MACHINE UNLEARNING FOR MOBILITY DATA: AN ALGORITHMIC PERSPECTIVE

ALI FARAJI

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS OF SCIENCE

GRADUATE PROGRAM IN ELECTRICAL ENGINEERING & COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

JUNE 2025

Abstract

This work addresses machine unlearning for trajectory data, sequences of spatiotemporal points representing movement. Motivated by growing privacy concerns and regulations like GDPR and CCPA, which grant users the right to request deletion of their personal data from trained models (the *right to be forgotten*), we propose TraceHiding, an algorithmic framework that removes the influence of specific trajectories without full model retraining. TraceHiding estimates the data point importance and applies gradient updates to reverse it proportionally. The framework includes: (i) Estimating data point importance, (ii) a teacher-student architecture, and (iii) a loss function using Importance Scores to compute reversal gradients. We evaluate TraceHiding on benchmark trajectory classification datasets. Results show it outperforms strong baselines and state-of-the-art unlearning methods (Bad-T, SCRUB, NegGrad, and NegGrad+), effectively removing deleted trajectory influence, preserving retained data performance, and improving efficiency over retraining. To our knowledge, this is the first machine unlearning approach designed specifically for trajectory data.

Acknowledgements

First and foremost, I express my deepest gratitude to my supervisor, Dr. Manos Papagelis, whose expertise, understanding, and patience, added considerably to my graduate experience. His insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. His encouragement and genuine support gave me the strength and confidence to overcome the challenges during my studies. His guidance was invaluable in writing this thesis.

I would also like to extend my sincere thanks to the members of my examining committee, Prof. Ruth Urner and Prof. Gunho Sohn, for their time, effort, and thoughtful feedback. Their constructive suggestions helped refine this work and greatly contributed to its clarity, depth, and final form.

I am particularly grateful to my family. Their sacrifices, love, and belief in me have been the bedrock of my resilience and persistence. This journey has been not only an academic pursuit but a life-changing experience, and I am grateful to everyone who played a part in it.

Finally, I'm deeply grateful to my friends for their unwavering support, good vibes, and patience, even when my thesis made no sense to them; your encouragement meant the world. I truly couldn't have made it through the chaos without your reassurance and belief in me.

Table of Contents

List of Tables

List of Figures

List of Symbols

C	Set of all users
$\mathcal{T}_c$	A Trajectory that belongs to the user $c \in C$
l	Trajectory Length
$\mathcal{D}_t$	Training Dataset
$\mathcal{D}_u$	Unlearning/Forgetting Dataset
$\mathcal{D}_r$	Remaining Dataset
M_t	Originally Trained Model on$\mathcal{D}_t$
M_u	Unlearned Model
M_r	Trained Model on $\mathcal{D}_r$
W_t	Weights of M_t trained on $\mathcal{D}_t$
W_u	Weights of Unlearned Model M_u
W_r	Weights of M_r trained on $\mathcal{D}_r$
$\mathcal{A}(\cdot)$	Learning Process
$\mathcal{U}(\cdot)$	Unlearning Process
$\xi(x)$	Importance Score of trajectory x
$\mathcal{B}$	Set of All Tokens (Block) in the Dataset
$\mathcal{D}_t^c$	Set of all trajectories in $\mathcal{D}_t$ that belong to user c

Chapter 1

Introduction

As machine learning models increasingly rely on personal data, ensuring user privacy has become a critical concern. Trajectory data are particularly sensitive due to their spatiotemporal specificity. While valuable for applications like traffic prediction and location-based services, such data also raise significant privacy risks. Machine unlearning offers a solution by enabling the selective removal of learned data, aligning models with legal and ethical standards. This thesis explores unlearning in trajectory classification, addressing the unique challenges posed by sequential spatial data and proposing an algorithmic framework that balances privacy with model performance.

1.1 Motivation

As technology increasingly permeates every aspect of life in the digital age, the importance of privacy and data security cannot be overstated. Trajectory data, records of individuals' movements collected through GPS navigation, location-based services, and other geospatial technologies, play a critical role in numerous applications, from optimizing traffic flow to enhancing personalized marketing strategies. However, the ubiquitous collection and analysis

of such data also raise significant privacy concerns, spatiotemporal tags embed who-was-where-when directly into a model's weights [1], so ML models can unknowingly replay an individual's movements or location-bound copyrighted work in ways that breach local laws or personal consent. Machine unlearning lets us verifiably delete those tagged traces, aligning the model with differing regional privacy and copyright rules and sustaining public trust in location-aware AI. Against this backdrop, we investigate *machine unlearning* for trajectory data in a classification task, a mechanism that selectively forgets data points to address privacy demands and comply with user requests and regulatory requirements.

1.2 Problem of Interest

Trajectory data mining entails analyzing sequential spatial data that trace the movement of objects or individuals over time. It has broad applications in transportation logistics, urban planning, and location-based services, where uncovering movement patterns can improve operational efficiency and user experience. Insights from trajectory mining underpin tasks ranging from traffic management to personalized travel recommendations. As machine-learning models drive these analyses, the ability to retract learned insights becomes critical. Machine unlearning seeks to remove selected data from trained models, thereby enhancing privacy and ensuring compliance with regulations such as the General Data Protection Regulation (GDPR) [2] without degrading model integrity.

1.3 Challenges & Limitations of Current Work

Deep neural networks (DNNs) possess complex, highly entangled representations that make selective unlearning difficult: high dimensionality and non-linear interactions mean that removing a data point can degrade global performance, often necessitating expensive retraining.

Although machine learning has been extensively applied to trajectory mining, unlearning in this domain remains largely unexplored. Trajectory data pose additional obstacles. Their inherent spatiotemporal correlations mean that erasing a single trajectory may distort the learned spatial or temporal patterns of others. Moreover, trajectory-classification models typically rely on shared internal representations (e.g., latent embeddings or clustered behavioral profiles), so influence from an individual contributor is deeply intertwined with the data and model's structure. The sequential nature of trajectories further complicates matters. Consequently, existing unlearning techniques struggle with scalability and fidelity when applied to real-world trajectory-classification problems. Addressing these challenges is essential for building models that are both privacy-preserving and practically useful.

1.4 Our Approach & Contributions

We propose a machine unlearning algorithmic framework for trajectory classification that leverages the *uniqueness* of trajectories within large-scale datasets to enable efficient unlearning while preserving model performance. Our main contributions are:

- Introduce TraceHiding, an importance-aware machine-unlearning algorithmic framework for trajectory data.
- Develop a hierarchical importance scoring scheme for tokens, trajectories, and users.
- Design a teacher–student distillation loss for selective forgetting.
- Demonstrate state-of-the-art speed and fidelity on three real-world datasets.
- Release code, benchmark, and evaluation tools to foster reproducibility.

We posit that the *uniqueness*, quantified later as an *Importance Score*, of a trajectory is a critical factor in assessing the influence of individual data points. Highly unique trajectories,

because of their distinct characteristics and sparse representation, exert large influence on the learned model. By dampening the impact of these unique points, the unlearning process becomes more tractable, whereas suppressing non-unique (uniform) data can unduly harm performance. Our approach therefore preserves the fundamental structure of the dataset while enabling selective forgetting. Although attenuating unique data may risk losing valuable insights or biasing the model toward common patterns, we demonstrate that an appropriate balance between privacy and utility can be achieved. An overview of the proposed algorithmic framework is shown in Figure 5.1.

The implementation and evaluation tools associated with this thesis are publicly available for reproducibility and further research:

TraceHiding Project Repository

The official source code for this thesis is available on GitHub:

`https://github.com/alifa98/TraceHiding`

This repository includes all relevant scripts, model definitions, data processing tools, and evaluation utilities used throughout the research.

Point2Hex: Data Preprocessing Tools Repository

A companion repository for preprocessing raw mobility data and converting them into higher-order representations:

`https://github.com/alifa98/point2hex`

This includes tools for spatial transformation, including routing, map matching, point to hexagon conversions, and visualization.

1.4.1 Publications

- Ali Faraji, Jing Li, Gian Alix, Mahmoud Alsaeed, Nina Yanin, Amirhossein Nadiri, and Manos Papagelis. 2023. POINT2HEX: Higher-order mobility flow data and resources. In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems* (SIGSPATIAL '23). ACM, New York, NY, USA, Article 69, 1–4. `https://doi.org/10.1145/3589132.3625619` [3]

- Ali Faraji and Manos Papagelis. 2025. TRACEHIDING: An algorithmic framework for machine unlearning in mobility data. *ACM Transactions on Spatial Algorithms and Systems* (submitted). [4]

1.5 Thesis Organization

Chapter 2 reviews related work in trajectory mining and machine unlearning. Chapter 3 describes the data representation and format used in this work. Chapter 4 introduces preliminary concepts in unlearning and formalizes the problem. Chapter 5 details our methodology, including components, mechanisms, and parameter choices. Chapter 6 presents experimental setups, datasets, baselines, evaluation metrics, and results, followed by discussion. Finally, Chapter 7 concludes the thesis, summarizing findings, contributions, and outlining future research directions.

Chapter 2

Literature Review

2.1 Trajectory Data Mining

Trajectory data mining is dedicated to the extraction of valuable patterns, information, and insights from trajectory data, and has been a prominent research focus for an extended period [5, 6, 7]. The process of mining interesting patterns and useful information from trajectories has applications across various fields such as intelligent transportation systems [8, 9], urban planning [10, 11, 12], environmental monitoring, and public health [13, 14, 15, 16]. There have been many works that are using deep learning for trajectory-data pattern recognition [17]. Given its broad impact, numerous trajectory-related research problems have attracted attention, including trajectory similarity, prediction [18, 19], clustering [20], classification [21, 22], simplification [23, 24, 25], outlier detection [26], and imputation [27]. Trajectory-user linking (TUL) refers to the task of classifying trajectories in order to associate them with their respective users. A number of traditional approaches are available, most of which function by comparing the similarity of the target trajectory to the trajectories for which the user is known. This work focuses on this problem as one of the trajectory classification tasks.

2.2 Machine Unlearning

To the best of our knowledge, there has been no direct work specifically addressing trajectory unlearning in recent years, indicating a gap in the current research landscape that needs further exploration. We discuss the works that have been done in the machine unlearning topic that are more general methods and mostly they are in other fields.

The most direct method for eliminating learned information is to start the training process over, which involves erasing the data designated for removal, and restarting training with the remaining dataset. Through this strategy, we ensure the complete deletion of the data's influence on the model. Termed *exact unlearning*, this technique yields a ground-truth model but is also notably the most resource-intensive option. However, nowadays implementing this procedure for each deletion request is unfeasible due to the enormous scale of data and models involved. Consequently, there is a shift towards circumventing the need for complete retraining, with alternative strategies known as *approximate unlearning* not requiring starting from scratch.

The *approximate unlearning* concept is divided into two different categories: *weak unlearning* and *strong unlearning* [28]. In *weak unlearning*, the aim is to adjust the output of the model, while in *strong unlearning*, the goal is to make the model's parameters similar to the ground truth model.

Aditinally, *analytic unlearning* denotes a family of data-deletion techniques that avoid both full retraining ("exact" unlearning) and error-tolerant heuristics ("approximate" unlearning) by exploiting a closed-form, or *analytic*, expression for model parameters. The learner is first rewritten so that its parameters depend only on a compact set of additive sufficient statistics—such as class-conditional counts, node-level histograms, or low-order moments. When one or more training examples must be forgotten, the algorithm simply subtracts each example's contribution from these cached statistics and recomputes the parameters in

closed form. Because no iterative optimisation is required, deletion latency drops by orders of magnitude relative to full retraining, while the resulting model is *exactly* what would have emerged had the data never been incorporated. This paradigm underlies the summation-form learners of Cao and Yang [29] and the DaRE forests of Brophy and Lowd [30], and it extends naturally to any algorithm whose optimum can be expressed as a deterministic function of additive statistics. We are not grouping this studies with other neural network-related studies because it does not introduce any unlearning techniques for neural networks.

2.2.1 Exact Unlearning

Few works have been done in the area of exact machine unlearning. A well-known solution for machine unlearning was presented by Bourtoule et al. [31]. They introduce the concept of SISA (Sharded, Isolated, Sliced, and Aggregated) training as a novel framework designed to address the challenge of efficiently unlearning specific data points from machine learning models. SISA training aims to mitigate the computational and time overhead associated with the unlearning process by strategically organizing training data and procedures into manageable parts. This approach not only facilitates compliance with privacy laws by enabling models to 'forget' data upon request but also maintains the practicality of deploying machine learning models in sensitive and dynamic environments where data privacy cannot be compromised. Through comprehensive evaluation across various datasets and learning tasks, they demonstrate the effectiveness of the SISA training approach in reducing the time and resources required for unlearning, while also highlighting potential challenges and limitations, such as the creation of weak learners and the complexity of hyperparameter search. Their findings suggest that SISA training presents a viable solution to the unlearning problem, balancing efficiency with the need to adhere to privacy regulations. It concludes with a discussion on the implications of SISA training for practical data governance and suggests

avenues for future research, marking a significant step forward in the field of machine learning with respect to data privacy and the right to be forgotten.

Since we cannot easily divide nodes and edges and train a model on different parts of graphs, exact unlearning in graph data is more complicated than in other earlier modalities. To address this, Chen et al. [32] proposed an alternative exact unlearning model for graphs called *GraphEraser*. In their work, Chen et al. [32] introduce two innovative graph partitioning methods aimed at achieving balance in shard sizes while considering the graph structure and node features. The first method, Balanced Graph Partitioning Using Community Detection (BLPA), enhances the Label Propagation Algorithm to create balanced shards through community detection, focusing on graph structure. It involves steps like random node assignment to shards, reassignment profile calculation and sorting, and balanced label propagation. The second method, Balanced Graph Partitioning Using Embedding Clustering (BEKM), integrates graph structure and node features via an embedding clustering approach, employing a pretrained GNN model for node embedding, centroid initialization, distance calculation and sorting, and balanced clustering. Additionally, the work introduces a Learning-based Aggregation (LBAggr) method to effectively combine different shard models' predictions into a final aggregated prediction by assigning and optimizing Importance Scores to each model based on their accuracy contributions. The unlearning process within the *GraphEraser* framework, whether for node or edge unlearning, involves identifying the shard containing the node, removing the node and its data, and retraining the shard model to forget the node's information, ensuring unlearning for deletion requests.

2.2.2 Approximate Unlearning

Approximate unlearning has emerged as a crucial research area, aiming to efficiently remove the influence of specific data points or subsets from trained models without the need for

complete retraining from scratch. Comprehensive surveys provide taxonomies and overviews of this rapidly developing field [33, 28, 34]. A variety of techniques have been proposed, including retrain-free methods that directly modify model parameters using techniques like selective synaptic dampening based on Fisher Information [35, 36] or error-maximizing noise injection [37]. Other approaches leverage model structure, such as inducing sparsity [38] or employing specialized architectures [39]. Further strategies involve unified gradient-based frameworks [40] and meta-algorithms adapting to unlearning difficulty [41]. Contributing to this line of research, our work introduces a data-driven importance-scoring scheme and embeds these scores within the optimization process to achieve efficient unlearning, preserving overall utility, and removing the desired data while keeping frequent patterns intact.

Beyond these general methods, research delves into specialized frameworks and addresses key challenges like fairness, certification, and applicability to modern architectures. Teacher-student models are employed for knowledge transfer and controlled forgetting [42, 43, 44]. Theoretical work focuses on providing guarantees under adaptive attacks [45], establishing efficiency bounds [46], integrating fairness considerations [47], and developing certified unlearning algorithms [48, 49]. There is also significant interest in applying and adapting unlearning techniques for large language and generative models [50, 51, 52, 53, 54, 55, 56, 57, 58, 59], as well as understanding and mitigating the security risks and potential attacks associated with the unlearning process itself [60, 61, 62, 63]. This is complemented by efforts to adapt unlearning for federated contexts [64]. For interested readers, a comprehensive literature review of approximate unlearning is provided in Section 2.3.

2.2.3 Analytic Unlearning

The work by Cao and Yang [29] focuses on enabling machine learning systems to efficiently forget specific training data by transforming traditional learning algorithms into a summation

form. This transformation allows the system to update only a few summations when data needs to be forgotten. The summation-based method ensures that the forgetting process is both quick and does not compromise the accuracy or integrity of the remaining learned model. Brophy and Lowd [30] introduce DaRE (Data Removal-Enabled) forests, an innovative variant of random forests that allows for efficient removal of training data with minimal retraining. By integrating random nodes at the upper levels and caching statistical data at each node, DaRE trees can update only necessary subtrees, maintaining model accuracy while significantly reducing the time required for data deletion. This method demonstrates substantial efficiency improvements, deleting data orders of magnitude faster than traditional retraining approaches, with experiments on various datasets showing minimal loss in predictive performance.

2.3 Related Work in Approximate Unlearning

Machine unlearning, particularly approximate methods that avoid full model retraining, has become a critical area of research for managing deployed machine learning systems, addressing privacy regulations (such as the Right to be Forgotten), and ensuring model maintainability. While exact unlearning guarantees perfect removal of data influence, it often incurs significant computational costs, similar to retraining. Approximate unlearning methods offer a trade-off, aiming for efficient removal with negligible residual influence, making them practical for large-scale models. This section provides a comprehensive overview of the diverse approaches, theoretical considerations, and challenges within the field of approximate unlearning, expanding upon the summary presented earlier in this chapter.

2.3.1 Methods in Approximate Unlearning

Surveys and Overviews

Several surveys [33, 28, 34] provide a thorough review of machine unlearning techniques, categorizing them into *exact* and *approximate* methods. They highlight the strengths and limitations of each approach and identify key challenges and future research directions.

General Approximate Unlearning Methods (Retrain-Free, Second-Order, Pruning, Federated, etc.)

