ABSTRACT

TRUSTWORTHY MACHINE LEARNING THROUGH THE LENS OF PRIVACY AND SECURITY

by
Thi Kim Phung Lai

Nowadays, machine learning (ML) becomes ubiquitous and it is transforming society. However, there are still many incidents caused by ML-based systems when ML is deployed in real-world scenarios. Therefore, to allow wide adoption of ML in the real world, especially in critical applications such as healthcare, finance, etc., it is crucial to develop ML models that are not only accurate but also trustworthy (e.g., explainable, privacy-preserving, secure, and robust). Achieving trustworthy ML with different machine learning paradigms (e.g., deep learning, centralized learning, federated learning, etc.), and application domains (e.g., computer vision, natural language, human study, malware systems, etc.) is challenging, given the complicated trade-off among utility, scalability, privacy, explainability, and security.

To bring trustworthy ML to real-world adoption with the trust of communities, this study makes a contribution of introducing a series of novel privacy-preserving mechanisms in which the trade-off between model utility and trustworthiness is optimized in different application domains, including natural language models, federated learning with human and mobile sensing applications, image classification, and explainable AI. The proposed mechanisms reach deployment levels of commercialized systems in real-world trials while providing trustworthiness with marginal utility drops and rigorous theoretical guarantees.The developed solutions enable safe, efficient, and practical analyses of rich and diverse user-generated data in many application domains.

TRUSTWORTHY MACHINE LEARNING
THROUGH THE LENS OF PRIVACY AND SECURITY

by
Thi Kim Phung Lai

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Information Systems

Department of Informatics

May 2023

APPROVAL PAGE

TRUSTWORTHY MACHINE LEARNING THROUGH THE LENS OF PRIVACY AND SECURITY

Thi Kim Phung Lai

Hai Phan, Dissertation Advisor — Date
Assistant Professor of Data Science, NJIT

Yi-fang Brook Wu, Committee Member — Date
Associate Professor of Informatics, NJIT

Yi Chen, Committee Member — Date
Professor of Business Data Science, NJIT

Tong Sun, Committee Member — Date
Director of Document Intelligence Labs, Adobe Inc., San Jose, California

Xiong Li, Committee Member — Date
Professor of Computer Science and Biomedical Informatics, Emory University, Atlanta, GA

BIOGRAPHICAL SKETCH

Author:	Thi Kim Phung Lai
Degree:	Doctor of Philosophy
Date:	May 2023

Undergraduate and Graduate Education:

- Ph.D. in Information Systems
  New Jersey Institute of Technology, Newark, NJ, 2023
- Master of Science in Computer Science
  Oregon State University, Corvallis, OR, 2018
- Bachelor of Science in Electronics and Telecommunications
  DaNang University of Science and Technology, Da Nang, Viet Nam, 2013

Major: Information Systems

Presentations and Publications:

Truc Nguyen*, **Phung Lai***, Hai Phan, My Thai, "XRAND: Differentially Private Defense against Explanation-Guided Attacks," *Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence*, 2023 (**AAAI 2023 Distinguished Paper (12 selected/8,777) - Oral presentation**) (* Equal contribution)

Truc Nguyen, **Phung Lai**, Khang Tran, Hai Phan, and My Thai, "Active Membership Inference Attack under Local Differential Privacy in Federated Learning", *International Conference on Artificial Intelligence and Statistics (**AISTATS**)*, 2023

Phung Lai, Tong Sun, Rajiv Jain, Franck Dernoncourt, Jiuxiang Gu, Nikolaos Barmpalios, Han Hu, and NhatHai Phan, "User-Entity Differential Privacy in Natural Language Modeling," *IEEE International Conference on Big Data (**IEEE BigData**)*, pp. 1465 – 1474, 2022 **(Oral presentation)**

Xiaopeng Jiang, Han Hu, Thinh On, **Phung Lai**, Vijaya Mayyuri, An Chen, Devu Shila, Adriaan Larmuseau, Ruoming Jin, Cristian Borcea, Hai Phan, "FLSys: Toward an Open Ecosystem for Federated Learning Mobile Apps", *IEEE Transactions on Mobile Computing (**IEEE TMC**)*, 2022

Khang Tran, **Phung Lai**, Hai Phan, Issa Khalil, Yao Ma, Abdallah Khreishah, My Thai, Xintao Wu, "DPGNN: Differential Privacy Preservation in Graph Neural Networks", *IEEE International Conference on Big Data (**IEEE BigData**)*, pp. 1582 – 1587, 2022 **(Oral presentation)**

Phung Lai, Hai Phan, Han Hu, Ruoming Jin, My Thai, An Chen, "Lifelong DP: Consistently Bounded Differential Privacy in Lifelong Learning", *Conference on Lifelong Learning Agents (**CoLLAs**) in Proceedings of Machine Learning Research (**JMLR**)*, pp. 778 – 797, 2022

Pradnya Desai*, **Phung Lai***, NhatHai Phan, and My Thai, "Continual Learning with Differential Privacy," *International Conference on Neural Information Processing (**ICONIP**)*, 2021 pp. 334 – 343, 2021 **(Oral presentation)** (* Equal contribution).

Pelin Ayranci*, **Phung Lai***, NhatHai Phan, David Newman, Alexander Kolinowski, and Deijing Dou, "OnML: An Ontology-based Approach for Interpretable Machine Learning," *Journal of Combinatorial Optimization (**JOCO - Springer**)*, vol. 44(1), pp. 770 – 793, 2022 (* Equal contribution)

Phung Lai, NhatHai Phan, David Newman, Han Hu, Anuja, and Dejing Dou, "Ontology-based interpretable machine learning with learnable anchors," *International Joint Conference on Neural Networks (**IJCNN**)*, 2020

Trung Vu*, **Phung Lai***, Raviv Raich, Anh Pham, Xiaoli Z Fern, and UK Arvind Rao, "A Novel Attribute-based Symmetric Multiple Instance Learning for Histopathological Image Analysis," *IEEE Transactions on Medical Imaging (**IEEE T-MI**)*, 39(10), 3125–3136, 2020 (* Equal contribution)

Phung Lai, NhatHai Phan, David Newman, Han Hu, Anuja, and Dejing Dou, "Ontology-based interpretable machine learning with learnable anchors," *Conference on Neural Information Processing Systems (NeurIPS) workshop – Knowledge Representation and Reasoning Meets Machine Learning (**KR2ML-NeurIPS workshop**)*, 2019.

Phung Lai, Raviv Raich, and Molly Megraw, "ConvMD: Convolutive matrix decomposition for classification of matrix data," *IEEE Statistical Signal Processing (**SSP**) Workshop*, 368–372, 2018.

Tam Nguyen, Raviv Raich, and **Phung Lai**, "Jeffreys prior regularization for logistic regression," *IEEE Statistical Signal Processing (**SSP**) Workshop*, 1–5, 2016.

Phung Lai and Tuan Pham, "Gaussian mixture model based object segmentation for fall detection," *International Workshop on Industrial IT Convergence*, 2014.

Phung Lai, Khue Tra, and Tuan Pham, "An ultra-low power consumption wireless ECG monitoring system," *International Symposium on Research and Development of E-health Systems*, 117–122, 2014.

Khue Tra, **Phung Lai**, and Tuan Pham, “Combination of analog and digital solutions for wireless ECG monitor,” *International Conference on BioSciences and BioElectronics*, 63–67, 2014.

Y Ngo, **Phung Lai**, and Tuan Pham, “A single-camera fall detection using neural network,” *UK Vietnam Advance Surveillance Systems workshop*, 270–280, 2012.

Patents:

Phung Lai, Tong Sun, Rajiv Jain, Franck Dernoncourt, Jiuxiang Gu, and Nikolaos Barmpalios, “Privacy-Aware Language Models Training,” *Non-provisional US Patent*, 2023.

Phung Lai, Tong Sun, Rajiv Jain, Franck Dernoncourt, Jiuxiang Gu, and Nikolaos Barmpalios, “Privacy-Aware Language Models Training,” *Provisional Adobe Patent*, 2022.

Phung Lai, Tong Sun, Rajiv Jain, Franck Dernoncourt, Jiuxiang Gu & Nikolaos Barmpalios, “Preserving User-Entity Differential Privacy in Natural Language Modeling,” *Non-provisional US Patent*, 2021 (To be published in 2023).

Phung Lai, Tong Sun, Rajiv Jain, Franck Dernoncourt, Jiuxiang Gu & Nikolaos Barmpalios, “User-Entity Differential Privacy in Natural Language Modeling,” *Provisional Adobe Patent*, 2021.

Personal Biography:

Thi Kim Phung Lai’s research interests focus on trustworthy machine learning with the core of privacy and security. There are various applications, such as human sensing, mobile computing, healthcare, social goods, etc. Lai has authored 13 publications and some forthcoming work in the field. Her work has been published at leading venues, including AAAI, AISTATS, IEEE BigData, IEEE transactions, etc. Lai is a recipient of the AAAI 2023 Distinguished Paper Award. In addition, Lai is a holder of several patents in privacy preservation in NLP. The NSF and industrial partners, including Adobe, Qualcomm, and Wells Fargo, have funded her work. After graduation, Lai will join the College of Emergency Preparedness, Homeland Security & Cybersecurity, University at Albany, SUNY as a tenure-track Assistant Professor. That position also belongs to the Albany Artificial Intelligence Supercomputing Initiative (Albany AI).

ACKNOWLEDGMENT

I am greatly indebted to Dr. Hai Phan for his exceptional guidance, patience, kindness, and tremendous support. He introduced me to the world of trustworthiness machine learning, particularly differential privacy, and consistently nurtured my passion for this field, which forms the basis of my current and future work. His dedication, passions, interests, and achievements to the professional life have deeply impressed and motivated me to pursue my academia aspirations. I was particularly struck by his remark, “Students’ success is my success,” which underscores his commitment to supporting and inspiring students. Always trying his best to support and encourage students made a big impression on me. He has always been there for me, even in the darkest of times, to give me wise, thorough, and timely advice. I am forever grateful for everything he has done for me. I am determined to uphold this admirable philosophy in my own academic career and when mentoring my future students. Once again, I extend my sincere appreciation to Dr. Phan for being an outstanding and supportive advisor.

I express my gratitude to the other members of my dissertation committee, namely Dr. Brook Wu, Dr. Li Xiong, and Dr. Li Chen, for their participation in my dissertation defense. I would also like to acknowledge the contributions of my former committee members, Dr. David Shaohua Wang, who served on my dissertation proposal committee, and Dr. Xinyue Ye, who served on my qualifying exam committee. Their valuable time, helpful guidance, and thoughtful feedback greatly aided in enhancing both my research study and my skills.

During my internship at Adobe, I had the honor of working with Dr. Tong Sun and Adobe’s Document Intelligence Labs (DIL) on developing privacy-preserving mechanisms for language models. Dr. Sun acted as my supervisor, and we continue to collaborate on privacy-preserving research. As my first industrial experience, I initially struggled to adapt, but thanks to Dr. Sun’s patience, friendliness, and comprehensive guidance, I received invaluable support and feedback when needed. She is brilliant and hard-working, and I have learned a lot from working with her. I would also like to express my gratitude to other members of DIL, including Jiuxiang Gu, Franck Dernoncourt, Rajiv Jain, and Nikolaos Barmpalios, for their valuable guidance and feedback throughout my work.

I appreciate the research funding from the National Science Foundation (NSF), Adobe Systems, and Qualcomm Inc. for during the Ph.D. program (grant numbers: NSF IIS-2041096, NSF CNS-1747798, NSF CNS-1935928, NSF CNS-1850094, Adobe Unrestricted Gift, and Qualcomm Inc. funding). I am especially grateful for my collaborators, including Dr. My Thai, Dr. Li Xiong, Dr. Issa Khalil, Dr. Xintao Wu, and Dr. Ruoming Jin, for their support and encouragement to keep pushing the research boundaries.

In addition, I would like to express my appreciation to my labmates, Han Hu, Pelin Ayranci, Khang Tran, and Khang Dang, for their support and collaboration during my graduate study. Their encouragement, enthusiasm, and constructive feedback have been instrumental in my success and growth as a researcher. I am also grateful to my friends, Truc Nguyen, Hieu Pham, Ngoc Nguyen, Dung Pham, Giang Ly, Danh Nguyen, Thang Bui, Anh Tong, Tam Nguyen, Thang Hoang, Trang Nguyen, Ni Trieu, Khoi Nguyen, and Hai Dao, for adding joy and making my life in the US more pleasant. I will always cherish the memories we have created together.

Last but not least, I'd like to have a special thank to my family, who have been my pillars of strength throughout my journey. My mom Thao Hoang, my dad Muoi Lai, my older sisters Anh Lai and Loan Lai have always supported me with their selfless sacrifices, unwavering encouragement, and unconditional love. Without them, I would not have had the strength and motivation to pursue my dreams. I would also like to express my gratitude to Tran's family, who have become my second family in the US. Their tremendous support, prayers, and well-wishes have been invaluable, and I feel blessed to have them in my life.

Once again, I want to reiterate my heartfelt appreciation for all of you who have been the pillars of my success with your love, understanding, encouragement, and sacrifices. I promise to continue making you proud of my future endeavors.

TABLE OF CONTENTS

Chapter Page

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Technical Challenges 2

1.3 Key Contributions 6

1.4 Organization 9

2 USER-ENTITY DIFFERENTIAL PRIVACY IN
LEARNING NATURAL LANGUAGE MODELS 10

2.1 Preamble 10

2.2 Background 10

2.2.1 Next Word Prediction 10

2.2.2 Sensitive Entities and Sentences 11

2.2.3 Privacy Threat Models 11

2.2.4 Different Levels of DP 12

2.2.5 Sensitive Entities and User Membership 14

2.2.6 UeDP Definition 15

2.3 Preserving UeDP in NLMs 16

2.3.1 Misidentifying Sensitive Entities 16

2.3.2 Preserving UeDP 17

2.4 Experimental Results 20

2.5 Discussion 28

3 SCALABLERR: DIMENSION-SCALABLE LOCAL DIFFERENTIAL PRIVACY
FOR FEDERATED LEARNING 33

3.1 Preamble 33

3.2 Background 33

3.3 Dimension-Scalability 36

TABLE OF CONTENTS
(Continued)

Chapter Page

3.4 Scalable Randomized Response 41

3.4.1 Bit-aware Randomization 45

3.4.2 Privacy Loss Bound 47

3.4.3 Tighter Privacy Loss Bound 50

3.4.4 Dimension-Scalability 52

3.4.5 Privacy, Utility, Dimensionality Trade-off 54

3.5 Experimental Results 57

3.6 Discussion 66

4 XRAND: DIFFERENTIALLY PRIVATE DEFENSE AGAINST EXPLANATION-GUIDED ATTACKS 69

4.1 Preamble 69

4.2 Background 69

4.3 XAI-guided Attack Against MLaaS 70

4.3.1 Exposing MLaaS via XAI 70

4.3.2 XAI-guided Backdoor Attack against MLaaS 71

4.4 XRAND – Local DP Defense 73

4.4.1 LDP-preserving Explanations 75

4.4.2 Privacy Guarantees of XRAND 76

4.5 Certified Robustness 78

4.5.1 Training-time Certified Robustness 78

4.5.2 Inference-time Certified Robustness 79

4.6 Experiments 81

4.7 Discussion 85

5 LIFELONG DP: CONSISTENTLY BOUNDED DIFFERENTIAL PRIVACY IN LIFELONG MACHINE LEARNING 86

5.1 Preamble 86

TABLE OF CONTENTS
(Continued)

Chapter Page

5.2 Background 86

5.3 Privacy Risk and Lifelong DP 88

5.4 Preserving Lifelong DP 92

5.5 Scalable and Heterogeneous Training 98

5.6 Experiments 99

5.7 Discussion 106

6 ADDITIONAL WORK: ONTOLOGY-BASED INTERPRETABLE MACHINE LEARNING FOR TEXTUAL DATA 108

6.1 Preamble 108

6.2 Background and Problem Definition 108

6.2.1 Interpretable Machine Learning 109

6.2.2 Ontology-based Approaches 112

6.2.3 Information Extraction 112

6.3 Ontology-based Interpretable Machine Learning for Textual Data 113

6.3.1 Ontology-based Tuples 113

6.3.2 Ontology-based Sampling Technique 115

6.3.3 Learnable Anchor Text 116

6.3.4 Generating Semantic Explanations 118

6.4 Experiment 120

6.4.1 Datasets and Domain Ontologies 120

6.4.2 Baseline Approaches 122

6.4.3 Experimental Settings 123

6.4.4 Experimental Results and Analysis 127

6.5 Discussion 132

7 CONCLUSION AND FUTURE WORK 133

7.1 Conclusion 133

TABLE OF CONTENTS
(Continued)

Chapter Page

7.2 Future Work 134

APPENDIX A 138

A.1 Sensitive Entity Recognition and Tool-kits 138

A.2 UeDP without Considering Extended Sensitive Entities 139

A.3 Datasets and Data Processing 141

A.4 Revisiting Word-level LDP Analysis in [120] 142

A.5 Supplemental Experimental Results for UeDP 146

A.6 Proof of Lemma 2 146

A.7 Lemma 4 and Its Proof 148

A.8 Proof of Theorem 2 149

A.9 Proof of Theorem 3 150

A.10 Proof of Theorem 4 156

A.11 Proof of Theorem 5 157

A.12 Extended Analysis for Interested Readers 158

A.13 Dimension-Scalability Analysis of LDP-based Approaches 158

A.13.1 Duchi's Mechanism (DM) [51, 201] 158

A.13.2 Piecewise Mechanism (PM) [201] 160

A.13.3 Hybrid Mechanism (HM) [201] 162

A.13.4 Three Outputs Mechanism [226] 162

A.13.5 OME [121] 164

A.14 Adaptive Privacy Budget in OME 167

A.15 Adaptive Privacy Budget in LATENT 170

A.16 RMSE Comparison in Mean Estimation 173

A.16.1 Supplementary Theoretical Results for ScalableRR 174

A.17 Expected Error at The Bit-Level 175

TABLE OF CONTENTS
(Continued)

Chapter Page

A.18 Supplementary Experimental Results for ScalableRR 176

A.19 Revisiting Randomized Response Mechanisms for LDP 180

A.20 Proof of Theorem 6 182

A.21 A Primer on Certified Robustness 182

A.22 Proof of Theorem 8 185

A.23 Certified Robustness Bound using boosting randomized smoothing. 188

A.24 Experimental settings and results 189

A.25 Visualizing XRand 191

A.26 Proof of Theorem 9 192

A.27 Proof of Theorem 10 192

A.28 L2DP-ML with Streaming Batch Training 195

A.29 HARW Dataset 195

A.30 Hyper-parameter Grid-Search and Supplemental Results 198

References 213

LIST OF TABLES

Table | Page

2.1 Description of Sensitive Entity Categories 30

2.2 Breakdown of CONLL-2003, AG, and SEC Datasets 32

3.1 Dataset Breakdown 60

3.2 Macro-model Performance for HAR-Wild for Different Types of Privacy Protection Mechanisms and Different Parameters 67

5.1 Average Forgetting Measure 101

5.2 Average Forgetting Measure on Random Orders of HARW Tasks 103

6.1 Data Statistical Analysis 126

6.2 AC and SC in Drug Abuse 126

A.1 Summarizing $(\gamma, \gamma_\epsilon)$-Dimension-Scalability of LDP-preserving RR Mechanisms . . 166

A.2 Several Empirical Values of α in Equation A.59 as a Function of ϵ_X when $l = 10$, $r = 1,000$, and $\delta = 1e-4$ 170

A.3 Results of LDP-FL without Privacy Accumulation on the AG Dataset 177

A.4 Statistics of the HARW Dataset 197

A.5 Baseline Results On The HARW Dataset 198

A.6 Average Accuracy (%) in Hyper-parameter Grid-search of NaiveGaussian Mechanism Given the Permuted CIFAR-10 Dataset 199

A.7 Average Forgetting Measure 200

A.8 Average Forgetting of the Order of [20Hz, 5Hz, 10Hz, 50Hz] 200

LIST OF FIGURES

Figure Page

2.1 User-Entity DP framework. 12

2.2 Next word prediction results using the GPT-2 model. 21

2.3 Text classification results on the AG dataset using the BERT model. 25

2.4 Privacy budget of UeDP-Alg $f_{\mathcal{E}}$, UeDP-Alg $f_{\mathcal{E}^+}$, and User-level DP as a function of iterations in CONLL-2003, AG, and SEC datasets. 26

2.5 Next word prediction results using the GPT-2 model with varying extended sensitive entities sampling rate q_s in training. 26

2.6 Next word prediction results with AWD-LSTM model. 26

2.7 Next word prediction results using the AWD-LSTM model with varying extended sensitive entities sampling rate q_s in training. 27

3.1 a) Privacy threat from the curious server in FL, b) Data reconstruction attacks from gradients, and c) from embeddings. 34

3.2 Binary representation. 42

3.3 Randomization probability q_i as a function of α with $\epsilon_X = 1.0$ and $r = 10,000$. 48

3.4 Different measurements for comparing dimension-scalability among LDP-based techniques. 50

3.5 Expected error for an embedding feature as a function of: (a) user-predefined budget ϵ_X, (b) expansion factor d. 54

3.6 Expected error at the bit-level ($\epsilon_X = 1.0$, $r = 10,000$) 55

3.7 Randomization probability q_i as a function of d. 56

3.8 Randomization probability q_i at the bit-level ($r = 10,000$, $\epsilon_X = 1$). 57

3.9 AUC values of LDP algorithms applied on the embedding features e in the AG, SEC, and FEMNIST datasets. 60

3.10 AUC values of LDP algorithms applied on the embedding features e in the CelebA dataset. 61

3.11 AUC values of LDP algorithms applied applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191]. 63

3.12 AUC values of LDP algorithms applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191] in the CelebA dataset. 63

LIST OF FIGURES
(Continued)

Figure Page

3.13 AUC values of ScalableRR and Adaptive OME when varying the factor d ($r = 1$) in the AG dataset. 64

3.14 ScalableRR v.s. reconstruction attacks. 65

3.15 Comparison of FL HAR-Wild Versions, with and without Data Augmentation, and with and without Privacy Protection. 66

4.1 System model of a cloud-hosted malware classifier that leverages crowdsourced data for model training. 72

4.2 Attack success rate as a function of privacy budget ε and the portion of poisoned samples on LightGBM and EmberNN. 82

4.3 Attack success rate as a function of trigger size and privacy budget ε. 83

4.4 Log-odds score of the explanations of 20,000 goodware and malware samples. 83

4.5 Visualizing the SHAP explanation and our XRAND explanation of a test sample. 84

5.1 Network design of L2DP-ML. 93

5.2 Gradient update in L2DP-ML. 93

5.3 Average accuracy in the a) Permuted MNIST (20 tasks), b) Permuted CIFAR-10 (17 tasks), and c) HARW. 101

5.4 Average accuracy with random task orders: a) HARW 50Hz - 20Hz - 10Hz - 5Hz, b) HARW 20Hz - 50Hz - 5Hz - 10Hz, and c) HARW 20Hz - 5Hz - 10Hz - 50Hz. 102

6.1 A flow chart of the OnML approach. 113

6.2 Drug abuse ontology. 114

6.3 Visualization of a drug abuse experiment. 119

6.4 Visualization of a consumer complaint experiment. 120

6.5 Consumer complaint ontology. 123

6.6 Average score changes in consumer complaint. 127

6.7 AMT results for drug abuse dataset. 128

6.8 AMT results for consumer complaint dataset. 129

6.9 Head-to-head comparison results. 130

LIST OF FIGURES
(Continued)

Figure **Page**

6.10 Length of complaints. 131

6.11 Prediction probabilities groups. 131

A.1 Distribution of users and sentences. 146

A.2 Privacy budget of UeDP-Alg $f_{\mathcal{E}}$, UeDP-Alg $f_{\mathcal{E}^+}$, and User-level DP as a function of iterations in CONLL-2003, AG, and SEC datasets. 147

A.3 Illustrating for the root of Eq. A.59. 171

A.4 Impacts of r and l on $\frac{\epsilon_{adaptive}}{\epsilon}$ in LATENT ($\epsilon_X = 0.1$). 174

A.5 Accuracy of ScalableRR and Adaptive OME when varying the factor d ($r = 1$) in the AG dataset. 178

A.6 AUC values of each algorithm applied on the gradients $\triangle\theta_t^u$ in the CelebA dataset. 201

A.7 Randomization probability q ($p = 1 - q$) as a function of r with fixed ϵ_X. . . 202

A.8 Randomization probability q_i ($p_i = 1 - q_i$) as a function of l with fixed r and ϵ. 203

A.9 RMSE comparison as a function of ϵ_X. 204

A.10 Expected error comparison between ScalableRR and OME. 204

A.11 Expected error at the bit-level as a function of ϵ_X with $r = 10,000$. 204

A.12 Expected error at the bit-level as a function of d with $\epsilon_X = 1.0$. 205

A.13 Expected error at the bit-level (e.g., the sign bit (highest important bit) and the lowest fraction bit (lowest important bit) as a function of d with $r = 1$. . . 205

A.14 Randomization probability q_i given $r = 10,000$. 205

A.15 Accuracy of LDP algorithms applied on the embedding features e in the AG, SEC, and FEMNIST datasets. 206

A.16 Accuracy of LDP algorithms applied on the gradients $\triangle\theta_t^u$ in the AG, SEC, and FEMNIST datasets. 206

A.17 Accuracy of LDP algorithms applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191]. 206

A.18 Accuracy of each mechanism applied on labels. 206

A.19 AUC values of LDP algorithms applied on the gradients $\triangle\theta_t^u$ in the AG, SEC, and FEMNIST datasets. 207

LIST OF FIGURES
(Continued)

Figure Page

A.20 AUC values of each mechanism applied on labels in the AG, SEC, and FEMNIST datasets. 207

A.21 SSE values as a function of ϵ and τ 207

A.22 Attack success rate as a function of privacy budget ε and the portion of poisoned samples on the Contagio PDF and Drebin datasets. 208

A.23 Certified robustness at the training time. The smaller privacy budget ε, the higher certified accuracy. 208

A.24 Visualizing the SHAP explanation and our XRAND explanation of test sample 1.209

A.25 Visualizing the SHAP explanation and XRAND explanation of test sample 2. 209

A.26 Visualizing the SHAP explanation and XRAND explanation of test sample 3. 210

A.27 p value for 2-tail t-tests on the (a) Permuted MNIST (20 tasks), b) Permuted CIFAR-10 (17 tasks), and (c) HARW (5Hz - 10Hz - 20Hz - 50Hz). 210

A.28 p value for 2-tail t-tests on the HARW dataset with random task orders: (a) HARW 50Hz - 20Hz - 10Hz - 5Hz, (b) HARW 20Hz - 50Hz - 5Hz - 10Hz, and (c) HARW 20Hz - 5Hz - 10Hz - 50Hz. 212

A.29 Average accuracy in the (a) Split MNIST (5 tasks), and b) Split CIFAR-10 and CIFAR-100 (11 tasks). 212

CHAPTER 1

INTRODUCTION

1.1 Motivation

Machine learning (ML) has gained significant attention in research due to its essential role in many recent scientific and technological breakthroughs [34, 44, 103, 105, 158]. Among these, the proliferation of ML systems in real-world applications has brought about important considerations regarding the trustworthiness of ML-based systems, including explanability, privacy preservation, security, and robustness. Many ML models remain black-box and are severely vulnerable to privacy attacks, therefore, it is critical to 1) understand how ML models work, 2) whether the trained models reveal about training data, and 3) how robust the model outcomes are. Answering these questions helps to bring trustworthy ML closer to real-world adoption with the trust from communities. Our main goal is to provide rigorous trustworthy ML models by addressing fundamental trustworthiness challenges in ML, i.e., privacy and security, explanability, and robustness, to allow broad applicability of ML-based systems and gain trust of end-users on using these systems.

Recent data privacy and security regulations, e.g., HIPAA/HITECH, pose major challenges in collecting and using personally sensitive data in ML applications. Federated learning (FL) is a promising technique to tackle the challenges by enabling clients to jointly train ML models via sharing and aggregating gradients computed from clients' local data through a server without sharing their data. Technically, recent attacks have shown that 1) deployed ML models can be used to accurately extract sensitive entities in private training data and to identify a data owner by the extracted names or unique phrases, and 2) clients' training data can be perfectly extracted from the shared gradients in FL. Furthermore, to improve transparency of ML systems, explanations are typically provided along with model predictions, such as in Machine-Learning-as-a-Service (MLaaS) systems.

Importantly, explanations also open a door for adversaries to gain insights into the black-box models in MLaaS, thereby making the models more vulnerable to several attacks. A recent explanation-guided backdoor attack (XBA) [178] was successfully carried out to misclassify a targeted label by exploiting the correlation between the model prediction and explainability to poison the data. These attacks underscore privacy risks in ML-based systems.

In addition to solve privacy and security issues of a single ML task, we also work with lifelong learning (L2M), which is crucial for machine learning (ML) to acquire new skills through continual learning, pushing ML toward a more human learning in reality. Given a stream of different tasks and data, a deep neural network (DNN) can quickly learn a new task, by leveraging the acquired knowledge after learning previous tasks, under constraints in terms of the amount of computing and memory required [30]. As a result, it is quite challenging to train an L2M model with a high utility. Orthogonal to this, L2M models are vulnerable to adversarial attacks, i.e., privacy model attacks [70, 146, 183, 203], when DNNs are trained on highly sensitive data, e.g., clinical records [33, 135], user profiles [171, 207], and medical images [83, 156].

1.2 Technical Challenges

Addressing the aforementioned problems is a non-trivial task, since: ***First,*** there is still a lack of scientific studies on how to effectively protect privacy for both users and their confidential information, i.e., sensitive entities in users' textual data, with an optimized trade-off between privacy loss and model utility; ***Second,*** existing privacy preserving mechanisms in FL typically suffer from the curse of dimensionality, rooted in the excessive privacy budget consumption proportional to the large dimensions of input features, gradients, and communication rounds between clients and the server, causing loose privacy protection or inferior model accuracy. This is challenging for these techniques to achieve good model utility under tight privacy guarantees given complex models and tasks, especially when deploying the models in practice; ***Third,*** there has not been scientific studies on protecting explanations meanwhile maintaining the faithfulness of explanations without affecting model performance; and ***Fourth,*** there is a lack of study offering privacy protection to the training data in L2M, in which memorizing previous tasks while learning new tasks further exposes private information in the training set, by continuously accessing the data from the previously learned tasks (i.e., data stored in an episodic memory [30,167,193]); or accessing adversarial examples produced from generative memories to imitate real examples of past tasks [144,180,205].

To address these challenges, first to protect users' data privacy, the naive solution is to anonymize (i.e., removing) sensitive entities. Heuristically, this technique is insufficient since the anonymized entities can be matched with non-anonymized data records in a different dataset [60]. While secure multi-party computation [75, 79, 200, 214] and homomorphic encryption [7,25,72,74,168,200] can be applied to protect privacy, these techniques usually come with a huge computation and resource overhead. Thus, we proposed to apply differential privacy [58], one of the most adequate solutions, given its formal privacy protection without undue sacrifice in computation efficiency and model utility. Differential privacy (DP) provides rigorous privacy protection as a probabilistic term, limiting the knowledge about a data record a ML model can leak while learning features of the whole training set [4, 66, 110, 137, 152, 153, 181, 190, 219]. Principally, existing DP levels of protection, including sample-level DP [4,20,60,170,208], user-level DP [128,162], element-level DP [15], and local feature-level DP [51,63,120,121], do not provide the privacy protection level demanded to solve the privacy implication of both users and sensitive entities. Given training data: 1) Sample-level DP protects privacy of a single sample. 2) User-level DP protects privacy of a single data owner, also called a single user, who may contribute one or more data samples. 3) Element-level DP partitions data owners' contribution to the training data into sensitive elements, e.g., a curse word, which will be protected. Element-level DP does not provide privacy protection to data owners. And 4) Local (feature-level) DP protects true values of a data sample from being inferred. There is a demand for a new level of DP to protect privacy simultaneously for both sensitive entities in the training data and the participation information of data owners in learning natural language modeling (NLM).

Second, to offer rigorous privacy protection without computational overheads, local differential privacy (LDP) [21, 39, 51, 63, 112, 142, 174, 204, 224] has emerged as a crucial component in a variety of FL applications [27, 116, 121, 176, 191, 196, 226] in both cross-device [26, 97, 99, 197, 218] and cross-silo [53, 82, 97, 220] settings. In cross-device FL, clients jointly train an FL model. existing LDP-preserving approaches can be categorized into two lines: **(1)** clients add noise to local gradients using RR mechanisms to protect the values of the local gradients [116, 191, 201, 226], and **(2)** clients add noise into each training sample using RR mechanisms to protect the value of each training sample [13, 121, 210], then the clients use these perturbed samples to derive local gradients. For both approaches, clients send LDP-preserved local gradients to the server for model updates in each training round. Unlike the classical survey applications of RR [96, 204], in which one-time collection for a survey question could be sufficient to estimate the population statistics accurately, a gradient perturbation-based RR mechanism in FL needs to be repeatedly applied over the training rounds [116, 191]. Consequently, the LDP guarantee for a local training data sample of a client significantly degrades (i.e., consumes a larger privacy budget) proportional to the number of communication rounds between the client and the coordinating server [201, 226]. In addition, when the number of elements in the local gradient vector (or the number of embedding features in data perturbation-based RR mechanisms) increases, the privacy budget has to increase accordingly to maintain utility [201, 226]. Mitigating the curse of dimensionality is non-trivial and remains an open problem. Existing approaches, such as anonymizers (assumed to be trusted), i.e., shuffler [32, 115] or anonymity approaches (e.g., faking source IP, VPN, mixnets, etc. [40, 191]), and dimension reduction approaches [116, 226], lessen the problem but also have limitations. The anonymizers could be compromised or collude with the server to extract sensitive information from local gradients [62]. Applying

RR mechanisms on reduced sets of embedding features or gradients using dimension reduction can work well with lightweight models [116, 201, 226]. Empirically, it becomes challenging for these techniques to achieve good utility under rigorous LDP guarantees with complex models and tasks, such as deep neural networks (DNNs). In complex models, the dimensions of the features, gradients, and training rounds are still large even after dimensionality reduction for acceptable model performance.

Third, despite the great potential of explainers to improve the transparency and understanding of ML models in MLaaS, they open a trade-off in terms of security. Specifically, they allow adversaries to gain insights into black-box models, essentially uncovering certain aspects of the models that make them more vulnerable. Such an attack vector has recently been exploited by the research community to conduct several *explanation-guided attacks* [134, 138, 178, 182, 225]. It was shown that an explainer may expose the top important features on which a black-box model is focusing, by aggregating over the explanations of multiple samples. An example of utilizing such information is the recent highly effective explanation-guided backdoor attack (XBA) against malware classifiers investigated by [178]. The authors suggest that SHAP can be used to extract the top-k goodware-oriented features. The attacker then selects a combination of those features and their values for crafting a trigger; and injects trigger-embedded goodware samples into the training dataset of a malware classifier, with an aim of changing the prediction of malware samples embedded with the same trigger at inference time. To prevent an adversary from exploiting the explanations, we need to control the information leaked through them, especially the top-k features. Since the explanation on each queried sample is returned to the end users, a viable defense is to randomize the explanation such that it is difficult for attackers to distinguish top-k features while maintaining valuable explainability for the decision-making process. A well-applied technique to achieve this is preserving local differential privacy (LDP) [63, 202] on the explanations. Technically, existing LDP-based approaches [63, 191, 209, 226] are not designed to protect the top-k features aggregated over the returned explanations on queried samples. Therefore, optimizing the trade-off between defenses against explanation-guided attacks and model explainability is an open problem.

Fourth, to preserve DP in L2M, one can employ the moments accountant [4] to train the model by injecting Gaussian noise into clipped gradients. The post-processing property in DP [59] can be applied to guarantee that gradients are also DP. To optimize this naive approach, one can adapt the management policy [109] to redistribute the privacy budget across tasks while limiting the total privacy budget ϵ to be smaller than a predefined upper bound, that is, the training will be terminated when ϵ reaches the predefined upper bound. Importantly, the challenge in bounding the privacy risk is still the same, centering around the growing number of tasks m and the heterogeneity among tasks: **(1)** The larger the number of tasks, the larger the privacy budget will be consumed by the $\sum$ function. It is difficult to identify an upper bound privacy budget given an unlimited number of streaming tasks in L2M; **(2)** Different tasks may require different numbers of training steps due to the difference in terms of the number of tuples in each task; thus, affecting the privacy budget ϵ; and **(3)** The order of training tasks also affect the privacy budget, since computing g_{ref} by using data in the episodic memory from one task may be more than other tasks. Therefore, bounding the DP budget in L2M is non-trivial.

1.3 Key Contributions

To overcome these aforementioned challenging issues, this study is structured around the following key contributions:

In our first work (Chapter 2), we propose a novel notion of user-entity adjacent databases (**Definition 2**), leading to formal guarantees of user-entity privacy rather than privacy for a single user or a single sensitive entity. To preserve UeDP, we introduce a novel algorithm, called **UeDP-Alg**, which leverages the recipe of DP-FEDAVG [128] to protect both sensitive entities and user membership under DP via the moments accountant [4]. Moments accountant was first developed to preserve DP in stochastic gradient descent

(SGD) for sample-level privacy. Our federated averaging approach groups multiple SGD updates computed from a two-level random sampling process, including a random sample of users and a random sample of sensitive entities. That enables large-step model updates and optimizes the trade-off between privacy loss and the model utility through a tight noise scale bound (**Lemma 1** and **Theorem 1**). Through theoretical analysis and rigorous experiments conducted on benchmark datasets, we show that our UeDP-Alg outperforms baseline approaches in terms of model utility on fundamental tasks, i.e., next word prediction and text classification, under the same privacy budget consumption.

In the second work (Chapter 3), we we formally study the impact of dimensions for LDP and propose ScalableRR as a novel mechanism for mitigating the curse of dimensionality in LDP for FL settings. Different from existing approaches, ScalableRR integrates dimensionality, data utility, and LDP into a unified framework for optimized randomization probabilities. As a result, ScalableRR can i) derive tighter privacy loss bounds and ii) achieve small expected error (i.e., measuring the expected change of a feature before and after the randomization). Smaller expected errors indicate better data utility under the same privacy loss. Hence, ScalableRR effectively addresses the trade-off between data utility, privacy, and dimensionality. In other words, ScalableRR achieves strong LDP guarantees of the clients' data without undue sacrifice in utility as dimensionality increases. To demonstrate that, we first introduce the notion of ***dimension-scalability*** to capture the correlation between randomization probabilities, dimensionality, and the ϵ-LDP guarantee of a training data sample. We define dimension-scalability as *the ability of an approach to maintain marginal to no degradation in the randomization probabilities and the level of ϵ-LDP guarantee when the dimensionality increases, resulting in better data utility*. We quantify, with increasing dimensionality: **(1)** the magnitude of degradation in the randomization probabilities (i.e., causing more noisy data for model training), given a fixed level of ϵ-LDP protection; and **(2)** the magnitude of degradation in the level of ϵ-LDP guarantee (i.e., causing looser privacy protection), given fixed randomization probabilities.

We show that ScalableRR is a dimension-scalable approach. In other words, it can work with large numbers of features and large models combined with an unlimited number of training rounds with only a marginal impact on the level of ϵ-LDP guarantee and data utility. ScalableRR can be used as a pre-processing step to create "noisy" local data that can be stored and reused in place of the original local data. This feature makes ScalableRR a permanent RR mechanism, which provides longitudinal LDP protection [63].

In the third work (Chapter 4), we introduce 1) a new concept of achieving LDP in model explanations that simultaneously protects the important features from being exploited by attackers while maintaining the faithfulness of the explanations. Based on this principle, we propose a defense against explanation-guided attacks on MLaaS, called XRAND, by devising a novel two-step LDP-preserving mechanism. First, at the aggregated explanation, we incorporate the explanation loss into the randomized probabilities in LDP to make important features indistinguishable to the attackers. Second, at the sample-level explanation, guided by the first step, we minimize the explanation loss on each sample while keeping the features at the aggregated explanation intact. 2) Then, we theoretically analyze the robustness of our defense against the XBA in MLaaS by establishing two certified robustness bounds in both training time and inference time. 3) Finally, we evaluate the effectiveness of XRAND in mitigating the XBA on cloud-hosted malware classifiers.

The fourth work (Chapter 5) is to preserve differential privacy (DP) [58], a rigorous formulation of privacy in probabilistic terms, in L2M. We introduce a new definition of lifelong differential privacy (Lifelong DP), in which the participation of any data tuple in any tasks is protected under a *consistently bounded* DP guarantee, given the released parameters in both learning new tasks and memorizing previous tasks (Definition 9). This is significant by allowing us to train and release new versions of an L2M model, given a stream of tasks and data, under DP protection. Based upon this, we propose a novel L2DP-ML algorithm to preserve Lifelong DP. In L2DP-ML, privacy-preserving noise is injected into inputs and hidden layers to achieve DP in learning private model parameters in each task (Alg. 3).

Then, we configure the episodic memory as a stream of fixed and disjoint batches of data, to efficiently achieve Lifelong DP (Theorem 10). The previous task memorizing constraint is solved, by inheriting the recipe of the well-known A-gem algorithm [30], under Lifelong DP. To our knowledge, our study establishes a formal connection between DP preservation and L2M given a growing number of learning tasks compared with existing works [67, 150]. Rigorous experiments, conducted on permuted MNIST [101], permuted CIFAR-10 datasets, and an L2M task on our collected dataset for human activity recognition in the wild show promising results in preserving DP in L2M.

1.4 Organization

In the next chapter, we present our user-entity DP mechanism, i.e., UeDP on simultaneously preserving for both sensitive entities in the training data and the participation information of data owners in learning NLMs. In Chapter 3, we introduce our dimension-scalable mechanism, called ScalabelRR, which mitigates the curse of dimensionality in LDP preservation in federated learning. Chapter 4 is to establish the connection between explainable AI, privacy, and robustness. Chapter 5 is to preserve DP in lifelong machine learning in a consistenly bounded privacy budget, given the heterogeneity in terms of data sizes and the training order of tasks, without affecting DP protection of the private training set. An extra work in interpretability, i.e., OnML to generate semantic explanations is presented in Chapter 6. Chapter 7 consists of concluding remarks and some potential future work following our current work.

CHAPTER 2

USER-ENTITY DIFFERENTIAL PRIVACY IN LEARNING NATURAL LANGUAGE MODELS

2.1 Preamble

In this chapter, we introduce a novel concept of user-entity differential privacy (UeDP) to provide formal privacy protection simultaneously to both sensitive entities in textual data and data owners in learning natural language models. To preserve UeDP, we developed a novel algorithm, called UeDP-Alg, optimizing the trade-off between privacy loss and model utility with a tight sensitivity bound. An extensive theoretical analysis and evaluation show that our UeDP-Alg outperforms baseline approaches in terms of model utility under the same privacy budget consumption on several NLM tasks, using benchmark datasets.

2.2 Background

In this section, we revisit NLM tasks, privacy risk, and differential privacy. For the sake of clarity, let us focus on the next word prediction, and we will extend it to text classification in Section 2.4. A list of sensitive entity categories is summarized in Table 2.1.

2.2.1 Next Word Prediction

Let D be a private training data containing U users (data owners) and a set of sensitive entities E. Each user $u \in U$ consists of n_u sentences. Given a vocabulary $\mathcal{V}$, each sentence is a sequence of words, presented as $x = x_1 x_2 \dots x_{m_u}$, where $x_i \in \mathcal{V}, (i \in [1, m_u])$ is a word in x and m_u is the length of x. In next word prediction, the first j words in x, i.e., $x_1, x_2, \dots, x_j$ ($\forall j < m_u$), are used to predict the next word x_{j+1}. Here, x_{j+1} can be considered as a label in the next word prediction task. Perplexity $PP = 2^{-\sum_{x \in D} p(x) \log_2 p(x)}$ is a measurement of how well a model predicts a sentence and is often used to evaluate language models, where $p(x)$ is a probability to predict the next word x_{j+1} in x [132]. A lower perplexity indicates a better model.

2.2.2 Sensitive Entities and Sentences

Each sensitive entity $e \in E$ consists of a word or consecutive words that must be protected. For instance, personal identifiable information (PII) related to an identifiable person, such as person names, locations, and phone numbers, can be considered sensitive entities. If a sentence x consists of a sensitive entity e, x is considered as a sensitive sentence; otherwise, x is a non-sensitive sentence.

For instance, in Figure 2.1, "David Johnson," "Maine," "September 18," and "Main Hospital" are considered sensitive entities, correspondingly categorized into PII, geopolitical entities (GPE) (i.e., countries, cities, and states), time, and organization names. The first and second sentences consisting of the sensitive entities are considered sensitive sentences. Meanwhile, the third and fourth sentences are non-sensitive since they do not contain any sensitive entities.

2.2.3 Privacy Threat Models

It is well-known that trained ML model parameters can disclose information about training data [28, 55], especially in NLMs [28, 128]. Given a data sample and model parameters, by using a membership inference attack [172, 183, 216], adversaries can infer whether the training used the sample or not. In NLMs, adversaries can accurately recover individual training examples, such as names, email addresses, and phone numbers of individuals, using training data extracting attacks [28]. Accessing to these can lead to severe privacy breaches.

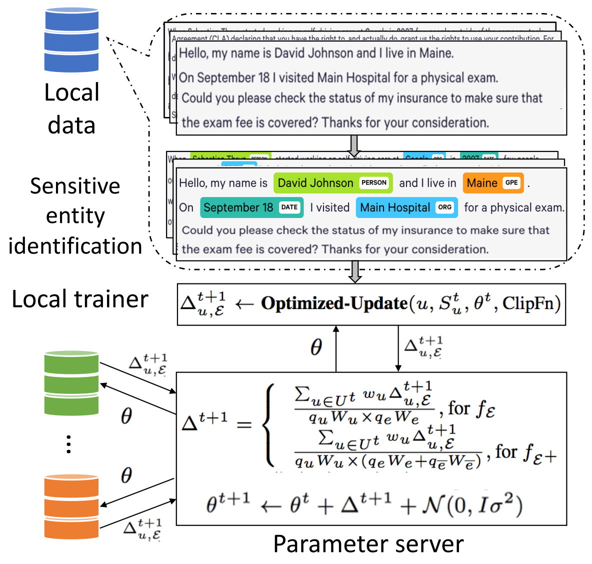

Local data
Hello, my name is David Johnson and I live in Maine.
On September 18 I visited Main Hospital for a physical exam.
Could you please check the status of my insurance to make sure that the exam fee is covered? Thanks for your consideration.
Sensitive entity identification
Hello, my name is David Johnson PERSON and I live in Maine GPE .
On September 18 DATE I visited Main Hospital ORG for a physical exam.
Could you please check the status of my insurance to make sure that the exam fee is covered? Thanks for your consideration.
Local trainer
$\Delta^{t+1}_{u,\mathcal{E}} \leftarrow \textbf{Optimized-Update}(u, S^t_u, \theta^t, \text{ClipFn})$
θ
$\Delta^{t+1}_{u,\mathcal{E}}$
$\Delta^{t+1} = \begin{cases} \frac{\sum_{u \in U^t} w_u \Delta^{t+1}_{u,\mathcal{E}}}{q_u W_u \times q_e W_e}, \text{for } f_{\mathcal{E}} \\ \frac{\sum_{u \in U^t} w_u \Delta^{t+1}_{u,\mathcal{E}}}{q_u W_u \times (q_e W_e + q_{\bar{e}} W_{\bar{e}})}, \text{for } f_{\mathcal{E}+} \end{cases}$
$\theta^{t+1} \leftarrow \theta^t + \Delta^{t+1} + \mathcal{N}(0, I\sigma^2)$
Parameter server

Figure 2.1 User-Entity DP framework.

2.2.4 Different Levels of DP

To avoid these privacy risks, DP guarantees restriction on the adversaries in what they can learn from the training data given the model parameters by ensuring similar model outcomes with and without any single training sample. Let us revisit the definition of DP, as follows:

Definition 1. *(ϵ, δ)-DP [58]. A randomized algorithm $\mathcal{A}$ fulfills (ϵ, δ)-DP, if for any two adjacent datasets D and D' differing by at most one sample, and for all outcomes $\mathcal{O} \subseteq Range(\mathcal{A})$:*

$$Pr[\mathcal{A}(D) = \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(D') = \mathcal{O}] + \delta \tag{2.1}$$

with a privacy budget ϵ and a broken probability δ.

The privacy budget ϵ controls the amount by which the distributions induced by D and D' may differ. A smaller ϵ enforces a stronger privacy guarantee. The broken probability δ means the highly unlikely "bad" events, in which an adversary can infer whether a particular data sample belongs to the training data, happen with the probability $\leq \delta$.

There are different levels of DP protection in literature categorized into four research lines, including sample-level DP, user-level DP, element-level DP, and local (feature-level)

DP. They are different from our goal since we focus on providing simultaneous protections to data owners and sensitive entities in textual data. Let us revisit these DP levels and distinguish them with our goal.

Sample-level DP Traditional DP mechanisms [60, 145, 170] ensure DP at the sample-level, in which adjacent datasets D and D' are different from at most a single training sample. Sample-level DP does not protect privacy for users. That is different from our goal. We aim at protecting privacy for users and sensitive entities, which are different from data samples.

User-level DP To protect privacy for users, who may contribute more than one training sample, rather than a single sample, [128] proposed a user-level DP, in which neighboring databases D and D' are defined to be different from all of the samples associated with an arbitrary user in the training set. Several works follow this direction [97, 162]. User-level DP differs from our goal, since it does not provide privacy protection for sensitive entities in the training set.

Element-level DP [15] introduce element-level DP, in which users are partitioned based on sensitive elements, which are protected in a way that an adversary cannot infer whether a user has a sensitive element in her/his data, e.g., if a user has ever sent a curse word in his/her messages or not. Similar to sample-level DP, element-level DP is different from our goal, since it does not provide DP protection for users.

Local (feature-level) DP [120] proposed word-level local DP for a sentence's embedding features, in which two adjacent sentences x and x' are different at most one word:

$$Pr[\mathcal{A}(f(x)) = \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(f(x')) = \mathcal{O}] \tag{2.2}$$

where $f(x)$ extracts embedded features of x and $\mathcal{A}$ is a randomized algorithm, such as a Laplace mechanism [60]. In a similar effort, [121] applied a randomized response mechanism [13, 63, 202] on top of binary encoding of embedded features' real values to achieve local DP feature embedding. The approaches proposed in [120, 121] are different from our goal, since they do not offer either user-level DP or word-level DP.

In this section, we focus on answering the question: *"Could we protect sensitive entities and user membership simultaneously by leveraging existing levels of DP and how?"* Based upon that, we propose our user-entity DP notion.

2.2.5 Sensitive Entities and User Membership

To protect sensitive entities and user membership, a potential approach is to decouple them into separated protection levels offering by existing DP notions. Heuristically, this approach has limitations as discussed next.

Let us consider a sentence consisting of one or more than one sensitive entities. We can leverage sample-level DP to protect the sentence, i.e., each sentence could be a sample, covering all the sensitive entities under DP. If each user has only one sentence, then this approach can also protect the user membership. In practice, one user may contribute many sentences to the training data. To address this issue, we can utilize group privacy [60] resulting in an amplification of the privacy budget proportional to the number of sentences a user may have in the training data.

Instead of group privacy, another potential solution is applying user-level DP on top of the sample-level DP to protect both sentence and user membership. In the sample-level DP, we can clip and inject Gaussian noise into the gradient derived from each sentence [4]. Meanwhile, in the user-level DP, an additional Gaussian noise is injected into the aggregation of gradients, each of which derived from a single user [128]. Although this combination of sample - user levels can cover both sensitive entities and user membership under DP protection, it has disadvantages. First, some sentences are sensitive and other sentences are not. Protecting all (sensitive and non-sensitive) sentences or removing all the sensitive sentences from the training data may cause significant model utility degradation. Second, different sentences may consist of different types and numbers of sensitive entities. Under the same sampling probability for training as in [4] for sample-level DP, these sentences expose different privacy risks to user identity and sensitive entities.

To address these issues, instead of the sentence level, one can work at the word level by extracting embedded features for every words in the training data. Embedded features of sensitive entities are randomized by local DP-preserving mechanisms [13]. The randomized embedded features are aggregated with embedded features of non-sensitive words to train NLMs. Then, user-level DP can be applied to clip gradients derived from each user's data with adding Gaussian noise into the aggregation of these gradients. Fundamentally, this approach suffers from a remarkable model utility degradation. Local DP provides rigorous privacy protection but it comes with a cost in terms of utility [200]. Then, adding the user-level DP adversely affects the utility.

The root cause of these limitations is that the combination of sentence-level DP and user-level DP notions does not capture the correlation between sensitive entities and user membership in unifying notion of DP. Meanwhile, working with word-level embedded features under local DP introduces expensive model utility costs. Therefore, there is a demand for a unifying notion of DP and an optimal approach to protect both sensitive entities and user membership in training NLMs.

2.2.6 UeDP Definition

To preserve privacy for both users and sensitive entities in NLMs, we propose a new definition of user-entity adjacent databases, as follows: Two databases D and D' are user-entity adjacent if they differ in a single user and a single sensitive entity; that is, one user u' and one sensitive entity e' are present in one database (i.e., D') and are absent in the other (i.e., D). Together with the absence of all sentences from the user u' in D, all sentences (across users) consisting of the sensitive entity e' are also absent in D. This is because one user can have multiple sentences, and one sensitive entity can exist in multiple sentences for training. The definition of our user-entity adjacent databases is as follows:

Definition 2. *User-Entity Adjacent Databases. Two databases D and D' are called user-entity adjacent if: $\|U - U'\|_1 \leq 1$ and $\|E - E'\|_1 \leq 1$, where U and E are the sets of users and sensitive entities in D, and U' and E' are the sets of users and sensitive entities in D'.*

Given the user-entity adjacent databases, we present UeDP in the following definition.

Definition 3. *(ϵ, δ)-UeDP. A randomized algorithm $\mathcal{A}$ is (ϵ, δ)-UeDP if for all outcomes $\mathcal{O} \subseteq Range(\mathcal{A})$ and for all user-entity adjacent databases D and D', we have:*

$$Pr[\mathcal{A}(D) = \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(D') = \mathcal{O}] + \delta \tag{2.3}$$

with a privacy budget ϵ and a broken probability δ.

2.3 Preserving UeDP in NLMs

UeDP provides rigorous privacy protection to both users and sensitive entities; however, the practicability of UeDP preservation depends on the reliability of sensitive entity detection from the training text data. In practice, misidentifying sensitive entities can introduce extra privacy risks. In addition to addressing this challenge, we focus on bounding the sensitivity of an NLM under UeDP and addressing the trade-off between privacy loss and model utility.

2.3.1 Misidentifying Sensitive Entities

Identifying all the sensitive entities typically requires intensive manual efforts [212]. We are aware of this issue in real-world applications. Fortunately, there are several ways to automatically identify sensitive entities in textual data, such as: 1) Using Named Entity Recognition (NER) [45, 173]; and 2) Using publicly available toolkits for detecting named entities or PII in text, e.g., spaCy [85], Stanza [157], and Microsoft Presidio[1]. These

[1]`https://microsoft.github.io/presidio/`

approaches and toolkits are user-friendly and reliable to reduce manual efforts in identifying sensitive entities and information. We found that the results from spaCy cover over 94% of sensitive information identified by Amazon Mechanical Turk (AMT) workers in a diverse set of datasets used in our experiments. More information about identifying sensitive entities is available in Appendix A.1.

Although effective, the small error rate (i.e., $\approxeq 6\%$) from these techniques introduces a certain level of privacy risk, that means, some sensitive entities may be misidentified to be non-sensitive, and vice-versa. Classifying non-sensitive entities to be sensitive entities does not incur any extra privacy risk. Meanwhile, classifying one (or more than one) sensitive entity to be non-sensitive in a sentence introduces two issues, as follows: **(1)** There may be sensitive sentences misidentified to become non-sensitive sentences. In order words, given a set of non-sensitive sentences detected by NER tools, we do not know which sentence is truly non-sensitive; and **(2)** Given a sensitive sentence x, some sensitive entities in x may not be identified by NER tools. Preserving UeDP in NLMs by directly using the results of NER tools will expose these misidentifying sensitive sentences and entities unprotected.

2.3.2 Preserving UeDP

To address the problem of sensitive entity misidentification in preserving UeDP, our key idea is:

(1) Extending UeDP by considering each sentence, identified to be non-sensitive using NER tools [85, 157], in the private training dataset as a single type of sensitive entity. *We denote this extended set of sensitive entities as S.* The private dataset D now consists of U users and a (sufficient) set of sensitive entities $E \cup S$ that will be protected.

(2) Upon forming the sufficient set of sensitive entities, we propose a two-step sampling approach to strictly preserve UeDP in NLMs. In our approach, at a training round t, we sample a set of users from U and a set of sensitive entities from $E \cup S$. We use sentences in the training data of the sampled users consisting of the sampled sensitive entities to train NLMs. In this sampling approach: (i) If a sensitive sentence x is not sampled for training, i.e., due to the fact that some sensitive entities in x are not identified by NER tools, x is not used for training at the round t; thus avoiding privacy risks exposed by x; and (ii) If the sensitive sentence x is sampled for training, then the sensitive entities in x, which are not identified by NER tools, are protected since x is protected under DP.

By covering all possible cases of sensitive entity misidentification, we strictly preserve UeDP without having additional privacy risks. The pseudo-code of UeDP is in Alg. 4.

At each iteration t, we randomly sample U^t users from U, E^t detected sensitive entities from E, and S^t extended sensitive entities from S, with sampling rates q_u, q_e, and q_s, respectively (Lines 8 and 10). Then, we use all sensitive sentences in $E_u^t \cup S_u^t$ consisting of the sensitive entities in E^t and S^t belonging to the selected users in U^t for training. Like [128], we leverage the basic federated learning setting in [127] to compute gradients of model parameters for a particular user, denoted as $\Delta_{u,\mathcal{E}}^{t+1}$ (Line 11). Here, we clip the per-user gradients so that its l_2-norm is bounded by a predefined gradient clipping bound β (Lines 20 - 29). Next, a weighted-average estimator $f_{\mathcal{E}^+}$ is employed to compute the average gradient Δ^{t+1} using the clipped gradients $\Delta_{u,\mathcal{E}}^{t+1}$ gathered from all the selected users (Line 13). Finally, we add random Gaussian noise $\mathcal{N}(0, I\sigma^2)$ to the model update (Line 15). During the training, the moments accountant $\mathcal{M}$ is used to compute the T training steps' privacy budget consumption (Lines 16 - 18).

To tighten the sensitivity bound, our weighted-average estimator $f_{\mathcal{E}^+}$ (Line 13) is:

$$f_{\mathcal{E}^+}(U^t, E^t) = \frac{\sum_{u \in U^t} w_u \Delta_{u,\mathcal{E}}^{t+1}}{q_u W_u (q_e W_e + q_s W_s)} \tag{2.4}$$

where $\Delta_{u,\mathcal{E}}^{t+1} = \sum_{e \in E_u^t} w_e \Delta_{u,e} + \sum_{s \in S_u^t} w_s \Delta_{u,s}$, and w_u, w_e, and $w_s \in [0, 1]$ are weights associated with a user u, a detected sensitive entity e, and an extended sensitive entity s.

These weights capture the influence of a user and sensitive entities to the model outcome. $\Delta_{u,e}$ and $\Delta_{u,s}$ are the parameter gradients computed using the sensitive entities $e \in E$ and $s \in S$. In addition, $W_u = \sum_{u \in U} w_u$, $W_e = \sum_{e \in E} w_e$, and $W_s = \sum_{s \in S} w_s$.

Since $\mathbb{E}[\sum_{e \in E_u^t} w_e + \sum_{s \in S_u^t} w_s] = q_e W_e + q_s W_s$, the estimator $f_{\mathcal{E}^+}$ is unbiased. The sensitivity of the estimator $\mathbb{S}(f_{\mathcal{E}^+})$ is computed as: $\mathbb{S}(f_{\mathcal{E}^+}) = \max_{u',e'} \| f_{\mathcal{E}^+}(\{U^t \cup u', (E^t \cup S^t) \cup e'\}) - f_{\mathcal{E}^+}(\{U^t, E^t \cup S^t\}) \|_2$. $\mathbb{S}(f_{\mathcal{E}^+})$ is bounded in the following lemma.

Lemma 1. *If for all users u we have $\|\Delta_{u,\mathcal{E}}^{t+1}\|_2 \leq \beta$, then $\mathbb{S}(f_{\mathcal{E}^+}) \leq \frac{(q_u|U|+1)\max(w_u)\beta}{q_u W_u (q_e W_e + q_s W_s)}$.*

Proof. If for all users $\|\Delta_{u,\mathcal{E}}^{t+1}\|_2 \leq \beta$, then

$$\begin{aligned}
\mathbb{S}(f_{\mathcal{E}^+}) = & \frac{\sum_{u \in U^t \cup u'} w_u \Big(\sum_{e \in E_u^t} w_e (\sum_{s \textit{ consists of } e} \Delta_{u,s}) \Big)}{q_u W_u (q_e W_e + q_s W_s)} \\
& + \frac{\sum_{u \in U^t \cup u'} w_u \Big(\sum_{s \in S_u^t} w_s \Delta_{u,s} \Big)}{q_u W_u (q_e W_e + q_s W_s)} + \frac{\sum_{u \in U^t \cup u'} w_u \Big[w_{e'} (\sum_{s \textit{ consists of } e'} \Delta_{u,s}) \Big]}{q_u W_u (q_e W_e + q_s W_s)} \\
& - \frac{\sum_{u \in U^t} w_u \Big(\sum_{e \in E_u^t} w_e (\sum_{s \textit{ consists of } e} \Delta_{u,s}) \Big)}{q_u W_u (q_e W_e + q_s W_s)} - \frac{\sum_{u \in U^t} w_u \Big(\sum_{s \in S_u^t} w_s \Delta_{u,s} \Big)}{q_u W_u (q_e W_e + q_s W_s)} \\
\leq & \frac{\sum_{u \in U^t \cup u'} [(w_u)\beta]}{q_u W_u (q_e W_e + q_s W_s)} \leq \frac{(q_u|U|+1)\max(w_u)\beta}{q_u W_u (q_e W_e + q_s W_s)}
\end{aligned} \tag{2.5}$$

Consequently, Lemma 1 holds. □

Once the sensitivity of the estimator $f_{\mathcal{E}^+}$ is bounded, we can add Gaussian noise scaled to the sensitivity $\mathbb{S}(f_{\mathcal{E}^+})$ to obtain a privacy guarantee. By applying Lemma 1, the noise scale σ becomes:

$$\sigma = z\mathbb{S}(f_{\mathcal{E}^+}) = \frac{z(q_u|U|+1)\max(w_u)\beta}{q_u W_u (q_e W_e + q_s W_s)} \tag{2.6}$$

The noise scale σ in Equation 2.6 is tighter than the noise scale in existing works [128, 162] proportional to the number of sensitive entities used in the training process (i.e., $q_e W_e + q_s W_s$). Therefore, we can inject less noise into our model under the same privacy budget while improving our model utility.

In extreme cases, that is also true: *(1) E is empty,* which means there are no detected sensitive entities. Given a fixed set of training data, while E is empty, S becomes larger (i.e., covering the whole dataset), resulting in a larger value of W_s. Therefore, we obtain a larger value of $q_s W_s$ (with a pre-defined q_s), enabling us to reduce the noise scale under the same UeDP guarantee. That is an advantage compared with the naive approach that only uses detected sensitive entities E in the training process (i.e., ignoring the term $q_s W_s$ in Equation 2.6). If E is empty, the naive approach will have no sentences for training; and *(2) S is empty,* that is, every sentence in the data consists of at least one detected sensitive entity $e \in E$. Similarly, given a fixed set of training data, if S is empty, then E and W_e become larger. It enables us to obtain a larger value of $q_e W_e$ (with a pre-defined q_e), which results in smaller noise scale while maintaining the high model utility.

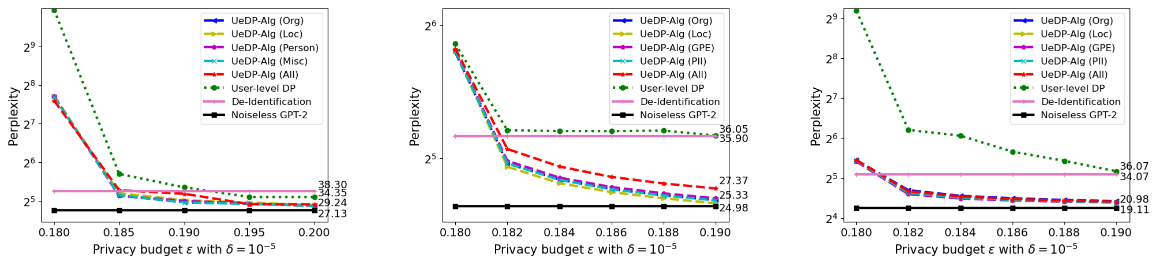
UeDP Guarantee. Given the bounded sensitivity of the estimator, the moments accountant $\mathcal{M}$ [4] is used to get a tight bound on the total UeDP privacy consumption of T steps of the Gaussian mechanism with the noise $\mathcal{N}(0, I\sigma^2)$ (Line 15).

Theorem 1. *For the estimator $f_{\mathcal{E}^+}$, the moments accountant of the sampled Gaussian mechanism correctly computes UeDP privacy loss with the scale $z = \sigma/\mathbb{S}(f_{\mathcal{E}^+})$ for T training steps.*

Proof. At each step, users, detected sensitive entities in E, and extended sensitive entities in S are selected randomly with probabilities q_u, q_e, and q_s, respectively. For $f_{\mathcal{E}^+}$, if the l_2-norm of each user's gradient update, using the sampled sensitive entities in $E_u^t \cup S_u^t$, is bounded by $\mathbb{S}(f_{\mathcal{E}^+})$, then the moments accountant can be bounded by that of the sampled Gaussian mechanism with sensitivity 1, the scale $z = \sigma/\mathbb{S}(f_{\mathcal{E}^+})$, and sampling probabilities q_u, q_e, and q_s. Thus, we can apply the composability as in Theorem 2.1 [4] to correctly compute the UeDP privacy loss with the scale $z = \sigma/\mathbb{S}(f_{\mathcal{E}^+})$ for T steps. $\square$

2.4 Experimental Results

We conducted extensive experiments, both in theory and on benchmark datasets, to shed light on understanding 1) the integrity of sensitive entity identification, 2) the interplay among the UeDP privacy budget (ϵ, δ), different types of sensitive entities (i.e., organization, location, and miscellaneous entities), and model utility, and 3) whether considering the extended set of sensitive entities S will improve model utility under the same UeDP protection.



((**a**)) CONLL-2003 dataset ((**b**)) AG dataset ((**c**)) SEC dataset

Figure 2.2 Next word prediction results using the GPT-2 model.

Baseline Approaches We evaluate our **UeDP-Alg** in comparison with both noiseless and privacy-preserving mechanisms (either user level or entity level), including: **(1) User-level DP** [128], which is the state-of-the-art DP-preserving model closely related to our work; **(2) De-Identification** [46], which is considered as a strong baseline to protect privacy for sensitive entities. Although sensitive entities are masked to hide them in the training process, De-Identification does not offer formal privacy protection to either the data owners or sensitive entities; and **(3)** A **Noiseless** model, which is a language model trained without any privacy-preserving mechanisms. In addition, we consider the naive approach, which is a variation of our algorithm, called **UeDP-Alg** $f_{\mathcal{E}}$. As a baseline, the estimator $f_{\mathcal{E}}$ is computed without taking the extended set of sensitive entities S into account (Appendix A.2). This is further used to comprehensively evaluate our proposed approach. In our experiment, our algorithms and baselines, i.e., UeDP-Alg, User-level DP, and De-Identification, are applied on the noiseless model in the training process. As in the literature review [15, 120], there are no other appropriate DP-preserving baselines for UeDP protection.

Evaluation Tasks and Metrics Our experiment considers two tasks: **(1)** next word prediction and **(2)** text classification. For the next word prediction, we employ the widely used perplexity [133]. The smaller perplexity is, the better model is. For the text classification, we use the test error rate as in earlier work [87]. Test error rate implies prediction error on a test set, so it is 1 - the test set's accuracy. The lower the test error rate is, the better model is.

Data and Model Configuration For the reproducibility sake, all details about our datasets and data processing are included in Appendix A.3. We carried out our experiment on three textual datasets, including the CONLL-2003 news dataset [173], AG's corpus of news articles[2], and our collected Security and Exchange Commision (SEC) financial contract dataset. The data breakdown is in Table 6.1.

For the next word prediction, we employ a GPT-2 model [160], which is one of the state-of-art models. To make the work easily reproducible, we use a version of the pretrained GPT-2 that has 12-layer, 768-hidden, 12-heads, 117M parameters, and then fine-tune with the aforementioned datasets as our Noiseless GPT-2 model. For the text classification, we fine-tune a Noiseless BERT (i.e., BERT-Base-Uncased[3]) pre-trained model [48] that has 12-layer, 768-hidden, 12-heads, and 110M parameters with an additional softmax function on top of the BERT model. Adam optimizer is used with the learning rate is 10^{-5}. Gradient clipping bound $\beta = 0.1$ and the scale $z = 2.5$. The sampling rates for users, detected sensitive entities, and extended sensitive entities q_u, q_e, and q_s are 0.05, 0.5, and 1.0.

To test the effectiveness and adaptability of our mechanism across models, we also conducted experiments with an AWD-LSTM model [130], which has a much fewer parameters compared with GPT-2 and BERT. In AWD-LSTM model, we use a three-layer LSTM model with $1,150$ units in the hidden layer and an embedding input layer of size 100. Embedding weights are uniformly initialized in the interval $[-0.1, 0.1]$ with dimension $d = 100$ and other weights are initialized between $[-\frac{1}{\sqrt{H}}, \frac{1}{\sqrt{H}}]$, where H is the size of all hidden layers. The values used for dropout on the embedding layer, the LSTM

[2]Retrieved on 01/01/2020 from `http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`
[3]Retrieved on 01/01/2020 from `https://huggingface.co/transformers/pretrained_models.html`

hidden-to-hidden matrix, and the final LSTM layer's output are 0.1, 0.3, and 0.5, respectively. Gradient clipping bound $\beta = 0.1$ and the scale $z = 2$. The sampling rates q_u, q_e, and q_s are 0.05, 0.5, and 1.0 (note that q_s is 0.6 in the text classification task). SGD optimizer is used. In the text classification with the AG dataset, a softmax layer is applied on top of the AWD-LSTM with the output dimension is 4, corresponding to four classes in the AG dataset. The same sets of sensitive entity categories are used for all models in the next word prediction and the text classification tasks.

Evaluation Results We conducted the following experiments: **(1)** examining how the sensitive entities detected by the entity recognition spaCy [85] covers the sensitive information clarified by AMT workers, **(2)** comparing estimators $f_{\mathcal{E}}$, $f_{\mathcal{E}^+}$, and User-level DP; **(3)** investigating the interplay between privacy budget and model utility; **(4)** studying the impacts of different sensitive entity categories in E on the privacy budget and model utility; and **(5)** confirming our results in the text classification task.

Our result is as follows:

• **Integrity of sensitive entities.** Our work utilizes spaCy [85], one of the state-of-the-art large-scale entity recognition systems, to identify sensitive entities for evaluation purposes on datasets that do not have ground-truth sensitive entities, including the AG and SEC datasets. For CONLL-2003, we consider the labels of four sensitive entity types (i.e., location, person, organization, and miscellaneous) from NER as the ground truth. To evaluate the integrity of identified sensitive entities, we conducted a clarification on AMT. We found that the results from spaCy cover over 94% of sensitive information as identified by AMT workers. We recruited master-level AMT workers for a high quality of results, and we provided detailed guidance before AMT workers conducted the task. Each sentence was assigned to 3 workers to mitigate bias and subjective views. Consequently, our experiments using the spaCy identified sensitive entities are solid.

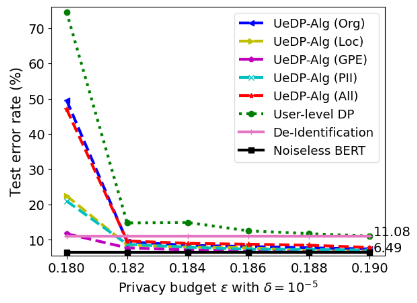
• **Comparing Estimators $f_{\mathcal{E}}$, $f_{\mathcal{E}^+}$, and User-level DP.** We set $q_u = 0.05$, $q_e = 0.5$, $q_s = 1$, $z = 2$, and compute privacy budget ϵ at $\delta = 10^{-5}$ (a typical value of δ in DP) as a function of the training steps T. Figure 2.4 shows curves of using different estimators and the User-level DP with all entities in CONLL-2003, AG, and SEC datasets.

Our UeDP-Alg with $f_{\mathcal{E}^+}$ achieves a notably tighter privacy budget compared with $f_{\mathcal{E}}$ and the User-level DP in all scenarios in CONLL-2003, AG, and SEC datasets. The key reason is that typically detected sensitive entities in E appear rarely in a dataset compared with extended sensitive entities. Thus, using only sensitive entities in E identified by the spaCy in training will cause information distortion, which can damage model utility and a loose privacy budget.

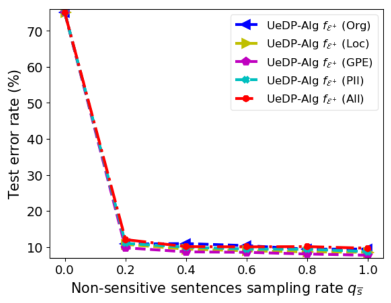
User-level DP consumes a much higher privacy budget ϵ compared with both of our estimators $f_{\mathcal{E}^+}$ and $f_{\mathcal{E}}$. For instance, at $T = 50$, the values of ϵ in all entities of $f_{\mathcal{E}^+}$ and $f_{\mathcal{E}}$, and the value of ϵ of the User-level DP in: **(1)** the CONLL-2003 dataset are 0.52, 0.62, and 1.18; **(2)** the AG dataset are 0.50, 0.75, and 1.48; and **(3)** the SEC dataset are 0.40, 0.71, and 1.40, respectively.

Significantly, the privacy budget (ϵ) gap between User-level DP, $f_{\mathcal{E}}$, and $f_{\mathcal{E}^+}$ is proportionally increased to the number of steps T. That means the more training steps T, the larger ϵ our model can save compared with User-level DP. That is a promising result in the context that our model provides DP protection for both users and sensitive entities, compared with only protection for users in User-level DP. We observe a similar phenomenon on different sensitive categories.

• **Privacy Budget (ϵ, δ)-UeDP and Model Utility.** From our theoretical analysis, $f_{\mathcal{E}^+}$ is better than the estimator $f_{\mathcal{E}}$. Therefore, for the sake of simplicity, we only consider UeDP-Alg $f_{\mathcal{E}^+}$ instead of showing results from both estimators. From now, UeDP-Alg is used to indicate the use of our estimator $f_{\mathcal{E}^+}$. Figure 2.3 illustrates the perplexity as a function of the privacy budget ϵ for an GPT-2 model trained on a variety of sensitive entity categories in UeDP, User-level DP, and De-Identification. The noiseless GPT-2 (for



((**a**)) AG dataset



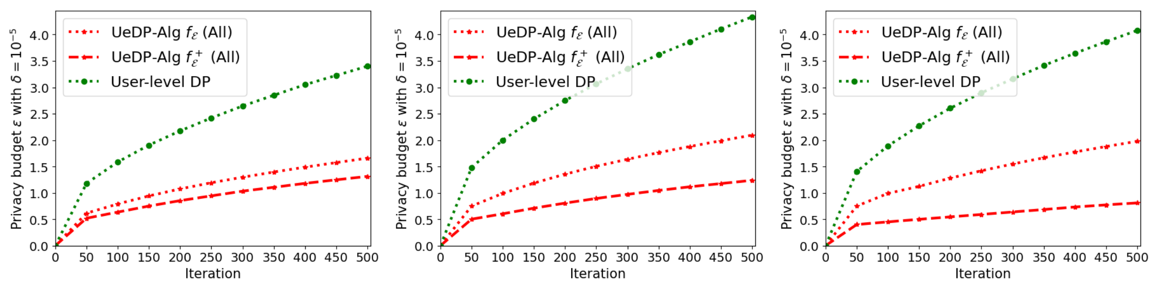
((**b**)) SEC dataset

Figure 2.3 Text classification results on the AG dataset using the BERT model.

the next word prediction) and BERT (for the text classification) models are considered an upper-bound performance mechanism without offering any privacy protection.

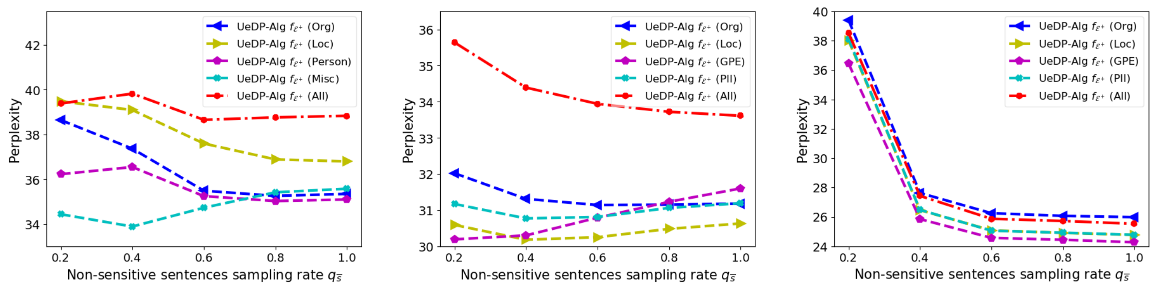
In the CONLL-2003 dataset (Figure 2.2(a)), there are NER labels for person, location, organization, and miscellaneous entities; therefore, we choose these types as sensitive entity categories to protect in UeDP-Alg. UeDP-Alg achieves a better perplexity compared with User-level DP under a tight privacy budget $\epsilon \in [0.18, 0.20]$. Also, from $\epsilon = 0.185$ (a tight privacy protection), our UeDP-Alg achieves a better perplexity than De-Identification. In fact, at $\epsilon = 0.185$, our UeDP-Alg achieves 35.09 for person, 35.34 for organization, 35.57 for miscellaneous, and 36.79 for location entities, compared with 52.01 in User-level DP. When spending more privacy budget ($\epsilon \geq 0.195$), both UeDP-Alg and User-level DP converge at a very competitive perplexity level, approaching the Noiseless GPT-2. For instance, at $\epsilon = 0.20$, there are significant perplexity drops given UeDP-Alg and User-level DP mechanisms, i.e., our UeDP-Alg is 29.24 for person, 29.35 for miscellaneous, 29.58 for organization, and 29.75 for location entities. Meanwhile, the perplexity values of User-level DP, De-Identification, and the noiseless GPT-2 model are 30.15, 38.30, and 27.13.

Results on AG and SEC datasets (Figures 2.2(b) and 2.2(c)) further strengthen our observations. In AG and SEC datasets, we applied spaCy to identify different sensitive entity categories, such as GPE, location, organization, and PII (i.e., person and location information). UeDP-Alg achieves better results compared with User-level



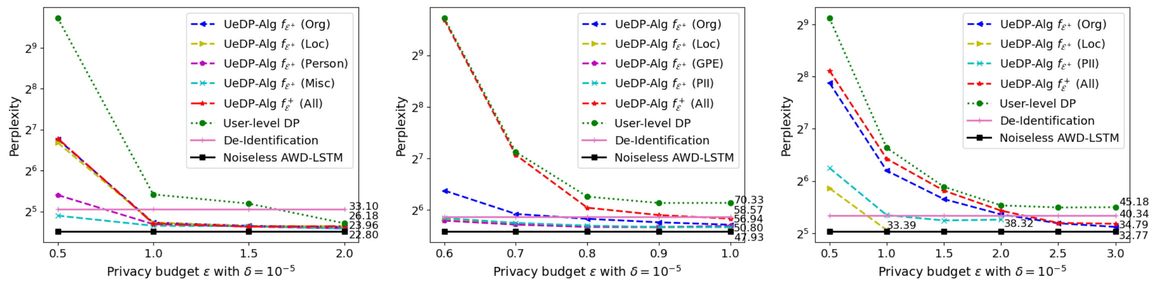
((**a**)) CONLL-2003-all entities ((**b**)) AG-all entities ((**c**)) SEC-all entities

Figure 2.4 Privacy budget of UeDP-Alg $f_{\mathcal{E}}$, UeDP-Alg $f_{\mathcal{E}^+}$, and User-level DP as a function of iterations in CONLL-2003, AG, and SEC datasets.



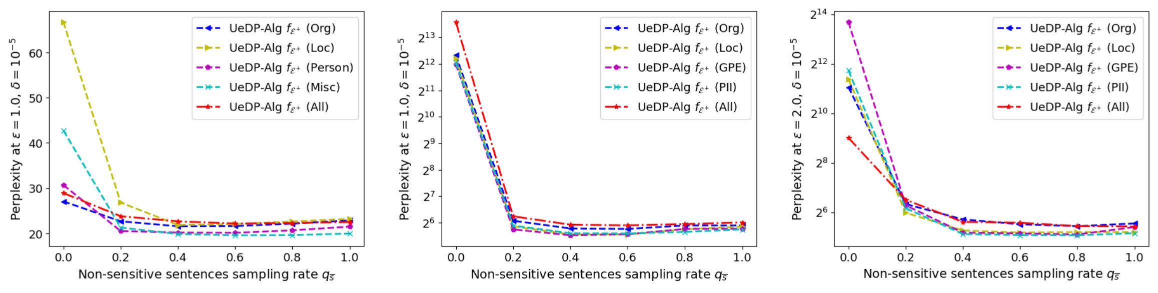
((**a**)) CONLL-2003 dataset ((**b**)) AG dataset ((**c**)) SEC dataset

Figure 2.5 Next word prediction results using the GPT-2 model with varying extended sensitive entities sampling rate q_s in training.



((**a**)) CONLL-2003 dataset ((**b**)) AG dataset ((**c**)) SEC dataset

Figure 2.6 Next word prediction results with AWD-LSTM model.



((**a**)) CONLL-2003 ((**b**)) AG ((**c**)) SEC

Figure 2.7 Next word prediction results using the AWD-LSTM model with varying extended sensitive entities sampling rate q_s in training.

DP in all considering sensitive entity categories and privacy budgets, and outperforms De-Identification in most cases. That is promising and consistent with our previous analysis. For instance, in the AG dataset, at $\epsilon = 0.19$, our UeDP-Alg achieves 25.33 for location, 25.72 for PII, 25.77 for organization, and 26.01 for GPE entities, compared with 36.05 in User-level DP. De-Identification obtains 35.90, and the upper bound result in the noiseless GPT-2 model is 24.98. Similarly, in the SEC dataset (Figure 2.2(c)), at $\epsilon = 0.19$, UeDP-Alg achieves perplexity of 20.98 in GPE, 21.12 in PII, 21.22 in location, 21.50 in organization, and 21.33 in all entities, compared with 36.07 in User-level DP, and 34.07 in De-Identification. In AG and SEC datasets, at a tight privacy budget, i.e., $\epsilon = 0.19$, our UeDP-Alg has better perplexity values than the De-Identification, approaching the noiseless GPT-2 model.

• **Sensitive Entity Categories.** In all datasets (Figures 2.3 and A.2, Appendix A.5, Supplementary[4]), *the more sensitive sentences to protect, the higher the privacy budget is needed, and the lower performance the model achieves* (i.e., higher perplexity values). For instance, in the SEC dataset, the number of sensitive sentences in each category is as follows: 60 in GPE, 273 in location, 357 in PII, $1,955$ in organization, and $2,166$ in all entities. After 500 steps, the values of ϵ are 0.19 in GPE, 0.24 in location, 0.26 in PII, 0.73 in organization, 0.81 in all entities, and 4.08 in User-level DP (Figure 2.4). At $\epsilon = 0.18$ (Figure 2.2(c)), we obtain perplexity values of 42.63 in GPE, 43.21 in location, 43.30 in PII, 43.70 in organization, 43.77 in all entities, and a 583.06 in User-level DP.

• **Text classification.** Figure 2.3(a) shows that our UeDP-Alg achieves lower test error rates in terms of text classification on the AG dataset than baseline approaches in most cases across different types of sensitive entities under a very tight UeDP protection ($\epsilon \in [0.18, 0.19]$). This is a promising result. When ϵ is higher, the test error rates of both UeDP-Alg and User-level DP drop, approaching the noiseless BERT model's result.

• **Extended Sensitive Entities.** To shed light into the impact the extended sensitive entity sampling rate q_s on model utility under UeDP protection, we varied the value of q_s from 0 to 1 in all datasets and tasks. Figures 2.3(b), 2.5, and 2.7 show that considering extended sensitive entities (i.e., $q_s > 0$) significantly improves model utility (i.e., perplexity or test error rate) compared with only considering sensitive entities $e \in E$ (i.e., $q_s = 0$). Technically, different tasks on different datasets may have different optimal values of q_s. This opens a new research question on how to theoretically approximate the optimal value of q_s.

Results on the AWD-LSTM model (Figures 2.6 and 2.7) further strengthen our observations. In our experiments, the AWD-LSTM model generally obtains comparable results with the GPT-2 model for next word prediction at a higher privacy budget range (i.e., $\epsilon \in [0.5, 3.0]$ in the AWD-LSTM model compared with $\epsilon \in [0.18, 0.2]$ in the GPT-2 model). This is because the GPT-2 model is pretrained on large-scale datasets, so that it is easily adapted to the idiosyncrasies of a target task (i.e., next word prediction) compared with the AWD-LSTM model trained from scratch.

2.5 Discussion

We developed a novel notion of user-entity DP (UeDP), protecting users' participation information and sensitive entities in NLMs. By incorporating user and sensitive entity sampling in the training process, we addressed the trade-off between model utility and privacy loss with a tight bound of model sensitivity. Theoretical analysis and rigorous experiments show that UeDP-Alg outperforms baselines in next word prediction and text classification under UeDP protection.

In practice, the list of sensitive entities and users can grow over time. Periodically updating the list of users and sensitive entities may incur extra privacy and computational cost. Therefore, we will focus on preserving UeDP given a growing list of users and sensitive entities in our future work.

Note that UeDP offers the privacy protection in the setting of centralized training, which requires to have a trusted coordinating server. We will present in the next chapter our proposed privacy-preserving mechanism for federated learning and there is no need of a trusted coordinating server.

Table 2.1 Description of Sensitive Entity Categories

Type	Description
Person	Person, i.e., people, including fictional
Loc	Location, i.e., non-GPE locations, mountain ranges, bodies of water
Org	Organization, i.e., companies, agencies, institutions, etc.
Misc	Miscellaneous, i.e., entities that do not belong to the person, location, and organization in CONLL-2003
GPE	Geopolitical entity, i.e., countries, cities, states
PII	Personal identification information, i.e., name, location, phone, etc.
Date	Absolute or relative dates or periods
NoRP	Nationalities or religious or political groups
Fac	Buildings, airports, highways, bridges, etc.
Product	Objects, vehicles, foods, etc. (Not services.)
Event	Named hurricanes, battles, wars, sports events, etc.
Law	Named documents made into laws
Language	Any named language
Work of art	Titles of books, songs, etc.
Time	Times smaller than a day
Percent	Percentage, including “%”
Money	Monetary values, including unit
Quantity	Measurements, as of weight or distance
Ordinal	“First”, “second”, etc.
Cardinal	Numerals that do not fall under another type

```
1: Input: Dataset $D$, set of sensitive entities $E$, extended set of sensitive entities $S$, sampling rates
   $q_u$, $q_e$, and $q_s$, clipping bound $\beta$, a hyper-parameter $z$, and number of iterations $T$
2: Initialize model $\theta^0$ and moments accountant $\mathcal{M}$
3: $w_u \leftarrow \min(\frac{n_u}{\hat{w}_u}, 1)$ for all users $u$ and $w_e \leftarrow \min(\frac{n_e}{\hat{w}_e}, 1)$ for all sensitive entities in $E$
4: $w_s \leftarrow \min(\frac{n_s}{\hat{w}_s}, 1)$ for all extended sensitive entities in $S$
5: where $n_u$, $n_e$, and $n_s$ are the number of sentences in user $u$, the number of sentences containing
   sensitive entities $e \in E$, the number of sensitive entities in $S$, and $\hat{w}_u$, $\hat{w}_e$, and $\hat{w}_s$ are per-user
   sentence cap, per-entity sentence cap, and per-entity entity cap.
6: $W_u \leftarrow \sum_{u \in U} w_u$, $W_e \leftarrow \sum_{e \in E} w_e$, $W_s \leftarrow \sum_{s \in S} w_s$
7: for $t \in T$ do
8:     $U^t \leftarrow$ sample users with probability $q_u$
9:     for each user $u \in U^t$ do
10:        $E_u^t \cup S_u^t \leftarrow$ sensitive entities (belonging to the user $u$) consisting of sensitive entities $E^t$
   sampled from $E$ with probability $q_e$ and extended sensitive entities $S^t$ sampled from $S$ with
   probability $q_s$
11:        $\Delta_{u,\mathcal{E}}^{t+1} \leftarrow$ Local-Update$(u, E_u^t \cup S_u^t, \theta^t, \text{ClipFn})$
12:    end for
13:    $\Delta^{t+1} \leftarrow \frac{\sum_{u \in U^t} w_u \Delta_{u,\mathcal{E}}^{t+1}}{q_u W_u (q_e W_e + q_s W_s)}$
14:    $\sigma \leftarrow \frac{z(q_u|U|+1)\max(w_u)\beta}{q_u W_u (q_e W_e + q_s W_s)}$
15:    $\theta^{t+1} \leftarrow \theta^t + \Delta^{t+1} + \mathcal{N}(0, I\sigma^2)$
16:    $\mathcal{M}$.accum_priv_spending(z)
17: end for
```

```
18: print $\mathcal{M}$.get_priv_spent()
19: **Output:** $(\epsilon, \delta)$-UeDP $\theta$, $\mathcal{M}$
20: **Local-Update**($u, E_u^t \cup S_u^t, \theta^0$, ClipFn):
21: $\theta \leftarrow \theta^0$
22: $\mathcal{B} \leftarrow u$'s data split into size $B$ batches
23: **for** batch $b \in \mathcal{B}$ **do**
24:     $\forall e \in E_u^t : \Delta_{u,e} \leftarrow \sum_{\textit{sentence } s\ (\in b)\ \textit{consists of } e} \bigtriangledown l(\theta, s)$
25:     $\forall s \in S_u^t \cap b : \Delta_{u,s} \leftarrow \bigtriangledown l(\theta, s)$
26:     $\Delta_{u,\mathcal{E}} \leftarrow \sum_{e \in E_u^t} w_e \Delta_{u,e} + \sum_{s \in S_u^t} w_s \Delta_{u,s}$
27:     $\theta \leftarrow \theta^0 - \eta \Delta_{u,\mathcal{E}}$
28: **end for**
29: return ClipFn($\theta - \theta^0$, $\beta$)
30: **ClipFn**$(\Delta, \beta)$: return $\pi(\Delta, \beta) \leftarrow \Delta \cdot \min\left(1, \frac{\beta}{\|\Delta\|}\right)$
```

Algorithm 1: UeDP-Alg

Table 2.2 Breakdown of CONLL-2003, AG, and SEC Datasets

Dataset	$\lvert\mathcal{V}\rvert$	**# of sentences**	**# of users**	**# of sensitive sentences**				
CONLL-2003	8,882	14,040	946	Org	Loc	Person	Misc	All
				5,187	5,433	4,406	3,438	11,176
AG	30,000	112,000	7,536	Org	Loc	GPE	PII	All
				58,177	39,988	18,506	42,683	67,157
SEC	12,651	5,188	1,592	Org	Loc	GPE	PII	All
				1,955	273	60	357	2,166

CHAPTER 3
SCALABLERR: DIMENSION-SCALABLE LOCAL DIFFERENTIAL PRIVACY FOR FEDERATED LEARNING

3.1 Preamble

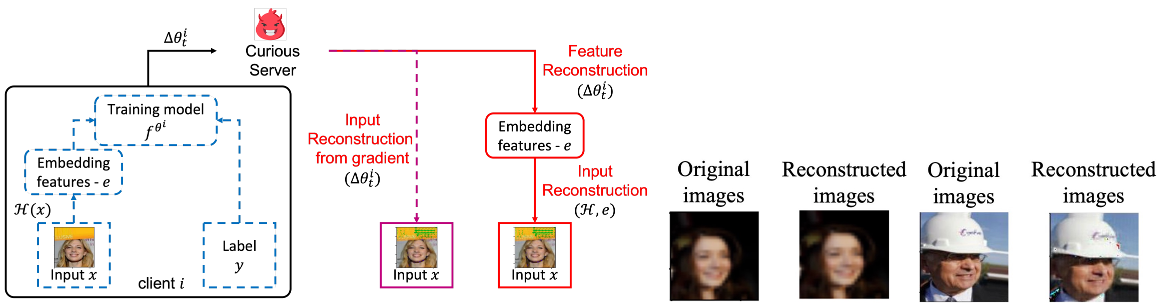
Local differential privacy (LDP) is one of the most viable options to protect the privacy of training data in federated learning systems. Briefly, existing LDP-preserving mechanisms usually suffer from the curse of dimensionality. In other words, increasing the number of features causes an increase in privacy budget or randomization probabilities, which in turn leads to either loose privacy protection or poor utility.

In this chapter, we mitigate the impact of increasing the dimension of the feature vector on the trade-off between privacy and data utility by designing Scalable Randomized Response (ScalableRR), a novel dimension-scalable and data utility-aware randomized response technique. ScalableRR integrates dimensionality, data utility, and privacy guarantees in a unified framework that is jointly optimized for better randomization probabilities. The key idea that enables ScalableRR to maintain a good privacy-utility balance for high-dimensional feature input is the use of variable randomization probabilities at the binary level representation of every feature as opposed to the use of uniformly distributed randomization probabilities.

Through extensive theoretical analysis and a wide set of experiments using federated learning, especially a trial in a real-world mobile-cloud federated learning system for human activity recognition, we show that ScalableRR can train deep neural networks with large numbers of features in an unlimited number of training rounds under modest privacy budgets while retaining high utility.

3.2 Background

This section briefly reviews FL, privacy threat models, existing defenses, and the curse of dimensionality.



((**a**)) Privacy threat from the curious server in FL ((**b**)) From the gradients ((**c**)) From the embeddings

Figure 3.1 a) Privacy threat from the curious server in FL, b) Data reconstruction attacks from gradients, and c) from embeddings.

Federated Learning. FL is a multi-round communication protocol between ***a coordination server*** and ***a set of*** N ***clients*** to jointly train a ML model f, e.g., a neural network. At round t, the server sends the latest model weights θ_t to a randomly sampled subset of clients S_t. Upon receiving θ_t, client u in S_t uses θ_t to train its local model, and generates model weights θ_t^u. Client u computes its local gradient $\triangle\theta_t^u = \theta_t^u - \theta_t$, and sends it back to the server. After receiving all the local gradients from all the clients in S_t, the server updates the model weights by aggregating all the received local gradients using an aggregation function $\mathcal{G} : R^{|S_t|\times n} \rightarrow R^n$ where n is the size of $\triangle\theta_t$. The aggregated gradient is added to θ_t:

$$\theta_{t+1} = \theta_t + \lambda\mathcal{G}(\{\triangle\theta_t^i\}_{i\in S_t}) \tag{3.1}$$

where λ is the server's learning rate. A typical aggregation function $\mathcal{G}$ is the (federated) weighted averaging, dubbed FedAvg, which is widely applied in FL [97].

By joining the FL protocol, the clients try to minimize the average of their loss functions (i.e., client u does not know losses from other clients) as follows: $\arg\min_\theta \frac{1}{N}\sum_{u=1}^N L_u(\theta)$. L_u is the loss function of a client u on a local training dataset D_u penalizing the mismatch between the predicted values $f(x,\theta)$ and the ground-truth label y, as follows: $L_u(\theta) = \frac{1}{|D_u|}\sum_{x\in D_u} L\big(f(x,\theta), y\big)$, where $|D_u|$ denotes the number of data samples in the local training dataset D_u.

Privacy Threat Model. Although having many applications, recent attacks [43, 49, 223, 227] underscore privacy risks in FL by showing that the server can precisely reconstruct the original sample x from the shared local gradients. That imposes a severe privacy risk if the data sample x is sensitive, such as medical images, health records, etc.

The server is assumed to follow the training procedure in the FL protocol but is curious about the clients' training data D_u. This is a practical threat model since service providers always aim at providing the best services to the clients [78, 195]. Clients compute their local gradients by using one of two approaches: **(1)** Directly using the data sample x; and **(2)** Using embedding features[1] extracted from x given a pre-trained model. In both cases, clients need to send their local gradients $\triangle\theta_t^u$ to the server for federated training.

In the former case, the server can apply gradient-based attacks [223, 227] to precisely reconstruct the original data sample x from the observed local gradients (Figure 3.1). In the latter case, the server can infer the embedding features from the local gradients by applying gradient-based attacks [223, 227]. Then, the server feeds the inferred embedding features to a reconstruction attack [43, 49, 143] to accurately reconstruct the original data sample x. Both approaches are efficient in reconstructing x (Figure 3.1b,c).

LDP in FL. An effective way to protect clients' local training data is to preserve LDP in FL [191, 196]. LDP-preserving mechanisms [8, 19, 52, 63, 202] generally build on the ideas of RR [204], which was initially introduced to allow survey respondents to provide their inputs while maintaining their confidentiality. The definition of ϵ-LDP is as follows:

Definition 4. *ϵ-LDP. A randomized algorithm $\mathcal{M}$ fulfills ϵ-LDP, if for any two inputs x and x', and for all possible outputs $\mathcal{O} \in Range(\mathcal{M})$, we have: $Pr[\mathcal{M}(x) = \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{M}(x') = \mathcal{O}]$, where ϵ is a privacy budget and $Range(\mathcal{M})$ denotes every possible output of $\mathcal{M}$.*

The privacy budget ϵ controls the amount by which the distributions induced by inputs x and x' may differ. A smaller ϵ enforces a stronger privacy guarantee. Existing

[1]Embedding features are typically used to represent a data sample in ML since it is necessary to reduce input dimensions while enriching the information captured from inputs.

LDP-preserving approaches preserve privacy in FL by either randomizing the local data before the training process [13,121,210] or by randomizing the local gradients before sending them to the server [116, 201, 226]. In this paper, we tailor ScalableRR to FL, motivated by the explosive growth and significant impacts of FL in real-world applications [11, 97, 111].

Fundamentally, these prior LDP-privacy preserving approaches do not address the trade-off between dimensionality, privacy, and data utility. In this work, we close this gap by developing a dimension-scalable RR mechanism, the first of its kind, that integrates dimensionality, data utility, and ϵ-LDP protection into a unified framework with optimized randomization probabilities.

3.3 Dimension-Scalability

To understand the curse of dimensionality and how we measure the magnitude of degradation in the randomization probabilities and privacy protection, we consider an essential and widely-used feature extraction setting in ML. That is: given a data sample x, we use a mechanism $\mathcal{H}$ to extract a vector of numerical features $e \in \mathbb{R}^r$ from x, denoted as $e = \mathcal{H}(x)$, where r is the number of features in e.

The extracted features $e = \mathcal{H}(x)$ can be used in different ways in practice and offer great benefits (e.g., dimension reduction, model utility improvement, being learnable and reusable [10, 31, 77, 98, 164, 169, 188, 221]). Technically, directly using the feature vector e without appropriate privacy protection can incur privacy risk to the original data sample x as illustrated in Figure 3.1 [43, 223, 227].

To ensure that using the feature vector e does not expose the original data sample x against the privacy threat model (Figure 3.1), one can repeatedly apply an existing LDP-preserving mechanism, e.g., [201], to randomize every feature a in the feature vector e. According to the strong privacy composition Theorem [59], if randomizing a single feature a consumes a privacy budget ϵ_a, randomizing the whole feature vector e consumes a total privacy budget $\epsilon = \sum_a \epsilon_a$ ($= r \times \epsilon_a$ when using a uniform privacy budget for the r features).

The Curse of Dimensionality. The curse of dimensionality occurs when we increase the number of features r in the feature vector e by an expansion factor d, where $d \in \mathbb{R}^+$, $d > 1$, and the new number of features $d \times r$ must be an integer. The increase could be, for example, to enrich the extracted embedding features or enlarge the size of local gradients derived from the data sample x for better data and model utility. Given the randomization probabilities of every feature a and the total privacy budget ϵ as the most critical factors affecting data utility and privacy protection, we seek a quantitative answer to the following question: *What is the impact of increasing the dimension of the feature vector on the total privacy budget ϵ and the randomization probabilities of every feature a?*

There are three possible answers for this question: **(1)** keep the total privacy budget ϵ the same and degrade the randomization probabilities of every feature a. This makes the features noisier proportional to the value of d and degrades model utility; **(2)** keep the randomization probabilities of every feature a the same and increase the total privacy budget from ϵ to $d \times \epsilon$ (guided by the strong privacy composition Theorem [59] in existing RR mechanisms). This leads to a proportional reduction in privacy guarantees; and **(3)** finding new randomization probabilities and a new privacy budget adapted to the value of d. The first two cases are directly applicable to most existing LDP-preserving mechanisms [51, 52, 201]. Meanwhile, the approaches proposed in [13, 121] follows the third direction.

Dimension-Scalability Measure. In all the cases, we need to quantify the *changes* in the new randomization probabilities and the new total privacy budget. Such quantification, dubbed the measure of dimension-scalability, informs how much an LDP-preserving mechanism is affected when the dimension of the feature vector e is increased by an expansion factor d. Intuitively, *smaller changes* indicate a *better dimension-scalability* of an LDP-preserving mechanism since its randomization probabilities and total privacy budget are *less affected* when the dimension of the vector e increases. Conversely, *more significant changes* indicate a *worse dimension-scalability* of an LDP-preserving mechanism.

To capture the dimension-scalability given the worst impact of increasing the dimension of the feature vector e by the expansion factor d, we quantify the change in the randomization probabilities, denoted γ, when the total privacy budget ϵ in the new feature vector is kept the same (i.e., fixed). Similarly, we quantify the change in the total privacy budget, denoted γ_ϵ, when the randomization probabilities of every feature a in the new feature vector are kept the same (i.e., fixed).

General Form of an RR Mechanism. We consider a general form of an ϵ-LDP-preserving RR mechanism $\mathcal{M}$ to formulate $(\gamma, \gamma_\epsilon)$-dimension-scalability for all existing ones. A mechanism $\mathcal{M}$ randomizes every feature a in the feature vector e creating a randomized feature vector e', as follows:

$$\mathcal{M}(e) : e \rightarrow e' \in \mathbb{R}^r \tag{3.2}$$

The key concept is to chop the output range $Range(\mathcal{M})$ of a mechanism $\mathcal{M}$ into a set of sub-ranges. Each sub-range R_j is associated with a probability $Pr(a_j|a, \cdot)$ to decide whether we report a random value a_j in the range R_j given the original feature a. For generalizability, the sub-ranges can be discrete (e.g., in Duchi's or Three-outputs mechanisms [51, 201, 226]) or continuous (e.g., in Piecewise mechanism [201]) and the values inside a subset R_j can be either uniformly or non-uniformly distributed depending on an RR mechanism.

For each feature a, $\mathcal{M}$ reports a randomized value $a' \in Range(\mathcal{M})$ with the randomization probabilities:

$$\forall a \in e : a' = \{a_j \in R_j \text{ with a probability } Pr(a_j|a, e, \epsilon)\}$$
$$\text{with } \sum_j Pr(a_j|a, e, \epsilon) = 1 \tag{3.3}$$

where the set of output sub-ranges $\{R_j\}$ is disjoint (i.e., $\cap_j R_j = \emptyset$) and the union of all the sub-ranges fully cover the output range of the mechanism (i.e., $\cup_j R_j = Range(\mathcal{M})$), and a_j is a (random) value in the output sub-range R_j.

By doing that, the general form can be directly and equivalently mapped to existing RR mechanisms and vice-versa. The output range $Range(\mathcal{M})$, the sub-ranges $\{R_j\}$, and their associated randomization probabilities in Equation 3.3 can be instantiated for different mechanisms.

For example, in Piecewise mechanism [201], the output range is $Range(\mathcal{M}) = [-C, C]$ which can be divided into two sub-ranges $R_1 = [-C, L) \cup (R, C]$ and $R_2 = [L, R]$, where $C = \frac{\exp(\epsilon/2)+1}{\exp(\epsilon/2)-1}$, $L = \frac{C+1}{2}a - \frac{C-1}{2}$, and $R = L + C - 1$. In Duchi's mechanism [51,201], the output is finite with $R_1 = -\frac{\exp(\epsilon)+1}{\exp(\epsilon)-1}C_r$ and $R_2 = +\frac{\exp(\epsilon)+1}{\exp(\epsilon)-1}C_r$, where $C_r = \frac{2^{r-1}}{\binom{(r-1)}{(r-1)/2}}$ if r is odd and $C_r = \frac{2^{r-1}+\frac{1}{2}\binom{r}{r/2}}{\binom{(r-1)}{r/2}}$ if r is even. The probability that the output falls inside the range R_1 is $\frac{\exp(\epsilon/2)}{1+\exp(\epsilon/2)}$ and in another range R_2 is $\frac{1}{1+\exp(\epsilon/2)}$. Interested readers can refer to our extended analysis[2] for details regarding all existing RR mechanisms.

Dimension-Scalability Definition. Our definition of $(\gamma, \gamma_\epsilon)$-dimension-scalability is as follows. We increase the dimension of the feature vector e by d to create a new feature vector e_d and compute its randomized version e'_d by applying the mechanism $\mathcal{M}$, i.e., $\mathcal{M}(e_d) : e_d \rightarrow e'_d \in \mathbb{R}^{d \times r}$ with a total privacy budget ϵ_d. We then apply different distance measures, denoted $\mathcal{D}$, such as Kullback–Leibler (KL) divergence and L_p-norms, to measure the changes in the randomization probabilities $\{Pr(a_j|a, \cdot)\}$ given the original feature vector e and the new one e_d.

[2]Section A.13, `https://www.dropbox.com/s/ythoyiq243lb24g/ExtendAL.pdf?dl=0`

Definition 5. *(γ, γ_ϵ)-Dimension-scalability. A randomized mechanism $\mathcal{M}(e) : e \rightarrow e' \in \mathbb{R}^r$ is (γ, γ_ϵ)-dimension-scalable if when the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d ($d \times r \in \mathbb{N}^+$ and $d > 1$), we have:*

***(1)** Given a fixed level of LDP protection, i.e., $\epsilon = \epsilon_d$, the change γ in the randomization probabilities is quantified as follows:*

$$\gamma = \sum_j \mathcal{D}\Big(Pr(R_j|a, e, \epsilon) \| Pr(R_j|a, e_d, \epsilon_d)\Big) \tag{3.4}$$

***(2)** Given fixed randomization probabilities, i.e., $\gamma = 0$ or $\forall j : Pr(a_j|a, e, \epsilon) = Pr(a_j|a, e_d, \epsilon_d)$, the change γ_ϵ in the total privacy budget is quantified as follows:*

$$\gamma_\epsilon = \epsilon_d - \epsilon \tag{3.5}$$

Considering the aforementioned distance measures in Equation 3.4: 1) If $\mathcal{D}$ is KL-$divergence$, we have $\mathcal{D}(\cdot) = \int_{a_j \in R_j} Pr(a_j|a, e, \epsilon) \log(\frac{Pr(a_j|a,e,\epsilon)}{Pr(a_j|a,e_d,\epsilon_d)}) da_j$; and 2) If $\mathcal{D}$ is L_2-norm, we have $\mathcal{D}(\cdot) = [\int_{a_j \in R_j} (Pr(a_j|a, e, \epsilon) - Pr(a_j|a, e, \epsilon_d)) da_j]^{\frac{1}{2}}$.

By measuring the dimension-scalability, we provide a guideline for end-users about the flexibility of LDP-preserving RR mechanisms to decide the best mechanism to handle their privacy and data utility trade-offs when the dimensionality increases. A detailed theoretical analysis of dimension-scalability of existing LDP-preserving RR mechanisms in FL, such as the Duchi's mechanism (DM) [51,52], Piecewise mechanism (PM) [201], Hybrid mechanism (HM) [201], Three-outputs [226], OME[3] [121] and LATENT[4] [13] is available for interested readers[4].

In these mechanisms, if we fix the privacy budget (i.e., $\epsilon = \epsilon_d$ or $\gamma_\epsilon = 0$), the randomization probabilities for each feature are notably affected leading to large values of γ under KL-$divergence$ and L_2-norm measures (Figure 3.4) and, consequently, data utility

[3]When we adapt OME, which reports randomized binary vectors as its final outputs, to our setting by using extra information on how the binary vector is created, it causes extra privacy risk. To address the problem, we apply Newton-Raphson method [22] to find new values of the temperature α that can match the new privacy budget, which is forced to be bounded by a user-predefined privacy budget.

[4]There exists no temperature α in LATENT to address the privacy budget exploding. Therefore, we do not consider LATENT in our experiments.

degradation (Figure 3.5). On the other hand, if we use the same randomization probabilities $Pr(a_j|a, \cdot)$ (i.e., $\gamma = 0$), the privacy budget is accumulated over the expanded vector e_d. That is, the privacy budget increases by the factor d ($\epsilon_d = d \times \epsilon$ and $\gamma_\epsilon = (d-1)\epsilon$). This causes privacy risk exaggeration when d is large.

Having either small γ or small γ_ϵ is sufficient to mitigate the curse of dimensionality. Accordingly, there are two potential approaches to achieve this goal. The first is to find a tighter privacy loss bound to overcome the strong composition Theorem. That would help us to achieve small changes γ_ϵ in the total privacy budget. An example is the well-known moments accountant [4], an instance of adaptive composition, or numerical composition [76]. Unfortunately, the moments accountant and the numerical composition are not directly applicable to RR mechanisms.

An alternative and feasible approach is to keep the total privacy budget ϵ fixed and focus on finding new randomization probabilities with small changes γ to lessen the impact of the expansion factor d. Our mechanism follows this approach to mitigate the curse of dimensionality by integrating dimension-scalability, data utility, and privacy protection into a unified framework to find optimal randomization probabilities. This fundamental difference from existing works helps ScalableRR significantly improve the trade-off among privacy, utility, and dimensionality.

3.4 Scalable Randomized Response

To develop a dimension-scalable RR mechanism, one can simply inject a considerable amount of noise into the randomization probabilities such that the noise countermeasures the impact of dimension expansion. Empirically, such treatment degenerates data utility and could even completely damage it. Alternatively, one can significantly reduce the noise so that the dimension increase does not affect the randomization probabilities. In this way, the privacy protection provided to the clients' local data degenerates. Therefore, providing

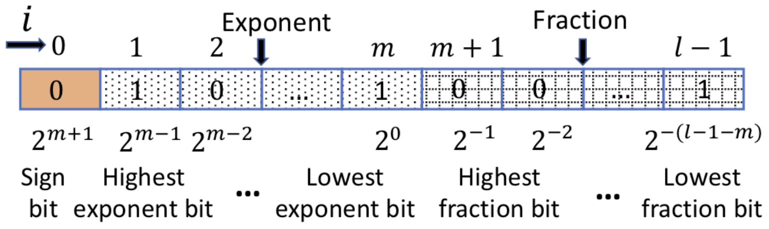


Figure 3.2 Binary representation.

a better understanding of the connection between dimension-scalability, data utility, and privacy protection, will enable us to achieve better data utility and privacy trade-offs.

To shed light on this connection, we consider a basic representation of a numerical feature a in the feature vector e, its binary representation. By doing that, we can assess the data utility through the influences of different binary bits on the value of the feature a. We represent the feature a using an l-binary bit vector v consisting of 1 sign bit, m exponent bits, and $l-m-1$ fraction bits (Figure 3.2) as

$$\forall i \in [0, l-1] : v_i = \lfloor 2^{i-m}|a| \rfloor \mod 2 \tag{3.6}$$

where $v_i \in \{0, 1\}$ is the binary value of v at the bit i. For instance, IEEE 754 standard binary32 includes 32 bits ($l = 32$) with 1 sign bit, 8 exponent bits ($m = 8$), and 23 fraction bits. In this study, we observe that with only 10 bits ($l = 10$), 1 sign bit, 5 exponent bits ($m = 5$), and 4 fraction bits, we can achieve nearly lossless model utility. Therefore, we use these numbers of bits as constants in the rest of the paper. Technically, our theoretical analysis shows that one can use larger numbers of bits without affecting the privacy-utility-dimensionality trade-offs as long as the value domain of the feature a is sufficiently covered.

Lossless Representation. Given a real-value of a, an l-binary bit vector may not be sufficient to represent a, that is, decoding the binary vector v introduces a tiny error: $a = \mathcal{E}(v) + \varphi$, where φ is a tiny error and $\mathcal{E}(\cdot)$ is the decoding function associated with Equation 3.6. We dismiss the tiny error φ by replacing a with $\mathcal{E}(v)$, i.e., $a = \mathcal{E}(v)$. This can be done as a data preprocessing step without incurring any privacy risk or notable utility loss.

Overview of ScalableRR. The pseudo-code of our mechanism is in Alg. 1. ScalableRR randomizes every bit in the binary vector representation v to create an LDP-preserving binary vector v', which will be decoded, based on the binary encoding in Equation 3.6, to generate a randomized feature a'. We apply ScalableRR to independently randomize every feature a in the feature vector e to create an LDP-preserving feature vector e', which is the final outcome of ScalableRR. One can use the LDP-preserving feature vector e' as a permanent replacement of the training data sample x in downstream tasks, such as computing the local gradients in FL. That enables us to significantly reduce the model size (i.e., r embedding features) compared with using the randomized binary vector representation v' (i.e., $r \times l$ binary bits) in training the model f.

The entire process of ScalableRR takes an input $e = \mathcal{H}(x)$ and outputs $e' = \mathcal{E}\big(\mathcal{M}(v)\big) \in \mathbb{R}^r$ is formulated as follows:

$$ScalableRR(x) : e = \mathcal{H}(x) \rightarrow e' = \mathcal{E}\big(\mathcal{M}(v)\big) \in \mathbb{R}^r \tag{3.7}$$

where $\mathcal{M}$ is a mechanism we use to randomize every bit in the binary vector v derived from the feature vector e.

By working at the binary bit level, we can integrate dimensionality, data utility, and privacy protection to optimize randomization probabilities. We introduce a ***bit-aware term*** for better data utility and a ***control parameter*** for better control of the randomization probabilities. The bit-aware term ensures that *more influential bits* on the data utility, such as sign bits and exponent bits, have *better (less noisy) randomization probabilities,* and vice-versa. In addition, when the dimension r of the feature vector e increases, one can always find a suitable control parameter such that *the changes in the randomization probabilities and in the total privacy budget are marginal.* As a result, ScalableRR achieves better dimension-scalability and data utility under the same privacy protection.

A Simple RR Mechanism. To provide insights into key components in ScalableRR, let us start with simple probabilities [204] to randomize every bit i in the binary vector v creating a randomized binary vector v', as follows:

$$\forall i \in [0, l-1] : \begin{cases} p_i \leftarrow P(v'_i = v_i) = \dfrac{\exp(\epsilon_X)}{1 + \exp(\epsilon_X)} \\ q_i \leftarrow P(v'_i = \overline{v}_i) = \dfrac{1}{1 + \exp(\epsilon_X)} \end{cases} \tag{3.8}$$

where $\overline{v}_i$ is the complement of v_i, ϵ_X is a user-predefined privacy budget, and $A \leftarrow B$ denotes B is assigned to A. In this work, we use p_i and q_i as the randomization probabilities, in which p_i is the probability to keep the bit i and q_i is the probability to flip the bit i.

The total privacy budget to randomize the binary vector v of the feature a is $\epsilon = l \times \epsilon_X$. If we apply these randomization probabilities on every feature in the feature vector e, the total privacy budget will be linear to the dimension r of the feature vector e: $\epsilon = rl \times \epsilon_X$ (the strong privacy composition Theorem [59]). It is obvious that this simple RR suffers from the curse of dimensionality for two reasons: **(1)** Every bit has the same randomization probabilities (the flat line in Figure 3.3(a)); and **(2)** Every bit randomization mandates the same privacy budget ϵ_X.

These issues motivate the following question: *Is there any different distribution of the randomization probabilities across the bits in v that could result in the same or less total privacy budget?*

To answer this question, we look into the probability q_i, which essentially is a sigmoid function of ϵ_X. To control the sigmoid function, one can use a scale factor s and a temperature α, as follows:

$$q_i = \frac{s}{1 + \exp(-\alpha\epsilon_X)} \tag{3.9}$$

Since all the probabilities are bounded, i.e., $q_i, p_i \in (0, 1)$, the scale factor s must be fixed to 1 to maximize the search space for all the probabilities irrespective of the dimensionality increase (decrease) of v or the feature vector e. Therefore, we focus on finding the temperature α as the control parameter of the randomization probabilities such that the randomization is bit-aware and dimension-scalable for better privacy-utility trade-offs. That is a challenging problem since the temperature α can have different forms.

3.4.1 Bit-aware Randomization

We tackle the problem of finding the appropriate temperature step by step. In this section, we focus on integrating the bit-aware term into the temperature α. We first introduce the concept of bit-influence.

Bit-Influence. Under the same LDP guarantee, the smaller the difference between a and its randomized version a', the more information is preserved through the randomization process; that is, less noise and better data utility. Hence, we need to understand the impact of randomizing the binary vector v on the LDP-preserving feature a' by bounding the amount of information that can be changed via randomizing different bits in v in the worse case (i.e., all bits flip).

In fact, each bit i has a different influence associated with its location in v. For example, flipping the sign bit in v, e.g., $v = \mathbf{1}010.01010\mathbf{1}$, to create a randomized vector $v' = \mathbf{0}010.010101$, results in the randomized feature $a' = -2.328125$ (i.e., derived from decoding v'), which is significantly different from the original feature $a = 2.328125$. Meanwhile, flipping the least significant fraction bit in v to create $v' = 1010.01010\mathbf{0}$ results in a much smaller distance between a and its randomized version a', i.e., $a' = 2.3125 \simeq a$. Flipping different bits introduces different data utility and privacy risks. The closer the randomized version a' to its original value a, the better the data utility and the greater the privacy risk; and vice-versa.

To address this trade-off, we quantify the influence of each bit i in the binary vector v by capturing the magnitude by which bit i changes the LDP-preserving feature a' in the worse case. The influence of a bit i is bounded as follows:

Lemma 2. *Bit-Influence. Consider the two binary vectors v and $v^{\neq i}$ such that $v^{\neq i}$ differs from v only at bit i, the l_1-norm influence $\mathcal{I}_i$ is computed as follows:*

$$\forall i \in [0, l-1] : \mathcal{I}_i = \max_{v} \|\mathcal{E}(v) - \mathcal{E}(v^{\neq i})\|_1 \tag{3.10}$$

$\forall i \in [0, l-1]$, $\mathcal{I}_i$ is bounded as follows:

$$\mathcal{I}_i \leq \begin{cases} 2^{m+1}, & \text{if } i \text{ is a sign bit} \\ 2^{m-i}, & \text{if } i \text{ is an exponent or fraction bit} \end{cases} \tag{3.11}$$

Proof of Lemma 2 is Appendix A.6.

Given the bit-influence in Lemma 2, we introduce a bit-aware term as part of the temperature control such that ***more influential bits*** on the model utility (e.g., sign and exponent bits) have ***smaller probabilities (q_i)*** to be flipped. In other words, more influential bits on data utility receive less perturbation and vice-versa. Since the bounded bit-influences are associated with the locations of the bits, we can formulate the bit-aware term to reflect the influential location of bit i in the binary vector v. That results in different randomization probabilities for different bits, as follows:

$$\forall i \in [0, l-1] : \begin{cases} p_i \leftarrow P(v'_i = v_i) = \dfrac{1}{1 + \alpha \exp\big(i \times \epsilon_X\big)} \\ q_i \leftarrow P(v'_i = \overline{v}_i) = \dfrac{\alpha \exp\big(i \times \epsilon_X\big)}{1 + \alpha \exp\big(i \times \epsilon_X\big)} \end{cases} \tag{3.12}$$

In Equation 3.12, we move α outside of the $\exp(\cdot)$ to better control the probability of the sign bit ($i = 0$) without affecting the generality of the sigmoid function. We can always find an α such that $q_i \leq p_i$ satisfying LDP (i.e., similar to typical RR definition [204]).

Figure 3.3(a) depicts the distributions of the randomization probabilities (q_i) across the bits of v for different values of the temperature α in Equation 3.12. Given fixed values of ϵ_X and $\alpha \in \mathbb{R}^+$, Figure 3.3(a) clearly shows that the probability (q_i) to flip more influential bits is smaller, e.g., $i = 0$ for the sign bit resulting in $q_0 = \frac{\alpha}{1+\alpha}$, compared with less influential bits, e.g., $i = l - 1$ for the least important fraction bit resulting in $q_{l-1} = \frac{\alpha \exp\big((l-1)\epsilon_X\big)}{1+\alpha \exp\big((l-1)\epsilon_X\big)}$. In fact, $q_0 < q_{l-1}$. The analysis is true for all the bits in v, i.e., $\forall i, j \in [0, l-1], i < j : q_i < q_j$. Such ***variable randomization probabilities*** enable us to ***achieve better data utility*** compared with the uniform randomization probability (q_i in Equation 3.8).

ScalableRR. If we apply our bit-aware randomization probabilities on every bit in the binary vector v, which has $r \times l$ bits created by concatenating the binary representations of all features in the feature vector e, Equation 3.12 becomes:

Randomization mechanism $\mathcal{M}$:

$$\forall i \in [0, rl-1] : \begin{cases} p_i \leftarrow P(v'_i = v_i) = \dfrac{1}{1 + \alpha \exp(\frac{i\%l}{l}\epsilon_X)} \\ q_i \leftarrow P(v'_i = \overline{v}_i) = \dfrac{\alpha \exp(\frac{i\%l}{l}\epsilon_X)}{1 + \alpha \exp(\frac{i\%l}{l}\epsilon_X)} \end{cases} \tag{3.13}$$

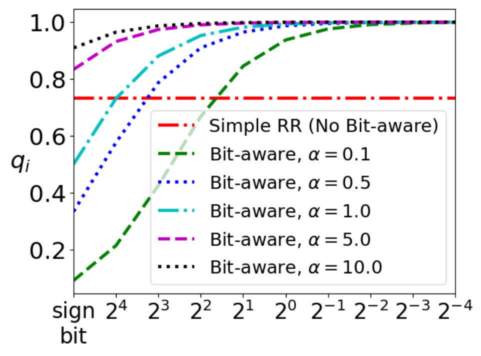
where the bit-aware term becomes $\frac{il}{l}$ (or $\frac{i\%l}{l}$ in short) for $\forall i \in [0, rl-1]$ to reflect the influential location of the bit i of each feature a in v.

After obtaining the randomized binary vector $v' = \mathcal{M}(v)$, as in Equation 3.7, ScalableRR decodes v' to create the LDP-preserving feature vector $e' = \mathcal{E}(v')$. ScalableRR focuses on optimizing α such that ϵ_X becomes the total privacy budget as in Theorems 2 and 3.

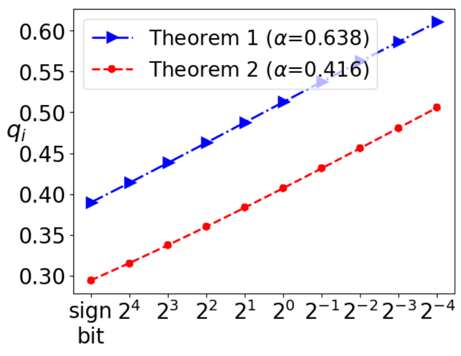
3.4.2 Privacy Loss Bound

The main challenge is to obtain good data utility while bounding the privacy loss to maintain acceptable level of protection. Therefore, ScalableRR searches the value of α that produces the distribution of randomization probabilities with the best LDP guarantee.

Intuitively, this α can be computed from Equation 3.13 under the constraint that the total privacy loss in randomizing the binary vector v is bounded by ϵ_X as in previous works [13, 121], such that $\frac{P(\mathcal{M}(v)=v_z)}{P(\mathcal{M}(\tilde{v})=v_z)} \leq \exp(\epsilon_X)$ where two vectors v and $\tilde{v}$ that could be different in any number of their bits and $v_z \in Range(\mathcal{M})$. Importantly, we need to train



((a)) q_i (ϵ_X is NOT the total privacy budget)



((b)) q_i (ϵ_X becomes the total privacy budget as in ScalableRR, Theorems 2-3)

Figure 3.3 Randomization probability q_i as a function of α with $\epsilon_X = 1.0$ and $r = 10,000$.

a significantly larger model f taking the randomized binary vector $v' = \mathcal{M}(v)$ (i.e., $r \times l$ bits) as an input. This approach is not scalable when the number of embedding features r increases by the factor d.

To address this problem, as in Equation 3.7, ScalableRR decodes the randomized binary vector to extract randomized embedding features $e' = \mathcal{E}\big(\mathcal{M}(v)\big)$. One can use the randomized embedding features e' to train their local model f. By working with the randomized embedding features e', our mechanism scales better by reducing the model size. The challenge is to bound the privacy loss given the final outcome of ScalableRR, i.e., the randomized embedding features: $\frac{P(\mathcal{E}(\mathcal{M}(v))=\mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v}))=\mathcal{E}(v_z))}$. This is because the decoding function uses extra information about the bits, that is, different bits have different influences on the randomized embedding features.

In this section, we derive a new and tighter privacy loss bound of ScalableRR. The key idea is to ensure that ***the privacy loss in randomizing every bit*** in the binary vector v is ***upper-bounded*** by ***the privacy loss incurred given the randomized embedding features***.

First, we quantify the sensitivity of ScalableRR. By following [59], the maximum amount of change in the output of ScalableRR caused by changing bit i in the input vector v is considered as the l_1-*sensitivity* of ScalableRR at the bit i. Therefore, the l_1-*sensitivity* is quantified as follows:

$$\forall i \in [0, rl-1] : \Delta_i = \max_v |\mathcal{E}\big(\mathcal{M}(v,i)\big) - \mathcal{E}\big(\mathcal{M}(v^{\neq i}, i)\big)|$$

where $\mathcal{M}(v, i)$ implies that $\mathcal{M}$ is applied to randomize only bit i while keeping all other bits in v unchanged.

Given a deterministic decoding function $\mathcal{E}$, upon bounding the l_1-sensitivity Δ_i in Lemma 4 (Appendix A.7), for the LDP condition to hold in randomizing a single bit i, the ratio is bounded as follows:

$$\frac{P(\mathcal{E}(\mathcal{M}(v,i)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(v^{\neq i},i)) = \mathcal{E}(v_z))} = \frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \times \prod_{j\neq i, j\in[0,rl-1]} \frac{P(\mathcal{M}(v_j) = v_{z,j})}{P(\mathcal{M}(v_j^{\neq i}) = v_{z,j})} \tag{3.14}$$
$$= \frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \leq \exp(\epsilon_i \frac{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}{\Delta_i})$$

where ϵ_i is the privacy loss for the bit i. For all the bits j different from the bit i, $P(\mathcal{M}(v_j) = v_{z,j}) = P(\mathcal{M}(v_j^{\neq i}) = v_{z,j})$ since all these bits j are the same in the binary vectors v_j and $v_j^{\neq i}$, i.e., $v_j = v_j^{\neq i}$.

We then apply the strong composition Theorem [59] to expand the privacy loss bound of a single bit in Equation 3.14 into the total loss over the whole binary vector v, as follows:

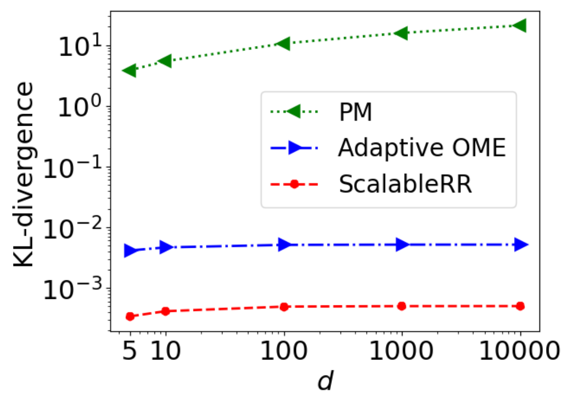
$$\frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} = \prod_{i=0}^{rl-1} \frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \tag{3.15}$$
$$\leq \exp\big(\sum_{i=0}^{rl-1} \frac{\epsilon_i |\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}{\Delta_i}\big) \leq \exp(\sum_{i=0}^{rl-1} \epsilon_i)$$

If we force the total privacy loss $\epsilon = \sum_{i=0}^{rl-1} \epsilon_i$ to be equal to the user-predefined budget ϵ_X, we can identify a closed form solution of the temperature α by solving the inequality:

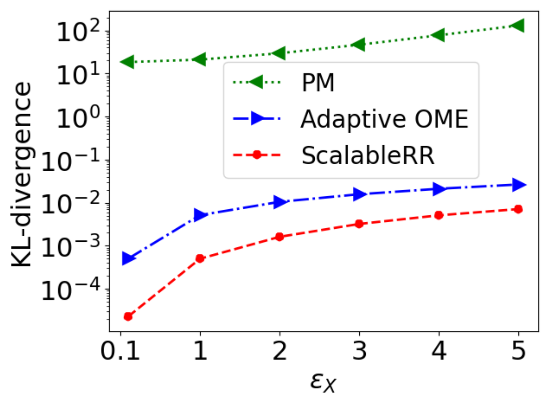
$$\frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} = \prod_{i=0}^{rl-1} \frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \leq \exp(\epsilon_X) \tag{3.16}$$

Theorem 2. *Given $\alpha = \exp(\frac{\epsilon_X - r\epsilon_X(l-1)}{2rl})$, ScalableRR preserves ϵ_X-LDP by satisfying Equation 3.16, where v and $\tilde{v}$ can be different in any number of bits, and $v_z \in Range(\mathcal{M})$.*

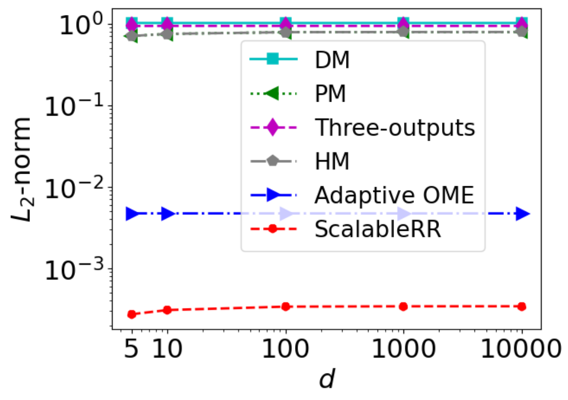
The proof of Theorem 2 is in Appendix A.8.



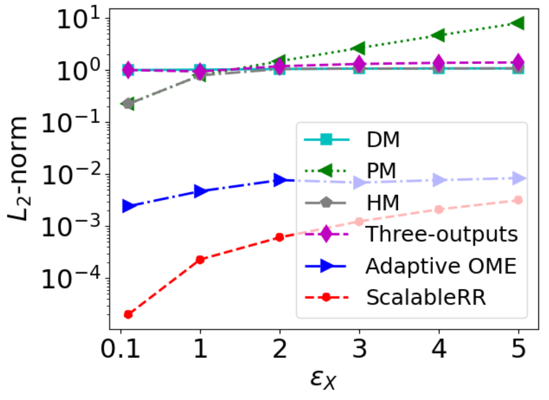
((a)) KL-divergence ($r = 10$, $\epsilon_X = 1$)



((b)) KL-divergence (r=10, d=10,000)



((c)) L_2-norm ($r = 10$, $\epsilon_X = 1$)



((d)) L_2-norm ($r = 10$, $d = 10,000$)

Figure 3.4 Different measurements for comparing dimension-scalability among LDP-based techniques.

3.4.3 Tighter Privacy Loss Bound

The privacy loss bound in Equation 3.16 is not tight enough since there is a gap between the actual bound, $\exp\left(\sum_{i=0}^{rl-1} \frac{\epsilon_i|\mathcal{E}(\mathcal{M}(v,i))-\mathcal{E}(\mathcal{M}(v^{\neq i},i))|}{\Delta_i}\right)$, and the user-predefined budget, ϵ_X $(=\sum_{i=0}^{rl-1}\epsilon_i)$, since $\exp(\sum_{i=0}^{rl-1}\epsilon_i) - \exp\left(\sum_{i=0}^{rl-1}\frac{\epsilon_i|\mathcal{E}(\mathcal{M}(v,i))-\mathcal{E}(\mathcal{M}(v^{\neq i},i))|}{\Delta_i}\right) \geq 0$. In other words, the current bound does not utilize all the user-predefined privacy budget. That may affect data utility given the solution of α in Theorem 2.

The root cause of the gap is the proportion $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|/\Delta_i \leq 1$. To close this gap, we revisit the privacy loss bound $\frac{P(\mathcal{M}(v,i)=v_z)}{P(\mathcal{M}(v^{\neq i},i)=v_z)}$ in Equation 3.14. There are two possible cases: **(1)** $\frac{P(\mathcal{M}(v,i)=v_z)}{P(\mathcal{M}(v^{\neq i},i)=v_z)} \geq 1$, and **(2)** $0 < \frac{P(\mathcal{M}(v,i)=v_z)}{P(\mathcal{M}(v^{\neq i},i)=v_z)} < 1$. Let us consider here the first case, while the second case is covered in Appendix A.9[5].

[5]We obtain the same closed form solution of α for the two cases.

Since Δ_i is the l_1-sensitivity, we have $\Delta_i / |\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i}, i))| \geq 1$. Therefore, for the first case, we have:

$$\frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \leq \Big(\frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}} \leq \exp(\epsilon_i) \tag{3.17}$$

Finding α such that $\Big(\frac{P(\mathcal{M}(v_i)=v_{z,i})}{P(\mathcal{M}(v_i^{\neq i})=v_{z,i})}\Big)^{\frac{\Delta_i}{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}} \leq \exp(\epsilon_i)$ always guarantees that the randomization of the bit i is ϵ_i-LDP, i.e., $\frac{P(\mathcal{M}(v_i)=v_{z,i})}{P(\mathcal{M}(v_i^{\neq i})=v_{z,i})} \leq \exp(\epsilon_i)$. The key advantage is that $\Big(\frac{P(\mathcal{M}(v_i)=v_{z,i})}{P(\mathcal{M}(v_i^{\neq i})=v_{z,i})}\Big)^{\frac{\Delta_i}{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}}$ is closer to $\exp(\epsilon_i)$ than the privacy loss $\frac{P(\mathcal{M}(v_i)=v_{z,i})}{P(\mathcal{M}(v_i^{\neq i})=v_{z,i})}$; hence, it offers a tighter privacy loss bound. As a result, we can derive a better value of the temperature α offering better randomization probabilities and data utility.

We directly extend Equation 3.17 to derive a tighter privacy loss bound covering all the bits in the binary vector v, as follows:

$$\frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} \leq \prod_{i=0}^{rl-1} \Big(\frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}} \leq \exp(\sum_{i=0}^{rl-1} \epsilon_i) \tag{3.18}$$

Similarly, if we force the total privacy budget $\epsilon = \sum_{i=0}^{rl-1} \epsilon_i$ to be equal to a user-predefined budget, we can identify a closed form solution of the temperature α by solving the following inequality:

$$\frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} \leq \prod_{i=0}^{rl-1} \Big(\frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|}} \leq \exp(\epsilon) = \exp(\epsilon_X) \tag{3.19}$$

Theorem 3. *Given* $\alpha = \sqrt{\frac{rl + (1-\rho)\epsilon_X}{2r \sum_{i=0}^{l-1} \exp(2\epsilon_X \frac{i}{l})}}$, *ScalableRR preserves* ϵ_X*-LDP,* v *and* $\tilde{v}$ *can be different in any number of bits, and* $v_z \in Range(\mathcal{M})$, $\rho = 2\sqrt{\frac{-\ln(\delta)}{2r}}$ *with a broken probability* $l \times \delta$.

The proof of Theorem 3 is in Appendix A.9.

A typical value of the broken probability is $l \times \delta = 1e-5$ [4]. As we can observe, the temperature α derived from Theorem 3 is smaller than the one from Theorem 2[6], resulting in a smaller probability q_i for a bit i to be flipped under the same ϵ_X (Figure 3.3(b)). Thus, ScalableRR retains higher data utility outperforming existing RR mechanisms.

3.4.4 Dimension-Scalability

This section sheds light on the trade-off among dimension-scalability, data utility, and privacy guarantees. It is challenging to achieve a good trade-off among dimension-scalability, data utility, and privacy. Injecting a large amount of noise cause poor utility or injecting little noise results in loose privacy. Dimension-scalability is only meaningful when it results in better data utility under the same privacy guarantees.

ScalableRR achieves significantly better (γ, $\cdot$)-dimension-scalability than existing mechanisms by limiting the impact of increasing the dimension of the feature vector e on the randomization probabilities and the total privacy budget through appropriate computation of the temperature α. To elaborate on this, let us consider the closed-form solution of α in Theorem 3, which is the only factor affecting the randomization probabilities q_i and p_i (Equation 3.13). Given a fixed user-predefined privacy budget ϵ_X (i.e., $\gamma_\epsilon = 0$), the term $\sum_{i=0}^{l-1} \exp(2\epsilon_X \frac{i}{l})$ is a constant[7], denoted as $\mathbb{C}$. Hence, we have $\alpha = \sqrt{(2l + \frac{(1-\rho)\epsilon_X}{r})/(\mathbb{C})}$. Given a typical value of r in practice, such as a hundred or more features in the feature vector e, the term $\frac{(1-\rho)\epsilon_X}{r}$ ($\approxeq 0.0$) is tiny compared with the number of bits l and the constant $\mathbb{C}$ since ϵ_X is usually small to provide rigorous privacy protection. Therefore, when the dimension r increases by the factor d, the impact of either the term $\frac{(1-\rho)\epsilon_X}{r}$ or the term $\frac{(1-\rho)\epsilon_X}{d \times r}$ on the temperature α is minimal. As a result, ***the impact of the expansion factor*** d ***on the randomization probabilities is marginal***.

In other words, when the dimension r of the feature vector e is increased, one can always find a new temperature α such that the randomization probabilities are marginally

[6] $\sqrt{(rl + (1-\rho)\epsilon_X)/[2r\sum_{i=0}^{l-1}\exp(2\epsilon_X \frac{i}{l})]} < \exp(\frac{\epsilon_X - r\epsilon_X(l-1)}{2rl})$ ($l\delta = 1e-5$ and $r \geq 1$).

[7] The number of bits l to represent a single numerical feature a is a constant.

affected under the same total privacy budget. That ability is fundamentally different from existing mechanisms. On the other hand, fixing the randomization probabilities ($\gamma = 0$) while expanding r by the factor d results in a total privacy budget of $d \times \epsilon_X$. This is similar to existing LDP-preserving RR mechanisms due to the strong composition Theorem.

From the definition of dimension-scalability, we can compute and bound the impact of the expansion factor d on the randomization probabilities given a fixed user-predefined ϵ_X by employing KL-divergence and L_p-norms to measure the changes in the randomization probabilities. Similarly, we can compute the impact of the factor d on the total privacy budget given fixed randomization probabilities.

Theorem 4. *(γ, γ_ϵ)-dimension-scalability of ScalableRR. When the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d ($d \times r \in \mathbb{N}^+$ and $d > 1$), given a user-predefined privacy budget ϵ_X, ScalableRR achieves (γ, γ_ϵ)-dimension-scalability where γ and γ_ϵ are calculated as follows:*

- $$\gamma = \begin{cases} \sum_{i=0}^{l-1} \log\left(\frac{1+\alpha_d t_i}{1+\alpha t_i}\right) + \sum_{i=0}^{l-1} q_i \log \frac{\alpha}{\alpha_d}, \\ \qquad\qquad \textit{for KL-divergence} \\ \sqrt{\prod_{i=0}^{l-1} A^i + \prod_{i=0}^{l-1} A_d^i - 2\prod_{i=0}^{l-1} B^i}, \textit{ for } L_2\textit{-norm} \end{cases}$$
- $\gamma_\epsilon = (d-1) \times \epsilon_X$

where α and α_d are the temperature parameters as in Theorem 3 for e and its expanded version e_d, respectively; $t_i = \exp(\frac{i\%l}{l}\epsilon_X)$, $A^i = \frac{1+\alpha^2 t_i^2}{(1+\alpha t_i)^2}$, $A_d^i = \frac{1+\alpha_d^2 t_i^2}{(1+\alpha_d t_i)^2}$, and $B^i = \frac{1+\alpha\alpha_d t_i^2}{(1+\alpha t_i)(1+\alpha_d t_i)}$.

The proof of Theorem 4 is in Appendix A.10.

In Figure 3.4, given a rigorous privacy budget, i.e., $\epsilon_X = 1.0$, when the expansion factor d increases (e.g., $d \in [5, 10^4]$), thanks to the bit-aware term and the temperature α derived from a tighter privacy loss bound, ScalableRR achieves the best dimension-scalability with significantly smaller changes γ of the randomization probabilities in KL-

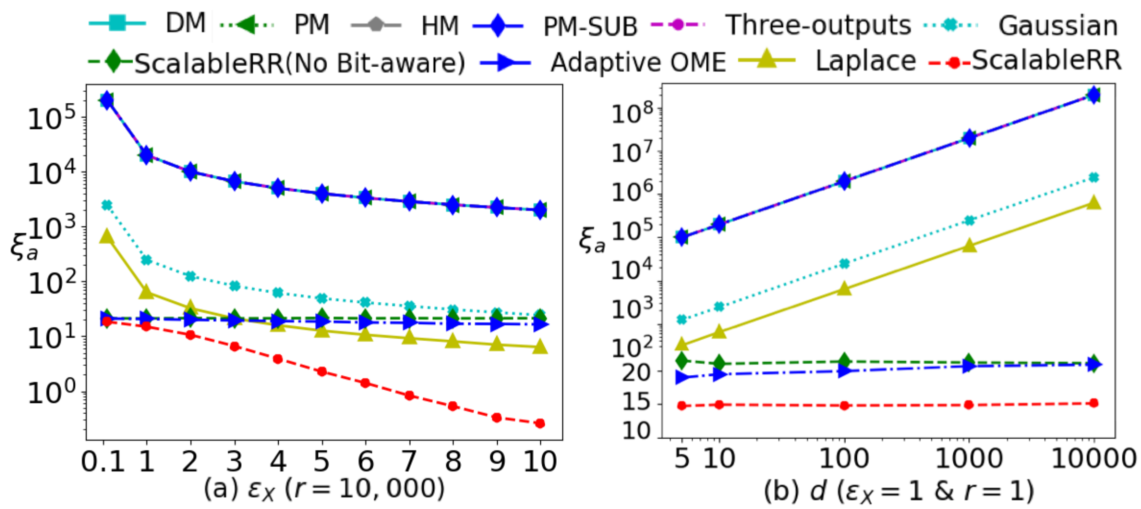


Figure 3.5 Expected error for an embedding feature as a function of: (a) user-predefined budget ϵ_X, (b) expansion factor d.

divergence and L_2-norm than existing approaches. We observe similar results across a wide-range of ϵ_X. Among the baselines, the adaptive OME obtains a much better dimension-scalability; technically, at much higher changes γ compared with ScalableRR, and is followed by PM mechanism and other baseline approaches.

3.4.5 Privacy, Utility, Dimensionality Trade-off

In addition to the bit-aware property and a tighter privacy loss bound, achieving good dimension-scalability enables us to significantly improve utility-privacy trade-off while handling bigger feature vectors e. To provide insights, we analyze the privacy-utility trade-off of ScalableRR, compared with existing approaches. We compute **(1)** the expected error as defined in the following Theorem 5, and **(2)** the randomization probabilities defined in Equation 3.13 as a function of **(i)** the total privacy budget, and **(ii)** the the feature expansion factor d. Note that ***all statistical tests are 2-tail t-tests.***

Expected Error. We use the expected error, denoted as ξ_a, as a proxy for the degradation in data utility. The expected error is measured as the expected change of an embedding feature a represented by a binary vector v after applying ScalableRR: $\xi_a =$

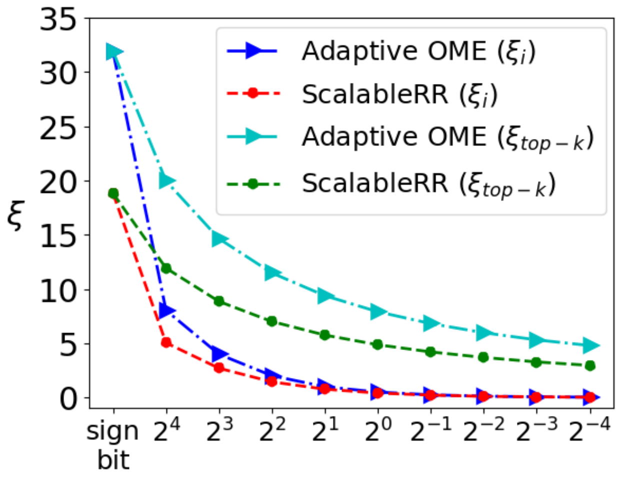


Figure 3.6 Expected error at the bit-level ($\epsilon_X = 1.0$, $r = 10,000$)

$\mathbb{E}|\mathcal{E}(\mathcal{M}(v)) - \mathcal{E}(v)|$. The ***smaller the expected error*** is, the ***better data utility*** a mechanism could achieve.

Theorem 5. *The ScalableRR expected error is quantified by* $\xi_a = \mathbb{E}|\mathcal{E}(\mathcal{M}(v_a)) - \mathcal{E}(v_a)| = \sum_{i\in[0,l-1]} q_i \times \Delta_i$.

The proof of Theorem 5 is in Appendix A.11.

Theorem 5 can be directly applied to quantify the expected error of different variants of ScalableRR, including ScalableRR without the bit-aware property $\frac{i\%l}{l}$ in Equation 3.13, ScalableRR without the tighter privacy loss bound (Theorem 2) and adaptive OME[5]. For existing mechanisms that utilize the numerical feature a, including Duchi's mechanism (DM) [51], Piecewise mechanism (PM) [201], Hybrid mechanism (HM) [201], Three-outputs mechanism [226], Suboptimal mechanism (PM-SUB) [226], Gaussian and Laplace [59], we derive a general form of the expected error ξ_a as: $\xi_a = \mathbb{E}|\mathcal{M}(a) - a| \approxeq 1/r \sum_{a\in e} |\mathcal{M}(a) - a|$, where $\mathcal{M}$ is an LDP preserving mechanism, since $\lim_{r\rightarrow\infty} \mathbb{E}|\mathcal{M}(a) - a| = 1/r \sum_{a\in e} |\mathcal{M}(a) - a|$.

Expected Error v.s. Privacy Budget. Figure 3.5a depicts the expected error as a function of the user-predefined total privacy budget ϵ_X. The figure shows that ScalableRR achieves significantly better-expected error than the baselines under a wide range of privacy budgets, given large numbers of embedding features, e.g., $r = 10,000$, ($p = 0.02$). In other

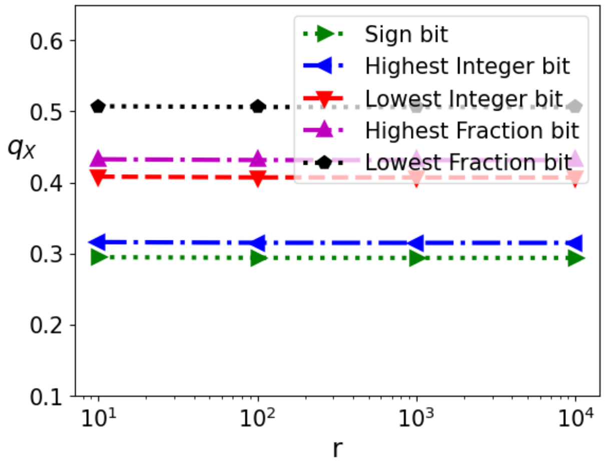


Figure 3.7 Randomization probability q_i as a function of d.

words, ScalableRR retains better data utility compared with the baselines. The improvement of ScalableRR over the baselines is more significant for bigger privacy budgets ϵ_X.

Expected Error v.s. Dimensionality. Figure 3.5b illustrates the expected error of each mechanism as a function of the expansion factor d. Thanks to the dimension-scalability, ScalableRR obtains notably smaller expected error compared with the baselines under a wide range of the factor d. This shows that ScalableRR can offer better data utility compared with the baselines. When d increases, the expected error of the numerical value-based approaches, including DM, HM, Three-outputs, Gaussian, and Laplace mechanisms, grows significantly, thus, negatively affecting data utility.

Expected Error at the Bit-Level. Our analysis of the expected error at the bit-level shows that, in all the 2^l (i.e., $l = 10$) possible values of the binary vector for a feature, ScalableRR achieves better expected error than the adaptive OME, the only baseline working at the bit-level (Figure 3.6). In addition, ScalableRR achieves smaller values of expected error for most significant bits (the sign bit and exponent bits) compared with adaptive OME, and comparable expected error for least significant (fraction) bits.

Randomization Probabilities v.s. Privacy Budget and Dimensionality. The unique bit-aware property of ScalableRR enables us to achieve lower randomization probabilities (q_i) across almost all the bits compared with the adaptive OME under a wide range of

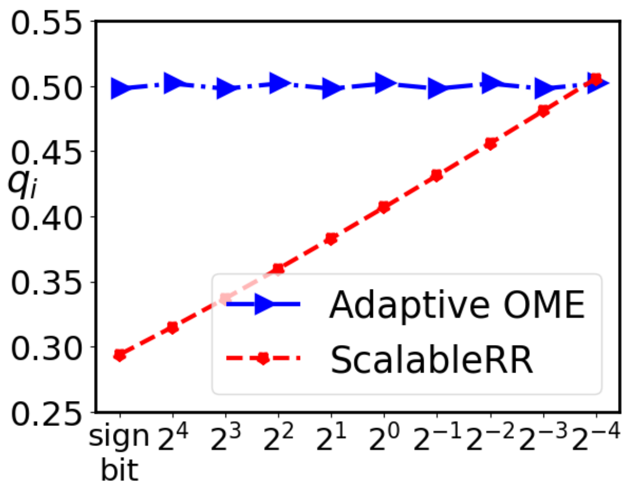


Figure 3.8 Randomization probability q_i at the bit-level ($r = 10,000$, $\epsilon_X = 1$).

user-predefined privacy budgets ($\epsilon_X \in \{0.1, 1, 2\}$) ($\epsilon_X = 1$ in Figure 3.8). Importantly, given ϵ_X in ScalableRR, varying $d \in [5, 10^4]$ marginally affects q_i for all the bits in v (Figure 3.7). Meanwhile, q_i noticeably changes in OME when d increases. In addition, ScalableRR achieves smaller values of q_i, resulting in less perturbations in the significant bits (e.g., the sign bit, the highest and lowest exponent bits, and the highest fraction bit), and hence better utility. Only at the lowest fraction bit, which is the least important bit to the data utility, ScalableRR has comparable values of q_i with OME when d is large. The bit-aware term and the temperature derived from the tighter privacy loss bound allow ScalableRR to achieve significantly better dimension-scalability given γ.

3.5 Experimental Results

In this paper, we evaluate ScalableRR in the FL setting. The pseudo-code is in Alg. 1. For a single input x and its label y in the local training data D_u, the client u extracts embedding features e from the input x using a pre-trained model $\mathcal{H}$ (line 4). The pre-trained model can be trained on public datasets to avoid extra privacy risks. To defend against the privacy threat model stated in Section 3.2: **(1)** We apply ScalableRR to protect the privacy of the embedding features e with ϵ_X-LDP guarantees; and **(2)** We apply Label-DP [73] to achieve ϵ_Y-LDP guarantee of the label y (lines 5-8). In addition to randomizing e, the randomization

Algorithm 1 ScalableRR for clients. The server side is the same with a typical FL protocol (FedAvg) [126].

```
1: Input: Training model $f(\theta)$, pre-trained model $\mathcal{H}$, loss function $L$, privacy budgets $\epsilon_X$ and $\epsilon_Y$, round $t$
2: At client $u \in [1, N]$:
3: for each data sample $(x, y) \in D_u$ do
4:     Extract embedding features: $e = \mathcal{H}(x)$
5:     $v \leftarrow BinaryEncoding(e)$ # using Equation 3.6
6:     Randomize $v$: $v' \leftarrow \mathcal{M}(v, \epsilon_X)$ # using Equation 3.13 with $\alpha$ in Theorem 3
7:     $e' \leftarrow \mathcal{E}(v')$ # decoding function associated with Equation 3.6
8:     Randomize $y$: $y' \leftarrow$ label-DP$(y, \epsilon_Y)$
9: end for
10:     ClientUpdate$(u, \theta_t)$:
11:     $\theta_t^{u*} \leftarrow \arg\min_{\theta_t} \frac{1}{n_u} \sum_{(e',y') \in D'_u} L\big(f(e', \theta_t), y'\big)$
12:     Return $\triangle\theta_t^u \leftarrow \theta_t^{u*} - \theta_t$
```

of the label y allows us to achieve a complete LDP protection of each local training sample (e, y). All the randomized samples (e', y') are used as permanent replacements for the original training samples in computing the local gradients $\triangle\theta_t^u$ at every training round t (lines 9-11). Hence, privacy protection is not affected by the size of the local gradients and the number of communication rounds.

For ***reproducibility***, the implementation of ScalableRR with model hyper-parameter configuration is available[8].

We conduct extensive experiments on benchmark datasets for text and image classification to shed light on **(1)** the interplay between privacy and model utility, **(2)** the effectiveness of the dimension-scalability and bit-aware properties, and **(3)** different settings of applying RR to preserve LDP.

[8] `https://anonymous.4open.science/r/ScalableRRcode/`

Datasets, Metrics, and Models. We carry out our experiments on two textual datasets and two image datasets, including the AG dataset [194], our collected Security and Exchange Commission (SEC) financial contract dataset, the large-scale celebFaces attributes (CelebA) dataset [117], and the Federated Extended MNIST (FEMNIST) dataset [26] (Table 3.1). We use the test accuracy and the test area under the curve (AUC) as evaluation metrics. Higher values of test accuracy and AUC are better. To extract embedding features, i.e., $e = \mathcal{H}(x)$, we use the BERT-Base (Uncased) pre-trained model [48] for the AG and SEC datasets and the ResNet-18 pre-trained model [81] for the CelebA and FEMNIST datasets. For text and image classification tasks, we use two fully connected layers on top of embedding features, each consisting of 1,500 neurons with ReLU activation.

Baselines. We consider a variety of LDP-preserving baselines: **(1)** the **Adaptive OME** [121]; **(2) LDP-FL** [191]; **(3)** Duchi's mechanism (**DM**) [51]; **(4)** Piecewise mechanism (**PM**) [201]; **(5)** Hybrid mechanism (**HM**) [201]; **(6) Three-Outputs** mechanism [226]; **(7)** Suboptimal mechanism (**PM-SUB**) [226]; and **(8) Label-Laplace** [149]. Note that [24] is considered as the newer version of DM; in this way, the privacy budget consumption is large in FL, i.e., $\epsilon_X \geq 100$, indicating a loose privacy protection [140]. Therefore, we do not consider it in our experiments. Each baseline is applied to randomize (when applicable): **(i)** Embedding features e; **(ii)** Gradients $\triangle^u_{\theta_t}$; and **(iii)** Gradients $\triangle^u_{\theta_t}$ with a recent anonymizer [191] to reduce the privacy budget consumption. We include the **Noiseless** FL model trained on the original data D_u to show upper-bounds and a **Random** guess model to better understand the impact on model utility.

These settings are widely used; hence, offering a comprehensive view of LDP preservation in FL. Note that Adaptive OME and ScalableRR are only applied on embedding features; while LDP-FL is only applied on the local gradients with or without the anonymizer.

Table 3.1 Dataset Breakdown

Dataset	Train	Test	# clients	Samples per client	$\mathcal{C}$
	# samples	# samples		(Average)	
AG	$120,000$	$7,600$	$2,800$	43	4
SEC	$5,188$	$1,021$	$1,592$	3	2
CelebA	$155,529$	$19,962$	$6,733$	22	40 (binary)
FEMNIST	$734,033$	$71,230$	$3,550$	207	62

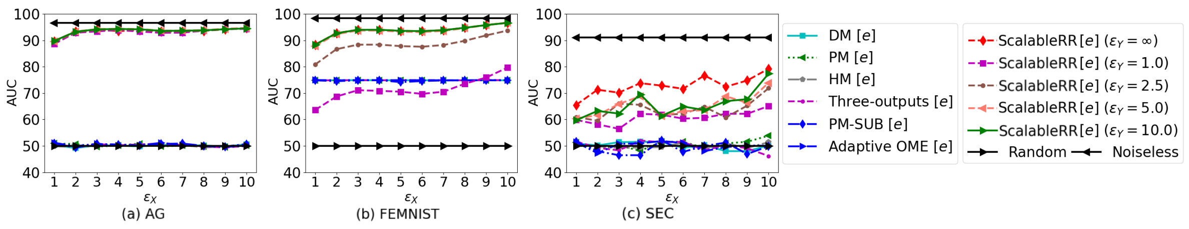
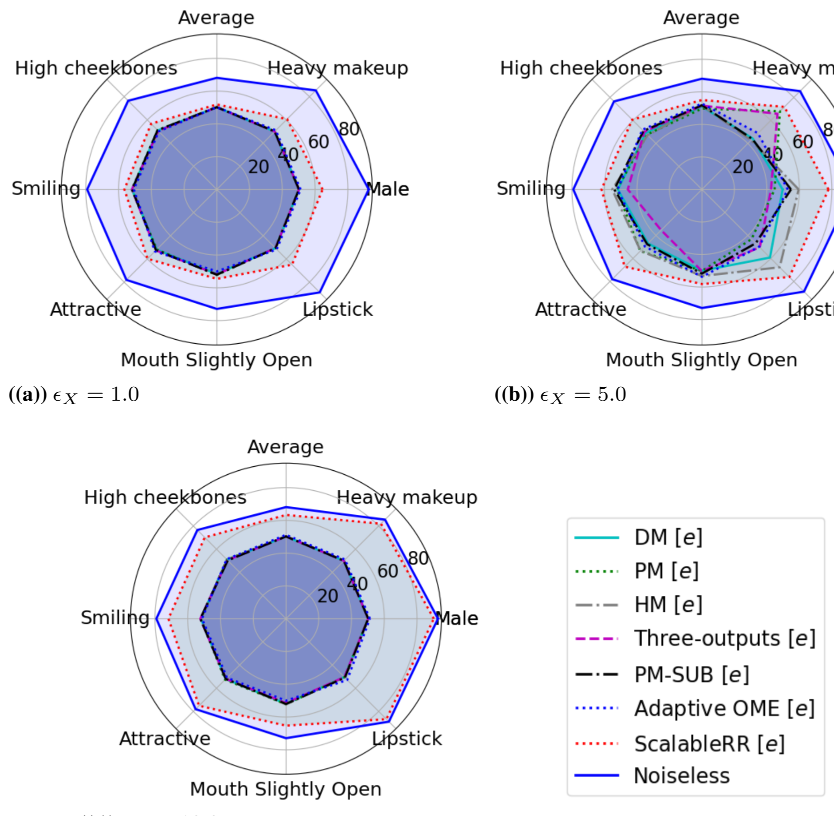


Figure 3.9 AUC values of LDP algorithms applied on the embedding features e in the AG, SEC, and FEMNIST datasets.

Evaluation Results. Without loss of generality, we report AUC results while the accuracy results are available for interested readers[9]. Experimental results show that ScalableRR offers stronger privacy protection with better model utility, compared with baseline approaches, as discussed next.

LDP-preserving Embedding Features. Figure 3.9 shows that baseline approaches do not work well when applied on embedding features. In SEC, AG, and FEMNIST datasets, ScalableRR achieves the highest model utility compared with the best baseline (i.e., adaptive OME) under a tight privacy budget $\epsilon_X = 1$. In terms of accuracy and AUC values, ScalableRR ($\epsilon_Y = \infty$ means no protection for the label) achieves an improvement over the baselines of 46.03% and 38.51% in the AG dataset ($p = 2.7e - 22$), 13.69% and

[9]Section A.18 in our extended analysis.



((a)) $\epsilon_X = 1.0$ ((b)) $\epsilon_X = 5.0$ ((c)) $\epsilon_X = 10.0$

Figure 3.10 AUC values of LDP algorithms applied on the embedding features e in the CelebA dataset.

13.79% in the SEC dataset ($p = 4.1e - 12$), and 21.62% and 13.42% in the FEMNIST dataset ($p = 5.6e - 11$), respectively. In the CelebA dataset (Figure 3.10), ScalableRR outperforms the best baseline (i.e., PM-SUB) with an average improvement of 1.66% across all 40 attributes in terms of AUC measure ($p = 1.2e - 2$). Since the CelebA dataset is highly imbalanced, we use the AUC measure instead of the model accuracy. The gaps between ScalableRR and the baselines are wider when ϵ_X is larger. In addition to $\epsilon_X = 1$, with a small privacy budget for the labels $\epsilon_Y \in \{1, 2.5\}$, ScalableRR still outperforms the baselines in most of the cases, offering stronger privacy protection with better model utility, i.e., LDP at both embedding feature and label levels instead of only LDP on the embedding features as in the baselines.

The reason is that, in the baseline approaches, the model utility is notably affected by the size of the embedding features. Thanks to the dimension-scalable and bit-aware properties, ScalableRR can achieve high model accuracy and AUC values under rigorous privacy budgets. In addition, ScalableRR achieves the highest improvement in the AG dataset, since it is a balanced dataset compared with the highly imbalanced CelebA dataset, in which ScalableRR achieves the least improvement. Addressing imbalanced data in FL under DP [89] is out-of-scope of this study.

LDP-preserving Gradients and the Anonymizer [191]. We observe the same phenomenon when baseline approaches are applied on gradients without and with the anonymizer [191], even though the gaps between ScalableRR and the baselines are (marginally) smaller. Without using the anonymizer in the baselines, ScalableRR (ϵ_X,$\epsilon_Y =$ 1) achieves accuracy and AUC improvements of 44.95% and 37.52% in the AG dataset ($p = 3.9e - 20$), 12.82% and 12.92% in the SEC dataset, 24.17% and 13.40% in the FEMNIST dataset ($p = 4.1e - 11$), respectively, over the best baseline PM-SUB.

When the anonymizer is applied in the baseline approaches (Figure 3.11), ScalableRR achieves accuracy and AUC improvements of 39.38% and 32.75% in the AG dataset ($p =$ $1.2e - 16$), 13.59% and 13.69% in the SEC dataset ($p = 2.1e - 10$), and 23.12% and 37.02% in the FEMNIST dataset ($p = 2.8e - 11$), respectively, over HM. In the case of the CelebA dataset (Figure 3.12), ScalableRR outperforms the best baselines (i.e., Three-outputs and HM) with AUC ($p = 3.1e - 2$) improvements of 1.28% and 0.3%, with and without the anonymizer, across all the 40 attributes. On the other hand, the model utility in the baselines is affected by the size of gradients and the training rounds (when the anonymizer is not applied) and their finite numbers of randomization outputs of the gradients [226].

Expansion Factor. On the one hand, increasing the feature vector dimensionality leads to better utility as it enriches the representation of the input domain in general. On the other hand, the increase in dimensionality can degrade the utility given fixed privacy guarantees (i.e., more noise is introduced in randomizing each embedding feature). Technically, our approach benefits from the increase in the dimensionality due to better control of its impact on utility. Hence, Figure 3.13 shows that when we expand the embedding

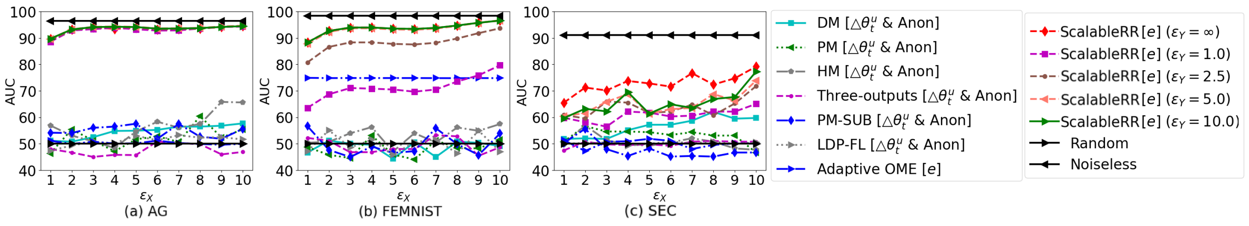


Figure 3.11 AUC values of LDP algorithms applied applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191].

feature vector by the factor d under the same total privacy budget, the model utility of ScalableRR is significantly improved, especially given rigorous privacy budgets. Given $\epsilon_X = 1$, ScalableRR registers an improvement of 27.64% in AUC and 32.70% in accuracy when d increases from 100 to 768 compared with almost no improvement of adaptive OME, the best baseline. This observation strengthens our theoretical analysis that ScalableRR achieves better dimension-scalability and data utility than the baselines.

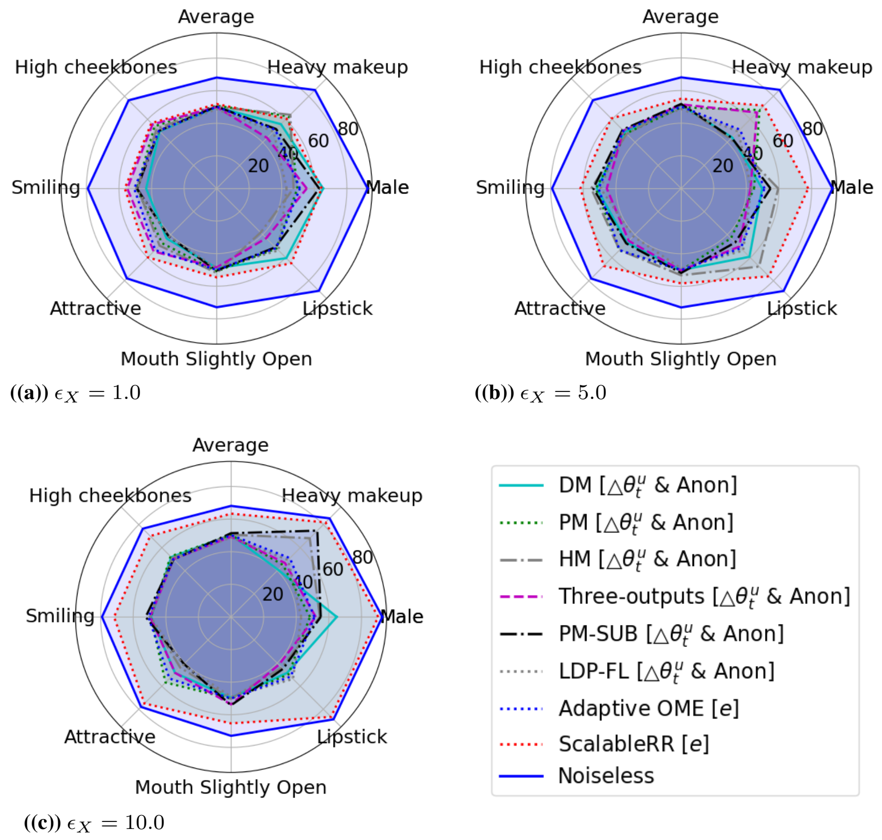


Figure 3.12 AUC values of LDP algorithms applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191] in the CelebA dataset.

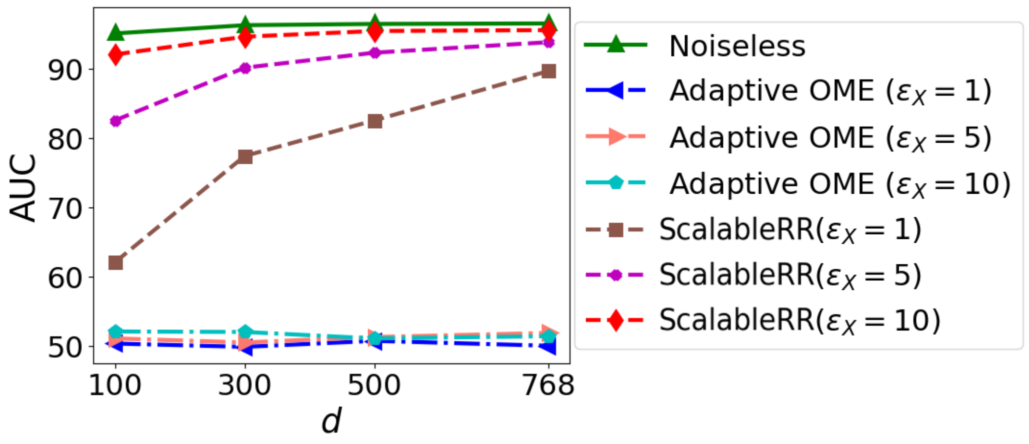
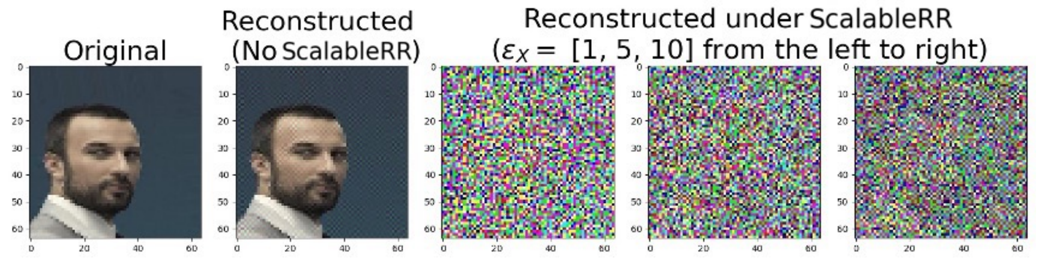


Figure 3.13 AUC values of ScalableRR and Adaptive OME when varying the factor d ($r = 1$) in the AG dataset.

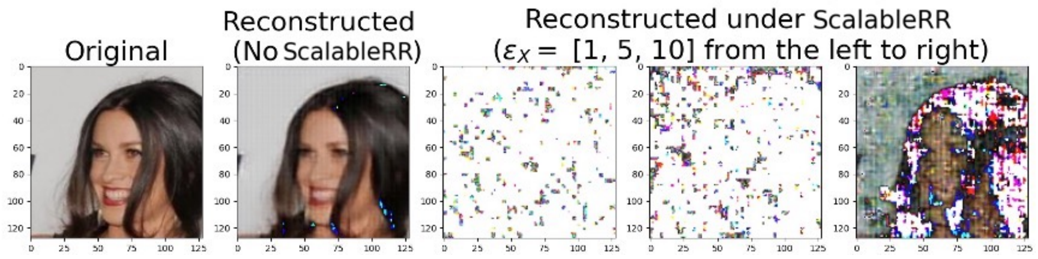
Defending Against Data Reconstruction Attacks. Figure 3.14 illustrates examples of reconstructed images in no-protection environment (Column 2) and under protection of ScalableRR correspondingly. We found that ScalableRR prevents the attacker from reconstructing the original images (Columns 3-6). Up to the privacy budget $\epsilon_X = 5$, the reconstructed images are very noisy. Importantly, we achieve high model utility under this level of LDP protection. Therefore, ScalableRR is effective in defending the attacks without sacrificing model utility. Up to a very high budget $\epsilon_X = 10$, parts of the original images start to appear informing noticeable privacy risk.

ScalableRR is Tested in a Real-world Mobile-Cloud Federated Learning System. To show how our proposed ScalableRR adopt in real-world scenarios, we incorporated ScalableRR into a data collection module of a FLSys [95], which is one of the state of the art real-world mobile-cloud FL systems, to protect local data in training a human activity recognition system.

FLSys is designed to work on smart phones with mobile sensing data. It balances model performance with resource consumption, tolerates communication failures, and achieves scalability. In FLSys, different DL models with different FL aggregation methods can be trained and accessed concurrently by different apps. Furthermore, FLSys provides advanced privacy preserving mechanisms and a common API for third-party app developers


Original
Reconstructed (No ScalableRR)
Reconstructed under ScalableRR (ε_X = [1, 5, 10] from the left to right)

((**a**)) Defending data reconstruction attack [143].


Original
Reconstructed (No ScalableRR)
Reconstructed under ScalableRR (ε_X = [1, 5, 10] from the left to right)

((**b**)) Defending data reconstruction attack [49].

Figure 3.14 ScalableRR v.s. reconstruction attacks.

to access FL models. FLSys adopts a modular design and is implemented in Android and AWS cloud. The FLSys is tested with a human activity recognition (HAR) model. HAR sensing data was collected in the wild from 100+ college students during a 4-month period. In HAR-Wild, a CNN model is tailored to mobile devices, with a data augmentation mechanism to mitigate the problem of non-Independent and Identically Distributed data.

For privacy protection mechanisms, we train the HAR-Wild with many DP mechanisms, including User-level DP mechanism, Duchi mechanim (DM), Piecewise mechanism (PM), Three-outputs, and ScalableRR. Then, we evaluated the trade-offs between model utility and privacy budget for different versions of HAR-Wild with privacy mechanisms, as shown in Table 3.2. As expected, the model utility decreases as privacy budget ϵ_X tightens. From this table, we select the best User-DP model (i.e., the one with $\epsilon_X = 8$) and the best LDP model (i.e., ScalableRR with $\epsilon_X = \epsilon_Y = 8$) in terms of accuracy, and compare them with the models with and without augmentation in Figure 3.15. The results show that HAR-Wild with User-DP achieves a model accuracy of 69.70%, which is just 2.11% lower than the model without privacy protection. HAR-Wild with LDP

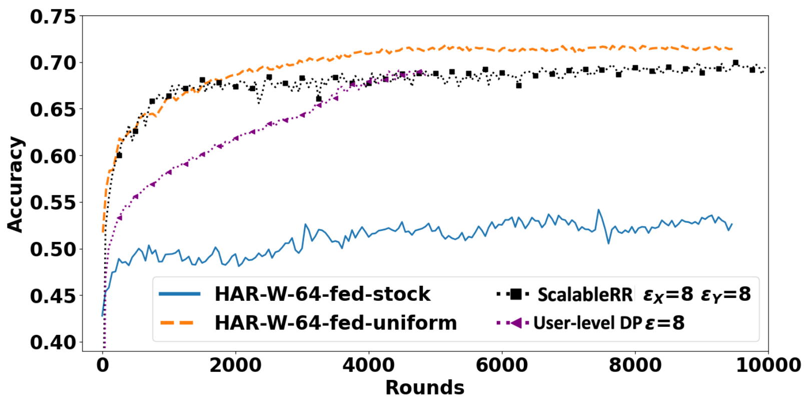


Figure 3.15 Comparison of FL HAR-Wild Versions, with and without Data Augmentation, and with and without Privacy Protection.

(ScalableRR) achieves an accuracy of 69.43%, which is just 2.38% lower than the noiseless model. Note that our defense successfully prevents the server to reconstruct recognizable sensor signals and infer its associated ground-truth labels. One of the reasons is that it is more challenging to infer whether a time series of sensor signals belongs to a particular client than other domain applications. When using a tighter privacy budget, e.g., $\epsilon_X = \epsilon_Y = 4$ or 2, the gap between ScalableRR and Noiseless model becomes bigger. This is due to the fact that ScalableRR has not been designed for imbalanced data and cannot work well with significantly imbalanced data as the HAR dataset, especially when reducing the privacy budget ϵ_Y for protecting the labels. It is noted that both privacy protection mechanisms offer rigorous privacy guarantees in FLSys without significant computational overhead.

3.6 Discussion

In this chapter, we proposed ScalableRR, a novel dimension-scalable and bit-aware RR mechanism. To optimize the trade-off among dimensionality, data utility, and privacy protection, ScalableRR introduces a ***bit-aware term*** for better data utility and a ***dimension-scalable temperature*** for better control of the randomization probabilities. Our key idea is that when the dimension of a feature vector increases, one can always find a suitable temperature such that the changes in the randomization probabilities and in the total

Table 3.2 Macro-model Performance for HAR-Wild for Different Types of Privacy Protection Mechanisms and Different Parameters

DP Mechanism	Privacy Budget	Accuracy	Precision	Recall	F1-score
Noiseless	$\epsilon_X \to \infty$	0.7181	0.7464	0.7419	0.7378
User-level DP	$\epsilon_X = 2$	0.5399	0.5264	0.5797	0.5259
User-level DP	$\epsilon_X = 4$	0.5973	0.5603	0.6297	0.5502
User-level DP	$\epsilon_X = 8$	0.6970	0.6333	0.7264	0.6523
ScalableRR	$\epsilon_X = \epsilon_Y = 2$	0.4251	0.3667	0.3715	0.3277
ScalableRR	$\epsilon_X = \epsilon_Y = 4$	0.5193	0.4607	0.5110	0.4416
ScalableRR	$\epsilon_X = \epsilon_Y = 8$	0.6943	0.6885	0.7359	0.7031
DM	$\epsilon_X = 2$	0.4846	0.4286	0.5233	0.4201
DM	$\epsilon_X = 4$	0.5122	0.4307	0.4998	0.4360
PM	$\epsilon_X = 2$	0.4857	0.4086	0.4267	0.3944
PM	$\epsilon_X = 4$	0.5065	0.4245	0.4686	0.4222
HM	$\epsilon_X = 2$	0.4791	0.3961	0.3714	0.3714
HM	$\epsilon_X = 4$	0.5353	0.4521	0.4508	0.4431
Three-Outputs	$\epsilon_X = 2$	0.2906	0.2662	0.2348	0.0192
Three-Outputs	$\epsilon_X = 4$	0.2946	0.3288	0.2424	0.2386

privacy budget are marginal. Hence, ScalableRR achieves better dimension-scalability and utility under rigorous LDP protection than existing mechanisms. Theoretical analysis and experiments, especially real-worl trials on the real-world mobile-cloud FLSys, show that ScalableRR outperforms baselines in text and image classification using several benchmark datasets and in real-world applications. The results also show that ScalableRR is effective in defending against data reconstruction attacks in FL.

ScalableRR opens several fundamental research directions. An exciting and challenging future work is developing a new adaptive composition that could be integrated into ScalableRR to enhance privacy-dimensionality-utility trade-offs. In addition, we can further investigate other non-linear functions, such as tanh, which may offer sharper curves to further optimize the randomization probabilities across bits.

Furthermore, one can adapt ScalableRR to randomize sparse data distribution. In certain scenarios, some features consist of more non-zero highly influential bits (sign and exponent bits); while other features consist of more non-zero low influential bits (fractions). In this case, we can use multiple encoding schemes (i.e., varying m) to encode the features. We can directly apply ScalableRR on each encoding scheme without affecting the total privacy budget consumed across all bits and features.

Given categorical data, one can advance the concept of ScalableRR to offer heterogenous protection across major classes (higher probability to be randomized) and minor classes (lower probability to be randomized). In addition, we can extract embedding features from discrete features, such as age, and apply ScalableRR on top. That means ScalableRR has a great potential to better balance privacy-utility trade-offs in applications using heterogeneous data. It is important to note that ScalableRR does not intend to defend against property inference attacks [129]. For instance, an accurate data reconstruction may not be needed to infer the gender or age group of the users. We will need to optimize the utility-privacy trade-offs further, and we call for future efforts from the community to address these specific settings.

CHAPTER 4

XRAND: DIFFERENTIALLY PRIVATE DEFENSE AGAINST EXPLANATION-GUIDED ATTACKS

4.1 Preamble

Recent development in the field of explainable artificial intelligence (XAI) has helped improve trust in Machine-Learning-as-a-Service (MLaaS) systems, in which an explanation is provided together with the model prediction in response to each query. Technically, XAI also opens a door for adversaries to gain insights into the black-box models in MLaaS, thereby making the models more vulnerable to several attacks. For example, feature-based explanations (e.g., SHAP) could expose the top important features that a black-box model focuses on. Such disclosure has been exploited to craft effective backdoor triggers against malware classifiers. To address this trade-off, we introduce a new concept of achieving local differential privacy (LDP) in the explanations, and from that we establish a defense, called XRAND, against such attacks. We show that our mechanism restricts the information that the adversary can learn about the top important features, while maintaining the faithfulness of the explanations.

4.2 Background

Local explainers. The goal of model explanations is to capture the importance of each feature of a given point of interest with respect to the decision made by the classifier and which class it is pushing that decision toward. Given a sample $x \in \mathbb{R}^d$ where x_j denotes the j^{th} feature of the sample, let f be a model in which $f(x)$ is the probability that x belongs to a certain class. An explanation of the model's output $f(x)$ takes the form of an explanation vector $w_x \in \mathbb{R}^d$ where the j^{th} element of w_x denotes the degree to which the feature x_j influences the model's decision. Generally, higher values of w_{x_j} imply a higher impact.

Perturbation-based explainers, such as SHAP [119], obtain an explanation vector w_x for x via training a surrogate model of the form $g(x) = w_{x_0} + \sum_{j=1}^{d} w_{x_j} x_j$ by minimizing a loss function $\mathcal{L}(f, g)$ that measures how unfaithful g is in approximating f.

- *Sample-level explanation.* In the context of this paper, we refer to w_x as a sample-level explanation.
- *Aggregated explanation.* We denote an aggregated explanation w as the sum of explanation vectors across samples in a certain set $\mathcal{X}$, i.e., $w_{\mathcal{X}} = \sum_{x \in \mathcal{X}} w_x$. When $\mathcal{X}$ is clear from the context, we shall use a shorter notation w.

Local Differential Privacy (LDP). LDP is one of the state-of-the-arts and provable approaches to achieve individual data privacy. LDP-preserving mechanisms [8, 19, 52, 63, 202] generally build on the ideas of randomized response (RR) [204].

Definition 6. *ε-LDP. A randomized algorithm $\mathcal{A}$ satisfies ε-LDP, if for any two inputs x and x', and for all possible outputs $\mathcal{O} \in Range(\mathcal{A})$, we have: $Pr[\mathcal{A}(x) = \mathcal{O}] \leq e^{\varepsilon} Pr[\mathcal{A}(x') = \mathcal{O}]$, where ε is a privacy budget and $Range(\mathcal{A})$ denotes every possible output of $\mathcal{A}$.*

The privacy budget ε controls the amount by which the distributions induced by inputs x and x' may differ. A smaller ε enforces a stronger privacy guarantee.

4.3 XAI-guided Attack Against MLaaS

We discuss how XAI can be used to gain insights into MLaaS models, and establish the threat model for our work.

4.3.1 Exposing MLaaS via XAI

From a security viewpoint, releasing additional information about a model's mechanism is a perilous prospect. As a function of the model that is trained on a private dataset, an explanation may unintentionally disclose critical information about the training set, more than what is needed to offer a useful interpretation. Moreover, the explanations may also expose the internal mechanism of the black-box models. For example, first, the behavior of explanations varies based on whether the query sample was a member of the training dataset, making the model vulnerable to membership inference attacks [182]. Second, the explanations can be coupled with the predictions to improve the performance of generative models which, in turn, strengthens some model inversion attacks [225]. Furthermore, releasing the explanations exposes how the black-box model acts upon an input sample, essentially giving up more information about its inner workings for each query, hence, model extractions attacks can be carried out with far fewer queries, as discussed in [134, 138].

Finally, [178] argues that the explanations allow an adversary to gain insight into a model's decision boundary in a generic, model-agnostic way. The SHAP values can be considered as an approximation of the confidence of the decision boundary along each feature dimension. Hence, features with SHAP values that are near zero infer low-confidence areas of the decision boundary. On the other hand, features with positive SHAP values imply that they strongly contribute to the decision made by the model. As a result, it provides us with an indication of the overall orientation for each feature, thereby exposing how the model rates the importance of each feature.

4.3.2 XAI-guided Backdoor Attack against MLaaS

The XBA on malware classifiers [178] suggests that explanations make the model vulnerable to backdoor attacks, as they reveal the top important features. Thus, it is natural to mount XBA against a black-box model in MLaaS where an explanation is returned for each query.

System Model. Figure 4.1 illustrates the system model for our work. We consider an MLaaS system where a malware classifier is deployed on a cloud-based platform. For training, the system crowdsources threat samples via user-submitted binaries to assemble a

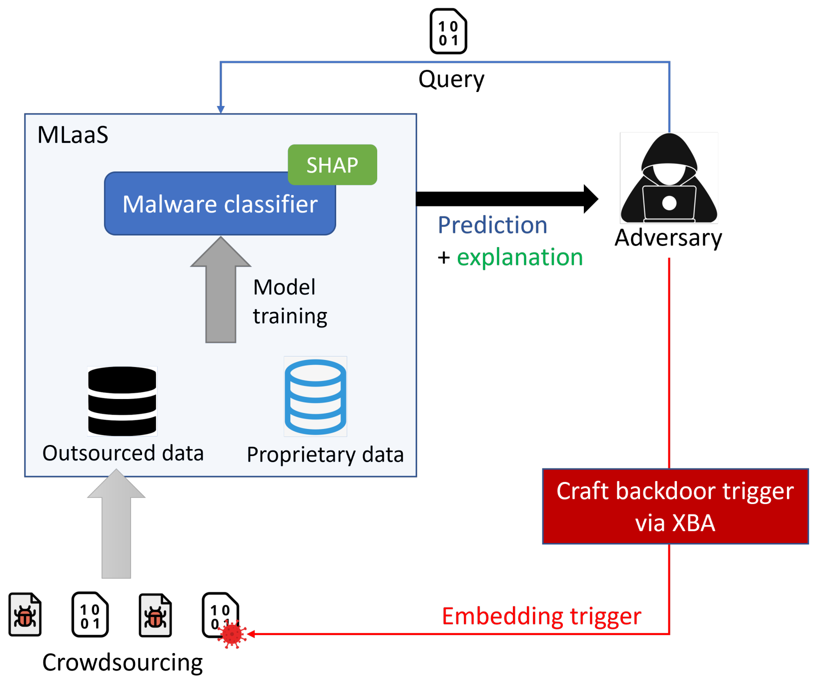


Figure 4.1 System model of a cloud-hosted malware classifier that leverages crowdsourced data for model training.

set of outsourced data. This set of outsourced data is then combined with a set of proprietary data to construct the training data to train the malware classifier.

We denote $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ as the set of proprietary training data. The dataset contains sample $x_n \in \mathbb{R}^d$ and its ground-truth label $y_n \in \{0, 1\}$, where $y_n = 0$ denotes a goodware sample, and $y_n = 1$ denotes a malware sample. On input x, the model $f : \mathbb{R}^d \to \mathbb{R}$ outputs the score $f(x) \in [0, 1]$. This score is then compared with a threshold of 0.5 to obtain the predicted label for x.

During inference time, given a query containing a binary sample x, the system returns the predicted label with a SHAP explanation w_x for the decision ($w_x \in \mathbb{R}^d$). We consider an adversary who plays the role of a user in this system and can send queries at his discretion. The adversary exploits the returned explanations to craft backdoor triggers that will be injected to the system via the crowdsourcing process, thereby poisoning the outsourced data.

Threat Model. The attacker's goal is to alter the training procedure by injecting poisoned samples into the training data, generating poisoned training data such that the resulting backdoored classifier differs from a clean classifier. An ideal backdoored classifier has the same response to a clean input as the clean classifier, but it gives an attack-chosen prediction when the input is embedded with the trigger.

Our defense assumes a strong adversary such that he can tamper with the training dataset at his discretion without major constraints. To prevent the adversary from setting arbitrary values for the features in the trigger, the set of values that can be used is limited to the ones that exist in the dataset. This threat model promotes a defense under worst-case scenarios from the perspective of the defender.

Crafting backdoor triggers. To craft a backdoor trigger in XBA, the adversary tries to obtain the top goodware-oriented feature by querying classifier f with samples $\{x\}_{x \in A}$ from their dataset A and obtaining the SHAP explanation w_x for each of them. The sum of the SHAP values across all queried samples $w_A = \sum_{x \in A} w_x$ indicates the importance of each feature, and whether it is goodware- or malware-oriented. Then, the attacker greedily selects a combination of the most goodware-oriented features to create the trigger [178].

4.4 XRAND – Local DP Defense

This section describes our defense, XRAND, a novel two-step explanation-guided randomized response (RR) mechanism. Our idea is to incorporate the model explainability into the randomization probabilities in XRAND to guide the aggregated explanation while minimizing the difference between the perturbed explanation's surrogate model $g'(x)$ and the model $f(x)$ at the sample-level explanation. We call the difference between $g'(x)$ and $f(x)$ an explanation loss $\mathcal{L}$, quantified as follows:

$$\mathcal{L} = \sum_{z \in N(x)} \left(g'(z) - f(z)\right)^2 \exp\left(-\frac{\|z - x\|^2}{\sigma^2}\right) \tag{4.1}$$

Algorithm 2 XRAND: Explanation-guided RR mechanism

```
Input: model $f$, dataset $\mathcal{D}$, aggregated explanation $w$, $\varepsilon$, $k$, $\tau$, test sample $x$
Output: $\mathbb{S}$, $\varepsilon$-LDP $w'_x$
1: Step 1 - At aggregated explanation:
2: for $x_n \in \mathcal{D}$ do
3:     Compute $\mathcal{L}(x_n)$ # using Equation 4.1
4:     for $i \in [1, k]$, $j \in [k+1, k+\tau]$ do
5:         Compute $\mathcal{L}(x_n)(i, j)$ # using Equation 4.1
6:         Compute $\Delta_{\mathcal{L}}(i, j)$ # using Equation 4.3
7:     end for
8: end for
9: Randomizing $w$:
10: $w' \leftarrow \text{XRAND}(w, \varepsilon, k, \tau, \Delta_{\mathcal{L}}(i, j))$ # using Equation 4.2
11: Return $\mathbb{S}$
12: Step 2 - At sample-level explanation:
13: $w_x \leftarrow$ SHAP explanation for $x$
14: $w'_x \leftarrow$ Solve the optimization problem in (4.5)
15: Return $w'_x$
```

where x is an input, its neighborhood $N(x)$ is generated by the explainer's sampling method. This function captures the difference between the modified explainer's linear surrogate model $g'(x)$ and $f(x)$, essentially measuring how unfaithful g' is in approximating f.

To defend against explanation-guided attacks that utilize the top-k features of the aggregated explanation (e.g., XBA exploits the top-k goodware-oriented features), our idea is to randomly disorder some top-k features under LDP guarantees, thereby protecting the privacy of those features. This raises the following question: *What top-k features and which data samples should be randomized to optimize the explainability of data samples while guaranteeing that the attackers cannot infer the top-k features?*

Algorithm Overview. To answer this question, we first integrate the explanation loss caused by potential changes of features in the aggregated explanation into the randomized probabilities to adaptively randomize each feature in the top-k. Then, we minimize the explanation loss on each sample while ensuring the order of the features at the aggregated explanation follows the results of the first step. By doing so, we are able to optimize the trade-off between the model explainability and the privacy budget ε used in XRAND, as verified both theoretically (Section 4.4.2) and experimentally (Section 4.6). The pseudo-code of XRAND is shown in Alg. 2.

4.4.1 LDP-preserving Explanations

Step 1 (Alg. 2, lines 1-10). We first compute the aggregated explanation w over the samples of the proprietary dataset $w = \sum_{x \in \mathcal{D}} w_x$. Then we sort w in descending order and retain a mapping $v : \mathbb{N} \to \mathbb{N}$ from the sorted indices to the original indices. Given that τ is a predefined threshold to control the range of out-of-top-k features that some of top-k features can swap, and β is a parameter bounded in Theorem 6 under a privacy budget ε, XRAND defines the probability of flipping a top-k feature i to an out-of-top-k feature j as follows:

$$\forall i \in [1, k], j \in [k+1, k+\tau], \tau \geq k :$$
$$i = \begin{cases} i, \text{ with probability } p_i = \dfrac{\exp(\beta)}{\exp(\beta) + \tau - 1}, \\ j, \text{ with probability } q_{i,j} = \dfrac{\tau - 1}{\exp(\beta) + \tau - 1} q_j \end{cases} \tag{4.2}$$

where $q_j = \frac{\exp(-\Delta_{\mathcal{L}}(i,j))}{\sum_{t \in [k+1, k+\tau]} \exp(-\Delta_{\mathcal{L}}(i,t))}$ and $\Delta_{\mathcal{L}}(i, j)$ is the aggregated changes of $\mathcal{L}$ (Equation 4.1) when flipping features i and j, which is calculated as follows:

$$\Delta_{\mathcal{L}}(i, j) = \frac{1}{N} \sum_{n=1}^{N} (|\mathcal{L}(x_n) - \mathcal{L}(x_n)(i, j)|) \tag{4.3}$$

where $\mathcal{L}(x_n)$ is the original loss $\mathcal{L}$ of a sample $x_n \in \mathcal{D}$ and $\mathcal{L}(x_n)(i, j)$ is the loss $\mathcal{L}$ of the sample x_n after flipping features i and j (Alg. 2, lines 3,5).

After randomizing the aggregated explanation, we obtain the set $\mathbb{S}$ of features that need to be flipped in the aggregated explanation, as follows:

$$\mathbb{S} = \{(i, j) | i \text{ and } j \text{ are flipped}, i \in [1, k], j \in [k+1, k+\tau]\} \tag{4.4}$$

Step 2 (Alg. 2, lines 11-14). For each input test sample x, we proceed with sample-level explanation for finding the noisy explanation w'_x. First, we generate a set of constraints $\mathbb{Q} = \{(i, j) | w'_{x_i} \leq w'_{x_j}\}$ that is *sufficient* for $\mathbb{S}$. In particular, for each pair $(i, j) \in \mathbb{S}$, we add the following pairs to $\mathbb{Q}$:

$$(v(i+1), v(j)); (v(j), v(i-1)); (v(i), v(j-1)); (v(j+1), v(i))$$

Given w_x as the SHAP explanation of x, we aim to find $\phi \in \mathbb{R}^d$ such that $w'_x = w_x + \phi$ satisfies the constraints in $\mathbb{Q}$ while minimizing the loss $\mathcal{L}$. To obtain ϕ, we solve the following optimization problem:

$$\min_{\phi} \sum_{z \in N(x)} \left((w_x + \phi)^T z - f(z)\right)^2 \exp\left(-\frac{\|z - x\|^2}{\sigma^2}\right) + \lambda \|\phi\| \tag{4.5}$$

$$\text{s.t. } w_{x_i} + \phi_i \leq w_{x_j} + \phi_j, \quad \forall (i, j) \in \mathbb{Q}$$

$$\phi_i = 0 \qquad \forall i \notin \mathbb{Q}$$

where λ is a regularization constant.

The resulting noisy explanation will be $w'_x = w_x + \phi$. This problem is convex and can be solved by convex optimization solvers [42, 100].

4.4.2 Privacy Guarantees of XRAND

To bound privacy loss of XRAND, we need to bound β in Equation 4.2 such that the top-k features in the explanation w' preserves LDP, as follows:

Theorem 6. *Given two distinct explanations w and $\widetilde{w}$ and a privacy budget ϵ_i,* XRAND *satisfies ε_i-LDP in randomizing each feature i in top-k features of w, i.e.,* $\frac{P(\text{XRAND}(w_i)=z|w)}{P(\text{XRAND}(\widetilde{w}_i)=z|\widetilde{w})} \leq$

$\exp(\varepsilon_i)$, *if:*

$$\beta \leq \varepsilon_i + \ln(\tau - 1) + \ln(\min \frac{\exp(-\Delta_{\mathcal{L}}(i,j))}{\sum_{t=k+1}^{k+\tau} \exp(-\Delta_{\mathcal{L}}(i,t))})$$

where $z \in Range(\text{XRAND})$. ***Proof:*** *See Appendix A.20.*

Based on Theorem 6, the total privacy budget ε to randomize all top-k features is the sum of all the privacy budget ϵ_i, i.e., $\varepsilon = \sum_{i=1}^{k} \varepsilon_i$, since each feature i is randomized independently. From Theorem 6 and Equation 4.2, it can be seen that as the privacy budget ε increases, β can increase and the flipping probability $q_{i,j}$ decreases. As a result, we switch fewer features out of top-k.

Privacy and Explainability Trade-off. To understand the privacy and explainability trade-off, we analyze the data utility of XRAND mechanism through the sum square error (SSE) of the original explanation w and the one resulting from XRAND w'. The smaller the SSE is, the better data utility the randomization mechanism achieves.

Theorem 7. *Utility of* XRAND*: $SSE = \sum_{x \in \mathcal{D}} \sum_{i=1}^{d} (w'_{x_i} - w_{x_i})^2 = \sum_{x \in \mathcal{D}} \sum_{i=1}^{k+\tau} (w'_{x_i} - w_{x_i})^2$, where d is the number of features in the explanation.*

Proof. It is easy to see that we only consider the probability of flipping the top-k features to be out-of-the-top-k up to the feature $k + \tau$. Thus, all features after $k + \tau$, i.e., from $k + \tau + 1$ to d are not changed. Hence the theorem follows. □

From the theorem, at the same ε, the smaller the τ, the higher the data utility that XRAND achieves. Intuitively, if τ is large, it is more flexible for the top-k features to be flipped out, but it will also impair the model explainability since the original top-k features are more likely to be moved far away from the top k. With high values of ε, we can obtain a smaller SSE, thus, achieving better data utility. The effect of ε and τ on the SSE value is illustrated in Figure A.21 (Appendix).

4.5 Certified Robustness

Our proposed XRAND can be used as a defense against the XBA since it protects the top-k important features. We further establish the connection between XRAND with certified robustness against XBA. Given a data sample x: 1) In the training time, we guarantee that up to a portion of poisoning samples in the outsourced training data, XBA fails to change the model predictions; and 2) In the inference time, we guarantee that up to a certain backdoor trigger size, XBA fails to change the model predictions. A primer on certified robustness is given in Appendix A.21.

4.5.1 Training-time Certified Robustness

We consider the original training data $\mathcal{D}$ as the proprietary data, and the explanation-guided backdoor samples $\mathcal{D}_o$ as the outsourced data inserted into the proprietary data. The outsourced data $\mathcal{D}_o$ alone may not be sufficient to train a good classifier. In addition, the outsourced data inserted into propriety data can lessen the certified robustness bound of the propriety data. Therefore, we cannot quantify the certified poisoning training size of the outsourced data $\mathcal{D}_o$ directly by applying a bagging technique [93]. To address this problem, we quantify the certified poisoning training size r of $\mathcal{D}_o$ against XBA by uncovering its correlation with the poisoned training data $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_o$.

Given a model prediction on a data sample x using $\mathcal{D}$, denoted as $f(\mathcal{D}, x)$, we ask a simple question: "What is the minimum number poisoning data samples, i.e., certified poisoning training size $r_{\mathcal{D}}$, added into $\mathcal{D}$ to change the model prediction on x: $f(\mathcal{D}, x) \neq f(\mathcal{D}_+, x)$?" After adding $\mathcal{D}_o$ into $\mathcal{D}$, we ask the same question: "What is the minimum number poisoning data samples, i.e., certified poisoning training size $r_{\mathcal{D}'}$, added into $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_o$ to change the model prediction on x: $f(\mathcal{D}', x) \neq f(\mathcal{D}'_+, x)$?" The difference between $r_{\mathcal{D}}$ and $r_{\mathcal{D}'}$ provides us a certified poisoning training size on $\mathcal{D}_o$. Intuitively, if $\mathcal{D}_o$ does not consist of poisoning data examples, then $r_{\mathcal{D}}$ is expected to be relatively the same with $r_{\mathcal{D}'}$.

Otherwise, $r_{\mathcal{D}'}$ can be smaller than $r_{\mathcal{D}}$ indicating that $\mathcal{D}_o$ is heavily poisoned with at least $r = r_{\mathcal{D}} - r_{\mathcal{D}'}$ number of poisoning data samples towards opening backdoors on x.

Theorem 8. *Given two certified poisoning training sizes $r^*_{\mathcal{D}} = \arg\min_{r_{\mathcal{D}}} r_{\mathcal{D}}$ and $r^*_{\mathcal{D}'} = \arg\min_{r_{\mathcal{D}'}} r_{\mathcal{D}'}$, the certified poisoning training size r of the outsourced data $\mathcal{D}_o$ is:*

$$r = r^*_{\mathcal{D}} - r^*_{\mathcal{D}'} \tag{4.6}$$

Proof: *Refer to Appendix A.22 for the proof and its tightness.*

4.5.2 Inference-time Certified Robustness

It is not straightforward to adapt existing certified robustness bounds at the inference-time into XRAND, since there is a gap between model training as in existing approaches [93, 108, 149] and the model training with explanation-guided poisoned data as in our system. Existing approaches can randomize x and then derive certified robustness bounds given the varying output. This process does not consider the explanation-guided poisoned data that can potentially affect certified robustness bounds in our system. To address this gap, note that we can always find a mechanism to inject random noise into the data samples x such that the samples achieve the same level of DP guarantee as the explanations. Based on this, we can generalize existing certified bounds against XBA at the inference time in XRAND.

When explanation-guided backdoor samples are inserted into the training data, upon bounding the sensitivity that these samples change the output of f, there always exists a noise α that can be injected into a benign sample x, i.e., $x + \alpha$, to achieve an equivalent ε-LDP protection. Given the explanation w_x of x, we focus on achieving a robustness condition to $L_p(\mu)$-norm attacks, where μ is the radius of the norm ball, as follows:

$$\forall \alpha \in L_p(\mu) : f_l(x + \alpha | w_x) > f_{\neg l}(x + \alpha | w_x) \tag{4.7}$$

where $l \in \{0, 1\}$ is the true label of x and $\neg l$ is the NOT operation of l in a binary classification problem.

There should exist a correlation among α and w_x that needs to be uncovered in order to bound the robustness condition in Equation 4.7. Fundamentally, it is challenging to find a direct mapping function $\mathcal{M} : w_x \to \alpha$ so that when we randomize w_x, the change of α is quantified. We address this challenge by quantifying the sensitivity of α given the average change of the explanation of multiple samples $x \in \mathcal{X}$, as follows:

$$\Delta_{\alpha|w} = \frac{1}{|\mathcal{X}|d} \sum_{x \in \mathcal{X}} |w_x - w'_x|_1 \tag{4.8}$$

where $|\mathcal{X}|$ is the size of $\mathcal{X}$.

$\Delta_{\alpha|w}$ can be considered as a bounded sensitivity of XRAND given the input x since: **(1)** We can achieve the same DP guarantee by injecting Laplace or Gaussian noise into the input x using the sensitivity $\Delta_{\alpha|w}$; and **(2)** The explanation perturbation happens only once and is permanent, that is, there is no other bounded sensitivity associated with the one-time explanation perturbation. The sensitivity $\Delta_{\alpha|w}$ establishes a new connection between explanation perturbation and the model sensitivity given the input sample x. That enables us to derive robustness bounds using different techniques, i.e., PixelDP [108, 152] and (boosting) randomized smoothing (RS) [37, 86], since we consider the sensitivity $\Delta_{\alpha|w}$ as a part of randomized smoothing to derive and enhance certified robustness bounds.

The rest of this section only discusses the bound using PixelDP, we refer the readers to Appendix A.23 for the bound using boosing RS. Given a randomized prediction $f(x)$ satisfying (ε, δ)-PixelDP w.r.t. a $L_p(\mu)$-norm metric, we have:

$$\forall l \in \{0, 1\}, \forall \alpha \in L_p(\mu) : \mathbb{E}f_l(x) \leq e^{\varepsilon}\mathbb{E}f_l(x + \alpha) + \delta \tag{4.9}$$

where $\mathbb{E}f_l(x)$ is the expected value of $f_l(x)$, ε is a predefined privacy budget, and δ is a broken probability. When we use a Laplace noise, $\delta = 0$.

We then apply PixelDP with the sensitivity $\Delta_{\alpha|w}$ and a noise standard deviation $\sigma = \frac{\Delta_{\alpha|w}\mu}{\varepsilon}$ for Laplace noise, or $\sigma = \frac{\Delta_{\alpha|w}\mu\sqrt{2\ln(1.25/\delta)}}{\varepsilon}$ for Gaussian noise. From that, when maximizing the attack trigger's magnitude μ: $\mu_{\max} = \max_{\mu \in \mathbb{R}^+} \mu$ such that the generalized robustness condition (Equation 4.9) holds, the prediction on x using XRAND is robust against the XBA up to $\mu_{\max}$. As a result, we have a robustness certificate of $\mu_{\max}$ for x.

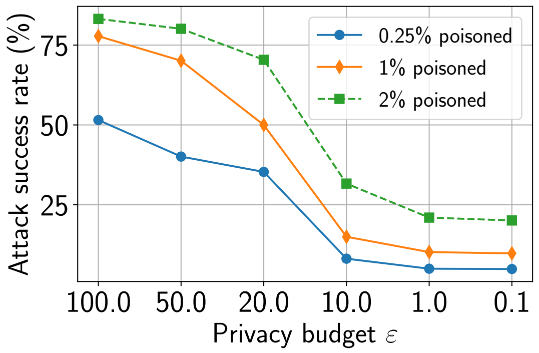
4.6 Experiments

To evaluate the performance of our defense, we conduct the XBA proposed by [178] against the explanations returned by XRAND to create backdoors on cloud-hosted malware classifiers. Our experiments aim to shed light on understanding (1) the effectiveness of the defense in mitigating the XBA, and (2) the faithfulness of the explanations returned by XRAND. The experiments are conducted using LightGBM [12] and EmberNN [178] classification models that are trained on the EMBER [12] dataset. A detailed description of the experimental settings can be found in Appendix A.24. We quantify XRAND using:

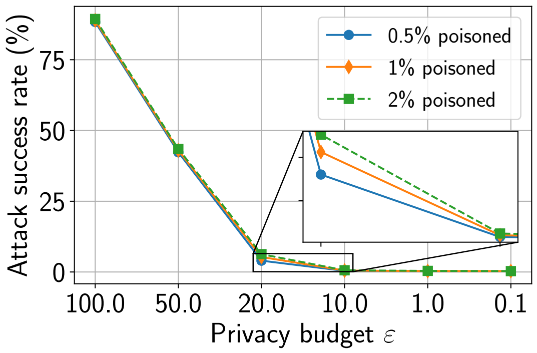
Attack success rate. This metric is defined as the portion of trigger-embedded malware samples that are classified as goodware by the backdoored model. Note that we only embed the trigger into malware samples that were classified correctly by the clean model. The primary goal of our defense is to reduce this value.

Log-odds. To evaluate the faithfulness of our XRAND explanation, we compare the log-odds score of the XRAND explanations with that of the ones originally returned by SHAP. Based on the explanation of a sample, the log-odds score is computed by identifying the top 20% important features that, if erased, can change the predicted label of the sample [184]. Then we obtain the change in log-odd of the prediction score of the original sample and the sample with those features erased. The higher the log-odds score, the better the explanation in terms of identifying important features. To maintain a faithful explainability, the XRAND explanations should have comparable log-odds scores as the original SHAP explanations.

Attack Mitigation. We observe the attack success rate of XBA when our XRAND explanations are used to craft backdoor triggers. We set k to be equal to the trigger size of the attack, and fix the predefined threshold $\tau = 50$. Figure 4.2 highlights the correlation



((**a**)) LightGBM

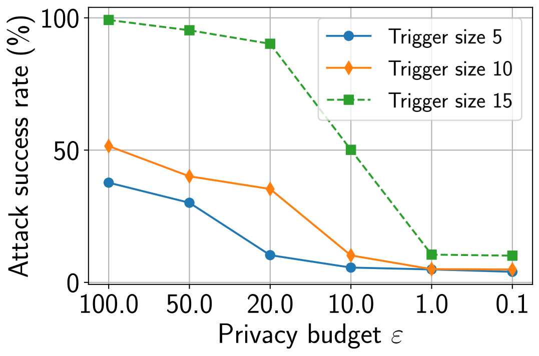


((**b**)) EmberNN

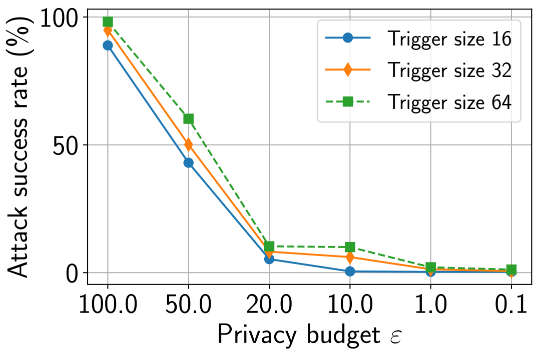
Figure 4.2 Attack success rate as a function of privacy budget ε and the portion of poisoned samples on LightGBM and EmberNN.

between the attack success rate and the privacy budget ε in our defense. Intuitively, the lower the ε, the more obfuscating the top goodware-oriented features become. Hence, Figs. 4.2(a) and 4.2(b) show that the attack success rate is greatly diminished as we tighten the privacy budget, since the attacker has less access to the desired features. Moreover, in a typical backdoor attack, injecting more poisoned samples into the training data makes the attack more effective. Such behavior is exhibited in both LightGBM (Figure 4.2(a)) and EmberNN (Figure 4.2(b)), though EmberNN is less susceptible to the increase of poisoned data. Empirically, the attacker wishes to keep the number of poisoning samples relatively low to remain stealthy. At a 1% poison rate, our defense manages to reduce the attack success rate from 77.8% to 10.2% with $\varepsilon = 1.0$ for LightGBM. It performs better with EmberNN where the attack success rate is reduced to 5.3% at $\varepsilon = 10.0$.

Additionally, we examine the effect of the trigger sizes on our defense. The trigger size denotes the number of features that the attacker modifies to craft poisoned samples. We vary the trigger size consistently with previous work [178]. In backdoor attacks, a larger trigger makes the backdoored model more prone to misclassification, thus improving the attack success rate. Figure 4.6 shows that the attack works better with large triggers on both models, technically, as aforementioned, the attacker would prefer small trigger sizes for the sake of stealthiness. This experiment shows that, for the trigger sizes that we tested, our



((**a**)) LightGBM



((**b**)) EmberNN

Figure 4.3 Attack success rate as a function of trigger size and privacy budget ε.

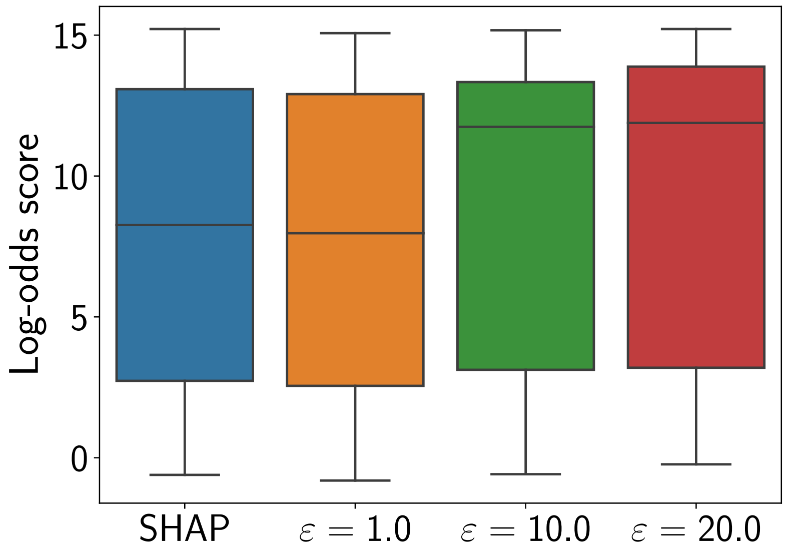
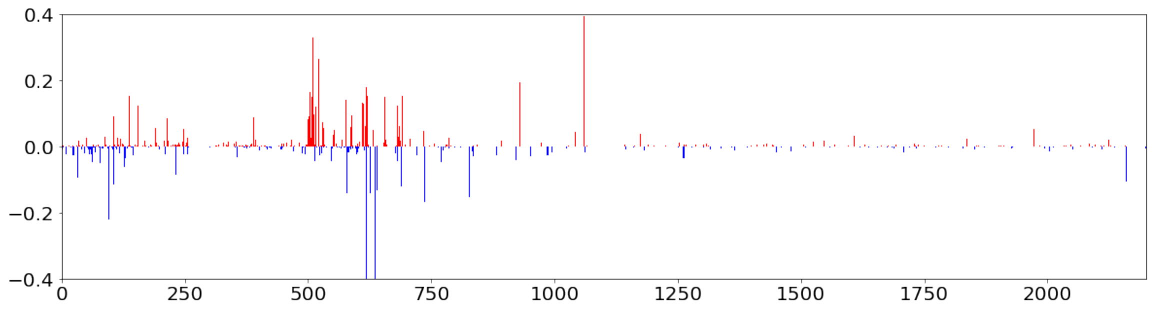


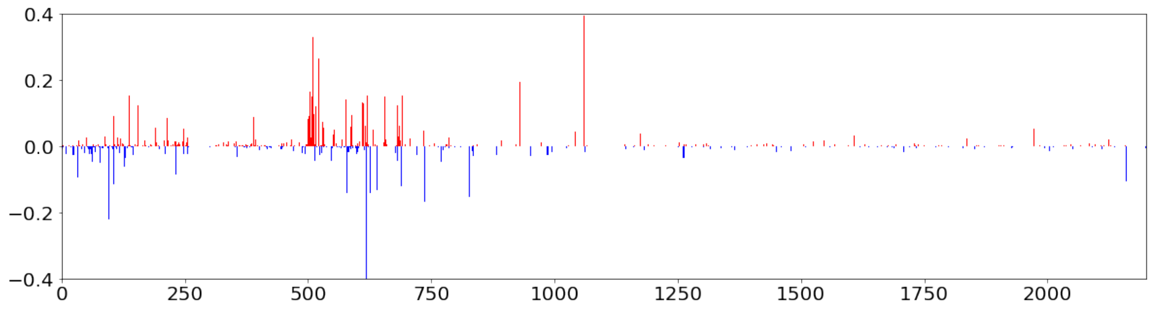
Figure 4.4 Log-odds score of the explanations of 20,000 goodware and malware samples.

proposed defense can successfully maintain a low attack success rate given a wide range of the privacy budget $\varepsilon \in [0.1, 10]$ (Figure 4.2(a), 4.2(b)).

We refer the readers to Appendix A.24 for more experimental results on additional malware datasets and the evaluation of our robustness bounds. In short, regarding the training-time bound, we observe that a smaller privacy budget $\varepsilon \in [0.1, 10]$ results in a more robust model against the XBA. As for the inference-time bound, we obtain high certified accuracy (Equation A.90) under rigorous privacy budgets, i.e., 89.17% and 90.42% at $\varepsilon \in [0.1, 1.0]$. We notice that the PixelDP-based bound attains a stronger performance



((**a**)) SHAP explanation (without XRAND)



((**b**)) XRAND explanation

Figure 4.5 Visualizing the SHAP explanation and our XRAND explanation of a test sample.

than using boosting RS. This is because boosting RS is not designed for models trained on backdoored data like XRAND.

Faithful Explainability. From the previous results, we observe that a wide range of the privacy budget $\varepsilon \in [0.1, 10]$ provides a good defense against the XBA. The question remains whether the explanations resulting from these values of ε are still faithful. Figure 4.4 shows the log-odds score of the original explanations returned by SHAP and of the ones after applying our XRAND mechanism. The XRAND explanations at $\varepsilon = 1.0, 10.0$ have comparable log-odds scores to those of SHAP. This is because our defense works with small values of k (e.g., $k = 10$). Therefore, XRAND only randomizes the SHAP values within a small set of top goodware-oriented features. As a result, XRAND can still capture the important features in its explanations.

Furthermore, we visualize the explanation of a test sample before and after applying XRAND in Figs. 4.5(a) and 4.5(b), respectively. As can be seen, the SHAP values of the two explanations largely resemble one another, except for minor differences in less than 10 features (out of 2,351 features). Importantly, the XRAND explanation evidently manifests similar sets of important malware and goodware features as the original explanation by SHAP, which also explains the comparable log-odds score in Figure 4.4. More visualizations on XRAND can be found in Appendix A.25.

4.7 Discussion

In this chapper, we have shown that, although explanations help improve the understanding and interpretability of black-box models, they also leak essential information about the inner workings of the models. Therefore, the black-box models become more vulnerable to attacks, especially in the context of MLaaS where the prediction and its explanation are returned for each query. With a novel two-step LDP-preserving mechanism, we have proposed XRAND to protect the model explanations from being exploited by adversaries via obfuscating the top important features, while maintaining the faithfulness of explanations.

CHAPTER 5
LIFELONG DP: CONSISTENTLY BOUNDED DIFFERENTIAL PRIVACY IN LIFELONG MACHINE LEARNING

5.1 Preamble

In this chapter, we show that the process of continually learning new tasks and memorizing previous tasks introduces unknown privacy risks and challenges to bound the privacy loss. Based upon this, we introduce a formal definition of **Lifelong DP**, in which the participation of any data tuples in the training set of any tasks is protected, under a consistently bounded DP protection, given a growing stream of tasks. A consistently bounded DP means having only one fixed value of the DP privacy budget, regardless of the number of tasks. To preserve Lifelong DP, we propose a scalable and heterogeneous algorithm, called **L2DP-ML** with a streaming batch training, to efficiently train and continue releasing new versions of an L2M model, given the heterogeneity in terms of data sizes and the training order of tasks, without affecting DP protection of the private training set. An end-to-end theoretical analysis and thorough evaluations show that our mechanism is significantly better than baseline approaches in preserving Lifelong DP.

5.2 Background

Let us first revisit L2M with A-gem and DP. In L2M, we learn a sequence of tasks $\mathbb{T} = \{t_1, \ldots, t_m\}$ one by one, such that the learning of each new task will not forget the models learned for the previous tasks. Let D_i be the dataset of the i-th task. Each tuple contains data $x \in [-1, 1]^d$ and a ground-truth label $y \in \mathbb{Z}_K$, which is a one-hot vector of K categorical outcomes $y = \{y_1, \ldots, y_K\}$. A single true class label $y_x \in y$ given x is assigned to only one of the K categories. All the training sets D_i are non-overlapping; that is, an arbitrary input (x, y) belongs to only one D_i, i.e., $\exists! i \in [1, m] : (x, y) \in D_i$ ($x \in D_i$ for simplicity). On input x and parameters θ, a model outputs class scores $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$ that map inputs x to a vector of scores $f(x) = \{f_1(x), \ldots, f_K(x)\}$ s.t. $\forall k \in [1, K] : f_k(x) \in [0, 1]$ and

$\sum_{k=1}^{K} f_k(x) = 1$. The class with the highest score is selected as the predicted label for x, denoted as $y(x) = \max_{k\in K} f_k(x)$. A loss function $L(f(\theta, x), y)$ presents the penalty for mismatching between the predicted values $f(\theta, x)$ and original values y.

Lifelong Learning. Given the current task τ ($\leq m$), let us denote $\mathbb{T}_\tau = \{t_1, \ldots, t_{\tau-1}\}$ is a set of tasks that have been learnt. Although there are different L2M settings, i.e., episodic memory [6, 61, 118, 161, 165, 167, 193] and generative memory [144, 180, 205], we leverage one of the state-of-the-art algorithms, i.e., A-gem [30], to demonstrate our privacy-preserving mechanism, without loss of the generality of our study. A-gem avoids catastrophic forgetting by storing an episodic memory M_i for each task $t_i \in \mathbb{T}_\tau$. When minimizing the loss on the current task τ, a typical approach is to treat the losses on the episodic memories of tasks $i < \tau$, given by $L(f(\theta, M_i)) = \frac{1}{|M_i|}\sum_{x\in M_i} L(f(\theta, x), y)$, as inequality constraints. In A-gem, the L2M objective function is:

$$\theta^\tau = \arg\min_\theta L\big(f(\theta, D_\tau)\big) \quad \text{s.t.} \quad L\big(f(\theta^\tau, \mathbb{M}_\tau)\big) \leq L\big(f(\theta^{\tau-1}, \mathbb{M}_\tau)\big) \tag{5.1}$$

where $\theta^{\tau-1}$ are the values of model parameters θ learned after training the task $t_{\tau-1}$, $\mathbb{M}_\tau = \cup_{i<\tau} M_i$ is the episodic memory with $\mathbb{M}_1 = \emptyset$, $L\big(f(\theta^{\tau-1}, \mathbb{M}_\tau)\big) = \sum_{i=1}^{\tau-1} L\big(f(\theta^{\tau-1}, M_i)\big)/(\tau - 1)$. Equation 5.1 indicates that learning θ^τ given the task τ will not forget previously learned tasks $\{t_1, \ldots, t_{\tau-1}\}$ enforced by the memory replaying constraint $L\big(f(\theta^\tau, \mathbb{M}_\tau)\big) \leq L\big(f(\theta^{\tau-1}, \mathbb{M}_\tau)\big)$.

At each training step, A-gem [30] has access to only D_τ and $\mathbb{M}_\tau$ to compute the *projected gradient* $\tilde{g}$ (i.e., by addressing the constraint in Equation 5.1), as follows:

$$\tilde{g} = g - \frac{g^\top g_{ref}}{g_{ref}^\top g_{ref}} g_{ref} \tag{5.2}$$

where g is the *updated gradient* computed on a batch sampled from D_τ, g_{ref} is an *episodic gradient* computed on a batch sampled from $\mathbb{M}_\tau$, and $\tilde{g}$ is used to update the model parameters θ in Equation 5.1.

Differential Privacy (DP). DP guarantees that the released statistical results, computed from the underlying sensitive data, is insensitive to the presence or absence of one tuple in a dataset. Let us briefly revisit the definition of DP, as:

Definition 7. *(ϵ, δ)-DP [58]. A randomized algorithm A is (ϵ, δ)-DP, if for any two neighboring databases D and D' differing at most one tuple, $\forall O \subseteq Range(A)$, we have:*

$$Pr[A(D) = O] \leq e^{\epsilon} Pr[A(D') = O] + \delta \tag{5.3}$$

where ϵ controls the amount by which the distributions induced by D and D' may differ, and δ is a broken probability. A smaller ϵ enforces a stronger privacy guarantee.

DP has been preserved in many ML models and tasks [5, 147, 152]. Technically, existing mechanisms have not been designed to preserve DP in L2M under a fixed and consistently bounded privacy budget given a growing stream of learning tasks. That differs from our goal in this study.

5.3 Privacy Risk and Lifelong DP

In this section, we focus on analyzing the unknown privacy risk in L2M and introduce a new concept of Lifelong DP.

Privacy Risk Analysis. One benefit of L2M is that end-users can use an L2M model after training each task τ, instead of waiting for the model to be trained on all the tasks. Thus, in practice, the adversary can observe the model parameters $\theta^1, \ldots, \theta^m$ after training each task $t_1, \ldots, t_m$. Note that the adversary does not observe any information about the (black-box) training algorithm. Another key property in an L2M model is the episodic memory, which is kept to be read at each training step incurring privacy leakage. Therefore, the training data D and episodic memory M need to be protected together across tasks. Finally, in L2M, at each training step for any task t_i $(i \in [1, m])$, we only have access to D_i and $\mathbb{M}_i$, without a complete view of the cumulative dataset of all the tasks $\cup_{i \in [1,m]} D_i$ and $\mathbb{M}_m = \cup_{i \in [1,m-1]} M_i$. This is different from the traditional definition of a database in both

DP (Def. 7) and in a model trained on a single task. To cope with this, we propose a new definition of lifelong neighboring databases, as follows:

Definition 8. *Lifelong Neighboring Databases. Given any two lifelong databases* $\mathsf{data}_m = \{\mathcal{D}, \mathcal{M}\}$ *and* $\mathsf{data}'_m = \{\mathcal{D}', \mathcal{M}'\}$*, where* $\mathcal{D} = \{D_1, \dots, D_m\}$*,* $\mathcal{D}' = \{D'_1, \dots, D'_m\}$*,* $\mathcal{M} = \{\mathbb{M}_1, \dots, \mathbb{M}_m\}$*,* $\mathcal{M}' = \{\mathbb{M}'_1, \dots, \mathbb{M}'_m\}$*,* $\mathbb{M}_i = \cup_{j \in [1, i-1]} M_j$*, and* $\mathbb{M}'_i = \cup_{j \in [1, i-1]} M'_j$*.* data_m *and* data'_m *are called lifelong neighboring databases if,* $\forall i \in [1, m]$*: **(1)*** D_i *and* D'_i *differ at most one tuple; and **(2)*** M_i *and* M'_i *differ at most one tuple.*

A Naive Mechanism. To preserve DP in L2M, one can employ the moments accountant [4] to train the model f by injecting Gaussian noise into clipped gradients g and g_{ref} (Equation 5.2), with privacy budgets ϵ_{D_τ} and $\epsilon_{\mathbb{M}_\tau}$ on each dataset D_τ and on the episodic memory $\mathbb{M}_\tau$, and a gradient clipping bound C. The post-processing property [59] can be applied to guarantee that $\tilde{g}$, computed from the perturbed g and g_{ref}, is also DP.

Let us denote this mechanism as A, and denote A_τ as A applied on the task τ. A naive approach [47] is to repeatedly apply A on the sequence of tasks $\mathbb{T}$. Since training data is non-overlapping among tasks, the parallel composition property in DP [56] can be applied to estimate the total privacy budget consumed across all the tasks, as follows:

$$Pr[A(\mathsf{data}_m) = \{\theta^i\}_{i \in [1,m]}] \leq e^\epsilon Pr[A(\mathsf{data}'_m) = \{\theta^i\}_{i \in [1,m]}] + \delta \tag{5.4}$$

where $\epsilon = \max_{i \in [1,m]}(\epsilon_{D_i} + \epsilon_{\mathbb{M}_i})$, and $\forall i, j \in [1, m] : \delta$ is the same for ϵ_{D_i} and $\epsilon_{\mathbb{M}_j}$.

$A(\mathsf{data}_m)$ indicates that the model is trained from scratch with the mechanism A, given randomly initiated parameters θ^0, i.e., $A(\theta^0, \mathsf{data}_m)$. Intuitively, we can achieve the traditional DP guarantee in L2M, as the participation of a particular data tuple in each dataset D_τ is protected under the released (ϵ, δ)-DP $\{\theta^i\}_{i \in [1,m]}$. In principle, this approach introduces unknown privacy risks in each task and in the whole training process, as discussed next.

Observing the intermediate parameters $\{\theta^i\}_{i < \tau}$ turns the mechanism A_τ into a list of adaptive DP mechanisms $A_1, \dots, A_\tau$ sequentially applied on tasks $t_1, \dots, t_\tau$, where

$A_i : (\prod_{j=1}^{i-1} \mathcal{R}_j) \times D_i \rightarrow \mathcal{R}_i$. This is an instance of adaptive composition, which we can model by using the output of all the previous mechanisms $\{\theta^i\}_{i<\tau}$ as the auxiliary input of the A_τ mechanism. Thus, given an outcome θ^τ, the privacy loss $c(\cdot)$ at θ^τ is as follows:

$$c(\theta^\tau; A_\tau, \{\theta^i\}_{i<\tau}, \mathsf{data}_\tau, \mathsf{data}'_\tau) = \log \frac{Pr[A_\tau(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau]}{Pr[A_\tau(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau]} \tag{5.5}$$

The privacy loss is accumulated across tasks, as follows:

Theorem 9. $\forall \tau > 1 : c(\theta^\tau; A_\tau, \{\theta^i\}_{i<\tau}, \mathsf{data}_\tau, \mathsf{data}'_\tau) = \sum_{i=1}^{\tau} c(\theta^i; A_i, \{\theta^j\}_{j<i}, \mathsf{data}_i, \mathsf{data}'_i)$.

As a result of the Theorem 9, the privacy budget at each task τ cannot be simply bounded by $\max_{\tau \in [1,m]}(\epsilon_{D_\tau} + \epsilon_{\mathbb{M}_\tau})$, given δ (Equation 5.4). This problem might be addressed by replacing the $\max$ function in Equation 5.4 with a summation function: $\epsilon = \sum_{\tau \in [1,m]}(\epsilon_{D_\tau} + \epsilon_{\mathbb{M}_\tau})$, to compute the upper bound of the privacy budget for an entire of the continual learning process. To optimize this naive approach, one can adapt the management policy [109] to redistribute the privacy budget across tasks while limiting the total privacy budget ϵ to be smaller than a predefined upper bound, that is, the training will be terminated when ϵ reaches the predefined upper bound.

Technically, the challenge in bounding the privacy risk is still the same, centering around the growing number of tasks m and the heterogeneity among tasks: **(1)** The larger the number of tasks, the larger the privacy budget will be consumed by the $\sum$ function. It is hard to identify an upper bound privacy budget given an unlimited number of streaming tasks in L2M; **(2)** Different tasks may require different numbers of training steps due to the difference in terms of the number of tuples in each task; thus, affecting the privacy budget ϵ; and **(3)** The order of training tasks also affect the privacy budget, since computing g_{ref} by using data in the episodic memory from one task may be more than other tasks. Therefore, bounding the DP budget in L2M is non-trivial.

Lifelong DP. To address these challenges, we propose a new definition of ϵ-Lifelong DP to guarantee that an adversary cannot infer whether a data tuple is in the lifelong training dataset data_m, given the released parameters $\{\theta^i\}_{i\in[1,m]}$ learned from a growing stream of an infinite number of new tasks, denoted $\forall m \in [1,\infty)$, under a consistently bounded DP budget ϵ (Equation 5.6). A consistently bounded DP means having only one fixed value of ϵ, regardless of the number of tasks m. In other words, it does not exist an $i \leq m$ and an $\epsilon' < \epsilon$, such that releasing $\{\theta^j\}_{j\in[1,i]}$ given training dataset data_i is ϵ'-DP (Equation 5.7). A consistently bounded DP is significant by enabling us to keep training and releasing an L2M model without intensifying the end-to-end privacy budget consumption. Lifelong DP can be formulated as follows:

Definition 9. *ϵ-Lifelong DP. Given a lifelong database* data_m*, a randomized algorithm A achieves ϵ-Lifelong DP, if for any of two lifelong neighboring databases* $(\mathsf{data}_m, \mathsf{data}'_m)$*, for all possible outputs $\{\theta^i\}_{i\in[1,m]} \in Range(A)$, $\forall m \in [1,\infty)$ we have that*

$$P\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big] \leq e^{\epsilon} P\big[A(\mathsf{data}'_m) = \{\theta^i\}_{i\in[1,m]}\big] \tag{5.6}$$

$$\nexists(\epsilon' < \epsilon, i \leq m) : P\big[A(\mathsf{data}_i) = \{\theta^j\}_{j\in[1,i]}\big] \leq e^{\epsilon'} P\big[A(\mathsf{data}'_i) = \{\theta^j\}_{j\in[1,i]}\big] \tag{5.7}$$

where $Range(A)$ denotes every possible output of A.

In our Lifelong DP definition, the episodic memory (data) $\mathcal{M}$ can be an empty set $\emptyset$ in the definition of lifelong neighboring databases (Def. 8) given L2M mechanisms that do not need to access $\mathcal{M}$ [80, 125, 159, 215, 217].

To preserve Lifelong DP, we need to address the following problems: **(1)** The privacy loss accumulation across tasks; **(2)** The overlapping between the episodic memory $\mathcal{M}$ and the training data $\mathcal{D}$; and **(3)** The data sampling process for computing the episodic gradient g_{ref} given the growing episodic memory $\mathcal{M}$. The root cause issue of these problems is that in an L2M model, the episodic memory $\mathcal{M}$, which accumulatively stores data from all of the previous tasks, is read at each training step. Thus, using the moments account to preserve Lifelong DP will cause the privacy budget accumulated, resulting in a loose privacy protection given a large number of tasks or training steps. Therefore, designing a mechanism to preserve Lifelong DP under a tight privacy budget is non-trivial.

Algorithm 3 L2DP-ML Algorithm

Input: $\epsilon_1, \epsilon_2, \mathbb{T}=\{t_i\}_{i\in[1,m]}, \{D_i\}_{i\in[1,m]}$

Output: $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-Lifelong DP parameters $\{\theta^i\}_{i\in[1,m]} = \{\theta^i_1, \theta^i_2\}_{i\in[1,m]}$

```
1:  **Draw Noise** $\chi_1 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})]^d$, $\chi_2 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})]^\beta$, $\chi_3 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{L}}}}{\epsilon_2})]^{|\mathbf{h}_\pi|}$
2:  **Randomly Initialize:** $\theta^0 = \{\theta^0_1, \theta^0_2\}$, $\mathbb{M}_1 = \emptyset$, $\forall \tau \in \mathbb{T} : \overline{D}_\tau = \{\overline{x}_r \leftarrow x_r + \frac{\chi_1}{|D_\tau|}\}_{x_r \in D_\tau}$,
    hidden layers $\{\mathbf{h}_1 + \frac{2\chi_2}{|D_\tau|}, \dots, \mathbf{h}_\pi\}$
3:  **for** $\tau \in [1, m]$ **do**
4:      **if** $\tau == 1$ **then**
5:          $g \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta^{\tau-1}_1), \nabla_{\theta_2}\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta^{\tau-1}_2)\}$ with the noise $\frac{\chi_3}{|D_\tau|}$
6:      **else**
7:          $\mathbb{M}_\tau \leftarrow \mathbb{M}_{\tau-1} \cup \{\overline{D}_{\tau-1}\}$
8:          **Randomly Pick** a dataset $\overline{D}_{ref} \in \mathbb{M}_\tau$
9:          $g \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta^{\tau-1}_1), \nabla_{\theta_2}\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta^{\tau-1}_2)\}$ with the noise $\frac{\chi_3}{|D_\tau|}$
10:         $g_{ref} \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_{\overline{D}_{ref}}(\theta^{\tau-1}_1), \nabla_{\theta_2}\overline{\mathcal{L}}_{\overline{D}_{ref}}(\theta^{\tau-1}_2)\}$ with the noise $\frac{\chi_3}{|D_{ref}|}$
11:         $\tilde{g} \leftarrow g - \frac{g^\top g_{ref}}{g^\top_{ref} g_{ref}} g_{ref}$
12:     **end if**
13:     **Descent:** $\{\theta^\tau_1, \theta^\tau_2\} \leftarrow \{\theta^{\tau-1}_1, \theta^{\tau-1}_2\} - \varrho\tilde{g}$ # learning rate $\varrho$
14:     **Release:** $\{\theta^\tau_1, \theta^\tau_2\}$
15: **end for**
```

5.4 Preserving Lifelong DP

To overcome the aforementioned issues, our idea is designing a L2M mechanism such that the privacy budget will not accumulate across training steps while memorizing previously learned tasks. More precisely, we design our network as a multi-layer neural network

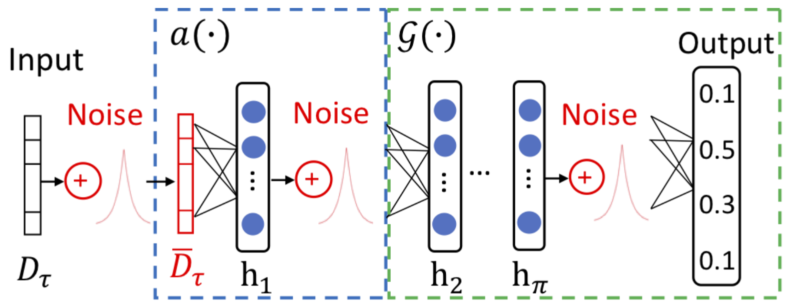


Figure 5.1 Network design of L2DP-ML.

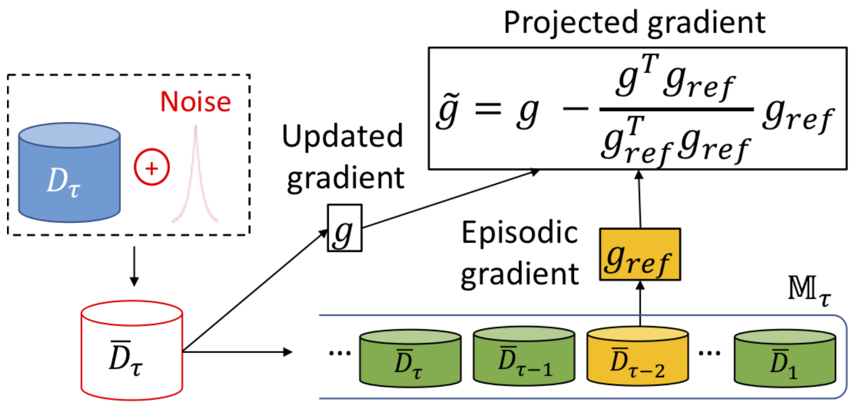


Figure 5.2 Gradient update in L2DP-ML.

stacked on top of a feature representation learning model. Then, we propose a new Laplace mechanism-based Lifelong DP algorithm, called L2DP-ML (Alg. 3), in computing the gradients g, g_{ref}, and $\tilde{g}$. Finally, to overcome expensive computation cost and heterogeneity among tasks, we develop a scalable and heterogeneous algorithm through a streaming batch training (Alg. 7), to efficiently learn Lifelong DP parameters (Theorem 10).

Network Design. In our Alg. 3 and Figure 5.1, a DNN is designed as a stack of an auto-encoder for feature representation learning and a typical multi-layer neural network, as follows: $f(x) = \mathcal{G}(a(x, \theta_1), \theta_2)$ where $a(\cdot)$ is the auto-encoder and $\mathcal{G}(\cdot)$ is the multi-layer neural network. The auto-encoder $a(\cdot)$ takes x as an input with model parameters θ_1; meanwhile, the multi-layer neural network $\mathcal{G}(\cdot)$ takes the output of the auto-encoder $a(\cdot)$ as its input with model parameters θ_2 and returns the class scores $f(x)$.

This network design allows us to: **(1)** Tighten the sensitivity of our model, since it is easy to train the auto-encoder using less sensitive objective functions, given its small sizes; **(2)** Reduce the privacy budget consumption, since the computations of the multi-layer neural network is DP when the output of the auto-encoder is DP; and **(3)** Provide a better re-usability, given that the auto-encoder can be reused for different predictive models.

Given a dataset D_τ, the objective functions of the auto-encoder and the multi-layer neural network can be the classical cross-entropy error functions for data reconstruction at the input layer and for classification at the output layer, denoted $\mathcal{R}_{D_\tau}(\theta_1)$ and $\mathcal{L}_{D_\tau}(\theta_2)$ respectively. Without loss of generality, we define the data reconstruction function $\mathcal{R}_{D_\tau}(\theta_1)$ and the classification function $\mathcal{L}_{D_\tau}(\theta_2)$ as follows:

$$\mathcal{R}_{D_\tau}(\theta_1) = \sum_{x_r \in D_\tau} \sum_{s=1}^{d} \Big[x_{rs} \log(1 + e^{-\theta_{1s} h_r}) \Big] + \sum_{x_r \in D_\tau} \sum_{s=1}^{d} \Big[(1 - x_{rs}) \log(1 + e^{\theta_{1s} h_r}) \Big] \tag{5.8}$$

$$\mathcal{L}_{D_\tau}(\theta_2) = - \sum_{x_r \in D_\tau} \sum_{k=1}^{K} \Big[y_{rk} \log(1 + e^{-\mathbf{h}_{\pi r} W_{\pi k}^T}) + (1 - y_{rk}) \log(1 + e^{\mathbf{h}_{\pi r} W_{\pi k}^T}) \Big] \tag{5.9}$$

where the transformation of x_r is $h_r = \theta_1^\top x_r$, the hidden layer $\mathbf{h}_1$ of $a(x, \theta_1)$ given D_τ is $\mathbf{h}_{1D_\tau} = \{\theta_1^\top x_r\}_{x_r \in D_\tau}$, $\widetilde{x}_r = \theta_1 h_r$ is the reconstruction of x_r, and $\mathbf{h}_{\pi r}$ computed from the x_r through the network with W_π is the parameter at the last hidden layer $\mathbf{h}_\pi$.

Our L2M objective function is defined as:

$$\begin{aligned} &\{\theta_1^\tau, \theta_2^\tau\} = \arg\min_{\theta_1, \theta_2} [\mathcal{R}_{D_\tau}(\theta_1) + \mathcal{L}_{D_\tau}(\theta_2)] \\ &\text{s.t.} \quad \mathcal{R}_{\mathbb{M}_\tau}(\theta_1^\tau) \leq \mathcal{R}_{\mathbb{M}_\tau}(\theta_1^{\tau-1}) \text{ and } \mathcal{L}_{\mathbb{M}_\tau}(\theta_2^\tau) \leq \mathcal{L}_{\mathbb{M}_\tau}(\theta_2^{\tau-1}) \end{aligned} \tag{5.10}$$

where $\{\theta_1, \theta_2\}$ are the model parameters; while, $\{\theta_1^\tau, \theta_2^\tau\}$ are the values of $\{\theta_1, \theta_2\}$ after learning task τ.

At each training step on the current task τ, to update the model parameters $\{\theta_1^\tau, \theta_2^\tau\}$ minimizing Equation 5.10, we need to compute the gradients g and g_{ref}, and then follow Equation 5.2 to compute the projected gradient $\tilde{g}$ for the model parameters $\{\theta_1^\tau, \theta_2^\tau\}$ (Figure

5.2). Given the projected $\tilde{g}$, we can update $\{\theta_1^\tau, \theta_2^\tau\}$ by applying typical descent operation, as follows.

Gradient Update g. To compute the gradient g for $\{\theta_1^\tau, \theta_2^\tau\}$ on the current task τ, we first derive polynomial forms of $\mathcal{R}_{D_\tau}(\theta_1)$ and $\mathcal{L}_{D_\tau}(\theta_2)$, by applying the 1st and 2nd orders of Taylor Expansion as follows:

$$\widetilde{\mathcal{R}}_{D_\tau}(\theta_1) = \sum_{x_r \in D_\tau} \sum_{s=1}^{d} \Big[\theta_{1s}\big(\frac{1}{2} - x_{rs}\big)h_r\Big] \tag{5.11}$$

$$\widetilde{\mathcal{L}}_{D_\tau}\big(\theta_2\big) = \sum_{k=1}^{K} \sum_{x_r \in D_\tau} \Big[\mathbf{h}_{\pi r} W_{\pi k} - (\mathbf{h}_{\pi r} W_{\pi k}) y_{rk}\Big] - \sum_{k=1}^{K} \sum_{x_r \in D_\tau} \Big[\frac{1}{2}|\mathbf{h}_{\pi r} W_{\pi k}| + \frac{1}{8}(\mathbf{h}_{\pi r} W_{\pi k})^2\Big] \tag{5.12}$$

To preserve ϵ_1-DP in learning θ_1, we leverage Functional Mechanism [222] to inject a Laplace noise into polynomial coefficients of the function $\widetilde{\mathcal{R}}_{D_\tau}(\theta_1)$, which are the input x and the first transformation $\mathbf{h}_1$. Laplace mechanism [59] is well-known in perturbing objective functions to prevent privacy budget accumulation in training ML models [151, 154, 155]. As in [151], the global sensitivity $\Delta_{\widetilde{\mathcal{R}}}$ is bounded as follows: $\Delta_{\widetilde{\mathcal{R}}} \leq d(|\mathbf{h}_1| + 2)$, with $|\mathbf{h}_1|$ is the number of neurons in $\mathbf{h}_1$. The perturbed $\widetilde{\mathcal{R}}$ function becomes:

$$\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1) = \sum_{\overline{x}_r \in \overline{D}_\tau} \Big[\sum_{s=1}^{d} \big(\frac{1}{2}\theta_{1s}\overline{h}_r\big) - \overline{x}_r \widetilde{x}_r\Big] \tag{5.13}$$

where $\overline{x}_r = x_r + \frac{1}{|\mathcal{D}_\tau|} Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1}), h_r = \theta_1^\top \overline{x}_r, \overline{h}_r = h_r + \frac{2}{|\mathcal{D}_\tau|} Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})$, $\widetilde{x}_r = \theta_1 \overline{h}_r$, h_r is clipped to $[-1, 1]$, and ϵ_1 is a privacy budget.

Importantly, the perturbation of each example x turns the original data D_τ into a $(\epsilon_1/\gamma_{\mathbf{x}})$-DP dataset $\overline{D}_\tau = \{\overline{x}_r\}_{x_r \in D_\tau}$ with $\gamma_{\mathbf{x}} = \Delta_{\widetilde{\mathcal{R}}}/|D_\tau|$ by following Lemma 2 in [151] (Alg. 3, line 2). Based upon that, all the computations on top of the $(\epsilon_1/\gamma_{\mathbf{x}})$-DP dataset $\overline{D}_\tau$, including h_r, $\overline{h}_r$, $\widetilde{x}_r$, and the computation of gradients g of the model parameters θ_1 are shown to be $(\epsilon_1/\gamma_{\mathbf{x}})$-DP without accessing any additional information from the original data D_τ, i.e., $\forall s \in [1, d] : \nabla_{\theta_{1s}} \overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1) = \frac{\delta \overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1)}{\delta \theta_{1s}} = \sum_{r=1}^{|\mathcal{D}_\tau|} \overline{h}_r(\frac{1}{2} - \overline{x}_{rs})$. This follows the post-processing property of DP [59]. Consequently, the total privacy budget used to perturb $\widetilde{\mathcal{R}}$ is $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}})$, by having $\frac{Pr\left(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1)\right)}{Pr\left(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1)\right)} \times \frac{Pr\left(\overline{D}_\tau\right)}{Pr\left(\overline{D}'_\tau\right)} \leq (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}})$. Details are available in our proof of Theorem 10, Appendix A.27.

A similar approach is applied to perturb the objective function $\widetilde{\mathcal{L}}_{D_\tau}(\theta_2)$ at the output layer with a privacy budget ϵ_2. The perturbed function of $\widetilde{\mathcal{L}}$ is denoted as $\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2)$. As in Lemma 3 [151], the output of the auto-encoder, which is the perturbed transformation $\overline{\mathbf{h}}_{1\overline{D}_\tau} = \{\overline{\theta}_1^\top \overline{x}_r + \frac{2}{|\mathcal{D}_\tau|} Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})\}_{\overline{x}_r \in \overline{D}_\tau}$, is (ϵ_1/γ)-DP, given $\gamma = \frac{2\Delta_{\widetilde{\mathcal{R}}}}{|\mathcal{D}_\tau| \|\overline{\theta}_1\|_{1,1}}$ and $\|\overline{\theta}_1\|_{1,1}$ is the maximum 1-norm of θ_1's columns[1]. As a result, the computations of all the hidden layers of the multi-layer neural network $\mathcal{G}(\cdot)$ that takes the output of the auto-encoder $\overline{\mathbf{h}}_{1\overline{D}_\tau}$ as its input, is (ϵ_1/γ)-DP, since $\overline{\mathbf{h}}_{1\overline{D}_\tau}$ is (ϵ_1/γ)-DP, following the post-processing property of DP [59] (Alg. 3, line 2).

That helps us to **(1)** avoid extra privacy budget consumption in computing the multi-layer neural network $\mathcal{G}(\cdot)$; **(2)** tighten the sensitivity of the function $\overline{\mathcal{L}}_{\overline{D}_\tau}$ (i.e., $\Delta_{\widetilde{\mathcal{L}}} \leq 2|\mathbf{h}_\pi|$); and **(3)** achieve DP gradient update for θ_2. The total privacy budget used to perturb $\widetilde{\mathcal{L}}$ is $(\epsilon_1/\gamma + \epsilon_2)$, i.e., $Pr\left(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2)\right)/Pr\left(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2)\right) \leq (\epsilon_1/\gamma + \epsilon_2)$. Consequently, the total privacy budget in computing the gradient updates g, i.e., $\{\nabla_{\theta_1}\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1}), \nabla_{\theta_2}\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})\}$, for the current task τ is $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-DP (Alg. 3, lines 5 and 10).

Episodic and Projected Gradients g_{ref} and $\tilde{g}$. Now, we are ready to present our approach in achieving Lifelong DP, by configuring the episodic memory at the current task τ (i.e., $\mathbb{M}_\tau$) as a *fixed* and *disjoint* set of datasets from previous tasks, i.e., $\mathbb{M}_\tau = \{\overline{D}_1, \ldots, \overline{D}_{\tau-1}\}$ (Alg. 3, line 7); such that, at each training step, the computation of episodic gradients g_{ref} for the model parameters $\{\theta_1, \theta_2\}$ using a randomly picked dataset $\overline{D}_{ref} \in \mathbb{M}_\tau$ (Alg. 3, lines 8 and 11), is $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-DP, without incurring any additional privacy budget consumption for the dataset D_{ref}. The *projected gradients* $\tilde{g}$ is computed from g and g_{ref} (Equation 5.2) is also $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-DP, following the post-processing property [59].

[1]`https://en.wikipedia.org/wiki/Operator_norm`

Hence, we reformulate the L2M objective function in Equation 5.10, as follows:

$$\begin{aligned}
&\{\theta_1^\tau, \theta_2^\tau\} = \arg\min_{\theta_1,\theta_2}[\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1) + \overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2)] \\
&\text{s.t.} \quad \overline{\mathcal{R}}_{\mathbb{M}_\tau}(\theta_1^\tau) \leq \overline{\mathcal{R}}_{\mathbb{M}_\tau}(\theta_1^{\tau-1}), \overline{\mathcal{L}}_{\mathbb{M}_\tau}(\theta_2^\tau) \leq \overline{\mathcal{L}}_{\mathbb{M}_\tau}(\theta_2^{\tau-1}) \\
&\text{where } \mathbb{M}_\tau = \{\overline{D}_1, \dots, \overline{D}_{\tau-1}\} \qquad (5.14)
\end{aligned}$$

By using the perturbed functions $\overline{\mathcal{R}}$ and $\overline{\mathcal{L}}$, the constrained optimization of Equation 5.14 can be addressed similarly to Equation 5.2, when the projected gradient $\tilde{g}$ is computed as: $\tilde{g} = g - (g^\top g_{ref})/(g_{ref}^\top g_{ref})g_{ref}$, where g is the gradient update on the current task τ, and g_{ref} is computed using a dataset $\overline{D}_{ref}$ randomly selected from the episodic memory $\mathbb{M}_\tau$.

Lifelong DP Guarantee. Given the aforementioned network $f(x)$ as the stack of the auto-encoder and the multi-layer neural network, and privacy budgets ϵ_1 and ϵ_2, the total Lifelong DP privacy consumption in learning the model parameters $\{\theta_1, \theta_2\}$ at each task is computed in Theorem 10.

Theorem 10. *Alg. 3 achieves* $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$*-Lifelong DP in learning* $\{\theta_1^i, \theta_2^i\}_{i\in[1,m]}$.

Theorem 10 shows that Alg. 3 achieves ϵ-Lifelong DP in learning the model parameters at each task $\{\theta_1^i, \theta_2^i\}_{i\in[1,m]}$, where $\epsilon = (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$. There are three key properties in the proof of Theorem 10 (Appendix A.27):

(1) For every input x in the whole training set $\overline{\mathcal{D}} = \{\overline{D}_i\}_{i\in[1,m]}$, x is included in *one and only one* dataset, denoted $\overline{D}_x \in \overline{\mathcal{D}}$ (Equation A.93). Hence, the DP guarantee to x in $\overline{\mathcal{D}}$ is equivalent to the DP guarantee to x in $\overline{D}_x$ (Equations A.95 and A.96).

(2) If we randomly sample tuples from the episodic memory to compute the episodic gradients g_{ref}, the sampling set and its neighboring set can have at most $i - 1$ different tuples ($i \in [1, m]$), since each $\overline{D}_i$ and its neighboring dataset $\overline{D}_i'$ can have at most 1 different tuple. In addition, a random sampling set of tuples in the episodic memory can overlap with more than one datasets $\overline{D}_i$, which is used to compute the gradient g. Importantly, different sampling sets from the episodic memory can overlap each other; thus, a simple data tuple potentially is used in multiple DP-preserving objective functions using these overlapping sets to compute the episodic gradients g_{ref}. These issues introduce additional privacy risk by following the group privacy theory and overlapping datasets in DP. We address this problem, by having the episodic memory as a *fixed* and *disjoint* set of datasets across $\mathbb{T}$ training tasks (Equation A.94). As a result, we can prevent the additional privacy leakage, caused by: **(i)** Differing at most $i-1$ tuples between neighboring $\mathbb{M}_i$ and $\mathbb{M}'_i$ for all $i \in (1, m]$; and **(ii)** Generating new and overlapping sets of data samples for computing the episodic gradient (which are considered overlapping datasets in the parlance of DP) in the typical training. Thus, the optimization on one task does not affect the DP protection of any other tasks, even the objective function given one task can be different from the objective function given other tasks (Equation A.97).

(3) Together with the results achieved in (1) and (2), by having one and only one privacy budget for every task, we can achieve Equations 5.6 and 5.7 in Lifelong DP (Def. 9). We present these steps in Equations A.103 and A.105.

5.5 Scalable and Heterogeneous Training

Although computing the gradients given the whole dataset $\overline{D}_\tau$ achieves Lifelong DP, it has some shortcomings: **(1)** consumes a large computational memory to store the episodic memory; **(2)** computational efficiency is low, since we need to use the whole dataset $\overline{D}_\tau$ and $\overline{D}_{ref}$ to compute the gradient update and the episodic gradient at each step; This results in a slow convergence speed and poor utility.

Scalability. To address this, we propose a streaming batch training (Alg. 7, Appendix A.28), in which a batch of data is used to train the model at each training step, by the following steps.

(1) Slitting the private training data $\overline{D}_\tau$ into disjoint and fixed batches (Alg. 7, line 4).

(2) Using a single draw of Laplace noise across batches (Alg. 7, lines 1-2). That prevents additional privacy leakage, caused by: (i) Generating multiple draws of noise (i.e., equivalent to applying one DP-preserving mechanism multiple times on the same dataset); (ii) Generating new and overlapping batches (which are considered overlapping datasets in the parlance of DP); and (iii)For any example x, x is included in *only one* batch. Hence, each *disjoint batch* of data in Alg. 7 can be considered as a *separate dataset* in Alg. 3.

(3) For each task, we randomly select a batch to place in the episodic memory (Alg. 7, line 17).

(4) At each training step, a batch from the current task is used to compute the gradient g, and a batch randomly selected from the episodic memory is used to compute the episodic gradient g_{ref} (Alg. 7, lines 11-14). Thus, Alg. 7 still preserves $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-Lifelong DP (Theorem 10).

By doing so, we significantly reduce the computational complexity and memory consumption, since only a small batch of data is stored in the episodic memory.

Heterogeneity. Based upon this, our algorithm can be applied to address the heterogeneity in terms of data sizes among tasks, which differs from multi-modal tasks [114]. We can train one task with multiple epochs, without affecting the Lifelong DP protection in Alg. 7, by 1) keeping all the batches fixed among epochs, and 2) at the end of training each task, we randomly select a batch of that task to place in the episodic memory. The order of the task does not affect the Lifelong DP, since the privacy budget is not accumulated across tasks. These properties enable us to customize our training, by having different numbers of training epochs for different tasks and having different training orders of tasks. Tasks with *smaller numbers of data tuples* can have *larger numbers of training epochs.* This helps us to achieve better model utility under the same privacy protection as shown in our experiments.

5.6 Experiments

Our validation focuses on understanding the impacts of the privacy budget ϵ and the heterogeneity on model utility. For reproducibility, our implementation is available.

Baseline Approaches. We consider **A-gem** [30] as an upper bound in terms of model performance, since A-gem is a noiseless model. We aim to show how much model utility is compromised for the Lifelong DP protection. Also, we consider the naive algorithm [47], called **NaiveGaussian**, as a baseline to demonstrate the effectiveness of our L2DP-ML mechanism. It is worth noting that there is a lack of a precise definition of adjacent databases resulting in an unclear or not well-justified DP protection for L2M in existing works [67,150]. Therefore, we do not consider them as baselines in our experiments.

To evaluate the heterogeneity, we further derive several versions of our algorithm (Alg. 7), including: **(1) Balanced L2DP-ML**, in which all the tasks have the same number of training steps, given a fixed batch size. This is also true for a **Balanced A-gem** algorithm; **(2) L2DP-ML** with the same number of epochs for all the tasks; and **(3) Heterogeneous L2DP-ML**, in which a fixed number of training epochs is assigned to each task. The numbers of epochs among tasks can be different. For instance, 5 epochs are used to train tasks with 5Hz, 10Hz, and 20Hz data, and 1 epoch is used to train the task with a larger volume of 50Hz data. The number of epochs is identified by the data size of each task, since the search space of the number of epochs for each task is exponentially large.

Datasets. We evaluate our approach using permuted and split MNIST [101], permuted and split CIFAR-10 [91], split CIFAR-100 datasets[2], and our human activity recognition in the wild (HARW) dataset. Permuted MNIST is a variant of MNIST [107] dataset, where each task has a random permutation of the input pixels, which is applied to all the images of that task. We adopt this approach to permute the CIFAR-10 dataset, including the input pixels and three color channels. Our HARW dataset was collected from 116 users, each of whom provided mobile sensor data and labels for their activities on Android phones consecutively in three months. HARW is an ultimate task for L2M, since different sensor sampling rates, e.g., 50Hz, 20Hz, 10Hz, and 5Hz, from different mobile devices are considered as L2M tasks. The classification output includes five classes of human activities, i.e., walking, sitting,

[2]Datasets were downloaded and evaluated by Phung Lai, Han Hu, and NhatHai Phan.

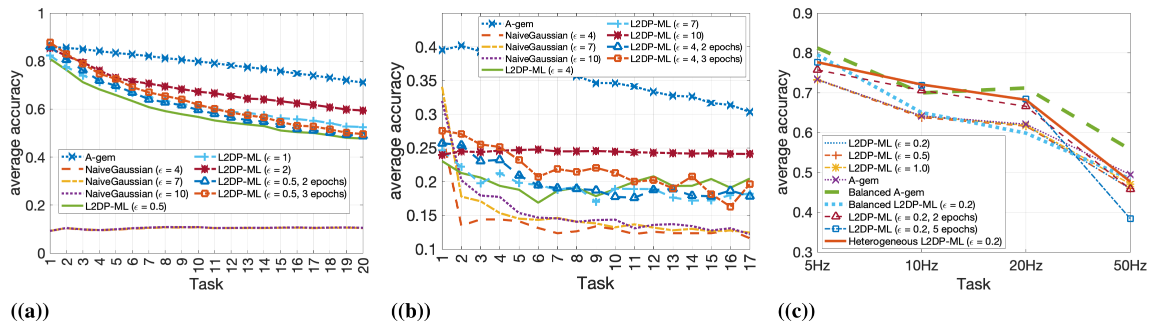


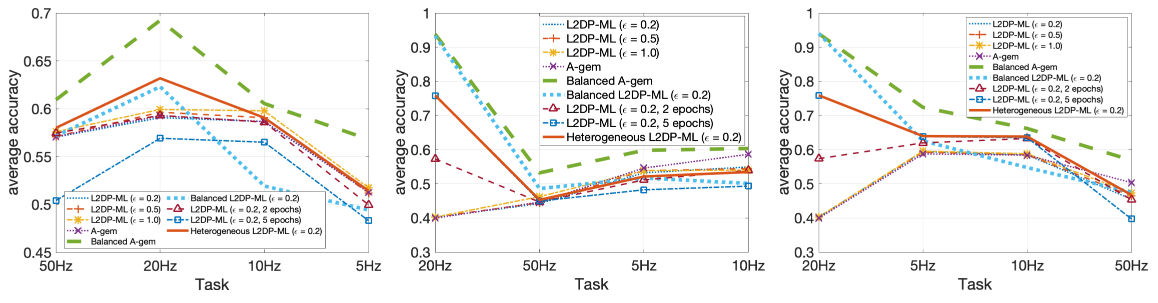
Figure 5.3 Average accuracy in the a) Permuted MNIST (20 tasks), b) Permuted CIFAR-10 (17 tasks), and c) HARW.

Table 5.1 Average Forgetting Measure

		L2DP-ML	NaiveGaussian	A-gem
Permuted MNIST	$\epsilon = 0.5$	0.305 ± 0.00886	0.012 ± 0.00271	0.162 ± 0.01096
	$\epsilon = 1$	0.278 ± 0.00907	0.015 ± 0.00457	
	$\epsilon = 2$	0.237 ± 0.00586	0.017 ± 0.00385	
Permuted CIFAR-10	$\epsilon = 4$	0.033 ± 0.00896	0.138 ± 0.00582	0.133 ± 0.00859
	$\epsilon = 7$	0.062 ± 0.01508	0.174 ± 0.01149	
	$\epsilon = 10$	0.034 ± 0.00184	0.181 ± 0.01956	
		L2DP-ML	Balanced L2DP-ML ($\epsilon = 0.2$)	A-gem
HARW (5Hz - 10Hz - 20Hz - 50Hz)	$\epsilon = 0.2$	0.1133 ± 0.0003	0.1309 ± 0.002	0.1269 ± 0.00045
	$\epsilon = 0.2$ (2 epochs)	0.1639 ± 0.00074		
	$\epsilon = 0.2$ (5 epochs)	0.2031 ± 0.0013		
	$\epsilon = 0.5$	0.1124 ± 0.00029	Heterogeneous L2DP-ML ($\epsilon = 0.2$)	Balanced A-gem
	$\epsilon = 1$	0.1106 ± 0.00026	0.1920 ± 0.00034	0.1593 ± 0.00021

in car, cycling, and running. The data collection and processing of our HARW dataset is in Appendix A.29. The setting of split CIFAR-10 and CIFAR-100, and split MNIST datasets are in Appendix A.30.

Model Configuration. In the permuted MNIST dataset, we used three convolutional layers (32, 64, and 96 features). In the permuted CIFAR-10 dataset, we used a Resnet-18 network (64, 64, 128, 128, and 160 features) with kernels (4, 3, 3, 3, and 3). In the HARW dataset, we used three convolutional layers (32, 64, and 96 features). Detailed model configurations are in the Appendix A.30. To conduct a fair comparison, we applied a



((a)) ((b)) ((c))

Figure 5.4 Average accuracy with random task orders: a) HARW 50Hz - 20Hz - 10Hz - 5Hz, b) HARW 20Hz - 50Hz - 5Hz - 10Hz, and c) HARW 20Hz - 5Hz - 10Hz - 50Hz.

grid-search for the best values of hyper-parameters, including the privacy budget $\epsilon \in [4, 10]$, the noise scale $z \in [1.1, 2.5]$, and the clipping bound $C \in [0.01, 1]$, in the NaiveGaussian mechanism. Based on the results of our hyper-parameter grid-search (Table A.6), we set $z = 2.2$ for $\epsilon = 4.0$, $z = 1.7$ for $\epsilon = 7.0$, and $z = 1.4$ for $\epsilon = 10.0$, and $C = 0.01$ is used for all values of ϵ.

Evaluation Metrics. We employ the well-applied average accuracy and forgetting measures after the model has been trained with all the batches up to task τ [29, 30], defined as follows: **(1)** *average accuracy*$_\tau = \frac{1}{\tau}\sum_{t=1}^{\tau} a_{\tau,n,t}$, where $a_{\tau,n,t} \in [0, 1]$ is the accuracy evaluated on the test set of task t, after the model has been trained with the n^{th} batch of task τ, and the training dataset of each task, D_τ, consists of a total n batches; **(2)** *average forgetting*$_\tau = \frac{1}{\tau-1}\sum_{t=1}^{\tau-1} f_t^\tau$, where f_t^τ is the forgetting on task t after the model is trained with all the batches up till task τ. f_t^τ is computed as follows: $f_t^\tau = \max_{l\in\{1,\dots,\tau-1\}}(a_{l,n,t} - a_{\tau,n,t})$; and **(3)** We measure the significant difference between two average accuracy curves induced by models A and B after task τ, using a p value (2-tail t-tests) curve: $p\ value = \big(\{\frac{1}{i}\sum_{t=1}^{i} a_{i,n,t}^{(A)}\}_{i\in[1,\tau]}, \{\frac{1}{i}\sum_{t=1}^{i} a_{i,n,t}^{(B)}\}_{i\in[1,\tau]}\big)$. All statistical tests are 2-tail t-tests.

Results in Permuted MNIST. Figure 5.3(a) and Table 5.1 illustrate the average accuracy and forgetting measure of each model as a function of the privacy budget ϵ on the permuted MNIST dataset. It is clear that the NaiveGaussian mechanism does not work

Table 5.2 Average Forgetting Measure on Random Orders of HARW Tasks

HARW (50Hz - 20Hz - 10Hz - 5Hz)	L2DP-ML ($\epsilon = 0.2$)	L2DP-ML ($\epsilon = 0.5$)	L2DP-ML ($\epsilon = 1$)
	0.1016 $\pm$ 0.0002	0.1012 $\pm$ 0.0001	0.098 $\pm$ 0.0001
	A-gem	Balanced A-gem	Balanced L2DP-ML ($\epsilon = 0.2$)
	0.1029 $\pm$ 0.0002	0.1241 $\pm$ 0.0002	0.1274 $\pm$ 0.0008
	L2DP-ML ($\epsilon = 0.2$, 2 epochs)	L2DP-ML ($\epsilon = 0.2$, 5 epochs)	Heterogeneous L2DP-ML ($\epsilon = 0.2$)
	0.1148 $\pm$ 0.0002	0.1012 $\pm$ 0.0014	0.1442 $\pm$ 0.0003

HARW (20Hz - 50Hz - 5Hz - 10Hz)	L2DP-ML ($\epsilon = 0.2$)	L2DP-ML ($\epsilon = 0.5$)	L2DP-ML ($\epsilon = 1$)
	0.0769 $\pm$ 2.07e-5	0.0761 $\pm$ 3.88e-5	0.0772 $\pm$ 6.7e-5
	A-gem	Balanced A-gem	Balanced L2DP-ML ($\epsilon = 0.2$)
	0.0781 $\pm$ 2.28e-5	0.14 $\pm$ 3.26e-4	0.1248 $\pm$ 0.0013
	L2DP-ML ($\epsilon = 0.2$, 2 epochs)	L2DP-ML ($\epsilon = 0.2$, 5 epochs)	Heterogeneous L2DP-ML ($\epsilon = 0.2$)
	0.0775 $\pm$ 8.45e-5	0.099 $\pm$ 0.0015	0.1268 $\pm$ 0.00028

well under a tight privacy budget $\epsilon \in [0.5, 2]$ given a large number of tasks $m = 20$. This is because each task can consume a tiny privacy budget ϵ/m resulting in either a large noise injected into the clipped gradients or a lack of training steps to achieve better model utility. By avoiding the privacy budget accumulation across tasks and training steps, our L2DP-ML models significantly outperform the NaiveGaussian mechanism. Our L2DP-ML model achieves 47.73% compared with 10.43% of the NaiveGaussian after 20 tasks given $\epsilon = 0.5$ ($p < 6.81e - 15$).

Regarding the upper bound performance, there is a small average accuracy gap between the noiseless A-gem model and our L2DP-ML models given a small number of tasks. The gap increases when the number of tasks increases (23.3% at $\epsilon = 0.5$ with 20 tasks). The larger the privacy budget (i.e., $\epsilon = 2.0$), the higher the average accuracy we can achieve, i.e., an improvement of 9.92% with $p < 2.83e - 14$, compared with smaller privacy budgets (i.e., $\epsilon = 0.5$). Also, our L2DP-ML models have a relatively good average forgetting with tight privacy protection ($\epsilon = 0.5, 1,$ and 2), compared with the noiseless A-gem model.

Results in Permuted CIFAR-10. Although permuted CIFAR-10 tasks are very difficult (Figure 5.3(b) and Table 5.1), even with the noiseless A-gem model, i.e., 35.24% accuracy on average, the results on the permuted CIFAR-10 further strengthen our observation. Our L2DP-ML models significantly outperform the NaiveGaussian mechanism. Our L2DP-ML model achieves an improvement of 8.84% in terms of average accuracy over the NaiveGaussian after 17 tasks given $\epsilon = 4$ ($p < 4.68e - 7$). We further observe that the NaiveGaussian mechanism has a remarkably larger average forgetting compared with our L2DP-ML (Table 5.1).

Interestingly, the gap between A-gem and our L2DP-ML models is notably shrunken when the number of tasks increases (from 16.47% with 1 task to 9.89% with 17 tasks, at $\epsilon = 4$). In addition, the average forgetting values in our L2DP-ML are better than the noiseless A-gem. This is a promising result. We also registered that the larger the privacy budget (i.e., $\epsilon = 10$), the higher the average accuracy that we can achieve, i.e., an improvement of 4.73% with $p < 1.15e - 9$, compared with smaller budgets (i.e., $\epsilon = 4$).

Heterogeneous Training. We now focus on shedding light into understanding the impacts of heterogeneity and privacy on model utility given different variants of our L2DP-ML mechanisms and the noiseless A-gem model. The p value curves are in Figures A.27 and A.28, Appendix A.30.

On the HARW task (Figure 5.3(c) and Table 5.1), our L2DP-ML model achieves a very competitive average accuracy, given a very tight DP budget $\epsilon = 0.2$ (i.e., 61.26%) compared with the noiseless A-gem model (i.e., 62.27%), across four tasks. Our model also achieves a better average forgetting, i.e., 11.33, compared with 12.69 of the noiseless A-gem model. That is promising. Increasing the privacy budget modestly increases the model performance. The differences in terms of average accuracy and forgetting are not significant. This is also true, when we randomly flip the order of the tasks (Figure 5.4 and Table 5.2). The results showed that our model effectively preserves Lifelong DP in HARW tasks.

Heterogeneous training, with customized numbers of epochs and task orders, further improves our model performance, under the same Lifelong DP protection. Figure A.27 illustrates the p values between the average accuracy curves of our L2DP-ML, given **1)** heterogeneous training with different numbers of epochs, **2)** task orders, and **3)** privacy budgets, over its basic settings, i.e., $\epsilon = 0.5$ for the permuted MNIST dataset, $\epsilon = 4$ for the permuted CIFAR-10 dataset, and $\epsilon = 0.2$ for the HARW dataset, with one training epoch.

• In the permuted MNIST dataset (Figures 5.3(a) and A.27(a)), when our L2DP-ML model is trained with 2 or 3 epochs per task, the average accuracy is improved, i.e., 2.81%, 4.8% given 2, 3 epochs, respectively, with $p < 8.44e - 9$. In the permuted CIFAR-10, using larger numbers of training epochs shows significant performance improvements over a small number of tasks (Figure A.27(b)). When the number of tasks becomes larger, the p values become less significant (even insignificant), compared with the p value curves of larger DP budgets (i.e., $\epsilon = 2$ and $\epsilon = 10$ in the permuted MNIST and permuted CIFAR-10). Meanwhile, training with a larger number of epochs yields better results with small numbers of tasks (i.e., fewer than 6 tasks), compared with larger DP budgets.

• In the HARW tasks, the improvement is more significant (Figures 5.3(c) and A.27(c)). Heterogeneous and Balanced L2DP-ML models outperform the basic settings with uniform numbers of training epochs, i.e., 1, 2, and 5 epochs. On average, we registered an improvement of 1.93% given the Balanced L2DP-ML and an improvement of 5.14% given the Heterogeneous L2DP-ML, over the basic setting (1 training epoch). The results are statistically significant (Figure A.27(c)). The average forgetting values of the Balanced L2DP-ML (0.1593) and the Heterogeneous L2DP-ML (0.1920) are higher than the basic setting (0.1133), with $p < 2.19e - 5$ (Table 5.1). This is expected as a primary trade-off in L2M, given a better average accuracy. In fact, the average forgetting values are also notably higher given larger uniform numbers of epochs, i.e, 2 and 5 epochs, and the Balanced A-gem. We do not address this fundamental issue in L2M since it is out-of-scope of this study. We focus on preserving Lifelong DP.

• We observe similar results in randomly flipping the order of the tasks (Figures 5.4 and A.28, Table 5.2). Among all task orders, our Heterogeneous L2DP-ML achieves the best average accuracy (66.4%) with the task order [5Hz, 10Hz, 20Hz, 50Hz] (Figure 5.3(c)) compared with the worse order [20Hz, 50Hz, 5Hz, 10Hz] (56.69%) (Figure 5.4(b)), i.e., $p < 9.9e - 5$. More importantly, in both average accuracy and forgetting, our Balanced and Heterogeneous L2DP-ML models achieve a competitive performance compared with the noiseless Balanced A-gem, which is considered to have the upper bound performance, and a better performance compared with having the uniform numbers of epochs across tasks. This obviously shown that the distinct ability to offer the heterogeneity in training across tasks greatly improves our model performance, under the same Lifelong DP protection.

Results in Split Tasks. We observe similar results on split CIFAR-10, CIFAR-100, and MNIST datasets as L2DP-ML achieves competitive average accuracy approaching the noiseless A-gem model under rigorous privacy budgets (Figure A.29, Appendix A.30). After 5 tasks of the split MNIST dataset, L2DP-ML achieves 73.54% and 81.83% in average accuracy at the privacy budgets 0.5 and 1 respectively, compared with 79.71% of the noiseless A-gem. Interestingly, our L2DP-ML has slightly higher average accuracy than the noiseless A-gem after 11 tasks of the split CIFAR-10 and CIFAR-100 dataset (14.83% in L2DP-ML at $\epsilon = 4$ compared with 13.44% in the noiseless A-gem). One reason is that Lifelong DP-preserving noise can help to mitigate the catastrophic forgetting. As showed in Table A.7 (Appendix A.30), our L2DP-ML obtains a significantly lower average forgetting (2.7% at $\epsilon = 4$) than the noiseless A-gem (19.5%).

5.7 Discussion

In this chapter, we showed that L2M introduces unknown privacy risk and challenges in preserving DP. To address this, we established a connection between DP preservation and L2M, through a new definition of Lifelong DP. To preserve Lifelong DP, we proposed the first scalable and heterogeneous mechanism, called L2DP-ML. Our model shows promising results in several tasks with different settings and opens a long-term avenue to achieve better model utility with lower computational cost, under Lifelong DP.

CHAPTER 6

ADDITIONAL WORK: ONTOLOGY-BASED INTERPRETABLE MACHINE LEARNING FOR TEXTUAL DATA

6.1 Preamble

In this chapter, we introduce a novel interpreting framework that learns an interpretable model based on an ontology-based sampling technique to explain agnostic prediction models. Different from existing approaches, our interpretable algorithm considers contextual correlation among words, described in domain knowledge ontologies, to generate semantic explanations. To narrow down the search space for explanations, which is exponentially large given long and complicated text data, we design a learnable anchor algorithm to better extract local and domain knowledge-oriented explanations. A set of regulations is further introduced, combining learned interpretable representations with anchors and information extraction to generate comprehensible semantic explanations. To carry out an extensive experiment, we first develop a drug abuse ontology (DAO) on a drug abuse dataset on the Twitter sphere, and a consumer complaint ontology (ConsO) on a consumer complaint dataset, especially for interpretable ML. Our results show that our approach generates more precise and more insightful explanations compared with a variety of baseline approaches.

6.2 Background and Problem Definition

In this section, we revisit IML, ontology-based approaches, and information extraction algorithms, which are often used to generate explanations. We further discuss the relation to previous frameworks and introduce our problem definition.

Let D be a database that consists of N samples, each of which is a sample $x \in \mathbb{R}^d$ associated with its label y. Each y is a one-hot vector of K categories $y = \{y_1, y_2, \ldots, y_K\}$. A classifier outputs class scores $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$that maps an input x to a vector of scores $f(x) = \{f_1(x), f_2(x), \ldots, f_K(x)\}$ s.t. $\forall k \in [1, K] : f_k(x) \in [0, 1]$ and $\sum_{k=1}^{K} f_k(x) = 1$.

The highest-score class is selected as the predicted label for the sample. By minimizing a loss function $\mathcal{L}(f(x), y)$ that penalizes a mismatching between the prediction $f(x)$ and the original value y, an optimal classifier is selected.

6.2.1 Interpretable Machine Learning

Let us revisit IML, starting with the spirit of interpreting either a model behavior or a model outcome, given an input. Conventional approaches focus on measuring the impacts of input features or components, e.g., a word, a sentences, a super-pixel, on each predicted outcome [113, 122, 166]. Given this common goal, IML algorithms can be classified into several research lines, based on their techniques, as follows: **(1)** Feature attribution approaches; **(2)** Backpropagation methods; and **(3)** Attention networks.

Feature attribution approaches One of the oldest models in IML is based on Shapley value [179]. Shapley value is the average contribution of a specific feature value to the prediction in different coalition scenarios. To obtain an explanation for each feature, Shapley-based methods [113, 189] generally require training models with and without the feature over all possible combinations of other features, and then take the weighted average as the explanation for that feature. SHAP [119] is another mechanism that applies Shapley value on local interpretability. In this way, Shapley-based approaches have several limitations, as follows: (1) expensive computation, since there are an exponential number of possible coalitions of features; (2) potential misinterpretation, as they return a simple value per feature, and cannot make statements about changes in prediction for manipulating in the input; and (3) not covering correlated features.

Sharing a similar idea of manipulating and measuring as Shapley-based approaches, LIME [166] focuses on training local surrogate models that can explain individual predictions of an instance. Given an interpretable model g, which provides insights and qualitative understanding about the prediction model f given an input x, there are two important criteria in learning g: (1) local fidelity, which implies the ability of interpretable models to approximate the prediction model in a vicinity of the input, and (2) interpretability, which is a sufficiently low complexity of interpretable models that makes it easier for humans to understand the explanations. In textual data, the complexity, denoted as $T(g)$, usually is the number of important words [123, 166], based upon what users can easily handle, to evaluate the generated explanations.

Let z be a sample of x, where z is generated by randomly selecting or removing features or words in x. $\phi_x(z)$ is a similarity function to measure the proximity between x and z. Given a d'-dimensional binary vector $z' \in \{0,1\}^{d'}$, $z'_i = 1$ indicates that the feature i-th ($\in x$) is present in z, and vice-versa. To achieve the interpretability and local fidelity, Ribeiro et al. [166] minimize a loss function $L(f, g, \phi_x)$, with a low complexity $T(g)$, by solving the following problem:

$$g^* = \arg\min_g L(f, g, \phi_x) + T(g) \tag{6.1}$$

where $L(f, g, \phi_x) = \sum_z \phi_x(z)(f(z) - g(z'))^2$,

$\phi_x(z) = \exp(-D(x,z)^2/\sigma^2)$ is an exponential kernel with $D(x,z)$ is a distance function (e.g., cosine distance for textual data) with a width σ, and $g(z') = w_g z'$.

To obtain the data z for learning g in Equation 6.1, sampling approaches are employed. In LIME, the authors draw nonzero elements of the original data x uniformly at random. Similar to this approach, a number of works follow [9, 94, 139]. In summary, in these approaches, data is sampled while ignoring the correlation among features, which might not be practical in real-world scenarios, since features usually are highly and semantically correlated.

Backpropagation methods Layer-wise relevance propagation (LRP) [16] and Deep Learning Important Features (DeepLIFT) [184] rely on model decomposition through a backpropagation process, in which a prediction $f(x)$ is decomposed into individual contributions of input neurons. The main idea behind LRP is that, for each output class separately, the prediction score is reallocated from upper-layer neurons to the input space (e.g., words in text or pixels in images) via a backpropagation procedure [14]. In the same line of work, DeepLIFT is a recursive prediction explanation for deep learning to assign importance scores to the input variables. DeepLIFT aims at examining the difference of the activation of each neuron given the original input compared with its reference activation from some reference inputs. Then, the difference is used to explain the importance of the original input. In spite of easily applying to many domains and data types, backpropagation approaches can be unstable since they mainly rely on gradients which can be noisy and unsteady. In fact, given adversarial examples, a tiny perturbation of the input x can cause the output score $f(x)$ and then the gradient to change significantly, which can lead to a noisy and wobbly backpropagation procedure.

Attention Networks Facilitating attention networks in interpretability has been an emerging trend in deep learning nowadays [92, 124, 211]. The attention mechanism [198] is to selectively focus on relevant and important components to the model outcome, while ignoring other irrelevances in a network. Hierarchical Attention Network (**HAN**) [213] is one of the promising approaches for document-level classification since it not only results in better predictive performance, but also provides insight into which words and sentences contribute to the classification. HAN uses stacked bidirectional GRUs [17] on the word-level, followed by an attention network to extract words that are critical to the meaning of a sentence, and then to aggregate the words to form a sentence vector. The same mechanism is applied to the generated sentence vectors to obtain important sentences which contribute to the classification of the document. As in [71], HAN is typically slow to train, due to its complex architecture; and so, gradients are very expensive to compute. Technically, this shortcoming does not impede practical applications of HAN in textual data; therefore, we use it as one of the comparisons with our proposed method. The key difference between HAN and OnML is that, OnML leverages domain knowledge to tie the explanations up to the predicted label and considers correlations among words in textual data to generate semantic explanations from the text; whereas HAN does not leverage domain knowledge.

6.2.2 Ontology-based Approaches

To capture semantic correlations among input features, an ontology can be applied. Ontology is used in [69] to filter and rank concepts from selected data points to conduct informative explanations. The explanations are derived in ontological forms. For example, the information, "a 30-year-old individual, with an operation that occurred in 1989," can be conveyed by the representation, "*TheSilentGeneration* $\sqcap$ *OperationIn1980s*." (TheSilentGeneration denotes people in the age range of 30-39.) Heuristically, building a rich contextual ontology is expensive; so, typically, ontology only captures a limited number of core concepts and their correlations. This is the reason why ontological forms cannot capture all common sense knowledge in the textual information. In reality, humans generally use natural languages in a variety of text presentations. Therefore, an appropriate combination of a single-form ontology with other approaches to generate semantic explanations is necessary.

In [38], Confalonieri et al. use ontology to learn an understandable decision tree, which is an approximation of a neural network classifier. Explanations are in a non-syntactic form, and they are not designed to explain a single and independent data point. Different from [38], we aim at generating semantic explanations for each input x. In this paper, generating semantic explanations is defined as a process of mapping a text to a representation of important information in *a syntactic* and *understandable* form.

6.2.3 Information Extraction

Apart from IML, information extraction (IE) is another direction to capture contextual information semantically. The first Open IE algorithm is TextRunner [64], which identifies arbitrary relation phrases in English sentences by automatically labeling data using heuristics for training the extractor. Following [64], a number of Open IE [65,186,206] were introduced.

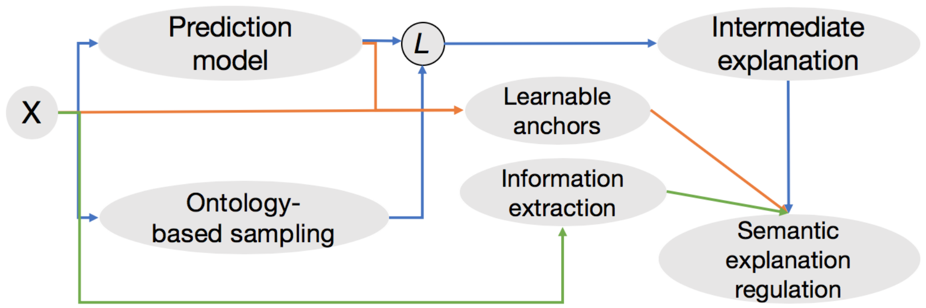


Figure 6.1 A flow chart of the OnML approach.

Unfortunately, these approaches ignore the context. OLLIE [175] includes contextual information; and extracts relations mediated by nouns, adjectives, and verbs; and outputs triplexes (subject, predicate, object). Compared with Open IE approaches, our algorithm mainly focuses on generating semantic explanations associated with the prediction label.

6.3 Ontology-based Interpretable Machine Learning for Textual Data

In this section, we formally present our proposed OnML framework (Figure . 6.1). Alg. 4 presents the main steps of our approach. Given an input x, an ontology $\mathcal{O}$, and a set of all concepts $\mathcal{C}$ in $\mathcal{O}$, we first present the notion of *ontology-based tuples* (Line 3), which will be used in an *ontology-based sampling technique* to learn the interpretable model g (Lines 4-6). Next, we learn potential anchor texts using the input x and the model $f(x)$ (Line 7). Meanwhile, OLIIE [175] is applied to extract triplexes, which have high confident scores, in x (Line 8). After learning g, learning anchor texts $\mathcal{A}$, and extracting triplexes $\mathcal{T}$, we introduce a set of regulations to combine them together to generate semantic explanations (Line 9). Let us first present the notion of ontology-based tuples as follows.

6.3.1 Ontology-based Tuples

Given concepts A and B, $A \mapsto B$ is used to indicate that A has a directed connection to B. In considerably correlated domains, such as text data, it is observed that 1) words appearing near to each other in a sentence have the same contextual information, and 2) different sentences usually have different contextual information. To encode the observations, we

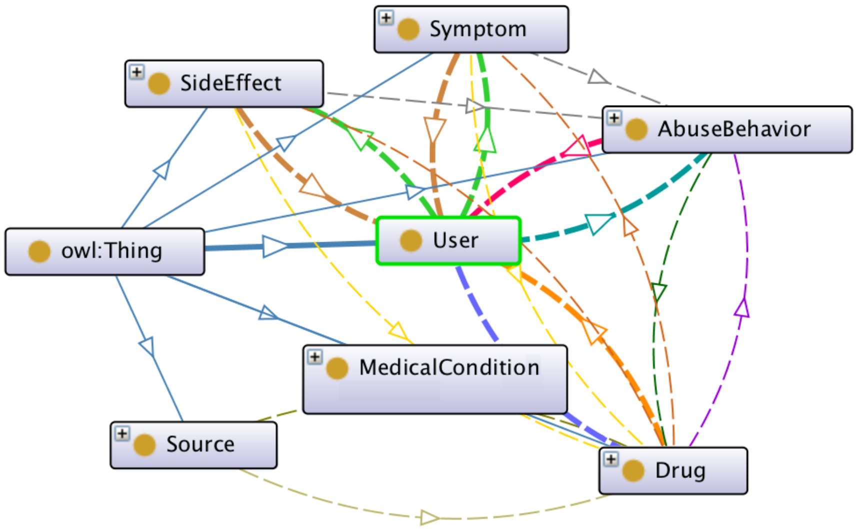


Figure 6.2 Drug abuse ontology.

introduce a *contextual constraint*, as follows:

$$\lambda_{x_k}(x_l) \leq \gamma \tag{6.2}$$

where x_k and x_l are two words in x, γ is a predefined threshold, and $\lambda_{x_k}(x_l)$ measures the distance between the positions of x_k and x_l in x. In text data, if x_k and x_l belong to two sentences, they are considered to be violating the contextual constraint. Intuitively, the constraint is used to connect words 1) that appear near to each other in a sentence (that are contextually correlated); and 2) that belong to connected concepts in the ontology (that are conceptually correlated). If there is no contextual constraint, there can be mismatched information between the domain knowledge and the explanation extracted in the text.

Definition 10. *Ontology-based tuple. Given x_k and x_l in x, (x_k, x_l) is called an ontology-based tuple, if and only if: (1) $\exists A, B \in \mathcal{C}$ s.t. $x_k \in A$ and $x_l \in B$; (2) $A \mapsto B$; and (3) $\lambda_{x_k}(x_l) \leq \gamma$.*

Since ontology has directed connections among its concepts, ontology-based tuples are asymmetric; i.e., (x_k, x_l) and (x_l, x_k) are different. For the sake of clarity without affecting the generality of the approach, we use a drug abuse ontology as an example (Figure 6.2).

Algorithm 4 OnML approach

```
1:  **Input:** Input $x$; ontology $\mathcal{O}$, and user-predefined anchor $\mathcal{A}_0$
2:  Classify $x$ by a prediction model $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$
3:  Find ontology-based tuples $(x_i, x_j)$ in $x$ based on concepts and relations in $\mathcal{O}$
4:  Sample $x$, based on ontology-based tuples found by our sampling technique to obtain
    sampled data $z \in \mathcal{Z}$
5:  Generate vectors of predictive scores $f(z)$ with $z \in \mathcal{Z}$
6:  Learn an interpretable model $g$ based on $f(z)$ and $g(z')$ by Equation 6.1
7:  Learn anchor text by our anchor learning algorithm (Alg. 5)
8:  Extract triplexes in $x$ using an existing Open IE technique
9:  Combine ontology-based tuples, learned anchors, and extracted triplexes by our
    proposed regulations
10: **Output:** Semantic explanation $\mathcal{E}$
```

Given the drug abuse ontology and x as *"She uses orange juice and does not like weed. She knows that smoke causes addiction and headache"*, a list of ontology-based words can be found {use, weed, smoke, addiction, headache}. These words are in "Abuse Behavior" (use and smoke), "Drug" (weed), "Side Effect" (addiction), and "Symptom" (headache) concepts. Following the aforementioned conditions (Equation 6.2 with $\gamma = 3$), two ontology-based tuples are found, which are (smoke, addiction) and (smoke, headache). In the meantime, (addiction, headache) and (weed, smoke) are not ontology-based tuples, since there is no directed connection between the "Side Effect" concept and the "Symptom" concept, and since "weed" and "smoke" are in different sentences. By using the contextual constraint, we can eliminate "use weed," which is contextually incorrect, from the explanation.

6.3.2 Ontology-based Sampling Technique

To integrate ontology-based tuples into learning g, we introduce a novel ontology-based sampling technique. To learn the local behavior of f in its vicinity (Equation 6.1), we approximate $L(f, g, \phi_x)$ by drawing samples based on x, with the proximity indicated by ϕ_x. A sample z can be sampled as:

$$z = \Big(\cup_{x_i \in x, i \neq k, i \neq l} \mathcal{R}(x_i) \Big) \cup \mathcal{R}(\{x_k, x_l\}) \tag{6.3}$$

where $\mathcal{R}(x_i)$ and $\mathcal{R}(\{x_k, x_l\})$ are probabilities randomly drawn for each word $x_i \in x(i \neq k, l)$ and words $x_k, x_l \in x$ together, respectively. If $\mathcal{R}$ is greater than a predefined threshold, then the word(s) will be included in z.

In our sampling process, x_k and x_l, i.e., an ontology-based tuple, are sampled together as a single element. This aims to integrate the semantic correlation between x_k and x_l, captured in an ontology-based tuple into the sampling process. In fact, we are sampling the semantic correlation, but not sampling each word/feature x_k or x_l independently. This enables us to measure the impact of this semantic correlation on $f(x)$. In addition, words, which are not in any ontology-based tuple, are sampled independently. After sampling x (Equation 6.3), we obtain the dataset $\mathcal{Z}$ that consists of sampled data points z associated with its label $f(z)$. $\mathcal{Z}$ is used to learn g^* by solving Equation 6.1.

6.3.3 Learnable Anchor Text

Before presenting our anchor mechanism, we introduce an *importance score* notion, which is to choose the best anchor and to calculate the importance of generated explanations.

Importance Score To get insights into the importance of generated explanations and their impacts upon the model outcome, we calculate an importance score (IC) for each explanation. Intuitively, the higher the importance score, the more important the explanation is. IC is calculated as:

$$IC(r) = \bar{c}_r \Big(f(x) - f(x/r) \Big) \tag{6.4}$$

where x/r is the original text x excluding words in the explanation r and $\bar{c}_r$ is average coefficients of g^* associated with all words in r.

Algorithm 5 Anchors learning algorithm

```
1:  Input: Input $x$; prediction model $f$; number of sentences in $x$, denoted as $M$; user-
    predefined anchors $\mathcal{A}_0$
2:  $\mathcal{A} \leftarrow \emptyset$ ($\mathcal{A}$ : set of anchors for $x$)
3:  for $i \in M$ do
4:      if any $\mathcal{A}_0$ appears in the sentence $i$ then
5:          Denote $D_{\mathcal{A}}$ as a set of ordered words appearing after $\mathcal{A}_0$ in the sentence $i$ in $x$
6:          $\mathcal{A}_n \leftarrow \emptyset$ ($\mathcal{A}_n$ is a set of candidate anchors)
7:          $\mathcal{F}_n \leftarrow \emptyset$ ($\mathcal{F}_n$ is a set of importance scores, associated with each candidate anchor)
8:          for $x_j \in D_{\mathcal{A}}$ do
9:              $\mathcal{A}_n \leftarrow \mathcal{A}_0 \cup x_j$; $\mathcal{A}_0 \leftarrow \mathcal{A}_n$; $\mathcal{F}_n \leftarrow \mathcal{F}_n \cup IC(\mathcal{A}_n)$
10:         end for
11:             Choose the best anchor for sentence $i$: $\mathcal{A}_i = \arg\max_{\mathcal{A}_n} \mathcal{F}_n$
12:     else
13:         $\mathcal{A}_i \leftarrow \emptyset$
14:     end if
15: end for
16:     $\mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{A}_i$
17: Output: $\mathcal{A}$
```

Anchor Text Learning Mechanism It is challenging to work with long and poor data, e.g., a large number of words, or misspelled text, since the contextual information is generally rich and complicated. Building an ontology to adequately represent such data is expensive, and insufficient in many cases. That results in a large undercovered search space for explanations. To address this problem, we introduce a learnable anchor mechanism to narrow down the search space.

The learning anchor technique is presented in Alg. 5. The anchor is initialized with an empty set (Line 2). A set of user-predefined anchors $\mathcal{A}_0$ is provided, which consists of starting-words that are further expanded by incrementally adding words to the end of the sentence. Then, the importance score of each candidate anchor is calculated, following Equation 6.4. The top-1 anchor $\mathcal{A}$, which has the highest important score, for each sentence is then chosen.

6.3.4 Generating Semantic Explanations

We further apply OLLIE [175] to extract triplexes $\mathcal{T}$ (subject, predicate, and object) to identify the syntactic structure in a sentence, which can shape our explanations in a readable form. To generate semantic explanations $\mathcal{E}$, we introduce a set of regulations to combine g^*, $\mathcal{A}$, and $\mathcal{T}$ together:

1) $\mathcal{E} \subseteq D_x$ with D_x is a set of all words in x.

2) If there is no ontology-based tuple found, $\mathcal{E}$ will only consist of the learned anchor texts.

3) In a sentence, if there are two or more ontology-based tuples, we introduce four rules to merge them together:

- *Simplification:*
  - Given (x_k, x_l) and (x_k, x_m), if x_l and x_m are in the same concept, then the ontology-based explanation is $\{x_k, x_l$ *and/or* $x_m\}$.
  - Given (x_k, x_m) and (x_l, x_m), if x_k and x_l are in the same concept, then the ontology-based explanation is $\{x_k$ *and/or* $x_l, x_m\}$.
  - Given (x_k, x_l) and (x_l, x_m), then the ontology-based explanation is $\{x_k, x_l, x_m\}$.
- *Union*: Given (x_k, x_l), (x_k, x_m), (x_l, x_m), and $\{x_k, x_l, x_m\}$, the ontology-based explanation is $\{x_k, x_l, x_m\}$.
- *Adding Causal words*: Semantic explanation can be in the form of a causal relation. Thus, if a causal word, e.g., “because,” “since,” “therefore,” “while,” “whereas,” “thus,” “thereby,” “meanwhile, “however,” “hence,” “otherwise,” “consequently,” “when,” or

"whenever" appears between any words in ontology-based tuples/explanations, we add the word to the explanation, following its position in x.

- *Combining with anchor texts $\mathcal{A}$ and triplexes $\mathcal{T}$:* After having ontology-based explanations, we combine them with $\mathcal{A}$ and $\mathcal{T}$ based on their positions in x. Then, the *semantic explanation* is generated from the beginning towards the end of all positions of words found in the ontology-based explanations, $\mathcal{A}$, and $\mathcal{T}$. For example, in the sentences, "We were filling out all the forms in the application. However, there is a letter, saying, Loss mitigation application denied for not sending information to us.", after the learning process, we obtain: 1) ontology-based explanation is (loss, application); 2) anchor text is "not sending information;" and 3) triple is "a letter; denied; mitigation application." The explanation $\mathcal{E}$ is "a letter saying loss mitigation application denied for not sending information."

4) If different ontology-based tuples are in different sentences in x, due to the contextual constraint in Equation 6.2, the explanation for each sentence follows the 3^{rd} regulation.

It is worth noting that we use the aforementioned regulations to combine ontology-based tuples to form a longer ontological term. This makes the ontology used a much better representation rather than independent and direct connections $A \mapsto B$.

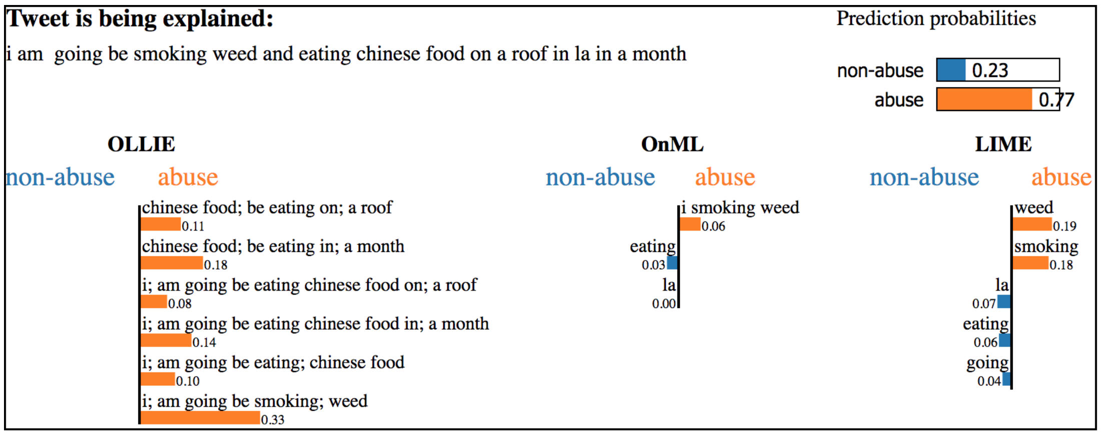
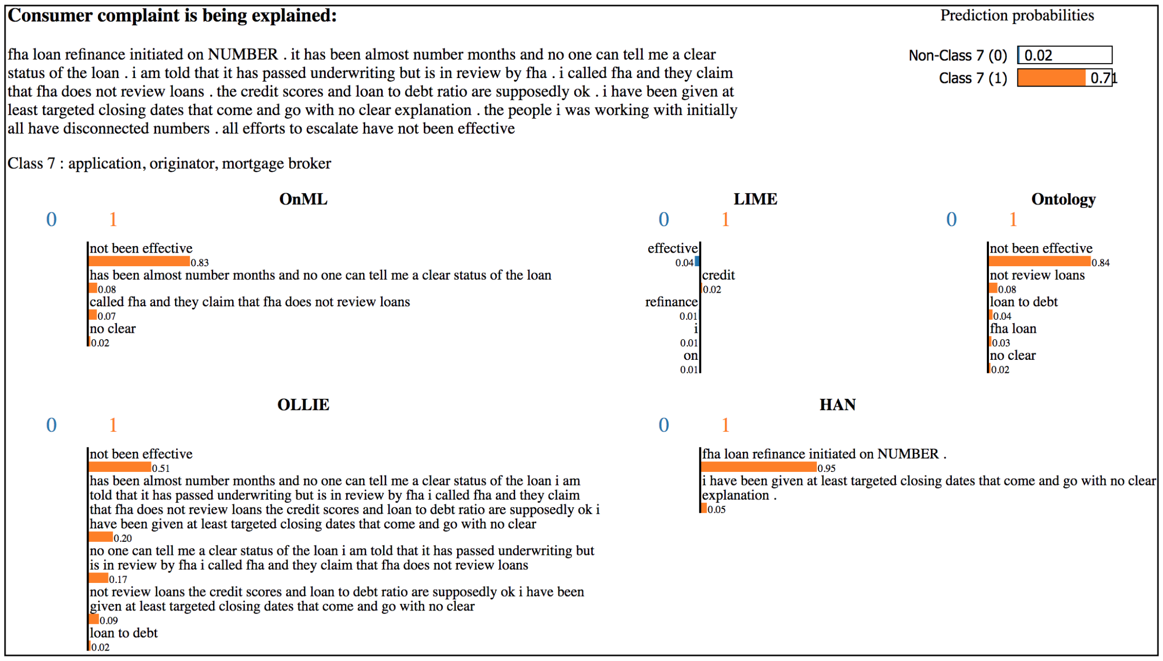


Figure 6.3 Visualization of a drug abuse experiment.



Consumer complaint is being explained:

fha loan refinance initiated on NUMBER . it has been almost number months and no one can tell me a clear status of the loan . i am told that it has passed underwriting but is in review by fha . i called fha and they claim that fha does not review loans . the credit scores and loan to debt ratio are supposedly ok . i have been given at least targeted closing dates that come and go with no clear explanation . the people i was working with initially all have disconnected numbers . all efforts to escalate have not been effective

Class 7 : application, originator, mortgage broker

Prediction probabilities

Non-Class 7 (0)	0.02
Class 7 (1)	0.71

OnML (0, 1)

not been effective	0.83
has been almost number months and no one can tell me a clear status of the loan	0.08
called fha and they claim that fha does not review loans	0.07
no clear	0.02

LIME (0, 1)

effective	0.04
credit	0.02
refinance	0.01
i	0.01
on	0.01

Ontology (0, 1)

not been effective	0.84
not review loans	0.08
loan to debt	0.04
fha loan	0.03
no clear	0.02

OLLIE (0, 1)

not been effective	0.51
has been almost number months and no one can tell me a clear status of the loan i am told that it has passed underwriting but is in review by fha i called fha and they claim that fha does not review loans the credit scores and loan to debt ratio are supposedly ok i have been given at least targeted closing dates that come and go with no clear	0.20
no one can tell me a clear status of the loan i am told that it has passed underwriting but is in review by fha i called fha and they claim that fha does not review loans	0.17
not review loans the credit scores and loan to debt ratio are supposedly ok i have been given at least targeted closing dates that come and go with no clear	0.09
loan to debt	0.02

HAN (0, 1)

fha loan refinance initiated on NUMBER .	0.95
i have been given at least targeted closing dates that come and go with no clear explanation .	0.05

Figure 6.4 Visualization of a consumer complaint experiment.

6.4 Experiment

We have conducted extensive experiments on two real-world datasets, including drug abuse (Twitter-sphere [88]) and consumer complaint analysis from the Consumer Financial Protection Bureau[1]. Our evaluation will focus on shedding light into these assertions: **(1)** Our OnML approach can be applied on different agnostic predictive models; and **(2)** Our approach can generate better explanations, compared with baseline approaches, in both quantitative and qualitative measures. Our code and data are available on Github[2].

6.4.1 Datasets and Domain Ontologies

In this study, we have developed two different domain ontologies, which are drug abuse ontology (Figure 6.2) and consumer complaint ontology (Figure 6.5). These ontologies were constructed for certain domains, e.g., drug abuse and consumer complaint, since it is necessary to capture specific semantic and causal relations among components. As default in Protégé [102], each arrow in a particular color demonstrates a certain type of causal relation

[1] `https://www.consumerfinance.gov/data-research/consumer-complaints/`

[2] `https://github.com/PhungLai728/OnML`

in which its tail represents a domain and its head represents a range of the relation. For example, in the drug abuse ontology (Figure 6.2), a purple arrow signifies “is involved with,” where domain is “Drug” and range is “Abuse Behavior”; while green arrows are for “suffer from.” These ontologies were semi-manually generated, in which concepts were grouped and collected from the dataset by a K-means clustering algorithm [68], and then judged by humans to reduce inappropriate concepts.

Drug Abuse Dataset We will use the term “drug abuse” in the wider sense, including abuse and use of Schedule 1 drugs, which are drugs with a high potential for abuse and presently with no recognized medical use, whether obtained legally, (e.g. legal painkillers) or illegally (e.g. getting drugs without prescription or even from black market); and misuse of Schedule 2 drugs, which have medical uses, yet have a potential for severe addiction, and which can be life-threatening [18]. Main concepts of the drug abuse ontology (**DrugAO**) (Figure 6.2) capture correlation among key concepts, including “abuse behaviors”, “drug types”, “drug sources”, “drug users”, “symptoms”, “side effects”, and “medical condition” when using drugs. The “ abuse behaviors” concept is about behaviors of abusers, such as abuse, addict, blunt, etc. Drug types consists of different types of legal and illegal drugs, e.g., narcotics, cocaine, and weed. “The Drug sources” concept categorizes where drug users, who are the main objects of the ontology, obtain their drugs. “Symptoms” and “side effects” are about different negative short-term and long-term effects of drugs on users. “Medical condition” contains terms about expression of disease and illness caused by using drugs. In total, DrugAO has 506 drug-abuse related terms, and 18 relations.

The drug abuse dataset (Table 6.1) consists of $9,700$ tweets labelled by [88] with a high agreement score. Among them, $3,043$ tweets are drug abuse tweets, labeled *positive* and the rest are non drug abuse tweets, labeled *negative*.

Consumer Complaint Dataset A consumer complaint is defined, here, as a complaint about a range of consumer financial products and services, sent to companies for response.

In complaints, consumers typically talk about their mortgage-related issues, such as: (1) Applying for a mortgage or refinancing an existing mortgage (application, credit decision, underwriting); (2) Closing on a mortgage (closing process, confusing or missing disclosures, cost); (3) Trouble during payment process (loan servicing, payment processing, escrow accounts); (4) Struggling to pay mortgage (loan modification, behind on payments, foreclosure); (5) Problem with credit report or credit score; (6) Problem with fraud alerts or security freezes, credit monitoring or identity theft protection services; and (7) Incorrect information on consumer's report or improper use of consumer's report. The main concepts of the consumer complaint ontology (**ConsO**) (Figure 6.5)) encode the relation among different entities pertaining to the consumer complaint: for instance, who is complaining; what happened to make consumers unhappy and then complain; etc. There are six major concepts in ConsO, which are "thing in role", "complaint," "event", "event outcome", "property", and "product". "Thing in role" identifies the people and organizations related to the complaint, such as buyers, investors, dealers, etc. "Event" and "event outcome" are about negative events that happened that caused consumer complaints. "Property" denotes things belonging to consumers, and "product" denotes substances of some parties (e.g., banks) offeredto consumers. In total, ConsO has 572 finance and product-related terms and 9 relations. The consumer complaint dataset consists of $33,635$ mortgage-related complaints, labeled with 16 categories. These complaints were used for learning a model to predict the issue regarding each complaint.

6.4.2 Baseline Approaches

Our OnML approach is evaluated in comparison with traditional approaches: **(1)** interpretable model-agnostic explanation, i.e., LIME [166]; **(2)** information extraction, i.e., OLLIE [175]; and **(3)** Hierarchical attention network, i.e., HAN [213]. LIME is one of the well-applied approaches in IML, especially for textual data, in which the predictions of any model are explained in a local region near the sample being explained. There are

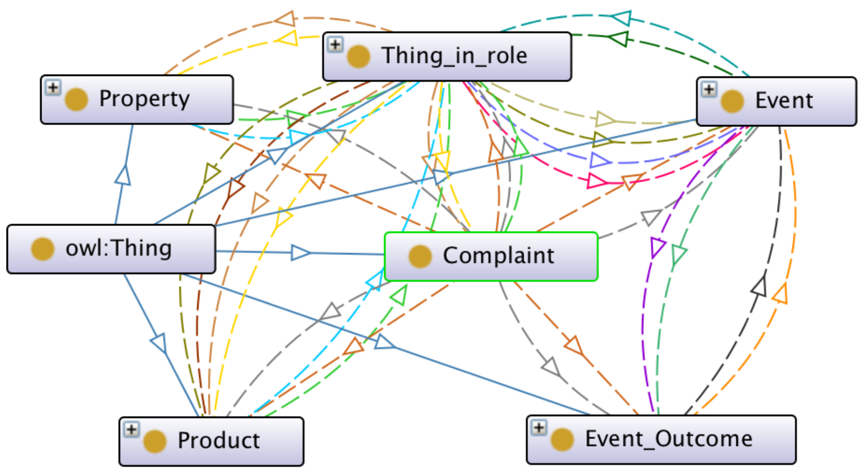


Figure 6.5 Consumer complaint ontology.

other algorithms sharing the same spirit as LIME, in terms of generating explanations [16, 119, 177, 184, 187, 192]. For the sake of clarity, we use LIME as a representative baseline regarding this line of work. OLLIE focuses more on grammatical analysis to extract triples from the text. HAN is one of the popular frameworks in attention networks in which such words and sentences with high attention weights are expected to be influential to the classification and to be good candidates for explanation. The key differences among OnML, LIME, OLLIE, and HAN are (a) that OnML leverages domain knowledge to tie the explanations to the predicted label and (b) that OnML considers correlations among words in textual data to generate semantic explanations. Meanwhile, no domain knowledge is applied in LIME, OLLIE, or HAN.

6.4.3 Experimental Settings

To achieve our goal, we carry out our evaluation through three approaches. First, by employing SVM [41] and LSTM [84], we aim to illustrate that OnML works well with different agnostic predictive models. HAN is also used as a predictive model and its sentence-level attention components are used as an explanation to compare with the other approaches. Second, we leverage the word deleting approach [14] as a quantitative evaluation. Third, we apply qualitative evaluation with Amazon Mechanical Turk (**AMT**).

Model Configurations In the drug abuse dataset, tweets were vectorized by TF-IDF [163] and then classified by a linear kernel SVM model. We achieved 83.6% accuracy. Tweets are short, i.e., the average and maximum numbers of words in a tweet are 12 and 37 (Table 6.1). Therefore, it is not necessary to apply the anchor learning algorithm and information extraction, which are designed to tighten down the search space and syntactically connect for long text data.

In the consumer complaint dataset, Word2vec [131] is applied for feature vectorization. Then, an LSTM is trained as a prediction model. In LSTM, we used an embedding input layer with $d = 300$, one hidden layer of 64 hidden neurons, and a softmax output layer with 16 outputs. In HAN, we used Glove embedding [148] with $d = 300$ for an embedding input layer, one hidden layer of 150 hidden neurons, and a softmax output layer with 16 outputs. An efficient ADAM optimization algorithm [100] with learning rate 0.01 was employed to train LSTM and HAN. For the prediction models, we used $33,635$ samples and observed the accuracy increasing to 60% compared with $13,965$ samples resulting in 53% accuracy as in our previous experiment [106]. We registered this accuracy increase to be in line with our expectations. For sufficiently learning anchors in consumer complaints, we have chosen a set of negative terms as user-predefined anchors $\mathcal{A}_0 = \{$not, no, illegal, against, without$\}$.

Importance scores in LIME are weights of the linear interpretable model. With OLLIE, importance scores of extracted triplexes are calculated in the same way as in our method (as shown in Equation 6.4). LIME and OLLIE settings are used as default in [166,175]. We only show OLLIE rules which have the confidence score greater than 0.7 and top-5 words from LIME to ensure that the best explanations from OLLIE and LIME are shown. The contextual constraint γ in Equation 6.2 is 3 for drug abuse and 10 for consumer complaint dataset. The pre-defined threshold in Equation 6.3 is 0.5. Note that, from previous experiments [106], it is shown that OLLIE did not work well, especially in complicated and lengthy texts, as in consumer complaints. In addition, OLLIE tends to be well-suited to information extraction, rather than an IML approach. Therefore, in this study, we drop the comparison with OLLIE, and add a comparison with HAN in the consumer complaint dataset. To compare with other approaches, we take the top-3 highest importance score sentences from HAN, since importance scores after third sentences tend to be negligibly small. For short complaints that consist of two or three sentences, we put constraints on the HAN-based explanation algorithm to have a one sentence explanation. In addition, another variation of our algorithm is to combine ontology-based terms and anchors, called an **Ontology** algorithm. This is further used to comprehensively evaluate our proposed approach.

Quantitative Evaluation In this article, we propose two metrics for quantitative evaluation, which are accuracy changes (**AC**) and prediction score changes (**SC**). These metrics are based on the word deleting approach [14], which deletes a sequence of words from a text and then re-classifies the text with missing words. By computing the differences between the original text and the missing text, we examine the importance of the explanation to the prediction. Intuitively, the higher values of AC and SC indicate the more important explanations derived. AC and SC are computed as follows.

$$\text{AC} = \text{Original accuracy} - \frac{\sum_{i=1}^{|test|} \text{Updating accuracy}}{|test|}$$

$$\text{SC} = \frac{\sum_{i=1}^{|test|} IC(\text{top-}k \text{ explanations of } i\text{-th sample})}{|test|}$$

where “Original accuracy” is the average testing accuracy of all original texts; “Updating accuracy” is the accuracy of each missing text; and $|test|$ is the number of testing examples.

In our experiment, we deleted the top-k highest importance score explanations in OnML, OLLIE (for drug abuse experiment only), Ontology, and HAN approaches and the top-m highest weighted words in LIME. To be impartial, m is the number of words in the k-deleted explanations in OnML. In drug abuse, $k = 1$ since the tweet is typically short, and so there are not many explanations generated. In consumer complaint classifying, $k \in \{1, 2, 3\}$.

Table 6.1 Data Statistical Analysis

Statistics \ Dataset	Drug abuse	Consumer complaint
# of samples	9,700	33,635
# of categories	2	16
Max # of words/sample	37	5,852
Mean # of words/sample	12	317

Table 6.2 AC and SC in Drug Abuse

	Accuracy changes (%)	Score changes (%)
LIME	15.04	26.98
OLLIE	15.47	23.52
OnML	**25.52**	**33.48**

Qualitative Evaluation We recruit human subjects on Amazon Mechanical Turk (AMT). This is a common means of evaluation of the needs for qualitative investigation by humans [122, 177]. Detailed guidance is provided to users before they conduct the task.

We asked AMT workers to choose the best explanation by seeing side-by-side (in drug abuse) or head-to-head (in consumer complaint) explanation algorithms. To be more specific, head-to-head comparison is employed in consumer complaint in this version since we observed that choosing one over two algorithms helps to increase the reliability and to reduce the incidental confusions of choices that AMT workers made, compared with asking them to make a decision based on multiple algorithms. On top of each visualization, we provided the original tweet or complaint associated with their labels and prediction results. **It is important to note that**, in our real experiment, to avoid bias, the name of each algorithm is hidden, and their positions in the visualization are randomized.

We were recruiting 4 users per tweet in the drug abuse experiment, and 3 users per complaint in each head-to-head consumer complaint experiment. To quantify the voting

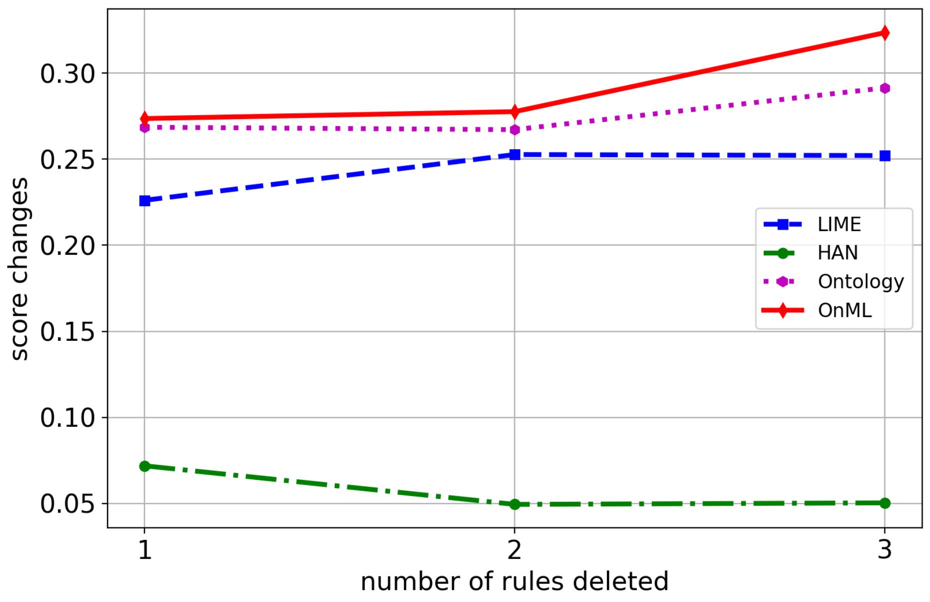


Figure 6.6 Average score changes in consumer complaint.

results from AMT users, we use: **(1)** Count the total number of votes, called *normal count*, i.e., the best algorithm is chosen over all $1,200$ votes (4 users/tweet $\times$ 300 tweets); and **(2)** Count the majority number of votes, called *majority count*, i.e., the best algorithm for each tweet is the algorithm with the largest number over 4 votes. Majority count is not used in head-to-head comparison. Relying on these voting results, we also investigate the effect of confidence levels of prediction probability and effect of text complication, i.e., length of text, on generated explanations.

6.4.4 Experimental Results and Analysis

To evaluate the interpretability of each approach, 300 positive tweets and $2,000$ consumer complaints, randomly selected, were used.

Drug Abuse Explanation As in Table 6.2, the accuracy is decreased significantly, and the predictive score changes the most in OnML. In fact, the values of AC and SC are 25.52$\%$ and 33.48$\%$ given OnML, compared with 15.47$\%$ and 23.52$\%$ given OLIIE, and 15.04$\%$ and 26.98$\%$ given LIME. This demonstrates that the explanations generated by our algorithm are more significant, compared with the ones generated by baseline approaches. In the

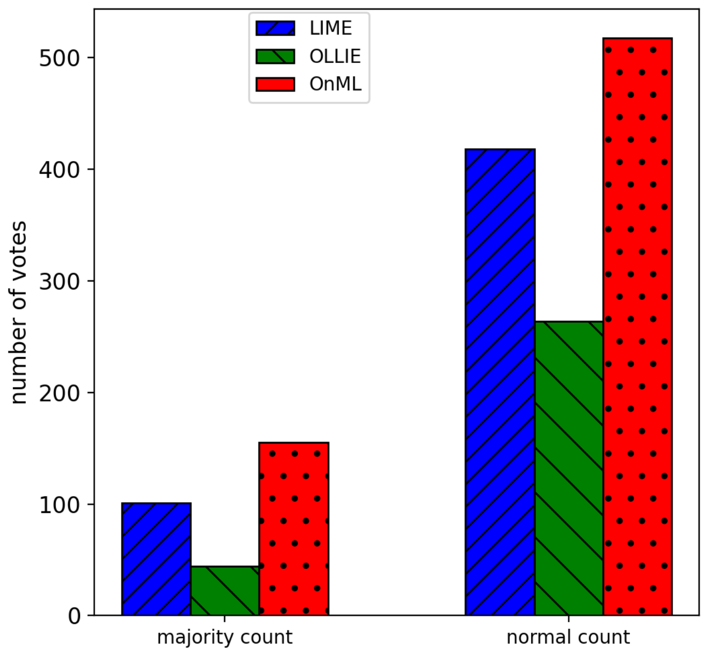


Figure 6.7 AMT results for drug abuse dataset.

evaluation by humans using AMT (Figure 6.7), OnML clearly outperforms LIME and OLLIE. Text in the tweet is generally short and can be represented by several key words. Therefore, individual words learned by LIME can be sufficient to generate more insightful explanations. Meanwhile, OLLIE tends to extract all possible triplexes in the text, which can be redundant and wordy explanations.

Consumer Complaint Explanation The results on the consumer complain dataset further strengthen our results. Figure 6.6 shows SC after deleting top-1, top-2, and top-3 explanations from OnML, Ontology, and OLLIE, as well as after deleting the most important words in LIME. In all three cases, score changes in OnML have the highest values, indicating that the explanations generated by OnML are the most significant to the prediction. In the evaluation by humans using AMT (Figure 6.8), OnML algorithm outperforms baseline approaches. Ontology approach achieves higher results than LIME and comparable results with HAN. LIME does not consider semantic correlations among words, resulting in a poor outcome in all cases.

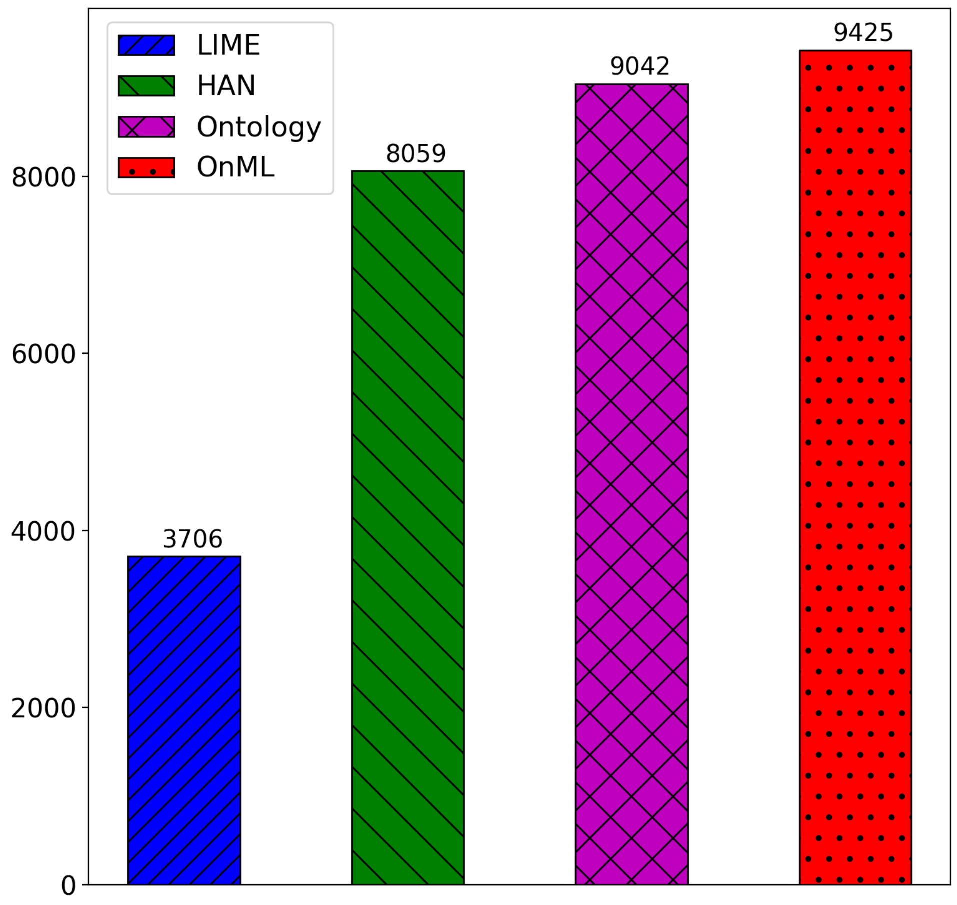


Figure 6.8 AMT results for consumer complaint dataset.

Completeness and Concision In Figure 6.4 (*top*), OnML generates “i smoking weed,” which provides concise and complete information about why it is predicted as a drug abuse tweet (smoking weed) and who was doing it (i) in a syntactic form. Meanwhile, 1) LIME derives relevant words to drug abuse (i.e., weed, smoking) without considering the correlation among these words; and 2) OLLIE generates lengthy and somewhat irrelevant explanations, e.g., “chinese food; be eating on; a roof.” In Figure 6.4 (*bottom*), OnML derived semantic explanations for consumer complaints, which tell us that consumers were facing issues in loan refinance, e.g., “called fha and they claim that fha does not review loans.” Compared with OnML, Ontology generates laconic explanations, e.g., “fha loan” that give no sense of why the consumer complained. LIME provides a set of fragmented words, and OLLIE generates wordy explanations, both of which are difficult to follow. HAN-selected

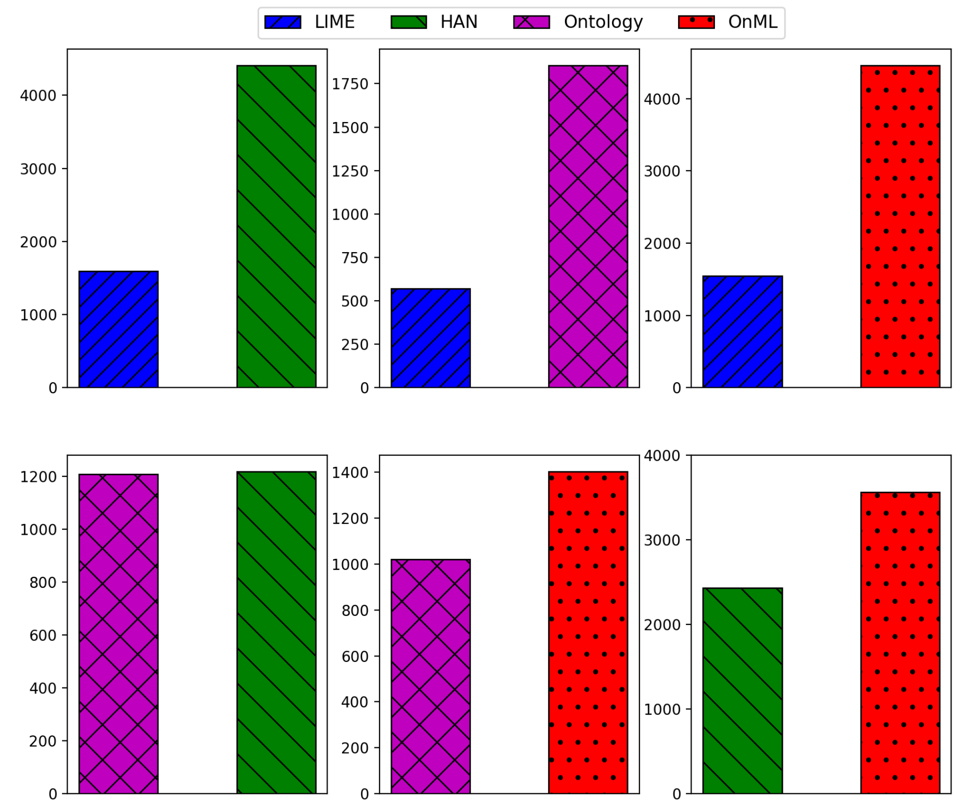


Figure 6.9 Head-to-head comparison results.

sentences do not describe key issues of the complaint, e.g., "fha loan refinance initiated on NUMBER" does not present why the consumer complained.

Text Complication and Confidence Levels We further investigate how text complication (i.e., lengths of text/document) and confidence levels of prediction (i.e., prediction scores) affect evaluation results. In this investigation, we only use OnML and HAN since the results of LIME are poor, as shown previously. As shown in Figure 6.10, OnML results are significantly higher than HAN in different lengths of text. Especially when lengths of text are between 90 and 150 words, OnML obtained nearly 400 more votes than HAN. For different confidence levels of prediction, as shown in Figure 6.11, with low confidence explanations (i.e., prediction score < 0.5), both OnML and HAN have poor results; meanwhile, with high confidence explanations (i.e., prediction score ≥ 0.5), OnML outperforms HAN. We used the 2-tail t-test to determine if there is a significant difference between the means of

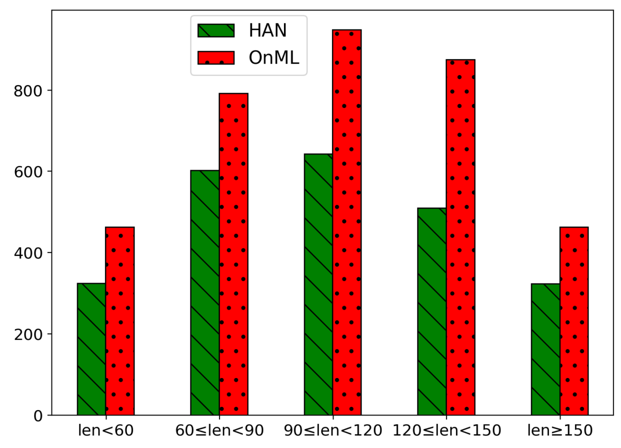


Figure 6.10 Length of complaints.

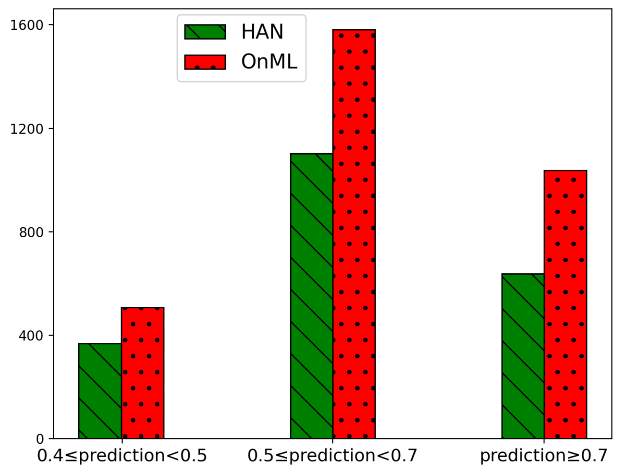


Figure 6.11 Prediction probabilities groups.

the two algorithms. For both text complication and confidence levels of prediction analysis, we obtained $p < 0.05$, which means the difference between HAN and OnML is significant.

Our key observations are: **(1)** Combining ontology-based tuples, learnable anchor texts, and information extraction can generate complete, concise, and insightful explanations to interpret the prediction model f; **(2)** Our OnML model outperforms other baseline approaches in both the quantitative and qualitative experiments, showing a promising result; and **(3)** Our OnML model outperforms HAN, which is one of the popular approaches in attention networks and can be used for explanations, across different levels of text complication and confidence levels.

6.5 Discussion

We proposed a novel ontology-based IML to generate semantic explanations, by integrating interpretable models, ontologies, and information extraction techniques. A new ontology-based sampling technique was introduced, to encode semantic correlations among features/terms in learning interpretable representations. An anchor learning algorithm was designed to limit the search space of semantic explanations. Then, a set of regulations for connecting learned ontology-based tuples, anchor texts, and extracted triplexes is introduced, to produce semantic explanations. Our approach achieves a better performance, in terms of semantic explanations, compared with baseline approaches, illustrating a better interpretability into ML models and data. Our approach paves an early brick on a new road towards gaining insights into machine learning using domain knowledge.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this study, we develop a trustworthy machine learning model in connection to privacy, security, explainable AI, and robustness. The proposed methods are widely applicable across machine learning paradigms and application domains.

First, we develope a novel notion of user-entity DP (UeDP), providing protection to both the participation information of users and sensitive entities in learning NLMs. By incorporating non-sensitive samples in the training process, we address the trade-off between model utility and privacy loss with a tight bound of model sensitivity. In addition, considering non-sensitive samples into our UeDP estimators notably improves model utility under the same UeDP protection. The more number of sensitive entities is, the lower the model utility will be; and vice-versa.

Second, wee propose ScalableRR, a novel dimension-scalable and bit-aware RR mechanism. To optimize the trade-off among dimensionality, data utility, and privacy protection, ScalableRR introduces a bit-aware term for better data utility and a dimension-scalable temperature for better control of the randomization probabilities. Our key idea is that when the dimension of a feature vector increases, one can always find a suitable temperature such that the changes in the randomization probabilities and in the total privacy budget are marginal. Hence, ScalableRR achieves better dimension-scalability and utility under rigorous LDP protection than existing mechanisms. Theoretical analysis and experiments show that ScalableRR outperforms baselines in text and image classification using several benchmark datasets. The results also show that ScalableRR is effective in defending against data reconstruction attacks in FL.

Third, we show that, although explanations help improve the understanding and interpretability of black-box models, they also leak essential information about the inner workings of the models. Therefore, the black-box models become more vulnerable to attacks, especially in the context of MLaaS where the prediction and its explanation are returned for each query. With a novel two-step LDP-preserving mechanism, we propose XRAND to protect the model explanations from being exploited by adversaries via obfuscating the top important features, while maintaining the faithfulness of explanations.

Fourth, we show that L2M introduces unknown privacy risk and challenges in preserving DP. To address this, we establish a connection between DP preservation and L2M, through a new definition of Lifelong DP. To preserve Lifelong DP, we proposed the first scalable and heterogeneous mechanism, called L2DP-ML. Our model shows promising results in several tasks with different settings and opens a long-term avenue to achieve better model utility with lower computational cost, under Lifelong DP.

In an additional work on explainable AI, we propose a novel ontology-based IML, called OnML, to generate semantic explanations, by integrating interpretable models, ontologies, and information extraction techniques. A new ontology-based sampling technique was introduced, to encode semantic correlations among features/terms in learning interpretable representations. An anchor learning algorithm was designed to limit the search space of semantic explanations. Then, a set of regulations for connecting learned ontology-based tuples, anchor texts, and extracted triplexes is introduced, to produce semantic explanations.

7.2 Future Work

We propose a series of privacy-preserving mechanisms in training ML models, across learning paradigms and application domains. Technically, there are still fundamental problems in order to bring trustworthy ML through the lens of privacy and security towards practical applications, which opens several prospective research directions.

Recent attacks have demonstrated that private information about clients' training data can be inferred via the shared gradients in FL. The jointly trained ML models can be poisoned by a small set of compromised clients, causing significant model integrity degradation. In addition, model's explanations can leak information about the model and data, which can be exploited to conduct attacks. The diversity of non-independent and identically distributed (**non-iid**) data across clients further exacerbates these vulnerabilities in real-world applications. Existing FL mechanisms have not been designed to be robust to such privacy and integrity risks given different levels of non-iid. I will develop PROXFL, the first **P**rivate, **Ro**bust, and E**X**plainable **FL** framework that provides robustness guarantees against privacy and model integrity attacks, without undue sacrifice in model utility and explanations' faithfulness. To put PROXFL to work, fundamental challenges in connecting DP, robustness in FL, and explainability given non-iid data need to be synergistically overcome. Specifically, I will explore a new surface of *non-iid-based attacks* to discover unrevealed vulnerabilities of FL systems. By either imitating the gradients from tail-clients, who usually have distorted gradients or exploiting different data densities among clients, a smaller number of compromised clients and a dishonestly curious server can carry out model integrity attacks and privacy inference attacks respectively, with guarantees of successful rates even under existing DP protection. In addition, I will theoretically connect FL and DP-preserving given non-iid data by introducing new concepts of *Proactive and Personalized Local DP* along with a new set of rigorous theories to address utility-privacy trade-offs. By uncovering the correlation among the number of compromised clients, the malicious input perturbation, and proactive and personalized LDP protection in a unified robustness condition for the first time, I will provide theoretical foundations to derive tighter and more reliable certified robustness bounds. Furthermore, I will explore the correlation between the explanation-guided attacks with DP, given different levels of non-iid. Finally, I will integrate the solutions into a unified FL framework with adaptive grouping algorithms to further optimize utility-privacy trade-offs while providing rigorous and reliable certified robustness and faithful explanations.

Another future direction is to create a secure and robust lifelong learning with certified defenses. Lifelong learning is important in ML to acquire new skills quickly without forgetting knowledge obtained from the preceding tasks. That exposes new and severe vulnerabilities in which deployed L2M models can be exploited 1) to reveal sensitive information in private training data, 2) to make the models misclassify with adversarial examples, and 3) to poison the training data for backdoor attacks. Technically, existing L2M mechanisms have not been designed to be robust to such privacy and integrity attacks. Such lack of protection and efficacy: 1) Significantly degrades the performance of CL systems and 2) Puts sensitive data at risk, thereby exposing service providers to legal actions under HIPAA/HITECH law. Furthermore, existing L2M works are still facing a catastrophic forgetting problem, resulting in a poor model utility when the number of tasks is large even without privacy-preserving mechanisms. The root cause is the gap between L2M and actual human learning, which diminishes L2M from effectively and efficiently learning over time. I will develop SECUREL2M to advance and seamlessly integrate different key techniques, including L2M, neuroscience, differential privacy, and adversarial learning offering tight and reliable robustness against both privacy and integrity attacks, while achieving superior model utility and less forgetting. To achieve the goals, first I will establish the connection between Neuroscience and ML to mimic the learning process of human learning into L2M. The connection can bridge the L2M-human learning gap, lessening the forgetting problems and improving model utility over time. Next, for training, I will establish a foundation for adversarial learning to improve the robustness of L2M models. To theoretically connect adversarial learning and DP in a *DP Adversarial Lifelong learning*, I will introduce a new concept of *Weighted DP Adversarial Examples* along with a new set of rigorous theories to address the trade-off between model utility, catastrophic forgetting, and privacy loss. In the inference phase, I will introduce a new theory of *Adaptive Composition Robustness* to establish robustness bounds against adversarial examples. A customized noise redistribution will be proposed to balance the trade-off among model utility, catastrophic forgetting, privacy, and robustness.

These projects will improve the trustworthiness of ML from different important aspects, i.e., privacy, robustness, explainability, and model utility. They enable safe, efficient, and deep analyses of data in many application domains, e.g., healthcare, social network, recommendation, etc.

APPENDIX A

A.1 Sensitive Entity Recognition and Tool-kits

If a training set does not have sensitive entity indicators, we suggest several ways to identify sensitive entities in textual data, as follows.

Using Named Entity Recognition (NER) datasets. NER datasets [45, 173] refer to textual data in which entities in a text are labeled based on several predefined categories. NER typically makes it easy for individuals and systems to identify and understand the subject of the given text quickly. Therefore, extracted entities are critical and should be protected. For instance, in the CONLL-2003 dataset [173], there are four entity types, i.e., location, person, organization, and miscellaneous.

Using Publicly Available Tool-kits. For textual datasets that do not have NER labels or sensitive entity indicators, there are publicly available tool-kits for detecting named entities or PII in text, for example, Spacy [85], Stanza [157], and Microsoft Presidio[3]. Spacy and Stanza deploy pre-trained NER models based on statistical learning methods to identify eighteen categories of named entities, including person, nationality or religious groups, facility, etc. (Table 2.1). Microsoft Presidio is another toolbox for PII detectors and NER models based on Spacy and regular expression[1]. For instance, Spacy is used as a sensitive entity identification in Fig. 2.1 to detect "David Johnson" a person entity, "Main" a GPE entity, "September 18" a date entity, and "Main Hospital" an organization entity.

We present descriptions of different sensitive entity categories in the CONLL-2003, AG, and SEC datasets in Table 2.1. The descriptions are from [173] and spaCy, supporting eighteen different entity types. In the current work, we play with four different types and their combinations. Note that, in UeDP, providing the name of an algorithm and a sensitive entity means we consider that type of entity as sensitive entities in the training process. For instance, in Fig. 2.4, UeDP-Alg $f_{\mathcal{E}^+}$ (Org) means we use all organization entities as

[1]`https://github.com/google/re2/`

sensitive entities in the UeDP-Alg algorithm. “All entities” means all types of sensitive entities considered for the dataset are used. For example, “all entities” in the CONLL-2003 dataset means all person, location, organization, and miscellaneous entities are regarded as sensitive entities. Meanwhile, in the AG and SEC datasets, “all entities” means that all organization, location, GPE, and PII entities are considered sensitive entities. More entity types are also presented in Table 2.1 so that users can have more choices when identifying sensitive entities.

A.2 UeDP without Considering Extended Sensitive Entities

At each iteration t, we randomly sample U^t users from U and E^t sensitive entities from E, with sampling rates q_u and q_e, respectively. Then, we use all sensitive sentences consisting of the sensitive entities in E^t belonging to the selected users in U^t for training. Like [128], we leverage the basic federated learning setting in [127] to compute gradients of model parameters for a particular user, denoted as $\Delta_{u,\mathcal{E}}^{t+1}$. Here, we clip the per-user gradients so that its l_2-norm is bounded by a predefined gradient clipping bound β. Next, a weighted-average estimator $f_{\mathcal{E}}$ is employed to compute the average gradient Δ^{t+1} using the clipped gradients $\Delta_{u,\mathcal{E}}^{t+1}$ gathered from all the selected users. Finally, we add random Gaussian noise $\mathcal{N}(0, I\sigma^2)$ to the model update. During the training, the moments accountant $\mathcal{M}$ is used to compute the T training steps' privacy budget consumption.

In this process, we need to bound the sensitivity of the weighted-average estimator $f_{\mathcal{E}}$ for per-user gradients $\Delta_{u,\mathcal{E}}^{t+1}$. We first consider the following simple estimator, with both sampling rates q_u for the user-level and q_e for the sensitive entity-level:

$$f_{\mathcal{E}}(U^t, E^t) = \frac{\sum_{u \in U^t} w_u \Delta_{u,\mathcal{E}}^{t+1}}{q_u W_u q_e W_e} \quad s.t. \quad \Delta_{u,\mathcal{E}}^{t+1} = \sum_{e \in E_u^t} w_e \Big(\sum_{s \text{ consists of } e} \Delta_{u,s} \Big) \tag{A.1}$$

where w_u and $w_e \in [0, 1]$ are weights associated with a user u and with a sensitive entity e. These weights capture the influence of a user and a sensitive entity to the model outcome.

$\Delta_{u,s}$ is the parameter gradients computed using a sensitive sentence s consisting of the sensitive entity e. In addition, $W_u = \sum_u w_u$ and $W_e = \sum_e w_e$.

The estimator $f_{\mathcal{E}}$ is unbiased to the sampling process; since $\mathbb{E}[\sum_{u \in U^t} w_u] = q_u W_u$ and $\mathbb{E}[\sum_{e \in E_u^t} w_e] = q_e W_e$. The sensitivity of the estimator $f_{\mathcal{E}}$ can be computed as: $\mathbb{S}(f_{\mathcal{E}}) = \max_{u',e'} \| f_{\mathcal{E}}(\{U^t \cup u', E^t \cup e'\}) - f_{\mathcal{E}}(\{U^t, E^t\}) \|_2$, where the added user u' can have arbitrary data and e' is an arbitrary sensitive entity.

Given that $\Delta_{u,\mathcal{E}}^{t+1}$ is $l_2(\beta)$-norm bounded, where β is the radius of the norm ball by replacing $\Delta_{u,\mathcal{E}}^{t+1}$ with $\Delta_{u,\mathcal{E}}^{t+1} \cdot \min\left(1, \frac{\beta}{\|\Delta_{u,\mathcal{E}}^{t+1}\|}\right)$, the sensitivity of $\mathbb{S}(f_{\mathcal{E}})$ is also bounded.

Lemma 3. *If for all users u we have $\|\Delta_{u,\mathcal{E}}^{t+1}\|_2 \leq \beta$, then $\mathbb{S}(f_{\mathcal{E}}) \leq \frac{(q_u|U|+1)\max(w_u)\beta}{q_u W_u \times q_e W_e}$.*

Proof. If for all users u we have $\|\Delta_{u,\mathcal{E}}^{t+1}\|_2 \leq \beta$, then $\mathbb{S}(f_{\mathcal{E}})$

$$\begin{aligned} &= \frac{\sum_{u \in U^t \cup u'} w_u[(\sum_{e \in E^t} w_e(\sum_{s \in S_{ue}^t} \Delta_{u,s}))]}{(q_u W_u \times q_e W_e)} + \frac{\sum_{u \in U^t \cup u'} w_u[w_{e'}(\sum_{s \in S_{ue'}^t} \Delta_{u,e'})]}{(q_u W_u \times q_e W_e)} \\ &- \frac{\sum_{u \in U^t} w_u[\sum_{e \in E^t} w_e(\sum_{s \in S_{ue}^t} \Delta_{u,s})]}{(q_u W_u \times q_e W_e)} \\ &\leq \frac{\sum_{u \in U^t \cup u'} w_u \beta}{q_u W_u \times q_e W_e} \leq \frac{(q_u|U|+1)\max(w_u)\beta}{q_u W_u \times q_e W_e} \end{aligned} \tag{A.2}$$

Consequently, Lemma 3 holds. □

By applying Lemma 3, given a hyper-parameter z, the noise scale σ for $f_{\mathcal{E}}$ is:

$$\sigma = z\mathbb{S}(f_{\mathcal{E}}) = \frac{z(q_u|U|+1)\max(w_u)\beta}{q_u W_u \times q_e W_e} \tag{A.3}$$

We show that this approach achieves (ϵ, δ)-UeDP, by applying the moments accountant $\mathcal{M}$ to bound the total privacy loss of T steps of the Gaussian mechanism with the noise $\mathcal{N}(0, I\sigma^2)$ in Theorem 1. However, this mechanism only uses sensitive entities detected by automatic toolkits to train the model ignoring a large number of extended sensitive entities. As a result, it introduces a loose sensitivity bound (Lemma 3) and affects our model utility.

A.3 Datasets and Data Processing

CONLL-2003 consists of Reuters news stories published between August 1996 and August 1997. CONLL-2003 is an NER dataset, where there are labels for four different types of named entities, including location, organization, person, and miscellaneous entities. These types of named entities are considered sensitive entities. In the CONLL-2003 dataset, there is no obvious user information; hence, we consider each document as a user consisting of multiple sentences in the next word prediction task.

AG dataset is a collection of news articles gathered from more than $2,000$ news sources by ComeToMyHead academic news search engine[2]. It is categorized into four classes: world, sport, business, and science/technology. Similar to the CONLL-2003 dataset, there is no user information in AG. To imitate a user indicator, we randomly divide news into different users based on Gaussian distribution. There are no named entities; thus, we apply pre-trained Spacy to find named entities and PII in the dataset. We choose different types of these named entities to be sensitive entities: organization, GPE (i.e., countries, cities, and states), location, and PII entities.

Our SEC dataset consists contract clauses collected from contracts submitted in SEC filings[3]. Since the contracts can be associated with a company ID, we use the ID as a user indicator. Similar to the AG dataset, we consider organization, GPE, location, and PII entities as sensitive entities to protect.

In addition, we conducted text classification on the AG dataset to further strengthen our observations. For text classification, the number of labels is not sufficient in the SEC dataset, and the labels do not exist in the CONLL-2003 dataset. Therefore, we do not utilize CONLL-2003 and SEC datasets for text classification in this study.

For data preprocessing, we changed all words to lower-case and removed punctuation marks. Fig. A.1 shows the distribution of the number of users and sentences in the CONLL-2003, AG, and SEC datasets. In the CONLL-2003 dataset, there is no obvious user

[2]`http://newsengine.di.unipi.it/`

[3]`https://www.sec.gov/edgar.shtml`

information; hence, we consider each document as a user consisting of multiple sentences. Like the CONLL-2003 dataset, in the AG dataset, there is no user information. Therefore, to imitate a user indicator, we randomly divide news into different users. The number of sentences per user follows a Gaussian distribution $\mathcal{N}(15, 2^2)$, i.e., there are 15 sentences per user on average, and the standard deviation is 2 sentences. In the SEC dataset, since the contracts can be associated with a company ID, we use the ID as a user indicator. The document related to the ID is considered to be that user's data.

A.4 Revisiting Word-level LDP Analysis in [120]

This section aims at revisiting privacy protection in [120] and describes a privacy accumulation issue over the embedding dimension. Then, we revise Theorems 1 and 2 in [120] and compare them with our approaches.

In [120], the authors aim at preserving the privacy of the extracted test representation from users while maintaining the good performance of the classifier, which is trained at a server by the data collected from users. To achieve the goal, they consider a word-level DP, that is, two inputs x and x' are adjacent if they differ by at most 1 word. Additionally, they introduce a DP noise layer r after a predefined feature extractor $f(x)$. To train a robust classifier at the server, they add the same level of noise as the test phase in the training process and optimize the classifier by minimizing the loss function as follows:

$$\mathcal{L}(x, y) = \mathcal{X}(C(f(x) + r), y) \tag{A.4}$$

where C is the classifier, y is the true label, and $\mathcal{X}$ is the cross entropy loss function.

The Laplace noise layer r is injected into the embedding $f(x)$ in which its coordinates $r = \{r_1, r_2, \ldots, r_k\}$ are random variables drawn from the Laplace distribution defined by $Lap(b)$ with $b = \frac{\Delta_f}{\epsilon}$, ϵ is the privacy budget, and Δ_f is the sensitivity of the extracted representation. Here, k is the dimension of $f(x)$.

Algorithm 6 describes how to derive DP-preserving representation from the feature extractor f. Note that x_s in the Algorithm 6 is a sentence (equivalent to x in our notation), which is considered to be sensitive and needs to be protected.

Revisting Theorems 1 and 2 in [120]. In the paper, the authors consider adjacent sentences differing by one word. Changing one word in x may change the entire embedding vector $f(x)$. Each element of $f(x)$ is normalized into the range $[0, 1]$ (Line 5, Algorithm 6), hence each element sensitivity of $f(x)$ is $\Delta_f = 1$, the noise is $Lap(\Delta_f/\epsilon)$. Therefore, each element of the embedding $f(x)$ consumes a privacy budget ϵ. Since the k elements of the embedding are derived from a single sensitive input x, applying the LDP mechanism $\mathcal{A}(.)$, i.e., $Lap(b)$, k times will consume the privacy budget $k \times \epsilon$. This follows the composition property in DP. Note that the k elements cannot be treated by using the parallel property in DP [57], since all of them are derived from a single (data) input x, NOT from k different inputs (k different data samples). Consequently, the privacy guarantees in Theorems 1 and 2 of [120] is $k\epsilon$-DP, instead of ϵ-DP as reported.

In their experimental results, e.g., Table 2 of [120], the approach could achieve almost the same (and even better) model utility with noiseless model given the extremely low $\epsilon = 0.05$ using BERT embeddings. As our analysis, the privacy budget in Theorems 1 and 2 is $k\epsilon$, instead of ϵ. Therefore, the proper privacy budget is at least $0.05 \times 768 = 38.4$. Similar results were reported through out the all in experiments. With this high value of the privacy budget, the word-level DP in [120] provides loose privacy protection.

Revisting Element-level DP in [120]. During our discussion with the authors of [120], the authors mentioned that their approach preserves a new notion of $(\epsilon, 0)$-element-level DP, i.e., two embeddings differ from one element, instead of a word-level DP. However, for the element-DP to hold, all the elements in the embedding $f(x)$ must be independent from each other, that is, changing one element will not result in changing any other element. If changing one element results in changing all the remaining elements, then element-DP will be suffered from the dimension of the embedding by following group privacy. In the

Algorithm 6 Differentially Private Neural Representation (DPNR) [120]

```
1: **Input**: Each sensitive input $x_s \in \mathbb{R}^d$, feature extractor $f$
2: **Parameters**: Dropout vector $I_n \in \{0,1\}^d$
3: Word dropout: $\tilde{x}_s \leftarrow x_s \odot I_n$, where $\odot$ performs a word-wise multiplication.
4: Extraction: $x_r \leftarrow f(\tilde{x}_s)$
5: Normalization: $x_r \leftarrow x_r - \min(x_r)/(\max(x_r) - \min(x_r))$
6: Perturbation: $\hat{x_r} \leftarrow x_r + r$, $r_i \sim Lap(b)$
7: **Output**: Perturbed representation $\hat{x_r}$.
```

current approach, changing one element means there is a change in the input data x to occur. Equivalently, using BERT, any change in the input data x will result in changing the whole embedding (all elements). Therefore, the condition of two neighboring embeddings only differing in only one element does NOT hold in theory and practice. Consequently, the introduced element-level DP does NOT hold at the level of $(\epsilon, 0)$-DP.

Our revising Theorems 1 and 2 in [120]. Based upon our analysis, we introduce revised versions of the Theorems 1 and 2 in [120], as follows.

Theorem 11. ***Revised Theorem 1 in [120].*** *Let the entries of the noise vector r be drawn from $Lap(b)$ with $b = \frac{\Delta_f}{\epsilon}$. The Algorithm 6 is $k\epsilon$-word-level DP, where k is dimension of the embedding $f(x)$.*

Proof. Each element of the embedding f is bounded in $[0, 1]$, so $\Delta_f = 1$ for each element. By adding random noise variables drawn from the Laplace $Lap(b)$ with $b = \frac{\Delta_f}{\epsilon}$ into each element of f, each element consumes ϵ/k privacy budget. Since the k elements of the embedding are derived from a single sensitive input x, applying the mechanism $Lap(b)$ k times on the k elements will consume the privacy budget $k\epsilon$. Therefore, the Algorithm 6 is $k\epsilon$-word-level DP. □

Theorem 12. *Given an input $x \in D$, suppose $\mathcal{A}(x) = f(x) + r$ is $k\epsilon$-word-level DP, let I_n with dropout rate μ be applied to x: $\tilde{x} = x \odot I_n$, then $\mathcal{A}(\tilde{x})$ is ϵ'-word level-DP, where $\epsilon' = \ln[(1-\mu)\exp(k\epsilon) + \mu]$.*

Proof. Suppose there are two adjacent inputs x_1 and x_2 that differ only in the i-th coordinate (word), say $x_{1i} = v$, $x_{2i} \neq v$. For arbitrary binary vector I_n, after dropout, $\tilde{x}_1 = x_1 \odot I_n$, $\tilde{x}_2 = x_2 \odot I_n$, there are two possible cases, i.e., $I_{ni} = 0$ and $I_{ni} = 1$.

If $I_{ni} = 0$: Since x_1 and x_2 differ only in i-th coordinate, after dropout $\tilde{x}_{1i} = \tilde{x}_{2i} = 0$, hence $x_1 \odot I_n = x_2 \odot I_n$. Then $Pr\{\mathcal{A}(x_1 \odot I_n) = S\} = Pr\{\mathcal{A}(x_2 \odot I_n) = S\}$.

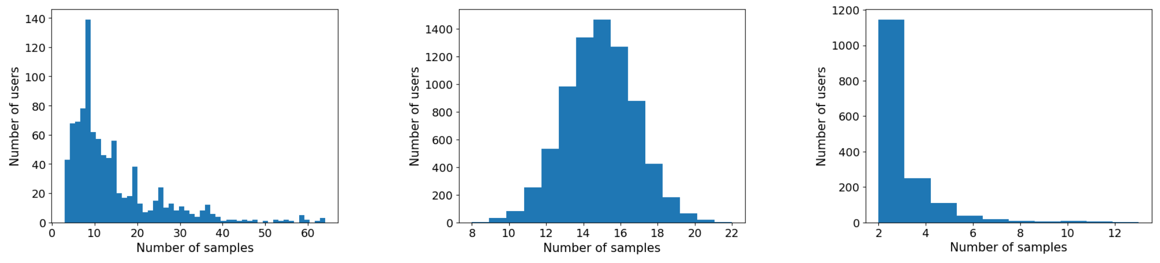
If $I_{ni} = 1$: Since x_1 and x_2 differ only in i-th coordinate, after dropout $\tilde{x}_{1i} = v$, and $\tilde{x}_{2i} \neq v$. Since $\mathcal{A}(x)$ is $k\epsilon$-word level-DP, then $Pr\{\mathcal{A}(x_1 \odot I_n) = S\} \leq \exp(k\epsilon) Pr\{\mathcal{A}(x_2 \odot I_n) = S\}$.

Combining these two cases, and $Pr[I_{ni} = 0] = \mu$, we have:

$$\begin{aligned}
&Pr\{\mathcal{A}(x_1 \odot I_n) = S\} \\
&= \mu Pr\{\mathcal{A}(x_1 \odot I_n) = S\} + (1-\mu) Pr\{\mathcal{A}(x_1 \odot I_n) = S\} \\
&\leq \mu Pr\{\mathcal{A}(x_2 \odot I_n) = S\} + (1-\mu)\exp(k\epsilon) Pr\{\mathcal{A}(x_2 \odot I_n) = S\} \\
&= [(1-\mu)\exp(k\epsilon) + \mu] Pr\{\mathcal{A}(x_2 \odot I_n) = S\} \\
&= \exp\Big(\ln[(1-\mu)\exp(k\epsilon) + \mu]\Big) Pr\{\mathcal{A}(x_2 \odot I_n) = S\} \qquad \text{(A.5)}
\end{aligned}$$

Therefore, after dropout, the privacy budget is $\epsilon' = \ln[(1-\mu)\exp(k\epsilon) + \mu]$. □

Comparison with UeDP. Apart from the privacy accumulation over the embedding dimension, in [120], during training the model, the Laplace or Gaussian noise is drawn at every training iteration. Therefore, the model accesses the raw data at every iteration. As a result, the privacy budget at the training phase is accumulated over the number of training iterations, which can be a large number causing an exploded privacy budget in training. [120] focuses on protecting privacy at the inference time and use the noise in the training phase to obtain a more robust model without considering training data privacy. This is different from our goal to protect users and sensitive entities of training data, which is a more challenging task. Our UeDP-preserving model can be deployed to the end-users for a direct use in the inference phase, without demanding that the end-users send their



((**a**)) CONLL-2003 dataset ((**b**)) AG dataset ((**c**)) SEC dataset

Figure A.1 Distribution of users and sentences.

data embedding to our server; therefore offering a more rigorous privacy protection and better usability. In addition to this, our approach offers more rigorous DP budget bounds compared with the DPNR algorithm in [120], since DPNR consumes large DP budgets that is proportional to the commonly large dimension of the embedding k.

A.5 Supplemental Experimental Results for UeDP

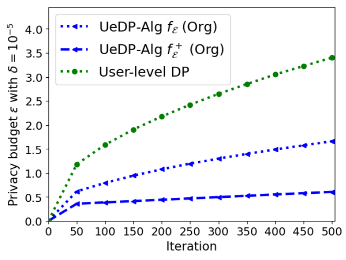
A.6 Proof of Lemma 2

Proof. Let v be the l-bit binary vector representation of a numerical feature a in the feature vector e. Given two neighboring vectors v and $v^{\neq i}$ that differ only at a bit i, the l_1-norm influence $\mathcal{I}_i$ of the bit i captures the magnitude by which the bit i can change the LDP-preserving feature a' in the worst case. We quantify $\mathcal{I}_i$ as follows.

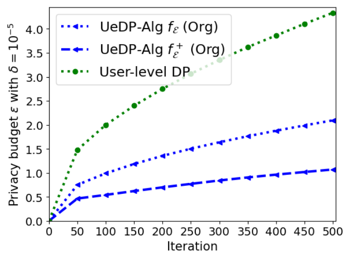
Let us denote $v_0 v_1 \ldots v_{l-1}$ as the binary representation v of a, where v_0 is the value of a sign bit ($v_0 = 0$ if $a \geq 0$ and $v_0 = 1$ if $a < 0$), and $\{v_j\}_{j=1}^{l-1} \in \{0, 1\}$ is the value of exponent bits and fraction bits. Since we consider the encoding function is lossless, the decoded feature of v is:

$$a = \mathcal{E}(v) = (2v_0 - 1) \sum_{j=1}^{l-1} v_j \times 2^{m-j} \tag{A.6}$$

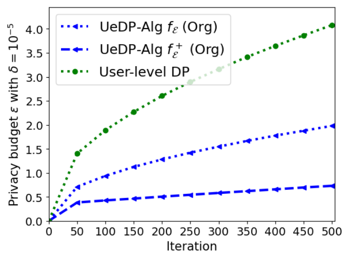
Similarly, the decoded feature of $v^{\neq i}$ is $a' = \mathcal{E}(v^{\neq i}) = (2v'_0 - 1) \sum_{j=1}^{l-1} v'_j \times 2^{m-j}$. Note that, v and $v^{\neq i}$ differ at the bit i; or, $\forall i, j \in [0, l-1], j \neq i, v_j = v'_j$ and $v_i \neq v'_i$. Let us consider different locations of the bit i to quantify its influence.



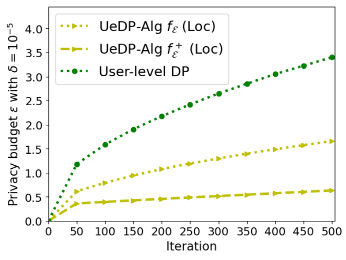
((**a**)) CONLL-2003-organization entities



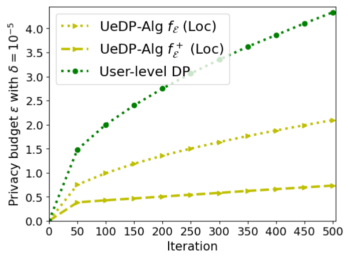
((**b**)) AG-organization entities



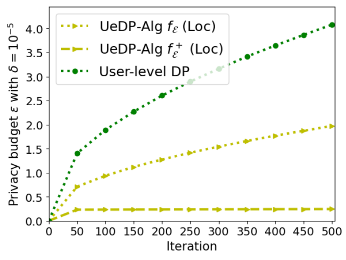
((**c**)) SEC-organization entities



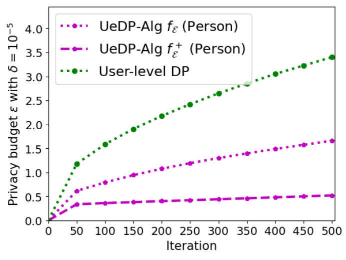
((**d**)) CONLL-2003-location entities



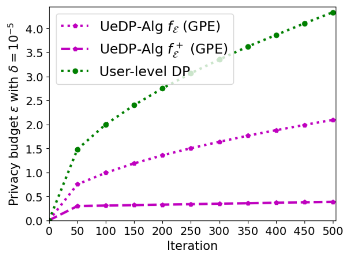
((**e**)) AG-location entities



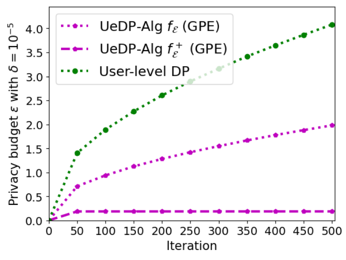
((**f**)) SEC-location entities



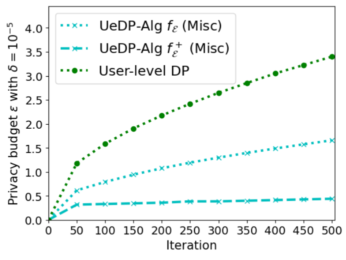
((**g**)) CONLL-2003-person entities



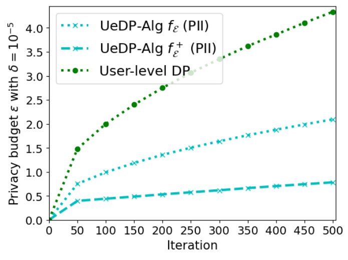
((**h**)) AG-GPE entities



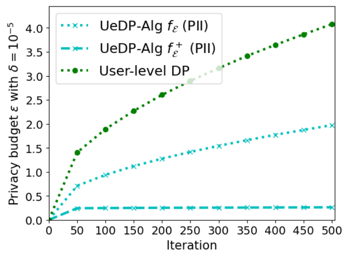
((**i**)) SEC-GPE entities



((**j**)) CONLL-2003-miscellanous entities



((**k**)) AG-PII entities



((**l**)) SEC-PII entities

Figure A.2 Privacy budget of UeDP-Alg $f_{\mathcal{E}}$, UeDP-Alg $f_{\mathcal{E}^+}$, and User-level DP as a function of iterations in CONLL-2003, AG, and SEC datasets.

- If i is an exponent or a fraction bit ($i \in [1, l-1]$), we have $v'_j = v_j$ with $j \neq i$. Therefore, $\mathcal{I}_i = \max_v \parallel \mathcal{E}(v) - \mathcal{E}(v^{\neq i}) \parallel_1 = \max_v \|(2v_0 - 1)\Big(\sum_{j=1, j\neq i}^{l-1} v_j 2^{m-j} + v_i 2^{m-i}\Big) - (2v'_0 - 1)\Big(\sum_{j=1,j\neq i}^{l-1} v'_j 2^{m-j} + v'_i 2^{m-i}\Big)\|_1 = \max_v \|(2v_0 - 1)(v_i - v'_i)2^{m-i}\|_1 \leq \max_v \Big(|(2v_0 - 1)| \|v_i - v'_i\|_1 2^{m-i}\Big)$. Since $v_0, v_i, v'_i \in \{0, 1\}$ and v_i and v'_i are different, $\max_v \Big(|(2v_0 - 1)| \|v_i - v'_i\|_1 2^{m-i}\Big) = 2^{m-i}$. Therefore, $\mathcal{I}_i \leq 2^{m-i}$ if i is an exponent or a fraction bit.

- If i is a sign bit ($i = 0$), we have $\mathcal{I}_i = \max_v \parallel \mathcal{E}(v) - \mathcal{E}(v^{\neq 0}) \parallel_1 = \max_v \|(2v_0 - 2v'_0) \sum_{j=1}^{l-1} v_j 2^{m-j}\|_1 \leq \max_v \Big(|2v_0 - 2v'_0| \| \sum_{j=1}^{l-1} v_j 2^{m-j}\|_1\Big)$. Since $\forall j \in [1, l-1]$: $v_0, v_j, v'_j \in \{0, 1\}$ and v_0 and v'_0 are different, $\max_v \Big(|2v_0 - 2v'_0| \| \sum_{j=1}^{l-1} v_j 2^{m-j}\|_1\Big) = 2\sum_{j=1}^{l-1} 2^{m-j}$. Since $2^1 + \ldots + 2^{l-2} = 2^{l-1} - 2$ [3], $2\sum_{j=1}^{l-1} 2^{m-j} = 2 \times 2^{m-(l-1)}(2^0 + 2^1 + \ldots + 2^{l-2}) = 2^{m+1-(l-1)}(2^{l-1} - 1) \leq 2^{m+1}$. Therefore, $\mathcal{I}_i \leq 2^{m+1}$ if i is a sign bit.

Consequently, Lemma 2 holds.

$\square$

A.7 Lemma 4 and Its Proof

Lemma 4. *ScalableRR l_1-sensitivity. Given two neighboring vectors v and $v^{\neq i}$ that differ only at a bit i, the l_1-sensitivity Δ_i captures the magnitude by which $\mathcal{M}(v, i)$ can change v given the decoding function $\mathcal{E}(\cdot)$ in the worst case, as follows:*

$$\forall i \in [0, l-1] : \Delta_i = \max_v \|\mathcal{E}\big(\mathcal{M}(v, i)\big) - \mathcal{E}\big(\mathcal{M}(v^{\neq i}, i)\big)\|_1 \tag{A.7}$$

where $\mathcal{M}(v, i)$ implies that ScalableRR is applied to randomize the bit i while keeping all other bits in v unchanged.

$\forall i \in [0, l-1]$, Δ_i is bounded as follows:

$$\Delta_i \leq \begin{cases} 2^{m+1}, & \textit{if } i \textit{ is a sign bit} \\ 2^{m-i}, & \textit{if } i \textit{ is an exponent or fraction bit} \end{cases} \tag{A.8}$$

Proof. Randomizing a bit i given v and $v^{\neq i}$ results in a smaller l_1-distance compared with $|\mathcal{E}(v) - \mathcal{E}(v^{\neq i})|$, i.e., $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))| \leq |\mathcal{E}(v) - \mathcal{E}(v^{\neq i})|$. This condition satisfies for all i in v and $v^{\neq i}$; therefore, we have $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))| \leq \Delta_i \leq |\mathcal{E}(v) - \mathcal{E}(v^{\neq i})| \leq \mathcal{I}_i$. Consequently, $\Delta_i \leq \mathcal{I}_i$ and from $\mathcal{I}_i$ in Lemma 2, Lemma 4 holds.

□

A.8 Proof of Theorem 2

Proof. For LDP condition to hold, we have:

$$
\begin{aligned}
&\frac{P(\mathcal{M}(v) = v_z)}{P(\mathcal{M}(\widetilde{v}) = v_z)} \leq \prod_{i=0}^{rl-1} \frac{P(\mathcal{M}(v_i) = v_{z,i})}{P(\mathcal{M}(v_i^{\neq i}) = v_{z,i})} \\
&= \prod_{i=0}^{rl-1} \frac{P(\mathcal{M}(v_i) = 0 | v_i = 1) P(\mathcal{M}(v_i) = 1 | v_i = 0)}{P(\mathcal{M}(v_i) = 1 | v_i = 1) P(\mathcal{M}(v_i) = 0 | v_i = 0)} \\
&= \prod_{i=0}^{rl-1} \left(\alpha^2 \exp(2\epsilon_X \frac{i\%l}{l})\right) = \prod_{i=0}^{l-1} \left(\alpha^2 \exp(2\epsilon_X \frac{i}{l})\right)^r \\
\Leftrightarrow &\ln\left(\frac{P(\mathcal{M}(v) = v_z)}{P(\mathcal{M}(\widetilde{v}) = v_z)}\right) \leq \sum_{i=0}^{l-1} \ln\left[\left(\alpha^2 \exp(2\epsilon_X \frac{i}{l})\right)^r\right] \\
&= \sum_{i=0}^{l-1} r \ln\left(\alpha^2 \exp(2\epsilon_X \frac{i}{l})\right) = 2rl \ln\alpha + 2r\epsilon_X \sum_{i=0}^{l-1} \frac{i}{l} = 2rl \ln\alpha + r\epsilon_X(l-1)
\end{aligned}
$$

To satisfy the ϵ_X-LDP condition, $\ln\left(\frac{P(\mathcal{M}(v)=v_z)}{P(\mathcal{M}(\widetilde{v})=v_z)}\right)$ needs to be bounded by ϵ_X. Therefore, we have:

$$
2rl \ln\alpha + r\epsilon_X(l-1) = \epsilon_X \Leftrightarrow \alpha = \exp(\frac{\epsilon_X - r\epsilon_X(l-1)}{2rl}) \tag{A.9}
$$

□

A.9 Proof of Theorem 3

Proof. In this section, we aim at finding a closed-form solution of α so that v' preserves ϵ_X-LDP given v. In other words, we need to find α in Eq. 3.13 so that LDP condition holds given the total privacy budget ϵ_X, as follows.

Let us denote v_a as the binary encoding vector of the feature a. We first consider the privacy loss of bits i across all features $a \in [1, r]$, as follows:

$$\forall i \in [0, l-1] : \prod_{a \in [1,r]} \frac{P(\mathcal{M}(v_{a,i}) = v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i}) = v_{z,i})} \leq \exp\big(\sum_{a \in [1,r]} \frac{\epsilon_i |\mathcal{E}(\mathcal{M}(v_a, i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i}, i))|}{\Delta_i} \big) \tag{A.10}$$

$$= \exp\Big(\frac{r\epsilon_i}{\Delta_i} \times \frac{1}{r}\big(\sum_{a \in [1,r]} |\mathcal{E}(\mathcal{M}(v_a, i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i}, i))| \big) \Big)$$

It is challenging to find a closed-form solution of α to satisfy Eq. A.10, since it is infeasible to precisely quantify the average term $\frac{1}{r}\big(\sum_{a \in [1,r]} |\mathcal{E}(\mathcal{M}(v_a, i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i}, i))| \big)$. Different features a may have different values of $|\mathcal{E}(\mathcal{M}(v_a, i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i}, i))|$. In other words, the average term is data-dependent. To address this problem, we use Hoeffding's inequality [2] to upper-bound the average term using its expectation value, denoted $\mathbb{E}$, given a positive error term ρ_i with a broken probability δ for the bits i. Let us denote $X_{a,i} = |\mathcal{E}(\mathcal{M}(v_a, i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i}, i))|$ where $X_{a,i} \in [0, \Delta_i]$. The Hoeffding's inequality with $\rho_i \mathbb{E}(X_{a,i}) \geq 0$ is as follows:

$$\begin{aligned}
\forall i \in [0, l-1] : P\Big(\big(\frac{1}{r} \sum_{a=1}^{r} X_{a,i}\big) - \mathbb{E}(X_{a,i}) \geq \rho_i \mathbb{E}(X_{a,i}) \Big) \\
= P\Big(\frac{1}{r} \sum_{a=1}^{r} \big(X_{a,i} - \mathbb{E}(X_{a,i})\big) \geq \rho_i \mathbb{E}(X_{a,i}) \Big) \\
\leq \exp\Big(- \frac{2r\big(\rho_i \mathbb{E}(X_{a,i})\big)^2}{\frac{1}{r}\sum_{a=1}^{r} \Delta_i^2} \Big) = \exp\Big(- \frac{2r\rho_i^2 (\mathbb{E}(X_{a,i}))^2}{\Delta_i^2} \Big) = \delta
\end{aligned} \tag{A.11}$$

From Eq. A.11, for the bits i we have: $\frac{1}{r}\sum_{a\in[1,r]}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))| < \mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|(1+\rho_i)$ with the broken probability δ. From that, we can derive a closed-form solution for α to effectively and precisely bound the privacy loss of the bits i across all features a. From Eq. A.10, we have $\forall i \in [0, l-1]$:

$$\prod_{a\in[1,r]} \frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})} < \exp\Big(\frac{r\epsilon_i}{\Delta_i}\times \mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|\big(1+\rho_i\big)\Big) \tag{A.12}$$

There are two possible cases: **(1)** $\frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\geq 1$, and **(2)** $0<\frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}<1$. Since Δ_i is the l_1-sensitivity, we have $\Delta_i/\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|\geq 1$.

In the first case, we have $\forall i \in [0, l-1]$:

$$\prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\big) \leq \prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i})=v_{z,i}}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|}}$$
$$< \exp\big((1+\rho_i)r\epsilon_i\big) \tag{A.13}$$

In the second case, $\forall i \in [0, l-1]$ we have:

$$\prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i})=v_{z,i}}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\big) \leq \prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}{P(\mathcal{M}(v_{a,i})=v_{z,i})}\big)$$
$$\leq \prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}{P(\mathcal{M}(v_{a,i})=v_{z,i}}\big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|}} < \exp\big((1+\rho_i)r\epsilon_i\big) \tag{A.14}$$

From Eqs. A.13 and A.14, for the whole vector, we have:

$$\frac{P(\mathcal{E}(\mathcal{M}(v))=\mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v}))=\mathcal{E}(v_z))} = \prod_{i\in[0,l-1]}\prod_{a\in[1,r]} \frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})} \tag{A.15}$$
$$\leq \prod_{i\in[0,l-1]}\prod_{a\in[1,r]} \big(\frac{P(\mathcal{M}(v_{a,i})=v_{z,i})}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|}} < \exp\big(\sum_{i=0}^{l-1}(1+\rho_i)r\epsilon_i\big)$$

The positive and tiny error term ρ_i is different across sign, exponent, and fraction bits. Therefore, we simplify the search of α in Eq. A.15 by using the upper-bound of the error terms across bits: $\rho = \lceil \rho_i \rceil$, without notably affecting privacy-utility trade-offs, as follows:

$$\prod_{i\in[0,l-1]} \prod_{a\in[1,r]} \Big(\frac{P(\mathcal{M}(v_{a,i}) = v_{z,i}}{P(\mathcal{M}(v_{a,i}^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|}} \leq \exp\Big((1-\rho)\sum_{i=0}^{l-1} r\epsilon_i\Big) \quad \text{(A.16)}$$

A temperature α that satisfies the LDP condition in Eq. A.16 will also satisfy the LDP condition in Eq. A.15 since $\exp\big((1-\rho)\sum_{i=0}^{l-1} r\epsilon_i\big) < \exp\big(\sum_{i=0}^{l-1}(1+\rho_i)r\epsilon_i\big)$. Note that $0 < (1-\rho) \leq 1$ with typical numbers of features $r \geq 100$ and small broken probability $\delta = 1e-5$. Eq. A.16 is equivalent to

$$\prod_{i\in[0,l-1]} \prod_{a\in[1,r]} \Big(\frac{P(\mathcal{M}(v_{a,i})=v_{z,i}}{P(\mathcal{M}(v_{a,i}^{\neq i})=v_{z,i})}\Big)^{\frac{\Delta_i}{(1-\rho)\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i))-\mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|}} \leq \exp\Big(\sum_{i=0}^{l-1} r\epsilon_i\Big) = \exp\Big(\sum_{i=0}^{rl-1}\epsilon_i\Big) \quad \text{(A.17)}$$

If we force the total privacy loss $\epsilon = \sum_{i=0}^{rl-1} \epsilon_i$ to be equal to a user-predefined budget ϵ_X, we can identify a closed form solution of the temperature α guaranteeing that the randomization of the vector v is ϵ_X-LDP, i.e., $\frac{P(\mathcal{E}(\mathcal{M}(v))=\mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v}))=\mathcal{E}(v_z))} \leq \exp(\sum_{i=0}^{rl-1}\epsilon_i) = \exp(\epsilon_X)$, with a small broken probability $l \times \delta$ (i.e., there are l error terms ρ_i and each of which has a broken probability δ).

To find α satisfying Eq. A.17, first, we need to calculate $\mathbb{E}|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|$. Given the worse case of v and $v^{\neq i}$, there are four possible cases of $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))|$:

- If $\mathcal{M}(v_i) = 1$ and $\mathcal{M}(v_i^{\neq i}) = 1$, then $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))| = 0$.
- If $\mathcal{M}(v_i) = 0$ and $\mathcal{M}(v_i^{\neq i}) = 0$, then $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))| = 0$.
- If $\mathcal{M}(v_i) = 1$ and $\mathcal{M}(v_i^{\neq i}) = 0$, then $|\mathcal{E}(\mathcal{M}(v,i)) - \mathcal{E}(\mathcal{M}(v^{\neq i},i))| = \Delta_i$. This happens with the probability $P(\mathcal{M}(v_i) = 1, \mathcal{M}(v_i^{\neq i}) = 0)$. To compute this probability, we use marginal probability and Bayes' theorem, as:

$$
\begin{aligned}
&P(\mathcal{M}(v_i)=1,\mathcal{M}(v_i^{\neq i})=0)=P\Big(\mathcal{M}(v_i)=1,\mathcal{M}(v_i^{\neq i})=0,v_i=1,v_i^{\neq i}=0\Big)\\
&+P\Big(\mathcal{M}(v_i)=1,\mathcal{M}(v_i^{\neq i})=0,v_i=0,v_i^{\neq i}=1\Big)\\
&=P\Big(\mathcal{M}(v_i)=1|v_i=1\Big)P(\mathcal{M}(v_i^{\neq i})=0|v_i^{\neq i}=0)P(v_i^{\neq i}=0)\\
&+P\Big(\mathcal{M}(v_i)=1|v_i=0\Big)P(\mathcal{M}(v_i^{\neq i})=0|v_i^{\neq i}=1)P(v_i^{\neq i}=1)\\
&=p_i^2P(v_i^{\neq i}=0)+q_i^2P(v_i^{\neq i}=1)
\end{aligned}
\tag{A.18}
$$

- If $\mathcal{M}(v_i)=0$ and $\mathcal{M}(v_i^{\neq i}))=1$, then $|\mathcal{E}(\mathcal{M}(v_i))-\mathcal{E}(\mathcal{M}(v_i^{\neq i}))|=\Delta_i$. This happens with the probability $P(\mathcal{M}(v_i)=0,\mathcal{M}(v_i^{\neq i})=1)$. To compute this probability, we use marginal probability and Bayes' theorem, as:

$$
\begin{aligned}
&P(\mathcal{M}(v_i)=0,\mathcal{M}(v_i^{\neq i})=1)=P\Big(\mathcal{M}(v_i)=0,\mathcal{M}(v_i^{\neq i})=1,v_i=1,v_i^{\neq i}=0\Big)\\
&+P\Big(\mathcal{M}(v_i)=0,\mathcal{M}(v_i^{\neq i})=1,v_i=0,v_i^{\neq i}=1\Big)\\
&=q_i^2P(v_i^{\neq i}=0)+p_i^2P(v_i^{\neq i}=1)
\end{aligned}
\tag{A.19}
$$

Consequently, the expectation $\mathbb{E}|\mathcal{E}(\mathcal{M}(v,i))-\mathcal{E}(\mathcal{M}(v^{\neq i},i))|$ is computed as follows:

$$
\begin{aligned}
&\mathbb{E}|\mathcal{E}(\mathcal{M}(v,i))-\mathcal{E}(\mathcal{M}(v^{\neq i},i))|\\
&=\Big(p_i^2P(v_i^{\neq i}=0)+q_i^2P(v_i^{\neq i}=1)\Big)\Delta_i+\Big(q_i^2P(v_i^{\neq i}=0)+p_i^2P(v_i^{\neq i}=1)\Big)\Delta_i\\
&=(p_i^2+q_i^2)\Delta_i
\end{aligned}
\tag{A.20}
$$

From Eqs. A.16, and A.20, we have:

$$\frac{P(\mathcal{E}(\mathcal{M}(v))=\mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v}))=\mathcal{E}(v_z))}\leq\prod_{i=0}^{rl-1}\Big(\frac{P(\mathcal{M}(v_i)=v_{z,i})}{P(\mathcal{M}(v_i^{\neq i})=v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v,i))-\mathcal{E}(\mathcal{M}(v^{\neq i},i))|}}$$
$$=\prod_{i=0}^{rl-1}\Big(\tfrac{P(\mathcal{M}(v_i)=0|v_i=1)P(\mathcal{M}(v_i)=1|v_i=0)}{P(\mathcal{M}(v_i)=1|v_i=1)P(\mathcal{M}(v_i)=0|v_i=0)}\Big)^{\frac{1}{(1-\rho)(p_i^2+q_i^2)}}=\prod_{i=0}^{l-1}\big(\alpha^2\exp(2\epsilon_X\frac{i}{l})\big)^{\frac{r}{(1-\rho)(p_i^2+q_i^2)}} \tag{A.21}$$

Taking the natural logarithm of two sides of Eq. A.21:

$$\ln\frac{P(\mathcal{E}(\mathcal{M}(v))=\mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v}))=\mathcal{E}(v_z))}\leq\sum_{i=0}^{l-1}\ln\Big(\alpha^2\exp(2\epsilon_X\frac{i}{l})\Big)^{\frac{r}{(1-\rho)(p_i^2+q_i^2)}}$$
$$=\sum_{i=0}^{l-1}\Big(\frac{r}{(1-\rho)(p_i^2+q_i^2)}\ln\big(\alpha^2\exp(2\epsilon_X\frac{i}{l})\big)\Big) \tag{A.22}$$

Let us bound the logarithm in Eq. A.22 using the inequality:

$$\ln(\mathcal{X})\leq\mathcal{X}-1\text{ for }\mathcal{X}>0 \tag{A.23}$$

The proof of Eq. A.23 is as follows. Let $\mathcal{X}>0$, we define $h(\mathcal{X})=\ln(\mathcal{X})-\mathcal{X}+1$. We have: $h'(\mathcal{X})=\frac{1}{\mathcal{X}}-1=0\Leftrightarrow\mathcal{X}=1$, and since $h''(\mathcal{X})=-\frac{1}{\mathcal{X}^2}<0,\forall\mathcal{X}>0$, we get the maximal point at $\mathcal{X}=1$. We also have: $\lim_{\mathcal{X}\to0+}h(\mathcal{X})=-\infty=\lim_{\mathcal{X}\to\infty}h(\mathcal{X})$. Therefore, $\mathcal{X}=1$ is the global maximal point and than $\forall\mathcal{X}>0,h(\mathcal{X})\leq h(\mathcal{X}=1)=0$, so $\ln(\mathcal{X})-\mathcal{X}+1\leq0$. Therefore, Eq. A.23 does hold.

Note that, to simultaneously satisfy the randomization probabilities $p_i=\frac{1}{1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)}\geq0$ and $q_i=\frac{\alpha\exp(\frac{i\%l}{l}\epsilon_X)}{1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)}\geq0$ in Eq. 3.13, we need to have: *(i)* $1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)\geq0$ and *(ii)* $\alpha\exp(\frac{i\%l}{l}\epsilon_X)\geq0$. Since $\exp(\frac{i\%l}{l}\epsilon_X)\geq0$ is always true, from *(i)*, $\alpha\geq-\exp(-\frac{i\%l}{l}\epsilon_X)$, and from *(ii)*, $\alpha\geq0$. Therefore, $\alpha\geq0$ is necessary to satisfy the condition $p_i\geq0$ and $q_i\geq0$. To apply Eq. A.23 into Eq. A.22, we need to have $a=\alpha^2\exp(2\epsilon_X\frac{i}{l})>0\Rightarrow\alpha\neq0$. As a result, we have that

$$\alpha>0 \tag{A.24}$$

Applying Eq. A.23 into Eq. A.22 where $\mathcal{X}=\alpha^2\exp(2\epsilon_X\frac{i}{l})$:

$$\ln \frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} \leq \sum_{i=0}^{l-1} \Big(\frac{r}{(1-\rho)(p_i^2+q_i^2)} \ln \big(\alpha^2 \exp(2\epsilon_X \frac{i}{l}) \big) \Big)$$

$$\leq \sum_{i=0}^{l-1} \frac{r\alpha^2 \exp(2\epsilon_X \frac{i}{l})}{(1-\rho)(p_i^2+q_i^2)} - \sum_{i=0}^{l-1} \frac{r}{(1-\rho)(p_i^2+q_i^2)} \tag{A.25}$$

To bound the logarithm in Eq. A.25, we use: $p_i^2 + q_i^2 = p_i^2 + (1-p_i)^2 = \Big(\frac{1}{1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)}\Big)^2 + \Big(\frac{\alpha\exp(\frac{i\%l}{l}\epsilon_X)}{1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)}\Big)^2 = \frac{1+\Big(\alpha\exp(\frac{i\%l}{l}\epsilon_X)\Big)^2}{\Big(1+\alpha\exp(\frac{i\%l}{l}\epsilon_X)\Big)^2} \leq 1$, and $p_i^2+q_i^2 = p_i^2 + (1-p_i)^2 \geq \frac{(p_i+1-p_i)^2}{2} = \frac{1}{2}$ (In fact, $\forall a, b : a^2 + b^2 \geq \frac{(a+b)^2}{2} \Leftrightarrow (a-b)^2 \geq 0$, which is true). Note that, from Eq. A.20, we have that $\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(\mathcal{M}(v_a,i)) - \mathcal{E}(\mathcal{M}(v_a^{\neq i},i))|} = \frac{1}{p_i^2+q_i^2} \geq 1$. Applying these inequalities in Eq. A.25, we obtain:

$$\ln \frac{P(\mathcal{E}(\mathcal{M}(v)) = \mathcal{E}(v_z))}{P(\mathcal{E}(\mathcal{M}(\tilde{v})) = \mathcal{E}(v_z))} \leq \sum_{i=0}^{l-1} \frac{r\alpha^2 \exp(2\epsilon_X \frac{i}{l})}{(1-\rho)(p_i^2+q_i^2)} - \sum_{i=0}^{l-1} \frac{r}{(1-\rho)(p_i^2+q_i^2)}$$

$$\leq \sum_{i=0}^{l-1} \frac{2r\alpha^2 \exp(2\epsilon_X \frac{i}{l})}{1-\rho} - \sum_{i=0}^{l-1} \frac{r}{1-\rho} = \sum_{i=0}^{l-1} \frac{2r\alpha^2 \exp(2\epsilon_X \frac{i}{l})}{1-\rho} - \frac{rl}{1-\rho} \tag{A.26}$$

For LDP condition to hold under the user-predefined privacy budget ϵ_X, from Eq. A.26 we have that

$$\sum_{i=0}^{l-1} \frac{2r\alpha^2 \exp(2\epsilon_X \frac{i}{l})}{1-\rho} - \frac{rl}{1-\rho} = \epsilon_X \Leftrightarrow \alpha^2 = \frac{rl + (1-\rho)\epsilon_X}{2r\sum_{i=0}^{l-1}\exp(2\epsilon_X \frac{i}{l})}$$

$$\Leftrightarrow \alpha = \sqrt{\frac{rl + (1-\rho)\epsilon_X}{2r\sum_{i=0}^{l-1}\exp(2\epsilon_X \frac{i}{l})}} \text{ (since } \alpha > 0 \text{ as in Eq. A.24)} \tag{A.27}$$

Next, we need to find the upper-bound of ρ_i. From Eqs. A.11 and A.20, then we have:

$$\exp\Big(-\frac{2r\rho_i^2(\mathbb{E}X_{a,i})^2}{\Delta_i^2}\Big) = \delta \Leftrightarrow \exp\big(-2r\rho_i^2(p_i^2+q_i^2)^2\big) = \delta \Leftrightarrow \rho_i = \frac{1}{p_i^2+q_i^2}\sqrt{\frac{-\ln\delta}{2r}} \tag{A.28}$$

Since $\forall i$, $p_i^2 + q_i^2 \geq \frac{1}{2}$, we have that

$$\rho = \lceil \rho_i \rceil = 2\sqrt{\frac{-\ln\delta}{2r}} \tag{A.29}$$

With $l\delta = 1e-5$: for $r = 1,000$, $\rho = 0.16$ and for $r = 10,000$, $rho = 0.05$. Therefore, we can consider ρ is a tiny and upper-bounded term, given a modest broken probability δ. Note that, the broken probability for the whole rl-bit vector is $l\delta$.

In summary, we have that given $\alpha = \sqrt{\frac{rl+(1-\rho)\epsilon_X}{2r\sum_{i=0}^{l-1}\exp(2\epsilon_X \frac{i}{l})}}$, the ScalableRR mechanism satisfies ϵ_X-LDP and $\rho \leq 2\sqrt{\frac{-\ln\delta}{2r}}$ is a tiny and upper-bound term, with a small broken probability $l \times \delta$. Consequently, Theorem 3 holds.

□

A.10 Proof of Theorem 4

Proof. ScalableRR (i.e., $\mathcal{M}$) can be mapped to the general form (Eq. 3.3), as follows:

$$\forall a \in e : P(a'|a) = P(v'|v) = \prod_{i=0}^{l-1} P(v'_i|v_i) \tag{A.30}$$

where $v'_i = \mathcal{M}(v_i)$ and $P(v'_i|v_i)$ is given by Eq. 3.13.

As shown in Theorem 3, $\alpha = \sqrt{\frac{rl+(1-\rho)\epsilon_X}{2r\sum_{i=0}^{l-1}\exp(2\epsilon_X \frac{i}{l})}}$. Given a privacy budget ϵ_X, when r increases, α decreases.

Let $P(a'|a)$ and $P_d(a'|a)$ be the randomization probabilities when the number of features are r and $d \times r$, respectively. Let α and α_d be the values of α when the number of features are r and $d \times r$, respectively. We denote $t_i = \exp\left(\epsilon_X \frac{i\%l}{l}\right)$.

KL-divergence. Following Eq. 3.13, for each bit i, we have:

$$\begin{aligned} KL(P||P_d)_i &= \sum_a P(a)\log\frac{P(a)}{P_d(a)} = p_i \log\frac{1+\alpha_d t_i}{1+\alpha t_i} + q_i \log\left(\frac{\alpha t_i}{\alpha_d t_i} \times \frac{1+\alpha_d t_i}{1+\alpha t_i}\right) \\ &= \log\left(\frac{1+\alpha_d t_i}{1+\alpha t_i}\right) + q_i \log(\frac{\alpha}{\alpha_d}) \end{aligned} \tag{A.31}$$

Therefore, at the feature level, the KL-divergence is:

$$KL(P||P_d) = \sum_{i=0}^{l-1}\left(\log\left(\frac{1+\alpha_d t_i}{1+\alpha t_i}\right) + q_i \log(\frac{\alpha}{\alpha_d})\right) \tag{A.32}$$

L_2**-norm.** We have the L_2-norm between P and P_d:

$$
\begin{aligned}
L_2^2(P,P_d) &= \int_{a'} (P(a'|a) - P_d(a'|a))^2 da' \\
&= \int_{v'_0} \cdots \int_{v'_{l-1}} \Big(\prod_{i=0}^{l-1} p(v'_i|v_i) - \prod_{i=0}^{l-1} p_d(v'_i|v_i)\Big)^2 dv'_0 \dots dv'_{l-1} \\
&= \int_{v'_0} \cdots \int_{v'_{l-1}} \Big(\prod_{i=0}^{l-1} p(v'_i|v_i)^2 - 2\prod_{i=0}^{l-1} p(v'_i|v_i)p_d(v'_i|v_i) + \prod_{i=0}^{l-1} p_d(v'_i|v_i)^2\Big) dv'_0 \dots dv'_{l-1} \\
&= \prod_{i=0}^{l-1} \int_{v'_i} p(v'_i|v_i)^2 dv'_i + \prod_{i=0}^{l-1} \int_{v'_i} p_d(v'_i|v_i)^2 dv'_i - 2\prod_{i=0}^{l-1} \int_{v'_i} p(v'_i|v_i)p_d(v'_i|v_i) dv'_i \qquad \text{(A.33)}
\end{aligned}
$$

We separately consider each integral in Eq. A.33. The first two integrals can be calculated in general as follows:

$$
A^i = \int_{v'_i} p(v'_i|v_i)^2 dv'_i = \sum_{v'_i} p(v'_i|v_i)^2 = \frac{1+\alpha^2 t_i^2}{(1+\alpha t_i)^2} \qquad \text{(A.34)}
$$

Therefore, the first and second integrals in Eq. A.33 are $A = \frac{1+\alpha^2 t_i^2}{(1+\alpha t_i)^2}$ and $A_d = \frac{1+\alpha_d^2 t_i^2}{(1+\alpha_d t_i)^2}$, respectively. The third integral in Eq. A.33 is $B^i = \int_{v'_i} p(v'_i|v_i)p_d(v'_i|v_i)dv'_i = \sum_{v'_i} p(v'_i|v_i)p_d(v'_i|v_i) = \frac{1+\alpha\alpha_d t_i^2}{(1+\alpha t_i)(1+\alpha_d t_i)}$. Therefore, the L_2-norm is calculated as:

$$
L_2^2(P,P_d) = \prod_{i=0}^{l-1} A^i + \prod_{i=0}^{l-1} A_d^i - 2\prod_{i=0}^{l-1} B^i \qquad \text{(A.35)}
$$

□

A.11 Proof of Theorem 5

Proof. We have $\xi_a = \mathbb{E}|\mathcal{E}(\mathcal{M}(v_a)) - \mathcal{E}(v_a)| = \sum_{i\in[0,l-1]}(p_i \times 0 + q_i \times \Delta_i) = \sum_{i\in[0,l-1]} q_i \times \Delta_i$. Therefore, Theorem 5 holds. □

A.12 Extended Analysis for Interested Readers

A.13 Dimension-Scalability Analysis of LDP-based Approaches

In this section, we analyze the dimension-scalability of existing LDP-based approaches and ScalableRR. Given a user-predefined privacy budget ϵ_X, we quantify the changes of the randomization probabilities when the number of dimensions increases from r to dr. For each mechanism, we consider two widely used quantitative measurements to calculate the changes, including KL-divergence and L_2-norm. The summary is in Table A.1.

A.13.1 Duchi's Mechanism (DM) [51, 201]

We consider DM for multidimensional numerical data as in [201]. The input is the vector $e \in [-1, 1]^r$ (each feature $a \in [-1, 1]$). The output of DM [52] is $e' \in \{-B, B\}^r$ (each randomized feature $a' \in \{-B, B\}$), where $B = \frac{\exp(\epsilon_X)+1}{\exp(\epsilon_X)-1} C_r$ and C_r is calculated as:

$$C_r = \begin{cases} \dfrac{2^{r-1}}{\binom{(r-1)}{(r-1)/2}}, & \text{if } r \text{ is odd} \\ \dfrac{2^{r-1} + \frac{1}{2}\binom{r}{r/2}}{\binom{(r-1)}{r/2}}, & \text{if } r \text{ is even} \end{cases} \tag{A.36}$$

The DM can be mapped to the general form (Eq. 3.3), as:

$$\forall a \in e : \begin{cases} a' = -B, & \text{with a probability } P(-B|a) \\ a' = B, & \text{with a probability } P(B|a) \end{cases} \tag{A.37}$$

First, if r is odd, let $t = r - 1$, we have:

$$C_r = \frac{2^{r-1}}{\binom{(r-1)}{(r-1)/2}} = \frac{2^t}{\binom{t}{t/2}} = \frac{2^t (\frac{t}{2}!)^2}{t!} \Leftrightarrow \ln C_r = t \ln 2 + 2 \ln(\frac{t}{2}!) - \ln(t!) \tag{A.38}$$

Using Stirling's approximation [54], Eq. A.38 becomes:

$$\ln C_r \approx t \ln 2 + 2\frac{t}{2}\ln(\frac{t}{2}) - 2\frac{t}{2} - t\ln t + t + \Theta(\ln t) = \Theta(\ln t)$$
$$\rightarrow \frac{\partial}{\partial r}\ln C_r = \Theta(\frac{1}{t}) > 0, \forall r \geq 1 \tag{A.39}$$

Second, if r is even, we have:

$$C_r = \frac{2^{r-1} + \frac{1}{2}\binom{r}{r/2}}{\binom{(r-1)}{r/2}} = \frac{2^r + \frac{r!}{(\frac{r}{2}!)^2}}{2\left(\frac{(r-1)!}{(\frac{r}{2}!)(\frac{r}{2}-1)!}\right)} = 2^r\frac{(r/2!)^2}{r!} + 1$$

Let $I = \frac{2^r(\frac{r}{2}!)^2}{r!}$ and use the Stirling's approximation, we have:

$$\ln I = r\ln 2 + 2\ln\left(\frac{r}{2}!\right) - \ln(r!) \approx \Theta(\ln(r)) \rightarrow \frac{\partial}{\partial r}\ln I = \Theta(\frac{1}{r}) > 0, \forall r \geq 1 \tag{A.40}$$

From Eqs. A.39 and A.40, C_r increases when r increases.

When the number of features r increases to $d \times r$, the new privacy budget will be $\frac{\epsilon_X}{d}$ to maintain the total privacy budget as the user-predefined privacy budget ϵ_X. Consider $J = \frac{\exp(\frac{\epsilon_X}{d})+1}{\exp(\frac{\epsilon_X}{d})-1}$, we have $\frac{\partial}{\partial d}J = \frac{2\epsilon_X}{(\exp(\frac{\epsilon_X}{d})-1)^2 d^2} > 0$, hence, B increases as r increases.

Let $k = \frac{e^{\epsilon_X}}{e^{\epsilon_X}+1}$. Following DM in [201], given a feature $a \in [-1, 1]$, DM samples v where $v = 1$ with probability $\frac{1+a}{2}$ and $v = -1$ with probability $\frac{1-a}{2}$. Then, DM samples $u \sim Bernoulli(k)$, if $u = 1$ then DM will return $a' = B$ or $a' = -B$ such that $a' \times v > 0$. Therefore, the probabilities in Eq. A.37 are calculated as follows:

$$P(B|a) = P(v=-1)P(u=0) + P(v=1)P(u=1) = \frac{1-a}{2} + ka$$
$$P(-B|a) = P(v=-1)P(u=1) + P(v=1)P(u=0) = \frac{1+a}{2} - ka \tag{A.41}$$

Theorem 13. *When the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d, where $d \times r \in \mathbb{N}^+$ and $d > 1$, given a user-predefined privacy budget ϵ_X, the change γ in the randomization probabilities quantified by KL-divergence and L_2-norm of Duchi's Mechanism are as follows:*

(1) *$KL(P||P_d) = +\infty$ (N.A.)*

(2) *$L_2(P, P_d) = \sqrt{2k_2^2a^2 - 2k_2a^2 + 2k_1^2a^2 - 2k_1a^2 + a^2 + 1}$*

Proof. **KL-divergence.** Let $P(a'|a)$ and $P_d(a'|a)$ be the randomization probabilities when the number of features are r and $d\times r$, respectively. Let B_1 and B_2 be the values of parameter B when the number of feature is r and $d \times r$. We have shown that B is an increasing function, so $B_2 > B_1$, since $d \times r > r$ (d¿1). Considering the discrete distributions in DM, KL-divergence between $P(a'|a)$ and $P_d(a'|a)$ is calculated as:

$$KL(P||P_d) = \sum_{a'} P(a'|a) \log\left(\frac{P(a'|a)}{P_d(a'|a)}\right) \tag{A.42}$$

When $a' = -B_1$ or $a' = B_1$, $P(a'|a) \neq 0$ and $P_d(a'|a) = 0$, therefore, $KL(P||P_d)$ is infinity (N.A.).

L_2-norm. Considering the L_2-norm between $P(a'|a)$ and $P_d(a'|a)$ we have:

$$\begin{aligned} L_2(P, P_d) &= \sqrt{\sum_{a'} (P(a'|a) - P_d(a'|a))^2} \\ &= \sqrt{P_d(-B_2|a)^2 + P(-B_1|a)^2 + P(B_1|a)^2 + P_d(B_2|a)^2} \\ &= \sqrt{2k_2^2a^2 - 2k_2a^2 + 2k_1^2a^2 - 2k_1a^2 + a^2 + 1} \end{aligned} \tag{A.43}$$

where $k_1 = \frac{e^{\epsilon_X}}{e^{\epsilon_X}+1}$ and $k_2 = \frac{e^{\epsilon_X/d}}{e^{\epsilon_X/d}+1}$.

□

A.13.2 Piecewise Mechanism (PM) [201]

Given a user-predefined privacy budget ϵ_X, we consider a feature $a \in [-1, 1]$ of a feature vectors $e \in [-1, 1]^r$. The output of Piecewise mechanism (PM) [201] is $a' \in [-C, C]$, where $C = \frac{e^{\epsilon_X/2}+1}{e^{\epsilon_X/2}-1}$. In addition, let $L = l(a) = \frac{C+1}{2}a - \frac{C-1}{2}$ and $R = r(a) = L + C - 1$.

The PM can be mapped to the general form (Eq. 3.3), as:

$$\forall a \in e : \begin{cases} a' \sim U([L,R]), \text{ with probability } \dfrac{1}{e^{\epsilon_X/2}+1} \\ a' \sim U([-C,L) \cup (R,C]), \text{with probability} \frac{e^{\frac{\epsilon_X}{2}}}{e^{\frac{\epsilon_X}{2}}+1} \end{cases}$$

where $U(\cdot)$ is a uniform distribution.

Similar to DM, when r increases to $d \times r$, the new privacy budget is $\frac{\epsilon_X}{d}$ to preserve the same ϵ_X as before. Consider $I = \frac{\exp(\frac{\epsilon_X}{2d})+1}{\exp(\frac{\epsilon_X}{2d})-1}$, we have $\frac{\partial}{\partial d} I = \frac{\epsilon}{(\exp(\frac{\epsilon_X}{2d})-1)^2 d^2} > 0$, hence, C increases as r increases.

Let $k = \frac{\exp(\epsilon_X/2)}{\exp(\epsilon_X/2)+1}$, PM works as follows: first, we sample $x \sim U([0,1])$; if $x < k$ then return $a' \sim U([-C,L) \cup (R,C])$ and if $x \geq k$ then return $a' \sim U([L,R])$. Therefore, we have:

$$\begin{aligned} P(a'|a) &= \int_x P(a'|x)P(x|a)dx = \int_x P(a'|x)P(x)dx \\ &= \int_0^k P(a'|x)P(x)dx + \int_k^1 P(a'|x)P(x)dx \\ &= \frac{1}{R-L}k + \frac{1}{2C+L-R}(1-k) = \frac{k}{C-1} + \frac{1-k}{C+1} = \frac{2k+C-1}{C^2-1} \end{aligned}$$

We denote $A_1 = \frac{2k_1+C_1-1}{C_1^2-1}$ and $A_2 = \frac{2k_2+C_2-1}{C_2^2-1}$, where $k_1 = \frac{\exp(\frac{\epsilon_X}{2})}{\exp(\frac{\epsilon_X}{2})+1}$, $C_1 = \frac{\exp(\frac{\epsilon_X}{2})+1}{\exp(\frac{\epsilon_X}{2})-1}$, $k_2 = \frac{\exp(\frac{\epsilon_X}{2d})}{\exp(\frac{\epsilon_X}{2d})+1}$, and $C_2 = \frac{\exp(\frac{\epsilon_X}{2d})+1}{\exp(\frac{\epsilon_X}{2d})-1}$.

Theorem 14. *When the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d, where $d \times r \in \mathbb{N}^+$ and $d > 1$, given a user-predefined privacy budget ϵ_X, the change γ in the randomization probabilities quantified by KL-divergence and L_2-norm of Piecewise Mechanism are as follows:*

(1) $KL(P||P_d) = 2A_1C_1 \log(A_1/A_2)$

(2) $L_2(P, P_d) = \sqrt{2A_2^2C_2 + 2A_1^2C_1 - 4C_1A_1A_2}$

Proof. **KL-divergence.** Let $P(a'|a)$ and $P_d(a'|a)$ be the randomization probabilities when the number of features are r and $d \times r$, respectively. In addition, let C_1 and C_2 be the values of parameter C when the number of feature is r and $d \times r$. Since C increases as r increases, $C_2 > C_1$. Considering the continuous distribution in PM, the KL-divergence between $P(a'|a)$ and $P_d(a'|a)$ is as follow:

$$\begin{aligned} KL(P||P_d) &= \int_{a'} P(a'|a) \log\Big(\frac{P(a'|a)}{P_d(a'|a)}\Big) da' = \int_{-C_1}^{C_1} P(a'|a) \log\Big(\frac{P(a'|a)}{P_d(a'|a)}\Big) da' \\ &= \int_{-C_1}^{C_1} A_1 \log\Big(\frac{A_1}{A_2}\Big) da' = 2C_1A_1 \log(A_1/A_2) \end{aligned} \tag{A.44}$$

L_2**-norm.** Considering the L_2-norm between $P(a'|a)$ and $P_d(a'|a)$ we have:

$$\begin{aligned} &L_2(P, P_d) = \sqrt{\int_{a'} (P(a'|a) - P_d(a'|a))^2 da'} \\ &= \Big[\int_{-C_2}^{-C_1} (P(a'|a) - P_d(a'|a))^2 da' + \int_{-C_1}^{C_1} (P(a'|a) - P_d(a'|a))^2 da' \\ &\quad + \int_{C_1}^{C_2} (P(a'|a) - P_d(a'|a))^2 da'\Big]^{\frac{1}{2}} \\ &= \sqrt{2A_2^2(C_2 - C_1) + 2C_1(A_2 - A_1)^2} = \sqrt{2A_2^2C_2 + 2A_1^2C_1 - 4C_1A_1A_2} \end{aligned} \tag{A.45}$$

□

A.13.3 Hybrid Mechanism (HM) [201]

As in [201], HM can be considered as a combination of DM and PM. We simply report the best case (smallest distance) of HM following DM and PM. Therefore, KL-divergence of HM is $2A_1C_1 \log\Big(A_1/A_2\Big)$ (following PM, since it is N.A. in DM) and L_2-norm of HM is the minimum L_2-norm values of DM and HM.

A.13.4 Three Outputs Mechanism [226]

Given a user-predefined privacy budget ϵ_X, the input is a feature $a \in [-1, 1]$ and the output is $a' \in \{-C, 0, C\}$, where $C = \frac{\exp(\epsilon_X)+1}{(\exp(\epsilon_X)-1)(1-P_{0\leftarrow 1})}$.

This mechanism can be mapped to the general form (Eq. 3.3), as:

$$\forall a \in e : \begin{cases} a' = -C, & \text{with a probability } P_{-C \leftarrow a} \\ a' = 0, & \text{with a probability } P_{0 \leftarrow a} \\ a' = C, & \text{with a probability } P_{C \leftarrow a} \end{cases}$$

where $P_{-C\leftarrow a}$, $P_{0\leftarrow a}$, and $P_{C\leftarrow a}$ are given by Eqs. $2-8$ in [226].

Theorem 15. *When the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d, where $d \times r \in \mathbb{N}^+$ and $d > 1$, given a user-predefined privacy budget ϵ_X, the change γ in the randomization probabilities quantified by KL-divergence and L_2-norm of Three outputs Mechanism are as follows:*

(1) *$KL(P||P_d) = +\infty$ (N.A.)*

(2) *$L_2(P, P_d) = \Big(P_d(-C_2|a)^2 + P(-C_1|a)^2 + [P(0|a) - P_d(0|a)]^2 + P(C_1|a)^2 + P_d(C_2|a)^2\Big)^{1/2}$*

Proof. **KL divergence.** Let $P(a'|a)$ and $P_d(a'|a)$ be the randomization probabilities when the number of features are r and $d\times r$, respectively. Let C_1 and C_2 be the values of parameter C when the number of feature is r and $d \times r$. Similar to the analysis of DM, $C_2 > C_1$. Considering the discrete distributions in DM, KL-divergence between $P(a'|a)$ and $P_d(a'|a)$ is calculated as:

$$KL(P||P_d) = \sum_{a'} P(a'|a) \log\Big(\frac{P(a'|a)}{P_d(a'|a)}\Big) \tag{A.46}$$

When $a' = -C_1$ or $a' = C_1$, $P(a'|a) \neq 0$ and $P_d(a'|a) = 0$, therefore, $KL(P||P_d)$ is infinity (N.A.).

L_2**-norm.** We have:

$$L_2(P, P_d) = \sqrt{\sum_{a'} (P(a'|a) - Q(a'|a))^2} \tag{A.47}$$

$$= \Big(P_d(-C_2|a)^2 + P(-C_1|a)^2 + \big(P(0|a) - P_d(0|a)\big)^2 + P(C_1|a)^2 + P_d(C_2|a)^2\Big)^{1/2}$$

where $P(a'|a)$ and $P_d(a'|a)$ following Eqs. $2-8$ as in [226].

$\square$

A.13.5 OME [121]

Given a user-predefined privacy budget ϵ_X, we consider a feature a of a feature vectors $e \in \mathbb{R}^r$. Different from aforementioned numerical-based approaches, OME works at the bit level in which each feature a is represented by a binary vector $v = \{v_i\}_{i=0}^{l} \in \{0,1\}^l$ (l is the number of bits used to present a, then the output of OME is the noisy vector $v' = \{v'_i\}_{i=0}^{l} \in \{0,1\}^l$. Let a' be the decoded value of v'.

The OME mechanism can be mapped to the general form for a feature (Eq. 3.3), as:

$$\forall a \in e : P(a'|a) = P(v'|v) = \prod_{i=0}^{l-1} P(v'_i|v_i) \tag{A.48}$$

where $P(v'_i|v_i)$ is given by Eq. A.52.

Let $P(a'|a)$ and $P_d(a'|a)$ be the randomization probabilities when the number of features are r and $d \times r$, respectively. In OME, there are different randomization probabilities, depending the value (i.e., 0 or 1) and position (i.e., even or odd) of the bit. In OME (and also in this paper), we consider l is even; therefore, $P_{even} = P_{odd} = \frac{1}{2}$, where P_{even} and P_{odd} are the probabilities that a bit is at a even and odd position, respectively. Therefore, we have:

$$\begin{aligned}
P_{0e} &= P_{i=2j}(v'_i = 0|v_i) = P_{even}\Big(\frac{\alpha \exp(\frac{\epsilon_X}{rl})}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{1}{1+\alpha}\Big) = \frac{k_1+k_2}{2} \\
P_{1e} &= P_{i=2j}(v'_i = 1|v_i) = P_{even}\Big(\frac{1}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{\alpha}{1+\alpha}\Big) \\
&= \frac{1}{2}(1-k_1+1-k_2) = 1 - \frac{k_1+k_2}{2} \\
P_{0o} &= P_{i=2j+1}(v'_i = 0|v_i) = P_{odd}\Big(\frac{\alpha \exp(\frac{\epsilon_X}{rl})}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{\alpha^3}{1+\alpha^3}\Big) = \frac{k_1+k_3}{2} \\
P_{1o} &= P_{i=2j+1}(v'_i = 1|v_i) P_{odd}\Big(\frac{1}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{1}{1+\alpha^3}\Big) = 1 - \frac{k_1+k_3}{2}
\end{aligned} \tag{A.49}$$

where $k_1 = \frac{\alpha \exp(\frac{\epsilon_X}{rl})}{1+\alpha\exp(\frac{\epsilon_X}{rl})}$, $k_2 = \frac{1}{1+\alpha}$, and $k_3 = \frac{\alpha^3}{1+\alpha^3}$. We also denote $k_{1d} = \frac{\alpha \exp(\frac{\epsilon_X}{rld})}{1+\alpha\exp(\frac{\epsilon_X}{rld})}$, $k_{2d} = k_2$, and $k_{3d} = k_3$.

Theorem 16. *When the dimension of the feature vector $e \in \mathbb{R}^r$ increases by a factor d, where $d \times r \in \mathbb{N}^+$ and $d > 1$, and given a user-predefined privacy budget ϵ_X, the change γ in the randomization probabilities quantified by KL-divergence and L_2-norm of OME Mechanism are as follows:*

(1) $KL(P||P_d) = \sum_{i=2j}^{l-1}\left(P_{0e}\log\frac{P_{0e}}{P_{d0e}} + P_{1e}\log\frac{P_{1e}}{P_{d1e}}\right) + \sum_{i=2j+1}^{l-1}\left(P_{0o}\log\frac{P_{0o}}{P_{d0o}} + P_{1o}\log\frac{P_{1o}}{P_{d1o}}\right)$

(2) $L_2(P,P_d) = A_{1,2}^{l/2}A_{1,3}^{l/2} - 2B_{1,2,d}^{l/2}B_{1,3,d}^{l/2} + A_{1d,2}^{l/2}A_{1d,3}^{l/2}$

Proof. **KL-divergence.**

$$
\begin{aligned}
KL(P||P_d) &= \sum_{v'} P(v'|v_i)\log\frac{P(v'|v_i)}{P_d(v'|v_i)} \qquad \text{(A.50)}\\
&= \sum_{i=0}^{l-1}\left(P(v_i'=0|v_i)\log\frac{P(v_i'=0|v_i)}{P_d(v_i'=0|v_i)} + P(v_i'=1|v_i)\log\frac{P(v_i'=1|v_i)}{P_d(v_i'=1|v_i)}\right)\\
&= \sum_{i=2j}^{l-1}\left(P_{0e}\log\frac{P_{0e}}{P_{d0e}} + P_{1e}\log\frac{P_{1e}}{P_{d1e}}\right) + \sum_{i=2j+1}^{l-1}\left(P_{0o}\log\frac{P_{0o}}{P_{d0o}} + P_{1o}\log\frac{P_{1o}}{P_{d1o}}\right)
\end{aligned}
$$

L_2**-norm.** We have: $P(v'|v) = \Pi_{i=0}^{l-1}P(v_i'|v_i)$. Therefore,

$$
\begin{aligned}
L_2^2(P,P_d) &= \int_{a'} (P(a'|a) - P_d(a'|a))^2 da' \\
&= \int_{v'_0} \cdots \int_{v'_{l-1}} \Big[\Pi_{i=0}^{l-1} P(v'_i|v_i) - \Pi_{i=0}^{l-1} P_d(v'_i|v_i)\Big]^2 dv'_0 \cdots dv'_{l-1} \\
&= \Pi_{i=2j}\Big[\frac{(k_1+k_2)^2}{4} + \frac{(1-\frac{k_1+k_2}{2})^2}{4}\Big]\Pi_{i=2j+1}\Big[\frac{(k_1+k_3)^2}{4} + \frac{(1-\frac{k_1+k_3}{2})^2}{4}\Big] \\
&- 2\Pi_{i=2j}\Big[\frac{1}{4}(k_1+k_2)(k_{1d}+k_{2d}) + \frac{1}{4}(1-\frac{k_1+k_2}{2})(1-\frac{k_{1d}+k_{2d}}{2})\Big] \\
&\times \Pi_{i=2j+1}\Big[\frac{1}{4}(k_1+k_3)(k_{1d}+k_{3d}) + \frac{1}{4}(1-\frac{k_1+k_3}{2})(1-\frac{k_{1d}+k_{3d}}{2})\Big] \\
&+ \Pi_{i=2j}\Big[\frac{1}{4}(k_{1d}+k_{2d})^2 + \frac{1}{4}(1-\frac{k_{1d}+k_{2d}}{2})^2\Big] \times \Pi_{i=2j+1}\Big[\frac{1}{4}(k_{1d}+k_{3d})^2 + \frac{1}{4}(1-\frac{k_{1d}+k_{3d}}{2})^2\Big] \\
&= \Big[\tfrac{(k_1+k_2)^2}{4} + \tfrac{(1-\frac{k_1+k_2}{2})^2}{4}\Big]^{l/2}\Big[\tfrac{(k_1+k_3)^2}{4} + \tfrac{(1-\frac{k_1+k_3}{2})^2}{4}\Big]^{l/2} - 2\Big[\tfrac{1}{4}(k_1+k_2)(k_{1d}+k_2) + \tfrac{1}{4}(1-\tfrac{k_1+k_2}{2})(1-\tfrac{k_{1d}+k_2}{2})\Big]^{l/2} \\
&\times \Big[\frac{1}{4}(k_1+k_3)(k_{1d}+k_3) + \frac{1}{4}(1-\frac{k_1+k_3}{2})(1-\frac{k_{1d}+k_3}{2})\Big]^{l/2} \\
&+ \Big[\frac{(k_{1d}+k_2)^2}{4} + \frac{(1-\frac{k_{1d}+k_2}{2})^2}{4}\Big]^{l/2}\Big[\frac{(k_{1d}+k_3)^2}{4} + \frac{(1-\frac{k_{1d}+k_3}{2})^2}{4}\Big]^{l/2} \\
&= A_{1,2}^{l/2}A_{1,3}^{l/2} - 2B_{1,2,d}^{l/2}B_{1,3,d}^{l/2} + A_{1d,2}^{l/2}A_{1d,3}^{l/2} \qquad \text{(A.51)}
\end{aligned}
$$

where $A_{i,j} = \frac{1}{4}(k_i+k_j)^2 + \frac{1}{4}(1-\frac{k_i+k_j}{2})^2$ and

$$B_{i,j,d} = \frac{1}{4}(k_i+k_j)(k_{id}+k_j) + \frac{1}{4}(1-\frac{k_i+k_j}{2})(1-\frac{k_{id}+k_j}{2})$$

$\square$

Table A.1 Summarizing $(\gamma, \gamma_\epsilon)$-Dimension-Scalability of LDP-preserving RR Mechanisms

Technique	γ (given $\gamma_\epsilon = 0$)		γ_ϵ (given $\gamma = 0$)
	KL-divergence	*L_2-norm*	
Duchi's mechanism (DM) [51,52]	N.A.	$\sqrt{2k_2^2a^2 - 2k_2a^2 + 2k_1^2a^2 - 2k_1a^2 + a^2 + 1}$	$(d-1)\times\epsilon$
Piecewise mechanism (PM) [201]	$2A_1C_1 \log\big(A_1/A_2\big)$	$\sqrt{2A_2^2C_2 + 2A_1^2C_1 - 4C_1A_1A_2}$	
Hybrid mechanism (HM) [201]	$2A_1C_1 \log\big(A_1/A_2\big)$	$\min\{DM, PM\}$	
Three-outputs [226]	N.A.	Theorem 15	
Adaptive OME [121]	Theorem 16	$\sqrt{A_{1,2}^{l/2}A_{1,3}^{l/2} - 2B_{1,2,d}^{l/2}B_{1,3,d}^{l/2} + A_{1d,2}^{l/2}A_{1d,3}^{l/2}}$	
ScalableRR (Ours)	$\sum_{i=0}^{l-1}\log\Big(\frac{1+\alpha_d t_i}{1+\alpha t_i}\Big) + \sum_{i=0}^{l-1} q_i \log(\frac{\alpha}{\alpha_d})$	$\sqrt{\prod_{i=0}^{l-1} A^i + \prod_{i=0}^{l-1} A_d^i - 2\prod_{i=0}^{l-1} B^i}$	

A.14 Adaptive Privacy Budget in OME

In this section, we aim at providing adaptive privacy budget for OME [121]. OME first encodes embedding features e into an rl-bit binary vector v. Then, each bit $i \in [0, rl-1]$ is randomized by the following f_{OME} mechanism:

$$\forall i \in [0, rl-1] : P(v'_i = 1) = \begin{cases} p_{1i} = \dfrac{\alpha}{1+\alpha}, & \text{if } i \in 2j, v_i = 1 \\ p_{2i} = \dfrac{1}{1+\alpha^3}, & \text{if } i \in 2j+1, v_i = 1 \\ q_i = \dfrac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}, & \text{if } \ v_i = 0 \end{cases} \tag{A.52}$$

From Eq. A.52, we also have that $P(v'_i = 0) = 1 - p_{1i} = \frac{1}{1+\alpha}$ if $v_i = 1$ and $i \in 2j$, $P(v'_i = 0) = 1 - p_{2i} = \frac{\alpha^3}{1+\alpha^3}$ if $v_i = 1$ and $i \in 2j+1$, and $P(v'_i = 0) = 1 - q_i = \frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})}$ if $v_i = 0$.

Theorem 17. *OME with the randomization probabilities as in Eq. A.52 preserves $\epsilon_{adaptive}$-LDP, where $\epsilon_{adaptive} = (\frac{rl}{Q_1} - \frac{rl}{Q_2})\ln(\alpha) + \frac{\epsilon}{2Q_1} + \frac{\epsilon}{2Q_2}$ in which $Q_1 = \frac{\alpha}{1+\alpha}\frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}\frac{1}{1+\alpha}$ and $Q_2 = \frac{1}{1+\alpha^3}\frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}\frac{\alpha^3}{1+\alpha^3}$.*

Proof. Similar to the analysis in **Section A.9**, we obtain:

$$\frac{P(f_{\text{OME}}(v) = v_z)}{P(f_{\text{OME}}(\widetilde{v}) = v_z)} \leq \prod_{i=0}^{rl-1}\Big(\frac{P(f_{\text{OME}}(v_i) = v_{z,i})}{P(f_{\text{OME}}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(f_{\text{OME}}(v,i)) - \mathcal{E}(f_{\text{OME}}(v^{\neq i},i))|}} \leq \exp(\epsilon) \tag{A.53}$$

and the expectation $\mathbb{E}|\mathcal{E}(f_{\text{OME}}(v,i)) - \mathcal{E}(f_{\text{OME}}(v^{\neq i},i))|$ is computed as follows:

$$\mathbb{E}|\mathcal{E}(f_{\text{OME}}(v,i)) - \mathcal{E}(f_{\text{OME}}(v^{\neq i},i))| \tag{A.54}$$
$$= \begin{cases} \big(p_{1i}(1-q_i) + q_i(1-p_{1i})\big)\Delta_i = Q_1\Delta_i, \text{if } i \in 2j \\ \big(p_{2i}(1-q_i) + q_i(1-p_{2i})\big)\Delta_i = Q_2\Delta_i, \text{if } i \in 2j+1 \end{cases}$$

where $Q_1 = p_{1i}(1-q_i) + q_i(1-p_{1i}) = \frac{\alpha}{1+\alpha}\frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}\frac{1}{1+\alpha}$, and $Q_2 = p_{2i}(1-q_i) + q_i(1-p_{2i}) = \frac{1}{1+\alpha^3}\frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}\frac{\alpha^3}{1+\alpha^3}$.

From Eqs. A.53 and A.54, we have:

$$\frac{P(f_{\text{OME}}(v) = v_z)}{P(f_{\text{OME}}(\widetilde{v}) = v_z)} \leq \prod_{i=0}^{rl-1}\Big(\frac{P(f_{\text{OME}}(v_i) = v_{z,i})}{P(f_{\text{OME}}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(f_{\text{OME}}(v,i)) - \mathcal{E}(f_{\text{OME}}(v^{\neq i},i))|}}$$
$$= \prod_{i\in 2j}\left(\frac{P(f_{\text{OME}}(v_i)=1|v_i=1)P(f_{\text{OME}}(v_i)=0|v_i=0)}{P(f_{\text{OME}}(v_i)=1|v_i=0)P(f_{\text{OME}}(v_i)=0|v_i=1)}\right)^{\frac{\Delta_i}{Q_1\Delta_i}} \times \prod_{i\in 2j+1}\left(\frac{P(f_{\text{OME}}(v_i)=1|v_i=1)P(f_{\text{OME}}(v_i)=0|v_i=0)}{P(f_{\text{OME}}(v_i)=1|v_i=0)P(f_{\text{OME}}(v_i)=0|v_i=1)}\right)^{\frac{\Delta_i}{Q_2\Delta_i}}$$
$$= \alpha^{\frac{rl}{Q_1} - \frac{rl}{Q_2}}\exp(\frac{\epsilon}{2Q_1} + \frac{\epsilon}{2Q_2}) \tag{A.55}$$

Then, from Eq. A.55, we have:

$$\epsilon_{adaptive} = \ln\Big(\alpha^{\frac{rl}{Q_1} - \frac{rl}{Q_2}}\exp(\frac{\epsilon}{2Q_1} + \frac{\epsilon}{2Q_2})\Big) = (\frac{rl}{Q_1} - \frac{rl}{Q_2})\ln(\alpha) + \frac{\epsilon}{2Q_1} + \frac{\epsilon}{2Q_2} \tag{A.56}$$

Consequently, Theorem 17 does hold. □

Following the experiment settings in OME [121], with the commonly used $\alpha = 100$, when changing $r \in \{10, 100, 1,000, 10,000\}$ with a fixed $l = 10$, or when changing $l \in \{5, 10, 20, 100, 1,000\}$ with a fixed $r = 1,000$ and under a tight privacy budget $\epsilon = 0.1$, the proportion $\epsilon_{adaptive}/\epsilon$ significantly changes among $[4.6e+6, 4.6e+9]$. In other words, the $\epsilon_{adaptive}$ is extremely larger than ϵ, for most r and l values in practice.

In OME [121], when the randomized responses follow Eq. A.52, the privacy budget is ϵ. However, when OME is adapted to our setting and privacy analysis, the privacy budget is $\epsilon_{adaptive} = (\frac{rl}{Q_1} - \frac{rl}{Q_2})\ln(\alpha) + \frac{\epsilon}{2Q_1} + \frac{\epsilon}{2Q_2}$, instead of ϵ. Given a privacy budget ϵ_X for the embedding features, in other to adapt OME with our setting following our privacy budget analysis, we need to find an α in Eq. A.56 so that we obtain the same privacy budget with the conventional analysis of OME. In other words, we need to find α satisfying the condition:

$$\epsilon_{adaptive} = (\frac{rl}{Q_1} - \frac{rl}{Q_2})\ln(\alpha) + \frac{\epsilon_X}{2Q_1} + \frac{\epsilon_X}{2Q_2} = \epsilon_X \tag{A.57}$$

Denote $t = \exp(\frac{\epsilon_X}{rl})$, we compute $Q_2 - Q_1$, $Q_1 + Q_2$, and $Q_1 Q_2$ as follows:

$$
\begin{aligned}
Q_1 &= \frac{\alpha}{1+\alpha}\frac{\alpha\exp(\frac{\epsilon_X}{rl})}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon_X}{rl})}\frac{1}{1+\alpha} = \frac{\alpha^2 t + 1}{(1+\alpha)(1+\alpha t)} \\
Q_2 &= p_{2i}(1-q_i) + q_i(1-p_{2i}) = \frac{1}{1+\alpha^3}\frac{\alpha\exp(\frac{\epsilon_X}{rl})}{1+\alpha\exp(\frac{\epsilon_X}{rl})} + \frac{1}{1+\alpha\exp(\frac{\epsilon_X}{rl})}\frac{\alpha^3}{1+\alpha^3} \\
&= \frac{\alpha^3 + \alpha t}{(1+\alpha^3)(1+\alpha t)} \\
Q_2 - Q_1 &= \frac{\alpha^3 + \alpha t}{(1+\alpha^3)(1+\alpha t)} - \frac{\alpha^2 t + 1}{(1+\alpha)(1+\alpha t)} = \frac{\alpha^3 + \alpha t - (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)}{(1+\alpha t)(1+\alpha^3)} \\
Q_1 + Q_2 &= \frac{\alpha^3 + \alpha t}{(1+\alpha^3)(1+\alpha t)} + \frac{\alpha^2 t + 1}{(1+\alpha)(1+\alpha t)} = \frac{\alpha^3 + \alpha t + (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)}{(1+\alpha t)(1+\alpha^3)} \\
Q_1 Q_2 &= \frac{\alpha^3 + \alpha t}{(1+\alpha^3)(1+\alpha t)}\frac{\alpha^2 t + 1}{(1+\alpha)(1+\alpha t)} = \frac{(\alpha^3 + \alpha t)(\alpha^2 t + 1)}{(1+\alpha t)^2(1+\alpha^3)(1+\alpha)}
\end{aligned}
\tag{A.58}
$$

Now, substituting Eq. A.58 into Eq. A.57, we have:

$$
\begin{aligned}
& rl\ln\alpha\frac{Q_2 - Q_1}{Q_1 Q_2} + \frac{\epsilon_X}{2}\frac{Q_1 + Q_2}{Q_1 Q_2} = \epsilon_X \\
& \Leftrightarrow rl\ln\alpha\frac{\Big[\alpha^3 + \alpha t - (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big]}{(1+\alpha t)(1+\alpha^3)}\frac{(1+\alpha t)^2(1+\alpha^3)(1+\alpha)}{(\alpha^3 + \alpha t)(\alpha^2 t + 1)} \\
& + \frac{\epsilon_X}{2}\frac{\Big[\alpha^3 + \alpha t + (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big]}{(1+\alpha t)(1+\alpha^3)}\frac{(1+\alpha t)^2(1+\alpha^3)(1+\alpha)}{(\alpha^3 + \alpha t)(\alpha^2 t + 1)} = \epsilon_X \\
& \Leftrightarrow rl\ln\alpha\frac{\Big[\alpha^3 + \alpha t - (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big](1+\alpha t)(1+\alpha)}{(\alpha^3 + \alpha t)(\alpha^2 t + 1)} \\
& + \frac{\epsilon_X}{2}\frac{\Big[\alpha^3 + \alpha t + (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big](1+\alpha t)(1+\alpha)}{(\alpha^3 + \alpha t)(\alpha^2 t + 1)} = \epsilon_X \\
& \Leftrightarrow \frac{(1+\alpha t)(1+\alpha)}{2(\alpha^3 + \alpha t)(\alpha^2 t + 1)}\Big[2rl\ln\alpha\Big(\alpha^3 + \alpha t - (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big) \\
& + \epsilon_X\Big(\alpha^3 + \alpha t + (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big)\Big] = \epsilon_X \\
& \Leftrightarrow (1+\alpha t)(1+\alpha)\Big[2rl\ln\alpha\Big(\alpha^3 + \alpha t - (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big) \\
& + \epsilon_X\Big(\alpha^3 + \alpha t + (\alpha^2 t + 1)(\alpha^2 + 1 - \alpha)\Big)\Big] - 2\epsilon_X(\alpha^3 + \alpha t)(\alpha^2 t + 1) = 0
\end{aligned}
\tag{A.59}
$$

Denoting $f(\alpha)$ as the left-hand side of Eq. A.59, $f(\alpha)$ is a continuous function in $(0, +\infty)$ since every component of $f(\alpha)$ is continuous in $(0, +\infty)$. Therefore, for fixed parameters (ϵ_X, r, l), if we can find $0 < a < b < +\infty$ such that $f(a)f(b) < 0$, there exists solutions for Eq. A.59. Choosing $a \to 0, b \to +\infty$, then we have $f(a) \to +\infty$ and $f(b) \to -\infty$, so, $f(a)f(b) < 0$ results in the existence of solutions for Eq. A.59. As shown in Fig. A.3, there is only one solution for Eq. A.59 since the plot of function $f(\alpha)$ only cross the x-axis once.

Since solving Eq. A.59 is intractable, we apply Newton-Raphson method [22] to find its solution. Given $f(\alpha)$, we find α^* such that $f(\alpha^*) = 0$ in a iterative way, as follows: starting with an initial guess α_0, we iteratively approximate the true solution by the intercept of the tangent line at the current point and the domain space of α until we meet the convergent condition. Formally, the approximated solution at iteration i is calculated as $\alpha_i = \alpha_{i-1} - f(\alpha_{i-1})/f'(\alpha_{i-1})$ and the process reaches convergence at iteration t if $|\alpha_t - \alpha_{t-1}| < \delta$ for a predefined $\delta > 0$. Then, the solution is approximate by $\alpha^* = \alpha_t$ which is different from the true solution by at most δ. Some empirical solutions α^* of Eq. A.59 are listed in Table A.2.

Privacy budget ϵ_X	Solution α^*
0.01	1.01
0.1	1.02
0.5	1.04
1.0	1.05
5.0	1.08
10.0	1.1

Table A.2 Several Empirical Values of α in Equation A.59 as a Function of ϵ_X when $l = 10$, $r = 1,000$, and $\delta = 1e-4$

A.15 Adaptive Privacy Budget in LATENT

In this section, we aim at providing adapted privacy budget bounds for LATENT [13]. LATENT first encodes embedding features e into an rl-bit binary vector v. Then, each bit $i \in [0, rl-1]$ is randomized by a RR mechanism (i.e., the MOUE algorithm for high sensitivities in Theorem 3.3 [13]), denoted f-LT, as follows:

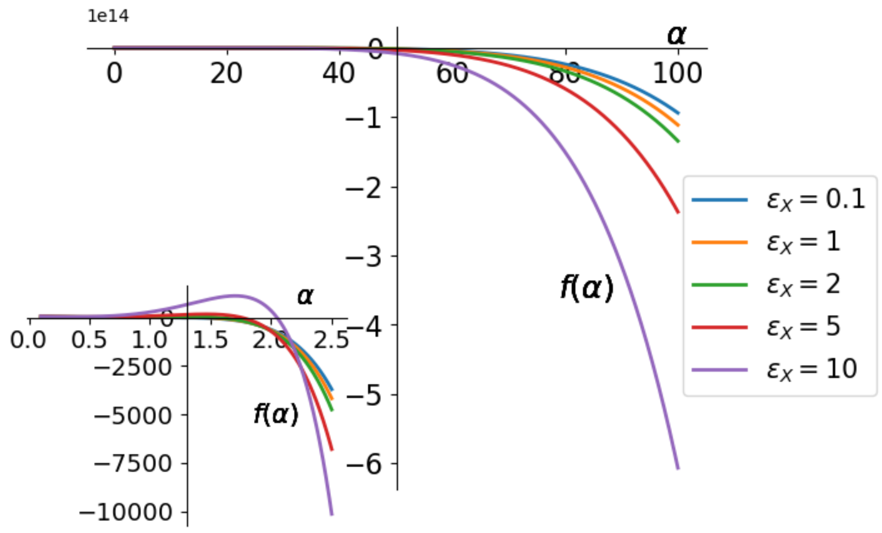


Figure A.3 Illustrating for the root of Eq. A.59.

$$\forall i \in [0, rl-1] : P(v'_i = 1) = \begin{cases} p_i = \dfrac{1}{1+\alpha}, & \text{if } v_i = 1 \\ q_i = \dfrac{1}{1+\alpha\exp(\frac{\epsilon}{rl})}, & \text{if } v_i = 0 \end{cases} \tag{A.60}$$

From Eq. A.60, we also have that $P(v'_i = 0) = 1 - p_i = \frac{\alpha}{1+\alpha}$ if $v_i = 1$, and $P(v'_i = 0) = 1 - q_i = \frac{\alpha\exp(\frac{\epsilon}{rl})}{1+\alpha\exp(\frac{\epsilon}{rl})}$ if $v_i = 0$.

Theorem 18. *LATENT with the randomization probabilities as in Eq. A.60 preserves $\epsilon_{adaptive}$-LDP, where $\epsilon_{adaptive} = \frac{(1+\alpha)(1+\alpha\exp(\frac{\epsilon}{rl}))}{\alpha(1+\exp(\frac{\epsilon}{rl}))}\epsilon$.*

Proof. Similar to the analysis in **Section A.9**, we obtain:

$$\frac{P(f_{LT}(v) = v_z)}{P(f_{LT}(\widetilde{v}) = v_z)} \leq \prod_{i=0}^{rl-1} \Big(\frac{P(f_{LT}(v_i) = v_{z,i})}{P(f_{LT}(v_i^{\neq i}) = v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(f_{LT}(v,i)) - \mathcal{E}(f_{LT}(v^{\neq i},i))|}} \leq \exp(\epsilon) \tag{A.61}$$

and $\mathbb{E}|\mathcal{E}(f_{LT}(v,i)) - \mathcal{E}(f_{LT}(v^{\neq i},i))|$ is computed as follows:

$$\mathbb{E}|\mathcal{E}(f_{LT}(v,i)) - \mathcal{E}(f_{LT}(v^{\neq i},i))| = \Big(p_i(1-q_i)P(v_i^{\neq i}=0) + q_i(1-p_i)P(v_i^{\neq i}=1)$$

$$+ (1-p_i)q_iP(v_i^{\neq i}=0) + (1-q_i)p_iP(v_i^{\neq i}=1)\Big)\Delta_i = \Big(p_i(1-q_i) + q_i(1-p_i)\Big)\Delta_i \quad \text{(A.62)}$$

Furthermore, we have:

$$p_i(1-q_i) + q_i(1-p_i) = \frac{\alpha(1+\exp(\frac{\epsilon}{rl}))}{(1+\alpha)(1+\alpha\exp(\frac{\epsilon}{rl}))} \quad \text{(A.63)}$$

From Eqs. A.61-A.63, we have that

$$\frac{P(f_{LT}(v)=v_z)}{P(f_{LT}(\widetilde{v})=v_z)}r \leq \prod_{i=0}^{rl-1}\Big(\frac{P(f_{LT}(v_i)=v_{z,i})}{P(f_{LT}(v_i^{\neq i})=v_{z,i})}\Big)^{\frac{\Delta_i}{\mathbb{E}|\mathcal{E}(f_{LT}(v,i))-\mathcal{E}(f_{LT}(v^{\neq i},i))|}}$$
$$= \prod_{i=0}^{rl-1}\Big(\frac{P(f_{LT}(v_i)=1|v_i=1)}{P(f_{LT}(v_i)=0|v_i=1)}\Big)^{\frac{\Delta_i}{(p_i(1-q_i)+q_i(1-p_i))\Delta_i}}$$
$$\times \prod_{i=0}^{rl-1}\Big(\frac{P(f_{LT}(v_i)=0|v_i=0)}{P(f_{LT}(v_i)=1|v_i=0)}\Big)^{\frac{\Delta_i}{(p_i(1-q_i)+q_i(1-p_i))\Delta_i}}$$
$$= \prod_{i=0}^{rl-1}\Big(\exp(\frac{\epsilon}{rl})\Big)^{\frac{1}{p_i(1-q_i)+q_i(1-p_i)}} \quad \text{(A.64)}$$

Then, from Eq. A.64, we have:

$$\epsilon_{adaptive} = \frac{(1+\alpha)(1+\alpha\exp(\frac{\epsilon}{rl}))}{\alpha(1+\exp(\frac{\epsilon}{rl}))}\epsilon \quad \text{(A.65)}$$

Consequently, Theorem 18 holds. □

From Theorem 18, we show the proportion $\epsilon_{adaptive}/\epsilon$ as a function of r and l in Fig. A.4. Following the experiment settings in LATENT [13], with the commonly used $\alpha = 7$, when changing $r \in \{10, 100, 1,000, 10,000\}$ with a fixed $l = 10$, or when changing $l \in \{5, 10, 20, 100, 1,000\}$ with a fixed $r = 1,000$ and under a tight privacy budget $\epsilon_X =$

0.1, the proportion $\epsilon_{adaptive}/\epsilon$ moderately changes among $[4.57, 4.75]$. In other words, the $\epsilon_{adaptive}$ is remarkably larger than ϵ, for most r and l values in practice. Unlike LATENT, our mechanism does not suffer from this problem, i.e., in ScalableRR, $\epsilon_{adaptive}/\epsilon = 1$, thanks to our bit-aware randomization probabilities for LDP in binary encoding (**Theorem 3**).

In LATENT [13], when the randomized responses follow Eq. A.60, the privacy budget is ϵ. However, when LATENT is adapted to our setting and privacy analysis, the privacy budget is $\epsilon_{adaptive} = \frac{(1+\alpha)(1+\alpha\exp(\frac{\epsilon}{rl}))}{\alpha(1+\exp(\frac{\epsilon}{rl}))}\epsilon$, instead of ϵ. Given a privacy budget ϵ_X for the embedding features, for the adaptive LATENT, we need to find an α in Eq. A.65 so that we obtain the same privacy budget with the conventional analysis of LATENT. In other words, we need to find α satisfying Eq. A.65 as follows:

$$\epsilon_{adaptive} = \frac{(1+\alpha)(1+\alpha\exp(\frac{\epsilon_X}{rl}))}{\alpha(1+\exp(\frac{\epsilon_X}{rl}))}\epsilon_X = \epsilon_X \Leftrightarrow \frac{(1+\alpha)(1+\alpha\exp(\frac{\epsilon_X}{rl}))}{\alpha(1+\exp(\frac{\epsilon_X}{rl}))} = 1$$
$$\Leftrightarrow 1 + \alpha^2\exp(\frac{\epsilon_X}{rl}) = 0 \tag{A.66}$$

Since $\alpha^2\exp(\frac{\epsilon_X}{rl}) \geq 0$, then then $1 + \alpha^2\exp(\frac{\epsilon_X}{rl}) \geq 1 > 0$. Hence, we cannot find any α satisfying Eq. A.66, and for all $\alpha \geq 0$, $\epsilon_{adaptive} \neq \epsilon_X$. Specifically, since $\forall\alpha, 1 + \alpha^2\exp(\frac{\epsilon_X}{rl}) > 0$, then $\epsilon_{adaptive} > \epsilon_X$. We call this a privacy risk exaggeration since when using the randomized responses as in Eq. A.60 the actual privacy budget is larger than ϵ_X. This problem can severely loosen the privacy protection. Therefore, we do not consider LATENT in our experiments.

A.16 RMSE Comparison in Mean Estimation

To investigate how our proposed approach ScalableRR works with statistical query, we study our ScalableRR and other baselines with a mean estimation. We created a synthetic data that consists of $N = 1,000$ data samples $\{x_i\}_{i=1}^N$, each of them has $d = 768$ dimensions. The mean estimation is calculated over each dimension as $f_j(D) = \frac{1}{N}\sum_{i=1}^N x_{ij}$ for $j \in [1, d]$. Root mean square error (RMSE) is used to evaluate the error between the original vector and

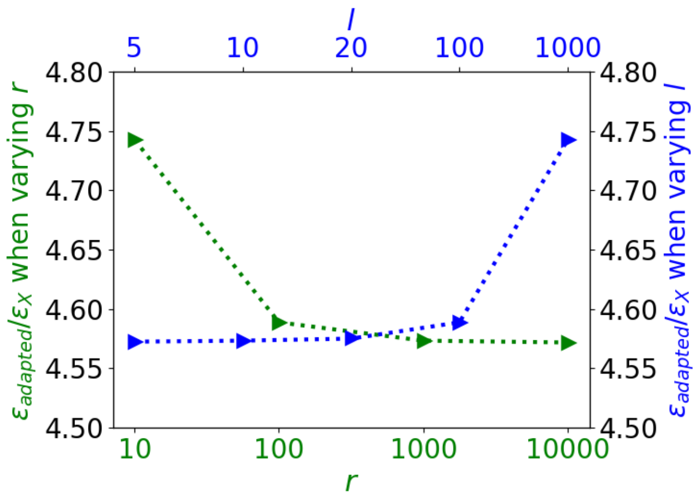


Figure A.4 Impacts of r and l on $\frac{\epsilon_{adaptive}}{\epsilon}$ in LATENT ($\epsilon_X = 0.1$).

the randomized/estimated vector. The binary-encoding-based approaches (i.e., ScalableRR, LATENT, and corrected OME) achieve a significantly small error compared with others. As can be seen in Fig. A.9, ScalableRR obtains the smallest error, which further shows the effectiveness of our proposed mechanism.

A.16.1 Supplementary Theoretical Results for ScalableRR

Computing Expected Error for Gaussian and Laplace mechanisms. The Gaussian and Laplace mechanisms naturally apply an addition operation, which adds noise into the data or embedding features. Hence, in our analysis of expected error bound comparison for an embedding feature (Fig. 3.5), we add noise into the embedding feature following the Gaussian and Laplace mechanisms. The sensitivity captures the magnitude by which an embedding feature can change in the worst case. In our experiment and analysis, we use $l = 10$ bits, i.e., 1 sign bit, 5 exponent bits, and 4 fraction bits. Therefore, the maximum the embedding feature can be change, i.e., the sensitivity, is $2\sum_{i=-4}^{4} 2^i$. Note that, we multiply $\sum_{i=-4}^{4} 2^i$ by 2 since when we flip the sign bit, it significantly changes the value of the embedding feature from $-a$ to a in which $a = \sum_{i=-4}^{4} 2^i$.

More theoretical results for how randomization probabilities change when varying r and l are shown in Figs. A.7 and A.8. A comparison of the randomization probability q_i between OME and ScalableRR for every bit is illustrated in Fig. A.14.

A.17 Expected Error at The Bit-Level

As in Theorem 5, for binary encoding mechanisms, i.e., ScalableRR and OME, the expected error bound is quantified by $\xi_a = \sum_{i \in [0,l-1]} q_i \times \Delta_i$ where Δ_i is as in Lemma 4. Now, let us quantify their expected error and then compare these values, as follows.

Expected Error for ScalableRR.

$$\xi_a = \sum_{i \in [0,l-1]} q_i \times \Delta_i = \sum_{i \in [0,l-1]} \frac{\alpha \exp(\frac{i}{l}\epsilon_X)}{1 + \alpha \exp(\frac{i}{l}\epsilon_X)} \times \Delta_i \tag{A.67}$$

Expected Error for OME. In OME, there are different randomization probabilities for bits 0 and 1 and for even and odd bits (Eq. A.52), the expected error of OME can vary depending on the numbers and the positions of bits 0 and 1 that represents for a feature a. From Eq. A.52, let us denote:

$$q_{10}^{even} = q(v_i' = 0 | v_i = 1, i = 2j) = \frac{1}{1+\alpha}; q_{10}^{odd} = q(v_i' = 0 | v_i = 1, i = 2j+1) = \frac{\alpha^3}{1+\alpha^3}$$
$$q_{01} = q_{01}^{even} = q_{01}^{odd} = q(v_i' = 1 | v_i = 0) = \frac{\alpha \exp(\frac{\epsilon_X}{rl})}{1 + \alpha \exp(\frac{\epsilon_X}{rl})} \tag{A.68}$$

The general formulation for ξ_a in OME is as follows:

$$\begin{aligned} \xi_a &= \sum_{i \in [0,l-1]} q_i \times \Delta_i \\ &= \big(\sum_{i=2j} I_{v_i=1} q_{01}^{even} + I_{v_i=0} q_{10}^{even} + \sum_{i=2j+1} (I_{v_i=1} q_{01}^{odd} + I_{v_i=0} q_{10}^{odd}\big)\Delta_i \end{aligned} \tag{A.69}$$

where $I_{v_i=k}$ ($k = \{0,1\}$) is an indicator function in which $I_{v_i=k} = 1$ if $v_i = k$ and $I_{v_i=k} = 0$, otherwise.

We use l binary bits to represent a feature a, and in OME, there are different randomization probabilities depending on whether the bit is at the even or odd position and whether the bit is 0 or 1. Therefore, we have 2^l possible cases of the binary representation v of the feature a. As shown in Fig. A.10, ScalableRR achieves a much better expected error (i.e., smaller values) compared with all $2^l = 2^{10} = 1,024$ cases of the vector v of OME.

Now, we take a deeper look into a bit-level analysis by relaxing Theorem 5 into **(1)** an expected error $\xi_i = q_i \times \Delta_i$ for each bit $i \in [0, l-1]$, and **(2)** an average top-k expected error $\xi_{top-k} = 1/k \sum_{i=1}^{k} \xi_{i-1}, \forall k \in [1, l]$. Figs. A.11 and A.12 shows that, at the bit-level, ScalableRR achieves smaller values of ξ_i for most important bits, especially the sign bit and exponent bits, compared with adaptive OME, the only baseline working at the bit-level, and comparable expected error ξ_i for least important bits, i.e., fraction bits. Although the noise is less for the most important bits, the expected error is higher due to the high influence of these important bits. The gap between ScalableRR and the baseline is larger given the ξ_{top-k}. We obtain smaller values of ξ_{top-k} for all $k \in [1, l]$, under a rigorous privacy budget $\epsilon_X = 0.1$ and large expansion factor $d = 10^4$. Importantly, when increasing privacy budget ϵ_X (Figs. A.11b,c) and the expansion factor d (Figs. A.12 and A.13), the gap between ScalableRR and the baselines is larger. In fact, the increase in the gap is mainly due to increase in the expected error of the baselines as the expected error in ScalableRR is marginally affected.

A.18 Supplementary Experimental Results for ScalableRR

Datasets and Data Processing. We carried out our experiments on two textual datasets and two image datasets, including the AG dataset [194], our collected Security and Exchange Commission (SEC) financial contract dataset, the large-scale celebFaces attributes (CelebA) dataset [117], and the Federated Extended MNIST (FEMNIST) dataset [26]. The AG dataset is a collection of news articles from more than $2,000$ news sources by [1]. It is categorized into four classes, i.e., world, sport, business, and science/technology. Our SEC dataset

Table A.3 Results of LDP-FL without Privacy Accumulation on the AG Dataset

ϵ_X	1	2	3	4	5	6	7	8	9	10
LDP-FL	78.73	82.21	83.25	84.19	84.36	84.22	84.64	84.43	84.31	84.72
ScalableRR	72.46	79.72	81.42	79.35	81.11	79.84	79.50	80.05	81.20	82.72
Noiseless	87.59									

consists of over $5,000$ contract clauses collected from contracts submitted in SEC filings[4]. The CelebA dataset consists of $202,599$ celebrity images, each with 40 attributes, e.g., attractive face, big lips, big noses, etc., which are used as binary classes. The FEMNIST dataset is built by partitioning the images in Extended MNIST [36] based on the writer of 62 handwritten digits and characters classes. For data preprocessing, we changed all words in the AG and SEC datasets to lower-case and removed punctuation marks. The breakdown of the datasets is in Table 3.1.

Model Configuration. We use the test accuracy and the test area under the curve (AUC) as evaluation metrics. Models with higher values of test accuracy and AUC are better. To extract embedding features, we use the pre-trained BERT-Base (Uncased) model [48] for the AG and SEC datasets, and the ResNet-18 [81] for the CelebA and FEMNIST datasets. Dimension of the extracted embedding features in the AG and SEC datasets is $r = 768$, and in the CelebA and FEMNIST datasets is $r = 512$. For text and image classification tasks, we use two fully connected layers on top of embedding features, each of which consists of $1,500$ hidden neurons and uses a ReLU activation function. The output dimension is corresponding to the number of classes, i.e., 4, 2, 40, and 62 in the AG, SEC, CelebA, and FEMNIST datasets. SGD optimizer with the learning rate is 0.01 in the AG and SEC datasets, and 0.1 in the FEMNIST and CelebA datasets.

Experimental Setting for Anonymization [191]. In LDP-FL [191], the authors design a LDP mechanism to perturb the weights at the local client, then each local client

[4]`https://www.sec.gov/edgar.shtml`

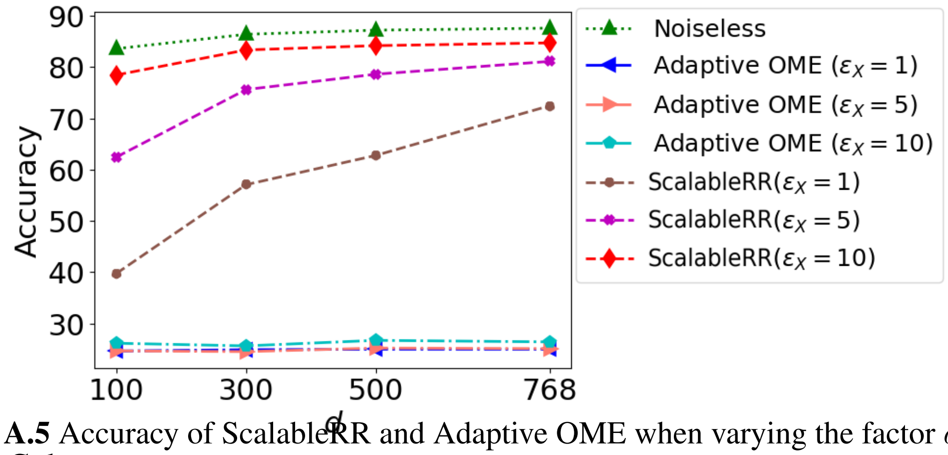


Figure A.5 Accuracy of ScalableRR and Adaptive OME when varying the factor d ($r = 1$) in the AG dataset.

applies a split and shuffle mechanism on the weights of local model and sends each weight through an anonymous mechanism to the cloud. The purpose of the shuffling mechanism is to break the linkage among the model weight updates from the same clients and to mix them among updates from other clients, making it harder for the cloud to combine more than one piece of updates to infer more information about any client. Therefore, the key idea of the shuffle mechanism in LDP-FL is to mitigate the privacy degradation by high data dimension and many training/query iterations. In other words, the client anonymity is preserved, and the privacy budget will not accumulate.

When comparing with LDP-FL, we maintain their mechanism's spirits of no privacy accumulation over the training iterations. It is equivalent to the shuffling step that breaks the linkage among the model weight updates with associated clients. We also used their RR mechanism, which is Eq. 2 [191], to perturb the weight.

In the revision, we added an experiment that do not consider the privacy accumulation over data dimension and training/query iterations. This completely follows the gist of LDP-FL. In addition, the weights we used for LDP-FL without actual shuffling or splitting can be considered a lossless process, therefore the results we reported here can be counted as an upper-bound result for LDP-FL.

As in Table A.3, we obtained the slightly higher accuracy of LDP-FL compared with ScalableRR on the AG dataset. LDP-FL can reduce the privacy accumulation issue by using the shuffling mechanism. However, as pointed out in [62], in the real world, it is possible that the anonymizers (i.e., shuffler) can either be compromised or collude with the server to extract sensitive information. Even though there is a marginally lower trade-off between privacy loss and model utility compared with LDP-FL, the advantage of ScalableRR is that it perturbs the data only once, then used the perturbed data for training process without facing an extra privacy risk potentially caused by the compromised or colluded anonymizer.

Data Reconstruction Attacks. It is shown in [223, 227] that the adversaries can reconstruct embeddings from gradients with high confidences. To evaluate the effectiveness of ScalableRR in the worse case against data reconstruction attacks, we consider that the adversary can losslessly reconstruct the embedding features from the gradients by using gradient-based attacks [223, 227]. Then, the adversary reconstructs original images from the losslessly reconstructed embedding features using two directions: (1) Train a decoder of the embedding features in which we extract the embedding features from a pre-trained model, then apply the model to reconstruct an image [49]; and (2) Optimize an input variable through backpropagation to create an image similar to the original image [143]. We carry out our attacks by strictly following the directions as follows.

First, inspired by [49], for the attack model, we train a decoder of the embedding features encoded by ResNet-50 [81]. To obtain the attack model, we consider an autoencoder in which the encoder part is the pre-trained ResNet-50, which will be freezed from training and only the decoder is trained. The decoder is a combination of two `Conv2DTranspose` layers and a `BatchNormalization` layer with ReLU activation. Adam optimizer is used. To conduct the attack, we train the attacks using the Labeled Faces in the Wild dataset (LFW) [23] and test on the CelebA dataset. That ensure no extra privacy leakage in the attack model.

Second, following the attack in [143], to reconstruct an image, we create a random image and feed it into the model. Then we evaluate the loss between the embedding features of the original image and the reconstructed image at the same layer, and update the input image by gradient descent. The attack is directly applied to the embeddings feature. To test the ability of ScalableRR, we apply ScalableRR with different privacy budgets ϵ_X and compare with the privacy protection free-environment.

A.19 Revisiting Randomized Response Mechanisms for LDP

To preserve LDP given the client's input x, we can apply existing RR mechanisms, such as unary encoding-based approaches [63, 202], hash-based approaches [8, 19, 202], binary encoding-based approaches [13, 121], etc. For instance, hash-based approaches such as those of Google RAPPOR [63] and OLH [202] hash the client's input x onto a bloom filter B of size k using h hash functions. Then, for each client's input x and a bit $i \in B$, RAPPOR creates a perturbed binary value B'_i from B_i with the following randomization probability:

$$B'_i = \begin{cases} 1, & \text{with probability } p/2 \\ 0, & \text{with probability } p/2 \\ B_i, & \text{with probability } 1-p \end{cases} \tag{A.70}$$

where p is a hyper-parameter. This B' is reused as the basis for all future analysis, learning, and reports on this distinct input x. This approach achieves ϵ_X-LDP, where $\epsilon_X = 2h\ln((1-\frac{p}{2})/\frac{p}{2})$, given that the sensitivity of every bit B_i is $\Delta_{B_i} = 1$ [63].

To deal with numerical inputs, e.g., embedding features, generalized RR mechanisms such as Duchi [24, 50], Piece-wise [201], Hybrid [201], Three-outputs [226], Suboptimal [226], LDP-FL [191], LATENT [13], and OME [121] can be applied.

Asymmetric version of RAPPOR (e.g., [202]) designs different randomization probabilities for different inputs. The technique is well-applied in the context of frequency estimation and successfully reduce the communication cost from $O(d)$ to $O(\log n)$ (d is data dimension and n is the number of samples). However, simply applying the mechanism [202] does not optimize the model utility and the privacy-utility trade-off when working with machine learning or deep learning models.

Another line of work in LDP is [136], which addresses the floating-point arithmetic in implementation of DP applications. The inconsistency between mathematical abstraction of Laplace mechanism with sampling "uniform" floating-point numbers can be exploited to carry out privacy attacks. Floating-point arithmetic is a leaky abstraction, which is ubiquitous in computer systems and is difficult to argue about formally and hard to get right in applications, including all the RR mechanisms.

However, different from the asymmetric version of RAPPOR and the floating-point arithmetic, our proposed ScalableRR mechanism focuses on mitigating the privacy-utility trade-off. To achieve that, besides the asymmetric nature of the randomization probabilities, our designed ScalableRR consists of two key components: 1) The bit-aware term $i\%l/l$, which indicates the location of the bit i in each embedding feature associated with the sensitivity of the bit at that location; and 2) The adjustable but bounded α, which takes into account the correlation between privacy loss and the sensitivity of embedding features to mitigate the privacy-utility trade-off and the curse of dimensionality.

The bit-aware property refers to the bits with a more substantial influence on the model utility have smaller randomization probabilities, and vice-versa, under the same privacy protection. By incorporating sensitivities of binary encoding bits into a generalized privacy loss bound, we show that increasing the dimensions of embedding features r, encoding bits l, and model outcomes C marginally affect the randomization probabilities in ScalableRR under the same privacy budget. This dimension-scalable property is crucial to mitigate the curse of dimensionality by retaining a high value of data transmitted correctly through our randomization given large dimensions of r, l, and C.

A.20 Proof of Theorem 6

Proof. Let us denote w_i as the SHAP score of feature i in the aggregated explanation. Given the two explanations w and $\widetilde{w}$ that can be different at any feature and any possible output $z \in Range(\text{XR}\textsc{and})$, where $Range(\text{XR}\textsc{and})$ denotes every possible output of $\text{XR}\textsc{and}$, we have:

$$\begin{aligned} &\frac{P(\text{XR}\textsc{and}(w_i) = z)}{P(\text{XR}\textsc{and}(\widetilde{w}_i) = z)} \leq \frac{\max P(\text{XR}\textsc{and}(w_i) = z)}{\min P(\text{XR}\textsc{and}(\widetilde{w}_i) = z)} \qquad \text{(A.71)} \\ &= \frac{\frac{\exp(\beta)}{\exp(\beta)+\tau-1}}{\min(\frac{\exp(-\Delta_{\mathcal{L}}(i,j))}{\sum_{t\in[k+1,k+\tau]}\exp(-\Delta_{\mathcal{L}}(i,t))} \frac{\tau-1}{\exp(\beta)+\tau-1})} = \frac{\exp(\beta)}{(\tau-1)\min(\frac{\exp(-\Delta_{\mathcal{L}}(i,j))}{\sum_{t\in[k+1,k+\tau]}\exp(-\Delta_{\mathcal{L}}(i,t))})} \leq e^{\varepsilon_i} \end{aligned}$$

Taking a natural logarithm of Eq. A.71, we obtain:

$$\begin{aligned} &\ln(\frac{\exp(\beta)}{(\tau-1)\min(\frac{\exp(-\Delta_{\mathcal{L}_2}(i,j))}{\sum_{t=k+1}^{k+\tau}\exp(-\Delta_{\mathcal{L}_2}(i,t))})}) \leq \ln(\exp(\varepsilon_i)) \\ &\Leftrightarrow \beta \leq \varepsilon_i + \ln(\tau-1) + \ln(\min \frac{\exp(-\Delta_{\mathcal{L}}(i,j))}{\sum_{t=k+1}^{k+\tau}\exp(-\Delta_{\mathcal{L}}(i,t))}) \end{aligned}$$

Consequently, Theorem 6 holds. □

A.21 A Primer on Certified Robustness

The ultimate goal of certified robustness is to guarantee consistency on the model performance under data perturbation. In specific, it has to ensure that a small perturbation in the input does not change the predicted label. Given a benign example x, the robustness condition to $l_p(\mu)$-norm attacks can be stated as follows:

$$\forall \alpha \in l_p(\mu) : f_i(x+\alpha) > \max_{j:j\neq i} f_j(x+\alpha) \qquad \text{(A.72)}$$

where i is the ground-truth label of the sample x. The condition essentially indicates that a small perturbation α in the input does not change the predicted label i.

PixelDP [108]. To achieve the robustness condition in Eq. A.72, Lecuyer et al. [108] introduce an algorithm, called **PixelDP**. By considering an input x as a database in DP parlance, and individual features as tuples, PixelDP shows that randomizing the function $f(x)$ to enforce DP on a small number of features in the input sample guarantees the robustness of predictions. To randomize $f(x)$, *random noise* σ_r is injected into either input x or an arbitrary hidden layer, resulting in the following (ϵ_r, δ_r)-PixelDP condition:

Lemma 5. (ϵ_r, δ_r)*-PixelDP [108]. Given a randomized scoring function* $f(x)$ *satisfying* (ϵ_r, δ_r)*-PixelDP w.r.t. a* l_p*-norm metric, we have:*

$$\forall i, \forall \alpha \in l_p(1) : \mathbb{E} f_i(x) \leq e^{\epsilon_r} \mathbb{E} f_i(x + \alpha) + \delta_r \tag{A.73}$$

where $\mathbb{E} f_i(x)$ *is the expected value of* $f_i(x)$*,* ϵ_r *is a predefined budget,* δ_r *is a broken probability.*

At the prediction time, a certified robustness check is implemented for each prediction:

$$\hat{\mathbb{E}}_{lb} f_i(x) > e^{2\epsilon_r} \max_{j:j \neq i} \hat{\mathbb{E}}_{ub} f_j(x) + (1 + e^{\epsilon_r})\delta_r \tag{A.74}$$

where $\hat{\mathbb{E}}_{lb}$ and $\hat{\mathbb{E}}_{ub}$ are the lower and upper bounds of the expected value $\hat{\mathbb{E}} f(x) = \frac{1}{n} \sum_n f(x)_n$, derived from the Monte Carlo estimation with an η-confidence, given n is the number of invocations of $f(x)$ with independent draws in the noise σ_r. Passing the check for a given input guarantees that no perturbation up to $l_p(1)$-norm (where 1 is the radius of the l_p-norm ball) can change the model's prediction.

Boosting Randomized Smoothing [86]. Another state-of-the-art approach to certifying robustness is boosting randomized smoothing (RS). In this scheme, an ensemble model $\bar{f}$ of M classifiers trained on the same dataset with different random seeds (same structures and settings). We denote p_1 as the success probability that the ground-truth label l is correctly predicted. Let c_i and $c_{j,j\neq i}$ be the expected values of the clean model over the randomness of the training process, such as $c_i = \mathbb{E}_i(f_i(x))$ and $c_{j,j\neq i} = \mathbb{E}_j(f_j(x))$. Let $t_i = \frac{c_i - c_j}{\sigma_i(M)}$ and z_i be the probability distribution of the classification margin over class i, we have the following robustness condition

$$p_1 := P\big(\bar{f}(x+\alpha) = i\big) \geq 1 - P\big(|z_i - (c_i - c_j)|\big)$$
$$\geq t_i \sigma_i(M) \geq 1 - \sum_{j, j \neq i} \frac{\sigma_j^2(M)}{(c_i - c_j)^2} \tag{A.75}$$

where $\sigma_i(M)$ is the variance of the classification margin z_i.

It is shown in [86] that when $\sigma(M)$ decreases (i.e., a better certified robustness bound), the number of ensemble models M increases, and the lower bound of the success probability p_1 approaches the ground-truth label l.

Given the success probability p_1, confidence γ, number of samples N, and perturbation variance σ_α^2 (up to an incorrect prediction), the probability distribution over the certified radius (R) is defined as follows:

$$P\Big(R = \sigma_\alpha \Phi^{-1}\big(\underline{p_1}(N_1, N, \alpha)\big)\Big) = \mathcal{B}(N_1, N, p_1) \tag{A.76}$$

where Φ^{-1} denotes the inverse Gaussian CDF, $\mathcal{B}(N_1, N, p_1)$ is the probability of drawing N_1 successes in N trials from a Binomial distribution with the success probability p_1 and $\underline{p_1}$ is the lower bound to the success probability of a Bernoulli experiment given N_1 success in N trials with confidence α according to the Clopper-Pearson interval [35]. The certified robustness bound is $R^* = \arg\max R$ such that the condition in Eq. A.75 is satisfied.

Bagging ensemble learning. For the certified robustness at the training-time against XBA, we leverage the bagging ensemble learning method-based certified robustness bounds, which have been demonstrated to be effective in defending against backdoor data poisoning attacks [93]. The ensemble learning approach trains a set of base models and leverages a majority vote to quantify the difference between the lower bound of the class with the highest probability and the upper bound of the class with the second highest probability. Base upon that, one can identify the minimum number of poisoning data samples, called *certified poisoning training size*, that can change the majority vote.

A.22 Proof of Theorem 8

Proof. Recall that $\mathcal{D}$, $\mathcal{D}_o$, and $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_o$ are the proprietary data, the outsourced data, and the poisoned training data, respectively. Given the model prediction on a data sample x using $\mathcal{D}$, denoted as $f(\mathcal{D}, x)$, we ask a simple question: "What is the minimum number poisoning data samples, i.e., certified poisoning training size $r_{\mathcal{D}}$, added into $\mathcal{D}$ to change the model prediction on x: $f(\mathcal{D}, x) \neq f(\mathcal{D}_+, x)$?" After adding $\mathcal{D}_o$ into $\mathcal{D}$, we ask the same question: "What is the minimum number poisoning data samples, i.e., certified poisoning training size $r_{\mathcal{D}'}$, added into $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}_o$ to change the model prediction on x: $f(\mathcal{D}', x) \neq f(\mathcal{D}'_+, x)$?" The difference between $r_{\mathcal{D}}$ and $r_{\mathcal{D}'}$ provides us a certified poisoning training size on $\mathcal{D}_o$. Intuitively, if $\mathcal{D}_o$ does not contain poisoning data examples, then $r_{\mathcal{D}}$ is expected to be relatively the same with $r_{\mathcal{D}'}$. Otherwise, $r_{\mathcal{D}'}$ can be significantly smaller than $r_{\mathcal{D}}$ indicating that $\mathcal{D}_o$ is heavily poisoned with at least $r = r_{\mathcal{D}} - r_{\mathcal{D}'}$ number of poisoning data samples towards opening backdoors on x. Now, our goal is to find $r_{\mathcal{D}}$, $r_{\mathcal{D}'}$, and their connection to r.

Let us denote two sets of poisoned training datasets with at most $r_{\mathcal{D}'}$ poisoned training samples in $\mathcal{D}'$ and at most $r_{\mathcal{D}}$ poisoned training samples in $\mathcal{D}$, namely $B(\mathcal{D}', r_{\mathcal{D}'})$ and $B(\mathcal{D}, r_{\mathcal{D}})$, respectively, as follows:

$$B(\mathcal{D}, r_{\mathcal{D}}) = \{\mathcal{D}_+ \text{ s.t. } |\mathcal{D}_+| - |\mathcal{D}| \leq r_{\mathcal{D}}\} \tag{A.77}$$

$$B(\mathcal{D}', r_{\mathcal{D}'}) = \{\mathcal{D}'_+ \text{ s.t. } |\mathcal{D}'_+| - |\mathcal{D}'| \leq r_{\mathcal{D}'}\} \tag{A.78}$$

We call $s(\mathcal{D})$ as a random subsample data that are sampled from $\mathcal{D}$ with replacement uniformly at random. We denote p_l as the label probability, in which $p_l = \Pr[f(s(\mathcal{D}), x) = l)]$ is the probability that the learned base model predicts label l for x. The ensemble classifier h predicts the label with the largest label probability for x, as:

$$h(\mathcal{D}, x) = \arg \max_{l \in \{0,1\}} p_l \tag{A.79}$$

To prove the certified robustness of the mechanism $h(\mathcal{D}, x)$ against XBA, we need to find certified poisoning training size, which is the minimum number of poisoning training data such that the ensemble classifier changes the prediction for x. Formally, we find the minimum $r_{\mathcal{D}}$ such that the following inequality is satisfied for $\forall \mathcal{D}_+ \in B(\mathcal{D}, r)$:

$$h(\mathcal{D}_+, x) \neq l \Leftrightarrow p'_l < p'_{\neg l} \tag{A.80}$$

where $l \in \{0, 1\}$ and $\neg l$ is the NOT operation of l in the binary classification.

Finding exact values of p'_l and $p'_{\neg l}$ is difficult. Instead of that, we find the lower bound of p'_l and upper bound of $p'_{\neg l}$ (l is the true label of x), we construct regions in the space Ω, which is the joint space of $X = s(\mathcal{D})$ and $Y = s(\mathcal{D}_+)$, satisfying the conditions of the Neyman-Pearson Lemma [141]. This enables us to derive the lower and upper bounds in these regions. Suppose we have a lower bound $\underline{p_l}$ of the largest label probability p_l and an upper bound $\overline{p}_{\neg l}$ of the second largest label probability $p_{\neg l}$ when the classifier is trained on the clean training dataset. Formally, $\underline{p_l}$ and $\overline{p}_{\neg l}$ satisfy:

$$p_l \geq \underline{p_l} \geq \overline{p}_{\neg l} \geq p_{\neg l} \tag{A.81}$$

Following Theorem 1 in [93], we can have the following certified poisoning training size $r_{\mathcal{D}}$ as follows:

Certified poisoning training size $r^*_{\mathcal{D}}$. Given a training dataset $\mathcal{D}$, a model $f(\cdot)$, and a testing sample x, the ensemble classifier h is defined in Eq. A.79. Suppose l is the label with the largest probability predicted by h for x and $\neg l$ is the NOT operation of l in the binary classification. We also have $\underline{p_l}$ and $\overline{p}_{\neg l}$ satisfy Eq. A.81. The h does not predict the label l for x when the certified poisoning training size $r_{\mathcal{D}}$ is bounded by $r^*_{\mathcal{D}}$, i.e., we have:

$$h(\mathcal{D}_+, x) \neq l, \forall \mathcal{D}_+ \in B(\mathcal{D}, r^*_{\mathcal{D}}), \tag{A.82}$$

where $r^*_{\mathcal{D}}$ is the solution to the following optimization problem:

$$r^*_{\mathcal{D}} = \arg\max_{r_{\mathcal{D}}} r_{\mathcal{D}} \tag{A.83}$$

$$\text{s.t.} \max_{|\mathcal{D}|-r_{\mathcal{D}} \leq |\mathcal{D}_+| \leq |\mathcal{D}|+r_{\mathcal{D}}} (\frac{|\mathcal{D}_+|}{|\mathcal{D}|})^k - 2\Big(\frac{\max(|\mathcal{D}, |\mathcal{D}_+|) - r_{\mathcal{D}}}{|\mathcal{D}|}\Big)^k + 1 - (\underline{p}_l - \overline{p}_{\neg l} - \sigma_l - \sigma_{\neg l}) < 0 \tag{A.84}$$

where $\sigma_l = \underline{p}_l - (\lfloor \underline{p}_l n^k \rfloor)/n^k$ and $\sigma_{\neg l} = \lceil \overline{p}_{\neg l} n^k \rceil / n^k - \overline{p}_{\neg l}$.

Solving the problem in Eqs. A.83 and in Eq. A.84 [93], we obtain the results:

$$r^*_{\mathcal{D}} = \lceil |\mathcal{D}| \Big(\sqrt[k]{1 + (\underline{p}_l - \overline{p}_{\neg l} - \sigma_l - \sigma_{\neg l})} \Big) - 1 \rceil \tag{A.85}$$

Certified poisoning training size $r^*_{\mathcal{D}'}$. Similar to find the certified poisoning training size $r^*_{\mathcal{D}}$, we obtain:

$$r^*_{\mathcal{D}'} = \lceil |\mathcal{D}'| \Big(\sqrt[k]{1 + (\underline{p}'_l - \overline{p}'_{\neg l} - \sigma'_l - \sigma'_{\neg l})} \Big) - 1 \rceil \tag{A.86}$$

where $\sigma'_l = \underline{p}'_l - (\lfloor \underline{p}'_l n'^k \rfloor)/n'^k$ and $\sigma'_{\neg l} = \lceil \overline{p}'_{\neg l} n'^k \rceil / n'^k - \overline{p}'_{\neg l}$.

Certified poisoning training size r. Intuitively, if $\mathcal{D}_o$ does not consist of poisoning data examples, then $r_{\mathcal{D}}$ is expected to be relatively the same with $r_{\mathcal{D}'}$. Otherwise, $r_{\mathcal{D}}$ can be significantly smaller than $r_{\mathcal{D}'}$ indicating that $\mathcal{D}_o$ is heavily poisoned with at least $r = r_{\mathcal{D}} - r_{\mathcal{D}'}$ number of poisoning data samples towards opening backdoors on x. Therefore, after obtaining $r^*_{\mathcal{D}'}$ and $r^*_{\mathcal{D}}$, we have:

$$r = r^*_{\mathcal{D}} - r^*_{\mathcal{D}'} \tag{A.87}$$

Tightness of the certified poisoning training size r. The bound in Theorem 8 is tight and there is no existing smaller value of r for the XBA to be successfully carried out. In our bound for the poisoning training size, the propriety data $\mathcal{D}$ cannot be changed by XBA; hence, $r^*_{\mathcal{D}}$ is fixed. In addition, $r^*_{\mathcal{D}'}$ is the minimum value that the prediction can be changed, so it is considered as a fixed value when the outsourced data $\mathcal{D}'$ remains the same, and it can be changed if the outsourced data is changed. For example, if one more poisoning sample is added into $\mathcal{D}_o$, so $r^*_{\mathcal{D}'}$ becomes $r^*_{\mathcal{D}'} + 1$, then the minimum value is also changed to be $r^*_{\mathcal{D}'} + 1$. As a result, the certified robustness bound derived in Eq. 4.6 is tight.

□

A.23 Certified Robustness Bound using boosting randomized smoothing.

Directly applying the boosting RS (Appx. A.21) to XRAND by *only* using the noise α that is associated with $\Delta_{\alpha|w}$ may not be effective. In the boosting RS, different base models (i.e., same structures and settings, just different random seeds when training), are trained on clean data; meanwhile, in XRAND, the training data is backdoored data. As a result, the backdoored (and noisy) data in XRAND along with different base models in the boosting RS may increase the variance of the outputs and then narrows down the certified robustness bound, which reduces the size of the bound. This results in a gap between the boosting RS and XRAND. To close this gap, we add a smaller amount of noise, namely α_1, into the input in addition to the noise α associated with $\Delta_{\alpha|w}$, as follows:

$$p_1 := P\big(\bar{f}(x + \alpha + \alpha_1) = l\big) \geq 1 - \frac{\sigma^2_{new}(M)}{(c_l - c_{\neg l})^2} \tag{A.88}$$

Similarly, σ^2_{new} is the variance of the classification margin, which is computed as in [86] associated with the new noise $(\alpha + \alpha_1)$. Given the success probability p_1 in Eq. A.88 and the perturbation variance $\sigma^2_{\alpha+\alpha_1}$ (up to an incorrect prediction), the probability distribution over the certifiable radius R is computed as follows:

$$P\Big(R = \sigma_{\alpha+\alpha_1}\Phi^{-1}\big(\underline{p_1}(N_1, N)\big)\Big) = \mathcal{B}(N_1, N, p_1) \tag{A.89}$$

where $\mathcal{B}(N_1, N, p_1)$ is the probability of drawing N_1 successes in N trials from a Binomial distribution with success probability p_1 and $\underline{p_1}(N_1, N)$ is the lower bound of the success probability of a Bernoulli experiment given N_1 successes in N trials. The certified robustness bound is $R^* = \arg\max R$ such that the robustness condition in Eq. A.88 is satisfied.

A.24 Experimental settings and results

Platform. Our experiments in this paper are implemented using Python 3.8 and conducted on a single GPU-assisted compute node that is installed with a Linux 64-bit operating system. The allocated resources include 8 CPU cores (AMD EPYC 7742 model) with 2 threads per core, and 100GB of RAM. The node is also equipped with 8 GPUs (NVIDIA DGX A100 SuperPod model), with 80GB of memory per GPU.

Dataset. We conduct the XBA on malware classifiers against XRAND explanations on three malware datasets. EMBER [12] is a representative benchmark dataset containing malicious and benign samples used for training malware classifiers. It consists of 2,351-dimensional feature vectors extracted from 1.1M Portable Executable (PE) files. The dataset contains $600,000$ training samples equally split between goodware and malware, and $200,000$ test samples, with the same class balance.

We also test with malicious PDF files using the Contagio PDF dataset [185] which contains 10,000 PDF files equally split between goodware and malware, each sample is represented by a 135-dimensional feature vector. Finally, we evaluate our work on the Drebin dataset [104] consisting of 5,560 malware and 123,453 goodware Android apps. Each sample contains 545,333 features extracted from the applications.

As mentioned in our threat model, we do not restrict the set of features that can be used by the attacker as backdoor triggers, so that our defense can be assessed against the strongest adversary.

Models. For the EMBER dataset, we train two classifiers: LightGBM and EmberNN. LightGBM is a gradient boosting model released together with the EMBER dataset. It achieves good performance for malware binary classification with 98.61% accuracy. Following Anderson et al. [12], we use default parameters for training LightGBM (100 trees and 31 leaves per tree). EmberNN composes of four fully connected layers, in which the first three layers use ReLU activation functions, and the last layer uses a sigmoid activation function [178]. EmberNN attains 99.14% accuracy.

Experimental results on Contagio and Drebin. We conduct additional experiments on the Contagio PDF [185] and Drebin [104] datasets. The Contagio PDF dataset contains 10,000 PDF files equally split between goodware and malware, each sample is represented by a 135-dimensional feature vector. The Drebin dataset consists of 5,560 malware and 123,453 goodware Android apps. Each sample contains 545,333 features extracted from the applications. We use a Random Forest classifier for Contagio and a Linear Support Vector Machine for Drebin, and fix the trigger size to 30 features. The classifiers are released by (Severi et al, 2021) and we keep the same experimental settings. Figure A.22 shows the attack success rate of XBA using the explanations returned by XRAND.

On the Contagio and Drebin datasets, we observe similar behavior to our experiments with the EMBER dataset. The attack success rate decreases as we tighten the privacy budget, since the attacker has less access to the desired features. At 1% poison rate and $\varepsilon = 10.0$, XRAND can maintain a low success rate of 9.6% in the Contagio dataset (Fig. A.22(a)) and 7% in the Drebin dataset (Fig. A.22(b)).

Certified Robustness. To evaluate the certified robustness mechanism, as in [93, 108] we use the following metric:

$$\text{certified accuracy} = \sum_{n=1}^{|test|} \frac{isCorrect(\mathcal{X}_n) \ \& \ isRobust(\mathcal{X}_n)}{|test|} \tag{A.90}$$

where $|test|$ is the number of testing samples, $isCorrect(\cdot)$ returns 1 if the model makes a correct prediction (else, returns 0), and $isRobust(\cdot)$ returns 1 if the robustness size is larger than a given attack size L (else, returns 0). When running with PixelDP and Boosting RS, we adopt the hyperparameter settings from the papers [108] and [86], respectively.

To verify the certified robustness at the training time, we conduct experiments with a wide range of $\varepsilon \in [0.1, 100.0]$. In this setting, we create $6,000$ poisoned samples by adding the trigger that follows the two-step LDP-preserving mechanism. From Fig. A.23, we observe that the smaller privacy budget ε, the higher certified accuracy. In fact, the smaller ε imposes more noise which provides a better LDP guarantee and the model trained with noisier data is more robust against the backdoor attacks; hence, resulting in higher certified accuracy. It is obvious that the certified accuracy decreases when the threshold number of poisoning training samples r_{tr} increases, since the term $isRobust(\cdot)$ decreases. For the inference-time bound, with the boosting RS certified robustness, we also conduct experiments with a wide range of privacy budget $\varepsilon \in [0.1, 50.0]$, the noise level $\sigma_{\alpha+\alpha_1} \in [0.25, 1.0]$, and the radius $r \in [0.25, 2.0]$. We consider $M = 5$ base classifiers in the ensemble model of the boosting RS. With PixelDP, we conduct experiments given tight privacy budgets, i.e., $\varepsilon \in [0.1, 1.0]$. We obtain high certified accuracy for these tight privacy budgets, i.e., 89.17% and 90.42% for $\varepsilon \in [0.1, 1.0]$, compared with 50% obtained with the boosting RS. Here, we are dealing with a binary classification problem; therefore, the variance of the model output is smaller than a multi-class classification problem. That potentially affects the effectiveness of the boosting RS.

A.25 Visualizing XRAND

Figures A.24, A.25, and A.26 visualize the explanations of some test samples before and after applying XRAND. For the most part, the explanations from SHAP and XRAND largely resemble one another.

A.26 Proof of Theorem 9

Proof. Let us denote $A_{1:i}$ as $A_1, \dots, A_i$, we have:

$$c(\theta^\tau; A_\tau, \{\theta^i\}_{i<\tau}, \mathsf{data}_\tau, \mathsf{data}'_\tau) = \log \frac{Pr[A_\tau(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau]}{Pr[A_\tau(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau]}$$
$$= \log \prod_{i=1}^{\tau} \frac{Pr[A_i(\theta^{i-1}, \mathsf{data}_i) = \theta^i | A_{1:i-1}(\{\theta^j\}_{j<i-1}, \mathsf{data}_{1:i-1}) = \theta^{1:i-1}]}{Pr[A_i(\theta^{i-1}, \mathsf{data}'_i) = \theta^i | A_{1:i-1}(\{\theta^j\}_{j<i-1}, \mathsf{data}'_{1:i-1}) = \theta^{1:i-1}]}$$
$$= \sum_{i=1}^{\tau} \log \frac{Pr[A_i(\theta^{i-1}, \mathsf{data}_i) = \theta^i | A_{1:i-1}(\{\theta^j\}_{j<i-1}, \mathsf{data}_{1:i-1}) = \theta^{1:i-1}]}{Pr[A_i(\theta^{i-1}, \mathsf{data}'_i) = \theta^i | A_{1:i-1}(\{\theta^j\}_{j<i-1}, \mathsf{data}'_{1:i-1}) = \theta^{1:i-1}]}$$
$$= \sum_{i=1}^{\tau} c(\theta^i; A_i, \{\theta^j\}_{j<i}, \mathsf{data}_i, \mathsf{data}'_i)$$

Consequently, Theorem 9 does hold. □

A.27 Proof of Theorem 10

Proof. $\forall \tau \in \mathbf{T}$, let $\overline{D}_\tau$ and $\overline{D}'_\tau$ be neighboring datasets differing at most one tuple $x_e \in \overline{D}_\tau$ and $x'_e \in \overline{D}'_\tau$, and any two neighboring episodic memories $\mathbb{M}_\tau$ and $\mathbb{M}'_\tau$. Let us denote Alg. 3 as the mechanism A in Definition 9. We first show that Alg. 3 achieves typical DP protection. $\forall \tau$ and D_{ref}, we have that

$$Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau\big] \tag{A.91}$$
$$= Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1})\big) Pr\big(\overline{D}_\tau\big) Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})\big) \times Pr\big(\overline{\mathcal{R}}_{\overline{D}_{ref}}(\theta_1^{\tau-1})\big) Pr\big(\overline{D}_{ref}\big) Pr\big(\overline{\mathcal{L}}_{\overline{D}_{ref}}(\theta_2^{\tau-1})\big)$$

Therefore, we further have

$$\frac{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau\big]}{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau\big]} \tag{A.92}$$
$$= \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1^{\tau-1})\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2^{\tau-1})\big)} \times \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_{ref}}(\theta_1^{\tau-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_{ref}}(\theta_1^{\tau-1})\big)} \frac{Pr\big(\overline{D}_{ref}\big)}{Pr\big(\overline{D}'_{ref}\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_{ref}}(\theta_2^{\tau-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_{ref}}(\theta_2^{\tau-1})\big)}$$

In addition, with $\overline{\mathcal{D}} = \{\overline{D}_1, \dots, \overline{D}_m\}$, we also have that:

$$\exists! \overline{D}_\tau \in \overline{\mathcal{D}} \textit{ s.t. } x_e \in \overline{D}_\tau \textit{ and } \exists! \overline{D}'_\tau \in \overline{\mathcal{D}}' \textit{ s.t. } x'_e \in \overline{D}'_\tau \tag{A.93}$$

Together with Eq. A.93, by having disjoint and fixed datasets in the episodic memory, we have that:

$$(x_e \in \overline{D}_\tau \textit{ or } x_e \in \overline{D}_{ref}), \textit{ but } (x_e \in \overline{D}_\tau \textit{ and } x_e \in \overline{D}_{ref}) \tag{A.94}$$

Without loss of the generality, we can assume that $x_e \in \overline{D}_\tau$: Eqs. A.92 - A.94 $\Rightarrow$

$$\frac{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau\big]}{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau\big]} = \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1^{\tau-1})\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2^{\tau-1})\big)}$$
$$\leq (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2) \tag{A.95}$$

This is also true when $x_e \in \overline{D}_{ref}$ *and* $x_e \notin \overline{D}_\tau$.

As a result, we have

$$\forall \tau \in [1, m] : \frac{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau\big]}{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau\big]} \leq (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2) \tag{A.96}$$

After one training step, $\overline{D}_\tau$ will be placed into the episodic memory $\mathbb{M}_\tau$ to create the memory $\mathbb{M}_{\tau+1}$. In the next training task, $\overline{D}_\tau$ can be randomly selected to compute the episodic gradient g_{ref}. This computation does not incur any additional privacy budget consumption for the dataset $\overline{D}_\tau$, by applying the Theorem 4 in [151], which allows us to *compute gradients across an unlimited number of training steps* using $\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1})$ and $\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})$. Therefore, if the same privacy budget is used for all the training tasks in $\mathbf{T}$, we will have only one privacy loss for every tuple in all the tasks. The optimization in one task does not affect the DP guarantee of any other tasks. Consequently, we have

$$\nexists \epsilon' < (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2), \exists i \leq m$$
$$\text{s.t. } Pr\big[A(\{\theta^j\}_{j<i}, \mathsf{data}_i) = \theta^i\big] \leq e^{\epsilon'} Pr\big[A(\{\theta^j\}_{j<i}, \mathsf{data}'_i) = \theta^i\big] \tag{A.97}$$

Eq. A.97 can be further used to prove the Lifelong DP protection. Given data_m where $M_t = \overline{D}_t$ in Alg. 3, we have that

$$Pr\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big] = \prod_{i=1}^{m} Pr\big[A(\{\theta^j\}_{j<i}, \mathsf{data}_i) = \theta^i\big] \tag{A.98}$$

Therefore, we have

$$\frac{Pr\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big]}{Pr\big[A(\mathsf{data}'_m) = \{\theta^i\}_{i\in[1,m]}\big]} = \prod_{i=1}^{m} \frac{Pr\big[A(\{\theta^j\}_{j<i}, \mathsf{data}_i) = \theta^i\big]}{Pr\big[A(\{\theta^j\}_{j<i}, \mathsf{data}'_i) = \theta^i\big]} \tag{A.99}$$
$$= \prod_{i=1}^{m} \Big[\frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_i}(\theta_1^{i-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_i}(\theta_1^{i-1})\big)} \frac{Pr\big(\overline{D}_i\big)}{Pr\big(\overline{D}'_i\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_i}(\theta_2^{i-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_i}(\theta_2^{i-1})\big)} \times \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}^i_{ref}}(\theta_1^{i-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}^{i'}_{ref}}(\theta_1^{i-1})\big)} \frac{Pr\big(\overline{D}^i_{ref}\big)}{Pr\big(\overline{D}^{i'}_{ref}\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}^i_{ref}}(\theta_2^{i-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}^{i'}_{ref}}(\theta_2^{i-1})\big)}\Big]$$

where $\mathsf{data}'_m = \{\overline{\mathcal{D}}, \{M'_i\}_{i\in[1,m]}\}$, and $M'_i = \overline{D}'_i$ in Alg. 3.

Since all the datasets are non-overlapping, i.e., $\cap_{i\in[1,m]} D_i = \emptyset$, given an arbitrary tuple x_e, we have that

$$\exists! \overline{D}_\tau \in \overline{\mathcal{D}} \textit{ s.t. } x_e \in \overline{D}_\tau \textit{ and } \exists! \overline{D}'_\tau \in \overline{\mathcal{D}}' \textit{ s.t. } x'_e \in \overline{D}'_\tau \tag{A.100}$$

Thus, the optimization of $\{\theta_1^i, \theta_2^i\} = \arg\min_{\theta_1,\theta_2}[\overline{\mathcal{R}}_{\overline{D}_i}(\theta_1^{i-1}) + \overline{\mathcal{L}}_{\overline{D}_i}(\theta_2^{i-1})]$ for any other task i different from τ does not affect the privacy protection of x_e in $\overline{\mathcal{D}}$. From Eqs. A.99 and A.100, we have

$$\frac{Pr\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big]}{Pr\big[A(\mathsf{data}'_m) = \{\theta^i\}_{i\in[1,m]}\big]} \tag{A.101}$$
$$= \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{\tau-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1^{\tau-1})\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{\tau-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2^{\tau-1})\big)} \times \prod_{i=1}^{m} \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}^i_{ref}}(\theta_1^{i-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}^{i'}_{ref}}(\theta_1^{i-1})\big)} \frac{Pr\big(\overline{D}^i_{ref}\big)}{Pr\big(\overline{D}^{i'}_{ref}\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}^i_{ref}}(\theta_2^{i-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}^{i'}_{ref}}(\theta_2^{i-1})\big)}$$

The worse privacy leakage case to x_e is that $\overline{D}_\tau$ is used in every $\overline{D}^i_{ref}$, i.e., $\tau = 1$ and $\forall i \in [2, m] : \overline{D}^i_{ref} = \overline{D}_\tau$, with $\overline{D}^1_{ref} = \emptyset$. Meanwhile, the least privacy leakage case to x_e is that $\overline{D}_\tau$ is not used in any $\overline{D}^i_{ref}$, i.e., $\forall i \in [2, m] : \overline{D}^i_{ref} \neq \overline{D}_\tau$, with $\overline{D}^1_{ref} = \emptyset$. In order to bound the privacy loss, we consider the worse case; therefore, from Eq. A.101, we have

$$\frac{Pr\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big]}{Pr\big[A(\mathsf{data}'_m) = \{\theta^i\}_{i\in[1,m]}\big]} \leq \prod_{i=1}^{m} \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{i-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1^{i-1})\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{i-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2^{i-1})\big)} \tag{A.102}$$

Eq. A.102 is equivalent to the continuously training of our model by optimizing $\overline{\mathcal{R}}$ and $\overline{\mathcal{L}}$ with $\overline{D}_\tau$ used as both the current task and the episodic memory, across m steps. By following the Theorem 4 in [151], the privacy budget is not accumulated across training steps. Therefore, we have that

$$\forall m \in [1, \infty) : \frac{Pr\big[A(\mathsf{data}_m) = \{\theta^i\}_{i\in[1,m]}\big]}{Pr\big[A(\mathsf{data}'_m) = \{\theta^i\}_{i\in[1,m]}\big]} \leq \prod_{i=1}^{m} \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1^{i-1})\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1^{i-1})\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2^{i-1})\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2^{i-1})\big)}$$

$$= \frac{Pr\big(\overline{\mathcal{R}}_{\overline{D}_\tau}(\theta_1)\big)}{Pr\big(\overline{\mathcal{R}}_{\overline{D}'_\tau}(\theta_1)\big)} \frac{Pr\big(\overline{D}_\tau\big)}{Pr\big(\overline{D}'_\tau\big)} \frac{Pr\big(\overline{\mathcal{L}}_{\overline{D}_\tau}(\theta_2)\big)}{Pr\big(\overline{\mathcal{L}}_{\overline{D}'_\tau}(\theta_2)\big)} \leq (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2) \tag{A.103}$$

In the least privacy leakage case, with $\forall\tau \leq m$, we have that

$$\frac{Pr\big[A(\mathsf{data}_\tau) = \{\theta^i\}_{i\in[1,\tau]}\big]}{Pr\big[A(\mathsf{data}'_\tau) = \{\theta^i\}_{i\in[1,\tau]}\big]} \geq \frac{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}_\tau) = \theta^\tau\big]}{Pr\big[A(\{\theta^i\}_{i<\tau}, \mathsf{data}'_\tau) = \theta^\tau\big]} \geq (\epsilon_1 + \frac{\epsilon_1}{\gamma_{\mathbf{x}}} + \frac{\epsilon_1}{\gamma} + \epsilon_2) \tag{A.104}$$

As a result, we have that

$$\nexists(\epsilon' < \epsilon, \tau \leq m) : Pr\big[A(\mathsf{data}_\tau) = \{\theta^i\}_{i\in[1,\tau]}\big] \leq e^{\epsilon'} Pr\big[A(\mathsf{data}'_\tau) = \{\theta^i\}_{i\in[1,\tau]}\big] \tag{A.105}$$

where $\epsilon = (\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$.

From Eqs. A.103 and A.105, we have that Alg. 3 achieves $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-Lifelong DP in learning $\{\theta^i\}_{i\in[1,m]} = \{\theta_1^i, \theta_2^i\}_{i\in[1,m]}$. Theorem 10 does hold. □

A.28 L2DP-ML with Streaming Batch Training

A.29 HARW Dataset

Data Collection. We utilize Android smartphones to collect smartphone sensor data "in the wild" from university students as subjects for the following reasons: **(1)** University students should have relatively good access to the smartphones and related technologies; **(2)** University students should be more credible and easier to be motivated than other sources (e.g., recruiting test subjects on crowd-sourcing websites); and **(3)** It will be easier for our team to recruit and distribute rewards to students. We launched two data collection runs at two universities for three months each. During the course of three months, we let the participants to collect data and labels by themselves (in the wild), and only intervene through reminding emails if we saw a decline in the amount of daily activities. A total of 116 participants were recorded after the two data collection runs.

Data Processing. For the demonstration purpose of this paper, we use only accelerometer data. Our data processing consists of the following steps: **(1)** Any duplicated data points (e.g., data points that have the same timestamp) are merged by taking the average of their sensor values; **(2)** Using 300 milliseconds as the threshold, continuous data sessions are identified and separated by breaking up the data sequences at any gap that is larger than the threshold; **(3)** Data sessions that have unstable or unsuitable sampling rates are filtered out. We only keep the data sessions that have a stable sampling rate of 5Hz, 10Hz, 20Hz, or 50Hz; **(4)** The label sessions that are associated with each data session (if any) are identified from the raw labels. Note that the label sessions are also filtered with the following two criteria to ensure good quality: (a) The first 10 seconds and the last 10 seconds of each label session are trimmed, due to the fact that users were likely operating the phone during these time periods; (b) Any label session longer than 30 minutes is trimmed down to 30 minutes, in order to mitigate the potential inaccurate labels due to users' negligence (forgot to turn off labeling); and **(5)** We sample data segments at the size of 100 data points with sliding windows. Different overlapping percentages were used for different classes and different sampling rates. The majority classes have 25% overlapping to reduce the number of data segments, while the minority classes have up to 90% overlapping to increase the available data segments. The same principle is applied to sessions with different sampling rates. We sample 15% of data for testing, while the rest are used for training (Table A.4).

Data Normalization. In our L2DP-ML models, we normalize the accelerometer data with the following steps: **(1)** We compute the mean and variance of each axis (i.e., X, Y, and Z) using only training data to avoid information leakage from the training phase to the testing phase. Then, both training and testing data are normalized with z-score, based on the mean and variance computed from training data; **(2)** Based on this, we clip the values in between $[min, max] = [-2, 2]$ for each axis, which covers at least 90% of possible data values; and **(3)** Finally, all values are linearly scaled to $[-1, 1]$ to finish the normalization process, as $x = 2 \times [\frac{x-min}{max-min} - 1/2]$.

Table A.4 Statistics of the HARW Dataset

Class	Description	N training	N testing
Walking	Walking	49376	8599
Sitting	Exclude in vehicle	52448	8744
In-Vehicle, Car	Driving, sitting	49536	8586
Cycling		14336	2537
Workout, Running		1984	319
*All classes exclude phone position = "Table"			

In the HARW dataset, each data tuple includes 100 values $\times$ 3 channels of the accelerometer sensor, i.e., 300 values in total as a model input. The classification output includes five classes of human activities, i.e., walking, sitting, in car, cycling, and running (Table A.4, Appx. A.29). Given 20Hz, 5Hz, 10Hz, and 50Hz tasks, we correspondingly have 881, 7553, 621, and 156,033 data points in training and 159, 1,297, 124, and 27,134 data points in testing.

Baseline Model Performance. We conducted experiments on the HARW dataset in a centralized training on the whole dataset including all the data sampling rates using following baselines: 1) CNN-based model with the numbers of convolution-channels set to 32, 64, 128, denoted as CNN-32, CNN-64, CNN-128, respectively; 2) Bidirectional LSTM (BiLSTM); and 3) CNN-based models proposed by [90], with additional features (CNN-Ig) and without additional features (CNN-Ig-featureless) using the Ignatov's recommended settings in [90].

As in Table A.5, our model trained on each task independently achieves competitive results with these baselines under a rigorous DP budget ($\epsilon = 0.2$), i.e., 77%, 76%, 75%, 58%, on the 5Hz, 10Hz, 20Hz, and 50Hz learning tasks respectively. Although the number of 50Hz training data points is larger than other tasks, the data labels are noisy and collected

Table A.5 Baseline Results On The HARW Dataset

Model	Accuracy (%)
CNN-32	81.86
CNN-64	82.49
CNN-128	82.62
BiLSTM	78.68
CNN-Ig	76.39
CNN-Ig-featureless	77.08

in short-time periods due to the limited computational resources on mobile devices; thus, the model performance in the 50Hz learning task is lower.

A.30 Hyper-parameter Grid-Search and Supplemental Results

Model Configuration. In the permuted MNIST and the Split MNIST datasets, we used three convolutional layers (32, 64, and 96 features). Each hidden neuron connects with a 5x5 unit patch. A fully-connected layer has 512 units. In the permuted CIFAR-10 and the Split CIFAR-10/100 datasets, we used a Resnet-18 network (64, 64, 128, 128, and 160 features) with kernels (4, 3, 3, 3, and 3). One fully-connected layer has 256 neurons. In the HARW dataset, we used three convolutional layers (32, 64, and 96 features). Each hidden neuron connects with a 2x2 unit patch. A fully-connected layer has 128 units.

In the Split CIFAR-10 and CIFAR-100 setting, there are 11 tasks, in which the first task is the full CIFAR-10 classification task, and the remaining 10 tasks consist of splits from the CIFAR-100 dataset. Each split contains 10 classes from the CIFAR-100. We adopt this approach from [199]. In the Split MNIST setting, there are 5 tasks, in which each task consists of 2 classes from the MNIST dataset. There is no overlapping classes between tasks in the Split CIFAR-10 and CIFAR-100, and in the Split MNIST.

In order to be fair in comparison with the L2DP-ML and A-gem mechanisms, we conducted experiments over a wide range of privacy hyper-parameters such as privacy budget (ϵ), noise scale (z), and sensitivity to select the best hyper-parameters in NaiveGaussian mechanism in our experiments. The search ranges and their results (i.e., average accuracy over all tasks) are provided in Table A.6. We reported the best results, i.e., highest average accuracy over all tasks, of the hyper-parameter grid-search experiments.

Table A.6 Average Accuracy (%) in Hyper-parameter Grid-search of NaiveGaussian Mechanism Given the Permuted CIFAR-10 Dataset

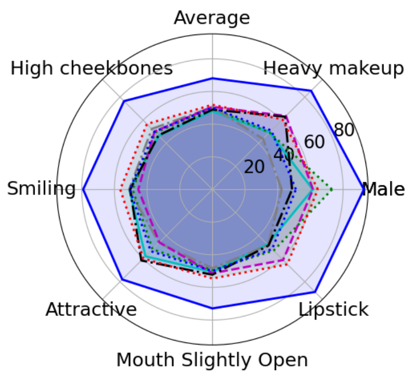
Privacy budget (ϵ)	Noise scale (z) \ Clipping bound	0.01	0.1	1.0
$\epsilon = 4.0$	$z = 2.5$	13.68	11.23	10.26
	$z = 2.4$	12.66	11.99	9.98
	$z = 2.3$	11.56	11.40	10.09
	$z = 2.2$	13.79	11.99	10.30
	$z = 2.1$	13.50	11.39	10.11
$\epsilon = 7.0$	$z = 2.0$	15.12	12.94	10.26
	$z = 1.9$	14.67	12.39	10.34
	$z = 1.8$	14.32	11.79	10.28
	$z = 1.7$	15.26	12.55	11.33
	$z = 1.6$	14.64	12.28	11.04
$\epsilon = 10.0$	$z = 1.5$	14.79	12.23	10.80
	$z = 1.4$	15.71	13.34	10.66
	$z = 1.3$	15.12	12.96	11.49
	$z = 1.2$	14.65	12.05	10.64
	$z = 1.1$	11.42	11.15	10.14

Table A.7 Average Forgetting Measure

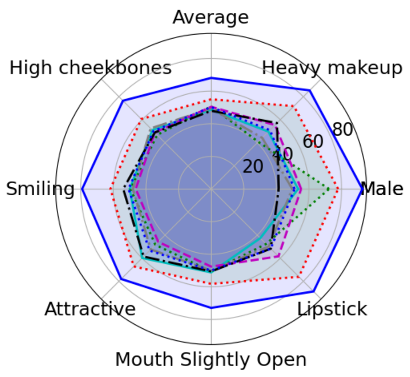
		L2DP-ML	A-gem
Split MNIST	$\epsilon = 0.5$	0.056 ± 0.00324	0.195 ± 0.00941
	$\epsilon = 1$	0.019 ± 0.00526	
Split CIFAR-10/100	$\epsilon = 4$	0.027 ± 0.00264	0.195 ± 0.00688
	$\epsilon = 4$ (2 epochs)	0.033 ± 0.00276	
	$\epsilon = 4$ (3 epochs)	0.046 ± 0.00307	
	$\epsilon = 7$	0.027 ± 0.00165	
	$\epsilon = 10$	0.021 ± 0.00429	

Table A.8 Average Forgetting of the Order of [20Hz, 5Hz, 10Hz, 50Hz]

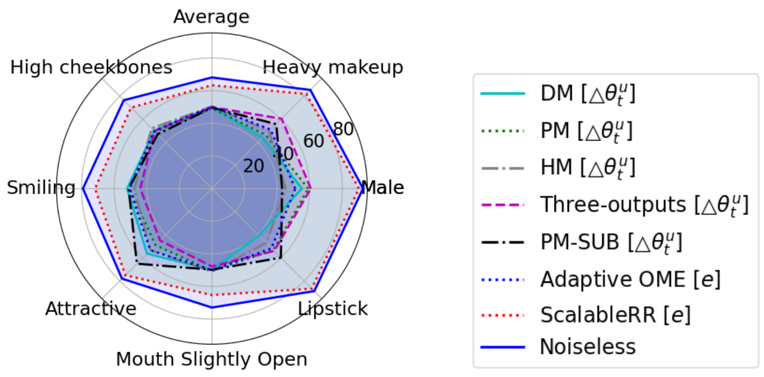
HARW (20Hz - 5Hz - 10Hz - 50Hz)	L2DP-ML ($\epsilon = 0.2$)	L2DP-ML ($\epsilon = 0.5$)	L2DP-ML ($\epsilon = 1$)
	$0.0928 \pm$ 5.34e-5	$0.0921 \pm$ 8.64e-5	$0.089 \pm$ 8.64e-5
	A-gem	Balanced A-gem	Balanced L2DP-ML ($\epsilon = 0.2$)
	$0.0866 \pm$ 1.1e-4	0.1723 ± 0.00066	0.144 ± 0.0031
	L2DP-ML ($\epsilon = 0.2$, 2 epochs)	L2DP-ML ($\epsilon = 0.2$, 5 epochs)	Heterogeneous L2DP-ML ($\epsilon = 0.2$)
	0.1161 ± 0.0003	0.1792 ± 0.0017	0.1395 ± 0.00026



((a)) $\epsilon_X = 1.0$

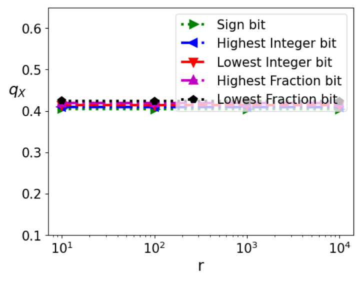


((b)) $\epsilon_X = 5.0$

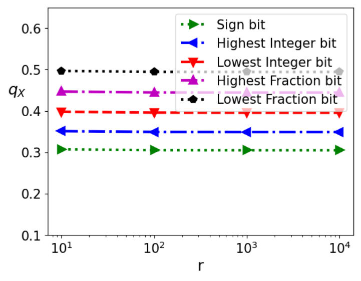


((c)) $\epsilon_X = 10.0$

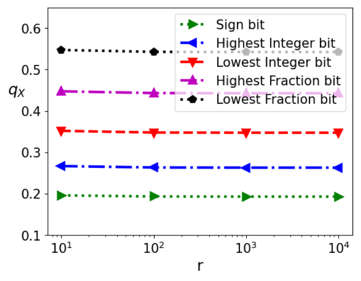
Figure A.6 AUC values of each algorithm applied on the gradients $\triangle\theta_t^u$ in the CelebA dataset.



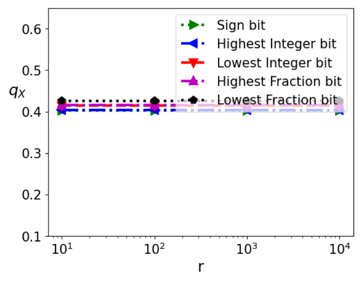
((**a**)) $l = 5$ and $\epsilon_X = 0.1$



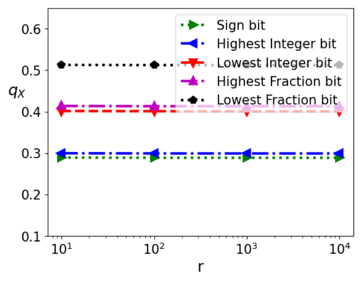
((**b**)) $l = 5$ and $\epsilon_X = 1.0$



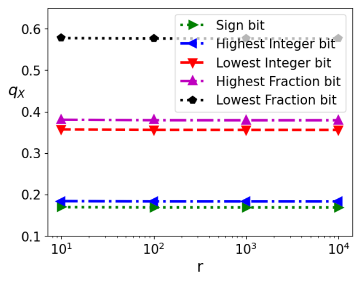
((**c**)) $l = 5$ and $\epsilon_X = 2.0$



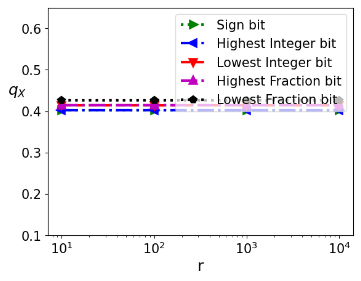
((**d**)) $l = 20$ and $\epsilon_X = 0.1$



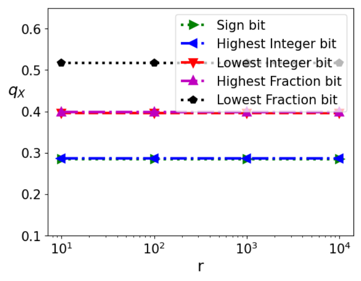
((**e**)) $l = 20$ and $\epsilon_X = 1.0$



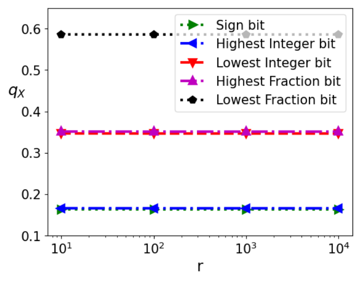
((**f**)) $l = 20$ and $\epsilon_X = 2.0$



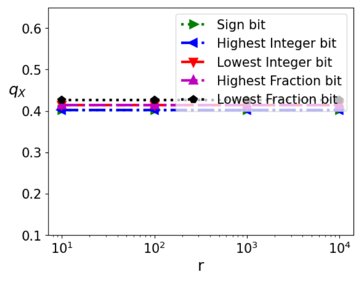
((**g**)) $l = 100$ and $\epsilon_X = 0.1$



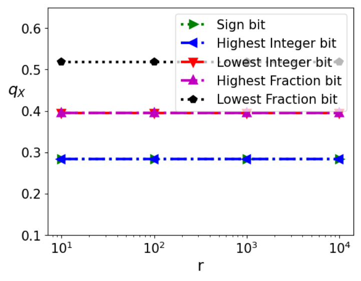
((**h**)) $l = 100$ and $\epsilon_X = 1.0$



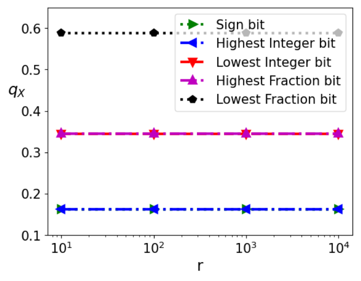
((**i**)) $l = 100$ and $\epsilon_X = 2.0$



((**j**)) $l = 1,000$ and $\epsilon_X = 0.1$

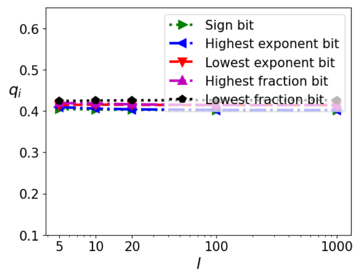


((**k**)) $l = 1,000$ and $\epsilon_X = 1.0$

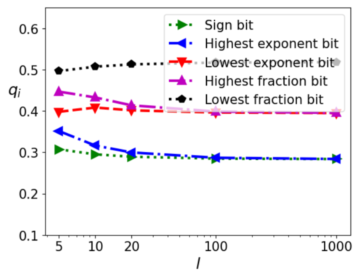


((**l**)) $l = 1,000$ and $\epsilon_X = 2.0$

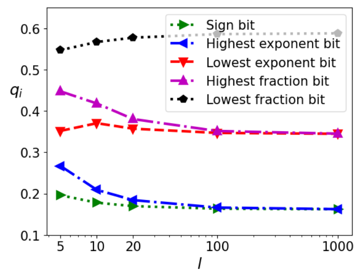
Figure A.7 Randomization probability q ($p = 1 - q$) as a function of r with fixed ϵ_X.



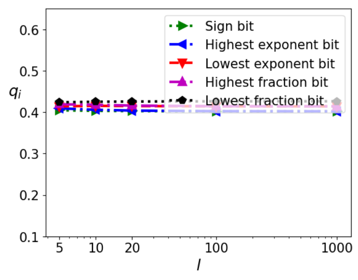
((a)) $r = 10$ and $\epsilon_X = 0.1$



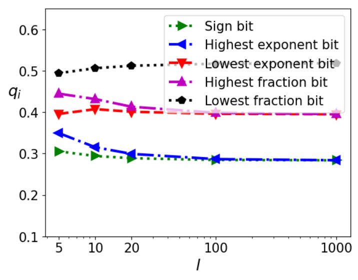
((b)) $r = 10$ and $\epsilon_X = 1.0$



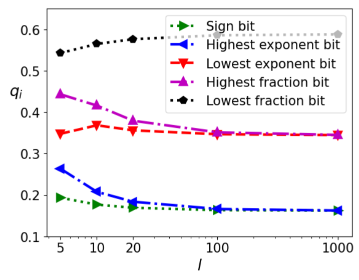
((c)) $r = 10$ and $\epsilon_X = 2.0$



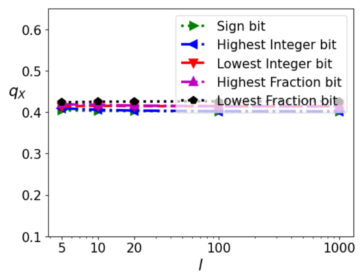
((d)) $r = 100$ and $\epsilon_X = 0.1$



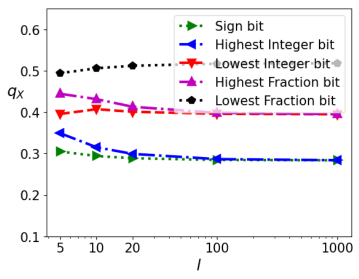
((e)) $r = 100$ and $\epsilon_X = 1.0$



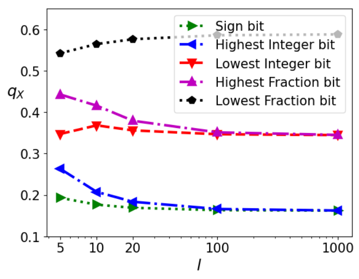
((f)) $r = 100$ and $\epsilon_X = 2.0$



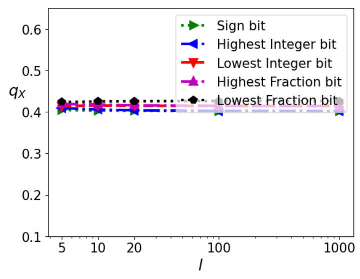
((g)) $r = 1,000$ and $\epsilon_X = 0.1$



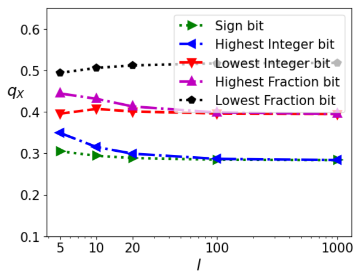
((h)) $r = 1,000$ and $\epsilon_X = 1.0$



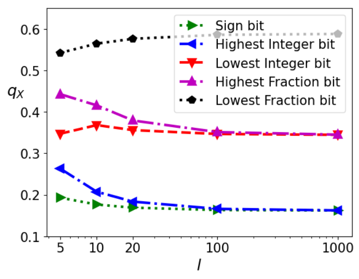
((i)) $r = 1,000$ and $\epsilon_X = 2.0$



((j)) $r = 10,000$ and $\epsilon_X = 0.1$



((k)) $r = 10,000$ and $\epsilon_X = 1.0$



((l)) $r = 10,000$ and $\epsilon_X = 2.0$

Figure A.8 Randomization probability q_i ($p_i = 1 - q_i$) as a function of l with fixed r and ϵ.

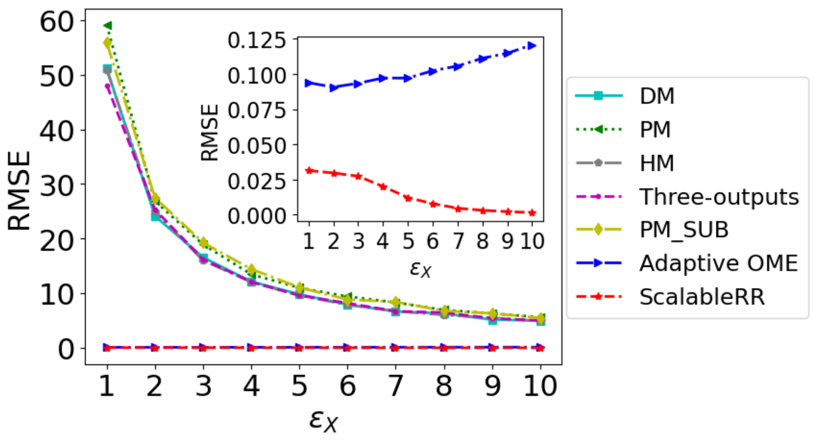


Figure A.9 RMSE comparison as a function of ϵ_X.

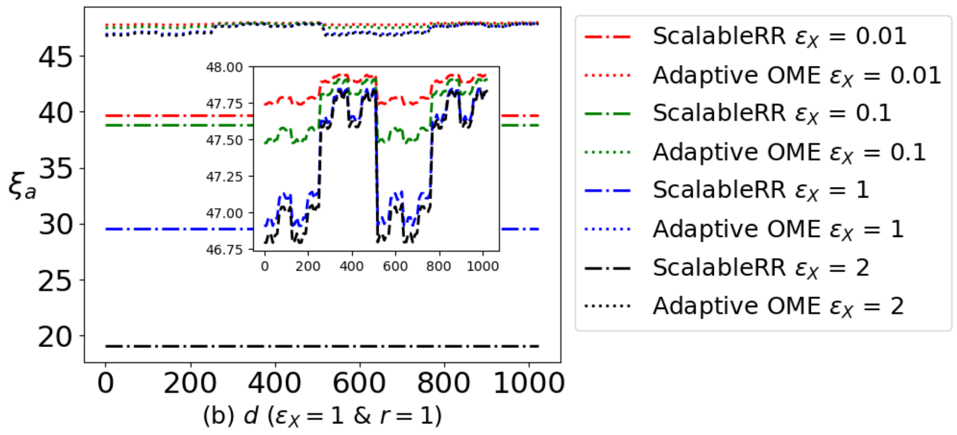
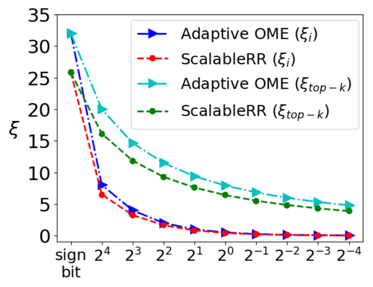
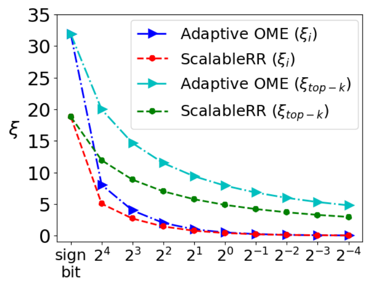


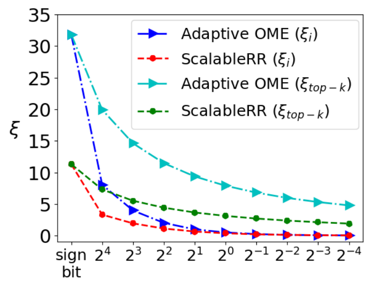
Figure A.10 Expected error comparison between ScalableRR and OME.



((a)) $\epsilon_X = 0.1$

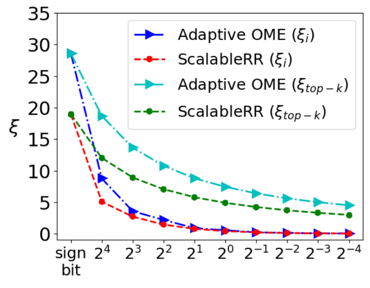


((b)) $\epsilon_X = 1.0$

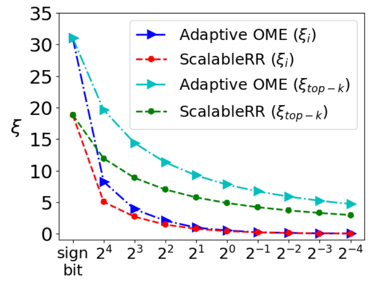


((c)) $\epsilon_X = 2.0$

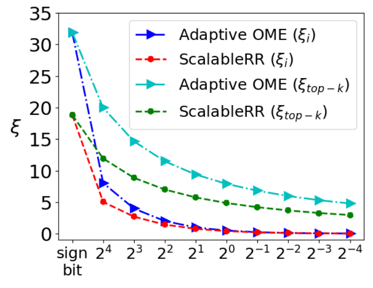
Figure A.11 Expected error at the bit-level as a function of ϵ_X with $r = 10,000$.



((a)) $d = 5$

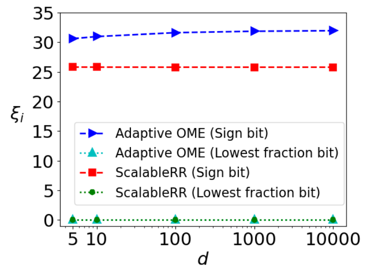


((b)) $d = 100$

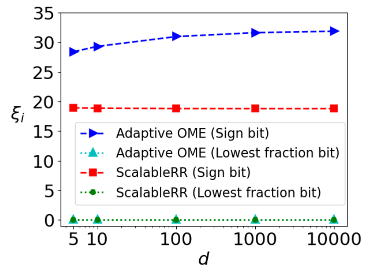


((c)) $d = 10,000$

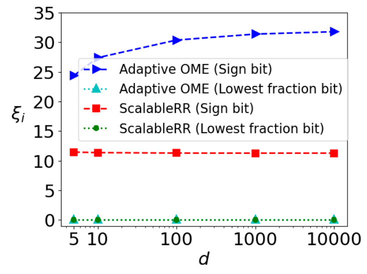
Figure A.12 Expected error at the bit-level as a function of d with $\epsilon_X = 1.0$.



((a)) $\epsilon_X = 0.1$

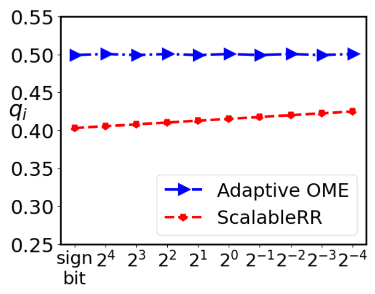


((b)) $\epsilon_X = 1.0$

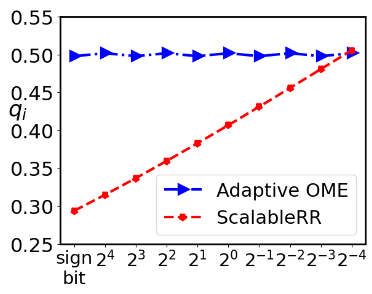


((c)) $\epsilon_X = 2.0$

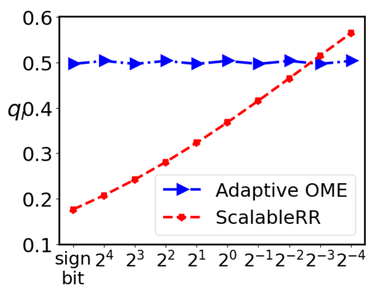
Figure A.13 Expected error at the bit-level (e.g., the sign bit (highest important bit) and the lowest fraction bit (lowest important bit) as a function of d with $r = 1$.



((a)) $\epsilon_X = 0.1$



((b)) $\epsilon_X = 1.0$



((c)) $\epsilon_X = 2.0$

Figure A.14 Randomization probability q_i given $r = 10,000$.

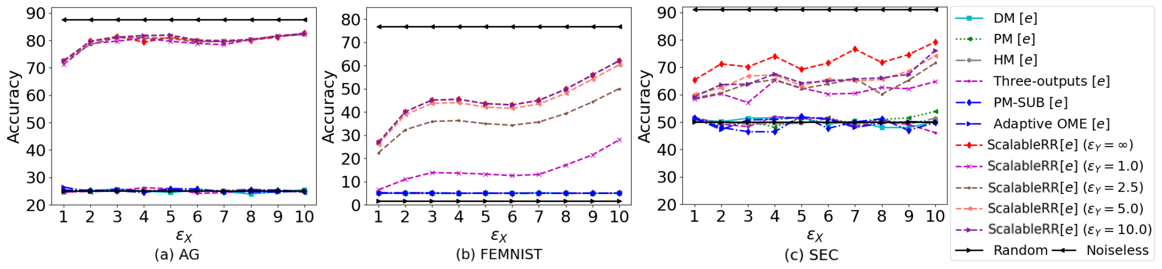


Figure A.15 Accuracy of LDP algorithms applied on the embedding features e in the AG, SEC, and FEMNIST datasets.

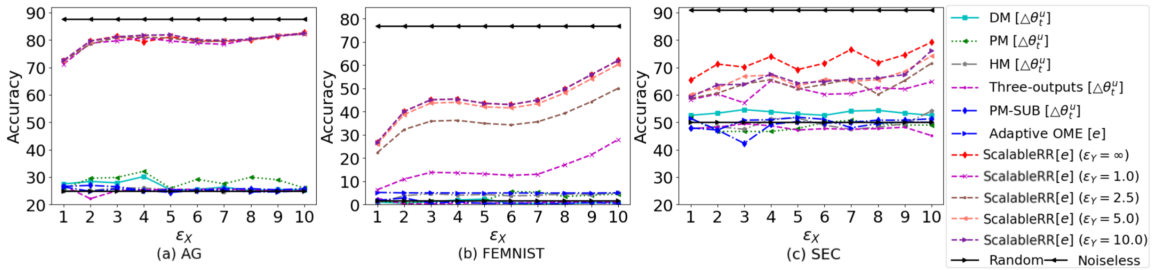


Figure A.16 Accuracy of LDP algorithms applied on the gradients $\triangle\theta_t^u$ in the AG, SEC, and FEMNIST datasets.

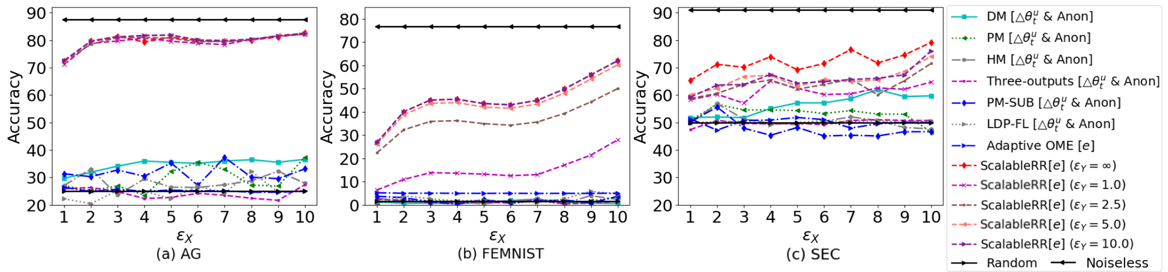


Figure A.17 Accuracy of LDP algorithms applied on the gradients $\triangle\theta_t^u$ with the anonymizer [191].

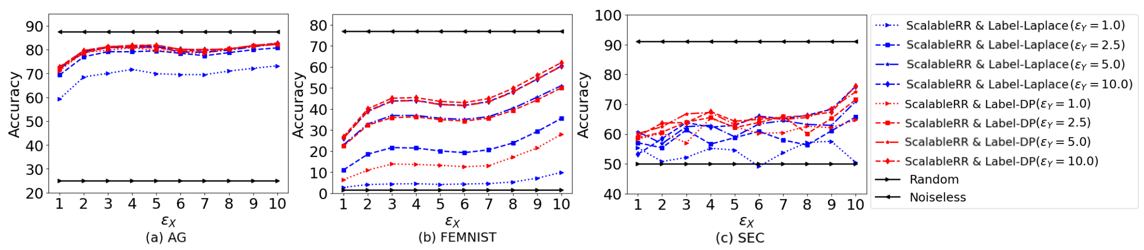


Figure A.18 Accuracy of each mechanism applied on labels.

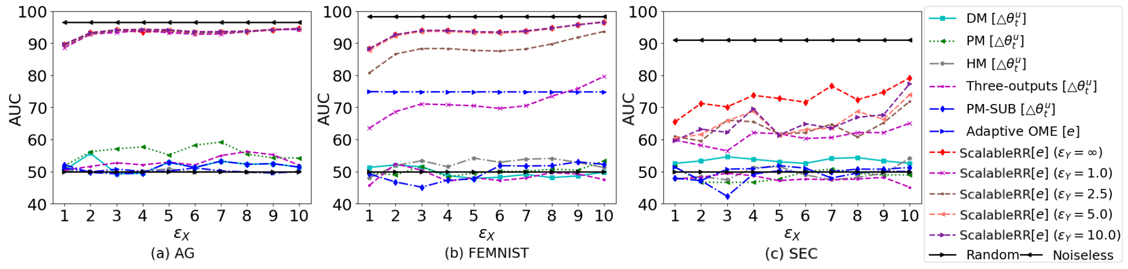


Figure A.19 AUC values of LDP algorithms applied on the gradients $\triangle\theta_t^u$ in the AG, SEC, and FEMNIST datasets.

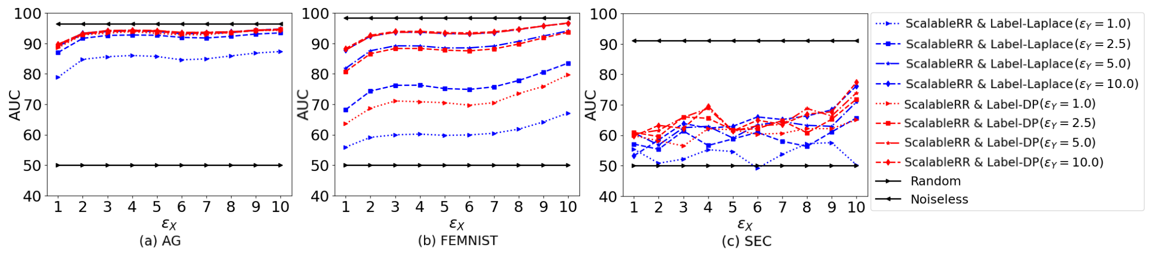


Figure A.20 AUC values of each mechanism applied on labels in the AG, SEC, and FEMNIST datasets.

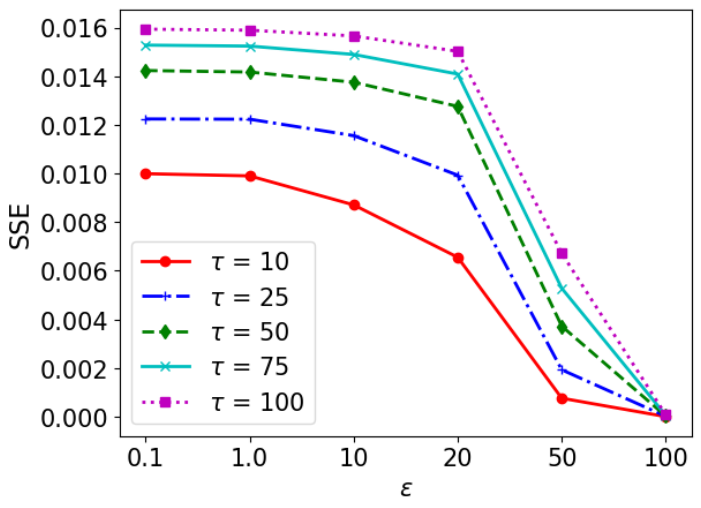
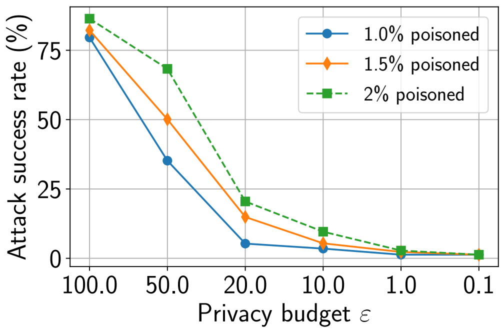
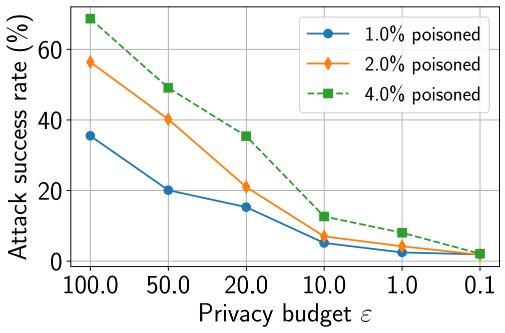


Figure A.21 SSE values as a function of ϵ and τ



((**a**)) Contagio



((**b**)) Drebin

Figure A.22 Attack success rate as a function of privacy budget ε and the portion of poisoned samples on the Contagio PDF and Drebin datasets.

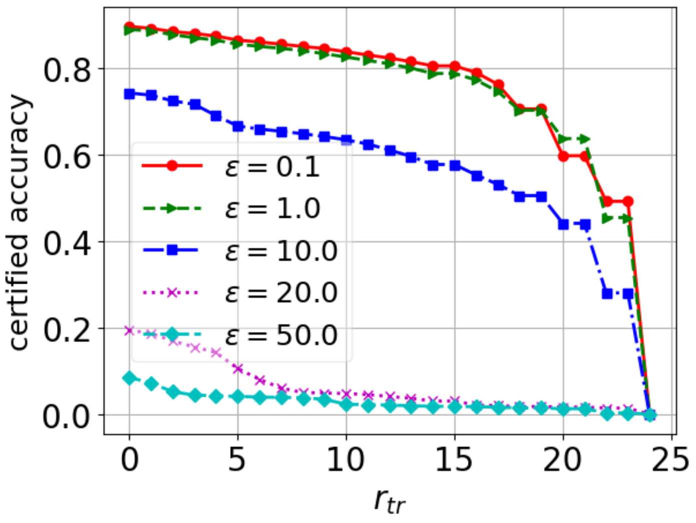
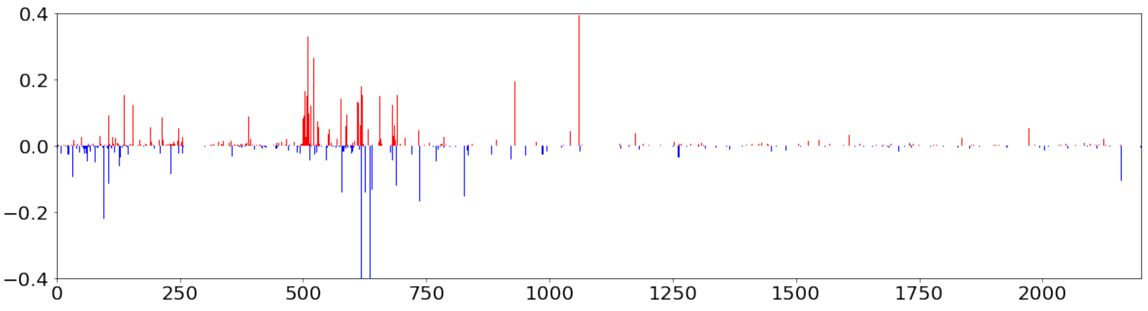


Figure A.23 Certified robustness at the training time. The smaller privacy budget ε, the higher certified accuracy.


((**a**)) SHAP explanation (without XRAND)

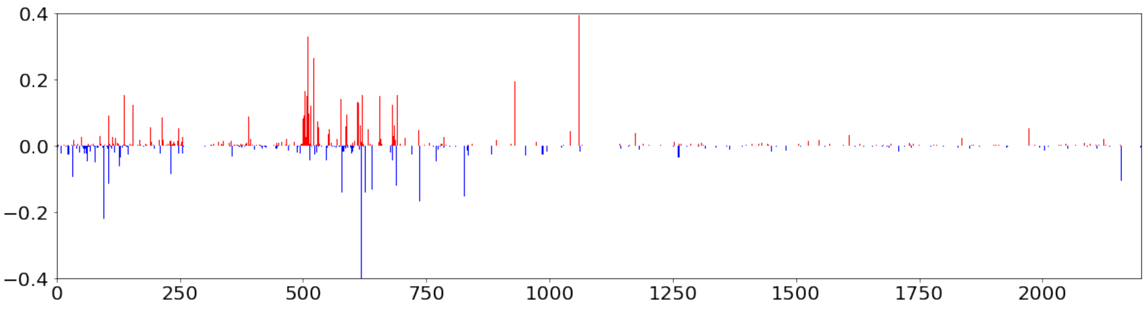
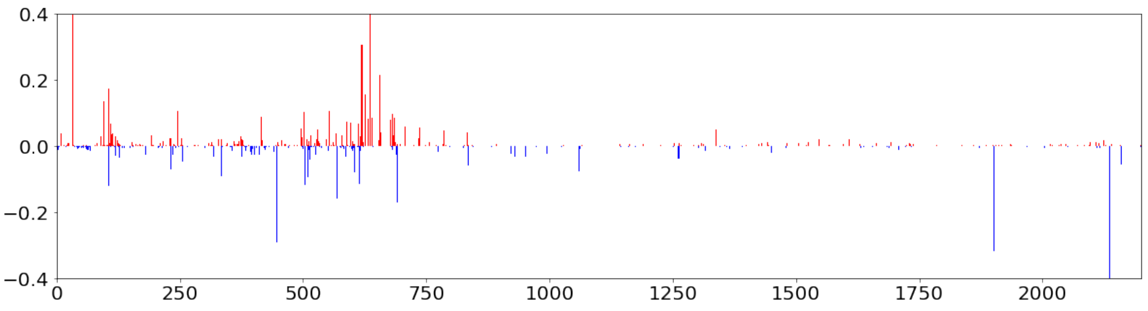

((**b**)) XRAND explanation

Figure A.24 Visualizing the SHAP explanation and our XRAND explanation of test sample 1.


((**a**)) SHAP explanation (without XRAND)

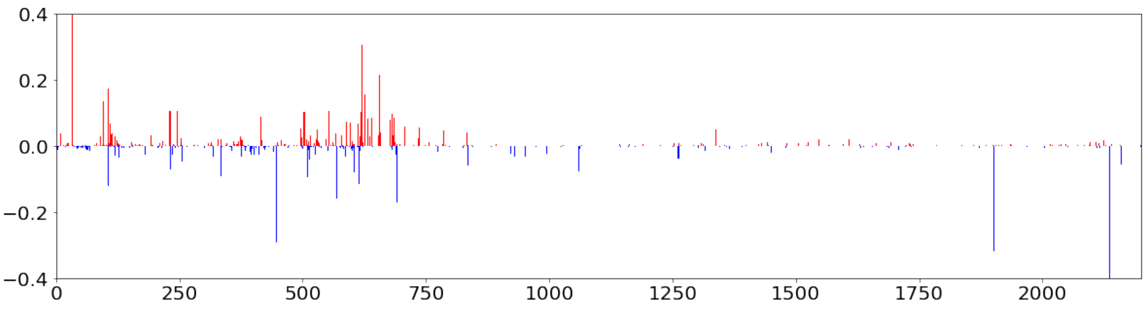
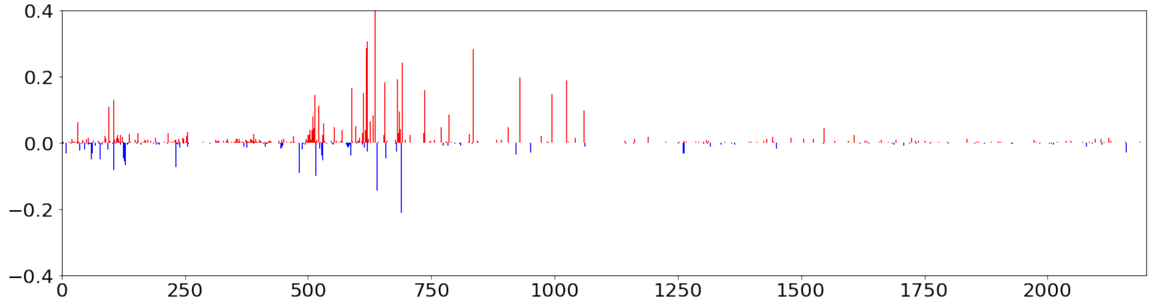
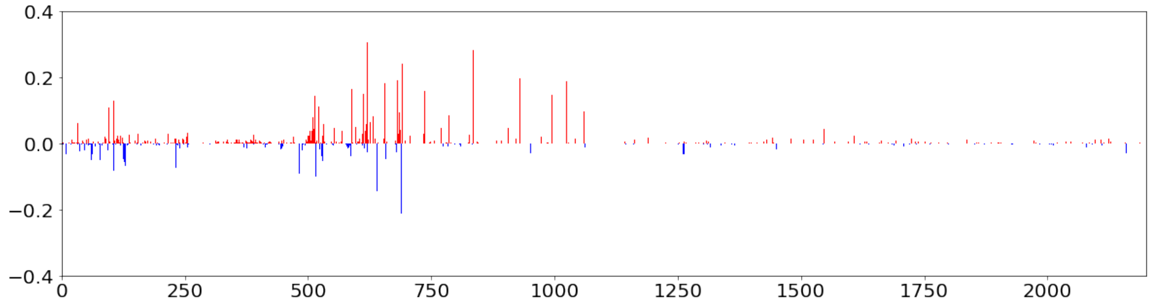

((**b**)) XRAND explanation

Figure A.25 Visualizing the SHAP explanation and XRAND explanation of test sample 2.

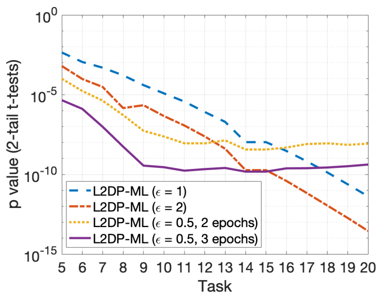


((**a**)) SHAP explanation (without XRAND)

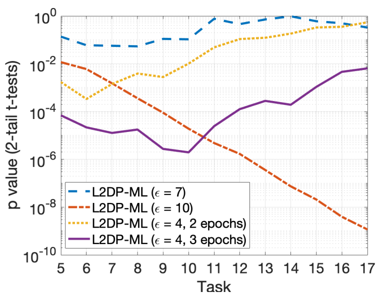


((**b**)) XRAND explanation

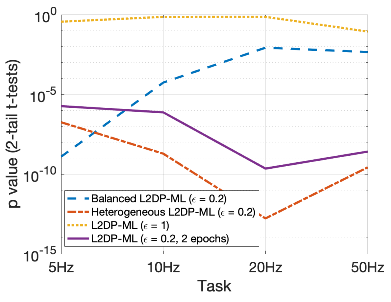
Figure A.26 Visualizing the SHAP explanation and XRAND explanation of test sample 3.



((**a**))



((**b**))

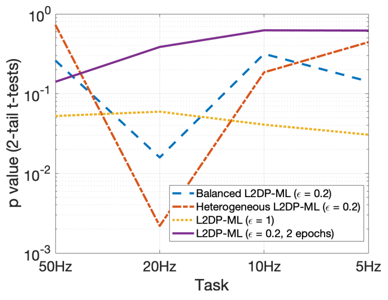


((**c**))

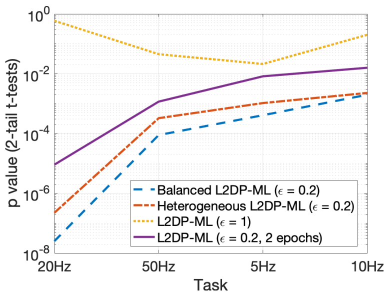
Figure A.27 p value for 2-tail t-tests on the (a) Permuted MNIST (20 tasks), b) Permuted CIFAR-10 (17 tasks), and (c) HARW (5Hz - 10Hz - 20Hz - 50Hz).

Algorithm 7 L2DP-ML with Streaming Batch Training

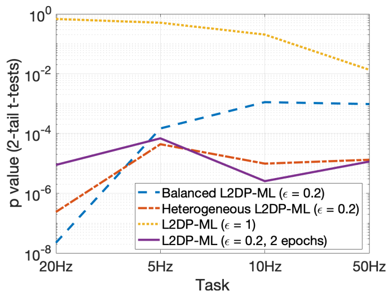
```
Input: T=$\{t_i\}_{i\in[1,m]}$, $\{D_i\}_{i\in[1,m]}$, batch size $\lambda$, privacy budgets: $\epsilon_1$ and $\epsilon_2$, learning rate $\varrho$
Output: $(\epsilon_1 + \epsilon_1/\gamma_{\mathbf{x}} + \epsilon_1/\gamma + \epsilon_2)$-Lifelong DP parameters $\{\theta^i\}_{i\in[1,m]} = \{\theta^i_1, \theta^i_2\}_{i\in[1,m]}$
1:  Draw Noise $\chi_1 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})]^d$, $\chi_2 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{R}}}}{\epsilon_1})]^\beta$, $\chi_3 \leftarrow [Lap(\frac{\Delta_{\widetilde{\mathcal{L}}}}{\epsilon_2})]^{|\mathbf{h}_\pi|}$
2:  Randomly Initialize $\theta = \{\theta_1, \theta_2\}$, $\mathbb{M}_1 = \emptyset$, $\forall\tau \in \mathbf{T} : \overline{D}_\tau = \{\overline{x}_r \leftarrow x_r + \frac{\chi_1}{\lambda}\}_{x_r\in D_\tau}$,
    hidden layers $\{\mathbf{h}_1 + \frac{2\chi_2}{\lambda}, \dots, \mathbf{h}_\pi\}$, where $\mathbf{h}_\pi$ is the last hidden layer
3:  for $\tau \in \mathbf{T}$ do
4:    $\mathbf{B} = \{B_1, \dots, B_n\}$ s.t. $\forall B \in \mathbf{B} : B$ is a random batch with the size $s$, $B_1 \cap \dots \cap B_n =$
      $\emptyset$, and $B_1 \cup \dots \cup B_n = \overline{D}_\tau$
5:    for $B \in \mathbf{B}$ do
6:      if $\tau == 0$ then
7:        Compute Gradients:
8:        $g \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_B(\theta_1^{\tau-1}), \nabla_{\theta_2}\overline{\mathcal{L}}_B(\theta_2^{\tau-1})\}$ with the noise $\frac{\chi_3}{\lambda}$
9:        Descent: $\{\theta_1^\tau, \theta_2^\tau\} \leftarrow \{\theta_1^{\tau-1}, \theta_2^{\tau-1}\} - \varrho g$
10:     else
11:       Select a batch $B_e$ randomly from a set of batches in episodic memory $\mathbb{M}_\tau$
12:       Compute Gradients:
13:       $g \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_B(\theta_1^{\tau-1}), \nabla_{\theta_2}\overline{\mathcal{L}}_B(\theta_2^{\tau-1})\}$ with the noise $\frac{\chi_3}{\lambda}$
14:       $g_{ref} \leftarrow \{\nabla_{\theta_1}\overline{\mathcal{R}}_{B_e}(\theta_1^{\tau-1}), \nabla_{\theta_2}\overline{\mathcal{L}}_{B_e}(\theta_2^{\tau-1})\}$ with the noise $\frac{\chi_3}{\lambda}$
15:       $\tilde{g} \leftarrow g - \frac{g^\top g_{ref}}{g_{ref}^\top g_{ref}} g_{ref}$
16:       Descent: $\{\theta_1^\tau, \theta_2^\tau\} \leftarrow \{\theta_1^{\tau-1}, \theta_2^{\tau-1}\} - \varrho\tilde{g}$
17:     end if
18:   end for
19:   Randomly Select a batch $B \in \mathbf{B}$
20:   $\mathbb{M}_\tau \leftarrow \mathbb{M}_{\tau-1} \cup B$
21: end for
```



((a))

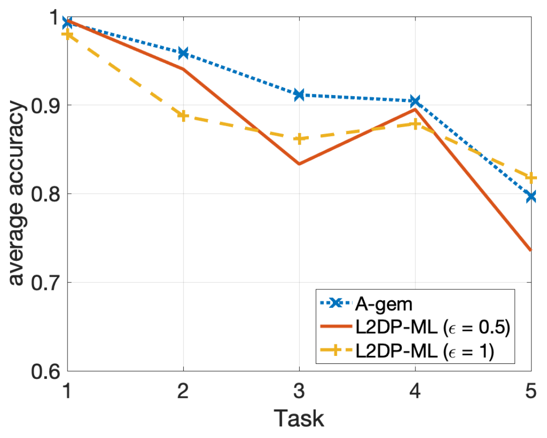


((b))

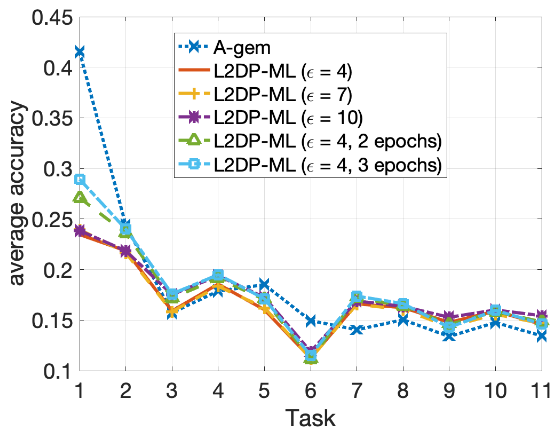


((c))

Figure A.28 p value for 2-tail t-tests on the HARW dataset with random task orders: (a) HARW 50Hz - 20Hz - 10Hz - 5Hz, (b) HARW 20Hz - 50Hz - 5Hz - 10Hz, and (c) HARW 20Hz - 5Hz - 10Hz - 50Hz.



((a))



((b))

Figure A.29 Average accuracy in the (a) Split MNIST (5 tasks), and b) Split CIFAR-10 and CIFAR-100 (11 tasks).

REFERENCES

[1] Cometomyhead academic news search engine (Retrieved on 01/01/2020). `http://newsengine.di.unipi.it`.

[2] Hoeffding's inequality (Retrieved on 01/01/2021). `https://web.stanford.edu/class/cs229t/2017/Lectures/concentration-slides.pdf`.

[3] Induction proofs (retrieved on 01/01/2020). `https://www.purplemath.com/modules/inductn3.htm`.

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

[5] M. Abadi, U. Erlingsson, I. Goodfellow, H.B. McMahan, I. Mironov, N. Papernot, K. Talwar, and L. Zhang. On the protection of private information in machine learning systems: Two recent approches. In *IEEE Computer Security Foundations Symposium (CSF)*, pages 1–6, 2017.

[6] D. Abati, T. Tomczak, J.and Blankevoort, S. Calderara, R. Cucchiara, and B.E. Bejnordi. Conditional channel gated networks for task-aware continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3931–3940, 2020.

[7] A. Acar, H. Aksu, A.S. Uluagac, and M. Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)*, 51(4):1–35, 2018.

[8] J. Acharya, Z. Sun, and H. Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1120–1129, 2019.

[9] A. Adhikari. Example and feature importance-based explanations for black-box machine learning models. 2018.

[10] M.G. Alaslani. Convolutional neural network based feature extraction for iris recognition. *International Journal of Computer Science & Information Technology (IJCSIT) Vol*, 10, 2018.

[11] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725, 2020.

[12] H.S. Anderson and P. Roth. Ember: an open dataset for training static pe malware machine learning models. *arXiv preprint arXiv:1804.04637*, 2018.

[13] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, and M. Atiquzzaman. Local differential privacy for deep learning. *IEEE Internet of Things Journal*, 7(7):5827–5842, 2019.

[14] L. Arras, F. Horn, G. Montavon, K. R. Müller, and W. Samek. "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8):e0181142, 2017.

[15] H. Asi, J. Duchi, and O. Javidbakht. Element level differential privacy: The right granularity of privacy. *arXiv preprint arXiv:1912.04042*, 2019.

[16] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[17] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[18] S. Barlas. Prescription drug abuse hits hospitals hard: Tighter federal steps aim to deflate crisis. *Pharmacy and Therapeutics*, 38(9):531, 2013.

[19] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. In *ACM Symposium on Theory of Computing (STOC)*, pages 127–135, 2015.

[20] R. Bassily, A. Smith, and A. Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 464–473, 2014.

[21] B. Bebensee. Local differential privacy: a tutorial. *arXiv preprint arXiv:1907.11908*, 2019.

[22] A. Ben-Israel. A newton-raphson method for the solution of systems of equations. *Journal of Mathematical analysis and applications*, 15(2):243–252, 1966.

[23] Gary B.H., Manu R., Tamara B., and Erik L.M. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[24] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.

[25] S. Bian, T. Wang, M. Hiromoto, Y. Shi, and T. Sato. Ensei: Efficient secure inference via frequency-domain homomorphic convolution for privacy-preserving visual recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9403–9412, 2020.

[26] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint:1812.01097*, 2018.

[27] H. Cao, S. Liu, R. Zhao, and X. Xiong. Ifed: A novel federated learning framework for local differential privacy in power internet of things. *International Journal of Distributed Sensor Networks*, 16(5):1550147720919698, 2020.

[28] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, et al. Extracting training data from large language models. In *USENIX Security Symposium*, volume 6, pages 2633–2650, 2021.

[29] A. Chaudhry, P.K. Dokania, T. Ajanthan, and P.H.S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[30] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with a-gem. *International Conference on Learning Representations (ICLR)*, 2019.

[31] F. Chen, Y.C. Wang, B. Wang, and C.C.J Kuo. Graph representation learning: a survey. *Asia-Pacific Signal and Information Processing Association (APSIPA) Transactions on Signal and Information Processing*, 9, 2020.

[32] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev. Distributed differential privacy via shuffling. In *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 375–403, 2019.

[33] E. Choi, A. Schuetz, W.F. Stewart, and J. Sun. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association*, 24(2):361–370, 2017.

[34] H. Choi and S. Park. A survey of machine learning-based system performance optimization techniques. *Applied Sciences*, 11(7):3235, 2021.

[35] C.J. Clopper and E.S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934.

[36] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. Emnist: Extending mnist to handwritten letters. In *International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926, 2017.

[37] J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, pages 1310–1320, 2019.

[38] R. Confalonieri, F. M. delPrado, S. Agramunt, D. Malagarriga, D. Faggion, T. Weyde, and T. R. Besold. An ontology-based approach to explaining artificial neural networks. *arXiv preprint arXiv:1906.08362*, 2019.

[39] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang. Privacy at scale: Local differential privacy in practice. In *International Conference on Management of Data*, pages 1655–1658, 2018.

[40] G. Cormode, T. Kulkarni, and D. Srivastava. Marginal release under local differential privacy. In *International Conference on Management of Data*, pages 131–146, 2018.

[41] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[42] Steven D. and Stephen B. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[43] D. Danon, M. Arar, D. Cohen-Or, and A. Shamir. Image resizing by reconstruction from deep features. *Computational Visual Media*, 7(4), 2021.

[44] K. Das and R.N. Behera. A survey on machine learning: concept, algorithms and applications. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(2):1301–1309, 2017.

[45] L. Derczynski, E. Nichols, M. van Erp, and N. Limsopatham. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Workshop on Noisy User-generated Text (WNUT)*, pages 140–147, 2017.

[46] F. Dernoncourt, J.Y. Lee, O. Uzuner, and P. Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606, 2017.

[47] P. Desai, P. Lai, N.H. Phan, and M.T. Thai. Continual learning with differential privacy. In *International Conference on Neural Information Processing (ICONIP)*, pages 334–343, 2021.

[48] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[49] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4829–4837, 2016.

[50] J. Duchi and R.n Rogers. Lower bounds for locally private estimation via communication complexity. In *Conference on Learning Theory*, pages 1161–1191, 2019.

[51] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *IEEE Protocols for secure computations*, pages 429–438, 2013.

[52] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.

[53] A. Durrant, M. Markovic, D. Matthews, D. May, J. Enright, and G. Leontidis. The role of cross-silo federated learning in facilitating data sharing in the agri-food sector. *arXiv preprint arXiv:2104.07468*, 2021.

[54] J. Dutka. The early history of the factorial function. *Archive for history of exact sciences*, pages 225–249, 1991.

[55] C. Dwork. Differential privacy: A survey of results. In *Theory and Applications of Models of Computation*, pages 1–19, 2008.

[56] C. Dwork and J. Lei. Differential privacy and robust statistics. In *ACM Symposium on Theory of Computing*, pages 371–380, 2009.

[57] C. Dwork and J. Lei. Differential privacy and robust statistics. In *ACM Symposium on Theory of Computing*, pages 371–380, 2009.

[58] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284, 2006.

[59] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[60] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *FnT-TCS*, 9:211–407, 2014.

[61] S. Ebrahimi, F. Meier, R. Calandra, T. Darrell, and M. Rohrbach. Adversarial continual learning. In *European Conference on Computer Vision (ECCV)*, pages 386–402, 2020.

[62] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 2468–2479, 2019.

[63] U. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067, 2014.

[64] O. Etzioni, M. Banko, S. Soderland, and D.S. Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, 2008.

[65] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545, 2011.

[66] L. Fan. Image pixelization with differential privacy. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 148–162, 2018.

[67] S. Farquhar and Y. Gal. Differentially private continual learning. *Privacy in Machine Learning and AI workshop at ICML*, 2018.

[68] E. W. Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[69] L. Freddy and W. Jiewen. Semantic explanations of predictions. arXiv:1805.10587v1, 2018.

[70] M. Fredrikson, S. Jha, and T. Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[71] S. Gao, M.T. Young, J.X. Qiu, H.J. Yoon, J.B. Christian, P.A. Fearn, G.D. Tourassi, and A. Ramanthan. Hierarchical attention networks for information extraction from cancer pathology reports. *Journal of the American Medical Informatics Association*, 25(3):321–330, 2018.

[72] C. Gentry et al. *A fully homomorphic encryption scheme*, volume 20. 2009.

[73] B. Ghazi, N. Golowich, R. Kumar, P. Manurangsi, and C. Zhang. On deep learning with label differential privacy. *arXiv preprint arXiv:2102.06062*, 2021.

[74] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning (ICML)*, pages 201–210, 2016.

[75] O. Goldreich. Secure multi-party computation. *Manuscript. Preliminary version*, 1998.

[76] S. Gopi, Y.T. Lee, and L. Wutschitz. Numerical composition of differential privacy. *Conference on Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[77] S. Grollmisch, E. Cano, C. Kehling, and M. Taenzer. Analyzing the potential of pre-trained embeddings for audio classification tasks. In *European Signal Processing Conference (EUSIPCO)*, 2021.

[78] A. Haeberlen, B. C. Pierce, and A. Narayan. Differential privacy under fire. In *USENIX Security Symposium*, volume 33, 2011.

[79] J. Hamm, Y. Cao, and M. Belkin. Learning privately from multiparty data. In *International Conference on Machine Learning (ICML)*, pages 555–563, 2016.

[80] J. He and F. Zhu. Online continual learning via candidates voting. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3154–3163, 2022.

[81] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[82] M. A. Heikkilä, A. Koskela, K. Shimizu, S. Kaski, and A. Honkela. Differentially private cross-silo federated learning. *arXiv preprint arXiv:2007.05553*, 2020.

[83] M. Helmstaedter, K.L. Briggman, S.C. Turaga, V. Jain, H.S. Seung, and W. Denk. Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500(7461):168–174, 2013.

[84] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[85] M. Honnibal and I. Montani. Spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1):411–420, 2017.

[86] M.Z. Horváth, M.N. Mueller, M. Fischer, and M. Vechev. Boosting randomized smoothing with variance reduced classifiers. In *International Conference on Learning Representations (ICLR)*, 2022.

[87] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *ACL*, page 328–339, 2018.

[88] H. Hu, N.H. Phan, J. Geller, S. Iezzi, H.T. Vo, D. Dou, and S.A. Chun. An ensemble deep learning model for drug abuse detection in sparse twitter-sphere. In *MedInfo*, pages 163–167, 2019.

[89] X. Huang, Y. Ding, Z. L. Jiang, S. Qi, X. Wang, and Q. Liao. DP-FL: a novel differentially private federated learning framework for the unbalanced data. *World Wide Web*, 23(4):2529–2545, 2020.

[90] A. Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922, 2018.

[91] C. Ivan. Convolutional neural networks on randomized data. In *CVPR Workshops*, pages 1–8, 2019.

[92] M. James, W. Sarah, D. Jon, S. Jimeng, and E. Jacob. Explainable prediction of medical codes from clinical text. *CoRR*, abs/1802.05695, 2018.

[93] J. Jia, X. Cao, and N. Z. Gong. Intrinsic certified robustness of bagging against data poisoning attacks. *AAAI Conference on Artificial Intelligence*, 35(9):7961–7969, 2021.

[94] Y. Jia, J. Bailey, K. Ramamohanarao, C. Leckie, and M.E. Houle. Improving the quality of explanations with local embedding perturbations. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 875–884, 2019.

[95] X. Jiang, H. Hu, T. On, P. Lai, V.D. Mayyuri, A. Chen, D.M. Shila, A. Larmuseau, R. Jin, and C. Borcea. Flsys: Toward an open ecosystem for federated learning mobile apps. *IEEE Transactions on Mobile Computing*, 2022.

[96] P. Kairouz, K. Bonawitz, and D. Ramage. Discrete distribution estimation under local privacy. In *International Conference on Machine Learning (ICML)*, pages 2436–2444, 2016.

[97] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 2021.

[98] B. Kieffer, M. Babaie, S. Kalra, and H.R. Tizhoosh. Convolutional neural networks for histopathology image classification: Training vs. using pre-trained networks. In *IPTA*, pages 1–6, 2017.

[99] H. Kim, J. Park, M. Bennis, and S. L. Kim. Blockchained on-device federated learning. *IEEE Communications Letters*, 24(6):1279–1283, 2019.

[100] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[101] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A.A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *National Academy of Sciences*, 114(13):3521–3526, 2017.

[102] H. Knublauch, R.W. Fergerson, N.F. Noy, and M.A. Musen. The protégé owl plugin: An open development environment for semantic web applications. In *International Semantic Web Conference (ISWC)*, pages 229–243, 2004.

[103] K. Kourou, T.P. Exarchos, K.P. Exarchos, M.V. Karamouzis, and D.I. Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17, 2015.

[104] R. Kumar, Z. Xiaosong, R.U. Khan, J. Kumar, and I. Ahad. Effective and explainable detection of android malware based on machine learning algorithms. In *International Conference on Computing and Artificial Intelligence (ICCAI)*, pages 35–40, 2018.

[105] S. Laghmati, A. Tmiri, and B. Cherradi. Machine learning based system for prediction of breast cancer severity. In *International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pages 1–5, 2019.

[106] P. Lai, N.H. Phan, H. Hu, A. Badeti, D. Newman, and D. Dou. Ontology-based interpretable machine learning for textual data. *International Joint Conference on Neural Networks (IJCNN)*, 2020.

[107] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[108] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, pages 656–672, 2019.

[109] M. Lécuyer, R. Spahn, K. Vodrahalli, R. Geambasu, and D. Hsu. Privacy accounting and quality control in the sage differentially private ml platform. In *ACM Symposium on Operating Systems Principles*, pages 181–195, 2019.

[110] J. Lee and D. Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1656–1665, 2018.

[111] L. Li, Y. Fan, M. Tse, and K.Y. Lin. A review of applications in federated learning. *Computers & Industrial Engineering*, 149:106854, 2020.

[112] N. Li and Q. Ye. Mobile data collection and analysis with local differential privacy. In *IEEE International Conference on Mobile Data Management (MDM)*, pages 4–7, 2019.

[113] S. Lipovetsky and M. Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17(4):319–330, 2001.

[114] H. Liu, F. Sun, and B. Fang. Lifelong learning for heterogeneous multi-modal tasks. In *International Conference on Robotics and Automation (ICRA)*, pages 6158–6164, 2019.

[115] R. Liu, Y. Cao, H. Chen, R. Guo, and M. Yoshikawa. Flame: Differentially private federated learning in the shuffle model. In *Conference on Artificial Intelligence (AAAI)*, pages 8688–8696, 2021.

[116] R. Liu, Y. Cao, M. Yoshikawa, and H. Chen. Fedsel: Federated sgd under local differential privacy with top-k dimension selection. In *International Conference on Database Systems for Advanced Applications*, pages 485–501, 2020.

[117] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

[118] D. Lopez-Paz and M.A. Ranzato. Gradient episodic memory for continual learning. *Conference on Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[119] S.M. Lundberg and S.I. Lee. A unified approach to interpreting model predictions. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 4768–4777, 2017.

[120] L. Lyu, X. He, and Y. Li. Differentially private representation for nlp: Formal guarantee and an empirical study on privacy and fairness. *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[121] L. Lyu, Y. Li, X. He, and T. Xiao. Towards differentially private text representations. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1813–1816, 2020.

[122] D. Martens, B. Baesens, T. Van G., and J. Vanthienen. Comprehensible credit scoring models using rule extraction from support vector machines. *European journal of Operational Research (EJOR)*, 183(3):1466–1476, 2007.

[123] D. Martens and F. Provost. Explaining data-driven document classifications. 2013.

[124] A. Martins and R. Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning (ICML)*, pages 1614–1623, 2016.

[125] B. Maschler, T.T.H. Pham, and M. Weyrich. Regularization-based continual learning for anomaly detection in discrete manufacturing. *Procedia CIRP*, 104:452–457, 2021.

[126] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[127] H.B. McMahan, E. Moore, D. Ramage, and B.A. y Arcas. Federated learning of deep networks using model averaging. *arXiv:1602.05629*, 2016.

[128] H.B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. *International Conference on Learning Representations (ICLR)*, 2017.

[129] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706, 2019.

[130] S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing LSTM language models. *International Conference on Learning Representations*, 2018.

[131] T. Mikolov, K. Chen, G. S. Corrado, and J. A. Dean. Computing numeric representations of words in a high-dimensional space, 2015. US Patent 9,037,464.

[132] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Černockỳ. Empirical evaluation and combination of advanced language modeling techniques. In *International Speech Communication Association*, 2011.

[133] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur. Extensions of recurrent neural network language model. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5528–5531, 2011.

[134] S. Milli, L. Schmidt, A.D. Dragan, and M. Hardt. Model reconstruction from model explanations. In *Conference on Fairness, Accountability, and Transparency (FAccT)*, pages 1–9, 2019.

[135] R. Miotto, L. Li, B.A. Kidd, and J.T. Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6(1):1–10, 2016.

[136] I. Mironov. On significance of the least significant bits for differential privacy. In *ACM Conference on Computer and Communications Security (CCS)*, pages 650–661, 2012.

[137] I. Mironov. Rényi differential privacy. In *Computer Security Foundations Symposium*, pages 263–275, 2017.

[138] T. Miura, S. Hasegawa, and T. Shibahara. Megex: Data-free model extraction attack against gradient-based explainable ai. *arXiv preprint arXiv:2107.08909*, 2021.

[139] S. Nagrecha, J. Z. Dillon, and N. V. Chawla. Mooc dropout prediction: lessons learned from making pipelines interpretable. In *International Conference on World Wide Web Companion*, pages 351–359, 2017.

[140] M. Nasr, S. Song, A. Thakurta, N. Papernot, and N. Carlini. Adversary instantiation: Lower bounds for differentially private machine learning. *arXiv preprint arXiv:2101.04535*, 2021.

[141] J. Neyman and E.S. Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

[142] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin. Collecting and analyzing data from smart device users with local differential privacy. *arXiv preprint:1606.05053*, 2016.

[143] H. Oh and Y. Lee. Exploring image reconstruction attack in deep learning computation offloading. In *International Workshop on Deep Learning for Mobile Systems and Applications*, 2019.

[144] O. Ostapenko, M. Puscas, T. Klein, P. Jahnichen, and M. Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, pages 11321–11329, 2019.

[145] X. Pan, M. Zhang, S. Ji, and M. Yang. Privacy risks of general-purpose language models. In *IEEE SP*, pages 1314–1331, 2020.

[146] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.

[147] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson. Scalable private learning with pate. *International Conference on Learning Representations (ICLR)*, 2018.

[148] J. Pennington, R. Socher, and C.D. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[149] H. Phan, M.T. Thai, H. Hu, R. Jin, T. Sun, and D. Dou. Scalable differential privacy with certified robustness in adversarial learning. In *International Conference on Machine Learning (ICML)*, pages 7683–7694, 2020.

[150] N.H. Phan, T. My, M.S. Devu, and R. Jin. Differentially private lifelong learning. In *Privacy in Machine Learning (PriML), NeurIPS'19 Workshop*, 2019.

[151] N.H. Phan, M.T. Thai, H. Hu, R. Jin, T. Sun, and D. Dou. Scalable differential privacy with certified robustness in adversarial learning. In *International Conference on Machine Learning (ICML)*, pages 7683–7694, 2020.

[152] N.H. Phan, M. Vu, Y. Liu, R. Jin, D. Dou, X. Wu, and M.T. Thai. Heterogeneous gaussian mechanism: Preserving differential privacy in deep learning with provable robustness. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

[153] N.H. Phan, Y. Wang, X. Wu, and D. Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In *Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI)*, volume 16, pages 1309–1316, 2016.

[154] N.H. Phan, X. Wu, and D. Dou. Preserving differential privacy in convolutional deep belief networks. *Machine learning*, 106(9):1681–1704, 2017.

[155] N.H. Phan, X. Wu, H. Hu, and D. Dou. Adaptive laplace mechanism: Differential privacy preservation in deep learning. In *IEEE International Conference on Data Mining (ICDM)*, pages 385–394, 2017.

[156] S.M. Plis, D.R. Hjelm, R. Salakhutdinov, E.A. Allen, H.J. Bockholt, J.D. Long, H.J. Johnson, J.S. Paulsen, J.A. Turner, and V.D. Calhoun. Deep learning for neuroimaging: a validation study. *Frontiers in neuroscience*, 8:229, 2014.

[157] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C.D. Manning. Stanza: A python natural language processing toolkit for many human languages. *ACL System Demonstration*, 2020.

[158] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016(1):1–16, 2016.

[159] H. Qu, H. Rahmani, L. Xu, B. Williams, and J. Liu. Recent advances of continual learning in computer vision: An overview. *arXiv preprint arXiv:2109.11369*, 2021.

[160] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[161] J. Rajasegaran, S. Khan, M. Hayat, F.S. Khan, and M. Shah. itaml: An incremental task-agnostic meta-learning approach. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, pages 13588–13597, 2020.

[162] S. Ramaswamy, O. Thakkar, R. Mathews, G. Andrew, H.B. McMahan, and F. Beaufays. Training production language models without memorizing user data. *arXiv*, 2020.

[163] J. Ramos et al. Using TF-IDF to determine word relevance in document queries. In *iCML*, volume 242, pages 133–142, 2003.

[164] L. Rasmy, Y. Xiang, Z. Xie, C. Tao, and D. Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):1–13, 2021.

[165] S.A. Rebuffi, A. Kolesnikov, G. Sperl, and C.H. Lampert. icarl: Incremental classifier and representation learning. In *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, pages 2001–2010, 2017.

[166] M.T. Ribeiro, S. Singh, and C. Guestrin. ”Why should i trust you?” explaining the predictions of any classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

[167] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *International Conference on Learning Representations (ICLR)*, 2019.

[168] R. L. Rivest, L. Adleman, M. L. Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[169] M. Roman, A. Shahid, S. Khan, A. Koubaa, and L. Yu. Citation intent classification using word embedding. *IEEE Access*, pages 9982–9995, 2021.

[170] A. Roth. Buying private data at auction: the sensitive surveyor's problem. *ACM SIGecom Exchanges*, 11(1):1–8, 2012.

[171] M. Roumia and S. Steinhubl. Improving cardiovascular outcomes using electronic health records. *Current cardiology reports*, 16(2):1–6, 2014.

[172] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *Network and Distributed System Security*, 2019.

[173] E.F. Sang and F. De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoNLL*, 2003.

[174] A. D. Sarwate and L. Sankar. A rate-disortion perspective on local differential privacy. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 903–908, 2014.

[175] M. Schmitz, R. Bart, S. Soderland, O. Etzioni, et al. Open language learning for information extraction. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, 2012.

[176] M. Seif, R. Tandon, and M. Li. Wireless federated learning with local differential privacy. In *IEEE International Symposium on Information Theory*, pages 2604–2609, 2020.

[177] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[178] G. Severi, J. Meyer, S. Coull, and A. Oprea. Explanation-guided backdoor poisoning attacks against malware classifiers. In *USENIX Security Symposium*, 2021.

[179] L.S. Shapley. *Notes on the n-Person Game — II: The Value of an n-Person Game*. Santa Monica, CA: RAND Corporation, 1951.

[180] H. Shin, J.K. Lee, J. Kim, and J. Kim. Continual learning with deep generative replay. *Conference on Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[181] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *ACM Conference on Computer and Communications Security (CCS)*, 2015.

[182] R. Shokri, M. Strobel, and Y. Zick. On the privacy risks of model explanations. In *AAAI/ACM Conference on AI, Ethics, and Society*, pages 231–241, 2021.

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

[184] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning (ICML)*, pages 3145–3153, 2017.

[185] C. Smutz and A. Stavrou. Malicious PDF detection using metadata and structural features. In *Annual Computer Security Applications Conference*, pages 239–248, 2012.

[186] S. Soderland, B. Roof, B. Qin, S. Xu, O. Etzioni, et al. Adapting open information extraction to domain-specific relations. *AI magazine*, 31(3):93–102, 2010.

[187] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[188] J. Stremmel and A. Singh. Pretraining federated text models for next word prediction. In *Future of Information and Communication Conference*, pages 477–488, 2021.

[189] E. Štrumbelj and I. Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

[190] L. Sun and L. Lyu. Federated model distillation with noise-free differential privacy. *arXiv:2009.05537*, 2020.

[191] L. Sun, J. Qian, and X. Chen. LDP-FL: Practical private aggregation in federated learning with local differential privacy. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[192] M. Sundararajan, A. Taly, and Q. Yan. Gradients of counterfactuals. *arXiv preprint arXiv:1611.02639*, 2016.

[193] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong. Few-shot class-incremental learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12183–12192, 2020.

[194] Gulli et al. Ag's corpus of news articles. `http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`, 2012.

[195] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou. A hybrid approach to privacy-preserving federated learning. In *ACM Workshop on Artificial Intelligence and Security*, 2019.

[196] S. Truex, L. Liu, K. H. Chow, M. E. Gursoy, and W. Wei. Ldp-fed: Federated learning with local differential privacy. In *ACM EdgeSys*, pages 61–66, 2020.

[197] M. H. ur Rehman, A. M. Dirir, K. Salah, E. Damiani, and D. S. Center. Trustfed: A framework for fair and trustworthy cross-device federated learning in iiot. *IEEE Transactions on Industrial Informatics*, 2021.

[198] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[199] J. Von Oswald, C. Henning, J. Sacramento, and B.F. Grewe. Continual learning with hypernetworks. *International Conference on Learning Representations (ICLR)*, 2020.

[200] S. Wagh, X. He, A. Machanavajjhala, and P. Mittal. Dp-cryptography: marrying differential privacy and cryptography in emerging applications. *Communications of the ACM*, 64(2):84–93, 2021.

[201] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu. Collecting and analyzing multidimensional data with local differential privacy. In *IEEE International Conference on Data Engineering (ICDE)*, pages 638–649, 2019.

[202] T. Wang, J. Blocki, N. Li, and S. Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security Symposium*, pages 729–745, 2017.

[203] Y. Wang, C. Si, and X. Wu. Regression model fitting under differential privacy and model inversion attack. In *International Joint Conference on Artificial Intelligence*, 2015.

[204] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[205] C. Wu, L. Herranz, X. Liu, J. van de Weijer, B. Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Conference on Neural Information Processing Systems (NeurIPS)*, 31, 2018.

[206] F. Wu and D. S. Weld. Open information extraction using wikipedia. In *Association for Computational Linguistics*, pages 118–127, 2010.

[207] J. Wu, J. Roy, and W.F. Stewart. Prediction modeling using ehr data: challenges, strategies, and a comparison of machine learning approaches. *Medical care*, pages S106–S113, 2010.

[208] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, and J. Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *ACM International Conference on Management of Data*, pages 1307–1322, 2017.

[209] X. Xiong, S. Liu, D. Li, Z. Cai, and X. Niu. A comprehensive survey on local differential privacy. *Security and Communication Networks*, 2020.

[210] C. Xu, J. Ren, L. She, Y. Zhang, Z. Qin, and K. Ren. Edgesanitizer: Locally differentially private deep inference at the edge for mobile data analytics. *IEEE Internet of Things Journal*, 6(3):5140–5151, 2019.

[211] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning (ICML)*, pages 2048–2057, 2015.

[212] Z. Yang and Z. Liang. Automated identification of sensitive data from implicit user specification. *Cybersecurity*, 1(1):1–15, 2018.

[213] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Conference of the North American chapter of the Association for Computational Linguistics: human language technologies*, pages 1480–1489, 2016.

[214] A. C. Yao. Protocols for secure computations. In *IEEE Symposium on Fundations of Computer Science*, pages 160–164, 1982.

[215] F. Ye and A.G. Bors. Learning latent representations across multiple data domains using lifelong vaegan. In *European Conference on Computer Vision (ECCV)*, pages 777–795, 2020.

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

[217] S.W. Yoon, D.Y. Kim, J. Seo, and J. Moon. Xtarnet: Learning to extract task-adaptive representation for incremental few-shot learning. In *International Conference on Machine Learning (ICML)*, pages 10852–10860, 2020.

[218] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang. A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems*, pages 58–69, 2020.

[219] L. Yu, L. Liu, C. Pu, M.E. Gursoy, and S. Truex. Differentially private model publishing for deep learning. In *IEEE Symposium on Security and Privacy (SP)*, pages 332–349, 2019.

[220] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu. Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning. In *USENIX Annual Technical Conference*, pages 493–506, 2020.

[221] C. Zhang, Z. Yang, X. He, and L. Deng. Multimodal intelligence: Representation learning, information fusion, and applications. *IEEE Journal of Selected Topics in Signal Processing*, pages 478–493, 2020.

[222] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett. Functional mechanism: regression analysis under differential privacy. *Very Large Data Bases (VLDB) Conference*, pages 21364—-1375, 2012.

[223] B. Zhao, K. R. Mopuri, and H. Bilen. *arXiv preprint arXiv:2001.02610*, 2020.

[224] P. Zhao, G. Zhang, S. Wan, G. Liu, and T. Umer. A survey of local differential privacy for securing internet of vehicles. *The Journal of Supercomputing*, pages 8391–8412, 2020.

[225] X. Zhao, W. Zhang, X. Xiao, and B. Lim. Exploiting explanations for model inversion attacks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 682–692, 2021.

[226] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K. Y. Lam. Local differential privacy based federated learning for internet of things. *IEEE Internet of Things Journal*, 2020.

[227] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. *Conference on Neural Information Processing Systems (NeurIPS)*, 32, 2019.