maarten Sap, Marcella Cindy Prasettio, Ari Holtzman, Hannah Rashkin, and Yejin Choi. Connotation frames of power and agency in modern films. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2329–2334, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[179] Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A Smith. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. *arXiv preprint arXiv:2111.07997*, 2021.

[180] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. In *Proceedings of AAAI*, 2006.

[181] Virat Shejwalkar, Huseyin A Inan, Amir Houmansadr, and Robert Sim. Membership inference attacks against nlp classification models. In *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021.

[182] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6833–6844, 2017.

[183] Emily Sheng, Josh Arnold, Zhou Yu, Kai-Wei Chang, and Nanyun Peng. Revealing persona biases in dialogue systems. *arXiv preprint arXiv:2104.08728*, 2021.

[184] Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, 2019.

[185] Rakshith Shetty, Bernt Schiele, and Mario Fritz. A4nt: Author attribute anonymity by adversarial training of neural machine translation. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1633–1650, Baltimore, MD, August 2018. USENIX Association.

[186] Seungjae Shin, Kyungwoo Song, JoonHo Jang, Hyemi Kim, Weonyoung Joo, and Il-Chul Moon. Neutralizing gender bias in word embedding with latent disentanglement and counterfactual generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 3126–3140, 2020.

[187] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2017.

[188] Reza Shokri. Auditing data privacy for machine learning. Santa Clara, CA, February 2022. USENIX Association.

[189] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[190] Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390, 2020.

[191] Congzheng Song and Vitaly Shmatikov. The natural auditor: How to tell if someone used your words to train their model. *ArXiv*, abs/1811.00513, 2018.

[192] Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 196–206, New York, NY, USA, 2019. Association for Computing Machinery.

[193] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *Proceedings of the 2013 IEEE Global Conference on Signal and Information Processing*, GlobalSIP '13, pages 245–248, Washington, DC, USA, 2013. IEEE Computer Society.

[194] Amber Stubbs and Özlem Uzuner. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of Biomedical Informatics*, 58:S20–S29, 2015. Supplement: Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data.

[195] Michael Tänzer, Sebastian Ruder, and Marek Rei. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[196] Zhiliang Tian, Yingxiu Zhao, Ziyue Huang, Yu-Xiang Wang, Nevin Zhang, and He He. Seqpate: Differentially private text generation via knowledge distillation. 2021.

[197] Kushal Tirumala, Aram H Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. *arXiv preprint arXiv:2205.10770*, 2022.

[198] Thomas Vakili and Hercules Dalianis. Are clinical bert models privacy preserving? the difficulty of extracting patient-condition associations. In *Proceedings of the AAAI 2021 Fall Symposium on Human Partnership with Medical AI : Design, Operationalization, and Ethics (AAAI-HUMAN 2021)*, number 3068 in CEUR Workshop Proceedings, 2021.

[199] Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl;dr: Mining Reddit to learn automatic summarization. In *Proceedings of the ACL 2017 Workshop on New Frontiers in Summarization*, pages 59–63, 2017.

[200] Eric Wallace, Mitchell Stern, and Dawn Xiaodong Song. Imitation attacks and defenses for black-box machine translation systems. In *EMNLP*, 2020.

[201] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2018.

[202] Liwen Wang, Yuanmeng Yan, Keqing He, Yanan Wu, and Weiran Xu. Dynamically disentangling social bias from task-oriented representations with adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3740–3750, Online, June 2021. Association for Computational Linguistics.

[203] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

[204] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 1307–1322, New York, NY, USA, 2017. Association for Computing Machinery.

[205] Yonghui Wu, Mike Schuster, Z. Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason R. Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144, 2016.

[206] Albert Xu, Eshaan Pathak, Eric Wallace, Suchin Gururangan, Maarten Sap, Dan Klein, and UC Berkeley. Detoxifying language models risks marginalizing minority voices.

[207] Qiongkai Xu, Lizhen Qu, Chenchen Xu, and Ran Cui. Privacy-aware text rewriting. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 247–257, 2019.

[208] Kevin Yang and Dan Klein. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online, June 2021. Association for Computational Linguistics.

[209] Yi Yang, Mark Christopher Siy Uy, and Allen Huang. Finbert: A pretrained language model for financial communications. *arXiv preprint arXiv:2006.08097*, 2020.

[210] Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. Unsupervised text style transfer using language models as discriminators. *arXiv preprint arXiv:1805.11749*, 2018.

[211] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[212] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282, 2018.

[213] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[214] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, ICLR '22, 2022.

[215] Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[216] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Large scale private learning via low-rank reparametrization. In *Proceedings of the 38th International Conference on Machine Learning*, ICML '21. JMLR, Inc., 2021.

[217] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR, 2017.

[218] Santiago Zanella-Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. Analyzing information leakage of updates to natural language models. In *ACM SIGSAC Conference on Computer and Communications Security*, 2020.

[219] Santiago Zanella-Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. *Analyzing Information Leakage of Updates to Natural Language Models*, pages 363–375. Association for Computing Machinery, New York, NY, USA, 2020.

[220] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[221] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020.

[222] Xuhui Zhou, Maarten Sap, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. Challenges in automated debiasing for toxic language detection. *arXiv preprint arXiv:2102.00086*, 2021.

[223] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.