Foster et al. [35] propose Selective Synaptic Dampening (SSD), a retrain-free, post-hoc approach for efficient machine unlearning that balances the trade-off between forgetting specific data and preserving model utility. Using the Fisher Information Matrix (FIM), SSD identifies parameters that are disproportionately important to the forget set and dampens them proportionally to their importance, ensuring minimal impact on the retain set. This granular method addresses limitations of prior retrain-free approaches, offering significantly faster execution while achieving performance comparable to retrain-based methods. Jia et al. [38] discussed model sparsification as a way to facilitate unlearning. The core innovation is the use of model sparsity - the idea that models with a large number of weights set to zero (sparse models) can simplify the unlearning process. The authors propose a method that first prunes the model to induce sparsity and then applies unlearning techniques. Through various experiments, Jia et al. work demonstrates that this approach effectively reduces the computational cost and complexity of machine unlearning. The experiments show a significant improvement in unlearning performance without compromising the accuracy of the retrained model. The findings suggest that model sparsity can serve as a powerful tool to simplify machine unlearning, offering a balance between efficiency and effectiveness. This contributes to the broader field of privacy-preserving machine learning by providing a practical solution

to the unlearning challenge. Tarun et al. [37] introduce a novel framework for unlearning data in machine learning models to enhance privacy and security. It proposes an error-maximizing noise generation and impair-repair weight manipulation method that efficiently unlearns data without revisiting the full training dataset, making the process fast and scalable. In their work, unlearning multiple classes requires a similar number of update steps as for a single class, making the proposed approach scalable to large problems. Golatkar et al. [36] explore a method for selectively removing specific information from trained deep neural networks without requiring retraining from scratch. The method involves applying a scrubbing function to the model's weights to remove information about specific data subsets without retraining from scratch. This is achieved by leveraging the stability of stochastic gradient descent (SGD) and using the Fisher Information Matrix (FIM) to optimally perturb the weights. The Forgetting Lagrangian balances the loss on retained data with the removal of information about the forgotten data, ensuring the model performs as if it never encountered the data to be forgotten. Lin et al. [39] propose a novel method for machine unlearning, aimed at efficiently and securely removing specific data contributions from machine learning models. The method combines an *Entanglement-Reduced Mask (ERM)* structure and a *Knowledge Transfer and Prohibition (KTP)* technique. The ERM structure reduces the knowledge overlap between classes during training, while the KTP method transfers the knowledge of non-target data points to a new model and prohibits the knowledge of target data points. Extensive experiments demonstrate the method's effectiveness, efficiency, high fidelity, and scalability. Foster et al. [35] present a comprehensive approach to machine unlearning through Selective Synaptic Dampening (SSD). The method is notable for its retrain-free nature, rapid execution, and competitive performance in preserving model accuracy while ensuring data privacy. SSD represents a significant step forward in the field of machine unlearning, offering a practical solution that balances effectiveness, efficiency, and storage requirements. Gu et al. [64] introduce Ferrari, a federated feature unlearning framework that uses feature sensitivity to selectively remove

specific features without requiring other clients' participation. By minimizing sensitivity through local optimization, Ferrari effectively unlearns features while preserving model utility. Tested across scenarios like sensitive, backdoor, and biased feature removal, it outperforms existing methods in effectiveness and utility preservation in the context of federated learning. Huang et al. [40] present a unified gradient-based framework for Machine Unlearning that addresses limitations of existing methods by leveraging remain-preserving manifolds and decomposing updates into forgetting, retaining, and saliency-modulated components. To improve scalability, they introduce a fast-slow parameter update strategy for efficient Hessian approximation. They have experimented and reported results of their proposed method in tasks like classification and image generation. Zhao et al. [41] investigate the factors that influence the difficulty of machine unlearning and propose the Refined-Unlearning Meta-algorithm (RUM) to address these challenges. They identify two key factors: the entanglement between forget and retain sets in the embedding space and the degree of memorization of forget set examples, both of which significantly affect unlearning performance. RUM refines the forget set into homogeneous subsets based on these factors and applies tailored and existing unlearning algorithms to each subset sequentially, improving the balance between forgetting quality and retain set utility. Evaluated on benchmarks like CIFAR-10 and CIFAR-100, RUM demonstrates substantial performance improvements over existing methods, advancing the understanding and efficacy of machine unlearning algorithms.

Teacher–Student Frameworks

Chundawat et al. [42] propose a novel unlearning method that involves using both competent and incompetent teachers in a student-teacher framework to induce forgetfulness in the model. By selectively transferring knowledge from these teachers to the student, the model can effectively forget certain data. They investigate the utility of a teacher-student framework with knowledge distillation to develop a robust unlearning method that supports forgetting

single or multiple classes of data, sub-class level forgetting, and random subset-level forgetting. The authors also address concerns about privacy leakage in unlearned models and propose a new metric to evaluate the generalization ability of unlearning methods. The work presented focuses on unlearning in deep networks, aiming to develop efficient and practical methods that do not require additional models or specific optimization techniques during training. Unlike existing methods that have constraints on the training process and high computational costs, the proposed method is versatile and applicable to different types of deep networks such as CNNs, Vision transformers, and LSTMs. The experiments conducted across various multimedia domains demonstrate the wide applicability and effectiveness of the proposed unlearning method, offering a practical solution for data removal in machine learning models. Kurmanji et al. [43] present SCRUB, a novel unlearning algorithm that addresses the limitations of existing methods in scalability and consistency across applications. SCRUB leverages a teacher-student framework to selectively unlearn data without compromising model utility. It introduces a contrastive optimization objective and a rewinding procedure to fine-tune the forgetting process. Empirical evaluations demonstrate SCRUB's superior performance in achieving high forget quality while maintaining accuracy on retained data, making it a robust solution for various unlearning scenarios, including removing biases, resolving confusion, and protecting user privacy. Chundawat et al. [44] propose zero-shot machine unlearning methods that function without access to any original training data. They introduce two novel approaches: one based on generating error-minimizing and error-maximizing noise to simulate the training data, and another using a *gated knowledge transfer (GKT)* framework that filters out information related to the forget class during the transfer from a teacher model to a student model. The effectiveness of these methods is demonstrated through extensive experiments on benchmark vision datasets, showing good protection against privacy attacks. They also introduce the *Anamnesis Index* (AIN) as a new metric to assess the quality of unlearning.

Theoretical Frameworks, Fairness, and Certified Unlearning

Gupta et al. [45] present a novel approach to efficiently handle data deletion requests in machine learning models, particularly under adaptive scenarios where deletion requests depend on previously published models. The authors propose a method that combines differential privacy with deletion algorithms. They highlight the limitations of previous approaches through theoretical analysis and practical attacks, and demonstrate the effectiveness of their method through experiments on standard datasets, showing that it can handle arbitrary model classes and training methodologies while maintaining strong deletion guarantees. Sekhari et al. [46] demonstrate that their unlearning algorithms can delete up to $O(n/d^{1/4})$ samples, significantly improving over differentially private methods which delete $O(n/d^{1/2})$ samples. By focusing on both computational and storage efficiency, they introduce methods that rely on storing simple data statistics rather than the entire dataset, ensuring that the unlearning process is scalable. The work bridges the gap between empirical risk minimization and generalization to unseen data. The methods and theoretical guarantees are specifically designed for convex and strongly convex loss functions, limiting their direct applicability to non-convex functions commonly found in modern machine learning applications. Extending these findings to non-convex settings would require significant additional research to address the unique challenges posed by non-convex optimization landscapes. Oesterling et al. [47] introduce a method for fair machine unlearning, addressing the challenge of maintaining fairness while efficiently removing data from machine learning models. The method combines a convex fair loss function with a second-order Newton update to adjust model parameters, ensuring both fairness and unlearning. Theoretical guarantees for statistical indistinguishability and fairness are provided, and extensive experiments on real-world datasets demonstrate the method's effectiveness in preserving fairness and accuracy compared to existing unlearning methods. Chien et al. [48] introduce a certified machine unlearning framework using Projected Noisy Stochastic

Gradient Descent (PNSGD) to efficiently remove data points while ensuring approximate unlearning guarantees through Wasserstein distance tracking. Supporting sequential and batch unlearning, PNSGD achieves superior privacy-utility-complexity trade-offs, requiring only 2–10 percents of the computations needed for retraining, while maintaining competitive utility on benchmarks like MNIST and CIFAR-10. Also, Chien et al. [49] introduce Langevin Unlearning, a privacy-preserving framework using noisy gradient descent to efficiently address approximate unlearning tasks, including non-convex problems. By leveraging Rényi Differential Privacy (RDP), it ensures robust privacy guarantees, supports batch and sequential unlearning, and demonstrates faster privacy recovery compared to retraining. Evaluations on MNIST and CIFAR10 show superior utility-privacy trade-offs and computational efficiency over methods like Delete-to-Descent (D2D).

Unlearning in LLMs and Generative Models

Huang et al. [50] propose δ-UNLEARNING, a method to unlearn sensitive data from black-box large language models (LLMs) without access to their internal weights. It achieves unlearning by adjusting the logit offsets using smaller models. This method maintains model performance on general tasks while effectively removing sensitive information. Li et al. [51] proposed Single Image Unlearning (SIU), an efficient method for machine unlearning in multimodal large language models (MLLMs) using only a single training image. SIU employs fine-tuning data designed to forget visual concepts while preserving factual and non-targeted knowledge. It introduces a Dual Masked KL-Divergence Loss to optimize unlearning without degrading model utility. Evaluated on their benchmark MMUBench, SIU outperforms existing methods in efficacy, generality, and robustness. Yao et al. [52] propose an efficient unlearning method for large language models, utilizing negative examples to eliminate undesirable behaviors such as harmful responses, copyrighted content, and hallucinations. This approach, based on gradient ascent, surpasses traditional reinforcement learning from human feedback

(RLHF) in alignment efficiency with notably less computational effort. Liu et al. [53] introduced ECO (Embedding-Corrupted Prompts) which enable large language models to unlearn specific information without retraining by classifying forget prompts and corrupting their embeddings. Tested on various unlearning tasks, ECO achieves high forget quality with minimal utility loss. Wang et al. [54] propose a data attribution method for text-to-image models using machine unlearning to identify influential training images without retraining. By increasing the training loss of synthesized images while preserving unrelated data with Fisher Information-based regularization, they efficiently pinpoint critical images. Validated on benchmarks like MSCOCO, their approach outperforms methods like TRAK and D-TRAK, for understanding training data influence in generative models. Ko et al. [55] introduce a framework for machine unlearning in text-to-image models, addressing the trade-off between unlearning undesired concepts and preserving text-image alignment. Using a restricted gradient approach and dataset diversification, their method ensures balanced updates and maintains model utility. Applied to CIFAR-10 and Stable Diffusion, it outperforms baselines like SalUn and ESD in effectively removing undesired concepts while retaining semantic alignment and generation quality. Park et al. [56] introduce Direct Unlearning Optimization (DUO), a method for robustly removing unsafe visual features from text-to-image models while preserving unrelated content generation. DUO uses paired datasets of unsafe and safe images generated via SDEdit and applies preference optimization to guide selective forgetting. Output-preserving regularization ensures the retention of unrelated features, and KL-constrained optimization minimizes deviation from the original model. Demonstrating superior robustness to adversarial attacks, DUO outperforms existing methods like ESD and SPM in tasks such as removing nudity and violence from Stable Diffusion models. Zhang et al. [57] propose AdvUnlearn, a robust concept erasure framework for diffusion models (DMs) using adversarial training (AT). By optimizing the text encoder and introducing utility-retaining regularization, it balances unlearning effectiveness and image quality while improving

efficiency through fast AT. AdvUnlearn demonstrates significant robustness against adversarial prompts and preserves utility across tasks like nudity, style, and object erasure, with its optimized text encoder transferable across DMs. Ji et al. [58] introduce ULD (Unlearning from Logit Difference), a framework that leverages an assistant LLM to efficiently unlearn specific knowledge in large language models (LLMs). By reversing conventional objectives and employing logit subtraction, ULD avoids degeneration and catastrophic forgetting while preserving model utility. Using Low-Rank Adaptation (LoRA) and data augmentation, ULD achieves near-perfect forgetting with no utility loss, outperforming baselines on benchmarks like TOFU [65] and Harry Potter with improved efficiency and scalability. Jia et al. [59] introduce WAGLE, a framework that enhances unlearning in large language models by leveraging weight attribution through bi-level optimization. WAGLE identifies influential weights critical for unlearning, improving efficacy across tasks like data removal and toxicity mitigation while preserving model utility. The method is computationally efficient, adaptable to various unlearning algorithms, and highlights the modularity of LLMs.

Security and Attacks in Unlearning

Also, there are some works discuss the potential risks and security in machine unlearning Lin et al. [60] introduced the Two-Stage Backdoor Defense (TSBD) framework to mitigate backdoor vulnerabilities in deep neural networks. TSBD identifies backdoor-related neurons by leveraging correlations in weight changes during clean and poison unlearning, followed by activeness-aware fine-tuning with gradient-norm regulation to restore clean accuracy and suppress backdoor reactivation. Experiments show that TSBD outperforms state-of-the-art defenses in reducing attack success rates while maintaining high clean accuracy. Bertran et al. [61] reveal that machine unlearning in simple models, such as linear regression, can enable reconstruction attacks that recover deleted data using parameter differences. By linking these changes to gradients, their method extends to complex models and loss functions,

achieving high accuracy in reconstructing data across various datasets. Their findings highlight significant privacy risks in unlearning without protections like differential privacy, challenging assumptions about the security of simple models. Schwinn et al. [62] proposed embedding space attacks, a novel threat model for open-source LLMs that manipulates continuous embeddings to bypass safety alignments, extract residual unlearned information, and recover pretraining data. These attacks outperform discrete methods and fine-tuning, achieving high success rates while exposing vulnerabilities in safety-aligned and unlearned models. The study highlights the significant risks these attacks pose to LLM safety and data privacy, emphasizing the need for stronger defenses. Chen et al. [63] investigate the unintended privacy risks associated with machine unlearning, where data is removed from a machine learning model's training set. The authors propose a membership inference attack leveraging the outputs of an ML model's original and unlearned versions to infer if a sample was part of the original training set. Extensive experiments on various datasets demonstrate that the proposed attack often outperforms classical membership inference attacks, especially on well-generalized models. The study further explores mitigation mechanisms such as releasing only the predicted label, temperature scaling, and differential privacy, showing their effectiveness in preserving privacy. This work highlights the potential privacy vulnerabilities in machine unlearning and suggests directions for more secure implementations.

2.3.2 Evaluation of Approximate Unlearning

The evaluation metrics used by Cao et al. [29], completeness, and timeliness, are crucial for assessing the efficacy and efficiency of the proposed machine unlearning method. Completeness ensures that the unlearning process thoroughly removes all traces of the data to be forgotten, maintaining the integrity and accuracy of the system. Timeliness measures the speed at which the system can forget the data, demonstrating the efficiency gains compared to traditional

retraining. Together, these metrics demonstrate that the unlearning method can quickly and completely remove data from learning systems. Completeness is achieved when the unlearned system behaves identically to a system that was retrained from scratch without the data to be forgotten. Cao et al. quantify completeness by comparing the prediction results of the unlearned model with those of a retrained model on a representative test dataset. A high percentage of matching results indicates a high level of completeness. Since every evaluation metric in some way measures the similarity between the retrained and unlearned models, the most of them basically come under the general category of completeness.

The evaluation of the SISA [66] training method revolves primarily around the metrics of "time to unlearn," "model accuracy," "speed-up," and "degradation in accuracy." Time to unlearn refers to the duration required to effectively remove a data point from the model's training set and update the model accordingly. Model accuracy measures the correctness of the model's predictions on unseen data, evaluated both pre- and post-unlearning to monitor performance impacts. Speed-up is calculated as the ratio of unlearning time using traditional methods to that using SISA, indicating the efficiency improvements offered by the new method. Lastly, degradation in accuracy looks at the reduction in model accuracy due to the unlearning process, important for understanding the trade-offs between unlearning efficiency and model reliability.

Chen et al. [32] use two key metrics for the evaluation of the unlearning method: 1) Unlearning Efficiency: Defined as the average time required to process an unlearning request compared to retraining the model from scratch. It reflects the speed and computational cost of removing a data point's influence from the model, aiming for a faster response than traditional retraining. 2) Model Utility: Measured using the Micro F1 Score, this metric assesses the accuracy of the unlearned model's predictions. The Micro F1 Score is particularly suitable for datasets with class imbalances, as it considers both precision (the accuracy of positive predictions) and recall (the completeness of positive predictions), offering a comprehensive

measure of model performance post-unlearning.

The Zero Retrain Forgetting (ZRF) metric introduced by Chundawat et al. [67] is a novel metric for evaluating machine unlearning methods without the need for a retrained model. This metric is particularly useful because it bypasses the computationally expensive and time-consuming process of retraining, making it practical for assessing the effectiveness of unlearning directly in deployed models. ZRF measures the randomness in a model's predictions by comparing them to those of a randomly initialized model (referred to as the "incompetent teacher" in the unlearning context). A higher ZRF score indicates that the model behaves more randomly on the forgotten set, which implies effective unlearning.

Jia et al. [38] evaluate the performance of machine unlearning methods using a comprehensive set of metrics: Unlearning Accuracy (UA), which measures the model's accuracy on a dataset intended to be forgotten, highlighting the effectiveness of the unlearning process; Membership Inference Attack (MIA) Efficacy on the forgetting dataset, assessing the unlearning method's ability to protect against privacy attacks by measuring how well the method prevents accurate prediction of forgotten data membership; Remaining Accuracy (RA), evaluating the model's performance on the data not targeted for unlearning, ensuring the unlearning process does not degrade the model's overall utility; Testing Accuracy (TA), which examines the model's generalization to unseen data post-unlearning, ensuring that the process of forgetting does not compromise the model's predictive capabilities on new data; and Run-time Efficiency (RTE), assessing the computational efficiency and practicality of the unlearning method by comparing the time cost against retraining from scratch. These metrics collectively provide a holistic view of an unlearning method's effectiveness, efficiency, and impact on model performance and privacy.

The *Anamnesis Index (AIN)* is defined by Chundawat et al. [44] which provided as a metric to measure the effectiveness of machine unlearning. The Anamnesis Index compares the relearning times of an unlearned model and a model trained from scratch, both aimed at

reaching the original model's performance within a margin of $\alpha\%$. AIN values closer to 1 indicate better unlearning efficacy. Values significantly lower than 1 suggest that the unlearned model retains substantial information about the classes it was supposed to forget, indicating ineffective unlearning. Conversely, very high AIN values might indicate over-compensation in the unlearning process, potentially leading to detrimental changes in the model's parameters.

2.4 Gaps and Positioning of The Thesis

In the intersection of trajectory mining and machine unlearning, this thesis identifies a significant gap in the current landscape of research. While trajectory mining provides deep insights into movement patterns, its integration with robust machine unlearning methods remains underexplored. This oversight is particularly critical given the sensitive nature of trajectory data, which often includes personally identifiable information that necessitates strict compliance with data privacy laws. This thesis positions itself to bridge this gap by proposing novel methodology that incorporate advanced machine unlearning techniques in the trajectory mining. These techniques ensure that the models remain adaptive and compliant with privacy standards, thereby addressing both the effectiveness in data utilization and the ethical implications of data retention.

Several scholarly articles have examined and suggested various approaches to reverse the process of learning data, as outlined in the literature review. While there have been studies suggesting a general approach to unlearning, we have not come across any specific research addressing the topic of unlearning trajectories, which use spatiotemporal data. In our proposed work, as mentioned in various articles [68], our focus is on spatiotemporal data structures. We can then apply these techniques in other domains, such as natural language in LLMs, as our data structures and texts share similarities. Further discussion on this topic is provided in chapter 3.

GPS data, by its nature, records continuous spatial movements over time, which results in complex data dependencies that can affect learning models in unforeseen ways when part of the data becomes obsolete or needs removal for compliance with privacy regulations. For instance, the removal of a subset of GPS data, due to either obsolescence or privacy concerns, can significantly alter the trajectory patterns learned by the model. This can have profound implications not just for user privacy but also for the predictive accuracy of the model in applications such as traffic management, route optimization, and location-based services.

Therefore, this thesis aims to extend the machine unlearning framework to address the intricacies of GPS-based spatiotemporal data. We propose to investigate methods that can efficiently minimize the residual data influence in trained models, thus ensuring that the unlearning process respects both the spatial and temporal dimensions inherent in such datasets. By doing so, we hope to establish foundational techniques that enhance the adaptability of ML models to dynamically changing environments, furthering the capability of these systems to deliver timely and accurate predictions while upholding data privacy standards.

Chapter 3

Trajectory Representation and Task Formulation

This chapter explains how raw spatiotemporal trajectory data are processed into a representation suitable for machine learning tasks, with a focus on trajectory classification. We first introduce key concepts, then describe the tokenization method we use. Finally, we formally define the main predictive task studied in this work: *Trajectory-User Linking* (TUL).

3.1 Trajectory Representation

We begin with the standard definition of a raw spatiotemporal trajectory.

Definition 3.1. ***(Spatiotemporal Trajectory*** T_c***)*** *A trajectory is a* temporally *ordered sequence of spatiotemporal points* p *that belong to a user* c*. Each point is a tuple* (l, t)*, where* l *is the location (e.g., GPS coordinates) and* t *is the timestamp.*

Raw spatiotemporal trajectories pose challenges for direct use in deep learning models. They are often noisy, high-dimensional, and irregularly sampled in space and time. To create representations that models can use, researchers often apply discretizations or tokenization.

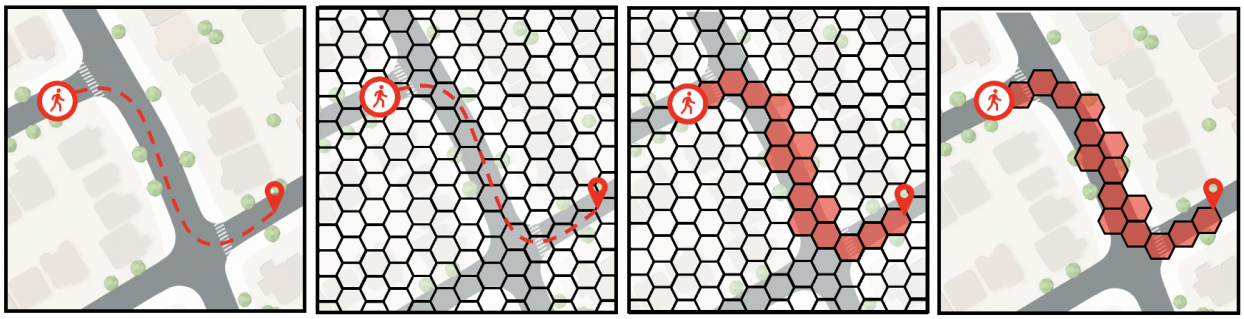


Figure 3.1: A trajectory traced over a hexagonal grid. Each visited hexagon forms a token in the sequence.

For our experiments, we utilizing the recently introduced data in our lab known as Higher-Order Mobility Flow [3]. You can see part of the tessellated map and trajectories heatmap in Figure 3.2. The sequence of GPS points is converted into tokens in two steps:

1. **Tessellate the map:** Partition the continuous spatial domain into cells (referred to as *hexagons* or *blocks*).

2. **Map points to cells:** Assign each GPS point (l, t) to the ID of the cell (e.g. hexagon) that contains l.

Figure 3.1 shows how a continuous path becomes a sequence of tokens which here are hexagons. This transformation lets us redefine a trajectory as a sequence of discrete tokens that capture higher-order movement patterns:

Definition 3.2. ***(Higher-Order Trajectory $\mathcal{T}_c$)*** *For user c, $\mathcal{T}_c$ is a* temporally *ordered sequence of discrete tokens (hexagon IDs), each marking a region the user visited.*

The same idea applies to **higher-order check-in datasets**. For example, check-ins at points of interest in location-based social networks. Each check-in (or its enclosing region) becomes a token, and the time-ordered list of tokens forms the trajectory $\mathcal{T}_c$.

In short, mapping raw spatiotemporal data to tokens through hexagonal tessellation yields a compact, noise-resilient representation that standardizes trajectories despite irregular

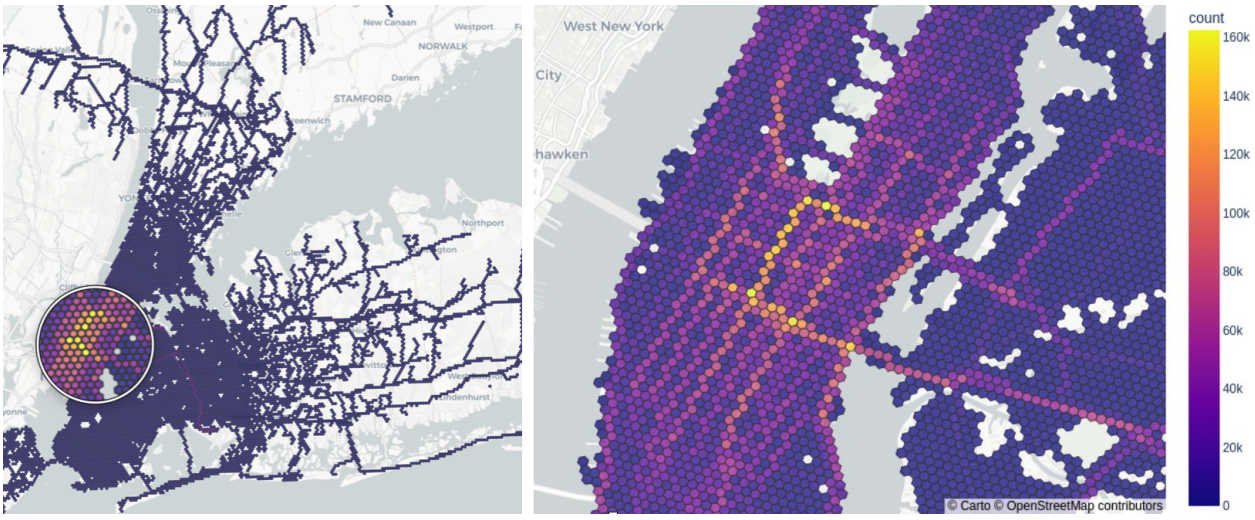


Figure 3.2: An example of a heatmap of hexagon-sequence trajectories on a tessellated map.

sampling and generalizes across diverse mobility data. We are using the higher-order mobility flow of the following datasets (see Table 3.1 for a summary of the datasets' statistics):

Rome [69]. Consisting of recorded traces of taxi cabs in Rome, Italy.

Geolife [70, 71, 72]. Collected from several users' Gps-based devices during their outdoor activities (a total distance of ∼1.2 million km)

Foursquare-NYC [73]. Poi check-ins recorded by the phones of several users in the city of New York.

Table 3.1: Statistics of the higher-order mobility flow datasets (the original datasets are prefixed by 'Ho-' to name the Higher-order datasets.)

Dataset	# Objects	# Trajectories	Time Period	Type	Resolution
Ho-Rome	315	5,873	02/01/14 – 03/02/14	Gps traces	8
Ho-Geolife	57	2,100	04/01/07 – 10/31/11	Gps traces	8
Ho-NYC	1,083	49,983	04/12/12 – 02/16/13	Visits	9

3.2 Analogy to Text Data

In the context of analyzing trajectories represented as sequences of tokens, we can draw parallels to natural language processing (NLP) techniques, particularly those used in sequence-to-sequence architectures and large language models (LLMs). Here's a detailed explanation:

- **Representation of Trajectories**: Each block in a given trajectory can be viewed similarly to a token in a sentence. This means that just as tokens combine to form meaningful sentences, sequences of blocks (or hexagon trajectories) come together to represent a specific path or movement pattern.

- **Application of NLP Techniques**: By using sequence-to-sequence models, which are typically employed in transforming one sequence of token to another, we can potentially transform hexagon sequences from one form to another. This could be useful in predicting future paths or understanding complex movement patterns. Similarly, the application of techniques from large language models can help in interpreting the context of these hexagon sequences or generating new, plausible trajectories based on learned patterns.

- **Semantic Differences and Similarities**: While each word in a language has a distinct meaning, the role of each hexagon in a sequence might not be as clearly defined. However, the overall sequence of hexagons carries significant information about the trajectory, akin to how a sentence conveys meaning through the arrangement of its words and the attention-based models (Transformers [74]) are working based on this fact [75].

- **Leveraging Recurring Patterns**: Just as certain facts in language (e.g., "Earth is in the solar system") recur frequently across different texts, specific patterns also emerge in trajectory data. For example, trajectories along a highway tend to share common

subsequences since there are limited options for direction changes. Recognizing and leveraging these repetitive patterns can enhance the model's ability to learn and predict trajectory behaviors efficiently. Also we can use this fact in unlearning and introduce the concept of Importance Score for a sequence.

- **Importance of Sequence in Unlearning**: In trajectory analysis, understanding the importance of certain sub/sequences, much like key phrases in a text, can be crucial. For trajectories on a highway, recognizing the segment of straightforward movement can help in simplifying the model and focusing on variations or deviations from the norm, which might be more informative.

By elaborating on these points, the concept becomes more accessible and easier to grasp, illustrating how techniques from NLP can be innovatively applied to the analysis of trajectories in hexagon sequences. This approach not only enhances our understanding of movement patterns but also opens up new possibilities for predictive modeling and data analysis in various applications.

3.3 Trajectory-User Linking (TUL)

Using the higher-order representation $\mathcal{T}_c$, we define a key task in mobility analytics and privacy research: *Trajectory-User Linking.*

Problem statement. Given an observed trajectory $\mathcal{T} = (t_1, t_2, \ldots, t_k)$ and a set of known users C, identify the user $c \in C$ who most likely generated $\mathcal{T}$.

The effectiveness of TUL highlights important **privacy risks**. Accurate re-identification from tokenized trajectories can expose individuals even when raw data are anonymized. These risks must be weighed carefully when sharing data or deploying machine learning models, especially when model unlearning is a concern.

Chapter 4

Problem

In this chapter, we lay out the mathematical foundations that underpin the remainder of the thesis. We begin by introducing the core terminology, symbols, and conventions on which our analysis relies; for quick reference, a complete glossary is provided in Table 4.1. After fixing this common language, we formalize the central problem addressed in the study, specifying the precise inputs, desired outputs, and any structural or computational constraints.

4.1 Preliminaries

Definition 4.1. ***(Training Dataset*** $\mathcal{D}_t$***)*** *The training dataset is the collection of trajectory data used to train a machine learning model for trajectory classification. It consists of sequences* $\mathcal{T}$*, each of varying length* l*, representing trajectories associated with a user* c*.*

Training dataset serves as the foundation for the model's learning process, enabling it to adjust its parameters to accurately classify trajectories before any unlearning is applied.

Definition 4.2. ***(Forgetting / Unlearning Dataset*** $\mathcal{D}_u$***)*** *The forgetting/unlearning dataset consists of specific trajectory data points targeted for removal from the machine learning*

Symbol	Description
C	Set of all users
$\mathcal{T}_c$	A Trajectory that belongs to the user $c \in C$
l	Trajectory Length
$\mathcal{D}_t$	Training Dataset
$\mathcal{D}_u$	Unlearning/Forgetting Dataset
$\mathcal{D}_r$	Remaining Dataset
M_t	Originally Trained Model on $\mathcal{D}_t$
M_u	Unlearned Model
M_r	Trained Model on $\mathcal{D}_r$
W_t	Weights of M_t trained on $\mathcal{D}_t$
W_u	Weights of Unlearned Model M_u
W_r	Weights of M_r trained on $\mathcal{D}_r$
$\mathcal{A}(\cdot)$	Learning Process
$\mathcal{U}(\cdot)$	Unlearning Process
$\xi(x)$	Importance Score of trajectory x
$\mathcal{B}$	Set of All Tokens (Block) in the Dataset
$\mathcal{D}_t^c$	Set of all trajectories in $\mathcal{D}_t$ that belong to user c

Table 4.1: Summary of notations.

model's knowledge. This dataset is a subset of the training dataset ($\mathcal{D}_u \subseteq \mathcal{D}_t$) and includes sequences $\mathcal{T}$ associated with the corresponding users.

The term "Unlearning Dataset" and "Forgetting Dataset" can be used interchangeably. We mostly use the term "Unlearning Dataset" throughout this work.

Definition 4.3. ***(Learning Process $\mathcal{A}$)*** *The learning process $\mathcal{A}(\cdot)$ is the mechanism through which a machine learning model is trained on a trajectory dataset. It involves the model learning to identify patterns in the trajectory sequences and to classify these trajectories based on the input data. The input to the learning process is the trajectory training dataset $\mathcal{D}_t$, and the output is a trained model M_t with learned parameters (weights) W_t, optimized for trajectory classification.*

Definition 4.4. ***(Remaining Dataset $\mathcal{D}_r$)*** *The remaining dataset is the portion of the original trajectory training dataset that remains after the unlearning dataset has been removed.*

It represents the trajectory data on which the model should retain its knowledge and continue to perform classification accurately. The equation $\mathcal{D}_r = \mathcal{D}_t \setminus \mathcal{D}_u$ indicates that this dataset is the difference between the training dataset and the unlearning dataset. If a model is retrained on this remaining dataset using the learning process, the resulting model M_r is referred to as the retrained model.

Definition 4.5. ***(Unlearning Process $\mathcal{U}$)*** *The unlearning process $\mathcal{U}(\cdot)$ is designed to remove the influence of the unlearning dataset, which consists of specific trajectory sequences, from a trained model. This process modifies the model to ensure that it behaves as if it had never been exposed to the removed trajectory data, thus preserving privacy or correcting the model's learned behavior. The output of the unlearning process is an unlearned model M_u with adjusted weights W_u, which no longer retains knowledge of the unlearned trajectories.*

4.2 Problem Statement

Now we define the machine unlearning task that we are trying to do in this work as follows:

Let $\mathcal{D}_t$ be a training dataset consisting of user-linked trajectories, where each trajectory is a sequence of tokens associated with a user, and let $\mathcal{D}_u \subseteq \mathcal{D}_t$ represent the subset of trajectories to be unlearned. The trained model $M_t = \mathcal{A}(\mathcal{D}_t)$ maps the trajectories to user predictions, where $\mathcal{A}$ is the training algorithm.

Unlearning in Trajectory Classification. The unlearning process is defined as $\mathcal{U}(M_t, \mathcal{D}_t, \mathcal{D}_u)$, where $\mathcal{U}$ is a procedure that takes the original trained model $M_t = \mathcal{A}(\mathcal{D}_t)$, the full training dataset $\mathcal{D}_t$, and the unlearning dataset $\mathcal{D}_u$ to produce an unlearned model $M_u = \mathcal{U}(M_t, \mathcal{D}_t, \mathcal{D}_u)$. The resulting model M_u should perform as if the trajectories in $\mathcal{D}_u$ were never used during training, ensuring that the influence of $\mathcal{D}_u$ is removed from M_t.

A trivial solution for this problem is to retrain the model M_t from scratch, but we are trying to avoid form retraining as it is impractical in most of the large models. In practice, M_u and M_r may not be identical for every unlearning process.

Chapter 5

Proposed Methodology

In this work, we are enhancing the framework presented by Kurmanji et al. [43] and Chundawat et al. [42]. Previous methods of machine unlearning, such as SCRUB [43] and BAD-T [42], rely on either contrastive learning (SCRUB) or a combination of competent and incompetent teachers (BAD-T) to unlearn data. SCRUB uses a teacher-student framework where the student learns to forget specific data points by maximizing the error on the unlearning dataset while minimizing it on the remaining dataset. On the other hand, BAD-T employs two teachers: a competent one to provide correct information about remaining data and an incompetent one to induce forgetfulness on the unlearning data by introducing noise. Both methods are effective in traditional supervised learning tasks, but they do not account for the varying importance or influence of individual data points within complex datasets like trajectories. Trajectory data often contains rich, interconnected patterns where certain trajectories or parts of trajectories can have more influence on the model's predictions. Therefore, introducing an *Importance Score* allows us to control how much each trajectory contributes to the unlearning process. By adjusting the importance of each trajectory, we can ensure that the unlearning process is more efficient and targeted, focusing on critical or highly impactful trajectories while preserving general model utility. This refinement addresses

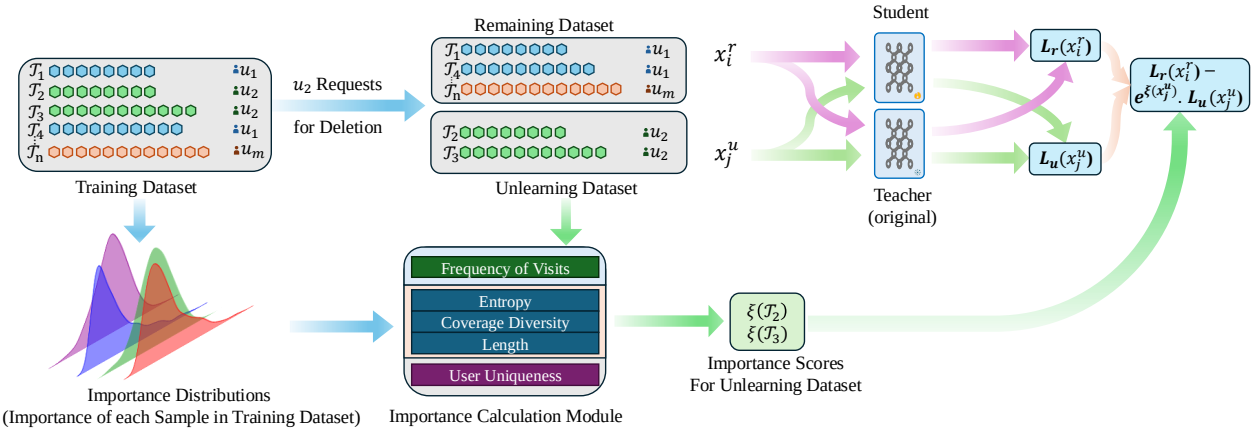


Figure 5.1: This diagram shows our unlearning process where the *Student Model* is trained to forget data from the *Unlearning Dataset* ($x_j^u \in \mathcal{D}_u$) while retaining knowledge from the *Remaining Dataset* ($x_i^r \in \mathcal{D}_r$). The *Teacher Model* guides this process, with two loss functions: L_r, ensuring retention of x_i^r, and L_u, promoting unlearning of x_j^u. The combined loss directs gradient calculation and then backpropagation for *student model*, allowing selective forgetting based on sample importance $\xi(x)$.

the limitations of previous methods, which treated all data points equally, making them less effective in scenarios where data points vary in their influence.

Our algorithmic framework is organized around three interconnected components: **(i) a data-driven importance calculation, (ii) a teacher–student module, and (iii) a dedicated loss function**. These elements and the way they interact are illustrated in Figure 5.1. The remainder of this chapter details each component in turn.

5.1 Data-driven Importance Score

We begin by assigning data points a scalar *Importance Score* that reflects their expected contribution to the learning objective. Rather than relying on ad-hoc heuristics, we exploit the wealth of *historical* trajectory data to quantify this contribution. Specifically, these scores are computed based purely on properties of the data, such as visit frequency, spatial coverage, entropy, and user uniqueness, without reference to any specific model. By mining these

statistics directly from the corpus, we derive Importance Scores that reflect real, observed influence patterns, grounded in the data distribution itself. In the following paragraph we define this Importance Score and then formalize each score and explain the intuition behind it.

Definition 5.1 (Importance Score ξ)**.** *Let $\mathcal{D} = \{x_i\}_{i=1}^n$ be a dataset where each x_i is a data point. An* Importance Score function *is a mapping $\xi : \mathcal{D} \to \mathbb{R}$ that assigns to each data point $x_i \in \mathcal{D}$ a real-valued score $\xi(x_i)$, representing the significance of x_i within $\mathcal{D}$ for the task of user linking.*

This score serves as a proxy for the contribution of a data point to the performance or behavior of a model trained on the training dataset. The concept of importance in trajectory data can be analyzed at various **hierarchical levels**, including:

- Token Level (e.g., Hexagon Level),
- Trajectory Level (across the entire trajectory), and
- User Level (aggregating multiple trajectories associated with a user).

These levels of importance are interrelated hierarchically. For instance, the importance at the Token Level can serve as a basis to define an Importance Score for an entire trajectory in the Trajectory Level. Subsequently, the importance of multiple trajectories belonging to a user can be aggregated to derive a User Level Importance Score.

5.1.1 Token Level Importance

At the finest granularity, we attach an Importance Score to each **token**, a small spatial segment along a trajectory, so that its local contribution to the training set is quantified. We introduce a Token Level importance measure based on token frequency, which counts how often a given token appears across all trajectories.

i) Frequency of Token: Let $f(b)$ denote the frequency of token b:

$$f(b) = \Big| \{\mathcal{T} \in \mathcal{D}_t \mid b \in \mathcal{T}\} \Big| \tag{5.1}$$

The importance of each token is defined as the inverse of its frequency, $1/f(b)$, so that seldom-seen tokens are deemed more significant. In the context of our trajectory datasets, $f(b)$ can be interpreted as the number of visits to the corresponding hexagon, and these visit counts will later serve as building blocks for Trajectory Level importance.

Why using frequency as a score? Intuitively, locations that are visited rarely reveal atypical movement patterns and therefore carry more information about the diversity of the data than locations that are traversed routinely.

5.1.2 Trajectory Level Importance

At the Trajectory Level, we calculate Importance Scores for each trajectory, considering the entire sequence of tokens as a single unit to evaluate its influence on the data distribution and its contribution to learning the dataset. We consider three Trajectory Level importance metrics: (i) **Coverage Diversity**, (ii) **Information-theoretic importance** (entropy), and (iii) **Trajectory Length**. At the end, we derive a unified Trajectory Level Importance Score that aggregates all of the Trajectory Level Importance Scores.

i) Coverage Diversity: Let $D(x)$ denote the number of unique blocks covered by trajectory $x \in \mathcal{T}$:

$$D(x) = \Big| \{b \mid b \in x\} \Big| \tag{5.2}$$

We can define the *Importance Score* related to the coverage of a trajectory x as follows:

$$\xi_{\text{coverage}}(x) = \frac{D(x)}{|\mathcal{B}|} \tag{5.3}$$

Where $|\mathcal{B}|$ is the total number of unique blocks in the dataset.

Why using coverage diversity as a score? A trajectory that visits many distinct blocks injects proportionally more spatial information and thus exerts a larger influence on the model's parameters.

ii) Information-theoretic: We can define the *Importance Score* of trajectory x as its entropy of the bi-grams in the sequence as follows:

$$\xi_{\text{entropy}}(x) = \frac{1}{H(x)} \tag{5.4}$$

Here, the notation $H(x)$ is the entropy of the sequence x which is defined in the Equation 5.5.

$$H(x) = -\sum_{b \in \text{bigram}(x)} p(b) \log_2 p(b) \tag{5.5}$$

where b is the bi-grams of sequence x. A bi-gram refers to a pair of consecutive elements within a sequence. For example, in a sequence of letters like "ABACAB", the bi-grams are "AB", "BA", "AC", "CA", "AB", we formally define bi-grams set in Equation 5.6.

$$\text{bigram}(s) = \{(s_i, s_{i+1}) \mid 1 \leq i < n\} \tag{5.6}$$

where s_i is the ith bock (token) in the sequence s, and n is the length of the sequence s. Also, the probability $p(x)$ in Equation 5.5 is defined based on the frequency of the bi-gram b in the dataset:

$$p(b) = \frac{\text{Frequency of } b}{\text{Total number of bi-grams}} \tag{5.7}$$

Why using bi-grams? 1) Captures Local Dependencies: Bi-grams consider the relationship between consecutive items, capturing local dependencies and patterns that single-item entropy

might miss. This is particularly useful in sequences where the order of items carries significant information, such as in trajectory, text, DNA sequences, or time series data. 2) Reveals Hidden Patterns: Single-item entropy might not reflect certain patterns or structures that exist within the sequence. Bi-grams can reveal these patterns by showing how often specific pairs of items occur together. For example, in natural language processing, bi-grams can capture common word pairs or phrases that convey more meaningful information than individual words. 3) Increased Sensitivity to Structure: Bi-grams provide a finer level of granularity in understanding the structure of the sequence. By analyzing the co-occurrence of consecutive items, they offer insights into the underlying order that single-item analysis might overlook.

However, using n-grams with $n > 2$ introduces significant challenges in terms of storage and computational efficiency. As the value of n increases, the number of possible n-grams grows exponentially, leading to a substantial increase in the memory required to store them and the processing power needed to update and analyze them. This can be particularly problematic when working with large datasets or sequences. In contrast, bi-grams strike a practical balance by capturing local dependencies and patterns between consecutive items while maintaining manageable computational and storage requirements. Thus, bi-grams offer a robust method for entropy calculation that is both effective and efficient, making them a preferred choice in this application.

Why using Entropy and what is the interpretation? If we interpret "unique" as meaning that the sequence has a specific, recognizable structure or pattern, then a high entropy would indicate the opposite. A highly random sequence does not have a unique, easily identifiable pattern, and thus does not provide much insight into the dataset. Conversely, a low entropy indicates that the sequence has lower randomness and more predictability. This can imply the presence of a unique pattern or structure within the sequence. Therefore, we are using the inverse of entropy as our *Importance Score*.

iii) Trajectory Length: Let $L(x)$ denote the length of a trajectory x, which is defined

as the number of elements (or blocks) in the sequence:

$$L(x) = |x| \tag{5.8}$$

We can define the *Importance Score* related to the length of a trajectory x as follows:

$$\xi_{\text{length}}(x) = L(x) \tag{5.9}$$

Here, $L(x)$ is the length of the trajectory x.

Why using trajectory length as a score? In sequence-based models, every token in a trajectory contributes a gradient update; therefore, longer trajectories inject more total signal during training and wield greater influence over the learned parameters. By weighting trajectories in proportion to their length, we obtain a direct proxy for their cumulative impact on the model.

iv) Unified Trajectory Importance Score: To evaluate the overall importance of a trajectory $x \in \mathcal{D}_t$, we define a single unified *Importance Score* that combines the three individual scores: Coverage Diversity, Entropy, and Length. This score is computed as a weighted combination of the normalized individual scores:

$$\xi_{\text{unified}}(x) = \alpha \cdot \xi_{\text{coverage}}(x) + \beta \cdot \xi_{\text{entropy}}(x) + \gamma \cdot \xi_{\text{length}}(x) \tag{5.10}$$

Here, the components are defined as follows:

- $\xi_{\text{coverage}}(x)$: the normalized (Equation 5.16) Importance Score based on coverage diversity, as defined in Equation 5.3.
- $\xi_{\text{entropy}}(x)$: the normalized (Equation 5.16) Importance Score based on entropy, as defined in Equation 5.4.

- $\xi_{\text{length}}(x)$: the normalized (Equation 5.16) Importance Score based on trajectory length, as defined in Equation 5.9.

The coefficients α, β, and γ are non-negative weights that allow adjusting the relative contribution of each component to the overall score. These weights can be determined based on the specific application or dataset characteristics, with the condition:

$$\alpha + \beta + \gamma = 1 \tag{5.11}$$

Why having a unified Importance Score? This single Importance Score provides a comprehensive measure of a trajectory's significance by considering its coverage of unique blocks, structural randomness (via entropy), and length. By combining these aspects, it captures both the diversity and complexity of the trajectory, as well as its overall scale, providing a holistic metric for evaluating its contribution to the dataset. In Section 6.4, we discuss how to define a unified version of importance and the method to weight each term within the unified version automatically.

5.1.3 User Level Importance

At the highest level, we determine the importance of each user by aggregating the contributions of all their trajectories. This provides insight into the overall influence of a user's data on the model. By calculating User Level importance through the aggregation of individual trajectory scores, we can capture a holistic view of the user's impact on the dataset. Various aggregation strategies, such as averaging, maximum selection, or median selection, offer flexibility to emphasize different aspects of a user's influence. Additionally, User Level importance such as *user Uniqueness*, help quantify the distinctiveness and contribution of a user's data in comparison to others, shedding light on their unique role in the dataset. We introduce two metrics to quantify a user's contribution: (i) **user uniqueness** and (ii) **entropy-based**

influence. Analogous to the Trajectory Level scores, we subsequently merge these metrics into a single, unified User Level Importance Score.

i) User Uniqueness: User uniqueness is quantified based on the set of blocks in their trajectories that do not appear in the trajectories of any other user. Let $\mathcal{B}_c$ denote the set of all blocks contained in all trajectories of user c, and let $\mathcal{B}_{-c}$ denote the set of all blocks from the trajectories of all users other than c. Then, the uniqueness score for user c is defined as:

$$\text{Uniq}(c) = |\mathcal{B}_c \setminus \mathcal{B}_{-c}| \tag{5.12}$$

where:

- $\mathcal{B}_c = \{b \in \mathcal{T} \mid \mathcal{T} \in \mathcal{D}_t^c\}$
- $\mathcal{B}_{-c} = \left\{b \in \mathcal{T} \;\middle|\; \mathcal{T} \in \mathcal{D}_t^{c'},\ \forall c' \neq c\right\}$

The *Importance Score* of a trajectory x which belongs to user c can then be defined as:

$$\xi_{\text{unique}}(x) = \text{Uniq}(c)$$

Why using user uniqueness as a score? Trajectories that contain location blocks rarely (or never) visited by other users inject highly individual-specific signals into the model, skewing its parameters toward that user's behaviour. When the objective is *machine unlearning*, erasing such distinctive data points is crucial: eliminating a user's unique blocks most effectively retracts their influence while preserving population-level patterns contributed by common routes. The uniqueness-based score therefore serves as a principled proxy for a trajectory's marginal contribution—and, by extension, for the impact its removal will have on the trained model.

ii) User Entropy-Based Influence: To evaluate the randomness or predictability of a user's trajectory set, we aggregate the entropy scores of their individual trajectories. The

User Level entropy importance is defined as the inverse of the aggregated entropy:

$$\xi_{\text{entropy}}(c) = \frac{1}{\sum_{x \in \mathcal{D}_t^c} H(x)} \tag{5.13}$$

Here, $H(x)$ is the entropy of trajectory x, as defined in Equation 5.5. By summing the entropies of all trajectories for a user, we can assess how structured or random their overall trajectory set is, with lower aggregated entropy indicating higher importance. The rationale is analogous to the information-theoretic (entropy) score defined at the Trajectory Level.

iii) Unified User Importance Score: To evaluate a user's overall importance, we define a single unified *User Importance Score*, $\xi(c)$, by combining the previously defined metrics: User Uniqueness and User Entropy-Based Influence. This score is computed as a weighted linear combination:

$$\xi(c) = \eta \cdot \xi_{\text{unique}}(c) + \lambda \cdot \xi_{\text{entropy}}(c) \tag{5.14}$$

Where:

- $\xi_{\text{unique}}(c)$ is the uniqueness of the user's trajectory set, as defined in Equation 5.12.
- $\xi_{\text{entropy}}(c)$ is the user's entropy-based importance, as defined in Equation 5.13.

The coefficients η and λ are non-negative weights that control the relative contribution of each metric. These weights can be adjusted to reflect the emphasis on different aspects of User Level importance, with the condition:

$$\eta + \lambda = 1 \tag{5.15}$$

The Unified User Importance Score offers a comprehensive evaluation of a user's overall contribution to the dataset, combining aspects of their data's uniqueness and structural

randomness. By adjusting the weights, $\xi(c)$ can be tailored to highlight specific facets of user influence in the dataset.

5.1.4 Normalization

There are different ways of defining the *Importance Score*, and we explored some options in Sections 5.1.1, 5.1.2, and 5.1.3. To make the *Importance Score* consistent and comparable across different score criteria, we normalize it using a min-max normalization technique. Min-max normalization ensures that the scores lie within a standardized range, typically between 0 and 1, and can be computed as follows:

$$\xi_{norm} = \frac{\xi - \xi_{min}}{\xi_{max} - \xi_{min}}, \tag{5.16}$$

where ξ_{min} and ξ_{max} are the minimum and maximum values of the raw Importance Score ξ in the dataset, respectively. By applying min-max normalization, the Importance Scores are rescaled proportionally within the range [0, 1], preserving the relative differences between data points while removing the influence of differing scales or units of the raw scores.

This normalization is particularly useful when the raw Importance Scores are derived from different metrics or measures with varying units or scales. It ensures that the resulting scores are dimensionless and comparable, making it easier to interpret and analyze the relative significance across different Importance Scores.

5.2 Teacher-Student Model

The unlearning process utilizes a teacher-student framework. The **Teacher model**, denoted as M_t, is the original model that has been trained on the dataset $\mathcal{D}_t$. The output of this model for a given input x is represented by $M_t(x)$. During the unlearning procedure, the

Teacher model's primary role is to provide reference outputs. Specifically, forward passes are performed on the Teacher model for batches of both the unlearning data $\mathcal{D}_u$ and the remaining data $\mathcal{D}_r$. The outputs generated by M_t for these data batches are then used to calculate the loss function (defined in Equation 5.17), which guides the update of the Student model. Crucially, the weights of the Teacher model (M_t) remain frozen and are not changed throughout the entire unlearning procedure.

The **Student model**, denoted as M_u, begins as an identical copy of the originally trained Teacher model M_t (also trained on $\mathcal{D}_t$). The fundamental distinction between the *student* and the *teacher* lies in their roles and behavior during the unlearning phase. While the Teacher model's weights are fixed, the Student model's weights are actively updated. This update mechanism relies on the loss calculated using the Teacher's outputs, which is then back-propagated through the Student model (M_u) to adjust its weights.

The purpose of this teacher-student interaction is twofold: First, by calculating a loss based on the Teacher's output for the unlearning data $\mathcal{D}_u$ (specifically, using the negative of the standard loss as implied by the unlearning objective), we compel the Student model (M_u) to deviate from the original model's behavior on this specific data. This process guides the Student model to effectively "forget" these targeted samples. Second, by calculating a loss based on the Teacher's output for the remaining data $\mathcal{D}_r$, we reinforce the Student model's ability to perform correctly on the data it is intended to retain. This step is vital to prevent the degradation of the model's accuracy on samples not designated for unlearning.

The loss function used for updating the Student model (as detailed in Equation 5.17) incorporates an *Importance Score* (defined in Equation 5.4) as a multiplicative factor for the loss values derived from the unlearning samples $\mathcal{D}_u$. This score quantifies the importance of an input (e.g., a user trajectory). The rationale for including this factor is to avoid excessively forcing the model to forget information that is common or abundant within the dataset.

For instance, consider a scenario where an unlearning sample pertains to a user's trajectory

that includes driving along a straight stretch of a major highway. This pattern – a straight path on a highway – is likely a very common occurrence within the broader dataset, representing a fundamental driving behavior the model should generally understand. If this common pattern were treated with the same unlearning intensity as a highly unique or anomalous trajectory, the model might be excessively penalized for recognizing and predicting straight highway driving. This could lead to a degradation in its ability to handle other, unrelated instances of highway driving for users whose data is not being unlearned.

To mitigate this, the Importance Score comes into play. For such a common pattern (the straight highway path), its calculated Importance Score would be relatively low. Consequently, when this sample is processed during unlearning, its contribution to the unlearning loss (the signal telling the model to "forget") is dampened by this lower score. The model is still nudged to disassociate the specific unlearned trajectory from its memory, but it's not forced to aggressively unlearn the general concept of "driving straight on a highway."

This nuanced approach, therefore, serves a critical purpose: it helps prevent the unlearning process from inadvertently degrading the model's performance on general, non-unique knowledge that is widely represented in the data. By down-weighting the unlearning impact of common patterns, the system can more effectively focus its efforts on targeting and removing truly unique or sensitive data points for which unlearning is paramount, while preserving the model's valuable learned representations of common scenarios. This ensures that the model remains robust and accurate for the vast majority of data it is expected to handle, even after specific information has been unlearned.

5.3 Loss Function

We assume the output of the *student model*, and the *teacher* for input x are $M_u(x)$, and $M_t(x)$, respectively. Now we can calculate the loss of the *student model* based on the *teacher*

output for input sample x as follows:

$$\mathcal{L}(x) = (1 - \vartheta_x) \cdot L_r(x) - \vartheta_x(e^{\xi_{norm}(x)} - 1) \cdot L_u(x) \tag{5.17}$$

Where ϑ_x indicates whether the input x is in unlearning dataset $\mathcal{D}_u$ or not as Equation 5.18. Also, functions $L_u(x)$ and $L_r(x)$ are defined in Equation 5.19 and Equation 5.20 respectively.

$$\vartheta_x = \begin{cases} 1 & \text{if } x \in \mathcal{D}_u \\ 0 & \text{if } x \in \mathcal{D}_r \end{cases} \tag{5.18}$$

The unlearning loss $L_u(x)$ that makes the model forget is defined as follows:

$$L_u(x) = D_{KL}(M_u(x) \parallel M_t(x)) \tag{5.19}$$

Also we define the remaining loss $L_r(x)$ that makes the model to maintain the performance on the remaining dataset:

$$L_r(x) = c_1 \cdot D_{KL}(M_u(x) \parallel M_t(x)) + c_2 \cdot \mathrm{CE}(\mathbf{y}, M_u(x)) \tag{5.20}$$

The scalars c_1 and c_2 are hyperparameters to adjust the loss values to the same magnitude. The cross-entropy loss for true probability distribution $\mathbf{y}$ and predicted probability distribution $\hat{\mathbf{y}}$ is given by:

$$\mathrm{CE}(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) \tag{5.21}$$

The *Importance Score* $\xi(x_i)$ is defined in definition 5.1 and we normalize it using Equation 5.16. Also, here we are using an exponential weighting scheme for incorporating the the *Importance Score* into the loss function.

Using the exponential term $(e^{\xi_{norm}(x)} - 1)$ as an Importance Score weight in Equation 5.17 is highly beneficial for targeted machine unlearning because it creates a non-linear scaling mechanism that strategically focuses the forgetting process. This exponential factor strongly amplifies the unlearning signal for samples deemed highly important or unique ($\xi_{norm}(x) \gg 0$), ensuring they receive significant pressure to be forgotten by the model. Conversely, for samples with low importance ($\xi_{norm}(x) \approx 0$), the factor approaches zero, effectively dampening the unlearning signal and preventing the model from inadvertently forgetting common knowledge or patterns that should be retained. This selective amplification of critical data removal while protecting general knowledge, combined with the function's smoothness for optimization, makes the exponential scheme an effective approach for precise and efficient unlearning.

Chapter 6

Experimental Evaluation

In this chapter, we present a comprehensive experimental evaluation of our method. First, we list the research questions we aim to explore. Then, we present details of the experimental configuration, datasets employed, the baseline methods, and evaluation metrics. Finally, we present the results and discuss insights and broader impact.

6.1 Overview and Research Questions

In this research, we address the machine unlearning scenario where users request the deletion of their data. Crucially, this involves unlearning the user entirely, meaning we completely remove all trajectory data linked to that specific user upon their request. This is distinct from scenarios involving the deletion of random or isolated trajectory segments. Our "User Deletion" approach ensures that any identifiable traces of the user's movement patterns are fully erased. By unlearning the complete history of trajectories associated with a user, the method aims to prevent the reconstruction of their behavior, thereby minimizing re-identification risks and strengthening user privacy control.

Before presenting our experimental evaluation, we first outline the key research questions

that guide this study. These questions are aimed at understanding how the concept of sample importance can be harnessed to improve the design and performance of machine unlearning techniques. By systematically addressing these questions, we seek to uncover not only the strengths and limitations of our proposed method, but also the broader trade-offs involved in balancing unlearning effectiveness, computational efficiency, and data utility.

- **RQ 1.** In what ways does the proposed method extend or enhance the capabilities of the baseline unlearning approach?
- **RQ 2.** How we can choose a proper Importance Score?
- **RQ 3.** How do different definitions or formulations of Importance Score affect the performance of the unlearning process?
- **RQ 4.** How does the uniform vs targeted sampling affect unlearning effectiveness?
- **RQ 5.** What trade-offs exist between unlearning accuracy and computational cost?

This chapter presents a comprehensive experimental evaluation of our proposed machine unlearning method (TraceHiding) based on Importance Scores. We begin by detailing the experimental setup in Section 6.2, including the datasets used (6.2.3), the models employed (6.2.4), and the baseline methods for comparison (6.5.2). We also discuss about choosing the Importance Scores (Section 6.3) and how to combine them (Section 6.3.) Following this, we present two key experimental studies: unlearning under uniform sampling (6.6.1) and unlearning under targeted sampling (6.6.2). Each experiment is conducted across multiple datasets and model architectures to assess generality (6.6.3, 6.6.4, 6.6.5, and 6.6.6), and includes comparisons to several established baselines. Finally, we summarize the empirical findings and reflect on how they address our research questions in Section 6.7.

6.2 Experimental Setup

To thoroughly evaluate the efficacy and robustness of our proposed unlearning methodology, we design a comprehensive experimental setup that captures various realistic scenarios and challenges. This section details the core components of our experiments, including the specific deletion scenarios considered, the user sampling strategies employed to select data for deletion, the datasets used for evaluation, the diverse trajectory classification models tested, and the metrics used to assess unlearning performance. Together, these components form a rigorous framework to systematically analyze how well unlearning methods perform under different conditions and model architectures.

We begin by defining the deletion scenarios to establish the context of data removal requests. Next, we describe the strategies used to sample users for deletion, highlighting both uniform and targeted approaches. We then introduce the datasets and their characteristics, followed by a presentation of the classification models leveraged to validate our approach across various architectures. Finally, we conclude with the evaluation metrics that quantify both the completeness and efficiency of the unlearning process.

6.2.1 Deletion Scenarios

In this work, we focus specifically on *user deletion*, which entails removing all data associated with an individual user from the model. Unlike approaches that randomly select data points for unlearning, our scenario assumes that a user initiates a deletion request, and consequently, the entire dataset corresponding to that user must be unlearned. This user-centric deletion reflects practical privacy demands and regulatory requirements, such as those outlined by data protection laws, where users have the right to request the complete removal of their personal data.

Although our primary focus is on User Level deletion, the proposed methods can be

naturally extended to handle random or partial deletion scenarios. This flexibility ensures that the framework remains applicable across a range of unlearning tasks, whether targeting specific data points, segments, or users as a whole.

6.2.2 Sampling Strategies for Deletion

We employ two distinct user sampling strategies to select candidates for deletion, each serving a specific purpose in evaluating the robustness of unlearning methods.

The first approach is *uniform sampling*, where users are selected randomly with equal probability. This strategy provides a baseline scenario, ensuring that deletions occur without bias or preference towards any particular user group. Uniform sampling allows us to assess the performance of deletion methods under conditions that simulate arbitrary and unbiased removal requests.

The second approach is *targeted sampling*, designed to sample users proportionally to their aggregated entropy values. Entropy, as a measure of uncertainty or information content, is computed for each user based on their trajectories. By sampling users according to these entropy values, we focus deletion efforts on users contributing higher informational complexity or variability. This strategy enables us to examine how deletion methods perform when faced with more challenging or influential user data, thereby providing insights into their effectiveness under targeted unlearning scenarios.

For each dataset, we conduct experiments using sample sizes of 1%, 5%, 10%, and 20% to evaluate the unlearning performance at varying scales of data removal. To ensure robustness and reliability of the results, each experiment is repeated across 5 independent runs, allowing us to account for variability due to random initialization or sampling. The final reported metrics are averaged over these runs, providing a more stable and statistically meaningful assessment of the unlearning methods. This rigorous evaluation protocol helps to minimize

the influence of random chance and supports confident conclusions about method effectiveness across different deletion magnitudes.

6.2.3 Datasets

We briefly describe each dataset and their statistics in Table 3.1 in Chapter 3. We are using two different types of trajectories: one is 1) the continuous path of users, and the other is 2) the sequence of check-ins that the user recorded. We treat both as sequences of tokens to capture the spatial-temporal patterns of user movement, regardless of the type of trajectory. The datasets are as follows:

Ho-Rome [69] consists of recorded mobility traces of taxis operating in Rome, Italy.

Ho-Geolife [70, 71, 72] consists of recorded mobility traces of individuals, covering a total distance of ∼1.2 million Km.

Ho-NYC [73] consists of Poi check-ins recorded by the phones of several users in the city of New York.

6.2.4 Trajectory Classification Models

We are testing various models for this task. Our goal is to evaluate how well the unlearning process works with each model. By doing this, we can make sure that our proposed method is effective, no matter which model and architecture is used. We conduct our experiments using the following four models: **GRU**, **LSTM**, **BERT**, and **ModernBERT**,

GRU and LSTM are classic recurrent neural network variants designed to capture long-range dependencies in sequential data. BERT and ModernBERT, conversely, are transformer-based architectures that leverage self-attention to achieve state-of-the-art performance across various language tasks. We include these models because they span the major families of sequence modeling, RNN and transformer-based, thereby ensuring our methodology is tested

across a diverse and representative set of architectures. This choice highlights that our proposed methodology is model-agnostic and addresses an orthogonal problem independent of any specific neural architecture.

6.2.5 Evaluation Metrics

In machine unlearning we have two main aspects to evaluate: 1) completeness, which is how completely they can forget data, and 2) timeliness, which is how quickly they can forget. In order to achieve completeness, it is necessary to ensure that when a data sample is deleted, all of its impacts on the model are completely reversed. It basically expresses how consistent a system that has been unlearned is with a system that has been completely retrained. Timeliness in unlearning refers to the speed at which the process of unlearning updates the model in the system compared to the process of retraining.

In the evaluation of unlearning methods we should be aware of Streisand Effect. If the process of unlearning is not done thoroughly or correctly, it might leave traces of the data that was supposed to be forgotten, leading to potential privacy breaches. And it could lead to a situation similar to the Streisand Effect, where trying to hide or minimize the breach only leads to greater attention and scrutiny.

We have discussed some of the evaluation metrics in section 2. For this work we opt for the following evaluation metrics.

Performance on Forget & Remaining Dataset

Accuracy: Measures the overall correctness of the model by dividing the number of correct predictions (both true positives and true negatives) by the total number of cases examined.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.1}$$

We calculate accuracy on different splits of the dataset such as: **(1) Remaining dataset accuracy (RA)**, and **(2) Test dataset accuracy (TA)**. Also we have another accuracy metric which is **(3) Unlearning dataset accuracy (UA)**, defined by $1-$ *accuracy of a model on the unlearning dataset*. This approach is common and has been used in different related works [76, 77, 78].

Membership Inference Attack (MIA)

To evaluate forgetting, we use tools inspired by MIAs (Membership Inference Attacks), such as LiRA (Likelihood Ratio Attack) [79]. MIAs, or membership inference attacks, originated in the field of privacy and security research. Their primary objective is to determine which specific samples were included in the training dataset. If unlearning is successful, the unlearned model will not retain any evidence of the forgotten examples. As a result, MIAs will not be effective. The attacker will be unable to figure out that the forgotten set was originally included in the training set. This approach has been used in the unlearning context [76]. We use the Area Under the ROC Curve (AUC) of the MIA to quantify the effectiveness of unlearning. The MIA AUC measures how well an adversary can distinguish between training and non-training data points using model outputs. A value of 1.0 indicates perfect classification by the attacker, while 0.5 corresponds to random guessing. In the context of unlearning, our objective is to remove information about certain users such that the model no longer retains any identifiable signal from them. Therefore, a lower MIA AUC indicates better forgetting, as it reflects reduced confidence or accuracy in the attacker's ability to infer membership. For a successfully unlearned model, the AUC should be close to 50%, indicating that the attack performs no better than a random guess. We decided to report the difference from a random classifier; the lower the difference, the better the quality of unlearning.

Unlearning Speedup

We are measuring the wall-clock unlearning time to compare the speed of unlearning, we made sure that the running environment is consistent across the runs. We report the speed up relative to the retraining model.

$$\text{Speedup} = \frac{\text{Runtime of Retraining}}{\text{Runtime of the method}} \tag{6.2}$$

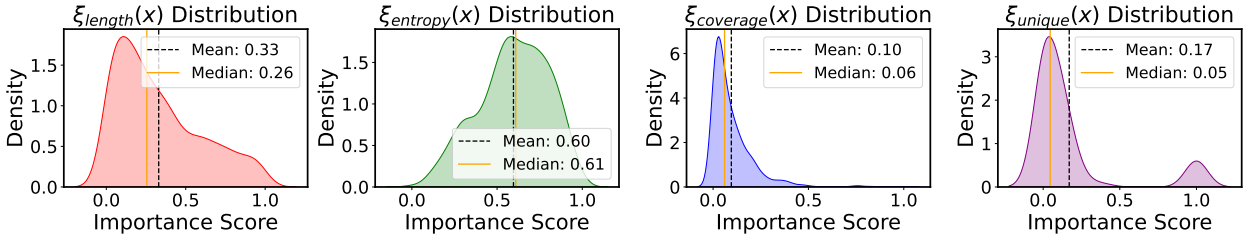
6.3 Choosing the Importance Score

The choice of an Importance Score is critical to effectively remove the influence of specific data samples while maintaining the integrity of the remaining model. Different downstream tasks may necessitate different importance metrics, with the choice often depending on dataset characteristics or model behavior. In our approach, we opted for the following Importance Scores in our experiments.

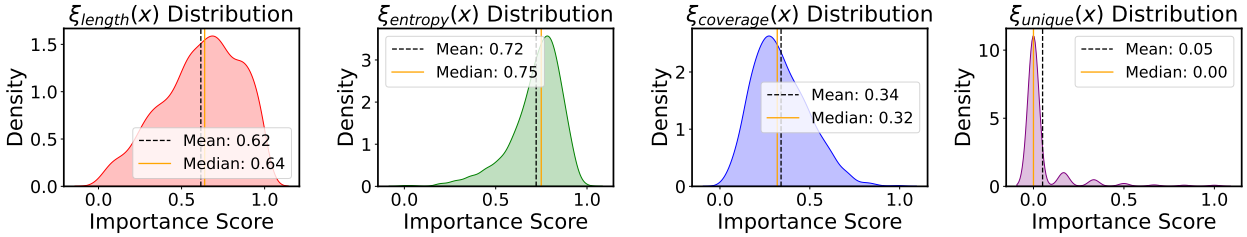
1) Coverage Diversity, 2) Information-theoretic, and 3) Unified Trajectory Importance Score including the length importance. We discuss how we can choose the weighting scheme of the unified score function in Section 6.4. The Importance Score distributions for different datasets are presented in Figure 6.1. Each subplot visualizes the distribution of Importance Scores, with mean and median values marked to indicate central tendencies. The differences in distributions across datasets highlight variations in the significance of these metrics.

The selected Importance Score is designed to reflect each sample's role in influencing the decision boundaries, making it particularly useful for identifying which data points contribute most to model behavior. This approach enables us to prioritize the unlearning of high-impact samples, thus minimizing model degradation after the removal.

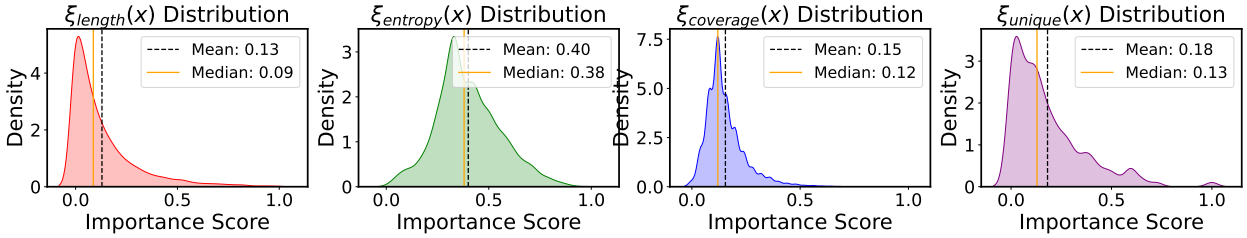
We have compared different Importance Scores within a dataset in Figure 6.2, we can see the positive correlation (upper triangle > 0) between length, entropy, and coverage diversity,



(a) HO-Geolife Dataset Importance Density Plot.



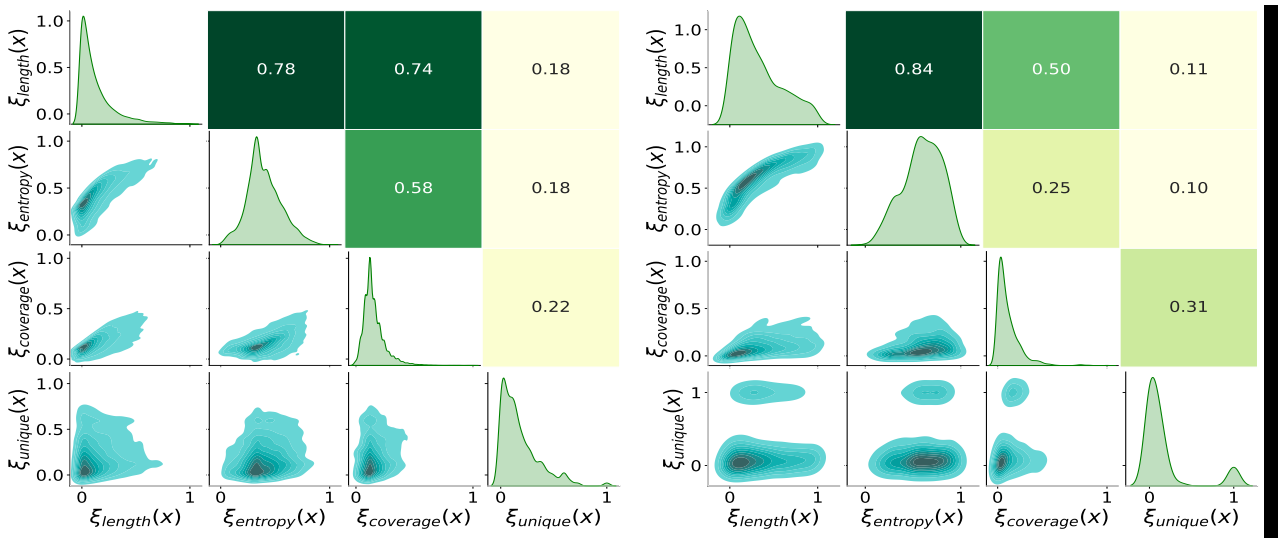
(b) HO-Rome Dataset Importance Density Plot.



(c) HO-NYC Dataset Importance Density Plot.

Figure 6.1: Density plots depicting the distribution of normalized Importance Scores across datasets with different definitions from the left to the right, (1) Length of Trajectory, (2) Entropy, (3) Coverage Diversity, and (4) User Uniqueness. Each plot shows the spread of normalized Importance Scores, with the x-axis representing the normalized score and the y-axis representing the density of occurrences. Mean and median Importance Scores are indicated, highlighting differences in different importance across datasets.

but user uniqueness shows a much weaker correlation with the other three Importance Scores and tends to cluster near zero, as depicted in Figure 6.1. This is due to the strict definition of the uniqueness function in Equation 5.12, which requires that a token (block) appears exclusively in a single user's trajectory to be considered unique. Given the nature of urban mobility, where users often share common pathways, achieving high uniqueness is uncommon. As a result, we have decided not to use user uniqueness in our experiments, as its low values may not contribute meaningfully to the overall importance evaluation. However, we include



(a) Ho-NYC Dataset

(b) Ho-Geolife Dataset

Figure 6.2: Pairwise analysis of the Importance Scores for two datasets. The lower triangle in each plot shows the joint distribution of the Importance Scores, the upper triangle shows the correlations, and the plots in the diagonal show the distribution of the importance itself.

it in our analysis to demonstrate that alternative definitions of importance exist, and this metric could still be useful in specific cases, such as datasets with sparse user distributions or disjoint mobility patterns across different parts of the city.

6.4 Weighting Parameters for Unified Score

We observe a strong positive correlation between the key Importance Scores length, coverage diversity, and entropy, indicating that even if we combine them into a single function, their contributions would not be contradictory. Since these three Importance Scores align well, they provide a consistent way to evaluate importance without introducing conflicting influences. Their strong interdependence suggests that they collectively capture the richness and comprehensiveness of user trajectories, making them reliable measures for our use case. When combining multiple scores into a single Importance Score, it is essential to determine the appropriate weighting parameters. This process can be approached through both **manual**

adjustments tailored to specific downstream tasks and **unsupervised** optimization techniques.

6.4.1 Manual Weight Selection for Downstream Tasks

In many cases, the relative importance of the individual scores may be informed by the requirements of a specific downstream task. For example:

- **Domain Knowledge:** Weights can be assigned manually based on expert knowledge or prior experience regarding the significance of each score.
- **Task-Specific Priorities:** Different tasks may prioritize certain scores over others. For instance, if interpretability is critical, simpler weighting schemes (e.g., equal weights) may be preferred, while performance-oriented tasks may rely on fine-tuned weights informed by data analysis.

6.4.2 Unsupervised Optimization Techniques

Unsupervised methods are particularly useful when there is no explicit ground truth or target variable to guide the weighting process. These techniques aim to optimize certain statistical properties of the unified score. For example:

- **Variance Maximization:** Allocates weights to maximize the spread of the unified score across users or samples, ensuring that it captures as much variability as possible from the underlying data.
- **Entropy Maximization:** Encourages a broad and balanced distribution of the unified score by maximizing its entropy, promoting uniform contributions from the individual scores.

- **Asymmetry and Tailedness Measures:** Incorporates additional statistical measures such as skewness and kurtosis. These metrics ensure that the unified score is not only spread out but also symmetric and devoid of extreme outliers, balancing its statistical properties. As shown in Figure 6.3, the optimization effectively incorporates these metrics by minimizing the objective function in Equation 6.3.

$$\text{Objective} = -(w_1 \cdot V + w_2 \cdot |S| + w_3 \cdot K_e), \tag{6.3}$$

- V: Variance of the user-level mean unified scores (here the function in Equation 5.10 has been used for combined score and visualization in Figure 6.3, which measures the spread of scores across users. Maximizing V ensures greater diversity in the distribution.

- S: Skewness of the user-level mean unified scores, quantifying the asymmetry of the distribution:

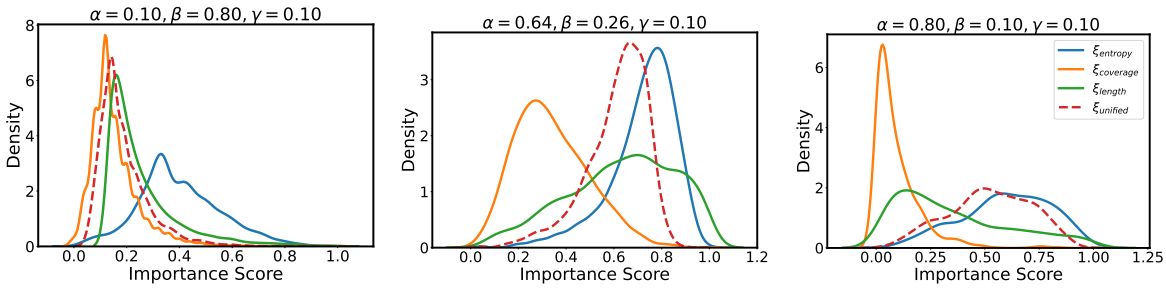
  - $S > 0$: Right-skewed (longer tail on the right).
  - $S < 0$: Left-skewed (longer tail on the left).

  The function minimizes the absolute skewness ($|S|$) to encourage a symmetric distribution.

- K_e: Excess kurtosis of the user-level mean unified scores, calculated as kurtosis minus 3, to center the value around 0 for a normal distribution:

  - $K_e > 0$: Heavier tails (more outliers).
  - $K_e < 0$: Lighter tails (fewer outliers).

  Minimizing K_e reduces the impact of extreme outliers in the distribution.



(a) HO-NYC Dataset (b) HO-Rome Dataset (c) HO-Geolife Dataset

Figure 6.3: The figure illustrates the density distribution of individual scores and the optimized unified score for different datasets. The optimization incorporates higher-order shape descriptors (skewness and kurtosis), ensuring that the composite score is spread out, symmetric, and robust to outliers.

- w_1, w_2, w_3: Fixed weights for variance, skewness, and kurtosis, respectively, which balance the contribution of each term in the optimization, Weh have chosen the default and trivial values $w_1 = \frac{1}{3}, w_2 = \frac{1}{3}, w_3 = \frac{1}{3}$.

By combining unsupervised techniques with manual adjustments, it is possible to create a flexible and robust method for determining the weighting parameters. This ensures that the unified score is both data-driven and aligned with the specific goals of the application.

6.5 Unlearning Methods

6.5.1 TraceHiding Variants

We introduce variants of our method, TraceHiding, which arise from different parameter choices, such as the Importance Score functions used for unlearning.

TraceHiding (Ent). In this variant, we employ the entropy-based importance function, given by Equation 5.4, to determine the significance of individual samples, which is a guide for unlearning.

TraceHiding (C.D.). In this variant, we utilize the coverage diversity-based

importance function, defined by Equation 5.3, to assess sample significance, thereby informing the method about importance of samples chosen for unlearning.

TraceHiding (Uni.). In this variant, we employ a unified equation to calculate the Importance Score (Equation 5.10). The selection of appropriate weights within this equation is discussed in Section 6.4.

6.5.2 Baselines

We use the following baselines for comparative analysis with our method. These methods are well-known in machine unlearning for classification and have appeared in most previous studies.

NegGrad [80]. This method applies gradient ascent exclusively to $\mathcal{D}_u$ (or equivalently, gradient descent on $\mathcal{D}_u$ with the gradient negated, which is why it is often called NegGrad). The underlying idea is to try to "delete" $\mathcal{D}_u$ by maximizing the loss on that data, effectively aiming to "reverse" the learning process the network previously performed to learn that data.

NegGrad+ [80]. NegGrad might reduce the performance on the remaining dataset $\mathcal{D}_r$ because it lacks motivation to preserve useful information while conducting gradient ascent. Essentially, if the remaining data is similar to the unlearning dataset, NegGrad's goal of maximizing the loss on $\mathcal{D}_u$ could inadvertently increase the loss on parts of $\mathcal{D}_r$, which is undesirable. To tackle this issue, it employs a more robust approach that performs gradient ascent on the unlearning dataset (like NegGrad) and gradient descent on the remaining dataset (similar to Fine-tuning). This method aims to achieve a balance by maximizing the loss on $\mathcal{D}_u$ while minimizing it on $\mathcal{D}_r$.

SCRUB [43]. SCRUB uses a teacher-student approach. The teacher is the original model, trained on the full dataset. The student starts with the teacher's knowledge and then learns to keep only the information relevant to the remaining dataset. The student has

two main goals: to become similar to the teacher for the remaining dataset and to become different from the teacher for the dataset to be forgotten. This way, information about the unlearning dataset is effectively removed. SCRUB aims to balance removing information about the unlearning dataset while keeping information about the remaining dataset. It does this through both positive and negative knowledge transfer from the original model.

FINETUNING. This method is finetuning on the remaining dataset $\mathcal{D}_r$. This method focuses on the remaining data, revisiting it to ensure we prioritize it over the data that hasn't undergone fine-tuning. By doing so, we concentrate on improving the parts that need more attention and forgetting the data that doesn't participate in fine-tuning.

BAD-T [42]. This work introduces a novel machine unlearning technique through a teacher-student framework. The core of this approach involves two distinct types of teachers: a competent teacher, which is a fully trained model that encapsulates the complete knowledge from the training data, and an incompetent teacher, which is a randomly initialized model that lacks any meaningful knowledge. The student model, initially carrying the knowledge of the competent teacher, undergoes a selective unlearning process guided by these two teachers. This method tries to achieve unlearning by minimizing a loss on the student's outputs and those of the incompetent teacher for the unlearning data, effectively erasing this information. Simultaneously, the loss is minimized between the student's outputs and those of the competent teacher for the retain data, preserving essential knowledge.

6.6 Results and Analysis

This section distills the extensive appendix tables (Tables A.1–A.6) into concise comparisons that answer our five research questions (**RQ1**–**RQ5**). We separate the discussion into *uniform* and *targeted* deletion scenarios, then analyze cross-dataset robustness, model sensitivity, and the influence of the Importance Score.

Table 6.1: The average rank of Unlearning Accuracy across different datasets and models under **uniform** sampling, where a lower rank indicates better performance. Results are shown for various proportions of data samples selected for removal.

Method	1 %	5 %	10 %	20 %
TraceHiding (Ent.)	2.88	3.67	2.25	2.25
NegGrad	3.25	2.62	2.92	3.08
TraceHiding (Uni.)	3.62	3.58	3.67	3.33
TraceHiding (C.D.)	4.08	3.75	3.33	3.58
NegGrad+	3.75	3.79	4.58	4.75
SCRUB	5.21	5.12	5.71	4.92
Bad-T	5.92	6.25	5.75	6.38
Finetuning	7.29	7.21	7.79	7.71

6.6.1 Unlearning under Uniform Sampling

Uniform sampling removes a random fraction of users (1 %, 5 %, 10 %, 20 %). Table 6.1 shows the *average rank* (lower = better) of each method's **Unlearning Accuracy (UA)** across all three datasets and four model architectures.

Across all removal sample sizes, the average-rank analysis shows a clear hierarchy: TraceHiding with entropy importance is the most reliable unlearning strategy, finishing first overall and never worse than third in any column, while vanilla NegGrad is the strongest of the "classical" baselines. The two other TraceHiding variants (unified and coverage diversity) form a second tier, roughly matching NegGrad at moderate fractions but slipping slightly when only 1% of users are erased. NegGrad+, SCRUB, and Bad-T occupy a third tier whose relative ordering fluctuates with the amount of data removed, indicating that these methods are more sensitive to the sampling regime. Finally, plain Finetuning consistently ranks last by a wide margin, reinforcing that dedicated techniques are essential for dependable user-level unlearning.

Table 6.2: Change in MIA AUC (*targeted* – *uniform*) at 10 % unlearning (mean over datasets/models; ↓ better).

Method	MIA AUC (Targeted)	MIA AUC (Uniform)	Δ
TraceHiding (Ent.)	31.06	36.28	-5.22
TraceHiding (C.D.)	31.06	35.14	-4.08
NegGrad	34.90	36.40	-1.50
Finetuning	37.23	36.71	0.52
NegGrad+	35.19	34.05	1.14
Bad-T	36.94	35.23	1.71
SCRUB	37.83	34.87	2.96
TraceHiding (Uni.)	37.81	34.38	3.43

6.6.2 Unlearning under Targeted Sampling

Targeted sampling removes the *most important* users, ranked by entropy. Table 6.2 presents the change in MIA AUC (relative to uniform sampling) under a 10% deletion scenario. A lower MIA AUC indicates better forgetting, meaning the attack model is less able to distinguish the unlearned dataset.

Since we are removing high-information users, we expect models to struggle more with unlearning in this setting. The metric reported in Table 6.2 allows us to compare the robustness of different unlearning methods under more adversarial deletion conditions.

Table 6.2 demonstrates that the TraceHiding variants perform best under targeted sampling. Specifically, TraceHiding (Ent.) shows the largest reduction in MIA AUC with a Δ of -5.22%, indicating superior forgetting when high-information users are removed. TraceHiding (C.D.) also performs well with a -4.08% change.

NegGrad shows a modest improvement (-1.50%), while Finetuning slightly worsens under targeted sampling, with a positive change of 0.52%. NegGrad+, Bad-T, SCRUB, and TraceHiding (Uni.) all have positive Δ values, meaning their performance degrades under targeted sampling. SCRUB and TraceHiding (Uni.) in particular suffer the most, with increases of 2.96% and 3.43% in MIA AUC, respectively. This suggests that these

Table 6.3: Mean Unlearning Accuracy (UA) and Test Accuracy (TA) at 10 % *uniform* deletion, averaged over four models per dataset. Methods are ranked by their average distance to the *Retraining* baseline (gold standard) on both UA and TA. The closest method is shown in **bold**; the second closest is underlined. Runtime is expressed as a multiplicative factor with respect to full retraining.

Method	HO-Rome		HO-Geolife		HO-NYC		Speedup
	UA	TA	UA	TA	UA	TA	
Retraining	100.00	18.82	100.00	45.07	100.00	70.81	1×
TraceHiding (Ent.)	88.31	**19.38**	**68.03**	57.37	**62.89**	73.71	10.01×
TraceHiding (C.D.)	85.17	20.72	59.58	59.02	55.75	75.38	10.03×
TraceHiding (Uni.)	83.08	20.80	55.31	59.31	60.24	74.37	9.81×
NegGrad+	88.01	20.24	53.73	60.05	47.47	**72.28**	12.62×
NegGrad	**92.64**	13.72	59.12	**57.10**	51.20	54.99	13.81×
SCRUB	80.40	21.25	47.26	60.16	52.44	73.89	9.51×
Bad-T	78.60	21.98	51.02	58.75	39.27	77.02	8.07×
Finetuning	76.57	21.88	36.56	63.13	10.58	81.18	5.21×

methods are more sensitive to the deletion of high-information users and less effective in such scenarios. Notably, while TraceHiding (Uni.) is a variant of the trace hiding approach, it performs poorly under targeted sampling, suggesting that the unified importance score may be less effective at identifying and mitigating the impact of high-information users in adversarial deletion scenarios. Overall, targeted unlearning stresses the unlearning methods differently, revealing TraceHiding variants, especially the entropy-based one, as the most resilient to adversarial deletions.

6.6.3 Cross-Dataset Robustness

Table 6.3 reports the mean Unlearning Accuracy (UA) and Test Accuracy (TA) obtained after deleting *10 %* of the training data *per dataset*. Across all three datasets, the relative ordering of the methods is stable: the proposed TraceHiding variants remain among the top performers, with NegGrad providing the strongest conventional baseline.

TraceHiding, especially the entropy variant, stands out as the method that hugs the retraining gold-standard most tightly, posting the smallest average gap on both unlearning accuracy and test accuracy while still running about ten times faster. By contrast, NegGrad and NegGrad+ variant lean toward preserving generalization at the expense of Unlearning Accuracy, and Finetuning drifts furthest from retraining, particularly on HO-NYC where its UA collapses—in short, TraceHiding (Ent.) is the closest proxy to full retraining across all datasets and metrics. For practitioners, this means that when fidelity to the retrained model matters more than raw speed, TraceHiding (Ent.) should be the default, with NegGrad+ as a secondary option for a larger speed-up.

6.6.4 Effect of the Deletion Sample Size

Our proposed TraceHiding (Ent.) method demonstrates competitive and often superior performance across various unlearning metrics, particularly in maintaining model utility after the unlearning process. As shown in Table 6.4 and Figure 6.4, TraceHiding (Ent.) consistently achieves higher Unlearning Accuracy (UA), Remaining Accuracy (RA), and Test Accuracy (TA) for the HO-Geolife dataset, and often for HO-Rome and HO-NYC as well. The bolded values indicate that our method's performance closely aligns with the gold-standard retraining-from-scratch approach, signifying effective removal of influence from the unlearned data while preserving the model's generalization capabilities on the remaining dataset. For instance, in HO-Rome, TraceHiding (Ent.) shows better UA and TA across multiple unlearning percentages, indicating a robust ability to forget specific data points without compromising overall model performance. This strong utility preservation is a critical advantage for real-world applications where unlearning must not severely degrade the model's predictive power.

A comparison with NegGrad+ reveals specific trade-offs, particularly concerning

Table 6.4: A quantitative comparison of four metrics across the datasets and varying sample sizes (1%, 5%, 10%, 20%) for two strong candidates for unlearning method: TraceHiding (Ent.) and NegGrad+. The bolded values indicate the performance closer to a retrained model, signifying better outcomes for each respective metric.

Dataset	Metric	TraceHiding (Ent.)				NegGrad+			
		1 %	5 %	10 %	20 %	1 %	5 %	10 %	20 %
HO-Rome	UA	82.94	**86.51**	**88.31**	**90.47**	**86.41**	86.05	88.01	88.76
	RA	38.39	35.39	32.17	28.77	**39.81**	**37.79**	**34.98**	**30.26**
	TA	**22.06**	**20.93**	**19.38**	**16.61**	22.65	21.95	20.24	17.58
	MIA	31.48	**22.56**	18.79	**14.24**	**27.37**	24.12	**15.77**	18.40
HO-Geolife	UA	**74.19**	**64.62**	**68.03**	**65.54**	72.76	63.64	53.73	54.00
	RA	**81.88**	**81.35**	**79.62**	**79.61**	83.04	83.65	82.43	81.51
	TA	**61.32**	**61.74**	**57.37**	**55.41**	61.89	63.13	60.05	58.72
	MIA	47.61	43.44	44.63	**42.86**	**41.44**	**39.18**	**41.22**	44.47
HO-NYC	UA	**46.24**	**44.98**	**62.89**	**69.09**	31.44	37.57	47.47	45.93
	RA	94.40	**93.81**	**93.01**	92.72	**94.51**	92.45	88.94	**87.42**
	TA	**79.68**	78.02	73.71	68.93	80.00	**77.24**	**72.28**	**68.74**
	MIA	49.71	48.02	45.41	42.96	**49.66**	**47.94**	**45.17**	**41.87**

membership inference attack (MIA) resilience (Figure 6.4d) and computational speedup (Table 6.5.) While NegGrad+ often provides superior protection against membership inference attacks, evident from its lower MIA AUC values on HO-Geolife and HO-NYC, the speedup analysis shows a varied picture. Although NegGrad+ can achieve significantly speedups at the smallest unlearning percentage (1%) across all datasets (and at 5% for HO-Geolife), TraceHiding (Ent.) frequently demonstrates superior speedups when unlearning a larger proportion of data (5%, 10%, and 20% for HO-Rome and HO-NYC). This indicates that for scenarios involving the removal of a substantial portion of the training data, TraceHiding (Ent.) can be the more computationally efficient choice. The selection between TraceHiding (Ent.) and NegGrad+ thus depends on the specific priorities of the application: prioritizing robust utility preservation and faster unlearning for larger deletion requests might favor TraceHiding (Ent.), whereas maximizing privacy

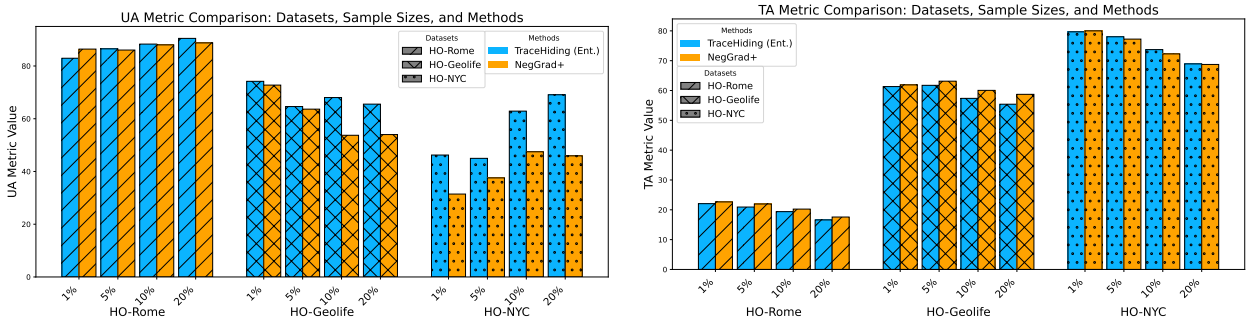
Table 6.5: Comparison of speedup (× faster than retraining) achieved by two strong unlearning candidates, TRACEHIDING (ENT.) and NEGGRAD+, across three datasets and varying sample sizes (1%, 5%, 10%, 20%). Bold values highlight the higher speedup between methods for a given configuration.

Dataset	Method	1%	5%	10%	20%
HO-Rome	TRACEHIDING (ENT.)	15.6×	**12.8**×	**10.6**×	**10.7**×
	NEGGRAD+	**15.7**×	11.9×	9.2×	9.1×
HO-Geolife	TRACEHIDING (ENT.)	5.0×	5.7×	5.5×	**4.0**×
	NEGGRAD+	**7.8**×	**12.0**×	5.5×	3.8×
HO-NYC	TRACEHIDING (ENT.)	11.9×	**14.8**×	**16.1**×	**10.1**×
	NEGGRAD+	**21.6**×	13.1×	14.9×	10.0×

guarantees against membership inference and achieving quicker unlearning for very small deletion requests might lean towards NEGGRAD+.

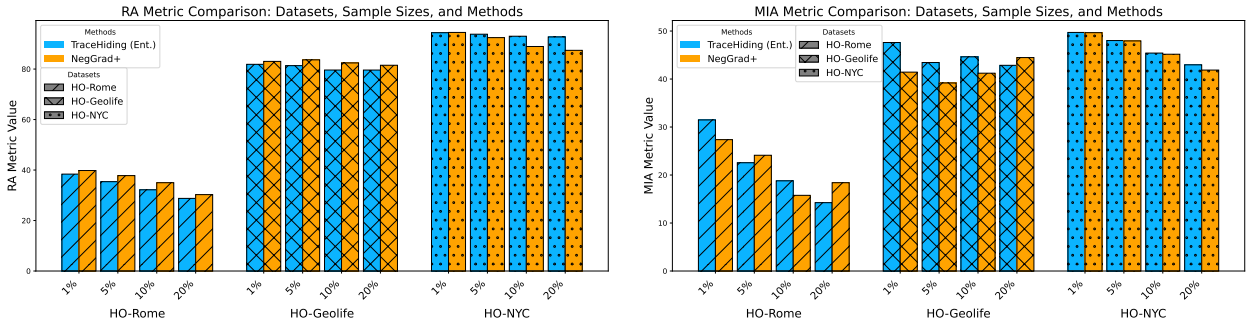
The quantitative analysis of unlearning methods, TRACEHIDING (ENT.) and NEGGRAD+, presented in Table 6.4 reveals distinct strengths and weaknesses across various datasets and unlearning sample sizes, highlighting the nuanced trade-offs inherent in machine unlearning. TRACEHIDING (ENT.) consistently demonstrated superior performance in Unlearning Accuracy (UA), particularly notable in the HO-Geolife and HO-NYC datasets where it was predominantly closer to the gold-standard retrained model. Furthermore, for datasets like HO-Rome and HO-Geolife, TRACEHIDING (ENT.) generally maintained better Test Accuracy (TA), suggesting its efficacy in preserving the model's overall generalization capabilities despite data removal. This indicates that TRACEHIDING (ENT.) is a strong contender when the primary objective is comprehensive forgetting of specific data points and maintaining broad model utility.

Conversely, NEGGRAD+ exhibited an advantage in preserving the model's accuracy on remaining data in HO-Rome dataset, as evidenced by its frequently bolded Remaining Accuracy (RA) values across the sample sizes. More critically, NEGGRAD+ consistently outperformed TRACEHIDING (ENT.) in mitigating Membership Inference Attacks (MIA),



(a) Unlearning Accuracy (UA)

(b) Test Accuracy (TA)



(c) Remaining Accuracy (RA)

(d) Membership Inference Attack (MIA) AUC

Figure 6.4: Comparative Analysis of Remaining Accuracy, Test Accuracy, Unlearning Accuracy, and Membership Inference Attack AUC.

achieving lower AUC scores, particularly prominent in the HO-Geolife and HO-NYC datasets. This robust privacy protection, coupled with its ability to retain performance on the retained data, positions NegGrad+ as a more suitable method when data privacy and the integrity of the retained knowledge are paramount. The findings underscore that the optimal unlearning strategy is not universal, but rather contingent on the unlearning objectives, be it maximizing forgetting and generalization or prioritizing utility preservation and privacy guarantees.

6.6.5 Model Architecture Sensitivity

The analysis of Unlearning Accuracy at 10% deletion (Table 6.6) reveals significant trends regarding unlearning effectiveness across different model architectures and methods. Notably,

Table 6.6: Comparative Unlearning Accuracy at 10% Deletion for RNN and Transformer Architectures under Various Unlearning Methods (Uniform Sampling). Best and second-best accuracies for each model are indicated by bold and underline, respectively.

Method	GRU	LSTM	BERT	ModernBERT
BAD-T	51.57	47.52	65.62	60.47
FINETUNING	46.08	44.03	35.49	39.34
NEGGRAD	57.88	**56.83**	<u>92.46</u>	63.43
NEGGRAD+	57.23	53.20	81.25	60.62
SCRUB	54.09	50.19	79.96	55.91
TRACEHIDING (ENT.)	**61.39**	<u>56.07</u>	**93.48**	**81.37**
TRACEHIDING (C.D.)	<u>60.52</u>	55.62	83.74	67.45
TRACEHIDING (UNI.)	59.90	53.15	81.44	<u>70.33</u>

Transformer-based models (BERT and ModernBERT) consistently achieve higher unlearning accuracies compared to RNN-based models (GRU and LSTM), suggesting their inherent architecture may be more amenable to effective data removal. The TRACEHIDING (ENT.) method demonstrates remarkable robustness and transferability, emerging as the best performer for GRU, BERT, and ModernBERT, and the second-best for LSTM, highlighting its broad applicability. Similarly, NEGGRAD exhibits strong performance, particularly for LSTM and BERT, underscoring its efficacy in certain contexts. Conversely, simpler approaches like FINETUNING consistently yield the lowest unlearning accuracies, indicating their inadequacy for true data unlearning. These findings suggest that advanced unlearning methodologies, especially those leveraging information-theoretic principles like TRACEHIDING (ENT.), are crucial for achieving high unlearning fidelity across diverse neural network architectures.

6.6.6 Effect of the Importance Score

As summarized in Table 6.7, the *Entropy* variant most faithfully imitates full retraining on the HO-Geolife and HO-NYC datasets. Its Unlearning Accuracy (UA) reaches 68.0 % and 62.9 %, respectively, substantially closer to the 100 % the gold standard than either *Coverage*

Table 6.7: TraceHiding variant comparison (10 %, uniform sampling).

Dataset	**Variant**	UA	RA	TA	MIA
HO-Geolife	Coverage Diversity	59.58	82.26	59.02	43.77
	Entropy	**68.03**	**79.62**	**57.37**	44.63
	Unified	55.31	82.26	59.31	**43.68**
	Retraining	100.00	62.71	45.07	29.32
HO-NYC	Coverage Diversity	55.75	94.14	75.38	**45.26**
	Entropy	**62.89**	**93.01**	**73.71**	45.41
	Unified	60.24	93.71	74.37	45.32
	Retraining	100.00	91.77	70.81	36.28
HO-Rome	Coverage Diversity	85.17	**36.37**	20.72	16.38
	Entropy	**88.31**	32.17	**19.38**	18.79
	Unified	83.08	36.09	20.80	**14.14**
	Retraining	100.00	37.64	18.82	9.30

Diversity or *Unified*. At the same time, Remaining Accuracy (RA) and Test Accuracy (TA) remain close to the Retraining on HO-Geolife and HO-NYC. This comes at a privacy cost: Membership-Inference Attack (MIA) success rates climb by roughly 15 percentage points. In other words, *Entropy* offers the best fidelity to retraining on learning-centric metrics (UA and TA) but weakens the privacy guarantees that retraining provides. Note that all of the numbers for MIA are close together and the differences are negligible in most cases for different variants.

6.7 Summary of Findings

Our comprehensive experimental evaluation demonstrates that the proposed TraceHiding method, particularly when utilizing an entropy-based Importance Score (TraceHiding (Ent.)), significantly enhances unlearning capabilities compared to baseline approaches (**RQ1**). Across uniform sampling scenarios (Section 6.6.1), TraceHiding (Ent.) consistently ranked as a top performer in Unlearning Accuracy (UA) across various datasets,

models, and deletion sample sizes, outperforming classical baselines like NegGrad and significantly surpassing simple Finetuning. This superiority was further confirmed by its strong performance in cross-dataset robustness checks (Section 6.6.3), where it most closely approximated the gold-standard retraining results in both UA and Test Accuracy (TA) while offering substantial speedups. The method also showed robust UA across different model architectures (Section 6.6.5).

The choice and formulation of the Importance Score are crucial for unlearning performance (**RQ2**, **RQ3**), as detailed in Sections 6.3 and 6.4, we analyzed the results of the different formulation in Section 6.6.6. The *Entropy* variant of TraceHiding generally provided the best fidelity to full retraining concerning UA and TA, particularly on the HO-Geolife and HO-NYC datasets, making it a strong candidate when these metrics are paramount. While other variants like *Unified* or *Coverage Diversity* sometimes offered marginally better MIA resilience, the *Entropy* score struck a good balance. Regarding the impact of deletion strategies (**RQ4**), our analysis of targeted sampling (Section 6.6.2) revealed that TraceHiding (Ent.) and TraceHiding (C.D.) are more resilient to the removal of high-information users, showing improved or less degraded MIA AUC compared to uniform sampling, unlike several other methods whose forgetting effectiveness diminished significantly under such adversarial conditions.

Finally, the experiments illuminated the trade-offs between unlearning effectiveness and computational cost (**RQ5**). TraceHiding (Ent.) generally provided UA and TA closest to retraining while being around 10 times faster (Table 6.3). A detailed comparison with NegGrad+ (Section 6.6.4) highlighted nuanced trade-offs: TraceHiding (Ent.) often excelled in UA, TA, and speedup for larger deletion percentages (5-20%), while NegGrad+ showed advantages in MIA resilience and sometimes speed for very small (1%) deletion requests. These findings underscore that the optimal unlearning strategy depends on specific priorities, such as the desired level of forgetting, utility preservation, privacy against MIA,

the nature of data deletion (uniform vs. targeted), and computational constraints.

Chapter 7

Conclusion

This thesis addressed the challenge of machine unlearning in the context of trajectory data, specifically focusing on the scenario where users request the complete deletion of their data to protect their privacy. We introduced TraceHiding, a novel unlearning methodology that enhances existing teacher-student frameworks by incorporating data-driven Importance Scores. The core idea was to selectively guide the unlearning process by quantifying the significance of individual data samples, thereby aiming for a more efficient and effective removal of user-specific information while preserving overall model utility.

7.1 Summary and Contributions

Our research began by formalizing the problem of machine unlearning for trajectory classification tasks, specifically Trajectory-User Linking (TUL). We then detailed the TraceHiding algorithmic framework, which leverages a teacher-student architecture where the student model is trained to forget specific user data based on a carefully designed loss function. A key innovation was the integration of Importance Scores (such as those based on trajectory length, coverage diversity, and entropy) into this loss function to modulate the unlearning

process. We explored various methods for defining and unifying these Importance Scores.

The comprehensive experimental evaluation yielded several key findings:

- **RQ1: Enhanced Capabilities:** The TraceHiding method, particularly the entropy-based variant (TraceHiding (Ent.)), demonstrated superior or competitive performance in Unlearning Accuracy (UA) compared to baseline methods like NegGrad, NegGrad+, SCRUB, Bad-T, and Finetuning across various datasets and model architectures under uniform user deletion scenarios. It often provided the closest approximation to full retraining in terms of UA and Test Accuracy (TA) while being significantly faster.

- **RQ2 & RQ3: Importance Score Impact:** The choice and formulation of the Importance Score were shown to be critical. The entropy-based Importance Score, TraceHiding (Ent.), generally offered the best balance in achieving high UA and maintaining TA, closely mirroring retraining results, especially for datasets like HO-Geolife and HO-NYC. Other scores, like the unified score or coverage diversity, sometimes provided slight advantages in MIA resilience, though differences in MIA were often negligible among TraceHiding variants. The information in Section 6.3 and analysis in Section 6.6.6 detailed how different formulations affected performance.

- **RQ4: Sampling Strategy Effects:** When evaluating unlearning under targeted sampling, TraceHiding (Ent.) and TraceHiding (C.D.) proved more resilient, showing better forgetting (lower or improved MIA AUC) compared to uniform sampling. This contrasted with several other methods whose effectiveness diminished under such adversarial deletion conditions.

- **RQ5: Unlearning Accuracy vs. Computational Cost Trade-offs:** TraceHiding methods, particularly TraceHiding (Ent.), generally achieved a good balance

by providing UA and TA close to retraining but with approximately 10x speedup. Compared to NegGrad+, TraceHiding (Ent.) often excelled in UA, TA, and speedup for larger deletion percentages, while NegGrad+ sometimes offered better MIA resilience and speed for very small deletion requests (e.g., 1,%).

Furthermore, our experiments demonstrated that Transformer-based models (BERT, ModernBERT) were generally more amenable to unlearning than RNN-based models (GRU, LSTM), achieving higher unlearning accuracies. TraceHiding (Ent.) showed robustness across these different architectures.

Moreover, TraceHiding is designed with scalability in mind. Since the Importance Scores are computed solely from static characteristics of the training data (e.g., entropy, trajectory length, and coverage diversity), they can be precomputed during data preprocessing and stored efficiently. In settings involving large training data, these scores typically do not need to be recomputed unless there is a significant shift in the data distribution. Instead, the unlearning process can retrieve them on demand using well-established data structures, such as hash maps or tree-based indices, based on trajectory or user identifiers. As a result, the scalability challenge shifts from computational overhead to efficient data retrieval and storage, a well-understood problem in systems and database design.

The main contributions of this thesis are:

1. **TraceHiding Algorithmic Framework.** We presented the first importance-aware machine-unlearning pipeline for Trajectory-User Linking (TUL), showing that targeted forgetting can be achieved without full retraining.

2. **Hierarchical Importance Scoring.** A novel, data-driven hierarchy of token-, trajectory-, and user-level scores pinpoints the samples whose removal maximally protects privacy while minimally harming utility.

3. **Teacher–Student Distillation.** Our loss function couples importance-weighted knowledge transfer with a contrastive term, suppressing sensitive representations and reinforcing retained knowledge.

4. **Comprehensive Empirical Validation.** Experiments on the HO-Rome, HO-GeoLife, and HO-NYC datasets show that TraceHiding reduces deletion time, boosts unlearning fidelity against strong baselines.

7.2 Future Work

While TraceHiding has shown promising results in trajectory-based unlearning, several directions remain open for future research. These works explore extensions to new tasks, richer data representations, real-time scenarios, and broader social considerations such as fairness and regulatory compliance.

7.2.1 Beyond TUL: Other Predictive Tasks

Future work could explore how the TraceHiding framework generalizes to other location-based generative or predictive tasks such as next-location prediction, transportation mode inference, or mobility anomaly detection. These tasks differ in their learning objectives and may require adapting the loss function or importance metrics. Testing across such use cases can offer deeper insight into the versatility and robustness of importance-aware unlearning.

7.2.2 Richer Importance Signals

The current implementation of importance scores focuses on entropy, length, and diversity. Future research might incorporate semantic context, such as Points of Interest (POI) types, user demographics, or even textual annotations, to better estimate the privacy sensitivity of

samples. Moreover, integrating gradient-based influence functions could help capture a more causal relationship between individual data points and model predictions, enabling even finer control during unlearning.

7.2.3 Streaming and Continual Settings

Deploying unlearning methods like TraceHiding in real-world applications requires addressing the challenges of streaming and continual learning. In such settings, user trajectories arrive in real time, necessitating fast, memory-efficient updates and the ability to handle concept drift. Maintaining unlearning guarantees under resource-constrained or dynamic environments will require new algorithmic strategies and optimizations.

7.2.4 Fairness and Compliance Analysis

Finally, any practical deployment of machine unlearning must ensure that the process does not inadvertently introduce biases or disproportionately affect certain user groups. Future research should involve fairness audits to monitor group-level performance shifts post-unlearning. Additionally, aligning unlearning procedures with evolving legal frameworks such as the GDPR and CCPA is essential for real-world applicability in sensitive domains like healthcare or finance.

Bibliography

[1] Yaron Kanza, Balachander Krishnamurthy, and Divesh Srivastava. "A Geospatial Perspective on Data Ownership, the Right to be Forgotten, Copyrights, and Plagiarism in Generative AI". In: *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*. SIGSPATIAL '24. Atlanta, GA, USA: Association for Computing Machinery, 2024, pp. 477–480. ISBN: 9798400711077. DOI: 10.1145/3678717.3691269. URL: https://doi.org/10.1145/3678717.3691269.

[2] General Data Protection Regulation. *Art. 17 GDPR–Right to erasure ('right to be forgotten')*. 2018.

[3] Ali Faraji, Jing Li, Gian Alix, Mahmoud Alsaeed, Nina Yanin, Amirhossein Nadiri, and Manos Papagelis. "Point2Hex: Higher-order Mobility Flow Data and Resources". In: *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*. SIGSPATIAL '23. Hamburg, Germany: Association for Computing Machinery, 2023. DOI: 10.1145/3589132.3625619. URL: https://doi.org/10.1145/3589132.3625619.

[4] Ali Faraji and Manos Papagelis. "TraceHiding: An algorithmic framework for machine unlearning in mobility data". In: *ACM Transactions on Spatial Algorithms and Systems* (2025).

[5] Yu Zheng. "Trajectory data mining: an overview". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 6.3 (2015), pp. 1–41.

[6] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. "Spatio-temporal data mining: A survey of problems and methods". In: *ACM Computing Surveys (CSUR)* 51.4 (2018), pp. 1–41.

[7] Ali Hamdi, Khaled Shaban, Abdelkarim Erradi, Amr Mohamed, Shakila Khan Rumi, and Flora D. Salim. "Spatiotemporal data mining: a survey on challenges and open problems". In: *Artificial Intelligence Review* 55.2 (Apr. 2021), pp. 1441–1488. DOI: 10.1007/s10462-021-09994-y. URL: https://doi.org/10.1007/s10462-021-09994-y.

[8] Fazel Arasteh, Soroush SheikhGarGar, and Manos Papagelis. "Network-aware multi-agent reinforcement learning for the vehicle navigation problem". In: *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 2022, pp. 1–4.

[9] Ali Nematichari, Tilemachos Pechlivanoglou, and Manos Papagelis. "Evaluating and forecasting the operational performance of road intersections". In: *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 2022, pp. 1–12.

[10] Abdullah Sawas, Abdullah Abuolaim, Mahmoud Afifi, and Manos Papagelis. "A versatile computational framework for group pattern mining of pedestrian trajectories". In: *GeoInformatica* 23.4 (2019), pp. 501–531.

[11] Tilemachos Pechlivanoglou, Vincent Chu, and Manos Papagelis. "Efficient mining and exploration of multiple axis-aligned intersecting objects". In: *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, pp. 1276–1281.

[12] Saim Mehmood and Manos Papagelis. “Learning semantic relationships of geographical areas based on trajectories”. In: *2020 21st IEEE International Conference on Mobile Data Management (MDM)*. IEEE. 2020, pp. 109–118.

[13] Tilemachos Pechlivanoglou, Jing Li, Jialin Sun, Farzaneh Heidari, and Manos Papagelis. “Epidemic Spreading in Trajectory Networks”. In: *Big Data Research* 27 (2022), p. 100275.

[14] Nina Yanin and Manos Papagelis. “Optimal Risk-aware POI Recommendations during Epidemics”. In: *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Spatial Computing for Epidemiology*. 2023, pp. 1–12.

[15] Gian Alix, Nina Yanin, Tilemachos Pechlivanoglou, Jing Li, Farzaneh Heidari, and Manos Papagelis. “A mobility-based recommendation system for mitigating the risk of infection during epidemics”. In: *2022 23rd IEEE International Conference on Mobile Data Management (MDM)*. IEEE. 2022, pp. 292–295.

[16] Tilemachos Pechlivanoglou, Gian Alix, Nina Yanin, Jing Li, Farzaneh Heidari, and Manos Papagelis. “Microscopic modeling of spatiotemporal epidemic dynamics”. In: *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Spatial Computing for Epidemiology*. 2022, pp. 11–21.

[17] Senzhang Wang, Jiannong Cao, and Philip Yu. “Deep learning for spatio-temporal data mining: A survey”. In: *IEEE transactions on knowledge and data engineering* (2020).

[18] Hao Xue, Bhanu Prakash Voutharoja, and Flora D. Salim. “Leveraging Language Foundation Models for Human Mobility Forecasting”. In: *Proc. of the 30th ACM SIGSPATIAL*. 2022. ISBN: 9781450395298.

[19] Amirhossein Nadiri, Jing Li, Ali Faraji, Ghadeer Abuoda, and Manos Papagelis. “TrajLearn: Trajectory Prediction Learning using Deep Generative Models”. In: *ACM Trans.*

Spatial Algorithms Syst. 11.3 (May 2025). ISSN: 2374-0353. DOI: 10.1145/3729226. URL: https://doi.org/10.1145/3729226.

[20] Nan Han, Shaojie Qiao, Kun Yue, Jianbin Huang, Qiang He, Tingting Tang, Faliang Huang, Chunlin He, and Chang-An Yuan. "Algorithms for Trajectory Points Clustering in Location-Based Social Networks". In: *ACM TIST* 13.3 (Mar. 2022). ISSN: 2157-6904.

[21] Congcong Miao, Jilong Wang, Heng Yu, Weichen Zhang, and Yinyao Qi. "Trajectory-User Linking with Attentive Recurrent Network". In: *Proc. of the 19th AAMAS*. 2020, pp. 878–886.

[22] Mahmoud Alsaeed, Ameeta Agrawal, and Manos Papagelis. "Trajectory-User Linking using Higher-order Mobility Flow Representations". In: *24th IEEE MDM*. 2023, In Press.

[23] Zheng Wang, Cheng Long, and Gao Cong. "Trajectory Simplification with Reinforcement Learning". In: *37th IEEE ICDE*. 2021, pp. 684–695.

[24] Gian Alix and Manos Papagelis. "PathletRL: Trajectory Pathlet Dictionary Construction using Reinforcement Learning". In: *Proc. of the 31st ACM SIGSPATIAL*. 2023, pp. 1–12.

[25] Gian Alix, Arian Haghparast, and Manos Papagelis. "PathletRL++: Optimizing Trajectory Pathlet Extraction and Dictionary Formation via Reinforcement Learning". In: *arXiv preprint arXiv:2412.03715* (2024).

[26] Christian Koetsier, Jelena Fiosina, Jan N. Gremmel, Jörg P. Müller, David M. Woisetschläger, and Monika Sester. "Detection of anomalous vehicle trajectories using federated learning". In: *IOJPRS* 4 (2022), p. 100013.

[27] Mashaal Musleh and Mohamed Mokbel. "A Demonstration of KAMEL: A Scalable BERT-based System for Trajectory Imputation". In: *ACM SIGMOD*. 2023.

[28] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S Yu. "Machine unlearning: A survey". In: *ACM Computing Surveys* 56.1 (2023), pp. 1–36.

[29] Yinzhi Cao and Junfeng Yang. "Towards Making Systems Forget with Machine Unlearning". In: *2015 IEEE Symposium on Security and Privacy*. 2015, pp. 463–480. DOI: 10.1109/SP.2015.35.

[30] Jonathan Brophy and Daniel Lowd. "Machine Unlearning for Random Forests". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021, pp. 1092–1104. URL: https://proceedings.mlr.press/v139/brophy21a.html.

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

[32] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. "Graph Unlearning". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. CCS '22. Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 499–513. ISBN: 9781450394505. DOI: 10.1145/3548606.3559352. URL: https://doi.org/10.1145/3548606.3559352.

[33] Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. "Machine Unlearning: Solutions and Challenges". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2024), pp. 1–19. DOI: 10.1109/TETCI.2024.3379240.

[34] Na Li, Chunyi Zhou, Yansong Gao, Hui Chen, Anmin Fu, Zhi Zhang, and Yu Shui. "Machine Unlearning: Taxonomy, Metrics, Applications, Challenges, and Prospects". In: *arXiv preprint arXiv:2403.08254* (2024).

[35] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. "Fast machine unlearning without retraining through selective synaptic dampening". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 11. 2024, pp. 12043–12051.

[36] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. "Eternal sunshine of the spotless net: Selective forgetting in deep networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9304–9312.

[37] Ayush K. Tarun, Vikram S. Chundawat, Murari Mandal, and Mohan Kankanhalli. "Fast Yet Effective Machine Unlearning". In: *IEEE Transactions on Neural Networks and Learning Systems* (2023), pp. 1–10. DOI: 10.1109/TNNLS.2023.3266233.

[38] Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, Pranay Sharma, and Sijia Liu. "Model sparsification can simplify machine unlearning". In: *arXiv preprint arXiv:2304.04934* (2023).

[39] Shen Lin, Xiaoyu Zhang, Chenyang Chen, Xiaofeng Chen, and Willy Susilo. "ERM-KTP: Knowledge-Level Machine Unlearning via Knowledge Transfer". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 20147–20155. DOI: 10.1109/CVPR52729.2023.01929.

[40] Zhehao Huang, Xinwen Cheng, JingHao Zheng, Haoran Wang, Zhengbao He, Tao Li, and Xiaolin Huang. "Unified Gradient-Based Machine Unlearning with Remain Geometry Enhancement". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=dheDf5EpBT.

[41] Kairan Zhao, Meghdad Kurmanji, George-Octavian Bărbulescu, Eleni Triantafillou, and Peter Triantafillou. "What makes unlearning hard and what to do about it". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=QAbhLBF72K.

[42] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. "Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 6. 2023, pp. 7210–7217.

[43] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. "Towards Unbounded Machine Unlearning". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 1957–1987. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/062d711fb777322e2152435459e6e9d9-Paper-Conference.pdf.

[44] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. "Zero-shot machine unlearning". In: *IEEE Transactions on Information Forensics and Security* (2023).

[45] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites. "Adaptive machine unlearning". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 16319–16330.

[46] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. "Remember What You Want to Forget: Algorithms for Machine Unlearning". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 18075–18086. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/9627c45df543c816a3ddf2d8ea686a99-Paper.pdf.

[47] Alex Oesterling, Jiaqi Ma, Flavio Calmon, and Himabindu Lakkaraju. "Fair Machine Unlearning: Data Removal while Mitigating Disparities". In: *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*. Ed. by Sanjoy Das-

gupta, Stephan Mandt, and Yingzhen Li. Vol. 238. Proceedings of Machine Learning Research. PMLR, May 2024, pp. 3736–3744. URL: https://proceedings.mlr.press/v238/oesterling24a.html.

[48] Eli Chien, Haoyu Peter Wang, Ziang Chen, and Pan Li. "Certified Machine Unlearning via Noisy Stochastic Gradient Descent". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=h3k2NXu5bJ.

[49] Eli Chien, Haoyu Peter Wang, Ziang Chen, and Pan Li. "Langevin Unlearning: A New Perspective of Noisy Gradient Descent for Machine Unlearning". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=3LKuC8rbyV.

[50] James Y Huang, Wenxuan Zhou, Fei Wang, Fred Morstatter, Sheng Zhang, Hoifung Poon, and Muhao Chen. "Offset Unlearning for Large Language Models". In: *arXiv preprint arXiv:2404.11045* (2024).

[51] Jiaqi Li, Qianshan Wei, Chuanyi Zhang, Guilin Qi, Miaozeng Du, Yongrui Chen, Sheng Bi, and Fan Liu. "Single Image Unlearning: Efficient Machine Unlearning in Multimodal Large Language Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=YNx7ai4zTs.

[52] Yuanshun Yao, Xiaojun Xu, and Yang Liu. "Large Language Model Unlearning". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=8Dy42ThoNe.

[53] Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. "Large Language Model Unlearning via Embedding-Corrupted Prompts". In: *The Thirty-eighth An-*

nual Conference on Neural Information Processing Systems. 2024. URL: https://openreview.net/forum?id=e5icsXBD8Q.

[54] Sheng-Yu Wang, Aaron Hertzmann, Alexei A Efros, Jun-Yan Zhu, and Richard Zhang. "Data Attribution for Text-to-Image Models by Unlearning Synthesized Images". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=kVr3L73pNH.

[55] Myeongseob Ko, Henry Li, Zhun Wang, Jonathan Patsenker, Jiachen T. Wang, Qinbin Li, Ming Jin, Dawn Song, and Ruoxi Jia. "Boosting Alignment for Post-Unlearning Text-to-Image Generative Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=93ktalFvnJ.

[56] Yong-Hyun Park, Sangdoo Yun, Jin-Hwa Kim, Junho Kim, Geonhui Jang, Yonghyun Jeong, Junghyo Jo, and Gayoung Lee. "Direct Unlearning Optimization for Robust and Safe Text-to-Image Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=UdXE5V2d0O.

[57] Yimeng Zhang, