Data Synthesis: Boosting Utility and Privacy in Natural Language Processing

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Xiang Yue, B.Eng.

Graduate Program in Computer Science and Engineering

The Ohio State University

2023

Dissertation Committee:

Dr. Huan Sun, Advisor

Dr. Srinivasan Parthasarathy

Dr. Yu Su

Dr. Macarius Donneyong

Abstract

Data is the new oil in today's rapidly developing era of Artificial Intelligence (AI). The large-scale collection of textual data, coupled with powerful computing capabilities, has transformed natural language processing (NLP). For instance, people can create question-answering (QA) systems to help users find precise answers to natural language queries, significantly changing the way people search for information and acquire knowledge. However, collecting, annotating, and processing real-world data can be costly and time-consuming. Additionally, real-world data may contain sensitive personal or confidential information, making it difficult to access due to privacy regulations like the General Data Protection Regulation (GDPR).

In this dissertation, we introduce ***synthetic data as a solution to boost the utility and privacy in NLP with minimal human effort and cost***. Synthetic data is a type of artificially generated data that mimics real-world data. We describe a standard NLP life cycle with three stages: data collection, model training and evaluation, as well as model deployment. We discuss the challenges in each cycle and how synthetic data can solve them. In the data collection stage, we propose generating high-quality synthetic data that preserves privacy using differential privacy (DP) to avoid privacy leaks and address regulatory concerns when collecting sensitive data. In the model training and evaluation stage, we pre-train large language models (LLMs) with synthetic data to address the training objective mismatch and knowledge acquisition gap between pre-training and fine-tuning. Synthetic data will also

be harnessed for improving model evaluation. In the model deployment stage, we adapt fully-trained models to the target domain with selected synthetic data to address the issue of domain shifts between source training data and target testing data.

There are several interesting future directions to explore in leveraging synthetic data to enhance utility and privacy for NLP models. In data collection, the quality and realism of synthetic data can be improved to better emulate real-world data. Recent advances in LLMs for text generation suggest their potential for generating synthetic data. The controllable generation of synthetic data to suit specific tasks or goals is an area for further work. In model training and evaluation, synthetic data augmentation and weighting real-world data with synthetic data during training are approaches to investigate improving utility metrics like robustness. During model deployment, incorporating domain-specific knowledge to make synthetic data more similar to real-world target domain data is another direction to explore. To summarize, synthetic data is a promising area for advancing NLP model utility and privacy through techniques spanning data collection, model development, and deployment.

This is dedicated to my parents and partner for their unconditional love

Acknowledgments

First and foremost, I am extremely grateful to my Ph.D. advisor, Dr. Huan Sun. Without her invaluable advice, continuous support, patience, and dedicated involvement in every step of the process, I would not have been able to complete the research, and I would not have made it through my Ph.D.! Her immense knowledge and plentiful experience have encouraged me in all the time of my Ph.D. research and daily life.

I would also like to show gratitude to my Ph.D. candidacy committee, including Dr. Srinivasan Parthasarathy, Dr. Yu Su, Dr. Ping Zhang, and Dr. Huseyin A. Inan. I met Dr. Srinivasan Parthasarathy in his Database Seminar during my first year at OSU. I still remember the first research topic he brought up in class Data Privacy, which is one of the key topics of this thesis. We discussed the General Data Protection Regulation (GDPR) and how it could impact each stage of the machine learning cycle. All of these discussions attracted my great interest in data privacy and led me to go deeper into this field later. During my Ph.D. study, I have been often discussing my research projects with him, and he gave me many invaluable suggestions and comments. Dr. Yu Su is a faculty member of our OSU NLP group. Though I haven't formally done a research project with him, we had many research discussions in our NLP group meetings and research seminars. He always raised constructive research comments and questions from different angles. These great comments drove me to think deeper and wider. I collaborated with Dr. Ping Zhang on my first Ph.D. research project about Biomedical Graph Embedding Learning. His expertise

in biomedical graphs led this project to a higher quality level. And this paper got ESI Highly Cited Paper later on. Dr. Huseyin A. Inan was my mentor during the 2022 summer internship at Microsoft Research. He was very patient and supportive. Every time I met difficulties, no matter technical wise or administrative wise, he was always lending me a hand and made a great experience for my internship. I would also express my deepest gratitude to the Graduate Faculty Representative, Dr. Macarius Donneyong, for his rigorous participation and unbiased assessment of my dissertation examination.

Getting through my thesis required more than academic support, and I would thank all my friends and colleagues for listening to me, tolerating me, and helping me during my Ph.D. study and life. They have supported me and had to put up with my stresses and moans for the past five years of study. I cannot begin to express my gratitude and appreciation for their friendship.

Lastly, I would be remiss in not mentioning my family, especially my parents, and partner. Their belief in me has kept my spirits and motivation high during this process. This dissertation stands as a testament to your unconditional love and encouragement.

Vita

Aug. 2018 - Present	Ph.D. student / Graduate Research Associate, Department of Computer Science and Engineering, The Ohio State University, USA
May. 2022 - Aug. 2022	Research Intern, Microsoft Research, USA
May. 2021 - Aug. 2021	Research Intern, Tencent AI Lab, USA
Sept. 2014 - June. 2018	B.Eng., School of Computer Science, Wuhan University, China

Publications

Research Publications

* indicates equal contributions

[1] **Yue, X.**, Wang, B., Zhang, K., Chen, Z., Su, Y., & Sun, H. (2023). Automatic evaluation of attribution by large language models. *arXiv preprint arXiv:2305.06311.*

[2] Wang, B., **Yue, X.**, & Sun, H. (2023). Can ChatGPT Defend the Truth? Automatic Dialectical Evaluation Elicits LLMs' Deficiencies in Reasoning. *arXiv preprint arXiv:2305.13160.*

[3] **Yue, X.**, Inan, H. A., Li, X., Kumar, G., McAnallen, J., Sun, H., ... & Sim, R. (2023). Synthetic Text Generation with Differential Privacy: A Simple and Practical Recipe. *In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).*

[4] Du, M.*, **Yue, X.***, Chow, S. S., Wang, T., Huang, C., & Sun, H. (2023). DP-Forward: Fine-tuning and Inference on Language Models with Differential Privacy in Forward Pass. *30th ACM Conference on Computer and Communications Security (CCS).*

[5] Du, M., **Yue, X.**, Chow, S. S., & Sun, H. (2023, April). Sanitizing Sentence Embeddings (and Labels) for Local Differential Privacy. *In Proceedings of the ACM Web Conference 2023 (pp. 2349-2359).*

[6] **Yue, X.**, Yao, Z., & Sun, H. (2022, May). Synthetic Question Value Estimation for Domain Adaptation of Question Answering. *In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1340-1351).*

[7] **Yue, X.**, Pan, X., Yao, W., Yu, D., Yu, D., & Chen, J. (2022, May). C-MORE: Pretraining to Answer Open-Domain Questions by Consulting Millions of References. *In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 371-377).*

[8] Chen, S., Chen, Z., ..., **Yue, X.**, ..., & Sun, H. (2022). Bootstrapping a User-Centered Task-Oriented Dialogue System. *1st Proceedings of Alexa Prize TaskBot.*

[9] Zhang, X. F., Sun, H., **Yue, X.**, Lin, S., & Sun, H. (2021, November). COUGH: A Challenge Dataset and Models for COVID-19 FAQ Retrieval. *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (pp. 3759-3769).*

[10] **Yue, X.***, Du, M.*, Wang, T., Li, Y., Sun, H., & Chow, S. S. (2021, August). Differential Privacy for Text Analytics via Natural Text Sanitization. *In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021 (pp. 3853-3866).*

[11] **Yue, X.***, Zhang, X*. F., Yao, Z., Lin, S., & Sun, H. (2021, December). CliniQG4QA: Generating diverse questions for domain adaptation of clinical question answering. *In 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 580-587).* ***Best Paper Award***

[12] **Yue, X.**, & Zhou, S. (2020, November). PHICON: Improving Generalization of Clinical Text De-identification Models via Data Augmentation. *In Proceedings of the 3rd Clinical Natural Language Processing Workshop (pp. 209-214).*

[13] Mani, K.*, **Yue, X.***, Gutierrez, B. J., Huang, Y., Lin, S., & Sun, H. (2020, December). Clinical Phrase Mining with Language Models. *In 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 1087-1090).*

[14] **Yue, X.**, Gutiérrez, B. J., & Sun, H. (2020, July). Clinical Reading Comprehension: A Thorough Analysis of the emrQA Dataset. *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (pp. 4474-4486).*

[15] **Yue, X.**, Wang, Z., Huang, J., Parthasarathy, S., Moosavinasab, S., Huang, Y., ... & Sun, H. (2020). Graph embedding on biomedical networks: methods, applications and evaluations. *Bioinformatics, 36(4), 1241-1251.* ***(ESI Highly Cited Paper: top 1% cited paper of its academic field)***

[16] Wang, Z., **Yue, X.**, Moosavinasab, S., Huang, Y., Lin, S., & Sun, H. (2019, July). Surfcon: Synonym discovery on privacy-aware clinical data. *In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 1578-1586).*

Fields of Study

Major Field: Computer Science and Engineering

Studies in:

- Artificial Intelligence: Prof. Huan Sun
- Data Mining: Prof. Srinivasan Parthasarathy
- Theory: Prof. Abhishek Gupta

Table of Contents

List of Figures

List of Tables

Part I: Introduction

Chapter 1: Introduction

Data, one of the most valuable resources in the world, is seen as the new oil in the current rapidly developing era of Artificial Intelligence (AI). The vast amount of available textual data, combined with powerful computing capabilities, has led to significant recent advancements in natural language processing (NLP). These advances have enabled the creation of various intelligent systems, such as question-answering and conversational agents, as well as tools for summarization, machine translation, and natural language to code generation.

However, creating NLP models can be a costly and time-consuming process, often requiring the collection, annotation, and processing of large amounts of data. For example, using a crowd-sourcing service like Amazon Mechanical Turk[1] and Scale[2] to annotate 100K samples could cost anywhere from $8,000-$80,000, depending on the complexity of the annotations.

Furthermore, the implementation of strict privacy protection regulations like the General Data Protection Regulation (GDPR)[3], the California Consumer Privacy Act (CCPA)[4], and

[1] https://www.mturk.com/pricing

[2] https://scale.com/pricing

[3] https://gdpr.eu/

[4] https://oag.ca.gov/privacy/ccpa

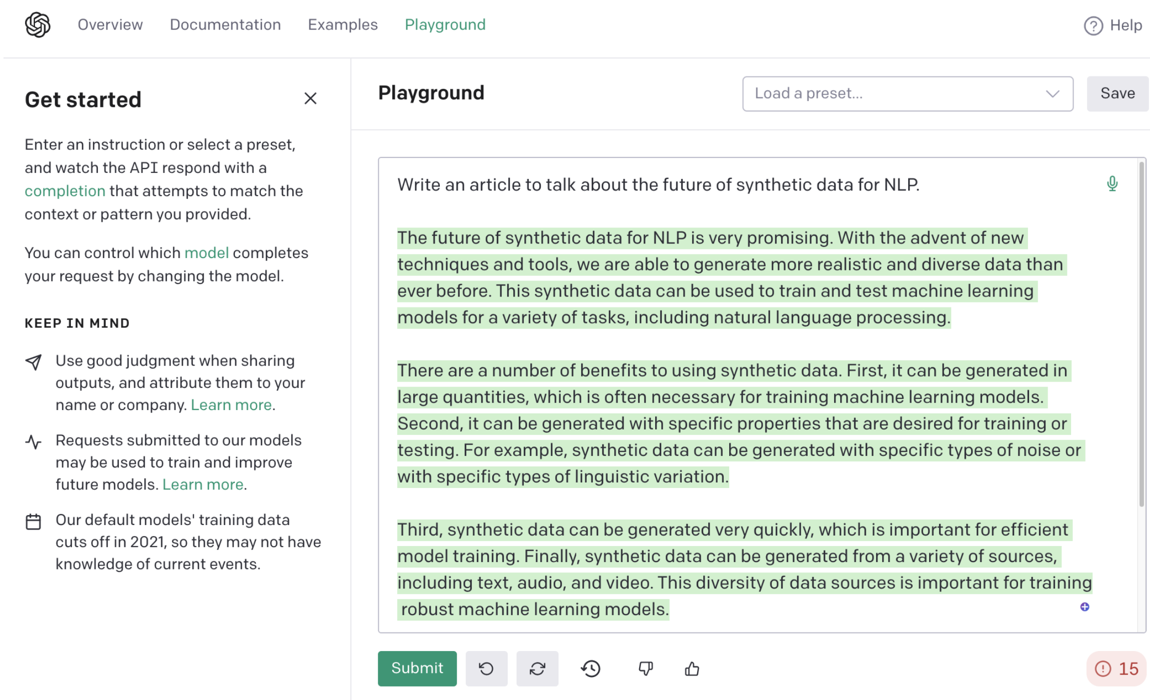


Figure 1.1: A synthetic article generated by the GPT-3 model [24] given the prompt *"Write an article to talk about the future of synthetic data for NLP"*. The screenshot is taken from OpenAI GPT-3 Playground.

the Health Insurance Portability and Accountability Act (HIPAA)[5] makes it difficult and costly to obtain real data for training privacy- and utility-preserving machine learning (ML) models. This adds to the already time-consuming and expensive process of building these models.

All of these constraints have opened up unbounded opportunities for adopting *Synthetic Data*, a type of artificially generated data that reflects statistical properties of real-world data. With the advancement of deep learning (DL), synthetic text generated by deep generative

[5] `https://www.hhs.gov/hipaa/index.html`

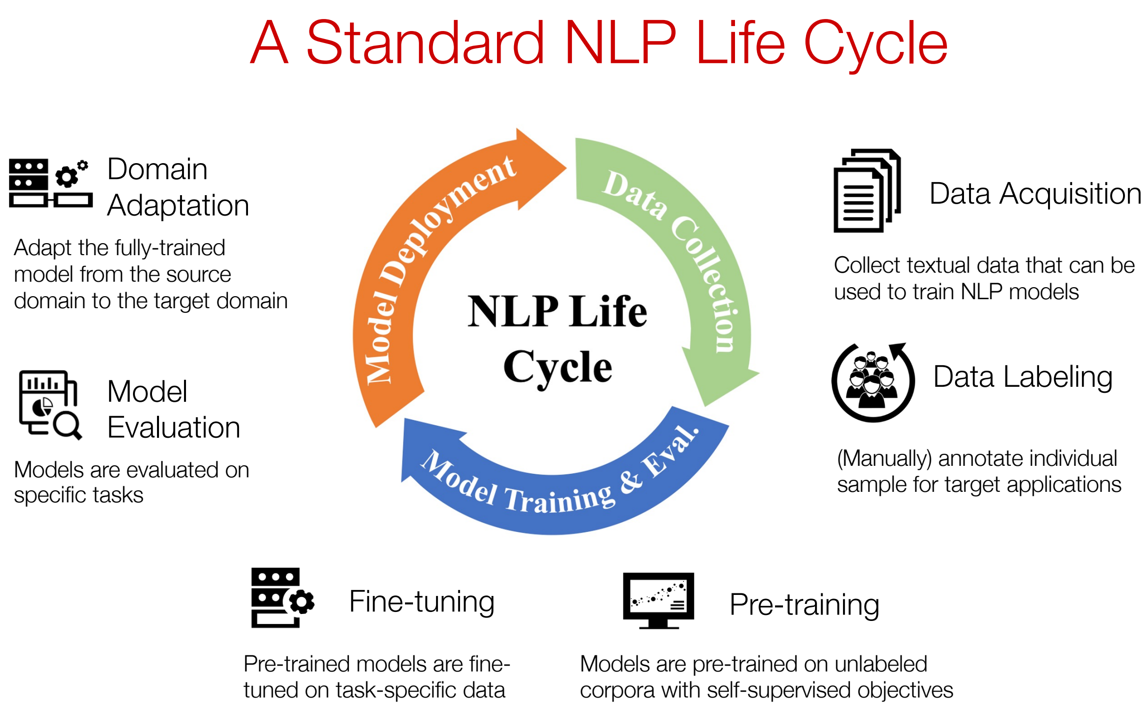
models can closely resemble text written by humans. For example, in Figure 1.1, a large pre-trained language model, GPT-3 [24], was able to generate a natural and logically-coherent paragraph when given the prompt *"Write an article to talk about the future of synthetic data for NLP"*.

This is very promising, as synthetic data generation can be a highly efficient and cost-effective method for constructing large-scale training datasets in a short amount of time. The synthetic data produced is of high quality and can be used to train accurate machine learning models, particularly when labeled training data is scarce. Additionally, incorporating privacy-preserving techniques like Differential Privacy (DP) in the data generation process can help address privacy concerns surrounding sensitive real data. In conclusion, generating synthetic data allows for an improvement in both the utility and privacy of NLP models with minimal human effort and cost.

In this chapter, we will begin by discussing the three main stages of the NLP life cycle: data collection, model training, and model deployment. We will then highlight some of the challenges that can arise in each stage of the NLP life cycle. Next, we will introduce the concept of synthetic data and explain how it can help address these challenges. We will also provide a brief overview of how synthetic textual data is generated. Finally, we will provide an outline of the rest of this thesis proposal and give a brief summary of each chapter.

1.1 A Standard NLP Life Cycle

In this section, we begin our discussion by introducing a standard NLP life cycle (as shown in Figure 1.2), which gives us a high-level view of building an NLP model. It



Data Synthesis: Boosting Utility and Privacy in Natural Language Processing

Stage	Challenges	Our Solutions
Data Collection	Privacy leakage and regulatory concerns arise when collecting data	Generate high-quality **synthetic data** meanwhile mitigating privacy risks *[Chapter 2]* Differential Privacy for Text Analytics via Natural Text Sanitization **[Findings of ACL-IJCNLP 2021]** *[Chapter 3]* Synthetic Text Generation with Differential Privacy: A Simple and Practical Recipe **[ACL 2023]**
Model Training & Evaluation	Pre-training objectives do not align with the downstream fine-tuning ones Lack of task-specific data for evaluation	Generate **synthetic data** for task-specific pre-training and evaluation *[Chapter 4]* C-MORE: Pretraining to Answer Open-Domain Questions by Consulting Millions of References **[ACL 2022]** *[Chapter 5]* Automatic Evaluation of Attribution by Large Language Models **[arXiv 2023; Submitted to EMNLP 2023]**
Model Deployment	Domain shifts exist between (source) training data and (target) testing data	Adapt models to the target domain with (selected) **synthetic data** *[Chapter 6]* CliniQG4QA:Generating diverse questions for domain adaptation of clinical question answering **[BIBM 2021]** *[Chapter 7]* Synthetic Question Value Estimation for Domain Adaptation of Question Answering **[ACL 2022]**

Figure 1.2: A standard NLP life cycle, the challenges across the NLP life cycle, and how synthetic data can help at each stage of the life cycle.

basically classifies the NLP life cycle into three main stages[6]: **Data Collection**, **Model Training**, and **Model Deployment**. The details of each stage are as follows:

Data Collection. The first stage of the NLP life cycle is data collection. As the foundation of modern machine learning, data is crucial for AI applications. Therefore, it is important to collect large amounts of data to build accurate NLP models. For instance, GPT-3 [24], one of the most powerful pre-trained language models, was trained on 45 TB of text data. Another example is CLIP [190], a well-known text-to-image generative model that was trained on 400 million pairs of images and text captions. The use of large amounts of training data significantly improves the performance of these models. Undoubtedly, there is even more need for collecting large-scale high-quality data in this foundation model era [19]. After acquiring data, it is often necessary to annotate the unlabeled data for different downstream tasks. For example, if we aim to build a question-answering (QA) model to answer user's questions, we need first to create a number of QA pairs over the knowledge source, such as text [194], knowledge bases (KB) [228], and images [8]. Data annotation is often costly and time-consuming. It also sometimes needs considerable domain expertise and faces ethical and privacy concerns (which we will discuss more in the next section). Data annotation is often an indispensable step as the number of labeled data has a significant impact on task performance.

Model Training and Evaluation. Once the data has been collected and processed, we can begin training a NLP model on it. In recent years, pre-training followed by fine-tuning has become a common approach for developing NLP applications. In the pre-training

[6]We acknowledge that there should be other (fine-grained) stages inside each major stage. We omit them for simplicity.

phase, neural models such as Transformers [240] are typically trained on large, general datasets like Wikipedia using self-supervised language modeling tasks, such as masked language modeling in BERT [51]. The pre-trained language models are then fine-tuned on task-specific corpora. For example, if we aim to build a QA model to teach the model how to answer a question based on a document, we will fine-tune a pre-trained language model on annotated QA pairs. The pre-training fine-tuning pipeline has become a nearly standard approach since it can well transfer the (linguistic) knowledge learned during the pre-training phase to downstream tasks, and thus significantly improves the performance compared with training models from scratch.

Model Deployment. After training and evaluating the model, we will deploy it for real use. However, it is common for the model's performance to decrease when used in the real world due to domain shifts between the original (source) training and real (target) testing data. This is where domain adaptation comes in. Domain adaptation is a research area focused on improving a model's performance when adapting it to the target domain. Various techniques [195] have been developed to help models achieve better accuracy in the target domain.

The NLP cycle is a process of continuous development, analysis, and improvement. Even after a model is adapted and deployed, one can still go over the three stages again to develop an improved version of the model. In the next section, we will discuss specific challenges in each stage.

1.2 Challenges across the NLP Life Cycle

Although recent years have seen great progress in NLP, there are still various challenges in each stage of the life cycle (Figure 1.2). In this section, we will discuss these challenges in detail.

Data Collection. The growth in available data has made it easier than ever to develop natural language processing (NLP) models. However, this increased accuracy comes at a cost. Large NLP models often have hundreds of millions to billions of parameters, which allows them to memorize most of the data they have seen. This memorization can lead to the leakage of (sensitive) training data and pose significant privacy concerns. For example, recent studies have shown that it is possible to extract verbatim text sequences from pre-trained language models without any knowledge of the model or training set [29, 33, 34, 128, 69, 239, 281]. These sequences can contain sensitive information such as names, addresses, and social media accounts [33]. This presents a practical threat to the safe release of large language models or APIs [34], as attackers can extract individual sequences used to train the model by interacting with them. Addressing these privacy concerns is essential, especially when the training data is sensitive and must be protected.

Model Training and Evaluation. Pre-trained language models have greatly improved the accuracy of various NLP tasks, but there are still gaps between pre-training and downstream task fine-tuning. These gaps can be seen in two ways: the "training objective gap" and the "task knowledge gap". The training objective gap refers to the mismatch between the pre-trained objectives and the task-specific fine-tuning objectives. For example, BERT models [51] use masked language modeling to predict masked tokens given their context, but fine-tuning it for an extractive QA model requires predicting the start and end indices of the answer within a given passage. The task knowledge gap refers to the difference between the general linguistic knowledge learned during pre-training and the task-specific knowledge acquired during the fine-tuning phase. These two gaps can lead to poor performance on downstream tasks, especially when there is not a lot of task data available. Therefore, aligning pre-training objectives and data with the fine-tuning phase is crucial for improving downstream task performance.

Model Deployment After training a model, it must be adapted and deployed for real-world use. In this stage, the testing data used is often different from the training data, which can cause the performance of the well-trained model to decrease significantly. To improve the model's performance on the target data, additional data from the target domain can be collected and annotated, but this process is time-consuming, costly, and may require domain expertise. How to better adapt and deploy the model for real uses with little human effort is the key challenge in this stage.

In the next section, we will introduce synthetic data and discuss how synthetic data could help address these challenges in each stage of the NLP life cycle.

1.3 How Synthetic Data can help at each stage of the NLP Life Cycle?

In the previous section, we discussed the various challenges that arise at each stage of the NLP life cycle. These challenges can take different forms, but they often stem from *a lack of large-scale, inexpensive, high-quality data that also protects privacy*. In this proposal, we propose using synthetic data to address these challenges. Synthetic data is artificially generated based on predefined distributions, rules, or generative models. We will provide an overview of how synthetic data can help at each stage of the NLP life cycle and provide more detailed information in later chapters.

Synthetic Data can address privacy and regulatory concerns in the Data Preparation stage. In the first stage of data preparation, a significant amount of time and effort is often spent on collecting or obtaining access to raw data. A recent Kaggle survey [109] found that getting access to raw data can take up to 50% of the time spent on an AI project. One of the major reasons for this bottleneck is privacy and regulatory concerns. The most recent privacy regulations, such as GDPR and CCPA, have much stricter requirements for privacy and anonymity. These regulations consider data to be anonymous only if no subjects can be re-identified, either directly or indirectly, by the data controller or by any third party. These regulations make data collection difficult, as simply applying pre-processing methods such as de-identification to the original data is not enough to meet the requirements. This has led to the need for generating synthetic data with formal privacy guarantees, such as differential privacy (DP) [59]. By incorporating DP mechanisms into synthetic data generation, it is possible to greatly limit the disclosure of any directly or indirectly linkable private information. As a result, synthetic data can be more freely shared and used for building downstream NLP models with fewer constraints.

Synthetic Data can provide well-aligned pre-training signals in the Model Training and Evaluation stage. In the second stage of model training, models are often pre-trained on plain texts using self-supervised objectives. However, this pre-training phase cannot acquire task-specific knowledge. To address this limitation, task-specific pre-training objectives can be designed and synthetic pre-training data can be created to match the form of downstream tasks. Recent research has shown that using tailored synthetic pre-training data can significantly improve the performance of downstream tasks [159, 263].

Synthetic Data can help models better adapt to the target domain in the Model Deployment Stage. In the third stage of model deployment, we often encounter issues where the data used to train the model is different from the data used to test or deploy it. This is known as domain shift. One way to address this issue is to use synthetic data generation methods to create additional training data for the target domain. This can help the model adapt to new environments without the need for costly re-annotation of real data.

In the next section, we will briefly introduce how to generate synthetic textual data in general.

1.4 How to Generate Synthetic Data?

The use of synthetic data for various applications has a long history, dating back to the 1930s with the development of audio and speech synthesis. One notable contribution to the field of synthetic data generation was Donald B. Rubin's 1993 proposal [204] for using synthetic data to address issues such as undercounting in censuses. Since then, various approaches to data synthesis have been developed, which can be broadly grouped into two categories: rule-based data generation and deep generative models.[7]

Rule-based Data Generation. People design specific rules to synthesize data for different applications. The complexity of the rules used to synthesize data for different applications can vary from simple to complex. The amount of human labor and expertise required, as

[7]It is worth noting that other methods of data generation, such as random generation from pre-defined distributions, exist but are typically not used for generating informative synthetic data for training machine learning models. Instead, they are often used for other purposes, such as stress testing. We skip the discussion of these approaches in this thesis.

well as the information contained in the synthetic data, is determined by the specific rules that are designed for the task at hand. As such, the level of complexity of the rules and human efforts can have a significant impact on the overall effectiveness of the synthetic data.

Deep Generative Models. In recent years, due to the rapid development of deep learning, people explore to train deep generative models on real data so that it can be used to generate new synthetic data that have similar distributions to the original one. Representative models include Generative adversarial networks (GANs) [82], Variational autoencoders (VAEs) [116], and Autoregressive Language Models. GANs consist of two sub-models: a generator and a discriminator. The generator is to synthesize the fake data, and the discriminator is to differentiate the generated fake data from the real data. The process continues until the generator can synthesize data items that the discriminator is not able to differentiate from the real data input. VAEs aim to reconstruct original data points by first transforming them into latent distributions, and back into the original space. Autoregressive language models learn the probability over sequences of words. They then create data by predicting the probability distribution of the next token given the previous context.

Using synthetic data generation methods in practice faces several challenges. The first challenge is *quality*. To train accurate NLP models, it is essential to obtain high-quality synthetic data. Despite the impressive progress made by deep generative models, synthetic data can still sometimes contain a lot of noise, such as factual hallucinations [151, 101] and repetitions [74]. Chapter 7 presents a method for filtering out noisy synthetic data and selecting high-quality samples for training accurate models. The second challenge is *diversity*. Lack of diversity is a common issue with synthetic data. Studies have shown that diverse training data can lead to more robust models [169]. Chapter 4 and 6 present approaches for synthesizing diverse data for training QA models. The third challenge is *privacy*. Synthetic data does not inherently provide privacy guarantees [106]. Generating privacy-preserving data is particularly important for sensitive applications such as medical or financial applications. Chapter 2 and Chapter 3 discuss how to protect privacy during synthetic text generation using DP.

1.5 Organization of this Thesis

Thesis Statement. In this thesis, we aim to ***generate and leverage synthetic data to boost utility and privacy in NLP with minimal human effort and cost***.

The rest of this thesis is organized as follows.

In Part II, we talk about how to generate synthetic data with differential privacy to protect training data privacy. We first propose a token-by-token text sanitization approach (Chapter 2) to replace the original (sensitive) tokens with similar tokens based on an exponential distribution that satisfies the notion of DP. We then propose a stricter privacy requirement for the synthetic text: we require that the *whole synthetic dataset* should be differentially private, i.e., any data instance should be statistically indistinguishable in the synthetic dataset. To achieve this, in Chapter 3, we propose to fine-tune a large language model with DP and then use it to generate synthetic text with carefully-designed natural prompts.

In Part III, we will discuss how synthetic data can help pre-training and evaluation. Specifically, in Chapter 4, we study open-domain question answering, one of the most important tasks in NLP. We first synthesize millions of pseudo question-answer-document triplets from Wikipedia articles and their references using some rule-based generation methods. We then further illustrate how to utilize these synthetic data to pre-train an open-domain QA system and improve both its retrieval and reading performance. In Chapter 5, we study the evaluation of attribution. We propose to train and evaluate an attribution score model with synthetic data from QA, fact-checking, NLI and summarization.

Finally, in Part IV, we will talk about how to better adapt and deploy fully-trained models to new environments in the wild with the help of synthetic data. Specifically, we study Machine Reading Comprehension (MRC) as the application case. We first propose a neural question generation method to synthesize diverse QA pairs on the target documents to help improve the performance of clinical QA models (Chapter 6). Due to the noisy nature of the synthetic data (e.g., the synthetic questions sometimes do not match with their answers), we further propose a question value estimator (QVE) to help us select the most useful synthetic questions that can improve the QA model's performance on the target domain (Chapter 7).

Part II: Data Synthesis with Differential Privacy

As concerns about data privacy continue to grow, privacy regulations have been implemented to provide extra protection. For example, the HIPAA includes a list of sensitive attributes, such as names, ages, and addresses, that must be removed or anonymized before a patient's record can be shared or used. As a result, various de-identification methods, including both rule-based and machine learning-based approaches, have been developed to identify this protected health information (PHI) or personally identifiable information (PII). Despite being widely adopted, these strategies are still far from perfect. An intelligent adversary can mine unique characteristics from seemingly harmless fields in a released data set and potentially link it to other "auxiliary knowledge" for re-identification or single-out attacks. Such attacks are common [163, 227, 220]. For example, Sweeney [227] was able to identify patients from anonymized health records released by Washington State using newspaper stories. However, de-identification methods often have false negatives, and the increasing number of sensitive attributes makes the de-identification process less efficient and accurate.

To solve these challenges, in this part, we propose to generate synthetic data with the formal guarantee of differential privacy. Firstly, in Chapter 2, we introduce a token-by-token text sanitization approach. It replaces each token in the sentence with a similar token randomly sampled from an exponential distribution that satisfies DP. In Chapter 3, we propose to fine-tune language models with differential privacy and use them to generate synthetic text.

Chapter 2: Differential Privacy for Text Analytics via Natural Text Sanitization

Text, a convenient medium conveying sophisticated knowledge, and hence a treasure for various data analytics tasks, often contains sensitive information that requires sanitization. Despite the success of differential privacy (DP) mechanisms in many domains, there remains a major unresolved challenge for differentially private text sanitization. In this chapter, we propose token-level DP mechanisms that sanitize an input text by replacing each token with a similar candidate sampled from an exponential distribution. The sanitized texts also contribute to our sanitization-aware pre-training and fine-tuning, enabling privacy-preserving natural language processing over the BERT language model with promising utility. Surprisingly, the high utility does not boost up the success rate of inference attacks.

2.1 Introduction

Natural language processing (NLP) requires a lot of training data, which can be sensitive. Naïve redaction approaches (*e.g.*, removing common personally identifiable information) is known to fail [227]: innocuous-looking fields can be linked to other information sources for reidentification. The recent success of many language models (LMs) has motivated security researchers to devise advanced privacy attacks. [31] recover texts from (a single document

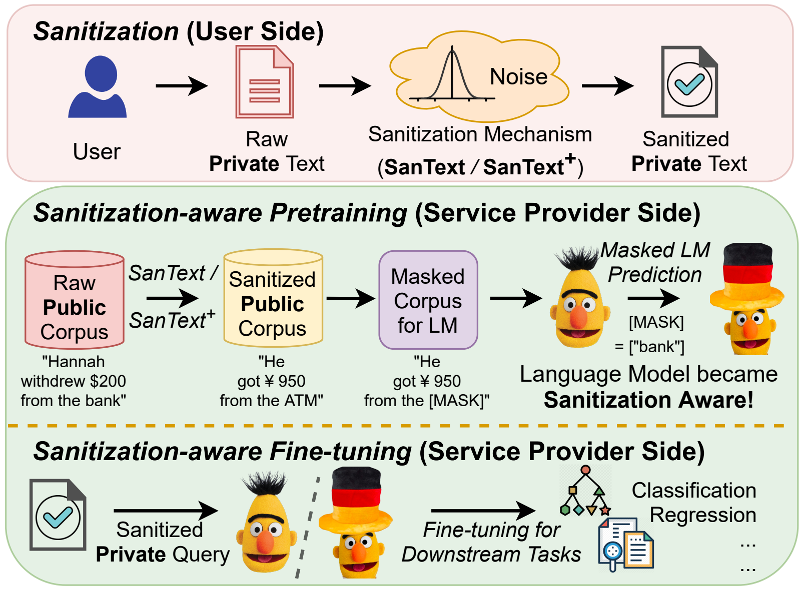


Figure 2.1: Workflow of our PPNLP pipeline, including the user-side sanitization and the service provider-side NLP modeling with pretraining/fine-tuning

of) the *training data* via querying to an LM pretrained from it. [175] and [219] target the text embedding, *e.g.*, revealing from an encoded *query* to an NLP service.

Emerging NLP works focus on only specific document-level (statistical) features [250] or producing private text representations [268, 47, 63, 138] as initial solutions to the first issue above on training-data privacy. However, the learned representations are *not* human-readable, which makes *transparency* (*e.g.*, required by GDPR) questionable: an average user may not have the technical know-how to verify whether sensitive attributes have been removed or not. Moreover, consider the whole NLP pipeline, the learned representations often entail extra modeling or non-trivial changes to existing NLP models, which take dedicated engineering efforts.

2.1.1 Sanitizing Sensitive Texts, Naturally

With this state-of-affairs of the security and the NLP research, we deem it better to address privacy from the root, *i.e.*, directly producing sanitized text documents. Being the most native format, they incur minimal changes to existing NLP pipelines. Being human-readable, they provide transparency (to privacy-concerning training-data contributors) and explainability (*e.g.*, to linguists who might find the need for investigating how the training data contribute to a certain result). Moreover, it naturally extends privacy protection to the inference phase. Users can apply our sanitization mechanism before sending queries (*e.g.*, medical history) to the NLP service provider (*e.g.*, diagnosis services).

Conceptually, we take a natural approach – we sanitize text documents into also (sanitized) text documents. This is in great contrast to the typical "post-processing" for injecting noises either to gradients in training (a deep neural network) [153] or the "cursed" *high-dimensional* text representations [148, 149, 71]. It also leads to our $O(1)$ efficiency, freeing us from re-synthesizing the document word-by-word via nearest neighbor searches over the entire vocabulary space $\mathcal{V}$ [71].

Technically, we aim for the *de facto* standard of local differential privacy (LDP) [56] to sanitize the user data *locally*, based on which the service provider can build NLP models without touching any raw data. DP has been successful in many contexts, *e.g.*, location privacy and survey statistics [7, 160]. However, DP text analytics appears to be a difficult pursuit (as discussed, also see Section 2.2), which probably explains why there are only a few works in DP-based text sanitization. In high-level terms, text is rich in semantics, differentiating it from other more structured data.

Our challenge here is to develop *efficient* and effective mechanisms that *preserve the utility* of the text data with *provable and quantifiable* privacy guarantees. Our insight is the formulation of a new LDP notion named *Utility-optimized Metric LDP (UMLDP)*. We attribute our success to the focus of UMLDP on protecting what matters (sensitive words) via "sacrificing" the privacy of non-sensitive (common) words. To achieve UMLDP, our mechanism directly samples noises on tokens.

Our result in this regard is already better than the state-of-the-art LDP solution producing sanitized documents [71] – we got 28% gain in accuracy on the SST-2 dataset [244] on average at the same privacy level (*i.e.*, the same LDP parameter) while being much more efficient ($\sim 60\times$ faster, precomputation included).

2.1.2 Privacy-Preserving NLP, Holistically

Text sanitization is essential but just one piece of the whole privacy-preserving NLP (PPNLP) pipeline. While most prior works in text privacy are motivated by producing useful data for some downstream tasks, the actual text analytics are hardly explored, not to say in the context of many recent general-purpose language models. As simple as it might seem, we start to see design choices that can be influential. Specifically, our challenge here is to adapt the currently dominating pretraining-fine-tuning paradigm (*e.g.*, BERT [51]) over sanitized texts for building the model.

Our design is to build in privacy at the root again, in contrast to the afterthought approach. We found it beneficial to sanitize even the public data before feeding them to training. It is not for protecting the public data per se. The intuition here is that it "prepares" the model to work with sanitized queries, which explains our eventual (slight) increase in accuracy while *additionally ensuring privacy*.

Specifically, we propose a sanitization-aware pretraining procedure (Figure 2.1). We first use our mechanisms to sanitize the public texts, mask the sanitized texts (as in BERT), and train the LM by predicting a MASK position as its *original unsanitized token*. LMs preptrained with our sanitization-aware procedure are expected to be more robust to noises in the sanitized texts and achieve better utility when fine-tuning on downstream tasks.

We conduct experiments on three representative NLP tasks to empirically confirm that our proposed PPNLP pipeline preserves both utility and privacy. It turns out that our sanitization-based pretraining (using only $1/6$ of data used in the original BERT pretraining) can even improve the utility of NLP tasks while maintaining privacy comparable to the original BERT. Note that there is an inherent tension between utility and privacy, and privacy attack is also inference in nature. To empirically demonstrate the privacy aspect of our pipeline, *i.e.*, it does not make our model a more powerful tool helping the attacker, we also conduct the "mask token inference" attack on private texts, which infers the masked token given its context based on BERT. As a highlight, our base solution SANTEXT improves the defense rate by 20% with only a 4% utility loss on the SST-2 dataset. We attribute our surprising result of mostly helping only good guys to our natural approach: to avoid the model memorizing sensitive texts "too well," we fed it with sanitized text.

2.2 Related Work

Privacy risks in NLP. A taxonomy of attacks that recover sensitive attributes or partial raw text from text embeddings output by popular LMs has been proposed [219], without any assumptions on the structures or patterns in input text. Carlini et al. [31] also show a powerful black-box attack on GPT-2 [189] that extracts verbatim texts of training data. Defense with rigorous guarantees (DP) is thus vital.

Differential privacy and its application in NLP. DP [57] has emerged as the *de facto* standard for statistical analytics [247, 248, 48]. A few efforts inject high-dimensional

DP noise into text representations [70, 71, 148, 149]. The noisy representations are not human-readable and not directly usable by existing NLP pipelines, *i.e.*, they consider a different problem not directly comparable to ours. More importantly, they fail to strike a nice privacy-utility balance due to "the curse of dimensionality," *i.e.*, the magnitude of the noise is too large for high-dimensional token embedding, and thus it becomes exponentially less likely to find a noisy embedding close to a real one on every dimension. This may also explain why an earlier work focuses on document-level statistics only, *e.g.*, term-frequency vectors [250].

Our approaches produce natively usable sanitized texts via directly sampling a substitution for each token from a precomputed distribution (to be detailed in Section 2.4), circumventing the dimension curse and striking a privacy-utility tradeoff while being much more efficient. A concurrent work [187] also considers the whole NLP pipeline, but it still builds on the token-projection approach [71].

Privacy-preserving text representations. Learning private text representations via adversarial training is also an active area [268, 47, 63, 138]. An adversary is trained to infer sensitive information jointly with the main model, while the main model is trained to maximize the adversary's loss and minimize the primary learning objective. While we share the same general goal, our aim is not such representations (similar to those with DP) but to release sanitized text for general purposes.

2.3 Defining (Local) Differential Privacy

Suppose each user holds a document $D = \langle x_i \rangle_{i=1}^{L}$ of L tokens (which can be a character, a subword, a word, or an n-gram), where x_i is from a vocabulary $\mathcal{V}$ of size $|\mathcal{V}|$. For privacy, each user derives a sanitized version $\hat{D}$ by running a common text sanitization mechanism

$\mathcal{M}$ over D on local devices. Specifically, $\mathcal{M}$ works by replacing every token x_i in D with a substitution $y_i \in \mathcal{V}$, assuming that x_i itself is unnecessary for NLP tasks while its semantics should be preserved for high utility. The output $\hat{D}$ is then shared with an NLP service provider.

We consider a typical threat model in which each user does not trust any other party and views them as an attacker with access to $\hat{D}$ in conjunction with any auxiliary information (including $\mathcal{M}$).

2.3.1 (Variants of) Local Differential Privacy

Let $\mathcal{X}$ and $\mathcal{Y}$ be the input and output spaces. A randomized mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is a probabilistic function that assigns a random output $y \in \mathcal{Y}$ to an input $x \in \mathcal{X}$. Every y induces a probability distribution on the underlying space. For sanitizing text, we set both $\mathcal{X}$ and $\mathcal{Y}$ as the vocabulary $\mathcal{V}$.

Definition 2.3.1 (ϵ-LDP [56])**.** Given a privacy parameter $\epsilon \geq 0$, $\mathcal{M}$ satisfies ϵ-local differential privacy (ϵ-LDP) if, for any $x, x', y \in \mathcal{V}$,

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(x') = y].$$

Given an observed output y, from the attacker's view, the likelihoods y is derived from x and x' are similar. A smaller ϵ means better privacy due to a higher indistinguishability level of output distributions, yet the outputs retain less utility.

ϵ-LDP is a very strong privacy notion for its homogeneous protection over all input pairs. However, this is also detrimental to the utility: no matter how unrelated x and x' are, their output distributions must be similar. As a result, a sanitized token y may not (approximately) capture the semantics of its input x, degrading the downstream tasks.

LDP over metric spaces. To capture semantics, we borrow the relaxed notion of Metric-LDP (MLDP) [5] originally proposed for location privacy [7] with the distance metric $d(\cdot, \cdot)$ between two locations (*e.g.*, Manhattan distance [36]).

Definition 2.3.2 (MLDP)**.** Given $\epsilon \geq 0$ and a distance metric $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$ over $\mathcal{V}$, $\mathcal{M}$ satisfies MLDP or $\epsilon \cdot d(x, x')$-LDP if, for any $x, x', y \in \mathcal{V}$,

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon \cdot d(x,x')} \cdot \Pr[\mathcal{M}(x') = y].$$

When $d(x, x') = 1 \, \forall x \neq x'$, MLDP becomes LDP. For MLDP, the indistinguishability of output distributions is further scaled by the distance between the respective inputs. Roughly, the effect of ϵ becomes "adaptive." To apply MLDP, one needs to carefully define the metric d (see Section 2.4.2).

Incorporating ULDP to further improve utility. Utility-optimized LDP [160] (ULDP) also relaxes LDP, which was originally proposed for aggregating ordinal responses. It exploits the fact that different inputs have different sensitivity levels to achieve higher utility. By assuming that the input space is split into *sensitive* and *non-sensitive* parts, ULDP achieves a privacy guarantee equivalent to LDP for *sensitive* inputs.

In our context, more formally speaking, let $\mathcal{V}_S \subseteq \mathcal{V}$ be the set of sensitive tokens common to all users, and $\mathcal{V}_N = \mathcal{V} \setminus \mathcal{V}_S$ be the set of remaining tokens. The output space $\mathcal{V}$ is split into the *protected* part $\mathcal{V}_P \subseteq \mathcal{V}$ and the *unprotected* part $\mathcal{V}_U = \mathcal{V} \setminus \mathcal{V}_P$.

The image of $\mathcal{V}_S$ is restricted to $\mathcal{V}_P$, *i.e.*, a sensitive $x \in \mathcal{V}_S$ can only be mapped to a protected $y \in \mathcal{V}_P$. For text, we can set $\mathcal{V}_S = \mathcal{V}_P$ for simplicity. While a non-sensitive $x \in V_N$ can be mapped to $\mathcal{V}_P$, every $y \in \mathcal{V}_U$ must be mapped from $\mathcal{V}_N$, which helps to improve the utility.

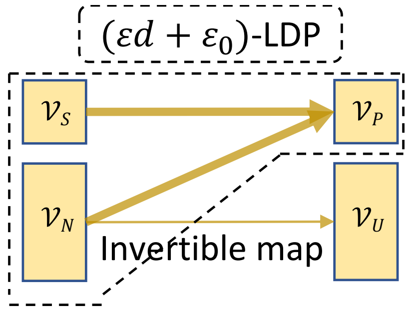


Figure 2.2: Overview of our new UMLDP notion

2.3.2 Our New Utility-optimized MLDP Notion

Among many variants of (L)DP notions, we found the above two variants (*i.e.*, ULDP and MLDP) provide useful insight in quantifying semantics and privacy of text data. We thus formulate the new privacy notion of utility-optimized MLDP (UMLDP).

Definition 2.3.3 (UMLDP)**.** Given $\mathcal{V}_S \cup \mathcal{V}_N = \mathcal{V}$, two privacy parameters $\epsilon, \epsilon_0 \geq 0$, and a distance metric $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$, $\mathcal{M}$ satisfies $(\mathcal{V}_S, \mathcal{V}_P, \epsilon, \epsilon_0)$-UMLDP, if

i) for any $x, x' \in \mathcal{V}$ and any $y \in \mathcal{V}_P$, we have

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon d(x,x')+\epsilon_0} \Pr[\mathcal{M}(x') = y];$$

ii) for any $y \in \mathcal{V}_U$, *i.e.*, from an unprotected set $\mathcal{V}_U$ where $\mathcal{V}_U \cap \mathcal{V}_P = \emptyset$, there is an $x \in \mathcal{V}_N$ such that

$$\Pr[\mathcal{M}(x) = y] > 0,$$
$$\Pr[\mathcal{M}(x') = y] = 0 \; \forall x' \in \mathcal{V} \setminus \{x\}.$$

Figure 2.2 summarizes the treatment of UMLDP. It exhibits "invertibility," *i.e.*, $y \in \mathcal{V}_U$ must be "noise-free" and mapped deterministically. Apart from generalizing ϵ in the ULDP definition (recalled in Appendix A.1.1) into $\epsilon d(x, x')$, we incorporate an additive bound ϵ_0 due to the invertibility, which makes the derivation of ϵ easier. Looking ahead, ϵ_0 would appear naturally in the analysis of our UMLDP mechanism for the invertible case.

UMLDP (and MLDP), as an LDP notion, satisfies the *composability* and *free post-processing*. The former means that the sequential execution of ϵ_1-LDP and ϵ_2-LDP mechanisms satisfies $(\epsilon_1 + \epsilon_2)$-LDP, *i.e.*, ϵ can be viewed as the privacy "budget" of a sophisticated task comprising multiple subroutines, each consumes a part of ϵ such that their sum equals ϵ. The latter means further processing the mechanism outputs incurs no extra privacy loss.

2.4 Our Privacy-Preserving NLP Pipeline

2.4.1 Overview

We propose two token-wise sanitization methods with (U)MLDP: SANTEXT and SANTEXT$^+$, which build atop a variant of the exponential mechanism (EM) [154] over the "native" text tokens as both input and output spaces to avoid going to the "cursed dimensions" of token embeddings. EM samples a replacement y for an input x based on an exponential distribution, with more "suitable" y's sampled with higher probability (detailed below). It is well-suited for (U)MLDP by considering the "suitability" as how well the semantics of x is preserved for the downstream tasks (run over the sanitized text y) to remain accurate.

To quantify this, we utilize an embedding model mapping tokens into a real-valued vector space. The semantic similarity among tokens can then be measured via the Euclidean distance between their corresponding vectors. Our base design SANTEXT outputs y with probability inverse proportional to the distance between x and y: the shorter the distance, the more semantically similar they are. SANTEXT$^+$ considers some tokens $\mathcal{V}_N$ in $\mathcal{V}$ are non-sensitive, and runs SANTEXT over the sensitive part $\mathcal{V}_S = V \setminus \mathcal{V}_N$ (*i.e.*, it degenerates

Algorithm 1 Base Mechanism SANTEXT

Input: A private document $D = \langle x_i \rangle_{i=1}^{L}$, and a privacy parameter $\epsilon \geq 0$
Output: Sanitized document $\hat{D}$.

1: Derive token vectors $\phi(x_i)$ for $i \in [1, L]$
2: **for** $i = 1, \ldots, L$ **do**
3: Run $\mathcal{M}(x_i)$ to sample a sanitized token y_i with probability defined in Eq. (2.1)
4: **end for**Output sanitized $\hat{D}$ as $\langle y_i \rangle_{i=1}^{L}$

to SANTEXT if $\mathcal{V}_S = \mathcal{V}$). For $\mathcal{V}_N$, we tailor a probability distribution to provide UMLDP as a whole.

With SANTEXT or SANTEXT^{+}, each user sanitizes D into $\hat{D}$ and uploads it to the service provider for performing any NLP task built atop a pretrained LM, *e.g.*, BERT. Typically, the task pipeline consists of an embedding layer, an encoder module, and task-specific layers, *e.g.*, for classification.

Without the raw text, the utility can degrade; we thus propose two approaches for improving it. The first one is to pretrain only the encoder on the sanitized public corpus to adapt to the noise. It is optional if pretraining is deemed costly. The second is to fine-tune the full pipeline on $\hat{D}$'s, which updates both the encoder and task layers.

2.4.2 Base Sanitization Mechanism: SANTEXT

In NLP, a common step is to employ an embedding model[8] mapping semantically similar tokens to close vectors in a Euclidean space. Concretely, an embedding model is an injective mapping $\phi : \mathcal{V} \to \mathbb{R}^m$, for dimensionality m. The distance between any two tokens x and x' can be measured by the Euclidean distance of their embeddings: $d(x, x') = d_{\mathsf{euc}}(\phi(x), \phi(x'))$. As ϕ is injective, d satisfies the axioms of a distance metric.

[8]We assume that it has been trained on a large public corpus and shared by all users.

Algorithm 2 Enhanced SANTEXT^{+}

Input: A private document $D = \langle x_i \rangle_{i=1}^{L}$, a privacy parameter $\epsilon \geq 0$, probability p for a biased coin, and sensitive $\mathcal{V}_S$

Output: Sanitized document $\hat{D}$.

1: **if** $x_i \in \mathcal{V}_S$ **then**
2: $\quad$ ▷ *Run* SANTEXT *over* $\mathcal{V}_S$ *and* $\mathcal{V}_P$
3: $\quad$ Sample a substitution $y_i \in \mathcal{V}_P = \mathcal{V}_S$ with probability given in Eq. (2.1)
4: **else**
5: $\quad$ Output $y_i = x_i$ with prob. $(1-p)$; or $y_i \in \mathcal{V}_P$ with prob. in Eq. (2.2);
6: **end if**

Algorithm 1 lists the pseudo-code of SANTEXT for sanitizing a private document D at the user side. The first step is to use ϕ to derive token embeddings of each token[9] x in D. Then, for each x, we run $\mathcal{M}(x)$ to sample a sanitized y with probability

$$\Pr[\mathcal{M}(x) = y] = C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x), \phi(y))} \tag{2.1}$$

where $C_x = (\sum_{y' \in \mathcal{V}} e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x), \phi(y'))})^{-1}$.

The smaller $d_{\mathsf{euc}}(\phi(x), \phi(y))$, the more likely y is to replace x. To boost the sanitizing efficiency, we can precompute a $|\mathcal{V}| \times |\mathcal{V}|$ probability matrix, where each entry (i, j) denotes the probability of outputting y_j on input x_i, upon obtaining $\phi(x)$ for $\forall x \in \mathcal{V}$. Lastly, the sanitized $\hat{D} = \langle y_i \rangle_{i=1}^{L}$ can be released to the service provider for NLP tasks.

2.4.3 Enhanced Mechanism: SANTEXT^{+}

In SANTEXT, all tokens in $\mathcal{V}$ are treated as sensitive, which leads to excessive protection and utility loss. Following the less-is-more principle, we divide $\mathcal{V}$ into $\mathcal{V}_S$ and $\mathcal{V}_N$, and focus on protecting $\mathcal{V}_S$.

Observing that most frequently used tokens (*e.g.*, a/an/the) are non-sensitive to virtually all users, we use token frequencies for division. A simple strategy, which is also used in our experiments, is to mark the top w of low-frequency tokens (according to a certain corpus) as

[9] For easy presentation, we omit the subscript i later.

$\mathcal{V}_S$, where w is a tunable parameter. Looking ahead, this "basic" method already showed promising results. (Further discussion can be found in Section 2.4.5).

Algorithm 2 lists the pseudo-code of SANTEXT$^+$ with $\mathcal{V}_S = \mathcal{V}_P$ and $\mathcal{V}_N = \mathcal{V}_U$ shared by all users. The first step, as in SANTEXT, is to derive the token embeddings in D. Then, for each token x, if it is in $\mathcal{V}_S$, we sample its substitution y from $\mathcal{V}_P$ with probability given in Eq. (2.1). (This is equivalent to running SANTEXT over $\mathcal{V}_S$ and $\mathcal{V}_P$.) For $x \in \mathcal{V}_N$, we toss a biased coin. With probability $(1-p)$, we output y as x (*i.e.*, the "invertibility"). Otherwise, we sample $y \in \mathcal{V}_P$ with probability

$$\Pr[\mathcal{M}(x) = y] = p \cdot C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x),\phi(y))} \tag{2.2}$$

where $C_x = (\sum_{y' \in \mathcal{V}_P} e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x),\phi(y'))})^{-1}$.

As in SANTEXT, we can also precompute two $|\mathcal{V}_S| \times |\mathcal{V}_P|$ and $|\mathcal{V}_N| \times |\mathcal{V}_P|$ probability matrices, which correspond to Eq. (2.1) and (2.2), for optimizing the sanitizing efficiency. Lastly, the sanitized $\hat{D}$ of $\langle y \rangle_{i=1}^{L}$ can be released to the service provider.

Theorem 1. Given $\epsilon \geq 0$ and d_{euc} over the embedding space ϕ of $\mathcal{V}$, SANTEXT satisfies MLDP.

Theorem 2. Given $(\mathcal{V}_S = \mathcal{V}_P) \subseteq \mathcal{V}$, $\epsilon \geq 0$, $\epsilon_0 = \ln \frac{1}{p} \geq 0$, and d_{euc} over the embedding space ϕ of $\mathcal{V}$, SANTEXT$^+$ satisfies $(\mathcal{V}_S, \mathcal{V}_P, \epsilon, \epsilon_0)$-UMLDP.

Proof of Theorem 1. Consider $L = 1$, *i.e.*, $D = \langle x \rangle$. For another document D' with $x' \in \mathcal{V} \setminus \{x\}$ and a possible output $y \in \mathcal{V}$:

$$\begin{aligned}
&\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} \\
=&\frac{C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x),\phi(y))}}{C_{x'} \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x'),\phi(y))}} \\
=&\frac{C_x}{C_{x'}} \cdot e^{\frac{1}{2}\epsilon \cdot [d(x',y) - d(x,y)]} \\
\leq&\frac{C_x}{C_{x'}} \cdot e^{\frac{1}{2}\epsilon \cdot d(x,x')} \\
=&\frac{\sum_{y' \in \mathcal{V}} e^{-\frac{1}{2}\epsilon \cdot d(x',y')}}{\sum_{y' \in \mathcal{V}} e^{-\frac{1}{2}\epsilon \cdot d(x,y')}} \cdot e^{\frac{1}{2}\epsilon \cdot d(x,x')} \\
\leq&e^{\frac{1}{2}\epsilon \cdot d(x,x')} \cdot e^{\frac{1}{2}\epsilon \cdot d(x,x')} = e^{\epsilon \cdot d(x,x')}
\end{aligned}$$

The proof, showing SANTEXT ensures $\epsilon \cdot d(x, x')$-LDP, mainly relies on the triangle inequality of d. To generalize to the case of $L > 1$, we sanitize every token x_i in D independently, and thus:

$$\Pr[\mathcal{M}(D) = \hat{D}] = \prod_{i=1}^{L} \Pr[\mathcal{M}(x_i) = y_i].$$

Then, for any D, D', the privacy bound is given as

$$\frac{\Pr[\mathcal{M}(D) = \hat{D}]}{\Pr[\mathcal{M}(D') = \hat{D}]} \leq e^{\epsilon \cdot \sum_{i=1}^{L} d(x_i, x_i')},$$

which follows from the composability. □

Proof of Theorem 2. Consider the case $L = 1$ with $D = x$ and $D' = x'$. For $x, x' \in \mathcal{V}_S$, the output y is restricted to $\mathcal{V}_P$, with the proof identical to the above theorem (as SANTEXT is run over $\mathcal{V}_S, \mathcal{V}_P$).

For $x, x' \in \mathcal{V}_N$ and $y \in \mathcal{V}_P$, we have

$$\begin{aligned}
\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} &= \frac{p \cdot C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x),\phi(y))}}{p \cdot C_{x'} \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x'),\phi(y))}} \\
&\leq e^{\epsilon \cdot d(x,x')}.
\end{aligned}$$

For $x \in \mathcal{V}_S$, $x' \in \mathcal{V}_N$, and $y \in \mathcal{V}_P$, we have

$$\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} = \frac{C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x), \phi(y))}}{p \cdot C_{x'} \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x'), \phi(y))}}$$
$$\leq \frac{1}{p} \cdot e^{\epsilon \cdot d(x, x')} = e^{\epsilon \cdot d(x, x') + \epsilon_0}.$$

The probability for $x \in \mathcal{V}_N$ is $(1 - p)$. The above inequalities thus show that SanText$^+$ ensures the properties of UMLDP. Similarly, we use the composability to generalize for $L > 1$. □

2.4.4 NLP over Sanitized Text

With $\hat{D}$'s (shared by the users), the service provider can perform any NLP task. In this work, we focus on those built on a pretrained LM, and in particular, we study BERT as an example due to its wide adoption and superior performance. The full NLP pipeline is deployed at the service provider.

Given a piece of (sanitized) text, the embedding layer maps it to a sequence of token embeddings. The encoder computes a sequence representation from the token embeddings, allowing task-specific layers to make predictions. For example, the task layer could be a feed-forward neural network for multi-label classification of a diagnosis system.

The injected noise deteriorates the performance of downstream tasks as the service provider cannot access the raw texts $\{D\}$. To mitigate this, we propose two approaches – pretraining the encoder and fine-tuning the full pipeline, which allow the tasks to be "adaptive" to the noise to some extent.

Pretraining BERT over sanitized public corpus. Besides $\hat{D}$'s, the service provider can also obtain a massive amount of text that is publicly available (say, the English Wikipedia). It also has access to the sanitization mechanisms, and it can produce the sanitized public text (as how users produce $\hat{D}$'s).

Our key idea is to let the service provider pretrain the encoder (*i.e.*, BERT) over the sanitized public text, making it more "robust" in handling $\hat{D}$'s. We thus initialize the encoder with the original BERT checkpoint and conduct further pretraining with an adapted masked language model (MLM) loss. In more detail, the adapted MLM objective is to predict the *original masked tokens* given the sanitized context instead of the one from the raw public text. We note that this is beneficial for improving the task utility, yet may breach the user privacy as the objective learns to "recover" the original tokens or semantics. In Section 2.5.4, our results will show that such pretrained BERT indeed improves accuracy, with comparable privacy as in original BERT.

Fine-tuning the full NLP pipeline. After pretraining BERT using sanitized public text, the service provider can further improve the efficacy of downstream tasks by fine-tuning the full pipeline. We assume that the ground-truth labels are available to the service provider, say, inferring from $\hat{D}$'s when they can preserve similar semantics to the raw text. Then, the sanitized text-label pairs are used for training/fine-tuning downstream task models, with gradients back-propagated to update the parameters of both the encoder and task layer. We leave more realistic/complex labeling processes based on sanitized texts as future work.

2.4.5 Definition of "Sensitivity"

Simply treating the top w of least frequent tokens (*e.g.*, according to a public reference corpus) as the sensitive token set already led to promising results (see Section 2.5.2). By this definition, stop words are mostly non-sensitive (*e.g.*, for $w = 0.9$ over the sentiment classification dataset we used, $\sim 98\%$ of the stop words are deemed non-sensitive). For context-specific corpus, this strategy is better than merely using stop words, *e.g.*, breast cancer becomes non-sensitive among breast-cancer patients.

Mechanisms	SST-2			MedSTS			QNLI		
	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$
Random	0.4986	0.4986	0.4986	0.0196	0.0196	0.0196	0.5152	0.5152	0.5152
FBDD[71]	0.5099	0.5143	0.5345	0.0201	0.0361	0.0452	0.5162	0.5256	0.5333
SANTEXT	0.5101	0.5838	0.8374	0.0351	0.5392	0.8159	0.5372	0.5598	0.8116
SANTEXT$^+$	**0.7796**	**0.7943**	**0.8516**	**0.4965**	**0.7082**	**0.8162**	**0.7699**	**0.7760**	**0.8131**
Unsanitized	0.9251			0.8527			0.9090		

Table 2.1: Utilities comparison of sanitization mechanisms under similar privacy levels using the GloVe embedding

Sophisticated machine-learning approaches or other heuristics could also be considered, *e.g.*, training over context-specific reference corpus or identifying tokens with personal (and hence sensitive) information (*e.g.*, names). We leave as future work.

Moreover, the definition of sensitivity may vary across users. Some may consider a token deemed non-sensitive by most other users sensitive. The original ULDP work [160] has discussed a personalized mechanism that preprocesses such tokens by mapping them to a set of semantic tags, which are the same for all users. These tags will be treated as sensitive tokens for the ULDP mechanism. Apparently, this approach is application-specific and may not be needed in some applications; hence we omit it in this work.

2.5 Experiments

2.5.1 Experimental Setup

We consider three representative downstream NLP tasks (datasets) with privacy implications.

Sentiment Classification (SST-2). When people write online reviews, especially the negative ones, they may worry about having their identity traced via writing too much that may provide hints of authorship or linkage to other online writings. For this task, we use the preprocessed version in GLUE benchmark [244] of (binary) Stanford Sentiment Treebank

(SST-2) dataset [218]. *Accuracy* (w.r.t. the ground truth included in the dataset) is used as the evaluation metric.

Medical Semantic Textual Similarity (MedSTS). Automated processing of patient records is a significant research direction, and one such task is computing the semantic similarity between clinical text snippets for the benefit of reducing the cognitive burden. We choose a very recent MedSTS dataset [249] for this task, which assigns a numerical score to each pair of sentences, indicating the degree of similarity. We report the *Pearson correlation coefficient* (between predicted similarities and human judgments) for this task.

Question Natural Language Inference (QNLI). Question-answering (QA) aims to automatically answer user questions based on documents. We consider a simplified setting of QA, namely QNLI, which predicts whether a given document contains the answer to the question. We use the QNLI dataset from GLUE benchmark [244].

We implement our sanitized mechanisms using Python and the sanitization-aware training using the Transformers library [260]. We use sanitized data to train and test prediction models for all three tasks. We either build vocabularies for the tasks using GloVe embeddings [179] or adopt the same BERT vocabulary [51]. Table 2.2 shows their sizes. Our sanitization-aware pretraining uses WikiCorpus (English version, a 2006 dump, 600M words) [200]. We start from the `bert-base-uncased` (instead of randomly initialized) model to accelerate the pretraining.

We set the maximum sequence length to 512, training epoch to 1, batch size to 6, learning rate to 5e-5, warmup steps to 2000, and MLM probability to 0.15. Our sanitization-aware fine-tuning uses the `bert-base-uncased` model for SST-2/QNLI, and `ClinicalBERT` [4] for MedSTS. We set the maximum sequence length to 128, training epochs to 3, batch size to 64 for SST-2/QNLI or 8 for MedSTS, and learning rate to 2e-5 for SST-2/QNLI or 5e-5

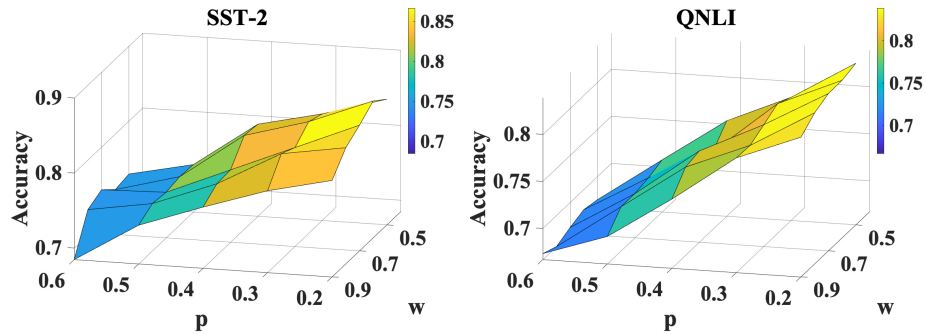


Figure 2.3: Performance of SANTEXT$^+$ over (w, p) when fixing $\epsilon = 2$ based on the GloVe embedding

for MedSTS. Other hyperparameters are kept default. Our hyperparameters followed the transformer library [260] and popular setups in the original dataset literature [244, 249].

2.5.2 Comparison of Sanitization Mechanisms

We first compare our SANTEXT and SANTEXT$^+$ with random sanitization and the state-of-the-art of [71] (FBDD). Here, we use the GloVe embedding as in FBDD for a fair comparison. Random sanitization picks a token from the vocabulary uniformly. We set the UMLDP parameters $p = 0.3, w = 0.9$ for SANTEXT$^+$ (while Figure 2.3 plots the impacts of p and w when fixing $\epsilon = 2$).

Table 2.1 shows the utility of the four mechanisms for the three selected tasks at different privacy levels. FBDD has a higher utility than random replacements. While both FBDD and SANTEXT are based on word embeddings, SANTEXT does not suffer from the "curse-of-dimensionality" and achieves better utility at the same privacy level. SANTEXT$^+$ achieves the best utilities in all cases since it allows the non-sensitive tokens to be noise-free, lowering the noise and improving the utility.

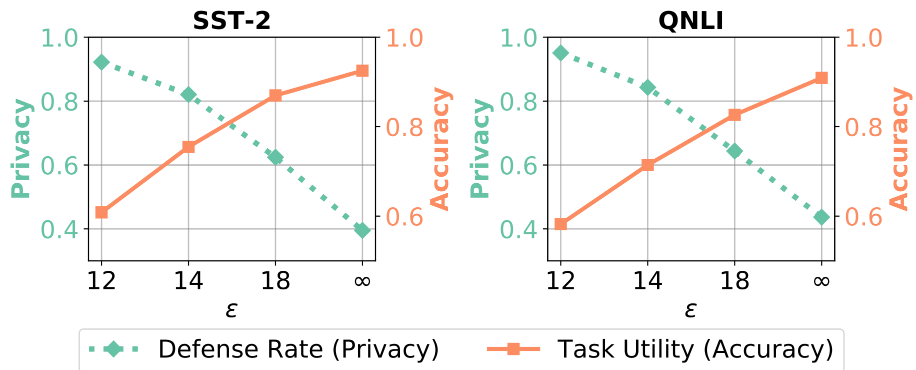


Figure 2.4: Privacy and Utility Tradeoffs of SANTEXT in terms of Defense Rate (of the Mask Token Inference Attack) versus Accuracy ($\epsilon = \infty$ means "unsanitized.")

In terms of efficiency, our SANTEXT and SANTEXT$^+$ are very efficient (*e.g.*, ~2min for the SST-2 dataset) compared with FBDD (~117min) when they all run on a 24 core CPU machine. This is because our mechanisms only need to compute the sampling probability once and use the same probability matrix for sampling each time, while FBDD needs to recalculate the additive noise and re-search the nearest neighbor each time.

Datasets	GloVe embeddings		BERT embeddings	
	$\mathcal{V}$	$\mathcal{V}_S$	$\mathcal{V}$	$\mathcal{V}_S$
SST-2	14,730	13,258	30,522	27,469
MedSTS	3,320	2,989		
QNLI	88,159	79,343		

Table 2.2: Sizes of vocabularies ($w = 0.9$ for $\mathcal{V}_S$)

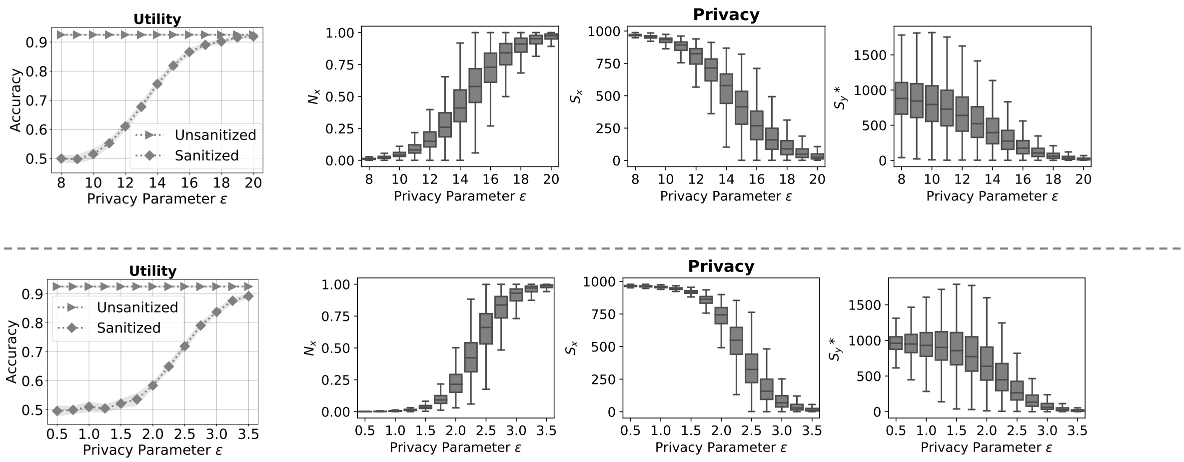


Figure 2.5: Influence of privacy parameter ϵ of SANTEXT on the utility and privacy (N_x, S_x, S_y^*) based on the SST-2 dataset: The top panel is based on BERT embeddings, and the bottom panel is based on GloVe embeddings.

2.5.3 Mask Token Inference Attack

From now on, we adopt the BERT embedding for its superiority. As (U)MLDP is distance-metric dependent, we need to use different ϵ's (*e.g.*, Figure 2.5) to ensure a similar privacy level, specifically, $\epsilon \cdot d$.

Our sanitization mechanisms provide broad protection for seen/unseen attacks at a fundamental level (by sampling noise to directly replace original tokens) with formally-proven DP, *e.g.*, two guesses of the original token with different styles are nearly probable in an attempt of authorship attribution [250] or other "indirect" attacks. Here, we consider a *mask token inference attack* as a representative study to "confirm the theory" by empirically measuring the "concrete" privacy level of sanitized texts.

To infer or recover original tokens given the sanitized text, one can let a pretrained BERT model infer the `MASK` token given its contexts. After all, BERT models are trained via masked language modeling. For each sanitized text of the downstream (private) corpus, we replace each token sequentially by the special token `[MASK]` and input the masked text to the

Datasets	ϵ	Utility		Δ_{privacy}
		Original	**+Pretrain**	
SST-2	12	0.6084	**0.6208**	0.0089
	14	0.7548	**0.7731**	0.0101
	16	0.8698	**0.8830**	-0.0046
QNLI	12	0.5822	**0.6037**	0.0076
	14	0.7143	**0.7309**	-0.0047
	16	0.8265	**0.8369**	-0.0039

Table 2.3: Sanitization-aware pretraining via SANTEXT

pretrained BERT model to obtain the prediction of the `[MASK]` position. Then, we compare the predicted token to the original token in the raw text. Figure 2.4 reports the defense rate (the proportion of unmatched tokens to total tokens) and task utility of sanitized texts (by SANTEXT) as well as unsanitized texts on SST-2 and QNLI. We see a privacy-utility trade-off: the more restrictive the privacy guarantee (smaller ϵ), the lower the utility score. Notably, we improve the defense rate substantially with only a small amount of privacy loss (*e.g.*, when $\epsilon = 16$, SANTEXT improves the defense rate by 20% with only 4% task utility loss over the SST-2 dataset in Figure 2.4).

2.5.4 Effectiveness of Pretraining

We then show how the sanitization-aware pretraining further improves the utility but does not hurt the original privacy. Specifically, Table 2.3 compares the accuracy of sanitization-aware fine-tuning based on the publicly-available `bert-base-uncased` model and our sanitization-aware pretrained one at different privacy levels on SST-2 and QNLI. Our sanitization-aware pretrained BERT models can obtain a 2% absolute gain on average. We conjecture that it can be improved since our pretraining only uses $1/6$ of the data used in the original BERT pretraining and 1 training epoch as an illustration.

To demonstrate that such utility improvement is not obtained by sacrificing privacy, we record the change of defense rate ($\Delta_{\mathsf{privacy}}$) in launching mask token inference attacks on the original BERT models and our sanitization-aware pretrained BERT models. As Table 2.3 confirmed, the privacy level of our sanitization-aware pretrained model is nearly the same as the original (sometimes even better).

2.5.5 Influence of Privacy Parameter

We aim at striking a nice balance between privacy and utility by tuning ϵ. To empirically show the influence of ϵ, we report the utility and privacy scores over the SST-2 dataset based on SANTEXT. The utility score is the accuracy over the test set. We define three metrics to "quantify" privacy. Firstly, $N_x = \Pr[\mathcal{M}(x) = x]$, which we estimate by the frequency of seeing no replacement by $\mathcal{M}()$. The output distribution of x has full support over $\mathcal{V}$, *i.e.*, $\Pr[\mathcal{M}(x) = y] > 0$ for any $y \in \mathcal{V}$. Yet, we are interested in the effective support $\mathcal{S}$, a set of y's with cumulative probability larger than a threshold, and then define S_x as its size. S_x can be estimated by the number of distinct tokens mapped from x. Both N_x and S_x can be related to two extremes of the Rényi entropy [203], defined as $H_\alpha(\mathcal{M}(x)) = \frac{1}{1-\alpha}\log(\sum_{y\in\mathcal{V}}\Pr[\mathcal{M}(x) = y]^\alpha)$, with an order $\alpha \geq 0$ and $\alpha \neq 1$. The two extremes are obtained by setting $\alpha = 0$ and $= \infty$, resulting in the Hartley entropy H_0 and the min-entropy H_∞. This implies that we can also approximate H_0 and H_∞ by $\log S_x$ and $-\log N_x$, respectively. Making them large increases the entropy of the distribution.

Another important notion is *plausible deniability* [15], *i.e.*, a set of x's could have led to an output y with a similar probability. We define S^*_y as the set size, estimated by the number of distinct tokens mapped to y.

We run SanText $1,000$ times for the whole SST-2 dataset vocabulary. As Figure 2.5 shows, when ϵ increases, the utility boosts and N_x increases while S_x, S_y^*, and the privacy level of the mechanism decrease, which gives some intuition in picking ϵ, *e.g.*, for $\sim$40% probability of replacing each token to a different one based on the BERT embeddings (top panel), we could set $\epsilon = 15$ since the median of N_x is $\sim$60% and the accuracy is $\sim$81%.

2.6 Conclusion

Great predictive power comes with great privacy risks. The success of language models enables inference attacks. There are only a few works in differentially private (DP) text sanitization, probably due to its intrinsic difficulty. A new approach addressing the (inherent) limitation (*e.g.*, in generality) of existing works is thus needed.

Theoretically, we formulate a new LDP notion, UMLDP, which considers both sensitivity and similarity. While it is motivated by text analytics, it remains interesting in its own right. UMLDP enables our natural sanitization mechanisms without the curse of dimensionality faced by existing works.

Practically, we consider the whole PPNLP pipeline and build in privacy at the root with our sanitization-aware pretraining and fine-tuning. With our simple and clear definition of sensitivity, our work already achieved promising performance. Future research in sophisticated sensitivity measures will further strengthen our approach.

Surprisingly, our PPNLP solution is discerning like a cryptographic solution: it is kind (maintains high utility) to the good but not as helpful to the bad (not boosting up inference attacks). We hope our results with different metrics for quantifying privacy can provide more insights in privacy-preserving NLP and make it accessible to a broad audience.

Chapter 3: Synthetic Text Generation with Differential Privacy: A Simple and Practical Recipe

In Chapter 2, we introduced a token-by-token text sanitization method with DP. However, the token-level DP cannot give us a very strong privacy guarantee and the token replacement does not consider the context information during the sanitization, which may hurt the utility of the generated text. In this chapter, we will propose a stricter privacy requirement for the synthetic text: we require that the *whole synthetic dataset* should be differentially private, i.e., any data instance should be statistically indistinguishable in the synthetic dataset. To achieve this, we show a simple, practical, and effective recipe: simply fine-tuning a generative language model with DP allows us to generate useful synthetic text while mitigating privacy concerns. Through extensive empirical analyses, we demonstrate that our method produces synthetic data that is competitive in terms of utility with its non-private counterpart and meanwhile provides strong protection against potential privacy leakages.

3.1 Introduction

The issue of privacy has gained increasing attention in the Natural Language Processing (NLP) community. Privacy attacks against common pipelines have demonstrated that models trained without formal guarantees can reveal membership information and reconstruct training data [214, 32]. Privacy concerns manifested through tightening legislation (e.g.,

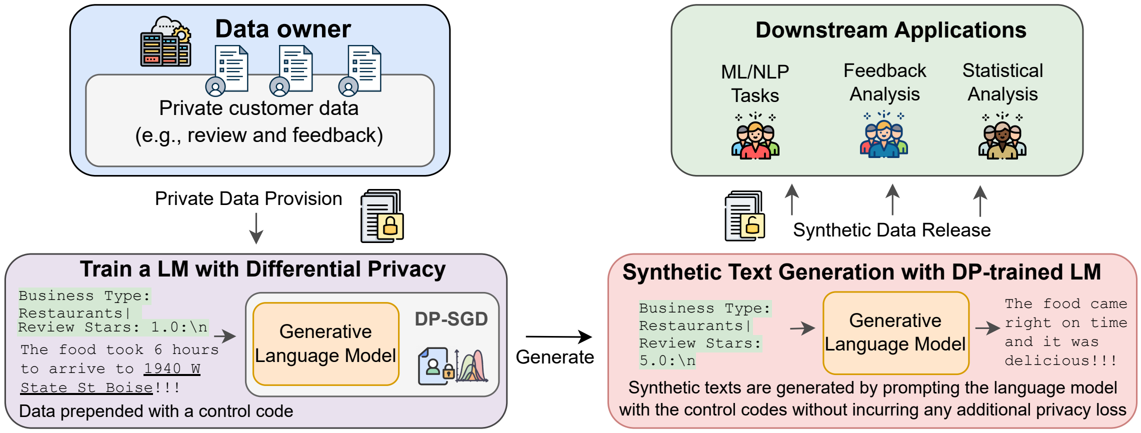


Figure 3.1: Illustration of our problem and methodology: Directly using private data for downstream analyses and applications often faces privacy issues. We propose to generate synthetic text with a formal privacy guarantee: we fine-tune a generative language model with DP and then leverage it for synthetic text generation using control codes. Privacy loss of the overall procedure can be controlled by the first stage as, by the robustness to post-processing property of DP, the second stage does not incur any additional privacy loss.

GDPR [9]) and growing discussions on policy and ethics call for improved approaches for privacy-preserving machine learning.

Among different approaches for learning with private data, learning with differential privacy (DP) [59] has become the gold standard as its formal guarantee enables reasoning about the privacy loss in a principled manner and makes the approach resilient to strong privacy attacks [28]. While recent developments have substantially improved the computational efficiency and privacy-utility trade-off of DP machine learning [224, 137, 277, 49, 25, 136, 156], these works have primarily focused on learning models that directly perform target tasks (e.g., text classification, dialog generation, etc.).

Different from the above works, in this work, we study synthetic text generation with DP by building generative text models with DP training algorithms (Figure 3.1). The goal of this approach is to learn a generative model that faithfully captures distributional properties of the training data (and the underlying distribution), as opposed to learning task-oriented models with specific functions. Compared to directly learning models for target tasks, this paradigm has several advantages: (1) DP-trained generative models can be used to draw synthetic data for learning an expanding set of task models without incurring any additional privacy loss; (2) Dataset debugging is made easy as synthetic text generated from DP trained models can be shared more freely, and inspecting its samples poses less of a privacy concern compared to examining the original private data directly [11]; (3) Synthetic data generated from a DP-trained model can likely be retained for longer periods of time under existing policies and guidelines (e.g., *right to be forgotten*) thanks to the fact that DP implies some degree of (approximate) machine unlearning [21, 210].

Despite the appeal, past work on DP synthetic text generation has not demonstrated the general feasibility of this approach in realistic settings [18]. Here, we initiate a systematic empirical study of the problem and show that DP language model fine-tuning, which was applied only to learn task models in previous works [137, 277], can be an effective solution to DP synthetic text generation. In particular, we show that simply fine-tuning progressively better (and larger) autoregressively pre-trained language models with appropriate hyper-parameters on (private) data leads to increasingly useful synthetic text. For instance, we fine-tune a GPT-2 Large model [189] on a review dataset with DP at $\epsilon = 4$ and then use it to generate synthetic text to build a classifier. The classification model achieves comparable performance to the classifier trained on the text generated by a GPT-2 model fine-tuned without DP, and only 2-4% (in accuracy) away from the classification model trained on the original dataset. Distributional similarity evaluation also confirms that the synthetic text distribution is close to the original data distribution. To empirically verify that DP fine-tuning limits the leakage of private information, we show that specially-crafted *canaries* [28] injected in the training data enjoy strong protection against potential leakage when the generators are fine-tuned with DP. Lastly, we uncover a novel phenomenon in language models trained with DP that is of independent interest. We show that text generation with a DP-trained model can exhibit a *length truncation effect* and produce completions that are generally shorter than their non-DP counterparts and instances in the original training dataset.

3.2 Background

3.2.1 Differential Privacy

We state the definition of differential privacy.

Definition 3.2.1 (Differential Privacy (DP) [60])**.** A randomized algorithm $M : \mathcal{D} \to \mathcal{S}$ is (ϵ, δ)-differentially private if for any two neighboring datasets $D, D' \in \mathcal{D}$ that differ exactly in a single data sample, and for all sets $S \subseteq \mathcal{S}$, we have:

$$\mathbb{P}[M(D) \in S] \leq e^{\epsilon}\mathbb{P}[M(D') \in S] + \delta$$

We note that this definition provides a rigorous privacy guarantee by theoretically bounding the effect of a single data sample in the dataset. For a differentially private algorithm, the output distribution is statistically similar whether any data sample appears or not in the input dataset. An algorithm can typically be made (ϵ, δ)-DP by bounding the contribution of a single data sample and adding controlled noise from a predetermined distribution (e.g., Gaussian) [58].

An appealing property of DP that is crucial to this work is robustness to post-processing. This property ensures that if the algorithm M satisfies (ϵ, δ)-DP, then so does $F \circ M$ for any deterministic or randomized function F. In other words, one can perform arbitrary post-processing without incurring any additional privacy loss.

3.2.2 Differentially Private Stochastic Gradient Descent

Deep learning models can be trained with DP via a modification of the stochastic gradient descent (SGD) algorithm [221, 12, 1]. The modified algorithm clips per-sample gradients to bound the contribution of individual examples. Noise from a Gaussian distribution is sampled and added to the sum of the clipped gradients in a batch to obfuscate the gradient update. The resulting algorithm, called Differentially Private Stochastic Gradient Descent (DP-SGD), can be shown to be DP for some (ϵ, δ) for each update of the model. Privacy parameters of the model at the end of the training can be computed via privacy composition algorithms [1, 84]. In the next section, we will utilize DP-SGD to train a language model with privacy for synthetic text generation.

3.3 Method

In this section, we formally state the problem and present our method (see Figure 3.1 for an illustration) that produces a synthetic version of private text data with differential privacy.

3.3.1 Problem Statement

Let $\mathcal{D}$ be a database representing the collection of token sequences from a fixed dictionary $\mathcal{V}$. We define a mapping $M : \mathcal{D} \to \mathcal{D}$ such that for a given dataset $D \in \mathcal{D}$, the goal is to generate a synthetic version $M(D) = \tilde{D}$ with privacy constraints and utility desiderata.

Regarding privacy constraints, we require that M be (ϵ, δ)-DP with domain $\mathcal{D}$. This requirement provides strong protection for the participants in the input dataset as this participation will be statistically indistinguishable to a certain degree through any adversary accessing the model or synthetic version of the dataset in the output.

For the case of utility, ideally, the synthetic version $\tilde{D}$ should be able to replace D in providing a training resource for models on relevant downstream applications. In other words, on target downstream tasks, models trained on the synthetic dataset $\tilde{D}$ are expected to have performance similar to the models trained on the original dataset D. More generally, distributional properties of the dataset D should be captured as much as possible in the synthetic version $\tilde{D}$ while satisfying the aforementioned privacy requirement. These will be extensively explored in Section 3.4.

3.3.2 Synthetic Text Generation with DP

Conventionally, to generate synthetic text, an auto-regressive language model (e.g. GPT-2 [189]) is trained on the original dataset and subsequently sampled using a sampling mechanism (e.g., beam search, top-k sampling [66], nucleus sampling [94], etc.) to produce synthetic sequences.

To make this operation differentially private, we adopt DP-SGD to fine-tune a pretrained generative language model. The post-processing property of DP ensures that once the language model has been fine-tuned with DP, sampling from the model incurs no extra privacy loss.

This approach, as it is, may not easily produce the same categorical distribution if it explicitly exists in the original dataset. As an example, feedback data collected from users on a set of products may contain product types and review scores associated with each data sample. It would be desirable to maintain the original distribution of product types and review scores in the synthetic version. We achieve this by building a conditional generator introduced in [114] to provide more explicit control over text generation. In this approach, using so-called control codes, the probability distribution of a text sequence

$x = (x_1, x_2, \ldots, x_n)$ is conditioned on a control code c and decomposed as

$$\mathbb{P}\left(x|c\right) = \prod_{i=1}^{n} \mathbb{P}\left(x_i|x_1, x_2, \ldots, x_{i-1}, c\right).$$

A neural network $p_\theta(\cdot)$ is then trained to model each conditional distribution. The model can later be used to generate new samples conditioned on a control code c by sequentially sampling $p_\theta(x_1|c), p_\theta(x_2|\tilde{x}_1, c), \ldots, p_\theta(x_m|\tilde{x}_1, \ldots \tilde{x}_{m-1}, c)$. The advantage of this approach is that it provides flexibility in the text generation of the model by allowing the conditional control codes to specify a particular style, domain, sentiment, category, etc. In the aforementioned example, control codes can be constructed as $c_{p,r} =$ `"Product type:` p `| Review score:` r`"` for different product type (p) and review score (r) pairs. In our method, we utilize control codes to prepend each sample with its corresponding categories as a simple preprocessing step. During the text generation, this allows us to use the control codes to generate as many samples as the original categorical distribution is preserved.

We point out that the categorical distribution in the original dataset may also be a piece of private information itself. However, its estimation could easily be privatized [61] and for simplicity, we ignore the low-cost privacy loss of this step and use the exact categorical distribution of the original dataset.

3.4 Analyses on a Public Review Dataset

In this section, we extensively analyze our method with experiments on a public benchmark dataset: Yelp Open Dataset[10], which has been widely adopted for language modeling and text classification tasks. We then apply our method to an internal private customer feedback dataset in Section 3.5.

[10]`https://www.yelp.com/dataset`

3.4.1 Experimental Setup

Dataset. The Yelp dataset contains review text data on businesses that can be studied for academic purposes. We select two attributes for the conditional generation as well as the downstream task applications: review stars (1-5) and business category. We sample 10 frequent business categories and remove the reviews that do not have ratings.[11] This results in a dataset that has 1.9M reviews for training, 5000 for validation, and 5000 for testing.

Implementation Details. We use the public repository [264] that is based on Huggingface [258] and Opacus [276] to fine-tune language models with DP. We fine-tune three language models GPT2 [189], GPT2-Medium, and GPT2-Large for synthetic text generation and RoBERTa-base [144] for all downstream text classification tasks. The control codes are constructed based on attributes such as `"Business Type: Restaurants | Review Stars: 5.0"` and prepended to each sample. Hyperparameters are specified in Appendix B.1. For both synthetic text generation and classification, we set the maximum sequence length as 128 unless otherwise stated. We evaluate the models during training on the validation dataset and use the checkpoint that achieves the best validation performance for the final evaluation on the test set.

We set the privacy parameters $\epsilon = 4$ and $\delta = 1/(N \cdot \log N)$, where N is the size of the original dataset. The additive noise scale is calculated by the numerical composition algorithm [83] given the batch size and number of epochs corresponding to each setting in Appendix B.1 for DP training.

For generating the synthetic text samples, we employ top-k [66] and nucleus sampling (top-p) [94], and set $k = 50$ and $p = 0.9$. To produce synthetic datasets preserving the

[11] 10 categories: Restaurants, Bars, Shopping, Event Planning & Services, Beauty & Spas, Arts & Entertainment, Hotels & Travel, Health & Medical, Grocery, Home & Garden.

Data Type	Data Generator	ϵ	Rating	Category
Original	-	-	0.7334	0.7752
Synthetic	GPT2	∞	0.6892	0.7584
		4	0.6656	0.7478
	GPT2-Medium	∞	0.6878	0.7550
		4	0.6756	0.7486
	GPT2-Large	∞	0.7090	0.7576
		4	0.6936	0.7568

Table 3.1: Synthetic text generation with DP produces models that have similar performance in the downstream tasks compared to the models trained on synthetic text produced without privacy. The first row shows the performance of models trained on the original dataset. The accuracies are reported on two classification tasks: Rating (review star) and Category (Business).

categorical distributions (i.e., business category, review star), we generate 100K samples from the fine-tuned models using the appropriate control codes.

3.4.2 Synthetic Data Performance on Downstream Tasks

One way to measure the quality of the synthetic dataset is to examine the performance of downstream task models trained on it.

We fine-tune RoBERTa-base models [144] for classifying ratings (review stars) as well as business categories on the synthetic dataset. We also compare them with models trained on the original dataset. All models are tested on the same original test set.

The results are presented in Table 3.1. The downstream task models trained on the synthetic data generated by GPT2 with DP ($\epsilon = 4$) achieve comparable performance to the ones trained on the synthetic data generated without DP ($\epsilon = \infty$) and the ones trained on the original dataset. We could also identify the quality of the synthetic generations increases when we employ larger pretrained language models and the performance gap between private and non-private generations decreases. We note an interesting observation

Data Type	Data Size	DP Position	Task Accuracy	
			Rating	Category
Original	1.9M	Task modeling	0.7014	0.7644
Original	100K	Task modeling	0.6689	0.7552
Synthetic	100K	Data Generator	0.6936	0.7568

Table 3.2: Models trained on synthetic data that is generated with DP-trained GPT2-Large (the last row) have similar performance compared to the models directly trained on the original dataset with DP (the first two rows).

that models trained on synthetic data generated by GPT2-Large with DP achieve better performance in rating and similar performance in the category compared to the models trained on synthetic data generated by GPT2 without DP.

These results demonstrate the great potential of our method used to generate synthetic data for different downstream applications.

3.4.3 Synthetic Data Generation with DP v.s. Downstream Task Modeling with DP

It is natural to ask how the downstream task models trained on the synthetic text generated by a language model trained with DP compare to the downstream task models directly trained on the original data with DP.

We show the results of this comparison in Table 3.2. Since for the default downstream task setting, we only generate 100K synthetic data, for a fair comparison, we report the downstream task models trained on the whole original dataset (1.9M) and a sampled original dataset (100K) with DP. As we can see, when adopting the same privacy parameter ($\epsilon = 4$), the two ways for leveraging DP (Synthetic Data Generation with DP and Downstream Task Modeling with DP) achieve very comparable performances. However, the overall privacy loss of training two models on the same dataset will be higher than $\epsilon = 4$ and it will

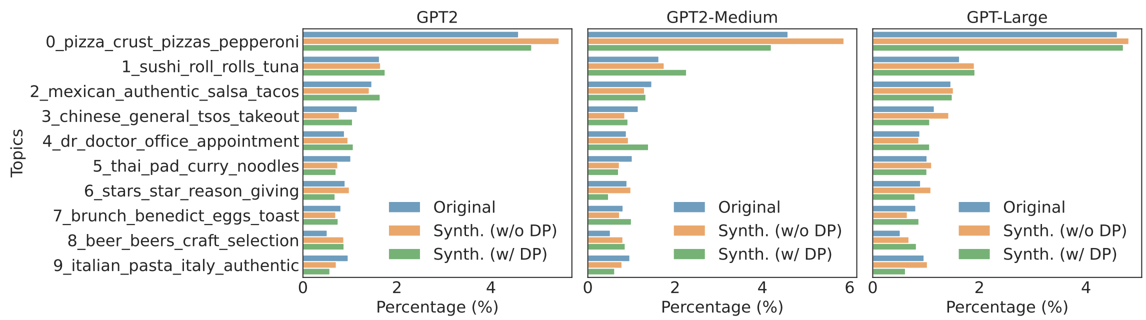


Figure 3.2: Topic distribution between synthetic and the original data is similar when the generations are produced by GPT2 models trained with DP. The similarity improves as the model size increases. Best viewed in color.

accumulate for additional downstream tasks, whereas by the post-processing property of DP, one can train any number of models for various downstream tasks on the synthetic data generated by a language model trained with DP without incurring any additional privacy loss.

3.4.4 Similarity between Synthetic and Original Datasets

To further test the quality of the synthetic generations, we measure the similarity between the synthetic dataset and the original dataset. Unlike typical natural language generation tasks (e.g, machine translation, summarization) that often have gold references for the evaluation, without a one-to-one mapping it is hard to directly compare the synthetic generations with the original dataset. Here, we measure the "similarity" from three different perspectives: Embedding Distance, Topic Difference, and Text Length Distribution.

Embedding Distance. We embed the synthetic generations and the original data with sentence-transformer [201] models and calculate the distance between two distributions based on various distance metrics:

Generator	ϵ	F1↑	FID↓	MAUVE↑
GPT2	∞	0.5199	3.2368	0.7158
	4	0.4786	4.7998	0.5579
GPT2-Medium	∞	0.5446	3.1464	0.7222
	4	0.5076	4.1880	0.6085
GPT2-Large	∞	0.5852	3.0978	0.7238
	4	0.5140	4.1352	0.6093

Table 3.3: Distribution distance between the synthetic and original data based on various metrics. Performance improves as larger models are used.

1) F1 Score (Harmonic mean of Precision and Recall) [124]. The Precision and Recall estimate the average sample quality and the coverage of the sample distribution by checking whether a generation falls within the surroundings (e.g., $k = 3$ nearest neighbors) of any original samples (measured by the Euclidean distances) and whether an original sample falls within the surroundings of any generations.

2) Fréchet Inception Distance (FID) [91]. The FID score is originally proposed to measure the quality of synthetic images in computer vision. Here we re-purpose it for synthetic text evaluation. It first calculates feature-wise mean and covariance matrices of the embedding vectors and then measures the distance of two sets based on Fréchet distance (Wasserstein-2 distance).

3) MAUVE [180] compares the distributions of the synthetic data and the original data using divergence frontiers. Specifically, after embedding the text into embedding vectors, it first groups them into several clusters and then count the cluster assignments to form histograms. Finally, a divergence curve built upon the histograms is plotted and the area under the curve is reported as the metric to measure the gap between the two distributions.

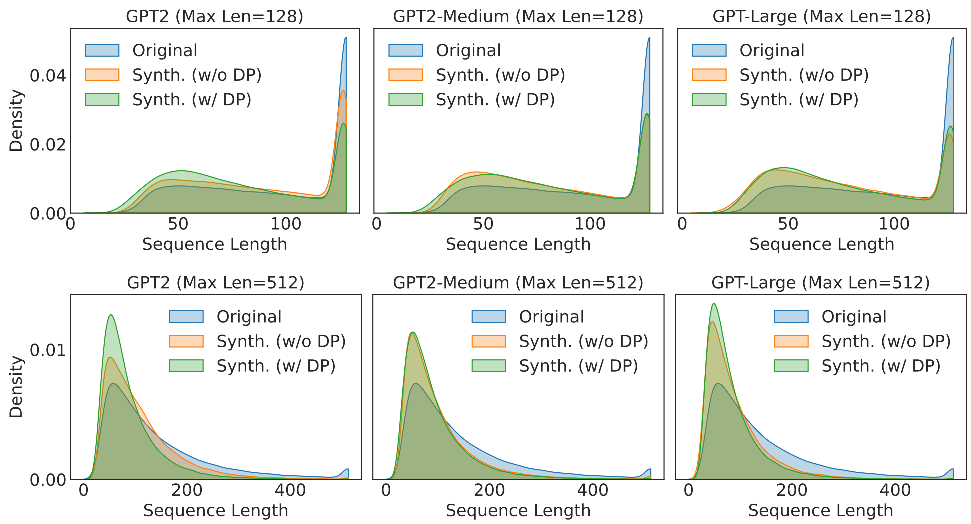


Figure 3.3: Synthetic data that is generated with or without privacy includes shorter sequences compared with the original data. This is more pronounced in the case where the synthetic data is produced with DP, especially for the small model.

It is important to note that the absolute scale of these metrics varies as different embedding models are used. We thus run the evaluations with 5 different pre-trained sentence transformers[12] and then take the average for each metric.

Table 3.3 shows the distribution distances between the synthetic data and the original data based on the metrics introduced above. Similarly, we observe that the quality of the synthetic data improves as we use larger pretrained models for private training. Similar to the results of the previous section, we observe that F1 score of GPT2-Large model with DP (the last row) matches the F1 score of GPT2 model without privacy (the first row). On the other hand, there remains a gap between synthetic generations with and without DP for FID and MAUVE.

[12]5 sentence-transformers we use from huggingface.co: "all-MiniLM-L6-v2", "paraphrase-MiniLM-L6-v2", "all-mpnet-base-v2", "stsb-roberta-base-v2", "distilbert-base-nli-stsb-mean-tokens".

Topic Difference. Another angle to measure the similarity between the synthetic and the original data is to look at their topic distributions. Topic modeling is a frequently used approach to identify hidden semantic structures (or abstract "topics") in a collection of documents. We add the synthetic generations and the original data in one collection and leverage an unsupervised topic model BERTopic [85] to extract the top 10 frequent topics in the documents. We plot the distributions of these topics for both synthetic data and the original data in Figure 3.2. We observe from the results that the distributions of topics for the synthetic data with or without DP are quite similar to the original. This further demonstrates the good quality of the synthetic data generated with our approach.

Text Length Distribution. We finally investigate the sequence lengths of the synthetic data and compare them with the original data. To better understand whether the maximum sequence length or the truncation during pre-processing phase would have a significant influence on the generations, we train two sets of generative models with the maximum sequence lengths of 128 and 512.

We plot the density of the sequence lengths in Figure 3.3. We note that the synthetic data that is generated with or without privacy is in general shorter than the original data. We further observe that for GPT2 the synthetic data that is generated with DP has more weight on shorter sequences compared to the one without DP. Although this issue is somewhat alleviated with the increase of model size as we observe that the distributions between the two cases are very close for GPT2-Medium and GPT2-Large, it is not fully resolved and we can still notice that the generations with DP are slightly shorter compared with their non-private counterparts (e.g., avg length 84.5 v.s. 89.4 for GPT2-Large).

ϵ	Repetition=1		Repetition=10		Repetition=100	
	Rank	Leaked	Rank	Leaked	Rank	Leaked
∞	1017/10000	0%	1/10000	0%	1/10000	80%
4	3926/10000	0%	3320/10000	0%	969/10000	0%

Table 3.4: Generations by a language model trained with DP show strong privacy protection against the leakage of injected canary sequences. The second column shows the percentage of canaries appearing in the synthetic generations and the first column shows the rank of the canaries among a similar set of candidates by the model's perplexity.

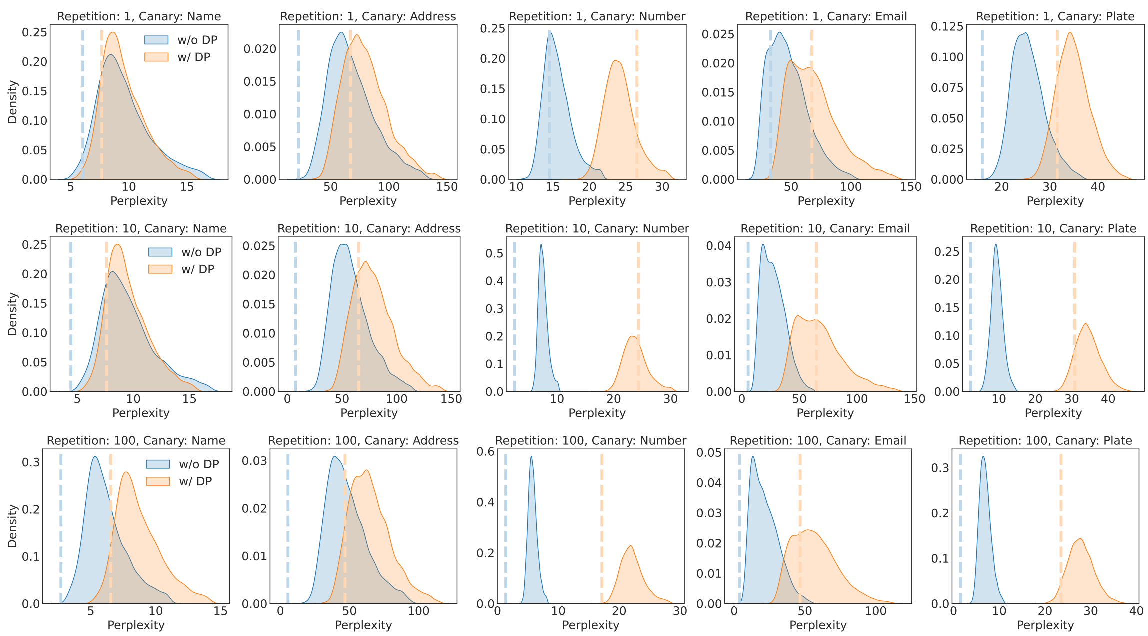


Figure 3.4: Distributions of perplexities of the secret parts of injected canary sequences among their similar set of candidates measured by GPT2 models trained with and without DP. The dashed lines represent the perplexity of the secret parts. Even a single-time occurring private information can achieve top rank in a non-private model which is not the case in the models trained with DP.

3.4.5 Leakage of Private Information

In this section, we empirically measure to what extent training with DP prevents the leakage of private data. For this purpose, we use the approach introduced in [28] to assess the risk of unintended memorization (and leakage in the synthetic version) of private information. We construct sequences with particular private information contained in them (called canaries) and inject them into the original training data to see how difficult it is to reconstruct them during the generation phase.

In our experiments, we construct 5 artificial review-style canary sequences (e.g., *"The food took literally 6 hours to arrive to 1940W State St Boise."*; The other canaries are stated in Appendix B.2). We perform a series of experiments where we inject these 5 canary sequences with various repetition rates into the original dataset. Once we generate the synthetic dataset, we check whether the secret part (underlined text in the example) of any of the canary sequences appears in the synthetic dataset. The results are shown in Table 3.4 under the column *"Leaked"*. We observe that the secret part of the canary sequences never appears in the synthetic dataset when the corresponding language model is trained with DP. On the other hand, at high a repetition rate such as 100, 4 out of 5 canary sequences verbatim appear in the synthetic dataset when the corresponding language model is trained without DP.

We note that the appearance of the canary sequences in the synthetic dataset is tied to the way we perform the text generation (i.e. the methods we use such as top-p sampling). This may not assess the privacy risk exhaustively as synthetic data can be generated with other decoding methods and hyperparameters. Therefore, we furthermore focus directly on the model and investigate the rank of the secret part of a canary based on its perplexity given by the model among a similar set of candidates. We construct 10K similar candidates for each canary type by replacing the canary placeholder with another sampled named entity in the same category. The named entity lists are either downloaded from the Web (e.g., name and address) or generated randomly following some regular expressions (e.g., number, email, plate).

We show the average rank of the secret part of canary sequences in Table 3.4 under the column *"Rank"*. Moreover, we also show the perplexity distributions of all similar candidates for each type in Figure 3.4. We conclude our interesting findings as follows:

1) For all repetition settings, we observe that training the language model with DP significantly lowers the privacy leakage risk as the secret part of canary sequences do not achieve low rank and are not distinguishable among the similar candidates.

2) When the canary sequence only appears once in the training set, the risk of being extracted in the generation is relatively low. However, for some canaries (e.g., Address and Plate in Figure 3.4), they are still ranked in first place. This indicates that in practice, even if some private information appears only once in the training set, models still are likely to memorize it and this can lead to a leakage in synthetic generations. And if we repeat the canary sequences for 10 and 100 times, they are always ranked in first place without DP protection. By contrast, when equipped with DP, the *Rank* of inserted sequences are much higher and the *Leaked* percentage is always 0.

3.5 Results on Private Customer Feedback

To further demonstrate our method's potential, we evaluate it on an internal private feedback dataset collected from customers.

Dataset. Microsoft receives large volumes of customer feedback for its products. While this collection of data contains highly valuable information for product improvements, it cannot be arbitrarily processed due to privacy regulations such as General Data Protection Regulation (GDPR) [9]. This necessitates the application of privacy-preserving tools while

Data Type	ϵ	A1	A2	A3
Original	-	0.686	0.602	0.638
Synthetic	∞	0.643	0.576	0.573
	4	0.592	0.536	0.520
	8	0.639	0.537	0.554

Table 3.5: Downstream task accuracy of models trained on the private customer feedback data and synthetic data generated by GPT2-Medium models with various privacy levels.

handling the usage of data collected from users. A potential way to overcome privacy concerns and perform downstream tasks is using a synthetic version of this dataset that is generated with a formal privacy guarantee such as DP. We pull 1M customer feedback in our scenario. For downstream tasks we are interested in three attributes of the feedback, which we call A(ttribute)1, A2 and A3. Attributes can be a number of product characteristics including, but not limited to, user satisfaction scores, date/time range, product name, product type, location, etc. Using the attributes (A1, A2, A3) together with a particular combination of their respective values, such as (V_{A1}, V_{A2}, V_{A3}), the conditional text generation prompt becomes: "`A1:` V_{A1} `| A2:` V_{A2} `| A3:` V_{A3}". We use GPT2-Medium model with the settings described in Section 3.4.1 in our scenario.

Downstream Task Performance. Similar to Section 3.4.2, to measure the quality of generated synthetic data we evaluate the performance of classification models trained on them. We train three multi-class classification models, to predict three attributes A1, A2, and A3 respectively. We present the results in Table 3.5. We observe that the performance gap between models trained on synthetic data generated with DP ($\epsilon = 4$) and without DP ($\epsilon = \infty$) is more pronounced in our scenario. This is primarily due to the dataset size, which is roughly half of the one adopted in Section 3.4. We note that the gap can also be improved by relaxing the privacy loss as it can be seen from the results with DP ($\epsilon = 8$).

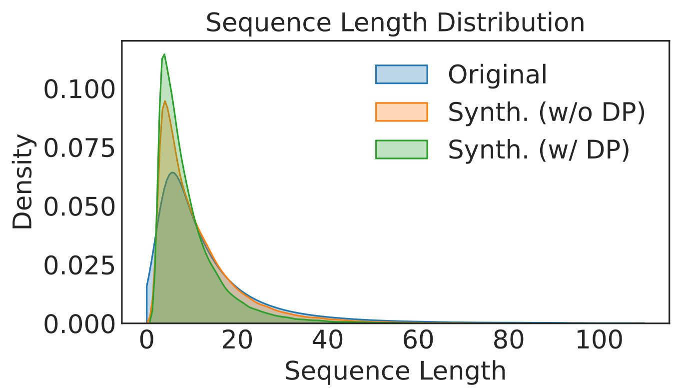


Figure 3.5: Synthetic data generated with DP tends to be shorter compared to the data generated without DP. The plot shows sequence length distributions of the synthetic data generated with and without DP and the original customer feedback data.

Text Length Distribution. We further compare the sequence lengths of the synthetic data generated with and without DP to the original dataset. The results are plotted in Figure 3.5. We notice a similar phenomenon that the data generated with DP exhibits a length truncation effect compared to the data generated without DP by the shift of the distribution from the tail towards the shorter sequence lengths with DP.

3.6 Related Work

Synthetic Data Generation with DP. The problem of DP synthetic data generation has been widely studied for tabular and image data in machine learning. Notable works in the literature on DP tabular data generation address the privacy-utility trade-off problem by building Bayesian networks [282], by preserving marginals [152], or through training generative adversarial networks with DP-SGD [121, 267, 105, 229]. The literature on DP image generation has so far mostly focused on GAN-based methods [11, 267, 166]. To the best of our knowledge, the only work in the literature on DP synthetic text generation is that by [18] which preliminarily outlined potential approaches without going in depth.

We point out that other one-to-one mapping approaches including both token-level [250, 70, 71, 269, 270, 16, 188, 279] and sentence-level [119, 89, 155, 251] perturbations fail to satisfy our privacy requirement outlined in Section 3.3.1 even though they possess certain DP guarantees themselves. This is because we require that the procedure of synthetic text generation should be statistically similar whether a data sample appears in the original dataset or not. These one-to-one mapping methods focus on producing a perturbed version of a single data sample, therefore, cannot fulfill this requirement. Besides, such one-to-one perturbations cannot meet the requirement of GDPR [9] with regard to “linkability” since the data owner can always link the perturbed text to a specific user as long as they keep the user meta record. However, our method can fulfill the requirement as the data owner cannot link any of the generated sequences to a specific user.

DP fine-tuning of language models. DP fine-tuning has been recently demonstrated to be a simple and effective approach for building performant models for solving a variety of NLP tasks including text classification, table-to-text generation, and dialog generation [137, 277]. However, past works have not studied these techniques for the problem of synthetic text generation. Unlike the above works, we initiate a careful empirical study of private fine-tuning for building synthetic text generation models, measure the different aspects of the approach, and demonstrate its general effectiveness as well as its unique limitations.

3.7 Conclusion

In this chapter, we show a simple and practical recipe for generating synthetic text data with privacy guarantees. Our method is built upon the large pretrained language models and differential privacy, where the former enables us to generate high-quality synthetic text data and the latter provides formal privacy guarantees. We conduct comprehensive experiments evaluating both the utility and privacy risks of the synthetic data. The results demonstrate that our method can generate high-quality data while protecting it from privacy leakage.

Part III: Data Synthesis for Model Training and Evaluation

In Part II, we discussed how to generate high-utility synthetic text meanwhile mitigating privacy concerns in the Data Collection stage. In this part, we move to the second stage of the NLP life cycle: Model Training and Evaluation.

For the model training, as pre-training fine-tuning becomes a standard paradigm for building NLP applications, how to better align these two phases becomes significantly crucial for improving the final task performance. Generally, the model is trained with self-supervised language modeling objectives in the pre-training phase and task-specific knowledge might not be learned. To solve the challenge, in Chapter 4, we will talk about how to leverage synthetic data to pre-train more accurate models for open-domain question answering. Model evaluation often requires a metric trained on diverse, high-quality data. Synthetic data, generated through rule-based systems and human intervention, can meet this need. In Chapter 5, we will discuss repurposing and producing such data for an attribution evaluation scenario.

Chapter 4: C-MORE: Pretraining to Answer Open-Domain Questions by Consulting Millions of References

We consider the problem of pretraining a two-stage open-domain question-answering (QA) system (retriever + reader) with strong transfer capabilities. The key challenge is how to construct a large number of high-quality question-answer-context triplets without task-specific annotations. Specifically, the triplets should align well with downstream tasks by: (i) covering a wide range of domains (for open-domain applications), (ii) linking a question to its semantically relevant context with supporting evidence (for training the retriever), and (iii) identifying the correct answer in the context (for training the reader). Previous pre-training approaches generally fall short of one or more of these requirements. In this work, we automatically construct a large-scale corpus that meets all three criteria by consulting millions of references cited within Wikipedia. The well-aligned pre-training signals benefit both the retriever and the reader significantly. Our pre-trained retriever leads to 2%-10% absolute gains in top-20 accuracy. And with our pre-trained reader, the entire system improves by up to 4% in exact match.

4.1 Introduction

Open-domain question answering (QA) aims to extract the answer to a question from a large set of passages. A simple yet powerful approach adopts a two-stage framework

[37, 113], which first employs a retriever to fetch a small subset of relevant passages from large corpora (i.e., *retriever*) and then feeds them into a *reader* to extract an answer (text span) from them. Due to its simplicity, a sparse retriever such as TF-IDF/BM25 is generally used together with a trainable reader [158]. However, recent advances show that transformer-based dense retrievers trained on supervised data [113] can greatly boost the performance, which better captures the semantic relevance between the question and the correct passages. Such approaches, albeit promising, are restricted by the limited amount of human annotated training data.

Inspired by the recent progresses of language models pretraining [50, 127, 87, 206], we would like to address the following central question: *can we pretrain a two-stage open-domain QA system (retriever + reader) without task-specific human annotations?* Unlike general language models, pretraining such a system that has strong transfer capabilities to downstream open-domain QA tasks is challenging. This is mainly due to the lack of well-aligned pretraining supervision signals. In particular, we need the constructed pretraining dataset (in the form of question-answer-context triplets) to: (i) cover a wide range of domains (for open-domain applications), (ii) link a question to its semantically relevant context with supporting evidence (for training the retriever), and (iii) identify the correct answer in the context (for training the reader).

There have been several recent attempts in addressing these challenges. ORQA [127] creates pseudo query-passage pairs by randomly sampling a sentence from a paragraph and treating the sampled sentence as the question while the rest sentences as the context. REALM [87] adopts a retrieve-then-predict approach, where the context is dynamically retrieved during training and an encoder (reader) predicts the masked token in the question based on the retrieved context. The retriever pretraining signals constructed in these approaches

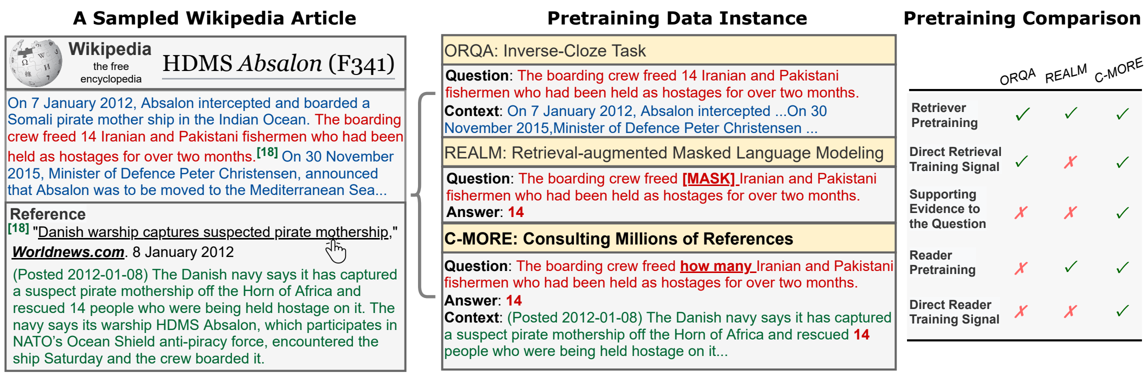


Figure 4.1: Different pretraining methods for open-domain QA. Our **C-MORE** pretrains both retriever and reader by using direct signals extracted from millions of references cited in the verified knowledge source.

are not aligned with question-context pairs in open-domain QA settings. For example, as shown in Figure 4.1, the context (in blue color) of ORQA pretraining data instance does not contain direct supporting evidence to the question. Likewise, the dynamically retrieved context in REALM cannot be guaranteed to contain direct supporting evidence either. In addition, existing pretraining methods [127, 87] mostly focus on the retriever and do not jointly provide direct pretraining signals for the reader (Figure 4.1).

To meet all three aforementioned criteria, we propose a pretraining approach named **C**onsulting **M**illions **O**f **RE**ferences (**C-MORE**), which automatically constructs pretraining data with well-aligned supervision signals (Figure 4.1). Specifically, we first extract three million statement-reference pairs from Wikipedia along with its cited references. Then, we transform them into question-answer-context triplets by replacing a potential answer span in the statement (e.g., *"14"* in the Figure 4.1) by an interrogative phrase (e.g, *"how many"*). Such kind of pseudo triplets are in the exact same form as human-annotated ones, and the question is linked to the context that contains the most direct-supporting evidence, a highly desirable feature for open-domain QA tasks. We experiment the pretraining with a widely-adopted open-domain QA system, Dense Passage Retriever (DPR) [113]. The experimental results show that our pretrained retriever not only outperforms both sparse and dense retrieval baselines in the zero-shot retrieval setting (2%-10% absolute gain in top-20 accuracy), but also leads to further improvement in the downstream task finetuning. By integrating with our pretrained reader, the entire open-domain pretraining improves the end-to-end QA performance by 4% in exact match.

4.2 Method

Recall that we want to automatically construct a large-scale open-domain QA pretraining dataset that satisfies three criteria: (i) The dataset should cover a wide range of domains for the open-domain QA purpose. (ii) The context passage is semantically relevant to the question and contains direct supporting evidence for answering the question. (iii) The correct answer span in the context passage for answering the question should also be identified for training the reader. This section first discusses how to extract a large amount of statement-reference pairs from the Wikipedia and then explain how to construct pseudo question-answer-context triplets for pretraining open-domain QA systems.

4.2.1 Statement-Reference Pairs Collection

Wikipedia articles usually contain a list of knowledge sources (references) at the end that are verified by human editors to support the statements in the articles [139]. And the reference documents always consist of strong supporting evidence to the statements. For example, as shown in Figure 4.1, the document (in green color) contains the direct evidence *"...rescued 14 people who were being held hostage on it..."* to support the query (red text) *"The boarding crew freed 14 Iranian and Pakistani fishermen who had been held as hostages over two months"*. Additionally, such knowledge sources are often organized in a good structure and can be automatically extracted and processed. Moreover, the statement-reference pairs in Wikipedia cover a wide range of topics and domains. Thus, when converted into question-context pairs, they satisfy the first two criteria and are suitable for training an accurate dense retriever at a large scale.

In our study, we extract around six million statement-reference pairs from Wikipedia. We filter the pairs whose reference documents are not reachable and finally obtain around three million statement-reference pairs (see statistics in Appendix Table 4.1). The data collection method we proposed is very general and therefore can be easily extended to other domains, e.g., WikiEM (wikem.org) for medical domain or other languages, e.g., Baidu Baike (baike.baidu.com) for Chinese.

4.2.2 QAC Triplets Construction

We now explain how to further convert the statement-reference pairs into question-answer-context pairs. Inspired by previous unsupervised extractive QA work [131], we extract entities as potential answers to construct pseudo question-answer-context pairs where an answer span is extracted from the context given an question to accommodate the extractive QA setting. Specifically, we first adopt an off-the-shelf named entity recognition tool spaCy [95] to identify entities in each query. Next, we filter the entities that do not appear in the evidence based on string matching. If multiple entities are found, we sample one of them as the potential answer to the query. The sampled entity in the query is replaced by an interrogative phrase based on the entity type (e.g., a `[DATE]` entity will be replaced by phrases such as *"when"*, *"what date"*. In this way, we can construct question-answer-context triplets to train open-domain QA models. See more question reformation rules in Appendix Table C.1).

Data Type	Dataset	Train	Dev	Test
Pretraining	`C-MORE`	2.96M	40K	-
Finetuning QA Data	NaturalQuestion	58,880	8,757	3,610
	TriviaQA	60,413	8,837	11,313
	WebQuestion	2,474	361	2,032

Table 4.1: Statistics of pretraining and finetuning data.

Settings	Methods	Training Data	Top-20 Accuracy			Top-100 Accuracy		
			NQ	TQA	WebQ	NQ	TQA	WebQ
Unsupervised	BM25	-	59.1*	66.9*	55.0*	73.7*	76.7*	71.1*
	ORQA [127]	Wikipedia	50.6	57.5	-	66.8	73.6	-
	REALM [87]	Wikipedia	59.8	68.2	-	74.9	79.4	-
	`C-MORE`	**Wikipedia**	**61.9**	**72.2**	**62.7**	**75.8**	**81.3**	**78.5**
Domain Aaptation	DPR-NQ	NaturalQuestion	-	69.7	69.0	-	79.2	78.8
	+ `C-MORE`	+ Wikipedia	-	**72.8**	**71.2**	-	**81.6**	**81.3**
	DPR-TQA	TriviaQA	69.2	-	71.5	80.3	-	81.0
	+ `C-MORE`	+ Wikipedia	**71.0**	-	**74.3**	**81.7**	-	**83.2**
	DPR-WebQ	WebQ	56.1	66.1	-	70.7	77.6	-
	+ `C-MORE`	+ Wikipedia	**67.3**	**74.2**	-	**79.2**	**82.6**	-
Supervised	DPR-supervised	Supervised Data	78.4*	79.4*	73.2*	85.4*	85.0*	81.4*
	+ `C-MORE`	+ Wikipedia	**80.3**	**81.3**	**75.0**	**86.7**	**85.9**	**83.2**

Table 4.2: Overall retrieval performance of different models. Results marked with "*" are from DPR [113], "" are from [206] and "-" means it does not apply to the current setting.

4.3 Experiment

4.3.1 Experimental Setup

Pretraining Model Architecture. Since conceptually the construed triplets is in the same format as the annotated QA data, they can be used to pretrain any existing neural open-domain QA model. Here, we adopt DPR [113], which consists of a dual-encoder as the retriever and a BERT reader, considering its effectiveness and popularity. Specifically, the retriever first retrieves top-k (up to 400 in our experiment) passages, and the reader assigns

Row	Model Architecture	Retriever		Reader		NQ	TQA	WebQ
		Pretrain	Finetune	Pretrain	Finetune			
1	ORQA [127]	✓	✓	✗	✓	33.3	45.0	36.4
2	REALM [87]	✓	✓	✓	✓	40.4	-	40.7
3	DPR [113]	✓	✗	✓	✗	11.3	24.8	4.5
4		✗	✗	✗	✓	32.6	52.4	29.9
5		✓	✗	✗	✓	**35.3**	**55.1**	**32.1**
6		✗	✓	✗	✓	41.5	56.8	34.6
7		✓	✓	✗	✓	**41.9**	58.6	35.6
8		✓	✓	✓	✓	41.6	**60.3**	**38.6**

Table 4.3: End-to-end QA performance based on different retrievers and readers. Note that we only test the effectiveness of `C-MORE` based on the DPR [113] model architecture. ORQA and REALM are listed here as references. The retriever of Row 4 is BM25, which does not involve either pretraining or finetuning.

a passage score to each retrieved passage and extracts an answer with a span score. The span with the highest passage selection score is regarded as the final answer. The reader and retriever can be instantiated with different models and we use `BERT-base-uncased` for both of them following [113].

Pretraining Data Processing. For our extracted pseudo question-answer-context triplets, sometimes the context (reference document) is too long to fit into a standard BERT (maximum 512 tokens) in the DPR model. Thus, we chunk a long document into n-word text blocks with a stride of m. Without loss of generality, we use multiple combinations of n and m: $n = \{128, 256, 512\}$, $n = \{64, 128, 256\}$. Then we calculate relevance scores (using BM25) of the derived blocks with the question and select the most relevant block as the context. Note that the retrieval step is done within the single document (usually less than 20 text blocks). In contrast, the baseline model (Section 4.3.2) - sparse retriever BM25 - looks up the entire knowledge corpus (20M text blocks). In this way, we can automatically collect the most relevant context that supports the query from a long article.

Finetuning QA Datasets. We consider three popular open-domain QA datasets for finetuning: NaturalQuestions (NQ) [122], TriviaQA (TQA) [108], and WebQuestions (WebQ) [14], whose statistics are shown in Table 4.1.

Following the setting of DPR [113], we use the Wikipedia as the knowledge source and split Wikipedia articles into 100-word units for retrieval. All the datasets we use are the processed versions from the DPR implementation.

Overlap between Pretraining and Finetuning Datasets. Though both **C-MORE** and downstream QA data are constructed based on Wikipedia, the overlap between them would be very little. **C-MORE** extracts query from Wikipedia while the queries of downstream QA data are annotated by human. **C-MORE** extracts contexts from the external referenced pages (general Web) while the downstream QA data extract contexts from Wikipedia.

Implementation Details. For pretraining, we set training epochs to 3, batch size to 56 for retrievers and 16 for readers, and learning rate to 2e-5. We select the best checkpoint based on the pretraining dev set. For finetuning, we use the same set of hyperparameters as the original DPR paper. The comparing baselines ORQA [127] and REALM [87] use 288-token truncation over Wikipedia, which are not directly comparable to our results. To enable a fair comparison, we report the retrieval results from a recent paper [206], which uses the same retrieval corpus as ours.

4.3.2 Retrieval Performance

We consider three settings to demonstrate the usefulness of our pretrained retriever.

Unsupervised. We assume no annotated training QA pairs are available. In this setting, We compare our method with existing unsupervised retrievers: a sparse retriever BM25 and two pretrained dense retrievers ORQA and REALM.

Domain Adaptation. We consider the condition in which there are QA training pairs in the source domain but no training data in the target domain. The task is to obtain good retrieval performance on the target test set only using source training data. We compare our method with two baselines: one is to directly train a dense retriever on the source domain while the other is to first pretrain a dense retriever on our constructed corpus and then finetune it on the source domain training set.

Supervised. In this setting, all the annotated QA training instances are used. Similar to the previous setting, we compare a supervised retriever with and without our **C-MORE** pretraining.

For all settings, we report the top-k retrieval accuracy ($k \in \{20, 100\}$) on the test set following [113]. See the overall retrieval performance of different models in each setting in Table 4.2. We have the following observations.

In the **unsupervised** setting, compared with the strong sparse retrieval baseline BM25, our pretrained dense retriever shows significant improvement. For example, we obtain around 7% absolute improvement in terms of both Top-20 and Top-100 accuracy on the WebQuestion dataset. Compared with pretrained dense retrievers (i.e., ORQA and REALM), our pretrained model outperforms them by a large margin. This is not surprising as our pretraining data contain better aligned retrieval supervision signals: reference documents often have supporting evidence for the question while their retrieval training signals are relatively indirect.

In the **domain adaptation** and **supervised** settings, our pretrained dense retriever provides a better finetuning initialization and leads to improvement compared with randomly initialized DPR models. Another surprising result is that our pretrained dense retriever even outperforms some DPR domain adaptation models. For example, on the TriviaQA testing set, our pretrained DPR model achieves 72.2% top-20 and 81.3% top-100 accuracy while the DPR-NQ model obtains 69.7% and 79.2% respectively. This indicates that our pretrained dense retriever can generalize well even without using any annotated QA instances.

All the results demonstrate the usefulness and generalization of our pretrained dense retriever for open-domain QA tasks.

4.3.3 End-to-End QA performance

We now examine how our pretrained retriever and reader improve the end-to-end QA performance, measured in exact match (EM). The results are shown in Table 4.3, from which we make the following observations. (i) Surprisingly, our fully-unsupervised system (pretrained retriever + pretrained reader) shows a certain level of open-domain QA ability (see row #3). For example, on TriviaQA, our fully-unsupervised system can answer around 25% of questions correctly. (ii) Compared to the system with BM25 retriever (row #4), the one with our pretrained dense retriever (line #5) retrieves more relevant passages, leading to better QA performance. (iii) Initializing either the retriever or the reader from our pretrained checkpoint can lead to further improvement (rows #6-#8). For example, on the TriviaQA and WebQuestion datasets, our entire pipeline pretrain leads to about 4% absolute gain in terms of EM. Note that on the WebQuestion dataset, all the DPR models perform worse than REALM, this is because of the limited training data of WebQuestion. The issue can be easily solved by adding *Multi* datasets for finetuning according to [113].

4.3.4 Computational Resource Comparison

In addition to the performance gain, another benefit of **C-MORE** is its training scalability. We compare the **C-MORE** pretraining with ORQA and REALM in terms of computational resources they use in Table 4.4. As can be seen, **C-MORE** only requires reasonable GPU

	GPU			TPU		
	#cards	batch size	Train steps	#cards	batch size	Train steps
ORQA	128	4096	100K	-	4096	100K
REALM	240	-	100K	64	512	200K
C-MORE	8	56	20K	-	-	-

Table 4.4: Computational resource comparison between different retriever pretraining methods. Our **C-MORE** provides more direct retrieval pretraining signals, thus leading to fast converge. ORQA and REALM GPU setups are from [206] and TPU setups are from their original papers.

computational resources, which could be normally conducted on an academic-level computational platform. On the contrary, due to the lack of direct retrieval supervision, ORQA and REALM often needs more computational resources and requires more training steps to converge.

4.4 Conclusion

This chapter proposes an effective approach for pretraining open-domain QA systems. Specifically, we automatically construct three million pseudo question-answer-context triplets from Wikipedia that align well with open-domain QA tasks. Extensive experiments show that pretraining a widely-used open-domain QA model (DPR) on our constructed data achieves promising performance gain in both retrieval and QA accuracies. Future work includes exploring the effectiveness of the constructed data on more open-domain QA models (e.g., REALM) and training strategies (e.g., joint optimizing the retriever and reader).

Chapter 5: Automatic Evaluation of Attribution by Large Language Models

The previous chapter addresses synthetic data use for model pre-training. In this chapter, we will talk about how to generate synthetic data for model evaluation.

A recent focus of large language model (LLM) development, as exemplified by generative search engines, is to incorporate external references to generate and support its claims. However, evaluating the attribution, i.e., verifying whether the generated statement is indeed fully supported by the cited reference remains an open problem. Although human evaluation is common practice, it is costly and time-consuming. In this chapter, we investigate the automatic evaluation of attribution by LLMs. We begin by providing a definition of attribution and then explore two approaches for automatic evaluation: prompting LLMs and fine-tuning smaller LMs. The fine-tuning data is repurposed from related tasks, such as question answering, fact-checking, natural language inference, and summarization. To facilitate the evaluation, we manually curate a set of test examples covering 12 domains from a generative search engine, New Bing. Our results on the curated test set and simulated test examples from existing benchmark questions highlight both promising signals as well as remaining challenges for the automatic evaluation of attribution. We hope our testbed, modeling methodology, and insights will help lay the foundation for future studies on this important problem.

5.1 Introduction

Generative large language models (LLMs) [24, 173, 44, 171, 172, *inter alia*] often struggle with producing factually accurate statements, resulting in hallucinations [102]. Recent efforts aim to alleviate this issue by augmenting LLMs with external tools [207] such as retrievers [216, 20] and search engines [162, 231, 217].

Incorporating external references for generation inherently implies that the generated statement is backed by these references. However, the validity of such attribution, i.e., *whether the generated statement is fully supported by the cited reference*, remains questionable.[13] According to Liu et al. [143], only 52% of the statements generated by state-of-the-art generative search engines such as New Bing and PerplexityAI are fully supported by their respective cited references.[14]

Inaccurate attribution compromises the trustworthiness of LLMs, introducing significant safety risks and potential harm. For instance, in healthcare, an LLM might attribute incorrect medical advice to a credible source, potentially leading users to make harmful health decisions. Similarly, in finance, faulty investment advice attributed to a reliable source may cause substantial financial losses.

To identify attribution errors, existing attributed LLMs [162, 231] rely heavily on human evaluation, which is both expensive and time-consuming. For instance, the average cost of annotating a single (query, answer, reference) example is about $1 in Liu et al. [143]. In the actual use of attributed LLMs, it is the user who needs to be wary of the attribution and manually verify it, which puts a tremendous burden on their side. Therefore, effective

[13] *Attribution* primarily refers to "the act of attributing something" in this chapter, which is similar to "verifiability" as defined in Liu et al. [143].

[14] `www.bing.com/new`, `www.perplexity.ai`

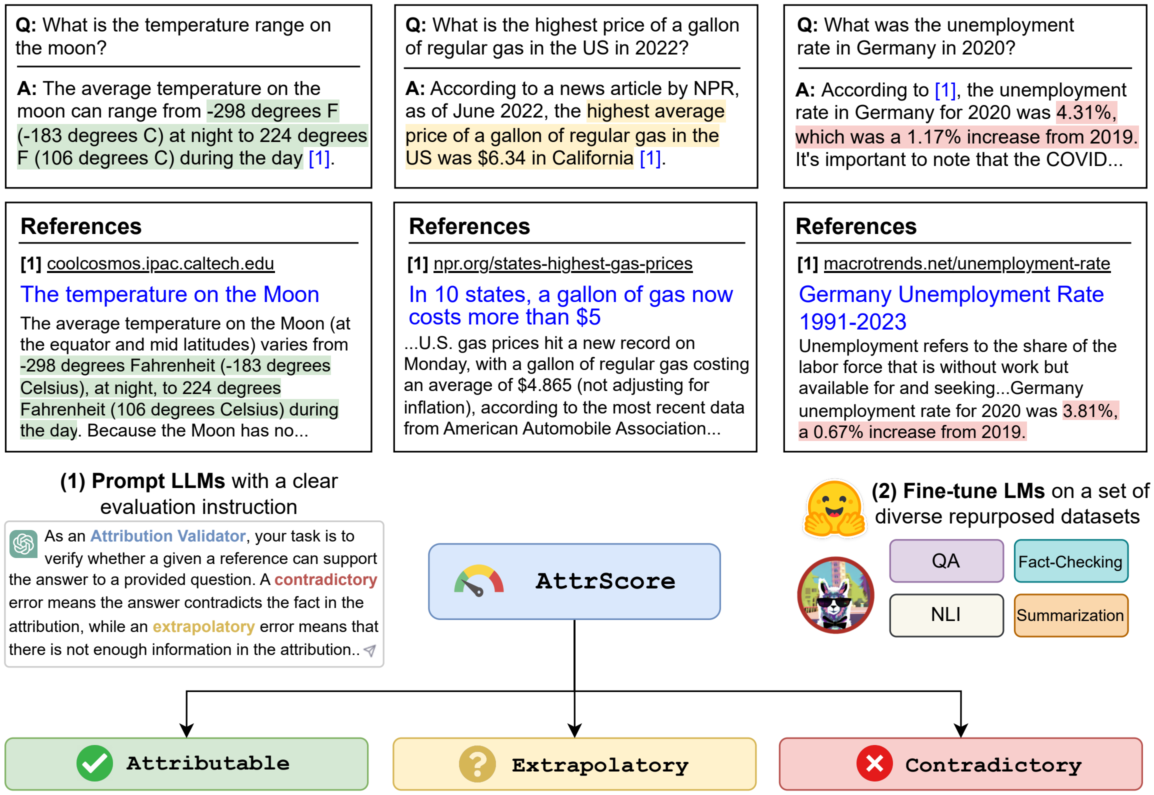


Figure 5.1: We make the first step towards automatically evaluating attribution and identifying specific types of errors with AttributionLens. We explore two approaches in AttributionLens: (1) prompting LLMs, and (2) fine-tuning LMs on simulated and repurposed datasets from related tasks.

and reliable methods to automatically evaluate attribution and identify potential attribution errors are highly desired.

Towards this goal, we take the first step by introducing *AttributionLens* (Figure 5.1), a framework designed for automatic evaluation of attribution and identification of specific types of attribution errors. We propose a new problem formulation that categorizes attribution into three types: 1) *attributable*: the reference fully supports the generated statement; 2) *extrapolatory*: the reference lacks sufficient information to support the generated statement, and 3) *contradictory*: the generated statement directly contradicts the cited reference. Unlike existing work [17] that uses binary categorization (i.e., attributable or not) and Liu et al. [143] that defines the degree of reference support for the generated statement as “full”, “partial”, or “no support”, our fine-grained error categorization aids humans in better understanding the type of an attribution error made by an LLM. This not only enhances safe system usage but also provides valuable insights for future development of mechanisms tailored to correct specific errors.

We explore two approaches in AttributionLens: 1) prompting LLMs and 2) fine-tuning LMs on simulated and repurposed data from related tasks such as question answering (QA), fact-checking, natural language inference (NLI), and summarization. For evaluation, unlike existing work [143, 76] that only uses queries from existing benchmarks, we curate a set of test examples covering 12 different domains from a generative search engine, New Bing. This is the first evaluation set for measuring the attribution of LLMs with queries created based on real-life interactions, hence avoiding the data contamination issue.

Our results indicate that both approaches show reasonable performance on our curated and simulated test sets; yet there is still substantial room for further improvement. Major sources of evaluation failures include insensitivity to fine-grained information comparisons, such as overlooking contextual cues in the reference, disregard for numerical values, and failure in performing symbolic operations. In light of these findings, we discuss potential directions for improving AttributionLens, including training models to be more strongly conditioned on the reference, and augmenting them with external tools for numerical and logical operations.

With the new formulation of attribution errors, the development of AttributionLens, the introduction of new test sets, and the insights into challenges and potential directions for future work, we hope our work can help lay the foundation for the important task of automatically evaluating LLM attributions.

5.2 Problem Formulation

The primary task in this chapter is to evaluate attribution, which involves verifying whether a reference provides sufficient support for a generated answer to a user's query. Our task setting prioritizes one reference per statement, a unit task that more complex scenarios can be decomposed to. We study such a setting as it forms the basis for dealing with multiple references or distinct segments [143, 76].

Prior work, such as Rashkin et al. [197], Gao et al. [75], Bohnet et al. [17], mainly focuses on binary verification, i.e., determining if a reference supports the generated answer or not. We propose advancing this task by introducing a more fine-grained categorization. Specifically, we classify attributions into three distinct categories:[15]

- **Attributable**: The reference fully supports the generated answer.
- **Extrapolatory**: The reference lacks sufficient information to validate the generated answer.
- **Contradictory**: The generated answer contradicts the information presented in the reference.

To illustrate, consider a contradictory example (Figure 5.1). The query is *"What was the unemployment rate in Germany in 2020?"*, and the generated answer is *"4.31%"*. However, the reference states that the rate was *"3.81%"*, contradicting the generated answer.

[15]We acknowledge that while these categories are generally mutually exclusive, complex scenarios might blur the boundaries between them. However, such cases are very rare. For the purpose of this study, we maintain their exclusivity to enable clear and focused error analysis.

An extrapolatory instance, on the other hand, would be a query about the *"gas price in California"*. While the reference is relevant, it does not contain specific information to verify the correctness of the generated answer.

Following these examples, we see the importance of granularity in error classification. A fine-grained classification allows us to pinpoint the nature of the errors, be it contradiction or extrapolation. Users can better understand the type of errors an LLM might make, enabling them to use the model more safely. Additionally, such an error identification system can guide future training processes of attributed LLMs, leading to specific mechanisms' development to correct such errors.

Our categorization also offers a departure from the existing approach [143], which emphasizes on degree of support ("full", "partial", or "none") rather than attribution error types. Our approach highlights specific issues in attribution evaluation for more effective error management and system improvement.

Formally, the task of attribution evaluation involves a natural language query q, a generated answer a, and a reference x from an attributed LLM. The goal is to develop a function, denoted as f, that inputs (q, a, x) and outputs a class label indicating whether "according to x, the answer a to the query q is attributable, extrapolatory or contradictory."[16]

5.3 Automatic Evaluation of Attribution

Following our problem definition, we introduce two approaches for automatic evaluation of attribution: prompting LLMs and fine-tuning LMs on simulated and repurposed data from related tasks.

[16]It is important to note that this evaluation focuses on the "verifiability" of the answer based on the reference. It does not measure the "relevance", i.e., whether the answer correctly responds to the query [143].

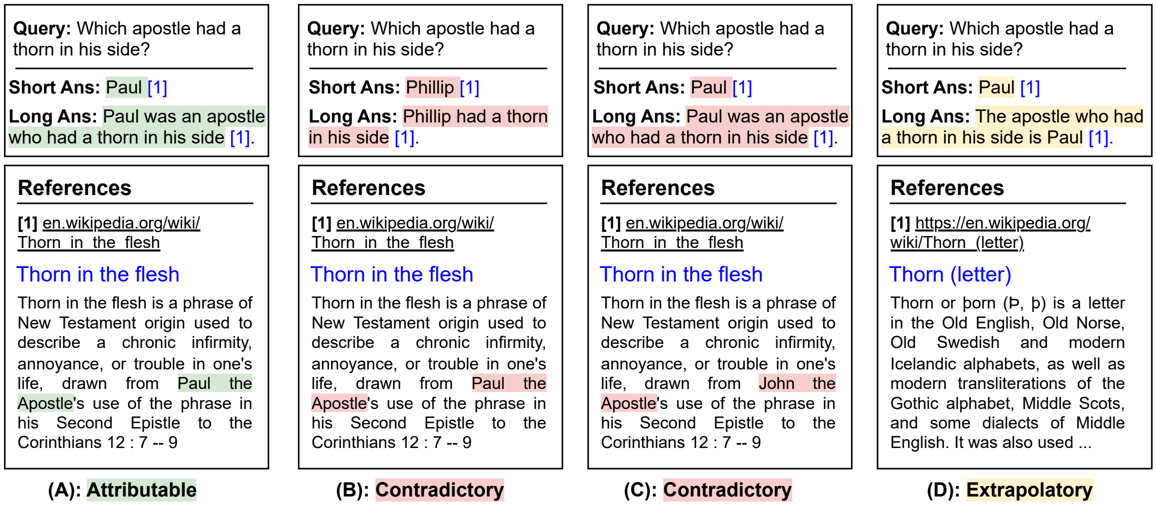


Figure 5.2: Examples simulated from open-domain QA. We 1) use the original (question, answer, context) pair as an *attributable* instance **(A)**, 2) substitute the answer or the answer span in the context to simulate a *contradictory* error example **(B, C)**, and 3) replace the context with alternatives to simulate an *extrapolatory* error example **(D)**. In order for models trained the simulated data to generalize well to the long answer setting in real-life search engines like New Bing, we convert the short answer to a long one (using ChatGPT).

5.3.1 Prompting LLMs

Recent research [73] has demonstrated the possibility of prompting LLMs to evaluate the quality of generated text using their emergent capabilities [253], such as zero-shot instruction [252] and in-context learning [24]. Following this approach, we prompt LLMs, such as ChatGPT [171], using a clear instruction that includes definitions of the two types of errors (as shown in Figure 5.1) and an input triple of the query, answer, and reference for evaluation. The complete prompt used in our study can be found in Appendix Table D.2.

5.3.2 Fine-tuning LMs on Repurposed Data

The primary challenge in fine-tuning LMs for automatic attribution evaluation is the lack of training data. One potential approach is to hire annotators to collect real samples, but the cost can be prohibitive.

Here, we first repurpose datasets from three related tasks (fact-checking, NLI, and summarization). We then propose to further simulate more realistic samples from existing QA benchmarks.

Repurpose data from fact-checking, NLI, and summarization tasks. Given the connections between our attribution evaluation task and the tasks of fact-checking, NLI, and summarization, we propose to utilize datasets from these fields to enrich our training examples. Fact-checking data and NLI data, with their emphasis on assessing the consistency and logical relationship between claims (hypothesis) and evidence (premise), mirrors our task's objective of checking the supporting relationship between reference documents and generated statements. Summarization datasets, especially those involving the detection of hallucinations (including both intrinsic and extrinsic [151], could provide a useful starting point for identifying attribution inconsistencies. Nevertheless, these datasets would require suitable adaptation. We keep their original data sequences and modify their data label space to suit the specific needs of the attribution evaluation definition. Additional information on this can be found in Appendix D.1.

Simulate data from open-domain QA. QA benchmarks provide an ideal platform for data simulation, as they comprise questions, their corresponding ground truth answers, and reference contexts. These elements can be directly employed as *attributable* examples (Figure 5.2, **A**). In open-domain QA datasets, answers are typically brief text spans. To cater to the long answer setting in most attributed LLMs, we convert these short answers into longer sentences using ChatGPT. For simulating *contradictory* errors, we propose two methods: (1) The first involves modifying the correct answer with an alternative candidate from an off-the-shelf QA model, an answer substitution model, or a random span generator (Figure 5.2, **B**). (2) The second retains the original answer but replaces the answer span in the reference context with a comparable candidate (Figure 5.2, **C**). To emulate *extrapolatory* errors, we employ a BM25 retriever on the question, retrieving relevant external documents from resources such as Wikipedia, which do not contain the ground truth answers (Figure 5.2, **D**). More details regarding the simulation of these errors from QA datasets can be found in Appendix D.1.

5.4 Experimental Setup

5.4.1 Datasets

This section presents the datasets utilized for training and testing methods for automatic attribution evaluation. In particular, we develop two evaluation sets, *AttrEval-Simulation* and *AttrEval-GenSearch*, derived from existing QA datasets and a generative search engine, respectively. The dataset statistics are presented in Table 5.1.

Training data. To repurpose and simulate training examples, we follow the method in Section 5.3.2 based on four similar tasks' datasets. For *QA*, we consider NaturalQuestions [122]. For *fact-checking*, we include FEVER [232], Adversarial FEVER [233], FEVEROUS [6], VITAMINC [208], MultiFC [10], PubHealth [118], and SciFact [242]. For *NLI*, we include SNLI [23], MultiNLI [256], ANLI [170] and SciTail [115]. For *summarization*, we include XSum-Halluc. [151], XENT [26], and FactCC [120]. We use all examples in the summarization task datasets, and sample 20K examples from QA, fact-checking, and NLI

Split	Related Tasks	Data Sources	#Samples
Train	QA	NaturalQuestions	20K
	Fact-checking	FEVER, VITAMINC, Adversarial FEVER, FEVEROUS, SciFact PubHealth, MultiFC	20K
	NLI	SNLI, MultiNLI ANLI, SciTail	20K
	Summarization	XSum-Hallucinations, XENT, FactCC	3.8K
Test	QA	PopQA, EntityQuestions, HotpotQA, TriviaQA, WebQuestions, TREC	4K
	-	Annotated samples from a generative search engine	242

Table 5.1: Statistics of the training and test datasets for attribution evaluation. We include the distributions of the labels and data sources in Appendix D.2.

task datasets. We combine all the simulated datasets to create the training set for our main experiment.

AttrEval-Simulation. For testing, we first simulate examples from six out-of-domain QA datasets: HotpotQA [272], EntityQuestions [209], PopQA [150], TREC [13], TriviaQA [108], and WebQuestions [14]. Note that we intend to use different QA datasets for training and testing, as to test the model's generalization ability, and evaluate its performance across a diverse set of domains and question formats. Our manual examination indicates that 84% of 50 randomly sampled examples accurately align with their category, and the labeling errors are primarily due to incorrect annotations in the original QA datasets or heuristics used to formulate comparable answer candidates for contradictory errors and to retrieve negative passages for extrapolatory errors.

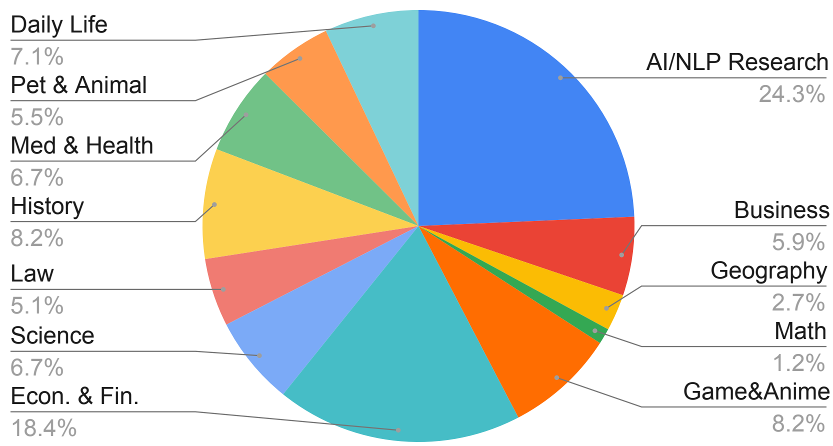


Figure 5.3: Domain distribution of our annotated AttrEval-GenSearch test set (covering 12 domains in total).

AttrEval-GenSearch. To examine the real-life application of automatic attribution evaluation, approximately 250 examples from the New Bing search engine are annotated carefully by the authors. This process comprises two subtasks: creating queries and verifying attributions. To avoid the issue of training data contamination, new queries are manually created across 12 domains (Figure 5.3).[17] To facilitate and motivate query annotation, keywords from a specific domain are randomly generated using ChatGPT, and relevant facts within that domain are compiled from the Web.[18]

In the verification process, queries are sent to the New Bing search engine under a balanced mode following Liu et al. [143], which balances accuracy and creativity. The validity of the output generated by New Bing is evaluated, where we consider only the first sentence that answers the question along with its reference. As we state in Section 5.2, our evaluation emphasizes the error type in a single reference per statement. In the case of a

[17] The "AI/NLP Research" domain is inspired by recent discussions on social media about testing LLMs' knowledge on researchers, e.g., *"Is XX a co-author of the chapter XX?"*

[18] We make an effort to collect new facts post-2021 to test about "knowledge confliction" [290, 266] between parametric and external knowledge.

Setting	Model (Size)	AttrEval-Simulation				AttrEval-GenSearch			
		Attri.	Contra.	Extra.	Overall	Attr.	Contra.	Extra.	Overall
Zero-shot	Alpaca (7B)	50.0	4.0	1.4	33.6	50.7	8.6	3.6	34.3
	Alpaca (13B)	48.3	5.6	2.2	33.5	50.6	6.1	19.3	34.7
	Vicuna (13B)	46.3	8.3	21.6	34.6	54.4	13.3	26.1	41.4
	ChatGPT	45.7	17.9	52.7	43.2	61.2	20.6	53.3	55.0
	GPT-4	**58.7**	**23.2**	**61.5**	**55.6**	**87.3**	**45.0**	**89.6**	**85.1**
Few-shot	Alpaca (7B)	45.4	8.2	9.6	31.9	49.6	5.2	13.5	37.2
	Alpaca (13B)	38.9	20.1	2.2	33.1	50.5	10.3	5.6	34.8
	Vicuna (13B)	35.4	**37.2**	0.3	32.6	50.6	9.1	8.4	34.1
	ChatGPT	46.6	27.6	35.8	39.2	62.6	26.8	49.5	53.3
	GPT-4	**61.1**	31.3	**68.8**	**60.0**	**85.2**	**53.3**	**88.9**	**84.3**
Fine-tuned	Roberta (330M)	62.5	54.6	74.7	65.0	47.2	25.2	62.3	49.8
	GPT2 (1.5B)	63.6	54.6	71.9	63.5	51.1	18.6	60.7	47.4
	T5 (770M)	45.9	**57.1**	71.6	59.1	58.5	24.3	72.5	61.6
	Flan-T5 (770M)	57.3	50.1	70.5	59.3	64.3	27.6	72.9	64.5
	Flan-T5 (3B)	48.1	48.7	67.1	55.7	77.7	**44.4**	**80.0**	**75.2**
	Flan-T5 (11B)	48.4	49.9	66.5	55.4	**81.6**	38.9	76.9	72.7
	LLaMA (7B)	62.2	50.7	74.6	62.8	77.9	41.1	78.3	72.5
	Alpaca (7B)	**66.8**	41.1	76.8	64.5	73.0	30.2	**80.0**	72.5
	Alpaca (13B)	63.6	48.9	75.8	63.6	77.5	34.5	79.4	73.3
	Vicuna (13B)	66.2	49.1	**78.6**	**66.0**	69.4	37.7	79.9	72.1

Table 5.2: The performance (F1 score) of AttributionLens with different models on AttrEval-Simulation and AttrEval-GenSearch sets. The best-performing result in each setting is in **bold**. The results show both promising signals and challenges (e.g., all models struggle with contradictory errors) in automatic evaluation of attribution.

sentence having multiple references or distinct segments (for example, “XXX [1][2]” or “XXX [1] and YYY [2]”), each reference or segment is treated as a separate sample, and the attributions are verified individually. Finally, the samples are categorized by the annotators as attributable, contradictory, or extrapolatory. Detailed annotation guidelines can be found in Appendix D.4.

5.4.2 Implementation Details

In the configuration of "prompting LLMs", we test Alpaca [230], Vicuna [41], ChatGPT [171] and GPT-4 [172], where we use OpenAI's official APIs (`gpt-3.5-turbo`, `gpt-4-0314`)[19], and weights from Alpaca and Vicuna from the official repository[20]. For Alpaca and Vicuna inference, documents are tokenized and truncated at a maximum of 2048 tokens. We generate text with a temperature of 0. The prompts for the task of evaluating attribution are provided in Appendix Table D.2, and our main results are averaged over 4 different prompts. For the few-shot prompting setting, we manually write 3 examples as demonstrations for both test sets as shown in Table D.3. If LLMs yield an attribution label with an explanation, we extract the predicted label with regular expression.

In the "fine-tuning LMs" setting, we fine-tune four types of LMs of various scales: Roberta (340M) [145], (FLAN-)T5 (770M, 3B, 11B) [191, 45], GPT2 (1.5B) [189], LLaMA (7B), Alpaca (7B, 11B) [230], and Vicuna (7B, 11B) [41]. Our implementation utilizes the Huggingface library [259] and Alpaca examples. The training is performed on 4 A100 80GB GPUs with a maximum of 512 tokens. For the LLaMA family of models, we use a batch size of 32 and train for 1 epoch. For the other models, we use a batch size of 64 and 3 epochs. We set the learning rate as 2e-5 and use a cosine learning rate decay with 0.03 warm-up steps.

Metrics. For evaluation, we present the F1 score for each individual class as well as the micro-F1 score, which is equivelant to the overall accuracy.

[19] `platform.openai.com/docs/api-reference/chat`. Given GPT-4's high cost and slow inference speed, we evaluate it on 500 random samples from AttrEval-Simulation.

[20] `https://github.com/tatsu-lab/stanford_alpaca`, `https://github.com/lm-sys/FastChat`

5.5 Results

5.5.1 Overall Performance

Table 5.2 presents an evaluation of different models on both the simulated dataset (AttrEval-Simulation) and the annotated dataset on New Bing (AttrEval-GenSearch). Our primary findings are as follows:

GPT-4 achieves promising results, reaching an overall accuracy of 84-85% on AttrEval-GenSearch and significantly outperforming other models. This suggests a promising potential for employing GPT-4 for automatic attribution evaluation to alleviate human annotation workloads, aligning with the emerging trend that uses GPT-4 for different evaluation tasks [41, 288]. However, it may still not be sufficiently accurate for practical use. We also note some potential concerns of bias (see Limitations Section 5.8).

Automatic attribution evaluation presents substantial challenges. This complex task requires not only understanding the reference information but also comparing it with the information in the statement, all of which can significantly vary across different datasets and test conditions. Against these challenges, models other than GPT-4 exhibit suboptimal performance in zero-shot and few-shot settings. Fine-tuning LMs on the simulated datasets from related tasks significantly improves the performance. For instance, the Vicuna (13B) model sees the overall accuracy on the two test sets rise from 34.6% and 41.4% in the zero-shot setting to 66.0% and 71.3%, respectively. And the fine-tuned FLAN-T5 (770M) model can even surpass ChatGPT on both test sets. Despite this, there is still a large room for further improvement. Some models that yielded better results on the simulated test set may be less effective on the annotated test set, indicating a lack of consistency across diverse testing settings, signaling generalizability challenges.

Error Type (percentage)	Example
Fine-grained information insensitivity (30.6%)	**Query:** Was ketchup a medicine before? **Answer:**: In the early 1800s, it was believed that ketchup could be used to treat indigestion, jaundice, and diarrhea **Reference:** According to John Cook Bennett, tomatoes could be used to treat diarrhea, ... Soon, Bennett was publishing tomato ketchup recipes in pill form ... According to Dr. Bennett, this tomato blend can help people with diarrhea, indigestion, rheumatism, and jaundice. ▷ **Ground Truth:** Extrapolatory ▷ **GPT-4:** Attributable. The reference supports the claim that in the early 1800s, ketchup was believed to be used as a medicine to treat indigestion, jaundice, and diarrhea. It mentions John Cook Bennett's belief in the medicinal properties of tomatoes and his publication of tomato ketchup recipes in pill form, which were sold as patent medicines in supermarkets and pharmacies.
Misunderstanding the task definition and logical relation implied by labels (22.2%)	**Query:** When GTA6 will be published? **Answer:** Rockstar Games has not officially announced the release date for GTA 6 yet. **Reference:** The most likely GTA 6 release date is holiday 2024, and although there is no official confirmation, this estimate is confirmed by Microsoft's stating that it "is expected" in 2024 in a legal filing and assorted insider leaks. ▷ **Ground Truth:** Attributable ▷ **GPT-4:** Extrapolatory. The reference provides an estimated release date for GTA 6 based on a legal filing from Microsoft and insider leaks, but it does not confirm an official announcement from Rockstar Games regarding the release date.
Failing on symbolic operators (13.9%)	**Query:** What's the average salary for a software engineer working at Amazon in 2022? **Answer:** According to Indeed.com, the average salary for a software engineer working at Amazon in the United States is $131,930 per year. **Reference:** Average salary $132,147 Salary estimated from 3,612 employees, users, and past and present job advertisements on Indeed in the past 12 months. Last updated: April 18, 2023. ▷ **Ground Truth:** Contradictory ▷ **GPT-4:** Attributable. The reference does provide an average salary of $132,147 for a software engineer working at Amazon, which is very close to the claim's figure of $131,930.

Table 5.3: Qualitative analysis on misclassified examples from prompting GPT-4 in AttributionLens.

Models struggle most notably with contradictory errors. Detecting contradictions is particularly complex because it requires the model to weigh one piece of information in the statement against another in the reference, a process that necessitates advanced fine-grained information comparison and reasoning capabilities. Consequently, even the best-performing model GPT-4 and the fine-tuned models often fail when faced with contradictory inputs, most often treating them as attributable (see qualitative analysis in Section 5.5.2).

5.5.2 Qualitative Analysis

To shed light on the space for future improvements in attribution evaluation, we qualitatively examine all the error examples of GPT-4 in the zero-shot setting. Representative examples are in Table 5.3.

Our first observation is that a significant portion (30.6%) of errors happen due to fine-grained information insensitivity: failure in comparing very fine-grained information such as numerical values, numbers, dates, and time. Besides, the model misunderstands task definition and misinterprets logical relations implied by labels (22.2%). The model also struggles with symbolic operators (13.9%). For example, it fails to distinguish 'equal to' (=) and 'approximately equal to' (≈) in numeric comparisons. In the left cases, the model tends to overlook the context clues and does not make judgments by conditioning on the reference (e.g., potentially relying on its own parametric knowledge).

Our observations point to two potential directions for improvement: 1) training or prompting models to be more faithful and strongly conditioned on the reference [290], especially paying attention to fine-grained information; and 2) augmenting an LM-based evaluation method with external tools for different types of numerical and logical operations that are hard to be accurately performed only by the LM itself [39, 157]. Similarly, we do qualitative analysis for ChatGPT in Appendix Section D.5.

5.5.3 Ablation Study

In this section, we perform an ablation study to test how each task influences the fine-tuned LMs' results and analyze the prompt sensitivity in zero-shot and few-shot settings for prompting LLMs.

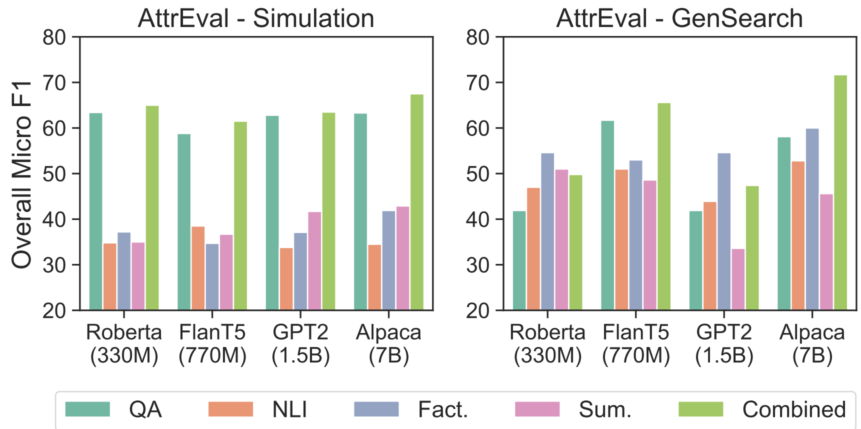


Figure 5.4: The influence of individual task data. Combining datasets generally improves model performance.

Contribution of individual task. We show the performance of models fine-tuned on individual task datasets and their combinations in Figure 5.4. We select a representative from each group of the models under the fine-tuned setting in Table 5.2. Our findings suggest that examples from our simulated QA and fact-checking task most significantly improve performance for the attribution evaluation task, hinting at a strong link between these tasks. Furthermore, integrating various related task datasets generally leads to better performance, particularly on out-of-domain test instances in AttrEval-GenSearch.

Sensitivity of prompts. The choice of prompts used to evaluate language models can have an impact on their performance. We evaluate the sensitivity of prompts for AttributionLens under both zero-shot and few-shot settings of Alpaca (7B) and ChatGPT. We show four types of prompts as mentioned earlier: a prompt designed specifically for our evaluation setting (Attri.), an NLI prompt, a fact-checking prompt (Fact.), and a summarization hallucination detection prompt (Sum.). These prompts are presented in Appendix Table D.2. As shown in Table 5.4, fact-checking and NLI prompts generally perform better, as similar tasks may have been seen during their instruction tuning phase.

5.6 Related Work

Attributed LMs. Generative LMs often produce hallucinations [151, 62, 126, 216, 245, 265, 102]. To alleviate the issue, recent work proposes to augment LLMs [157] with external tools [207, 134, 185] such as retrievers [88, 132, 216, 99, 100, 20, 236, 184] and search engines [162, 117, 231, 274, 80, 217, 178]. Incorporating external references for generation inherently implies that the generated statement is backed by these references. However, the validity of such attribution remains questionable.

Evaluation of attribution. To evaluate attribution, Liu et al. [143] conduct a human evaluation to audit the verifiability of responses from generative search engines. They find that these engines frequently contain unsupported statements and inaccurate citations, which strengthen the need to carefully examine the attribution of generations [197]. However, human evaluations are very expensive and time-consuming. Gao et al. [75], Bohnet et al. [17], Gao et al. [76] propose to automatically evaluate attribution by levering NLI models [96, 110, 77]. We study this problem in a more comprehensive and realistic manner: 1) we explore how helpful other relevant tasks besides NLI are to attribution evaluation; 2) our evaluation setting is based on both benchmark examples and real examples.

5.7 Conclusion

In this chapter, we investigate the important problem of automatically evaluating attribution given by LLMs. We begin by defining different types of attribution errors and then explore two approaches for automatic evaluation: prompting LLMs and fine-tuning smaller LMs. We experiment with both simulated test examples and manually curated test examples from a real-life generative search engine. The results highlight both promising signals and

Models	Task Prompts	Zero-shot		Few-shot	
		Sim.	Gen.	Sim.	Gen.
Alpaca	Attr.	34.8	34.4	31.3	33.5
	NLI	32.1	35.5	32.1	33.6
	Fact.	34.0	33.9	32.7	46.7
	Sum.	33.6	33.5	31.6	34.8
	Average	33.6	34.3	31.9	37.2
ChatGPT	Attr.	37.2	45.1	37.6	51.4
	NLI	45.0	61.7	35.8	56.1
	Fact.	44.8	54.9	43.2	54.9
	Sum.	45.6	58.1	40.2	50.6
	Average	43.2	55.0	39.2	53.3

Table 5.4: Sensitivity of prompts for prompting LLMs on AttrEval-Simulation (Sim.) and -GenSearch (Gen.). The prompts include a prompt for attribution (Attri.), a NLI prompt, a fact-checking prompt (Fact.), and a summarization hallucination detection prompt (Sum.).

remaining challenges for the automatic evaluation of attribution. We hope our work could lay the foundation for future studies on this important problem.

5.8 Limitations

Currently, smaller models in AttributionLens are fine-tuned on the combination of simulated or repurposed datasets from related tasks. However, this dataset still has gaps from the real scenario. Moreover, the error patterns in these simulated datasets might be overly simplistic and lack diversity, which can limit the models' ability to effectively handle more complex and varied real-world errors. It is also worth noting that these simulated datasets may contain noise and erroneous labels, which could further impede the models' learning and subsequent performance. How to obtain higher-quality training data for attribution evaluation at scale can be a major focus for future development.

Our annotated evaluation set, AttrEval-GenSearch, is derived from New Bing, which uses GPT-4 as its backbone. It is crucial to note that we also use GPT-4 for evaluating attribution on AttrEval-GenSearch, which achieves the best performance with around 85% overall accuracy. Some bias might come from GPT-4 both generating the test examples and evaluating the attribution, which could potentially skew our understanding of the model's true performance. We therefore caution against over-optimism. We also acknowledge that the size of AttrEval-GenSearch is moderate, which may not fully represent the real use setting of attributed LLMs.

While acknowledging current limitations, several promising directions emerge for future research and enhancement. For example, one can diversify data sources to include examples from a variety of generative search engines, not just New Bing. In addition, it may be beneficial to annotate larger-scale queries that cover a broad spectrum of topics, styles, and perspectives.

5.9 Future Work

Several promising directions exist for advancing attribution evaluation capabilities:

Connecting to Explainable Fact-Checking. Recent work on fact-checking [241, 64] focuses on generating natural language explanations to justify fact-checking decisions. For example, Vedula and Parthasarathy [241] construct relevant knowledge graphs and retrieves supporting textual context to explain veracity predictions. Such explanation generation techniques are highly relevant for attribution evaluation. Future work may adapt such explanation fack-checking methods to provide explanations justifying attribution decisions and evaluate their effectiveness for this task. Moreover, FactKeg shows improved performance when focusing on structured domains - this suggests attribution evaluation may also benefit from domain-specific datasets where models can learn specialized reasoning patterns. Future attribution benchmarks should be developed for domains like finance and healthcare where claims are more regularized.

Incorporating Structured Knowledge External knowledge bases (KBs) and knowledge graphs (KGs) can provide structured facts to verify attributions. Future work could consider developing domain-specific KBs in areas like medicine, finance, and science to provide rich specialized knowledge. New models could be developed to effectively integrate these structured knowledge stores, enabling more reliable reasoning. Comparative benchmarking of models conditioned on just text versus text + KBs will shed light on their complementary benefits.

Handling Scientific Uncertainty Capturing uncertainty poses challenges for attribution where evidence may be limited or conflicting. Future models should identify uncertain attributions and abstain when evidence is inconclusive rather than making spurious claims. Uncertainty metrics should be developed to capture reliability. New evaluation metrics are also needed to assess calibration and reliability as evidence becomes more ambiguous.

Targeted future work in this space also includes: integrating numerical reasoning modules to address common model failures, developing better prompts to force comparison between statements and references, and benchmarking model architectures augmented with retrievers to reason over multiple references. By pursuing these research avenues, we aim to ultimately develop models capable of robust, trustworthy attribution evaluation across diverse real-world applications.

5.10 Ethics Statement

This research project involves evaluating attribution given by attributed LLMs. We collect and annotate data for evaluation using publicly available information on the web, with the assistance of a generative search engine, New Bing. We acknowledge that LLMs have the potential to reproduce and amplify harmful information present in the data. We made an effort to mitigate this risk by carefully selecting our evaluation data and by conducting analyses to identify and mitigate potential risks in the process.

Part IV: Data Synthesis for Domain Adaptation

In Part III, we discussed how to leverage synthetic data for model training. After training a model, one needs to adapt and deploy it to the target environment for real use. The target testing set may have different distributions from the source training data in the second stage. In this part, we will discuss how to utilize synthetic data for domain adaptation in the Model Deployment stage. We will focus on Machine Reading Comprehension. In Chapter 6, we first propose a question generator to synthesize *diverse* questions for clinical machine reading comprehension. We will show that the synthesized QA corpus can be greatly useful to boost the QA performance on the target domain. Due to the noisy nature of synthetic data, in Chapter 7, we further propose a question value estimator to select the most high-quality and useful synthetic QA pairs for improving the QA performance on the target domain.

Chapter 6: CliniQG4QA: Generating Diverse Questions for Domain Adaptation of Clinical Question Answering

Clinical question answering (QA) aims to automatically answer questions from medical professionals based on clinical texts. Studies show that neural QA models trained on one corpus may not generalize well to new clinical texts from a different institute or a different patient group, where large-scale QA pairs are not readily available for model retraining. To address this challenge, we propose a simple yet effective framework, **QVE**, which leverages question generation (QG) to synthesize QA pairs on new clinical contexts and boosts QA models without requiring manual annotations. In order to generate diverse types of questions that are essential for training QA models, we further introduce a seq2seq-based question phrase prediction (QPP) module that can be used together with most existing QG models to diversify the generation. Our comprehensive experiment results show that the QA corpus generated by our framework can improve QA models on the new contexts (up to 8% absolute gain in terms of Exact Match), and that the QPP module plays a crucial role in achieving the gain.

6.1 Introduction

Clinical question answering (QA), which aims to automatically answer natural language questions based on clinical texts in Electronic Medical Records (EMR), has been identified as an important task to assist clinical practitioners [177, 192, 174, 68, 198]. Neural QA models in recent years [37, 51, 198] show promising results in this research. However, answering clinical questions still remains challenging in real-world scenarios, because well-trained QA systems may not generalize well to new clinical contexts from a different institute or a different patient group. For example, as pointed out in [278], when a clinical QA model that was trained on the emrQA dataset [174] is deployed to answer questions based on MIMIC-III clinical texts [104], its performance drops dramatically by around 30% even on the questions that are similar to those in training, simply because clinical texts of the two datasets are different (e.g., different topics, note structures, writing styles).

One straightforward solution is to annotate QA pairs on new contexts and retrain a QA model. However, manually creating large-scale QA pairs in clinical domain is extremely challenging due to the requirement of tremendous expert effort, data privacy concerns and other ethical issues.

In this work, we study the problem of *constructing clinical QA models on new contexts without human-annotated QA pairs* (which is referred to as domain adaptation). We assume the availability of a large set of QA pairs on *source* contexts, and our goal is to better answer questions on new documents (*target* contexts[21]), where only unlabeled documents are provided.

To this end, we introduce our framework, **QVE**, which leverages question generation (QG), a recent technique of automatically generating questions from given contexts [55], to synthesize clinical QA pairs on target contexts to facilitate the QA model training (Figure 6.2). The QG model is built up by reusing the QA pairs on source contexts as training data. To apply QG to target contexts, our framework also includes an *answer*

[21]We use “new" and “target" contexts interchangeably.

evidence extractor (AEE) to extract meaningful text spans, which are worthwhile to ask questions about, from the clinical documents. Intrinsically, our framework is backed by the observation that questions in the clinical domain generally follow similar patterns even across different contexts, and clinical QG suffers less from the context shift compared with clinical QA. This allows us to utilize QG models trained on source clinical contexts to boost QA models on target contexts.

However, our preliminary studies find that many existing QG models often fall short on generating questions that are *diverse* enough to serve as useful training data for clinical QA models. To tackle the problem, we introduce a *question phrase prediction* (QPP) module, which takes an answer evidence as input and sequentially predicts potential question phrases (e.g., "What treatment", "How often") that signify what types of questions humans would likely ask about the answer evidence. By directly forcing a QG model to produce specified question phrases in the beginning of the question generation process (both in training and inference), QPP enables diverse questions to be generated.

Due to the lack of publicly-available clinical QA pairs for our proposed domain adaptation evaluation setting, we ask clinical experts to annotate a new test set on the sampled MIMIC-III [104] clinical texts. We conduct extensive experiments to evaluate **QVE**, using emrQA [174] as the source contexts and our annotated MIMIC-III [104] as the target ones. We instantiate our framework with a variety of widely adopted base QG models and base QA models.

By performing comprehensive analyses, we show that the proposed QPP module can substantially help generate much more diverse types of questions (e.g., "When" and "Why" questions). More importantly, we systematically demonstrate the strong capability of **QVE** for improving QA performance on new contexts by evaluating it on our constructed MIMIC-III

QA dataset. When using QA pairs automatically synthesized by our QPP-enhanced QG models as the training corpus, we are able to boost QA models' performance by up to 8% in terms of Exact Match (EM), compared with their counterparts directly trained on the emrQA dataset. To further investigate why QG boosts QA, we provide both quantitative and qualitative analyses, indicating that QA models can benefit from seeing more target contexts as well as more diverse questions generated on them.

6.2 Preliminary and Related Work

Clinical Question Answering aims to extract a text span (a sentence or multiple sentences) as the answer from a patient clinical note given a question (Fig. 6.1 left) [278]. Though many neural models [211, 37, 51, 198, 254] have achieved impressive results on this task, their performance on new clinical contexts, whose data distributions could be different from the ones that these models were trained on, is still far from satisfactory [278]. Though one can improve the performance by adding more QA pairs on new contexts into training, however, manually creating large-scale QA pairs in the clinical domain often involves tremendous expert effort and data privacy concerns. Moreover, during the pandemic, clinical QA models can also be deployed to answer COVID-19 related questions [182, 285].

Question Generation seeks to automatically generate questions given a sentence or paragraph (Fig. 6.1 right). Existing QG models [55, 289, 226, 287, 164, 237, 271, 54, 3, 284] in the open domain usually adopt a seq2seq (encoder-decoder) architecture. One of the drawback of such models is that they can only generate one question given one input and fail to generate multiple diverse questions, which we find is crucial to the QA task. Some recent work [111, 42, 141] explores the diverse QG in the open domain, but they cannot be directly applied to the clinical domain as their models usually require a short answer (e.g.,

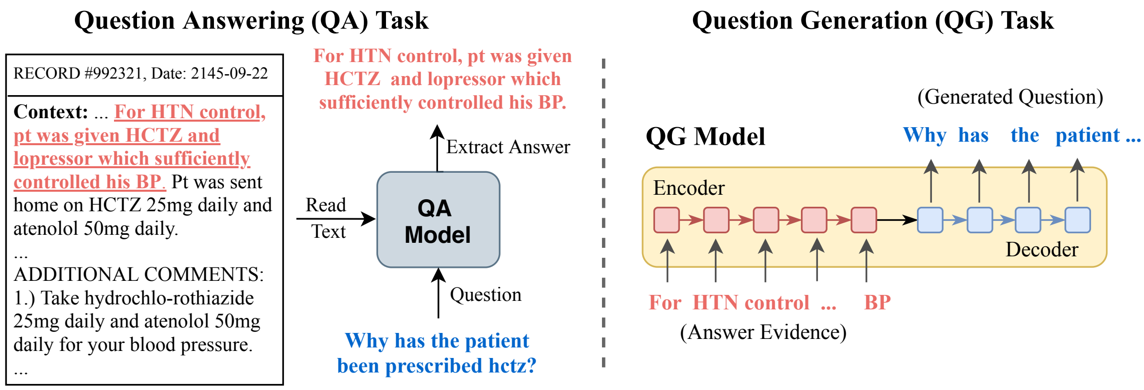


Figure 6.1: Illustration of Clinical Question Answering (QA) and Question Generation (QG) task.

an entity) as input but that information sometimes is not available in the clinical QA dataset (e.g. emrQA [174]), rendering the difficulty of directly deploying their model on the clinical QA.

In the clinical and medical domain, [213] and [222, 223] apply Variational Autoencoder (VAE) models to generate or paraphrase medical or clinical questions. However, none of them explore leveraging QG to improve QA performance on new contexts.

Our aim is to improve clinical QA on new clinical texts (i.e., domain adaptation of clinical QA). We assume the availability of a large set of QA pairs and corresponding clinical documents (source contexts), and our goal is to better answer questions on new documents (target contexts) where only unlabeled documents are provided. We leverage a QG model to synthesize diverse QA pairs to save medical experts annotation efforts and improve QA performance without requiring extra annotations. Our setting is very practical in the real-world scenario, since it is infeasible to always annotate QA pairs on new clinical texts when deploying a QA system into a new environment.

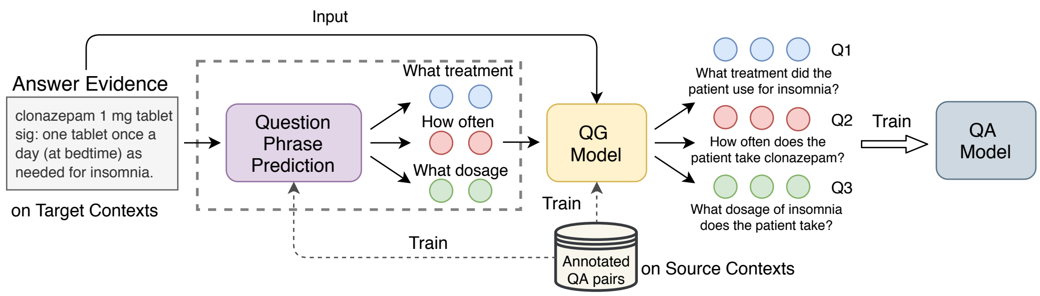


Figure 6.2: Illustration of our Question Phrase Prediction (QPP) module, which can be used together with QG models to diversify generations.

6.3 Methods

6.3.1 Overview of Our Framework

We first give an overview of our **QVE** framework (Fig 6.2). **QVE** improves clinical QA on new contexts by automatically synthesizing QA pairs for new clinical contexts. To approach this, we first leverage an *answer evidence extractor* to extract meaningful text spans from unlabeled documents, based on which a QG model can be applied to generate questions.

In order to encourage diverse questions, we reformulate the question generation process as two-stage. In the first stage, we propose a *question phrase prediction* module to predict a set of question phrases, which represent the types of questions humans would ask, given an answer evidence. In the second stage, following a specific question phrase predicted by our QPP, a QG model is used to complete the rest of the question.

Therefore, our framework **QVE** is able to produce questions of more diverse types. The generated QA pairs by QG models are finally used to train QA models on new contexts.

6.3.2 Answer Evidence Extractor (AEE)

When human annotators create questions, they first read a document and then select a text span to ask questions about. To imitate this process, we implement an *answer evidence* *extractor* to extract possible text spans from a document. Following [174, 278], we focus on longer text spans (as answer evidences) instead of short answers (e.g., a single named entity), since longer text spans often contain richer information compared with short ones, which are very important for the clinical QA task.

More formally, given a document (context) $\mathbf{p} = \{p_1, p_2, ..., p_m\}$, where p_i is the i-th token of the document and m is the total number of tokens, we aim to extract potential evidence sequences. Since the answer evidence is not always a single sentence (sometimes could be multiple sentences), instead of treating it as a sentence selection task, we formulate it as a sequence labeling (or tagging) task. We follow the `BIO` tagging (short for beginning, inside, outside), a commonly used sequence labeling scheme [196], to label answer evidences.

Firstly, we adopt the ClinicalBERT model [4] to encode the document:

$$\mathbf{U} = \texttt{ClinicalBERT}\{p_1, ..., p_m\}. \tag{6.1}$$

where $\mathbf{U} \in \mathbb{R}^{m \times d}$, and d is size of the dimension.

Following the same paradigm of the BERT model for the sequence labeling task [51], we use a linear layer on top of the hidden states output by ClinicalBERT followed by a softmax function to do the classification:

$$\Pr(a_j|p_i) = \text{softmax}(\mathbf{U} \cdot \mathbf{W} + \mathbf{b}),\ \forall p_i \in \mathbf{p} \tag{6.2}$$

where a_j is the predicted `BIO` tag.

After prediction, we observe that the extracted answer evidences sometimes are broken sentences due to the noisy nature and uninformative language (e.g., acronyms) of clinical texts. To make sure that the extracted evidences are meaningful, we designed a *"merge-and-drop"* heuristic rule to further improve the extractor's accuracy. Specifically, for each extracted evidence candidate, we first examine the *length* (number of tokens) of the extracted evidence. If the length is larger than the threshold η, we keep this evidence; otherwise, we compute the *distance*, i.e., the number of tokens between the current candidate span and another closest candidate span If the *distance* is smaller than the threshold γ, we merge these two "close-sitting" spans; otherwise, we drop this overly-short evidence span. In our experiments, we set η and γ to be 3 and 3, respectively, since they help achieve the best performance on the dev set.

6.3.3 Question Phrase Prediction (QPP)

Existing QG models are often biased to generate limited types of questions. To address this problem, we introduce our question phrase prediction module that can be used to diversify the generation of existing QG models.

Formally, denote $V_l = \{s_1, ..., s_L\}$ as the vocabulary of all available question phrases of length l in the training data and $L = |V_l|$ as its size. V_l can be obtained by collecting the first n-gram words in the questions. We set $n = 2$ in our experiment as it achieves the best performance on the dev set. Given an answer evidence $\mathbf{a}$, the goal of QPP is to map $\mathbf{a} \to \mathbf{y} = (y_1, ..., y_L) \in \{0, 1\}^L$, where $y_i = 1$ indicates predicting s_i in V_l as a question phrase for the evidence $\mathbf{a}$. Instead of treating it as a common multi-label classification problem, we formulate the task as *a sequence prediction* problem and adopt a commonly used seq2seq model with an attention mechanism [147] to predict a sequence of question phrases $\mathbf{s} = (s_{j_1}, ..., s_{j_{|\mathbf{s}|}})$ (e.g., "What treatment" (s_{j_1}) $\to$ "How often" (s_{j_2}) $\to$ "What dosage" (s_{j_3}), with $|\mathbf{s}| = 3$).

During training, we assume that the set of question phrases is arranged in a pre-defined order. Such orderings can be obtained with some heuristic methods, e.g., using a descending

Algorithm 3 **QVE** training procedure

Input: labeled *source* data $\{(P_S, A_S, Q_S)\}$, unlabeled *target* data $\{P_T\}$
Output: Generated QA pairs $\{(A'_T, Q'_T)\}$ on *target* contexts; An optimized QA model for answering questions on target contexts;

Pretraining Stage

1: Train *Answer Evidence Extractor* based on the *source* data $\{(P_S, A_S)\}$ using Eq. 6.3
2: Obtain question phrase data Y_S from Q_S and train *Question Phrase Prediction* module on the *source* data $\{(A_S, Y_S)\}$ using Eq. 6.4
3: Train a *QPP-enhanced QG* model on the *source* data $\{(A_S, Y_S, Q_S)\}$ using Eq. 6.5

Training Stage

4: Use *AEE* to extract potential answer evidences $\{A'_T\}$ on the *target* contexts $\{P_T\}$
5: Use *QPP* to predict potential question phrases set $\{Y'_T\}$ on $\{A'_T\}$
6: Use *QPP-enhanced QG* to generate diverse questions $\{Q'_T\}$ based on $\{(A'_T, Y'_T)\}$
7: Train a *QA* model on synthetic *target* data $\{(P_T, A'_T, Q'_T)\}$ using Eq. 6.6

order based on question phrase frequency in the corpus[22]. In the inference stage, QPP can dynamically decide the number of question phrases for each answer evidence by predicting a special `[STOP]` token. By decomposing QG into two steps (diversification followed by generation), the implemented QPP can increase the diversity in a more controllable way.

6.3.4 Training

Algorithm 3 illustrates the pretraining and training procedure of our **QVE**.

During the *pretraining* stage, we first train the answer evidence extractor (AEE) module on the source contexts by minimizing the negative log-likelihood loss:

$$L_{AEE} = -\sum_i \log P(\mathbf{a}|\mathbf{p}; \phi) \tag{6.3}$$

where ϕ represents all the parameters of the answer evidence extractor. For the supervision signals, we identify all evidences in the source data as ground-truth chunks which are marked using the `BIO` scheme.

[22] In our dataset, each answer evidence is tied with multiple questions, which allows the training for QPP.

Table 6.1: Statistics of the datasets. We synthesize a machine-generated dev set and ask human experts to annotate a test set for MIMIC-III.

(Question / Context)	emrQA	MIMIC-III
# Train	781,857 / 337	- / 337
# Dev	86,663 / 41	8,824 / 40
# Test	98,994 / 42	1,287 / 36
# Total	967,514 / 420	- / 413
for purpose of	QG & QA (source)	QA (target)

Moving to the Question Phrase Prediction (QPP) module, given an answer evidence $\mathbf{a}$, we aim to predict a question phrase sequence $\mathbf{y}$ and minimize:

$$L_{QPP} = -\sum_{i} \log P(\mathbf{y}|\mathbf{a};\theta) \tag{6.4}$$

where θ denotes all the parameters of QPP.

Then we can train any QG model (e.g, NQG [55]) on source data by minimizing:

$$L_{QG} = -\sum_{i} \log P(\mathbf{q}|\mathbf{a},\mathbf{y};\mu) \tag{6.5}$$

where μ denotes all parameters of the QG model.

During the *training* stage, given unlabeled target clinical documents, we first extract answer evidences, based on which QPP can be “plugged" into the QG model to generate diverse questions. Finally, a QA model (e.g., DocReader [37]) can be trained on the generated QA pairs of the target documents:

$$L_{QA} = -\sum_{i} \log P(\mathbf{a}|\mathbf{q},\mathbf{p};\delta) \tag{6.6}$$

where δ denotes all parameters of the QA model.

6.4 Generalizability Test Set Construction

Unlike open domain, there are very few publicly available QA datasets in the clinical domain. EmrQA dataset [174], which was generated based on medical expert-made question

Table 6.2: The QA performance on MIMIC-III test set. emrQA is also included as a baseline dataset to help illustrate the generated diverse questions on MIMIC-III are useful to improve the QA model performance on new contexts.

QA Datasets	**DocReader** [37]						**ClinicalBERT** [4]					
	Human Generated		**Human Verified**		**Overall Test**		**Human Generated**		**Human Verified**		**Overall Test**	
	EM	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
emrQA [174]	69.87	83.66	61.44	78.82	63.48	79.99	69.23	82.83	61.23	78.56	63.17	79.59
NQG [55]	66.99	79.67	64.71	79.36	65.26	79.43	67.30	82.59	59.49	76.68	61.38	78.11
+ BeamSearch	71.15	83.07	67.07	81.21	68.07	81.66	68.91	84.26	63.17	79.17	64.56	80.40
+ Top-k Sampling	71.58	83.48	66.77	80.45	67.94	81.19	67.74	81.96	60.82	78.16	62.50	79.08
+ Nucleus Sampling	70.62	83.68	67.16	80.37	68.00	81.17	68.70	83.21	62.36	77.89	63.90	79.18
+ QPP (Ours)	**74.36**	**85.18**	**68.82**	**82.89**	**70.09**	**83.44**	**69.23**	**84.33**	**63.79**	**79.56**	**65.11**	**80.72**
NQG++ [289]	66.34	81.34	65.94	78.71	66.04	79.35	65.06	80.11	59.59	75.85	60.92	76.88
+ BeamSearch	72.11	84.56	68.10	80.09	69.07	81.17	68.26	83.70	64.61	80.30	65.50	81.12
+ Top-k Sampling	73.29	85.56	69.11	82.38	69.41	83.35	70.19	85.61	62.84	79.77	64.62	81.19
+ Nucleus Sampling	73.34	84.95	68.94	81.72	70.01	82.51	70.19	84.72	63.93	79.54	65.45	80.80
+ QPP (Ours)	**74.68**	**85.92**	**70.05**	**83.47**	**71.10**	**84.06**	**70.83**	**85.76**	**65.33**	**80.64**	**66.67**	**81.88**
BERT-SQG [35]	70.19	81.47	66.05	79.64	67.05	80.08	65.06	82.20	59.59	78.04	60.92	79.05
+ BeamSearch	73.71	84.44	68.71	81.98	69.93	82.58	67.31	82.54	61.94	79.02	63.25	79.88
+ Top-k Sampling	72.81	84.16	69.20	82.24	70.07	82.71	69.12	84.20	60.44	78.27	62.55	79.71
+ Nucleus Sampling	70.73	83.60	68.56	81.80	69.09	82.24	67.74	83.16	61.61	78.74	63.09	79.81
+ QPP (Ours)	**74.36**	**85.53**	**70.77**	**83.60**	**71.64**	**84.07**	**69.23**	**85.38**	**64.21**	**80.53**	**65.43**	**81.71**

templates and existing annotations on n2c2 challenge datasets [161], is a commonly adopted dataset for clinical reading comprehension.

However, all the QA pairs in emrQA are based on n2c2 clinical texts and thus not suitable for our generalization setting. [278] studied a similar problem and annotated a test set on MIMIC-III clinical texts [104]. However, their test set is too small (only 50 QA pairs) and not publicly available. Given the lack of a reasonably large clinical QA test set for studying generalization, with the help of three clinical experts, we create 1287 QA pairs on a sampled set of MIMIC-III [104] clinical notes, *which have been reviewed and approved by PhysioNet*[23] *and is downloadable by following the instructions*[24]. **Annotation**

[23] `https://physionet.org/`. PhysioNet is a resource center with missions to conduct and catalyze for biomedical research, which offers free access to large collections of physiological and clinical data, such as MIMIC-III [104].

[24] https://physionet.org/content/mimic-iii-question-answer/1.0.0/.

Process. We sample 36 MIMIC-III clinical notes as contexts. When sampling MIMIC-III notes, we ensure that all the sampled clinical texts do not appear in emrQA, acknowledging that there is a small overlap between the two datasets. For each context, clinical experts can ask any questions as long as an answer can be extracted from the context. To save annotation effort, QA pairs generated by QG models (i.e., all base QG models and their diversity-enhanced variants; see Section 6.5.1) are provided as references, and duplicates are removed. Meanwhile, clinical experts are *highly encouraged* to create new questions based on the given clinical text (which are marked as *"human-generated"/"HG"*). But if they do find the machine-generated questions sound natural and match the provided answer, they can keep them (which are marked as *"human-verified"/"HV"*). After obtaining the annotated questions, we ask another clinical expert to do a final pass of the questions in order to further ensure the quality of the test set. The final test set consists of 1287 questions (of which 975 are *"human-verified"* and 312 are *"human-generated"*).

We understand that there might be potential bias when evaluating QA models on the HV set (i.e, a QG model which is used to generate training questions for a QA model also contributes questions to the HV set as well). However, such bias might exist in human annotated data as well (e.g., the same set of humans create both training and testing dataset). Note that the contexts used to generate questions in HV/HG are separated from those to generate training questions. Besides, due to the relatively limited language patterns in clinical domain, we find most questions in HV set sound like what humans would ask. As such, we still deem it as a valuable asset and potential future research could leverage our HV set as their dev set to tune hyper-parameters.

To help tune the model, we also construct dev set of MIMIC-III by sampling generated questions from QG models and their variants and is used to tune the hyper-parameters. In the following sections, we consider emrQA as the *source* dataset and our annotated MIMIC-III QA dataset as the *target* data. Detailed statistics of the two datasets are in Table 6.1.

6.5 Experimental Setup

6.5.1 Base QG models

We instantiate our **QVE** framework using three base QG models:

• **NQG** [55] is the first seq2seq model with a global attention mechanism [147] for question generation.

• **NQG++** [289] is one of the most commonly adopted QG baselines with a feature-enriched encoder (e.g., lexical features) and a copy mechanism [86].

• **BERT-SQG** [35] uses a pretrained BERT model (we use ClinicalBERT [4] to accommodate clinical setting) as the encoder and formulates the decoding as a “MASK" token prediction problem.

It has been studied that beam search and sampling strategies show competitive performance in diversifying generations [98, 225]. We thus include Top-k [67] and Nucleus samplings [93] as representative sampling strategies in our experiments.

As such, to investigate the effectiveness of diverse QG for QA, we consider the following variants of each base QG model: (1) Base Model: Inference with greedy search; (2) Base Model + Beam Search: Inference with Beam Search of beam size K and keep top K beams ($K = 3$); (3) Base Model + Top-k sampling: Inference with sampling from top-k tokens ($k = 20$); (4) Base Model + Nucleus sampling: Inference with sampling from top-p tokens ($p = 0.95$); (5) Base Model + QPP: Inference with greedy search for both QPP module and Base model.

6.5.2 Base QA models

For QA, we instantiate **QVE** with two base models, DocReader [37] and ClinicalBERT [4]. When training a QA model, we only use the synthetic data on the target contexts and do not combine the synthetic data with the source data since the combination does not help in our preliminary studies.

Note that more complex QG/QA models and training strategies can also be used in our framework. As this work focuses on exploring how *diverse* questions help QA on target contexts, we adopt fundamental QG/QA models and training strategies, and leave more advanced ones that are complementary to our framework as future work.

6.5.3 Evaluation Metrics

For QA evaluation, we report exact match (EM) (percentage of predictions that match the ground truth answers exactly) and F1 (average overlap between the predictions and ground truth answers) as in [194]. Since our main goal is to evaluate whether the generated questions are useful to improve the QA performance on the target contexts, the common language generation metrics such as BLEU [176] and ROUGE-L [140] are not suitable to reflect the quality of the generated questions, and thus we do not adopt these metrics in our experiments.

6.5.4 Implementation Details

Base QG Models: We re-implement three base QG models using Pytorch and have ensured that they achieve comparable performance as originally reported. Best QG models are selected using the per-token accuracy of both the QPP module (if applicable) and QG on dev set.

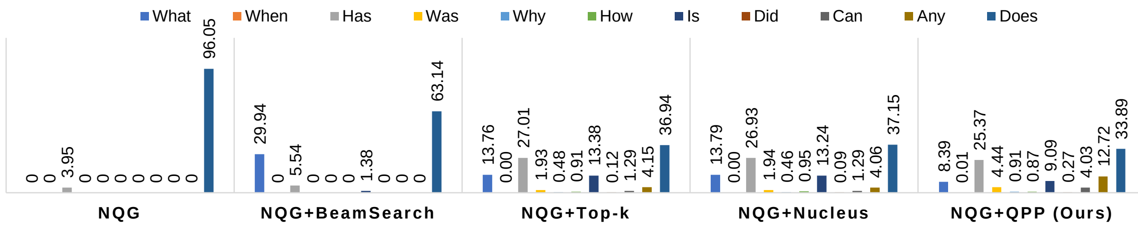


Figure 6.3: Distributions over types of questions generated by NQG models.

Base QA Models: We use the open-sourced implementation.[25][26] Best QA models are selected using EM and F1 on dev set.

Hyperparameters Search: Hyperparameters of QG models are set to be the same as in original papers and hyperparameters of QA models are set according to [278]. Specifically, we train NQG and NQG++ up to 20 epochs, BERT-SQG up to 5 epochs, DocReader up to 5 epochs and ClinicalBERT up to 3 epochs.

6.6 Experimental Results

6.6.1 Can Generated Questions Help QA on New Contexts?

Table 6.2 summarizes the performance of two widely used QA models, DocReader [37] and ClinicalBERT [4], on the MIMIC-III testing set. The QA models are trained based on different corpora, including the emrQA dataset as well as QA pairs generated by different models. For a fair comparison, we keep the total number of generated QA pairs roughly the same as emrQA. As can be seen from the table, the QA models based on the corpora that are generated using the three base QG models can only achieve roughly the same or even worse performance compared with the QA models trained on the emrQA dataset. Though the Beam Search and sampling strategies could boost the diversity of generated questions to

[25]DocReader: https://github.com/facebookresearch/DrQA

[26]ClinicalBERT: https://github.com/EmilyAlsentzer/clinicalBERT

some extent, and thus lead to the improvement of QA models, our proposed QPP module can improve the QA performance by a larger margin. For example, training DocReader using questions generated by NQG++ with our QPP module outperforms that using the emrQA dataset by around 8% under EM and 4% under F1 on the overall test set. Moreover, the results on human-generated portion are consistently better than that on human-verified. It's attributed to the fact that human-created questions are more readable and sensible while human-verified questions are a bit of less natural though correctness is ensured.

All these results indicate that generating a diverse QA corpus is useful for downstream QA on new contexts, and our simple QPP module can help existing QG models achieve such a goal.

6.6.2 Why QG Boosts QA on New Contexts?

To further explore why QG can boost QA, we consider three major factors when generating a QA corpus: the number of documents, the number of answer evidences per document, and the number of generated questions per answer evidence. When we test one factor, we fix the other two. For example, we fix the number of answer evidences and questions at 20 and 6 when we test the influence of the number of documents. We use NQG++ and DocReader as our base QG and QA models to instantiate our **QVE** framework and report the performance on the Dev set.

As can seen from Fig 6.4, the performance steadily increases when we use more documents and more answer evidences during QA corpus generation. This can demonstrate the first hypothesis: The generated corpus enables a QA model to see more new contexts during training, which can help the QA model get a better understanding of similar contexts during testing. The more contexts it sees, the more benefits it could obtain. We can also see

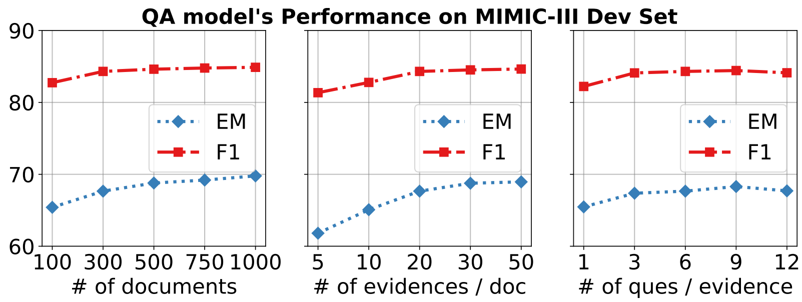


Figure 6.4: Influence of the number of documents, number of evidences per document, number of QA pairs per evidence on QA performance.

that with the increase of the number of generated questions per evidence, the performance generally rises up. This indicates that multiple diverse questions are essential for boosting QA performance.

A Closer Look at Generated Question Types. To further demonstrate QPP module can help generate diverse questions, we show the distribution over the types of questions generated by NQG-based models in Fig 6.3.

We observe that questions generated by base NQG and NQG+BeamSearch are limited in terms of the question types. However, more types of questions (e.g., “How", “Why") can be generated when enabling sampling strategies. Furthermore, when being equipped with our QPP module, the NQG model can even generate questions of an extremely rare type, i.e., "When" questions. Though Top-k and Nucleus sampling methods also generate questions of less frequent types, our QPP module could cover even more types.

In summary, we think seeing many new contexts and diverse questions are the two main reasons why QA models are boosted.

QA Example from MIMIC-III

Context: ... he was guaiac negative on admission. hematocrit remained stable overnight. **5. abd pain: suspect secondary to chronic pancreatitis.** amylase unchanged from previous levels. ...

Question: Why did the patient get abd pain?
Answer by QA model trained on
-emrQA: 5. abd pain
-NQG: 5. abd pain:
-NQG+BeamSearch: 5. abd pain:
-NQG+Top-k: 5. abd pain:
-NQG+Nucleus: 5. abd pain:
-NQG+QPP: 5. abd pain: suspect secondary to chronic pancreatitis**.**

QG Example from MIMIC-III

Context: ... the patient was taking at home prior to admission were not restarted. **25. acetaminophen 325-650 mg po/ng q6h:prn pain** 26. dabigatran etexilate 150 mg po bid...

Questions generated by
-NQG: **Does** the patient have any pain?
-NQG+BeamSearch: **Does** the patient have any pain history? **Does** the patient have pain? **Does** the patient have any pain?
-NQG+Top-k: **Has** the patient ever had any pain? **Has** the patient ever reported pain? **Does** the patient have a history pain?
-NQG+Nucleus: **Has** the patient ever gone into pain? **What** happened when she was given morphine? Is there mention pain anywhere in the record?
-NQG+QPP: **Why** did the patient have acetaminophen? **What** treatment has the patient had for his pain? **How** was pain treated? **Does** the patient have any pain? ...

Figure 6.5: QA and QG examples. The red parts in contexts are ground-truth answer evidences.

6.6.3 Diverse Questions Really Matter for QA: Two Real Cases.

In Fig 6.5, we present a QA example and a QG example from MIMIC-III for qualitative analysis.

In the QA example, this “why" question can be correctly answered by the QA model (DocReader) trained on the “NQG+QPP" generated corpus while the QA models trained on other generated corpora fail. This is because, as shown in Fig 6.3, the NQG model and “NQG+BeamSearch" cannot generate any “why" questions and sampling strategies

Table 6.3: The QA performance on MIMIC-III test set when QPP is employed with sampling strategies

QA Datasets	DocReader [37]					
	Human Generated		Human Verified		Overall Test	
	EM	F1	EM	F1	EM	F1
NQG	66.99	79.67	64.71	79.36	65.26	79.43
+ QPP	**74.36**	**85.18**	**68.82**	**82.89**	**70.09**	**83.44**
+ Top-k	71.58	83.48	66.77	80.45	67.94	81.19
+ Tok-k + QPP	72.52	84.98	67.67	81.79	68.84	82.56
+ Nucleus	70.62	83.68	67.16	80.37	68.00	81.17
+ Nucleus + QPP	74.12	85.08	68.10	81.36	69.56	82.26

could only help generate a limited number of “why” questions. Thus QA models trained on such corpora cannot answer questions of less frequent types. Though the emrQA dataset contains diverse questions (including “why" questions), its contexts might be different from MIMIC-III in terms of topic, note structures, writing styles, etc. So the model trained on emrQA struggles to answer some questions as well.

In the QG example, the base model NQG can only generate one question. Though utilizing the Beam Search enables the model to explore multiple candidates, the generated questions are quite similar and are less likely to help improve QA. Sampling strategies, though further diversifying the generation during decoding, suffer from generating irrelevant contents (e.g., “NQG+Nucleus” generates a irrelevant “morphine” token). Enabling our QPP module helps generate relevant and diverse questions including “Why", “What", “How", etc.

6.6.4 Ablation Study

Performance of QPP with Sampling Strategies. Since our QPP is compatible with sampling strategies, we further study the performance after combining these two techniques.

Table 6.4: Choosing seq2seq-based QPP over alternative multi-label classification methods. HL: Hamming Loss.

Models	**HL**	**Precision**	**Recall**	**F1**
Binary Relevance	0.0524	**99.22**	90.89	94.87
Classifier Chain	0.0524	**99.22**	90.89	94.87
QPP	**0.0346**	97.28	**96.20**	**96.74**

Table 6.3 shows the results, which indicate that combining two techniques can improve the sampling strategies' performance but do not lead to further improvement compared with using QPP only. This demonstrate that our QPP module is good enough to generate diverse useful questions for improving QA.

Alternative Approaches for QPP. There are many model options for the QPP task, e.g., those for multi-label classification. To justify our choice of a seq2seq model, we compare it with two commonly-adopted multi-label classification methods: binary relevance (BR) and classifier chain (CC) [22, 199]. BR develops multiple binary classifiers independently while CC builds a chain of classifiers and predicts labels sequentially. We use multi-layer perceptron as the base model for both BR and CC. For each answer evidence, the input is the representation from the same LSTM encoder as our QPP module.

From Table 6.4, we can see: (1) The seq2seq design in our QPP module performs better overall and especially in terms of Recall, which is particularly important since we aim for generating diverse question types; (2) A simple seq2seq model achieves great performance across all metrics, which renders developing more complex models for this task less necessary.

6.7 Conclusion

This chapter proposes a simple yet effective framework for improving clinical QA on new contexts. It leverages a seq2seq-based question phrase prediction module to enable QG models to generate diverse questions. Our comprehensive experiments and analyses allow for a better understanding of why diverse question generation can help QA on new clinical documents.

Chapter 7: Synthetic Question Value Estimation for Domain Adaptation of Question Answering

Synthesizing QA pairs with a question generator (QG) on the target domain has become a popular approach for domain adaptation of question-answering (QA) models. Since synthetic questions are often noisy in practice, existing work adapts scores from a pre-trained QA (or QG) model as criteria to select high-quality questions. However, these scores do not directly serve the ultimate goal of improving QA performance on the target domain. In this paper, we introduce a novel idea of training a *question value estimator (QVE)* that directly estimates the usefulness of synthetic questions for improving the target-domain QA performance. By conducting comprehensive experiments, we show that the synthetic questions selected by QVE can help achieve better target-domain QA performance, in comparison with existing techniques. We additionally show that by using such questions and only around 15% of the human annotations on the target domain, we can achieve comparable performance to the fully-supervised baselines.

7.1 Introduction

Question answering (QA) systems based on pretrained language models such as BERT [50] have recently achieved promising performance in machine reading comprehension. However, neural QA systems trained on one domain may not generalize well to another,

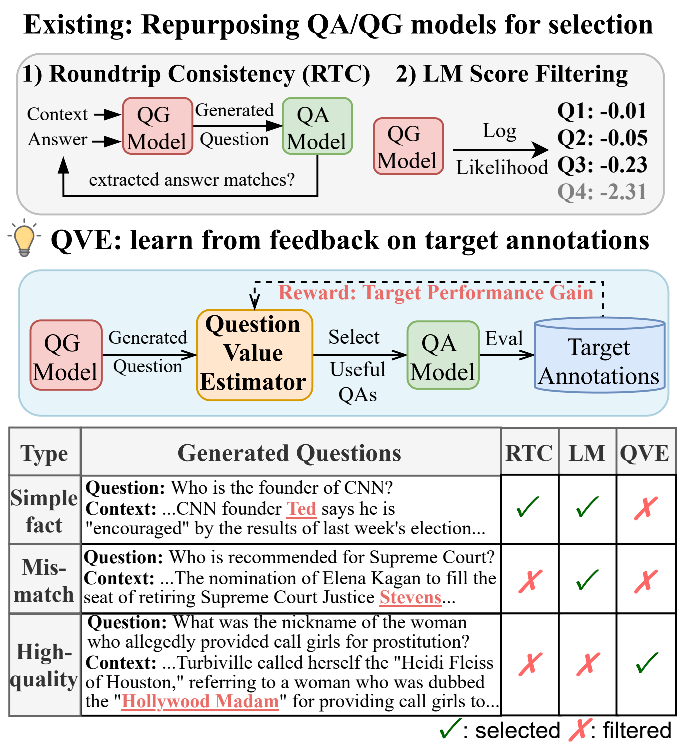


Figure 7.1: Existing work repurposes a pretrained QA (or QG) model to evaluate the quality of the generated questions, which is not directly associated with the target-domain QA performance and may select questions that are semantically-mismatched or ask about a simple fact. In contrast, our Question Value Estimator (QVE) learns to select useful questions with target-domain QA performance gain as direct feedback.

leaving it challenging to deploy such systems on new domains that lack large-scale QA training data[27]. In this chapter, we are interested in *semi-supervised domain adaptation*: we aim to build a target QA model with source-domain data and a small number of target-domain annotated QA pairs.

Due to high annotation costs, existing work [81, 53, 246, 183, 40, 280] proposes to synthesize target-domain QA pairs via neural question generation (QG) models. The synthetic data are then used to train a QA model on the target domain. In practice, however,

[27] Large-scale training data are typically 60-100K in size.

the generated questions are often of low quality, such as being semantically mismatched with their paired answers or asking about simple facts (Figure 7.1). Including all such questions for QA training is less likely to bring substantial improvements. This inspires us to study a crucial problem:

Given a set of target-domain synthetic QA pairs, how to select high-quality ones that are useful to improve target-domain QA training?

To address the problem, Alberti et al. [2] propose the Roundtrip Consistency (RTC) method, which filters[28] questions that cannot be correctly answered by a pretrained QA model. Other work [212] considers using the generation log-likelihood by the QG model (LM Score) as a metric to filter noisy questions (Figure 7.1, top). Although these filtering techniques have been shown to improve the question quality to some extent [202], they are not directly optimized for *selecting questions that can improve QA performance on the target domain*. For example, some useful but difficult questions (e.g., the last example in Figure 7.1) may be filtered by the Roundtrip method, since they cannot be answered correctly by the pretrained QA model. However, these questions are often crucial to further improving QA performance when added into training.

In this chapter, we propose a *question value estimator (QVE)* (Figure 7.1, middle) to select questions that can improve QA performance on the target domain. QVE takes in generated QA examples and outputs real-valued scores (i.e., question values), which are expected to represent the usefulness of generated questions in terms of improving target-domain QA performance. However, training the QVE model towards this goal is challenging due to the lack of supervision (i.e., true question values).

[28]We interchangeably use "filter" (noisy/low-quality questions) and "select" (useful/high-quality questions).

To solve the problem, we propose to train the QVE with direct QA feedback from the target domain. Intuitively, if a batch of synthetic questions (when used for training) leads to increasing accuracy of the target-domain QA model, QVE should assign high values to them; the more the accuracy increases, the higher the question values should be. Thus, we optimize QVE with the *target-domain QA performance gain* after adding the selected questions into training. More formally, given the discrete and non-differentiable question selection process, we formulate the question selection of QVE as a reinforcement learning [257] problem (Figure 2.1). The QVE receives a batch of synthetic samples each time and learns to select high-quality ones based on their estimated values. The selected samples are then used to train the target-domain QA model, with the resulting performance gain (on the available target-domain annotations) as the reward. The reward guides the optimization of QVE such that it will eventually make proper question value estimation and selection.

To evaluate the QVE model, we instantiate the QG and the QA model based on the pretrained BART [130] and BERT [50], respectively. By carrying out comprehensive experiments on four commonly-used reading comprehension datasets [235, 107, 273, 123], we show that: (1) our QVE model trained with the target-domain QA feedback substantially outperforms the question selection techniques trained without direct QA feedback [2, 212]. (2) When using our QVE model to select synthetic questions, QA models can achieve comparable performance to fully-supervised baselines while using only 15% of the full target-domain annotations, which indicates that our method can greatly alleviate human annotation effort in practice. (3) To understand why QVE brings superior improvement, we conduct human evaluation and find that QVE can better identify semantically-matched and difficult questions.

7.2 Related Work

Domain Adaptation of Question Answering. In this field, some work [255, 46, 90, 27] assumes that target-domain annotated questions are available, however, manually creating questions is costly. Therefore, another line of research work [81, 246, 125, 212] investigates a domain adaptation setting where annotated questions are not available on the target domain. A commonly-adopted approach of this line is to leverage a neural question generation (QG) model [55, 289, 226, 287, 164, 237] to automatically synthesize questions given unlabeled contexts [54, 283, 246, 141, 81, 246, 125, 212, 280]; see more discussions in Section 7.3. However, it is very challenging to achieve satisfying performance without any target annotations. In our work, we study *semi-supervised domain adaptation of QA*, and assume *a small number of target annotations are available*, which can greatly help models adapt to the target domain while requiring minimal human effort.

Unsupervised and Semi-supervised QA are two other research topics relevant to our work [65, 139, 133, 52]. Unlike domain adaptation, these two settings do not assume the existence of the "source domain" and synthesize cloze-style questions via rule-based methods for building QA models. Since rule-based QG methods typically have much worse performance than neural ones (pretrained on the source data), we do not compare with these two lines of research in experiments.

Data Selection methods aim to select a useful subset from the (noisy) training data. Though (RL-based) data selection methods were explored in other NLP tasks [205, 186, 142], none of them can be directly applied with trivial efforts to our QA scenario and semi-supervised setting. For example, [205] and [142] reward or measure the selection with the distribution distance between the selected data and target data, while we reward the selection by measuring how large the improvement the selected data can bring for target-domain QA training, which is more aligned with the end goal. Our work is mostly inspired by recent research on data selection in machine learning community [79, 103], particularly [275]. However, the significant differences between our work and [275] are as follows: 1) we study a very challenging task, domain adaptation of question answering, which was not studied in [275]. How to develop a method in a similar spirit for this task is unexplored. 2) In order to study the task, we begin our method by first proposing two data selection methods that are not covered in [275] but achieve comparable results to existing baselines. We then introduce our RL-based method with a carefully-designed reward, which is well connected to the end goal of improving target-QA performance.

7.3 Background

7.3.1 Domain Adaptation of QA via QG

Semi-supervised Domain Adaptation. We study the semi-supervised domain adaptation of *extractive* question answering, where the source-domain and a small number[29] of target-domain QA annotations are provided. Formally, we denote the source-domain QA dataset as $D^{\mathsf{s}} = \{(c_i^{\mathsf{s}}, q_i^{\mathsf{s}}, a_i^{\mathsf{s}})\}_{i=1}^{N}$, where large-scale tuples of context c_i^{s}, question q_i^{s}, and answer a_i^{s} are available. For the target domain, only a small set of annotated QA pairs $D^{\mathsf{t}} = \{(c_j^{\mathsf{t}}, q_j^{\mathsf{t}}, a_j^{\mathsf{t}})\}_{j=1}^{M}$ are available ($M \ll N$). Since unlabeled contexts are easy to collect, we assume that they are largely available: $C^{\mathsf{t}} = \{c_l^{\mathsf{t}}\}_{l=1}^{L}$ ($L \gg M$). The task is to build a QA model that can accurately answer questions on the target domain, given D^{s}, D^{t}, and C^{t}.

Domain Adaptation via Question Generation. Given the lack of large-scale target-domain annotations, an intuitive approach to domain adaptation is first synthesizing target-domain QA data $D_{syn}^{\mathsf{t}} = \{(c_l^{\mathsf{t}}, q_l^{\mathsf{t}}, a_l^{\mathsf{t}})\}_{l=1}^{L}$ automatically from the unlabeled contexts C^{t}, and then

[29] In our experiments, we assume 1,000 target annotations available, which is around 1-1.5% of the original training data.

training a target-domain QA model on the synthetic (D^{t}_{syn}) and the small-size annotated (D^{t}) target-domain data. In such an approach, a question generator (QG) g_ϕ is first pretrained on the source training data and further finetuned on the available target-domain annotated QA pairs. A well-trained QG model then takes target-domain context-answer pairs as input to generate a question: $q^{\mathsf{t}}_l = g_\phi(c^{\mathsf{t}}_l, a^{\mathsf{t}}_l)$.

Although this approach has been shown promising, in practice, its effectiveness is restricted by the quality of synthetic questions. Thus, learning to select ones that can lead to a better target-domain QA model becomes a crucial problem.

With respect to how to obtain a^{t}_l for QG, here we assume an answer a^{t}_l (i.e., a text span in the context c^{t}_l) is given, following Du et al. [55]. When the answer a^{t}_l is not given, it can be extracted from the given context by using an entity recognition tool [54], a classifier [183] or a seq2seq model [212]. Note that noise caused by such answer extraction tools will further lower the overall quality of the synthesized questions. In this chapter, we focus on how to select useful synthetic questions in general (i.e., those questions can be synthesized by any QG process) and assume answers are given for simplicity.

7.3.2 Synthetic Question Selection

Given the synthetic target-domain QA data D^{t}_{syn}, the task is to select high-quality pairs from D^{t}_{syn} that are useful to improve target-domain QA training. Such a selection decision is often made based on some scores that can indicate the quality of the pairs. For example, Roundtrip filtering [2] selects questions based on the extracted answer's correctness by a pretrained QA model. Similarly, LM filtering [212] selects questions with high log-likelihood scores in the generation. However, these scores do not directly serve the goal of improving target-domain QA training. Inspired by recent research on data selection in the

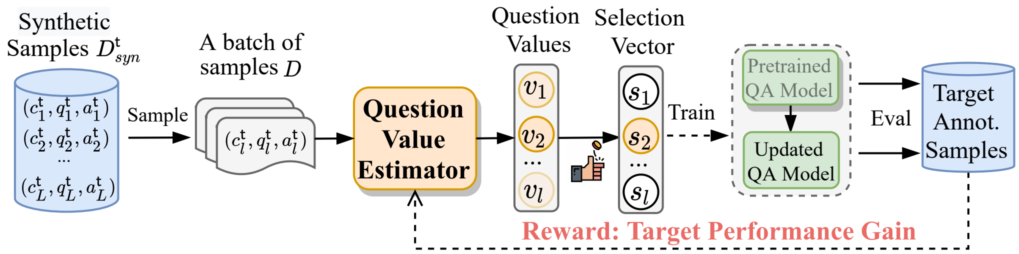


Figure 7.2: Illustration of QVE training based on the direct feedback from QA. Specifically, in the forward pass, QVE estimates the question values of a batch of synthetic questions and draws a Bernoulli sampling to select questions. The selected questions are then used to finetune a pretrained QA model. The performance gain (before and after the QA finetuning) on the target annotations is calculated as the reward for REINFORCED QVE training.

machine learning community [79, 103, 275], we propose a new idea of training a *question value estimator*, which predicts the usefulness of a synthetic question for target-domain QA.

7.4 Question Value Estimator (QVE)

Formally, we design a question value estimator (QVE), e_γ, which takes in a synthetic QA example (c_l, q_l, a_l) (for simplicity, we omit the superscript $\mathtt{t}$) and outputs a score indicating its "value," i.e., $v_l = e_\gamma(c_l, q_l, a_l)$. The "value" can imply "the potential for improving the target-domain QA performance when being used as a training sample". With this score, one can select most useful synthetic examples for the target-domain QA training.

We use a BERT model as the backbone of the QVE. Specifically, we concatenate the context, question and answer as input to the QVE, and use BERT to encode the sequence [50].

$$\mathbf{h} = \mathsf{BERT}\,[\mathsf{<CLS>}\ q\ \mathsf{<ANS>}\ a\ \mathsf{<SEP>}\ c]$$

where q, a, c represent the question, answer, and context, respectively. $\mathbf{h} \in \mathbb{R}^H$ denotes the hidden representation of the input sequence derived from the "<CLS>" token. <ANS> and <SEP> are two special tokens used as delimiters.

In our preliminary experiments, we find that adding the answer (start index and end index) probabilities (p_s, p_e) by a pretrained QA model as additional features to the hidden representation $\mathbf{h}$ can accelerate the QVE training convergence and lead to better performance. Thus, we add these two features (p_s, p_e) followed by linear transformations of the original hidden representation, and then build a linear classifier to output the question value.

$$\mathbf{h}' = \sigma(W_2\sigma(W_1\mathbf{h} + b_1) + b_2)$$

$$\mathbf{h}'' = \sigma(W_3(\mathbf{h}' \oplus p_s \oplus p_e) + b_3)$$

$$v_l = W_4\mathbf{h}'' + b_4$$

where $W_1 \in \mathbb{R}^{H_1 \times H}, W_2 \in \mathbb{R}^{H_2 \times H_1}, W_3 \in \mathbb{R}^{H_3 \times H_2}, W_4 \in \mathbb{R}^{H_3}$, $b_1 \in \mathbb{R}^{H_1}, b_2 \in \mathbb{R}^{H_2}, b_3 \in \mathbb{R}^{H_3}, b_4 \in \mathbb{R}$ are trainable parameters of linear layers. σ is the activation function `tanh`.

Learning such a question value estimator is challenging because we do not have direct supervision on the true value or usefulness of a synthetic question. We discuss two straightforward baselines to train QVE in Section 7.4.1, and a more advanced one based on reinforcement learning in Section 7.4.2.

7.4.1 QVE Training: Two Baselines

Binary Classifier: One straightforward solution is to treat QVE as a binary classifier and train it based on the human-annotated (positive) and the machine-synthesized (negative) QA pairs. Given the scarcity of target-domain data, we first pretrain the classifier on the source domain and then finetune it on the target domain. More specifically, we train a QG model on 70% of the source training data and generate synthetic questions on the remaining 30%

of the source training contexts. The generated questions and the source-domain annotated questions are used to train this binary classifier. The classifier is then finetuned based on the small set of target-domain annotations (positive) and the samples synthesized on the same target-domain contexts (negative).

However, not all of the generated questions are bad. Simply treating all synthetic samples as negatives may mislead the classifier. Thus, we loose this assumption and introduce a ranking baseline.

Ranking Baseline: We assume that the quality of human-annotated questions is not inferior than that of machine-synthesized ones. Thus, we train QVE based on a ranking triplet loss defined as follows:

$$L_r = \sum \max(0, m + v_s - v_h)$$

where v_s, v_h are the estimated question values of the machine-synthesized sample and human-annotated sample. m is set to 0.15 as the margin.

The two baseline methods have two obvious drawbacks: (1) they are trained to differentiate between human-annotated and machine-synthesized samples, which is mismatched with our goal of selecting high-quality samples *among machine-synthesized data*; (2) similar as [2, 212], the two baselines are *not* trained with direct signals that can represent the usefulness of a synthetic question. In the next section, we will introduce a task-specific training method, which directly uses the target-domain QA feedback to optimize QVE.

7.4.2 QVE Training: Direct Feedback from QA

A well-trained QVE is expected to assign high values to synthetic questions that can improve the target-domain QA performance. Therefore, an intuitive way to measure the value of a synthetic question is to consider the downstream QA performance gain (on

Algorithm 4 QVE REINFORCED Training

Input: pretrained QA model f_θ; target synthetic QA pairs D^{t}_{syn}; small target annotations D^{t}.
Hyperparameters: outer iterations I_o, outer batch size B_o, inner iterations I_n, inner batch size B_n, QVE learning rate α_o, QA learning rate α_n.
Output: QVE e_γ.

Randomly initialize e_γ
Store $\theta_0 \leftarrow \theta$ (pretrained QA checkpoint)
for outer iteration $= 1$ to I_o **do**
  ▷ ① *Sample a batch of synthetic QA pairs:*
  Sample $\mathcal{D} = \{(c_l, q_l, a_l)\}_{l=1}^{B_o}$ from D^{t}_{syn}
  ▷ ② *Estimate question values:*
  $\mathcal{V} = e_\gamma(\mathcal{D})$
  ▷ ③ *Sample selection vector:*
  $\mathcal{S} \sim \texttt{Bernoulli}(\mathcal{V})$
  ▷ ④ *Update QA on selected samples:*
  for inner iteration $= 1$ to I_n **do**
    Sample $\{(c_l, q_l, a_l)\}_{l=1}^{B_n} \sim \mathcal{D}$
    $\theta \leftarrow \theta - \frac{\alpha_n}{B_n} \sum_{l=1}^{B_n} s_l \cdot \nabla_\theta \mathcal{L}_{qa}$
  end for
  ▷ ⑤ *Calculate QA gain as QVE reward:*
  $r_{qve} = \texttt{reward\_fn}(f_{\theta_0}, f_\theta, D^{\mathrm{t}})$
  ▷ ⑥ *Update QVE based on Eq. 7.1:*
  $\gamma \leftarrow \gamma - \alpha_o \cdot \nabla_\gamma \mathcal{L}_\gamma$
  Reset $\theta \leftarrow \theta_0$
end for
return e_γ

the available target annotations) before and after this question is included in the training set. However, this "leave-one-out" formulation is computationally expensive and time-consuming, given that it can estimate the value of only one single synthetic question in each forward pass. In light of this challenge, we instead estimate question values in a *batch-wise* fashion. Algorithm 4 and Figure 7.2 describe the learning process.

Generally speaking, we frame the QVE model learning as a reinforcement learning problem [257], and stimulate QVE to assign higher values to more useful questions by using performance-driven rewards. Specially, for a batch of synthetic examples $\mathcal{D} =$

$\{(c_l, q_l, a_l)\}_{l=1}^{B_o}$ in the outer training iteration (Line 4-5), the QVE model *selects a subset of examples* that are most likely to boost the QA performance on the target domain, based on its judgment on their values.

Mathematically, the decision-making outcome is represented by the selection vector $\mathcal{S} = (s_1, s_2, ..., s_{B_o})$, where $s_l \in \{0, 1\}$ $l = 1, ..., B_o$ (Line 6-9). The whole batch-level decision making policy π_γ is described as follows:

$$v_l = e_\gamma(c_l, q_l, a_l)$$

$$s_l \sim \texttt{Bernoulli}(v_l)$$

$$\pi_\gamma(\mathcal{S}|\mathcal{D}) = \prod_{l=1}^{B_o} [v_l^{s_l} \cdot (1 - v_l)^{1-s_l}],$$

where the selection of a certain example (c_l, q_l, a_l) is formulated as sampling from a Bernoulli distribution of probability v_l (i.e., its estimated question value). We adopt the Bernoulli sampling based on the estimated value v_l instead of setting a hard threshold to encourage the policy exploration.

The model is rewarded based on how much performance gain the selected examples could bring when they are used to train the target-domain QA model. To this end, we finetune the QA model f_θ on the selected batch samples based on $\mathcal{L}_{qa}$, which typically is a cross-entropy loss:

$$\mathcal{L}_{qa} = -\sum_{l}^{B_o} \log P(a_l | q_l, c_l; \theta)$$

In practice, to stabilize the QVE training, we choose a large outer batch size B_o in each outer training iteration. For finetuning the QA model, we pick a relatively smaller inner batch size B_n and repeat the training for I_n times, such that the QVE-selected samples are fully utilized (Line 10-14).

The reward r_{qve} is defined as the QA performance gain on the target-domain annotations D^{t} before (f_{θ_0}) and after (f_θ) finetuning (Line 15-16),

$$r_{qve} = \texttt{reward\_fn}(f_{\theta_0}, f_\theta, D^{\mathsf{t}})$$

where `reward_fn` is Exact Match (EM) gain[30].

Given the discrete and non-differentiable question selection process, we update the QVE model using the REINFORCE algorithm [257]. Mathematically, we aim to minimize:

$$\mathcal{L}_\gamma = - \mathop{\mathbb{E}}_{\mathcal{S} \sim \pi_\gamma(\cdot|\mathcal{D})} [r_{qve}].$$

The gradient of the loss function is derived as:

$$\begin{aligned} \nabla_\gamma \mathcal{L}_\gamma &= - \mathop{\mathbb{E}}_{\mathcal{S} \sim \pi_\gamma} [r_{qve} \nabla_\gamma \log \pi_\gamma(\mathcal{S}|\mathcal{D})] \\ &= - \mathop{\mathbb{E}}_{\mathcal{S} \sim \pi_\gamma} [r_{qve} \nabla_\gamma \sum_{l=1}^{B_o} \log[v_l^{s_l}(1 - v_l)^{1-s_l}]]. \end{aligned} \tag{7.1}$$

Notably, to mitigate the instability in reinforcement learning, we reset the QA model to its pretrained checkpoint at the end of each outer iteration (Line 19), and keep the pretrained QG model unchanged.

After training QVE, we can use it to calculate the question value for all the synthetic questions on the target domain. Then we can select top $K\%$ synthetic QA pairs as the training corpus to train the target-domain QA model.

7.5 Experimental Setup

7.5.1 Datasets

We use datasets in the MRQA 2019 Shared Task [72], a popular challenge focusing on generalization in reading comprehension. Specifically, following [212], we use **SQuAD 1.1**

[30] We also tried F1 gain and loss drop as the `reward_fn` and the EM gain is slightly better than the other two.

[193] as the *source-domain* dataset. For the *target-domain* datasets, we consider **NewsQA** [235], **Natural Questions (NQ)** [123], **HotpotQA** [273] and **TriviaQA** [107] as they are commonly used and have sufficient contexts for the QG model to generate synthetic samples. Since there is no test set available for each dataset, we use the original dev set as the test set. Detailed descriptions of each dataset are in Appendix F.1.

For the target-domain datasets, we assume all the contexts and n annotated QA pairs in the original training sets are available for training. We set $n = 1000$ (about 1%-1.5% of original training sets) as default and discuss the impact of n in Section 7.6.2.

7.5.2 Implementation Details

We implement models using the Hugging Face transformers [261] library. We instantiate the QA model with `BERT-base-uncased` [50], and the QG model with `BART-base` [130]. For training QVE (Algorithm 4), we use `BERT-base-uncased` model and set $H_1 = H_3 = H = 768$ and $H_2 = 64$ for linear layers. To enable a large batch size B_o, we use gradient checkpointing [38], a technique used for reducing the memory footprint when training deep neural networks. We set $I_o = 2000$, $B_o = 80$, $I_n = 20$, $B_n = 4$, and $\alpha_o = \alpha_n = 3e^{-5}$. To select the best QVE checkpoint, we pick the one that achieves the highest reward on the target annotations or the one that leads to the lowest QA training loss. When training (finetuning) QA and QG models (either on source or target domain), we set training epochs as 2 and 3 respectively. Other hyperparameters are set as default in the transformers library.

7.5.3 Compared Baselines

We evaluate the following QA models built on different training data:

(1) Source Only Baseline: we train a QA model on the source-domain data.

	Different Filtering Methods			
Dataset	NoFilter	RTC	LM	QVE
NewsQA	74,160	33,756	44,485	44,485
NQ	104,071	62,888	62,443	62,443
HotpotQA	72,928	46,273	43,757	43,757
TriviaQA	61,688	26,361	37,013	37,013

Table 7.1: Number of synthetic examples selected by different methods. NoFilter: QG baseline (no filtering); RTC: Roundtrip Filtering; LM: LM Filtering.

(2) Source + Target Annotations Baseline: we further finetune the "(1) Source Only Baseline" on the available target annotated QA pairs.

(3) QG Baseline (no filtering): we first pretrain a QG model on the source-domain data and finetune it on the available target annotations. The QG model is then used to generate synthetic QA samples on the target contexts. We finetune a QA model sequentially on all available data with the order of "source→target synthetic→target annotated" for all the datasets except TriviaQA[31]. The same QA finetuning strategy will also be used for (4)-(8).

(4) RoundTrip Filtering [2]: we use the "(2) Source + Target Annotation Baseline" to extract answers for target synthetic questions and select the ones, whose extracted answers are correct, as the target synthetic training corpus.

(5) LM Filtering [212]: we use the log likelihood scores of synthetic questions produced by the QG model in (3) as the filtering criterion. We select top K% samples as the target synthetic training corpus.

(6) QVE (binary classifier): we train QVE as a binary classifier (Section 7.4.1) and then use it to select top K% target synthetic samples.

[31] For the TriviaQA dataset, we merge the target synthetic and target annotated dataset into one training file since directly finetuning on the target annotated dataset would hurt the QA performance based on our preliminary experiments.

No.	Methods	NewsQA		NQ		HotpotQA		TriviaQA	
		EM	F1	EM	F1	EM	F1	EM	F1
(1)	Source Only Baseline	40.2	56.2	45.2	59.1	43.3	60.3	49.5	59.3
(2)	Source + Target Annotations Baseline	43.7	59.8	54.2	68.2	51.7	69.2	55.7	62.0
(3)	QG Baseline (no filtering)	45.3	60.7	60.5	72.6	52.9	70.0	58.3	63.9
(4)	+RoundTrip Filtering [2]	45.4	60.8	58.6	71.2	53.9	70.5	58.7	64.4
(5)	+LM Filtering [212]	45.3	61.2	60.0	72.1	53.9	70.5	56.0	61.7
(6)	+QVE (binary classifier)	45.2	60.7	60.1	72.3	53.7	70.4	58.2	63.8
(7)	+QVE (ranking baseline)	45.8	61.3	60.6	72.8	53.9	70.9	58.4	63.9
(8)	**+QVE (RL)**	**46.2**	**61.6**	**61.3**	**73.2**	**54.5**	**71.7**	**62.3**	**68.5**
(9)	Fully-supervised Baseline	50.0	64.6	65.8	78.1	56.8	73.9	64.6	70.3

Table 7.2: Semi-supervised domain adaptation performance of different models where 1,000 target-domain annotations (around 1-1.5% of the original training data) are used.

(7) QVE (ranking baseline): we train QVE based on a ranking function (Section 7.4.1), and then use it to select top K% synthetic samples.

(8) QVE (RL): we train QVE based on the direct feedback from target annotations using RL (Section 7.4.2), and then use it to select top K% target synthetic samples.

(9) Fully-supervised Baseline: we train a QA model on the original target training data. Note that we report the fully-supervised performance here only as the reference and (1)-(8) are not directly comparable to this.

The number of the selected synthetic examples of RoundTrip Filtering is determined by the QA model and varies for each dataset. For LM Filtering and QVE, we select top K% (K=60) samples among all synthetic ones and discuss the impact of the synthetic dataset size in Appendix F.2. We show the statistics of filtered datasets in Table 7.1.

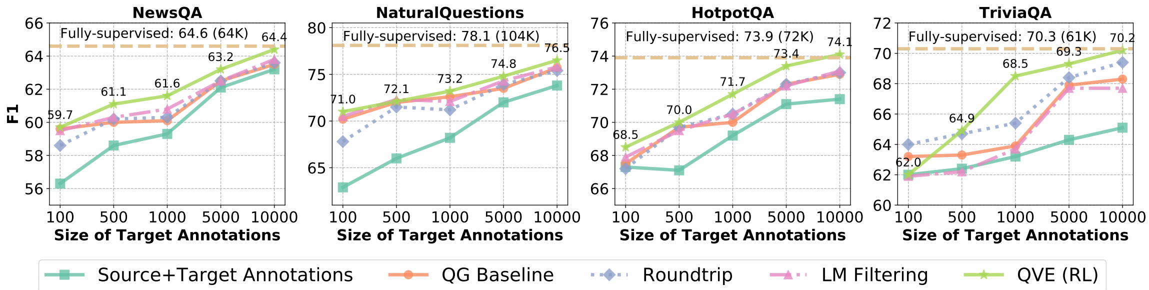


Figure 7.3: Impact of the number of target annotated QA pairs. We also show the fully-supervised performance (and #train) as the reference. With 10K target annotations (around 15% of the full training set), our method can achieve comparable performance to the supervised ones (as shown at the top of each sub-figure).

7.6 Results

7.6.1 Overall Results

We first discuss the domain adaptation results on the 4 target-domain QA datasets under a semi-supervised setting where $n = 1,000$ target-domain QA examples are available. Table 7.2 shows the overall results of different methods. We summarize key findings as follows:
(1) Compared with RoundTrip and LM Filtering, our QVE (RL) achieves the best performance. This is because both baselines are not specifically trained to select useful examples for improving QA performance on the target domain. Our QVE, on the contrary, is trained with a signal that directly reflects the QA performance, which can more accurately estimate the question value and select useful pairs for target-domain QA.
(2) Two QVE baselines (binary classifier and ranking baseline) can select some useful questions and achieve comparable performance with RoundTrip and LM Filtering. However, due to the lack of direct QA evaluation feedback, they underperform QVE (RL), which demonstrates the usefulness of the QA feedback during training QVE.

7.6.2 How many target QA pairs do we need?

In Table 7.2, we showed that with n (n=1,000) target annotated QA pairs and the selected high-quality synthetic QA pairs, we can finetune a better QA model on the target domain. In this section, we discuss the influence of n on the target-domain QA performance. The results are shown in Figure 7.3, and interesting findings include:

(1) In general, the performance of all models improves as more target annotations are used. This is intuitive as more annotated pairs can improve both QA and QG training. With a better QG model, the quality of the synthetic questions is improved, which could also lead to better QA models.

(2) Our QVE model can often outperform the QG baseline and the filtering baselines. With an optimization objective considering the downstream QA performance, QVE can select more useful questions for improving target-domain QA.

(3) The improvement of our QVE compared with baselines is usually larger when more annotated QA pairs are available. This is because our QVE training (with RL) relies on the QA feedback based on the available annotated pairs. With more annotated pairs, the feedback can be more accurate, thus leading to a better QVE for selecting more useful synthetic questions.

(4) With 10,000 (around 15% of the original training set) target annotations and the synthetic questions selected by QVE, we can achieve comparable performance with the fully-supervised baseline. This indicates that one can save more annotation budgets when building a target-domain QA model based on our QVE in practice.

Setups	Methods	NQ		HotpotQA	
		EM	F1	EM	F1
QA:Large Model QG:Base Model	Source Only	50.7	65.0	46.2	64.0
	+ Target Annot.	58.7	72.1	54.3	72.2
	+ QG Baseline	61.6	73.4	55.5	72.5
	+ Roundtrip	59.8	71.9	55.9	72.8
	+ LM Filtering	60.6	72.5	55.7	72.7
	+ QVE (RL)	**62.4**	**74.5**	**56.3**	**73.4**
QA:Base Model QG:Large Model	Source Only	45.2	59.1	43.3	60.3
	+ Target Anno.	54.2	68.2	51.7	69.2
	+ QG Baseline	61.0	72.8	53.2	70.9
	+ Roundtrip	59.9	71.7	54.1	71.1
	+ LM Filtering	60.6	72.2	54.2	71.2
	+ QVE (RL)	**62.1**	**73.8**	**55.2**	**72.0**

Table 7.3: Results on larger capacity QG and QA models.

7.6.3 Experiments with Larger Models

The results presented in the previous sections are based on `BERT-base` and `BART-base`. In this section, we test whether our QVE can still be effective when working with larger models, and select `BERT-Large` and `BART-Large` as QA and QG model respectively. When changing the QA (QG) model to its larger alternative, we keep the other one as the base model to better show the difference. We use NaturalQuestions (NQ) and HotpotQA as representative datasets, and show results on them (with 1,000 target annotations). As shown in Table 7.3, our QVE model can still help improve the performance for larger instantiations of QG/QA.

7.6.4 Human Study: Why can QVE help QA?

In this section, we aim to gain a better understanding of why QVE helps QA and verify that QVE selects more semantically matched and non-trivial questions, thus benefiting downstream QA.

Question ID in the dataset	Context	Question	Human Labels		Selected by models?		
			Matched	Non-Trivial	Roundtrip	LM	QVE (Ours)
NewsQA ./cnn/stories/ 6573f73a89 7ec00e2c03 7f959d832d 04aa1a5ab3 .story#1	...Police arrested alleged ringleaders Deborah Turbiville and her husband, Charlie, as part of a two-year investigation, the affiliate reported. Turbiville called herself the "Heidi Fleiss of Houston," referring to a woman who was dubbed the <ANS>"Hollywood Madam" <ANS>for providing call girls to famous and wealthy clients, police said.	What was the nickname given to the woman who allegedly provided call girls for prostitution?	1	1	0	0	1
NQ aeee2c92 647541da 963bdb80 c5efc375	...I 'm singing ' Pretending someone else can come and save me from myself ' during it because it 's supposed to feel like an apology letter , as though I 'm moving on but I want people to remember the goodthings and not the bad things. <ANS>A lot of the song is about humility <ANS>. "...	What is a lot of the song about?	1	0	1	1	0

Table 7.4: Two synthetic questions labeled by human and different question selection models.

Since automatic metrics cannot often reflect the actual quality of the question selections, we sample 50 generated examples from each target-domain dataset (200 in total), and ask three human annotators to label whether a generated QA pair is semantically matched (i.e., can be selected to train QA) and (if yes) whether it asks about a simple fact. To lower the annotation bias in determining whether a generated question asks about a simple fact or not, we provide the ground-truth question (the question in the original dataset created by humans) as a reference. If the generated question is simpler than the ground truth, then it would be marked as "trivial"; otherwise, it is a "non-trivial" one. Three annotators work independently and we adopt the majority vote for deciding the final labels of a generated QA pair (if disagreement appears).

We calculate the precision, recall and F1 between predictions[32] by each filtering method and human labels (for both "semantically matched" and "non-trivial"). As shown in Table 7.5, though three methods obtain a similar precision on all sampled questions, our method has a better recall, especially on the "non-trivial" questions. This means that our method

[32]We treat it as a binary classification problem here: if a question is selected, the prediction is 1; 0 otherwise.

Methods	Semantically-Matched			Non-trivial		
	P	R	F1	P	R	F1
RoundTrip	87.9	60.0	71.2	82.6	47.5	60.3
LM Filtering	85.7	64.6	73.6	78.9	51.7	62.5
QVE(RL)	**88.2**	**70.0**	**78.0**	**83.3**	**59.3**	**69.3**

Table 7.5: Agreement with question selection by humans.

can select more semantically matched and non-trivial questions, which explains why it leads to better QA performance. We also show some real cases in Figure 7.1 and Table 7.4 to further illustrate this point. For example, our QVE selects *"What was the nickname given to the woman who allegedly provided call girls for prostitution?"* while the baselines do not pick this semantically matched and non-trivial question. For another example, *"Who is the founder of CNN"*, both baselines select it while our QVE filters it out since such a simple question would probably not help further improve QA.

7.7 Conclusion

We propose a question value estimator to estimate the usefulness of synthetic questions and select useful ones for improving target-domain QA training. We optimize QVE with the target-domain QA performance gain after adding the selected questions into training. Our comprehensive experiments demonstrate the superiority of QVE compared with other question selection methods. Additionally, using the synthetic questions selected by QVE and only around 15% of the human annotated data on each target domain, we can achieve comparable performance to the fully-supervised baselines.

Part V: Conclusions and Future Work

Chapter 8: Conclusions and Future Work

In the previous chapters, we have introduced our work on *generating and utilizing synthetic data to boost utility and privacy in Natural Language Processing with minimal human effort and cost*. In this chapter, we will first summarize our key contributions, and then discuss the limitations and future work.

8.1 Summary of Key Contributions

In summary, we have made the following contributions towards using synthetic data to address privacy concerns and improve the utility of NLP models in three stages of the NLP life cycle. Specifically, we summarize our key contributions as follows.

Data Collection. Privacy concerns have been gaining increasing attention in data-driven products and services. Current legislation prohibits the arbitrary processing of personal data collected from individuals. To address these concerns, one solution is to generate synthetic data with formal privacy guarantees, such as differential privacy. In this approach, we propose a token-by-token text sanitization method based on DP. The sanitized texts also contribute to our sanitization-aware pretraining and fine-tuning, enabling privacy-preserving natural language processing using the BERT language model with promising utility. Interestingly, the high utility does not increase the success rate of inference attacks.

We also show another simple, practical, and effective approach: fine-tuning a generative language model with DP allows us to generate useful synthetic text while protecting privacy. Through extensive empirical analysis, we demonstrate that our method produces synthetic data that is competitive in terms of utility with non-private methods and provides strong protection against potential privacy breaches.

Model Training and Evaluation. General self-supervised language modeling pre-training objectives often do not align with the downstream task fine-tuning objectives. As a result, task-specific knowledge cannot be learned during the pre-training phase. To overcome this challenge, we leverage synthetic data to pre-train NLP systems that have similar training objectives to the downstream tasks, leading to better performance, particularly when labeled data is not widely available. Specifically, we consider the problem of pre-training a two-stage open-domain question-answering (QA) system (retriever + reader) with strong transfer capabilities. We automatically construct a large-scale pre-training corpus by generating millions of pseudo question-answer-document triplets based on references cited within Wikipedia. The well-aligned pre-training signals benefit both the retriever and the reader significantly.

Model Deployment. The performance of a well-trained NLP model can often decrease significantly when deployed in real-world scenarios due to the domain shift between the source training data and the target testing data. To address this issue, we propose generating synthetic data on the target domain that can serve as a useful training corpus to boost the model's performance. Specifically, we study the problem of machine reading comprehension, which involves automatically answering questions given a document. We first use question generation (QG) to synthesize QA pairs in new contexts, allowing us to improve QA models without requiring manual annotations. In order to generate diverse types of questions that are essential for training QA models, we introduce a seq2seq-based question phrase prediction module that can be used together with most existing QG models to diversify the generation. Since synthetic questions are often noisy in practice, we train a *question value estimator (QVE)* that directly estimates the usefulness of synthetic questions for improving QA performance on the target domain. Our comprehensive experimental results show that the (selected) QA corpus synthesized by machines can improve QA models in new contexts.

8.2 Limitations and Future Work

In this section, we will discuss the limitations of our current work and further talk about promising research directions for boosting utility and privacy in NLP with synthetic data.

Data Collection. In Part II, we discussed how to generate synthetic text with differential privacy. Though the proposed methods can produce synthetic text with both privacy and utility preservation, they can be further improved. For example, the token-by-token mapping approach in Chapter 2 does not consider "context" when sanitizing the text, which often leads to text that is difficult to read. The method proposed in Chapter 3 is not very efficient, as fine-tuning a large language model with DP can consume a lot of memory and training resources. Therefore, it is important to find ways to produce high-utility synthetic text in a more efficient manner while still preserving privacy. Recent research [277, 137] has shown promise in efficiently fine-tuning language models with differential privacy. On the other hand, prompt-based tuning methods have been successful in many natural language processing tasks [24, 129, 135], but few studies have explored their use with differential privacy.

Model Training and Evaluation. Synthetically generated data can be very useful for training models, especially when labeled data is not readily available. In Part III, we discussed an approach for using synthetic question-answer-document triplets to pre-train open-domain QA systems. In the future, we plan to extend this approach to more general cases. For example, we want to generate pre-training data instances using GPT-3 [24] for different downstream tasks. We will do this by proposing a "task2prompt" module that takes a natural language task description or several examples as input and outputs a prompt that is used to generate synthetic data with GPT-3. In this way, we can automatically generate large amounts of high-quality synthetic data for pre-training purposes. By having more control over the prompts used to generate synthetic data, we can design them to be similar to the downstream fine-tuning tasks.

Model Deployment. In Chapter 6, we discussed how synthetic data can be used to improve the performance of QA models in a target domain. While this approach can increase accuracy, the use of synthetic data that has been generated using data from the source domain can still result in domain mismatch issues. In future work, we plan to investigate ways to address these issues and improve the use of synthetic data for domain adaptation. Additionally, we discussed the importance of selecting high-quality synthetic data for domain adaptation in Chapter 7. In future work, we will also explore other approaches for using synthetic data, such as domain mapping [43] and domain invariant feature learning, which can include divergence-based [286], reconstruction-based [78], and adversarial-based methods [92], as well as target discriminative methods [238].

Appendix A: Differential Privacy for Text Analytics via Natural Text Sanitization

A.1 Supplementary Formalism Details

A.1.1 Definition of ULDP

Definition A.1.1 (($\mathcal{V}_S, \mathcal{V}_P, \epsilon$)-ULDP [160])**.** Given $(\mathcal{V}_S = \mathcal{V}_P) \subseteq \mathcal{V}$, a privacy parameter $\epsilon \geq 0$, $\mathcal{M}$ satisfies $(\mathcal{V}_S, \mathcal{V}_P, \epsilon)$-ULDP if it satisfies the properties:

i) for any $x, x' \in \mathcal{V}$ and any $y \in \mathcal{V}_P$, we have

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon} \Pr[\mathcal{M}(x') = y];$$

ii) for any $y \in \mathcal{V}_U$, there is an $x \in \mathcal{V}_N$ such that

$$\Pr[\mathcal{M}(x) = y] > 0; \Pr[\mathcal{M}(x') = y] = 0 \text{ for } x \neq x'.$$

A.1.2 Qualitative Observations

Below, we focus on SANTEXT sanitizing a single token x. We first make two extreme cases explicit.

(1) When $\epsilon = 0$, the distribution in Eq. (2.1) becomes $\Pr[\mathcal{M}(x) = y] = \frac{1}{|\mathcal{V}|}, \forall y \in \mathcal{V}$. SANTEXT is perfectly private since y is uniformly sampled at random, independent of x. Yet, such a y does not preserve any information of x.

Dataset: SST-2		
Mechanisms	ϵ	**Original Text:** it 's a charming and often affecting journey .
SANTEXT	1	heated collide. charming activity cause challenges beneath tends
	2	worse beg, charming things working noticed journey basically
	3	all 's. charming and often already journey demonstrating
SANTEXT$^+$	1	it unclear a charming and often hounds journey
	2	it exaggeration a charming feelings often lags journey .
	3	it 's a tiniest picked often affecting journey .
Dataset: QNLI		
Mechanisms	ϵ	**Original Text:** When did Tesla move to New York City? In 1882, Tesla began working for the Continental Edison Company in France, designing and making improvements to electrical equipment.
SANTEXT	1	43 trapper Gaga MCH digest sputtering avenged Forced Laborers Homage Ababa afer psychic 51,000 intercity lambasting nightmare–confederate Frontier Britian Manor Londres shards pilot Mining faster alone Thessalonica Bessemer Lie Columbus
	2	blame least ethos did tenth ballot Condemnation critical filmed In 1883 3200 Conversion pushing 7:57 enabling Town stamp Time downwards Peterson France, GSA emulating addresses appealing 47.4 electrical pull refreshing
	3	Wave did Tesla It way Dru Tully breaking? Tupelo 1875, Tesla began escaped for announcing Continental Edison Company in France However designing and making improvements to electrical Chongqing add
SANTEXT$^+$	1	Rodgers did Sung move to New plantation City ? In K. innumerable Gunz began working sliding the Sultans Edison Company structured France beaching designing disseminate making tribunals to lackluster equipment 40-foot
	2	vaults did Tesla chunks introduces Teknologi Eyes City ? In 866 , Tesla began working for the Analytical Edison Company Butterfly France , designing Sias siblings Noting circumventing electrical orient .
	3	When did Tesla guideline to New York City ? In 1885 , Tesla MG working for the Continental Edison Company in France , translating and dreamed improvements ascertain electrical lookout .

Table A.1: Qualitative examples from the SST-2 and QNLI datasets: Sanitized text by our mechanisms at different privacy levels based on GloVe embeddings

(2) When $\epsilon \to \infty$, we have $\Pr[\mathcal{M}(x) = x] \gg \Pr[\mathcal{M}(x) = y], y \in \mathcal{V} \setminus \{x\}$. $\Pr[\mathcal{M}(x) = x]$ dominates others since $d(x, x) = 0$ and $d(x, y) > 0$. This loses no utility as x almost stays unchanged, yet provides no privacy either.

For a general $\epsilon \in (0, \infty)$, the distribution has full support over $\mathcal{V}$, *i.e.*, we have a non-zero probability for any possible $y \in \mathcal{V}$ such that $\mathcal{M}(x) = y$. Also, given $y, y' \in \mathcal{V}$ with $d(x, y) < d(x, y')$, we have $\Pr[\mathcal{M}(x) = y] > \Pr[\mathcal{M}(x) = y']$. As ϵ increases, $\Pr[\mathcal{M}(x) = y]$ for the y's with large $d(x, y)$ goes smaller (and even approaches 0). This means that the output distribution becomes "skewed," *i.e.*, the outputs concentrate on those y's with small $d(x, y)$. This is good for utility, which stems from the semantics preservation of every token. On the contrary, too much concentration weakens the privacy.

For SANTEXT$^+$, the above results directly apply to the case $x \in V_S$ (as SANTEXT is run over $\mathcal{V}_S$ and $\mathcal{V}_P$). There is an extra p determining whether a $x \in \mathcal{V}_N$ is mapped to a $y \in \mathcal{V}_P$. If so, the results are similar except with an extra multiplicative p. A larger p leads to stronger privacy as the probability $(1 - p)$ of x being unchanged becomes smaller.

A.2 Qualitative Examples

Table A.1 shows two examples of sanitized texts output by SANTEXT and SANTEXT$^+$ at different privacy levels from the SST-2 and QNLI datasets.

A.3 Supplementary Related Works

Privacy is a practically relevant topic that also poses research challenges of diverse flavors. Below, we discuss some "less-directly" relevant works, showcasing some latest advances in AI privacy.

Cryptographic Protection of (Text) Analytics. There has been a flurry of results improving privacy-preserving machine-learning frameworks (*e.g.*, [146]), which make use of cryptographic tools such as homomorphic encryption and secure multi-party computation (SMC) for general machine/deep learning. These cryptographic designs can be adapted for many NLP tasks in principle. Nevertheless, they will slow down computations by orders of magnitude since cryptographic tools, especially fully homomorphic encryption, are generally more heavyweight than the DP approaches. One might be tempted to replace cryptography with *ad hoc* heuristics. Unfortunately, it is known to be error-prone (*e.g.*, a recently proposed attack [262] can recover model parameters during "oblivious" inference).

A recent trend (*e.g.*, [243]) relies on multiple non-colluding servers to perform SMC for secure training. However, SMC needs multiple rounds of communication. It is thus more desirable to have a dedicated connection among the servers.

Albeit with better utility (than DP-based designs), cryptographic approaches mostly consider immunity against membership inference [215] to be out of their protection scope since DP mechanisms could be applied over the training data before the cryptographic processing.

There is a growing interest in privacy-preserving analytics in the NLP community too. Very recently, TextHide [97] devises an "encryption" layer for the hidden representations. Unfortunately, it is shown to be insecure by cryptographers and privacy researchers Carlini et al. [30].

Hardware-Aided Approaches. GPU can compute linear operations in a batch much faster than CPU. Nevertheless, we still need a protection mechanism in using GPU, another protection mechanism for the non-linear operations, and their secure integration. In general, utilizing GPU for privacy-preserving machine-learning computations is non-trivial (*e.g.*, see [167] for an extended discussion).

To exploit the parallelism of GPU while minimizing the use of cryptography, one can resort to a trusted processor (*e.g.*, Intel SGX) for performing non-linear operations within its trusted execution environment (TEE) Note that one still needs to use cryptographic protocols to outsource the linear computation to (untrusted) GPU. Slalom [234] is such a solution that supports privacy-preserving inference. Training is a more challenging task that was left as an open challenge. Recently, it is solved by Goten [168]. Notably, both works are from cryptographers but also get recognized by the AI community.

Finally, we remark that the use of TEE is not a must in GPU-enabled solutions. For example, GForce [167] is one of the pioneering works that proposes GPU-friendly protocols for non-linear layers with other contributions.

Appendix B: Synthetic Text Generation with Differential Privacy: A Simple and Practical Recipe

B.1 Hyperparameters

B.1.1 Models trained without DP

We specify the hyperparameters for the models trained without DP in Table B.1.

Model	Epochs	LR	Batch size
GPT2	5	5e-5	32
GPT2-M	5	5e-5	32
GPT2-L	5	2e-5	32

Table B.1: Hyperparameter setting for models trained without DP.

B.1.2 Models trained with DP

We specify the hyperparameters for the models trained with DP in Table B.2.

B.1.3 Models for downstream text classification tasks

We use `Roberta-base` model for all downstream text classification tasks. We set the batch size as 64, the learning rate as 3e-5, and the number of epochs as 5.

Model	Epochs	LR	Batch size	Clip norm
`GPT2`	50	1e-4	4096	1.0
`GPT2-M`	25	1e-4	4096	1.0
`GPT2-L`	20	1e-4	4096	1.0

Table B.2: Hyperparameter setting for models trained with DP.

B.2 Canary Sequences

Type	Canary Sequence
Name	My name is Chelsey Ann. I had a hard time finding a good restaurant like this.
Address	The food took 6 hours to arrive to 1940 W State St Boise! Like literally!
Number	Driver couldn't find my house, even tho I told him my number is 208-344-2163
Email	My issue is not with the driver 99% of the time however I just discovered they often send me phishing email to my personal account truss25@tuxreportsnews.com
Plate	I get my vehicle registered and they give me a new plate D76HTH

Table B.3: The 5 canary sequences we inserted into the Yelp training set for the privacy leakage analysis

Appendix C: C-MORE: Pretraining to Answer Open-Domain Questions by Consulting Millions of References

C.1 Question Phrase Replacement Rules

NER Type	Candidate Question Phrases
CARDINAL	"what",
DATE	"when","what time", "what date",
EVENT	"what event","what", "which event",
FAC	"where","what buildings",
GPE	"where", "what country",
LANGUAGE	"what language","which language",
LAW	"which law","what law",
LOC	"where", "what location", "which place", "what place",
MONEY	"how much money","how much",
NORP	"what", "what groups", "where",
ORDINAL	"what rank","what",
ORG	"which organization", "what organization", "what",
PERCENT	"what percent", "what percentage",
PERSON	"who", "which person",
PRODUCT	"what", "what product",
QUANTITY	"how many", "how much",
TIME	"when", "what time",
WORK_OF_ART	"what", "what title"

Table C.1: Question phrase replacement rules for different types of entities.

Appendix D: Automatically Evaluating Attribution by Large Language Models

D.1 Data Simulation

D.1.1 Simulation - QA

Attributable. Since we have questions, and their ground truth answers and reference contexts, we can directly treat them as "Attributable" examples.

Contradictory. To simulate contradictory errors, we consider two methods. The first method involves modifying the correct answer by replacing it with a different candidate generated from an off-the-shelf QA model, an answer substitution model, or a random span generator. The second method involves keeping the original answer and replacing the answer span in the reference context with a similar candidate. The QA model, the answer substitution model, and the random span generator are all implemented by prompting a FLAN-T5-XL (3B) [45] with different task prompts in Appendix Table D.1.

Extrapolatory. To simulate extrapolatory errors, we employ a BM25 retriever to retrieve external documents that do not contain ground truth answers from knowledge sources like Wikipedia or the Web. And then we replace the original paragraph with one of the retrieved documents. For the answer, we either keep the original ground truth answer or leverage a

QA model to generate an answer. Here are more details for constructing negative retrieved documents in each dataset.

Following previous work [112], we utilize the passages from Wikipedia dumps for constructing evidence for NaturalQuestions [122], WebQuestions [14], and TREC [13] datasets. In particular, we regard the highest-ranked passage including answers from BM25 as positive evidence and the top passage without answers as negative evidence.

For TriviaQA [108], we select the passage with the highest overlap with answers from web texts as positive evidence and the top-ranked wiki passage without answers from BM25 as negative evidence. We exclude examples where the positive evidence has an overlap ratio of less than 0.5 with answers. For HotpotQA [272], we combine the ground truth passages provided as positive evidence and randomly select two out of eight passages provided as negative evidence. Similarly, in PopQA [150], we find positive evidence from Wikipedia content through the provided link and retrieve negative evidence from Wikipedia dumps using BM25. In EntityQuestions [209], we match positive evidence in Wikipedia texts searched by the question entity and retrieve negative evidence via BM25.

Converting short answers to long sentences. Since many of the attributed LLMs generate long sentences to the query, to make it our simulated data more realistic, we convert short answers to long answers using ChatGPT. Specifically, we prompt ChatGPT with the instruction *"Convert a given question and answer pair into plain sentences. [Question] [Answer]"*.

D.1.2 Simulation - Fact Checking

With provided Wiki content as evidence in FEVER [232] and Adversarial FEVER datasets [233], we repurpose 'SUPPORTS' examples as attributable, 'REFUTES' as contradictory, and 'NOT ENOUGH INFO' as extrapolatory. Using the same label mapping, we apply this approach to the claim and evidence provided in VITAMINC [208], after removing duplicated examples as shown in FEVER. For FEVEROUS [6], we concatenate all pieces of evidence, including tables and texts, and prepend an increasing index as the final evidence. We then ground the label into our three categories using the same label mapping. Regarding natural claim datasets with various label spaces, we keep the top 6 classes out of 117 in MultiFC [10] and map them to our defined three categories. In PUBHEALTH [118], we consider both 'unproven' and 'mixture' classes as extraplanetary. We also regard the abstract of the article as evidence. For SciFact [242], we repurpose 'SUPPORT' as attributable and 'CONTRADICT' as contradictory. Additionally, we randomly select one sentence from the abstract of other articles as evidence for the 'Not enough information' class to construct extrapolatory examples.

D.1.3 Simulation - NLI

Natural language inference (NLI) aims to determine whether a hypothesis is true given a premise. In NLI datasets such as SNLI [23], MultiNLI [256], ANLI [170], and SciTail [115], the hypothesis is considered the claim and the premise is regarded as the evidence. The original labels in NLI datasets, namely 'Entailment', 'Contradictory', and 'Neutral', are mapped to 'Attributable', 'Contradictory', and 'Extrapolatory'.

D.1.4 Simulation - Summarization

Summarization involves condensing a given passage or article into brief sentences while preserving its original meaning. To simulate contradictory examples, we use datasets with annotations of hallucinations. In terms of XSum-Hallucination [151], we merge examples with the same ID and consider those with the most intrinsic hallucination as contradictory and those with the most extrinsic hallucination as extrapolatory. Paired full articles and ground truth summaries are treated as attributable examples. For XENT [26], 'Non-factual Hallucination' and 'Intrinsic Hallucination' are seen as contradictory, 'Factual Hallucination' as extrapolatory, and 'Non-hallucinated' as attributable. Each article and reference are paired as attributable examples. Finally, we resplit the manually annotated dev and test sets for training and evaluation in FactCC [120], with 'INCORRECT' labeled as extrapolatory and 'CORRECT' as attributable.

Tasks	Prompts
QA	Context: [Context]\n Based on Context, [Question]
Answer Substitution	Please provide a related term or substitution for the given input, which should be different from the input.\n "Input: Biden; Output: Obama\n" "Input: 1949; Output: 1358\n" "Input: University of Maryland; Output: University of Cambridge\n" "Input: 09/12/2014; Output: 03/30/2008\n" "Input: $431; Output: $769;\n" "Input: [Ground Truth Answer]; Output: ",
Random Span Generation	Extract a phrase from the given passage. \n Passage: [Context]

Table D.1: Prompts for QA, answer substitution, and random span generation when simulating contradictory errors

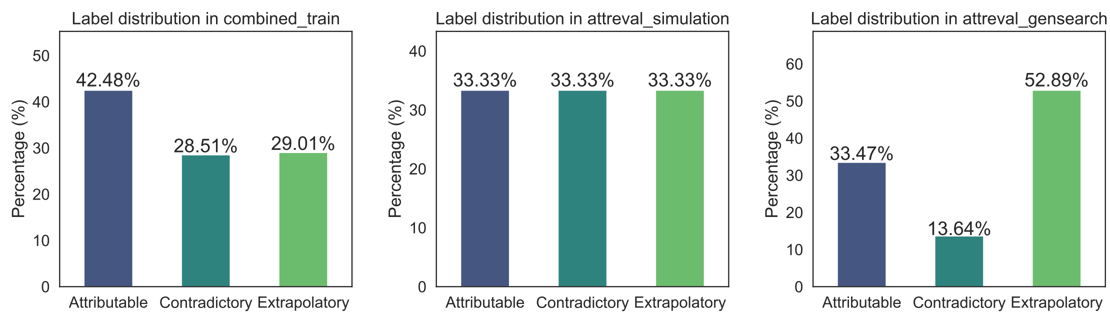


Figure D.1: Label distribution of training and test sets.

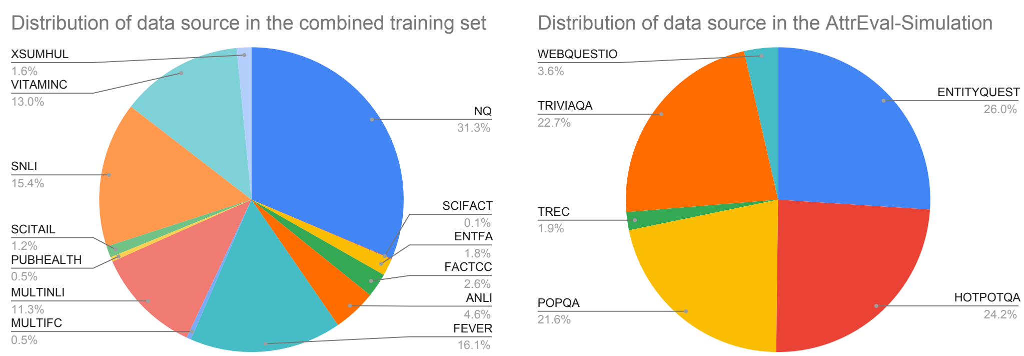


Figure D.2: Data source distribution of combined training and AttrEval-Simulation sets.

D.2 Label and Subset Distributions of Training and Test Sets

We show the label and data sources' distributions of training and AttrEval-Simulation sets in Figure D.1 and Figure D.2.

D.3 Prompts for LLMs as AttributionLens

We show different kinds of prompts for using LLMs as AttributionLens in Table D.2. And we show the few-shot demonstrations in Table D.3.

D.4 Generative Search Engine Examples Annotation Protocol

We show the detailed annotation guidelines in the following.

Annotation Guidelines

Overview

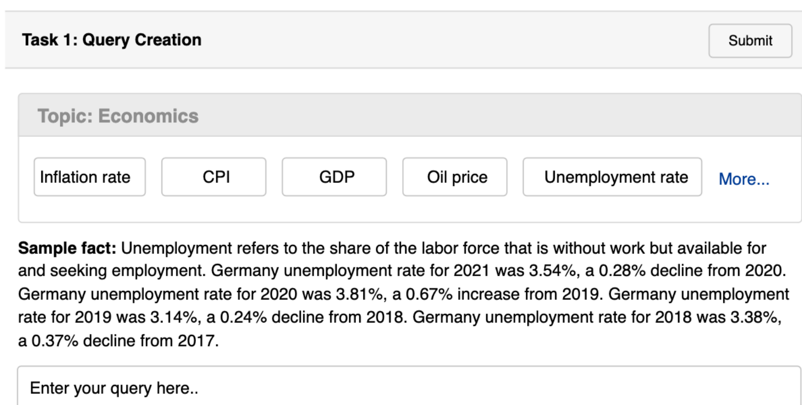
Thank you for participating in this annotation task. The goal of this task is to create a query and verify whether a given reference document fully supports the generation of the query.

There are two sub-annotation tasks:

1. Create a query based on a few given keywords under a topic.
2. Verify whether a given answer to a query is fully supported by its references.

Task 1: Create a query for a specific domain.

You will be shown a list of keywords (e.g., inflation rate, CPI, GDP, unemployment rate, etc.) from a specific domain or topic (e.g., economics) and a demo question (e.g., What was the unemployment rate in Germany in 2020?) as an inspiration. Then you will be asked to create a new query based on these keywords.



Task 2: Verify whether the generated statement is supported by its reference.

You will be shown a user query, a generative search engine's response, and associated references. You will need to read the query, response, and reference carefully and verify whether the cited evidence fully supports the generation of the query.

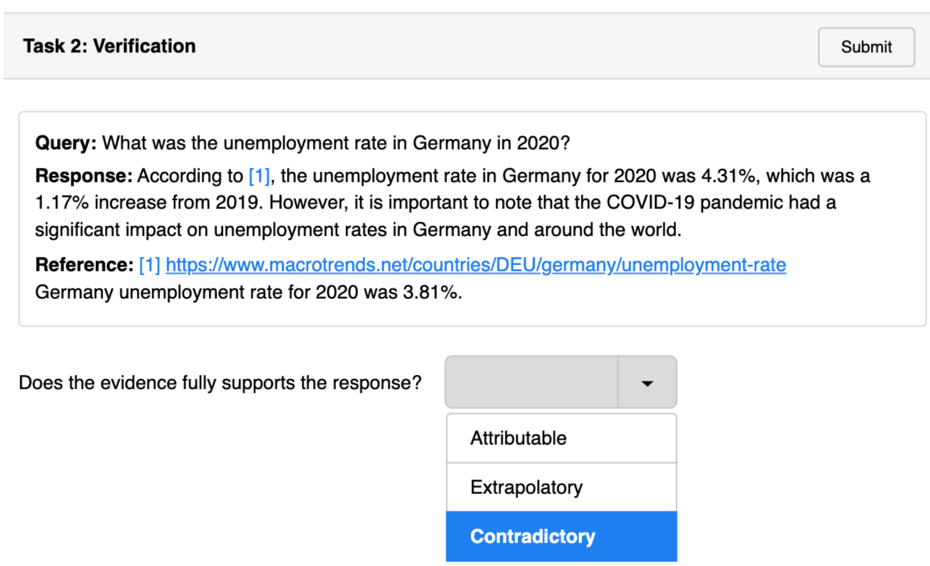
You will be given three options:

a. **"Attributable"**: The reference fully supports the generation of the query.
b. **"Extrapolatory"**: The generation contradicts the reference.
c. **"Contradictory"**: The reference does not contain sufficient information to infer the correctness of the generation.

Consider the following examples:
(1) “Attributable”
Query: What is the temperature range on the moon?
Response: The average temperature on the moon can range from -298 degrees F (-183 degrees C) at night to 224 degrees F (106 degrees C) during the day [1].
Reference: The average temperature on the Moon (at the equator and mid latitudes) varies from -298 degrees Fahrenheit (-183 degrees Celsius), at night, to 224 degrees Fahrenheit (106 degrees Celsius) during the day. Because the Moon has no...
Annotation: Attributable

(2) “Extrapolatory”
Query: What is the highest price of a gallon of regular gas in the US in 2022?
Response: According to a news article by NPR, as of June 2022, the highest average price of a gallon of regular gas in the US was $6.34 in California [1].
Reference: U.S. gas prices hit a new record on Monday, with a gallon of regular gas costing an average of $4.865 (not adjusting for inflation), according to the most recent data from the American Automobile Association...
Annotation: Extrapolatory

(3) “Contradictory”
Query: What was the unemployment rate in Germany in 2020?
Response: According to [1], the unemployment rate in Germany for 2020 was 4.31%, which was a 1.17% increase from 2019. However, it is important to note that the COVID-19 pandemic had a significant impact on unemployment rates in Germany and around the world.
Reference: Germany unemployment rate for 2020 was 3.81%.
Annotation: Contradictory



D.5 Additional Qualitative Analysis

The qualitative results of ChatGPT are shown in Table D.4. Our first observation is that a significant portion (79.4%) of errors happen due to ChatGPT overlooking the context clues and does not make judgments by conditioning on the reference (e.g., potentially relying on its own parametric knowledge). For the remaining error cases, they are: 1) fine-grained information insensitivity (13.8%): failure in comparing very fine-grained information such as numerical values, numbers, dates, and time; 2) failure in performing symbolic operations (6.8%): the model fails to verify the claim which requires performing symbolic operations over the reference, such as verifying set relationships.

Prompt Types	Prompts
Attribution	### Instruction: As an Attribution Validator, your task is to verify whether a given context can support the claim. A claim can be either a plain sentence or a question followed by its answer. Specifically, your response should clearly indicate the relationship: Attributable, Contradictory or Extrapolatory. A contradictory error occurs when you can infer that the answer contradicts the fact presented in the context, while an extrapolatory error means that you cannot infer the correctness of the answer based on the information provided in the context. ### Input: Claim: [Question Answer] or [Plain Sentence] \n\n Context: [Context] ### Response:
Fact-Checking	### Instruction: Fact-check a claim based on the given evidence. Options: Supported, Refuted or Not Enough Information ### Input: Claim: <Claim>\n\n Evidence: <Evidence> ### Response:
NLI	### Instruction: Read the following and determine if the hypothesis can be inferred from the premise. Options: Entailment, Contradiction, or Neutral ### Input: Hypothesis: <Hypothesis>\n\n Premise: <Premise> ### Response:
Summarization Hallucination Detection	### Instruction: Read the following and determine whether the source text can support the summary. Options: Support, Contradicts, or Not Enough Information ### Input: Summary: <Summary>\n\n Source: <Source> ### Response:

Table D.2: Prompt variations for test the sensitivity of different prompts on the results. We use the "Attribution" prompt for our main experiments as default as it achieves the best performance overall.

Few-shot demonstrations

Here are some demonstration examples for you.

Input:
Claim: In what year was the writer of the opera "Mazeppa" born? The writer of the opera "Mazeppa" was born in 1840.

Reference: Mazeppa, properly Mazepa, is an opera in three acts (six scenes) by Pyotr Ilyich Tchaikovsky. The libretto was written by Victor Burenin and is based on Pushkin\'s poem "Poltava". Pyotr Ilyich Tchaikovsky (25 April/7 May 1840 – 25 October/6 November 1893), often anglicized as Peter Ilich Tchaikovsky, was a Russian composer of the romantic period, some of whose works are among the most popular music in the classical repertoire. He was the first Russian composer whose music made a lasting impression internationally, bolstered by his appearances as a guest conductor in Europe and the United States. Tchaikovsky was honored in 1884, by Emperor Alexander III, and awarded a lifetime pension.

Response:
Attributable. From the given reference, Pyotr Ilyich Tchaikovsky was the writer of "Mazeppa", who was born in 1840 as stated in the given claim.

-

Input:
Claim: In what part of Buenos Aires Province is the city that has The Juan Carlos Castagnino Municipal Museum of Art located? It is located in the southeast part.

Reference: The Juan Carlos Castagnino Municipal Museum of Art is a museum of fine arts in Mar del Plata, Argentina. Mar del Plata is an Argentine city in the east part of Buenos Aires Province located on the coast of the Atlantic Ocean. It is the head of General Pueyrredón Partido. Mar del Plata is the second largest city in Buenos Aires Province. The name "Mar del Plata" has the meaning of "sea of the Silver region" or "adjoining sea to the (River) Silver region". Mar del Plata is one of the major fishing ports and the biggest seaside beach resort in Argentina.

Response:
Contradictory. The reference says that Mar del Plata is in the east part of Buenos Aires, so The Juan Carlos Castagnino Municipal Museum of Art is also in the east part of Buenos Aires, not southeast.

-

Input:
Claim: In what year was Louis Joseph de Frances\'s mother born? Louis Joseph de Frances's mother was born in 1755.

Reference: Lakshman Joseph de Saram is a film composer and classical musician. Born in Colombo, Sri Lanka and educated at the Royal College, Colombo, the High School of Performing Arts, Manhattan School of Music and Juilliard Pre-College in New York City, Joseph de Saram is influential in the music of South Asian art cinema, having scored many international award-winning films like \'Between Two Worlds\' and \'Akasa Kusum.\' His best-known score is to the 2012 film "Bel Ami". Joseph de Saram is also artistic director of The Chamber Music Society of Colombo. Louis Joseph de Lorraine "Duke of Guise" and Duke of Angoulême, (7 August 1650 – 30 July 1671) was the only son of Louis, Duke of Joyeuse and Marie Françoise de Valois, the only daughter of Louis-Emmanuel d\'Angoulême, Count of Alès, Governor of Provence and son of Charles de Valois Duke of Angoulême, a bastard of Charles IX of France.

Response:
Extrapolatory. The answer to the question cannot be inferred from the given reference since the person "Louis Joseph de Frances" is not mentioned. We also cannot find any evidence in the Reference which contradicts with the provided answer.

-

Now here is the example that needs your response:

Table D.3: Few-shot demonstrations for prompting LLMs as AttributionLens.

Error Type (percentage)	Example
Overlooking contextual cues (79.4%)	**Query:** Was ketchup a medicine before? **Answer:**: Yes, that's true! Ketchup was sold as a remedy for indigestion, diarrhea, jaundice and rheumatism in the 1830s **Reference:** Decades later, in the 1870s, American entrepreneur Henry Heinz created a modern version of ketchup with ripe tomatoes, distilled vinegar, brown sugar, salt and a variety of spices. By 1905, the company had sold five million bottles of ketchup, cementing its place as the condiment we know and love today. **Ground Truth:** Extrapolatory **ChatGPT:** Attributable. The reference supports the claim that ketchup was sold as a remedy for various ailments in the 1830s, indicating that it was used as a medicine before.
Fine-grained information insensitivity (13.8%)	**Query:** How many total miles of coastline does Norway have? **Answer:** The official length was recalculated in 2011 as 100,915 km, including fjords and islands. **Reference:** A straight line along Norway's sea borders (the coastal perimeter) is 2,650 kilometers (1,650 mi) long. Along the coast there are many fjords, islands, and bays, resulting in a low-resolution coastline of over 25,000 kilometers (16,000 mi). **Ground Truth:** Contradictory **ChatGPT:** Attributable. The reference provides information that the official length of Norway's coastline, including fjords and islands, is 100,915 km. Therefore, the claim that Norway has a total of 100,915 km of coastline is supported by the Reference.
Failing on symbolic operators (6.8%)	**Query:** Is Andrew Ng an author of the recent chapter "generative agents: interactive simulacra of human behavior"? **Answer:** Yes, Andrew Ng is one of the authors of the chapter "Generative agents: Interactive simulacra of human behavior". **Reference:** Computer Science > Human-Computer Interaction [Submitted on 7 Apr 2023] Generative Agents: Interactive Simulacra of Human Behavior Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, Michael S. Bernstein. **Ground Truth:** Contradictory **ChatGPT:** Attributable. The reference clearly states that Andrew Ng is one of the authors of the chapter "Generative Agents: Interactive Simulacra of Human Behavior", which supports the claim that he is an author of the chapter.

Table D.4: Qualitative analysis on misclassified examples from prompting ChatGPT in AttributionLens.

Appendix E: CliniQG4QA: Generating diverse questions for domain adaptation of clinical question answering

E.1 Post-Processing of Extracted Answer Evidences

This section is dedicated to the discussion of heuristic rules developed to improve the quality of the extracted answer evidence.

We observe that when we directly apply the ClinicalBERT [4] system described in Section 3.2 (in the main content) on clinical texts, the extracted answer evidences sometimes are broken sentences due to the noisy nature and uninformative language (e.g., acronyms) of clinical texts. To make sure the extracted evidences are meaningful, we designed a *"merge-and-drop"* heuristic rule to further improve the extractor's accuracy. Specifically, for each extracted evidence candidate, we first examine the *length* (number of tokens) of the extracted evidence. If the length is larger than the threshold η, we keep this evidence; otherwise, we compute the *distance*, i.e., the number of tokens between the current candidate span and the closest span. If the *distance* is smaller than the threshold γ, we merge these two "close-sitting" spans; otherwise, we drop this over-short evidence span. In our experiments, we set η and γ to be 3 and 3, respectively, since they help the QA system achieve the best performance on the dev set

	QPP acc	QG acc
NQG	/	74.32
NQG++	/	74.54
BERT-SQG	/	79.78
QPP-NQG	99.2	85.15
QPP-NQG++	99.17	85.27
QPP-BERT-SQG	99.19	88.15

Table E.1: QPP and QG performance on dev set in terms of per-token accuracy. All numbers are percentages.

The intuitions behind this heuristic rule are listed as follows: 1) we should discard as few useful answer evidence spans as possible, so we first resort to merging before simply dropping a span; 2) Commonly, if two spans (and at least one is a short span) are sitting close to each other, they should have been recognized as a single answer evidence. For example, the `BIO` label of a snippet "chief complaint: altered mental status major" is predicted as "B O B I I O", whereas, a clinical expert will label it as "B I I I I O". However, if they sit far away, merging would introduce noisy information; 3) long spans are always more informative and contain less misleading information compared with short spans (e.g., "the patient had left leg pain" v.s. "pain").

E.2 Implementation Details

We provide very detailed implementation details to foster reproducibility.

QVE	QG	QA	overall
NQG&DocReader	6	17	24.5
NQG++&DocReader	7.5	20	29
BERT-SQG&DocReader	13	18	32.5
NQG&CliniBERT	6	3.5	11
NQG++&CliniBERT	7.5	4	13
BERT-SQG&CliniBERT	13.5	3.5	18.5

Table E.2: The running time (hour) of QPP-augmented QG, QA and overall QVE based on our selected QG & QA combinations

Parameter	Search Trials	Best
Dropout	[0.3,0.45,0.6,0.75,0.9]	0.75
LSTM Layers	[1,2,3,4]	3
QP Length	[1,2,3]	2

Table E.3: Hyperparameter searches for Question Phrase Prediction (QPP) Module

E.2.1 Prepossessing

Dataset Prepossessing

emrQA[33] We prune the emrQA set by removing QA pairs whose question is an indicator (e.g., "meds") so that all remaining QA pairs contain valid questions. We also leverage SciSpacy[34], a package containing spaCy models for processing clinical text, to do tokenization in order for the trained QG models to have a better understanding over clinical notes.

MIMIC-III[35] Similarly, we also leverage SciSpacy to do tokenization.

[33]https://github.com/panushri25/emrQA

[34]https://allenai.github.io/scispacy/

[35]https://mimic.physionet.org/gettingstarted/access/

Parameter	Search Trials	Best
length (η)	[1,3,5,10]	3
distance (γ)	[1,3,5,10]	3

Table E.4: Hyperparameter searches for *“merge-and-drop"* method

Question Phrases Identification

In order to utilize our Question Phrase Prediction (QPP) module and make our QPP module generic enough without loss of generality, we identify valid n-gram Question Phrases in an automatic way.

To prepare an exhaustive list of valid n-gram Question Phrases, we first collect all of the first *n* words appearing in Ground Truth Questions in emrQA, forming three (i.e., *n*=1, 2, 3) raw Question Phrases set.

We observe that all uni-grams are valid question phrases (e.g., “How”, “When”, “What”), so we don’t do any pruning and keep the uni-gram question phrases set as it is.

As for n-gram (n $\geq$ 2) Question Phrases set, we conduct fine-grained filtering. We only consider n-grams with occurrence frequency greater than the threshold ζ as valid n-gram Question Phrases. In our experiment, we set ζ as 0.02%. Less frequent n-gram words (i.e., frequency $< 0.02\%$) will degrade to unigram Question Phrases in accordance with corresponding question types (e.g., “Has lasix” $\rightarrow$ “Has”*) so as to maintain lossless. In the end, n-gram (n $\geq$ 2) Question Phrases sets, without any information loss, are consisting of both n-gram Question Phrases and degraded unigram Question Phrases.

E.2.2 Models Implementation

Base QA & QG Models We re-implement the three base QG models using Pytorch and have ensured that they achieve comparable performance as originally reported. The best QG

Table E.5: Distributions of the generated questions of different models and the ground truth in the emrQA dataset. QPP: Question Phrase Prediction; KL: Kullback–Leibler divergence. All numbers are percentages.

Models	**What**	**When**	**Has**	**Was**	**Why**	**How**	**Is**	**Did**	**Can**	**Any**	**Does**	**KL (Gen||GT)**
NQG	0.00	0.00	3.95	0.00	0.00	0.00	0.00	0.00	0.00	0.00	96.05	84.2
+BeamSearch	29.94	0.00	5.54	0.00	0.00	0.00	1.38	0.00	0.00	0.00	63.14	45.2
+QPP (Ours)	8.39	0.01	25.37	4.44	0.91	0.87	9.09	0.27	4.03	12.72	33.89	**11.0**
NQG++	0.09	0.00	3.53	0.00	0.00	0.10	0.00	0.00	0.00	0.00	96.28	84.3
+BeamSearch	44.04	0.00	0.09	0.00	0.00	0.22	2.01	0.00	0.00	0.00	53.64	66.4
+QPP (Ours)	8.09	0.01	25.54	4.42	0.75	0.81	9.16	0.23	4.05	12.80	34.13	**11.2**
BERT-SQG	0.72	0.00	6.32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	92.96	74.5
+BeamSearch	29.31	0.00	0.03	0.00	0.00	0.00	25.05	0.00	0.00	0.00	45.62	47.4
+QPP (Ours)	8.01	0.01	25.56	4.44	0.82	0.80	9.17	0.25	4.05	12.74	34.16	**11.2**
GT	13.80	0.01	26.73	1.94	0.63	1.16	13.24	0.14	1.31	4.11	36.93	-

model is selected using the per-token accuracy of both the QPP module (if applicable) and QG on dev set, and dev results are listed in Table E.1.

For QA models, we used their open-sourced implementation.[36] The best QA model is selected using EM and F1 on dev set, and dev results are also included in Table 3 (in the main content). Hyperparameters of QG models are set to be the same as in the original paper and hyperparameters of QA models are set according to the guidance of [278].

Question Phrase Prediction (QPP) Module. Word embeddings are initialized by Glove 300d vectors[37]. We adopt a feature-rich Encoder[38] to effectively encode Clinical lexical information. We set the LSTM hidden unit size to 600 and set the number of layers of LSTMs to 3 in both encoder and decoder. Optimization is performed using stochastic gradient descent (SGD) for 20 epochs, with an initial learning rate of 1.0. After each epoch, we evaluate the per-label accuracy on the dev set. If the accuracy does not improve, we

[36]DocReader: https://github.com/facebookresearch/DrQA. ClinicalBERT: https://github.com/EmilyAlsentzer/clinicalBERT.

[37]http://nlp.stanford.edu/data/glove.840B.300d.zip

[38]Lexical features are extracted by [165]

halve the learning rate. The mini-batch size is set at 128. Dropout with probability 0.75 is applied between vertical LSTM layers. The gradient is clipped when its norm exceeds 5. Besides, we set the length of a question phrase l to 2, which gives the best performance on validation. The total number of parameters is around 17M under our best-performing setting.

Answer Evidence Extractor. We fine-tune a ClinicalBERT model in Named Entity Recognition (NER) fashion using `BIO` tagging scheme[39]. When conducting fine-tuning on our QG train set, we set max length, batch size, number of epochs, and random seed to be 510, 16, 20 and 6, respectively. We adopt the official NER evaluation script[40] to do the evaluation on QG dev set, and obtained 80.17 F1 score. We then deployed this system to extract raw answer evidence spans. After raw extraction, we utilized our own heuristic rules (i.e., *"merge-and-drop"*) to further polish the raw spans as described in Appendix E.1.

Multi-Label Classification (MLC) Comparison. We implement Binary Relevance (BR) and Classifier Chain (CC) by means of Scikit-Multilearn [181], an open-source library for the MLC task.

Computational Resources. All experiments are conducted using one single GeForce GTX 2080 Ti 12 GB GPU (with significant CPU resources). We train the QPP module and QG model together though they can be trained separately. The overall running time of our **`QVE`** system depends on the particular QG and QA models adopted. For our selected QG and QA models, the approximated overall running time of **`QVE`** is listed in Table E.2. However, training a QPP module separately is fast, which only takes less than 1 hour with the current

[39]https://github.com/huggingface/transformers/tree/master/ examples/token-classification

[40]http://deeplearning.net/tutorial/code/conlleval.pl

setting. Meanwhile, the running time of our Answer Evidence Extractor roughly takes 1.5 hours on average.

E.2.3 Hyperparameter Search

In order to have a best-performing Question Phrase Prediction (QPP) module, we manually tuned the hyperparameters listed in Table E.3. The hyperparameters are tuned on QG dev set using Relevance and Diversity Metrics listed in Section 4.3 (in the main content).

In order to have a best-performing post-processing method (i.e., *"merge-and-drop"*) in Answer Evidence Extraction module, we manually tuned the hyperparameters listed in Table E.4. The hyperparameters are tuned on the QA dev set using Exact Match (EM) and F1.

E.3 Distributions of Generated Questions of Different QG Models

The detailed distributions of the generated questions of different models and the ground truth in emrQA dataset are listed in table E.5.

Appendix F: Synthetic Question Value Estimation for Domain Adaptation of Question Answering

F.1 Details of Datasets

Specifically, following Shakeri et al. [212], we use **SQuAD 1.1** [193], a large reading comprehension dataset that consists of 100k questions on more than 500 articles from Wikipedia, as the *source-domain* dataset. For the *target-domain* datasets, we consider the following 4 datasets since they are commonly used and have sufficient contexts to train the models.

NewsQA [235] consists of questions and answers based on a set of over 10k news articles from CNN News.

Natural Questions (NQ) [123] contains questions extracted from Google user search queries and passages from Wikipedia.

HotpotQA [273] is a multi-hop question answering dataset based on Wikipedia passages.

TriviaQA [107] includes QA pairs authored by trivia enthusiasts, as well as evidence documents independently gathered from Web search results and Wikipedia articles.

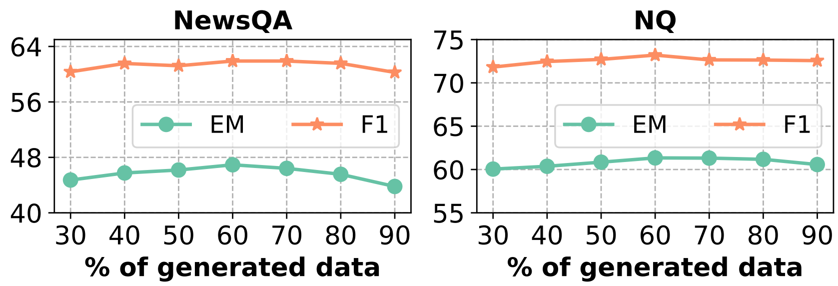


Figure F.1: Impact of synthetic dataset size.

F.2 Impact of Synthetic Dataset Size

In Figure F.1, we show how the synthetic dataset size (i.e., the number of selected QA pairs) impacts the QA performance, based on our QVE (RL) filtering. As we expect, at the beginning, the target QA performance improves when more synthetic data is added to the training set. However, the performance reaches the peak at 60-70% and then goes down. This is reasonable since adding less valuable QA pairs from the noisy synthetic data will hurt the QA model training. We suggest 60%-70% (50K-70K QA pairs) for setting the synthetic data size in practice.

Bibliography

[1] Martín Abadi, Andy Chu, Ian J. 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, Vienna, Austria, October 24-28, 2016*, pages 308–318. ACM, 2016. doi: 10.1145/2976749.2978318. URL `https://doi.org/10.1145/2976749.2978318`.

[2] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. Synthetic QA corpora generation with roundtrip consistency. In *ACL'19*, pages 6168–6173. Association for Computational Linguistics, 2019. URL `https://doi.org/10.18653/v1/p19-1620`.

[3] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. Synthetic qa corpora generation with roundtrip consistency. In *ACL'19*, pages 6168–6173, 2019.

[4] Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. Publicly available clinical BERT embeddings. In *2nd Clinical Natural Language Processing Workshop*, pages 72–78, 2019. Also available at arXiv:1904.03323.

[5] Mário S. Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. Local differential privacy on metric spaces: Optimizing the trade-off with utility. In *CSF*, pages 262–267, 2018.

[6] Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. Feverous: Fact extraction and verification over unstructured and structured information. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[7] Miguel E. Andrés, Nicolás Emilio Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In *CCS*, pages 901–914, 2013.

[8] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[9] Art. 29 WP. Opinion 05/2014 on "Anonymisation Techniques", 2014. URL `https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf`.

[10] Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. MultiFC: A real-world

multi-domain dataset for evidence-based fact checking of claims. 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 4685–4697, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1475. URL https://aclanthology.org/D19-1475.

[11] Sean Augenstein, H Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, et al. Generative models for effective ml on private, decentralized datasets. *arXiv preprint arXiv:1911.06679*, 2019.

[12] 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.

[13] Petr Baudiš and Jan Šedivỳ. Modeling of the question answering task in the yodaqa system. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 6th International Conference of the CLEF Association, CLEF'15, Toulouse, France, September 8-11, 2015, Proceedings 6*, pages 222–228. Springer, 2015.

[14] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544, 2013.

[15] Vincent Bindschaedler, Reza Shokri, and Carl A. Gunter. Plausible deniability for privacy-preserving data synthesis. *PVLDB*, 10(5):481–492, 2017.

[16] Haohan Bo, Steven H. H. Ding, Benjamin C. M. Fung, and Farkhund Iqbal. ER-AE: differentially private text generation for authorship anonymization. In *NAACL-HLT*, pages 3997–4007, 2021.

[17] Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*, 2022.

[18] Rishi Bommasani, Steven Wu, and Xanda Schofield. Towards private synthetic text generation. In *NeurIPS 2019 Machine Learning with Guarantees Workshop*, 2019.

[19] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[20] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[21] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

[22] Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. Learning multi-label scene classification. *Pattern Recognition*, page 1757–1771, 2004.

[23] Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, 2015.

[24] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[25] Zhiqi Bu, Jialin Mao, and Shiyun Xu. Scalable and efficient training of large convolutional neural networks with differential privacy. *arXiv preprint arXiv:2205.10683*, 2022.

[26] Meng Cao, Yue Dong, and Jackie Chi Kit Cheung. Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3354, 2022.

[27] Yu Cao, Meng Fang, Baosheng Yu, and Joey Tianyi Zhou. Unsupervised domain adaptation on reading comprehension. In *AAAI'20*, 2020.

[28] 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 (USENIX Security 19)*, pages 267–284, 2019.

[29] 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*, pages 267–284, 2019.

[30] Nicholas Carlini, Samuel Deng, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, Shuang Song, Abhradeep Thakurta, and Florian Tramèr. An attack on InstaHide: Is private learning possible with instance encoding? arXiv:2011.05315, 2020.

[31] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. arXiv:2012.07805, 2020.

[32] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[33] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security 2021*, pages 2633–2650. USENIX Association, 2021. URL `https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting`.

[34] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. Quantifying memorization across neural language models. *CoRR*, abs/2202.07646, 2022. URL https://arxiv.org/abs/2202.07646.

[35] Ying-Hong Chan and Yao-Chung Fan. A recurrent bert-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, 2019.

[36] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In *PETS*, pages 82–102, 2013.

[37] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *ACL'17*, pages 1870–1879, 2017.

[38] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *CoRR*, abs/1604.06174, 2016. URL http://arxiv.org/abs/1604.06174.

[39] Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryxjnREFwH.

[40] Yanda Chen, Md. Arafat Sultan, and Vittorio Castelli. Improved synthetic training for reading comprehension. *CoRR*, abs/2010.12776, 2020. URL https://arxiv.org/abs/2010.12776.

[41] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

[42] Jaemin Cho, Minjoon Seo, and Hannaneh Hajishirzi. Mixture content selection for diverse sequence generation. In *EMNLP-IJCNLP'19*, pages 3112–3122, 2019.

[43] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[44] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[45] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[46] Yu-An Chung, Hung-Yi Lee, and James Glass. Supervised and unsupervised transfer learning for question answering. In *NAACL-HLT'18*, pages 1585–1594, 2018.

[47] Maximin Coavoux, Shashi Narayan, and Shay B. Cohen. Privacy-preserving neural representations of text. In *EMNLP*, pages 1–10, 2018.

[48] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Marginal release under local differential privacy. In *SIGMOD*, pages 131–146, 2018.

[49] 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.

[50] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT'19*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

[51] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT'19*, pages 4171–4186, 2019.

[52] Bhuwan Dhingra, Danish Danish, and Dheeraj Rajagopal. Simple and effective semi-supervised question answering. In *NAACL'18*, pages 582–587, 2018.

[53] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *NeurIPS'19*, pages 13042–13054, 2019. URL `http://papers.nips.cc/paper/9464-unified-language-model-pre-training-for-natural-language-understanding-an`

[54] Xinya Du and Claire Cardie. Harvesting paragraph-level question-answer pairs from wikipedia. In *ACL'18*, pages 1907–1917, 2018.

[55] Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. In *ACL'17*, pages 1342–1352, 2017. URL `https://www.aclweb.org/anthology/P17-1123.pdf`.

[56] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *FOCS*, pages 429–438, 2013.

[57] Cynthia Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.

[58] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014. doi: 10.1561/0400000042. URL `https://doi.org/10.1561/0400000042`.

[59] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[60] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. URL `https://doi.org/10.1007/11681878_14`.

[61] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[62] Nouha Dziri, Andrea Madotto, Osmar R Zaiane, and Avishek Joey Bose. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. In

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2197–2214, 2021.

[63] Yanai Elazar and Yoav Goldberg. Adversarial removal of demographic attributes from text data. In *EMNLP*, pages 11–21, 2018.

[64] Alex Estes, Nikhita Vedula, Marcus Collins, Matt Cecil, and Oleg Rokhlenko. Fact checking machine generated text with dependency trees. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 458–466, 2022.

[65] Alexander Richard Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. Template-based question generation from retrieved sentences for improved unsupervised question answering. In *ACL'20*, pages 4508–4513, 2020.

[66] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[67] Angela Fan, Mike Lewis, and Yann N. Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pages 889–898, 2018.

[68] Jungwei Fan. Annotating and characterizing clinical sentences with explicit why-qa cues. In *NAACL Clinical NLP Workshop*, pages 101–106, 2019.

[69] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In *NeurIPS*, 2020.

[70] Oluwaseyi Feyisetan, Tom Diethe, and Thomas Drake. Leveraging hierarchical representations for preserving privacy and utility in text. In *ICDM*, pages 210–219, 2019.

[71] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy- and utility-preserving textual analysis via calibrated multivariate perturbations. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 178–186. ACM, 2020. URL `https://doi.org/10.1145/3336191.3371856`.

[72] Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *MRQA@EMNLP'19*, pages 1–13. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5801. URL `https://doi.org/10.18653/v1/D19-5801`.

[73] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

[74] Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12848–12856, 2021.

[75] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. Rarr: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*, 2022.

[76] Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*, 2023.

[77] Zorik Gekhman, Jonathan Herzig, Roee Aharoni, Chen Elkind, and Idan Szpektor. Trueteacher: Learning factual consistency evaluation with large language models. *ArXiv preprint*, abs/2305.11171, 2023. doi: 10.48550/arXiv.2305.11171. URL https://doi.org/10.48550/arXiv.2305.11171.

[78] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV'16*, pages 597–613. Springer, 2016.

[79] Amirata Ghorbani and James Y. Zou. Data shapley: Equitable valuation of data for machine learning. In *ICML'19*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR, 2019. URL http://proceedings.mlr.press/v97/ghorbani19c.html.

[80] Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

[81] David Golub, Po-Sen Huang, Xiaodong He, and Li Deng. Two-stage synthesis networks for transfer learning in machine comprehension. In *EMNLP'17*, pages 835–844. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1087. URL https://doi.org/10.18653/v1/d17-1087.

[82] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

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

[84] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. Numerical composition of differential privacy. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 11631–11642, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/6097d8f3714205740f30debe1166744e-Abstract.html.

[85] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*, 2022.

[86] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. Pointing the unknown words. In *ACL'16*, pages 140–149, 2016.

[87] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020.

[88] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

[89] Ivan Habernal. When differential privacy meets NLP: the devil is in the detail. In *EMNLP*, pages 1522–1528, 2021.

[90] Timothy J Hazen, Shehzaad Dhuliawala, and Daniel Boies. Towards domain adaptation from limited data for question answering using deep neural networks. *arXiv preprint arXiv:1911.02655*, 2019.

[91] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6626–6637, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html`.

[92] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018.

[93] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

[94] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

[95] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[96] Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. TRUE: re-evaluating factual consistency evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 3905–3920. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.naacl-main.287. URL https://doi.org/10.18653/v1/2022.naacl-main.287.

[97] Yangsibo Huang, Zhao Song, Danqi Chen, Kai Li, and Sanjeev Arora. TextHide: Tackling data privacy for language understanding tasks. In *EMNLP (Findings)*, pages 1368–1382, 2020.

[98] Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. Comparison of diverse decoding methods from conditional language models. In *ACL '19*, pages 3752–3762. Association for Computational Linguistics, 2019.

[99] Gautier Izacard and Édouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, 2021.

[100] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv*, 2208, 2022.

[101] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 2022.

[102] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[103] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J. Spanos. Towards efficient data valuation based on the shapley value. In *AISTATS'19*, volume 89 of *Proceedings of Machine Learning Research*, pages 1167–1176. PMLR, 2019. URL `http://proceedings.mlr.press/v89/jia19a.html`.

[104] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and

Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3: 160035, 2016.

[105] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. Pate-gan: Generating synthetic data with differential privacy guarantees. In *International conference on learning representations*, 2018.

[106] James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N Cohen, and Adrian Weller. Synthetic data–what, why and how? *arXiv preprint arXiv:2205.03257*, 2022.

[107] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL'17*, pages 1601–1611. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1147. URL `https://doi.org/10.18653/v1/P17-1147`.

[108] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

[109] Kaggle. 2017 kaggle machine learning & data science survey. `https://www.kaggle.com/code/sudalairajkumar/an-interactive-deep-dive-into-survey-results/notebook`, 2017. Accessed: 2022-11-10.

[110] Ryo Kamoi, Tanya Goyal, Juan Diego Rodriguez, and Greg Durrett. Wice: Real-world entailment for claims in wikipedia. *ArXiv preprint*, abs/2303.01432, 2023.

doi: 10.48550/arXiv.2303.01432. URL `https://doi.org/10.48550/arXiv.2303.01432`.

[111] Junmo Kang, Haritz Puerto San Roman, et al. Let me know what to ask: Interrogative-word-aware question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 163–171, 2019.

[112] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL `https://aclanthology.org/2020.emnlp-main.550`.

[113] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

[114] 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.

[115] Tushar Khot, Ashish Sabharwal, and Peter Clark. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[116] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

[117] Mojtaba Komeili, Kurt Shuster, and Jason Weston. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, 2022.

[118] Neema Kotonya and Francesca Toni. Explainable automated fact-checking for public health claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7740–7754, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.623. URL https://aclanthology.org/2020.emnlp-main.623.

[119] Satyapriya Krishna, Rahul Gupta, and Christophe Dupuy. Adept: Auto-encoder based differentially private text transformation. In *EACL*, pages 2435–2439, 2021.

[120] Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, 2020.

[121] Aditya Kunar, Robert Birke, Zilong Zhao, and Lydia Chen. Dtgan: Differential private training for tabular gans. *arXiv preprint arXiv:2107.02521*, 2021.

[122] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee,

et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[123] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019. URL https://transacl.org/ojs/index.php/tacl/article/view/1455.

[124] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3929–3938, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/0234c510bc6d908b28c70ff313743079-Abstract.html.

[125] Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. Generating diverse and consistent QA pairs from contexts with information-maximizing hierarchical conditional vaes. In *ACL'20*, pages 208–224. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.20. URL https://doi.org/10.18653/v1/2020.acl-main.20.

[126] Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. Hallucinations in neural machine translation. In *ICLR*, 2019.

[127] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, 2019.

[128] Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron C. Wallace. Does BERT pretrained on clinical notes reveal sensitive data? In *NAACL-HLT*, pages 946–959, 2021.

[129] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[130] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL'20*, pages 7871–7880. Association for Computational Linguistics, 2020. URL `https://www.aclweb.org/anthology/2020.acl-main.703/`.

[131] Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, 2019.

[132] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[133] Patrick S. H. Lewis, Ludovic Denoyer, and Sebastian Riedel. Unsupervised question answering by cloze translation. In *ACL'19*, pages 4896–4910. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1484. URL https://doi.org/10.18653/v1/p19-1484.

[134] Minghao Li, Feifan Song, Bowen Yu, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A benchmark for tool-augmented llms. *arXiv preprint arXiv:2304.08244*, 2023.

[135] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[136] Xuechen Li, Daogao Liu, Tatsunori 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*, 2022.

[137] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=bVuP3ltATMz.

[138] Yitong Li, Timothy Baldwin, and Trevor Cohn. Towards robust and privacy-preserving text representations. In *ACL*, pages 25–30, 2018.

[139] Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. Harvesting and refining question-answer pairs for unsupervised qa. In *ACL'20*, pages 6719–6728, 2020.

[140] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[141] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. Asking questions the human way: Scalable question-answer generation from text corpus. In *WWW'20*, pages 2032–2043, 2020.

[142] Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. Reinforced training data selection for domain adaptation. In *ACL*, pages 1957–1968. Association for Computational Linguistics, 2019. URL https://doi.org/10.18653/v1/p19-1189.

[143] Nelson F. Liu, Tianyi Zhang, and Percy Liang. Evaluating verifiability in generative search engines. *arXiv preprint arXiv:2304.09848*, 2023.

[144] 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. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

[145] 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. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

[146] Qian Lou, Bo Feng, Geoffrey Charles Fox, and Lei Jiang. Glyph: Fast and accurately training deep neural networks on encrypted data. In *NeurIPS*, 2020.

[147] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *EMNLP'15*, pages 1412–1421, 2015.

[148] Lingjuan Lyu, Xuanli He, and Yitong Li. Differentially private representation for NLP: Formal guarantee and an empirical study on privacy and fairness. In *EMNLP (Findings)*, pages 2355–2365, 2020.

[149] Lingjuan Lyu, Yitong Li, Xuanli He, and Tong Xiao. Towards differentially private text representations. In *SIGIR*, pages 1813–1816, 2020.

[150] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 2022.

[151] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, 2020.

[152] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. Winning the nist contest: A scalable and general approach to differentially private synthetic data. *arXiv preprint arXiv:2108.04978*, 2021.

[153] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *ICLR (Poster)*, 2018.

[154] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[155] Casey Meehan, Khalil Mrini, and Kamalika Chaudhuri. Sentence-level privacy for document embeddings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3367–3380. Association for Computational Linguistics, 2022. URL `https://doi.org/10.18653/v1/2022.acl-long.238`.

[156] Harsh Mehta, Abhradeep Thakurta, Alexey Kurakin, and Ashok Cutkosky. Large scale transfer learning for differentially private image classification. *arXiv preprint arXiv:2205.02973*, 2022.

[157] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. Augmented language models: a survey, 2023.

[158] Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. A discrete hard em approach for weakly supervised question answering. 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 2851–2864, 2019.

[159] Samarth Mishra, Rameswar Panda, Cheng Perng Phoo, Chun-Fu Richard Chen, Leonid Karlinsky, Kate Saenko, Venkatesh Saligrama, and Rogerio S Feris. Task2sim: Towards effective pre-training and transfer from synthetic data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9194–9204, 2022.

[160] Takao Murakami and Yusuke Kawamoto. Utility-optimized local differential privacy mechanisms for distribution estimation. In *USENIX Security*, pages 1877–1894, 2019.

[161] n2c2. n2c2 nlp research data sets, 2006. URL https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/.

[162] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[163] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *S&P*, pages 111–125, 2008.

[164] Preksha Nema, Akash Kumar Mohankumar, Mitesh M Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. Let's ask again: Refine network for automatic question generation. In *EMNLP-IJCNLP'19*, pages 3305–3314, 2019. URL https://www.aclweb.org/anthology/D19-1326.pdf.

[165] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. Scispacy: Fast and robust models for biomedical natural language processing. In *ACL BioNLP Workshop 2019*, 2019.

[166] Marcel Neunhoeffer, Zhiwei Steven Wu, and Cynthia Dwork. Private post-gan boosting. *arXiv preprint arXiv:2007.11934*, 2020.

[167] Lucien K. L. Ng and Sherman S. M. Chow. GForce: GPU-friendly oblivious and rapid neural network inference. In *USENIX Security*, 2021.

[168] Lucien K. L. Ng, Sherman S. M. Chow, Anna P. Y. Woo, Donald P. H. Wong, and Yongjun Zhao. Goten: GPU-Outsourcing Trusted Execution of Neural Network Training. In *AAAI*, 2021.

[169] Thao Nguyen, Gabriel Ilharco, Mitchell Wortsman, Sewoong Oh, and Ludwig Schmidt. Quality not quantity: On the interaction between dataset design and robustness of clip. *arXiv preprint arXiv:2208.05516*, 2022.

[170] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, 2020.

[171] OpenAI. Chatgpt (mar 14 version) [large language model]. https://chat.openai.com/chat, 2023.

[172] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[173] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[174] Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. emrqa: A large corpus for question answering on electronic medical records. In *EMNLP'18*, pages 2357–2368, 2018.

[175] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language models. In *S&P*, pages 1314–1331, 2020.

[176] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL'02*, pages 311–318. Association for Computational Linguistics, 2002.

[177] Jon Patrick and Min Li. An ontology for clinical questions about the contents of patient notes. *JBI*, 45(2):292–306, 2012.

[178] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*, 2023.

[179] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014. 300-d GloVe embeddings available at `http://nlp.stanford.edu/data/glove.840B.300d.zip`.

[180] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaïd Harchaoui. MAUVE: measuring the gap between neural text and human text using divergence frontiers. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 4816–4828, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/260c2432a0eecc28ce03c10dadc078a4-Abstract.html`.

[181] Tomasz Kajdanowicz Piotr Szymański. A scikit-based python environment for performing multi-label classification. *arXiv preprint arXiv:1702.01460*, 2017.

[182] Adam Poliak, Max Fleming, Cash Costello, Kenton W. Murray, Mahsa Yarmohammadi, Shivani Pandya, Darius Irani, Milind Agarwal, Udit Sharma, Shuo Sun, Nicola Ivanov, Lingxi Shang, Kaushik Srinivasan, Seolhwa Lee, Xu Han, Smisha Agarwal, and João Sedoc. Collecting verified COVID-19 question answer pairs. In *Proceedings of the 1st Workshop on NLP for COVID-19@ EMNLP 2020, Online, December 2020*. Association for Computational Linguistics, 2020.

[183] Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. Training question answering models from synthetic data. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *EMNLP'20*, pages 5811–5826. Association for Computational Linguistics, 2020. URL https://www.aclweb.org/anthology/2020.emnlp-main.468/.

[184] Hongjing Qian, Yutao Zhu, Zhicheng Dou, Haoqi Gu, Xinyu Zhang, Zheng Liu, Ruofei Lai, Zhao Cao, Jian-Yun Nie, and Ji-Rong Wen. Webbrain: Learning to generate factually correct articles for queries by grounding on large web corpus. *arXiv preprint arXiv:2304.04358*, 2023.

[185] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models. *CoRR*, abs/2304.08354, 2023.

[186] Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W. Bruce Croft. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In *WSDM*, pages 699–707. ACM, 2019. URL `https://doi.org/10.1145/3289600.3290978`.

[187] Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. Privacy-adaptive BERT for natural language understanding. arXiv:2104.07504, 2021.

[188] Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. Natural language understanding with privacy-preserving bert. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1488–1497, 2021.

[189] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[190] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[191] 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. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

[192] Preethi Raghavan, Siddharth Patwardhan, Jennifer J Liang, and Murthy V Devarakonda. Annotating electronic medical records for question answering. *arXiv preprint arXiv:1805.06816*, 2018.

[193] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *EMNLP'16*, pages 2383–2392. The Association for Computational Linguistics, 2016. doi: 10.18653/v1/d16-1264. URL https://doi.org/10.18653/v1/d16-1264.

[194] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP'16*, pages 2383–2392, 2016.

[195] Alan Ramponi and Barbara Plank. Neural unsupervised domain adaptation in nlp—a survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, 2020.

[196] Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999.

[197] Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. Measuring attribution in natural language generation models. *arXiv preprint arXiv:2112.12870*, 2021.

[198] Bhanu Pratap Singh Rawat, Wei-Hung Weng, Preethi Raghavan, and Peter Szolovits. Entity-enriched neural models for clinical question answering. *arXiv preprint arXiv:2005.06587*, 2020.

[199] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333, 2011.

[200] Samuel Reese, Gemma Boleda, Montse Cuadros, Lluís Padró, and German Rigau. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. In *LREC*, 2010.

[201] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL `http://arxiv.org/abs/1908.10084`.

[202] Steven J. Rennie, Etienne Marcheret, Neil Mallinar, David Nahamoo, and Vaibhava Goel. Unsupervised adaptation of question answering systems via generative self-training. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *EMNLP'20*, pages 1148–1157. Association for Computational Linguistics, 2020. URL `https://www.aclweb.org/anthology/2020.emnlp-main.87/`.

[203] Alfréd Rényi. On measures of entropy and information. In *Berkeley Symposium on Mathematical Statistics and Probability*, pages 547–561, 1961.

[204] Donald B Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 9(2): 461–468, 1993.

[205] Sebastian Ruder and Barbara Plank. Learning to select data for transfer learning with bayesian optimization. In *EMNLP*, pages 372–382. Association for Computational Linguistics, 2017. URL `https://doi.org/10.18653/v1/d17-1038`.

[206] Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. End-to-end training of neural retrievers for open-domain question answering. In *ACL/IJCNLP 2021*, pages 6648–6662. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.519.

[207] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

[208] Tal Schuster, Adam Fisch, and Regina Barzilay. Get your vitamin c! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, 2021.

[209] Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, 2021.

[210] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

[211] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hananneh Hajishirzi. Bidirectional attention flow for machine comprehension. In *ICLR'17*, 2017.

[212] Siamak Shakeri, Cícero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. End-to-end synthetic data generation for domain adaptation of question answering systems. In *EMNLP'20*, pages 5445–5460. Association for Computational Linguistics, 2020. URL `https://www.aclweb.org/anthology/2020.emnlp-main.439/`.

[213] Sheng Shen, Yaliang Li, Nan Du, Xian Wu, Yusheng Xie, Shen Ge, Tao Yang, Kai Wang, Xingzheng Liang, and Wei Fan. On the generation of medical question-answer pairs. In *AAAI*, pages 8822–8829, 2020.

[214] 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.

[215] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *S&P*, pages 3–18, 2017.

[216] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, 2021.

[217] Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, et al. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.

[218] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

[219] Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *CCS*, pages 377–390, 2020.

[220] Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *KDD*, pages 196–206, 2019.

[221] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *2013 IEEE global conference on signal and information processing*, pages 245–248. IEEE, 2013.

[222] Sarvesh Soni and Kirk Roberts. A paraphrase generation system for ehr question answering. In *18th BioNLP Workshop*, pages 20–29, 2019.

[223] Sarvesh Soni and Kirk Roberts. Paraphrasing to improve the performance of electronic health records question answering. *AMIA Summits*, 2020:626, 2020.

[224] Pranav Subramani, Nicholas Vadivelu, and Gautam Kamath. Enabling fast differentially private SGD via just-in-time compilation and vectorization. *Advances in Neural Information Processing Systems*, 34:26409–26421, 2021.

[225] Md. Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. On the importance of diversity in question generation for QA. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 5651–5656, 2020.

[226] Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. Answer-focused and position-aware neural question generation. In *EMNLP'18*, pages 3930–3939, 2018.

[227] Latanya Sweeney. Only you, your doctor, and many others may know. *Technology Science*, 2015092903(9):29, 2015.

[228] Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*, 2018.

[229] Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Benchmarking differentially private synthetic data generation algorithms. *arXiv preprint arXiv:2112.09238*, 2021.

[230] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[231] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[232] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 809–819, 2018.

[233] James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. The fever2. 0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 1–6, 2019.

[234] Florian Tramèr and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. In *ICLR*, 2019.

[235] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. In *Rep4NLP@ACL'17*, pages 191–200. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-2623. URL `https://doi.org/10.18653/v1/w17-2623`.

[236] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*, 2022.

[237] Luu Anh Tuan, Darsh J Shah, and Regina Barzilay. Capturing greater context for question generation. In *AAAI'20*, 2020.

[238] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR'17*, pages 7167–7176, 2017.

[239] Gerrit J. J. van den Burg and Christopher K. I. Williams. On memorization in probabilistic deep generative models. In *NeurIPS*, 2021. Also available as arXiv:2106.03216.

[240] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[241] Nikhita Vedula and Srinivasan Parthasarathy. Face-keg: Fact checking explained using knowledge graphs. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 526–534, 2021.

[242] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.609. URL `https://aclanthology.org/2020.emnlp-main.609`.

[243] Sameer Wagh, Shruti Tople, Fabrice Benhamouda, Eyal Kushilevitz, Prateek Mittal, and Tal Rabin. FALCON: Honest-majority maliciously secure framework for private deep learning. *PoPETs*, 2021(1):187–207, 2021.

[244] 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 *ICLR (Poster)*, 2019.

[245] Chaojun Wang and Rico Sennrich. On exposure bias, hallucination and domain shift in neural machine translation. In *2020 Annual Conference of the Association for Computational Linguistics*, pages 3544–3552. Association for Computational Linguistics (ACL), 2020.

[246] Huazheng Wang, Zhe Gan, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, and Hongning Wang. Adversarial domain adaptation for machine reading comprehension. In *EMNLP-IJCNLP'19*, pages 2510–2520. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1254. URL https://doi.org/10.18653/v1/D19-1254.

[247] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security*, pages 729–745, 2017.

[248] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private frequent itemset mining. In *S&P*, pages 127–143, 2018.

[249] Yanshan Wang, Naveed Afzal, Sunyang Fu, Liwei Wang, Feichen Shen, Majid Rastegar-Mojarad, and Hongfang Liu. MedSTS: a resource for clinical semantic textual similarity. *Lang. Resour. Evaluation*, 54(1):57–72, 2020.

[250] Benjamin Weggenmann and Florian Kerschbaum. Syntf: Synthetic and differentially private term frequency vectors for privacy-preserving text mining. In *SIGIR*, pages 305–314, 2018.

[251] Benjamin Weggenmann, Valentin Rublack, Michael Andrejczuk, Justus Mattern, and Florian Kerschbaum. DP-VAE: human-readable text anonymization for online reviews with differentially private variational autoencoders. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 721–731. ACM, 2022. URL https://doi.org/10.1145/3485447.3512232.

[252] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *ICLR*, 2022.

[253] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[254] Andrew Wen, Mohamed Y Elwazir, Sungrim Moon, and Jungwei Fan. Adapting and evaluating a deep learning language model for clinical why-question answering. *JAMIA Open*, 3(1):16–20, 2020.

[255] Georg Wiese, Dirk Weissenborn, and Mariana Neves. Neural domain adaptation for biomedical question answering. In *CoNLL'17*, pages 281–289, 2017.

[256] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, 2018.

[257] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL `https://doi.org/10.1007/BF00992696`.

[258] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint*

arXiv:1910.03771, 2019.

[259] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[260] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *EMNLP (System Demonstrations)*, pages 38–45, 2020. Also available at arXiv:1910.03771, and https://github.com/huggingface.

[261] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *EMNLP'20*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

[262] Harry W. H. Wong, Jack P. K. Ma, Donald P. H. Wong, Lucien K. L. Ng, and Sherman S. M. Chow. Learning model with error - exposing the hidden model of BAYHENN. In *IJCAI*, pages 3529–3535, 2020.

[263] Yuhuai Wu, Felix Li, and Percy Liang. Insights into pre-training via simpler synthetic tasks. *arXiv preprint arXiv:2206.10139*, 2022.

[264] Lukas Wutschitz, Huseyin A. Inan, and Andre Manoel. dp-transformers: Training transformer models with differential privacy. `https://www.microsoft.com/en-us/research/project/dp-transformers`, August 2022.

[265] Yijun Xiao and William Yang Wang. On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744, 2021.

[266] Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Unraveling the behavior of large language models in knowledge conflicts. *arXiv preprint arXiv:2305.13300*, 2023.

[267] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

[268] Qizhe Xie, Zihang Dai, Yulun Du, Eduard H. Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. In *NeurIPS*, pages 585–596, 2017.

[269] Nan Xu, Oluwaseyi Feyisetan, Abhinav Aggarwal, Zekun Xu, and Nathanael Teissier. Density-aware differentially private textual perturbations using truncated gumbel noise. In *FLAIRS*, 2021.

[270] Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. On a utilitarian approach to privacy preserving text generation. In *PrivateNLP@NAACL*, 2021.

[271] Zhilin Yang, Junjie Hu, Ruslan Salakhutdinov, and William Cohen. Semi-supervised qa with generative domain-adaptive nets. In *ACL'17*, pages 1040–1050, 2017.

[272] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP'18*, pages 2369–2380, 2018.

[273] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP'18*, pages 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1259. URL `https://doi.org/10.18653/v1/d18-1259`.

[274] Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

[275] Jinsung Yoon, Sercan Arik, and Tomas Pfister. Data valuation using reinforcement learning. In *ICML'20*, pages 10842–10851. PMLR, 2020. URL `http://proceedings.mlr.press/v119/yoon20a/yoon20a.pdf`.

[276] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

[277] 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*, 2022.

[278] Xiang Yue, Bernal Jimenez Gutierrez, and Huan Sun. Clinical reading comprehension: A thorough analysis of the emrqa dataset. In *ACL'20*, 2020.

[279] Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. Differential privacy for text analytics via natural text sanitization. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3853–3866. Association for Computational Linguistics, 2021. URL https://doi.org/10.18653/v1/2021.findings-acl.337.

[280] Zhenrui Yue, Bernhard Kratzwald, and Stefan Feuerriegel. Contrastive domain adaptation for question answering using limited text corpora. In *EMNLP'21*, 2021.

[281] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual memorization in neural language models. arXiv:2112.12938, 2021.

[282] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

[283] Shiyue Zhang and Mohit Bansal. Addressing semantic drift in question generation for semi-supervised question answering. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *EMNLP-IJCNLP'19*, pages 2495–2509. Association for Computational Linguistics, 2019. URL https://doi.org/10.18653/v1/D19-1253.

[284] Shiyue Zhang and Mohit Bansal. Addressing semantic drift in question generation for semi-supervised question answering. In *EMNLP-IJCNLP'19*, pages 2495–2509, 2019.

[285] Xinliang Frederick Zhang, Heming Sun, Xiang Yue, Simon M. Lin, and Huan Sun. COUGH: A challenge dataset and models for COVID-19 FAQ retrieval. In *EMNLP 2021*, pages 3759–3769. Association for Computational Linguistics, 2021.

[286] Zhen Zhang, Mianzhi Wang, Yan Huang, and Arye Nehorai. Aligning infinite-dimensional covariance matrices in reproducing kernel hilbert spaces for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3437–3445, 2018.

[287] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *EMNLP'18*, pages 3901–3910. Association for Computational Linguistics, 2018. URL https://doi.org/10.18653/v1/d18-1424.

[288] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv preprint*, abs/2306.05685, 2023. doi: 10.48550/arXiv.2306.05685. URL https://doi.org/10.48550/arXiv.2306.05685.

[289] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. Neural question generation from text: A preliminary study. In *National CCF Conference*

on Natural Language Processing and Chinese Computing, pages 662–671. Springer, 2017. URL https://doi.org/10.1007/978-3-319-73618-1_56.

[290] Wenxuan Zhou, Sheng Zhang, Hoifung Poon, and Muhao Chen. Context-faithful prompting for large language models, 2023.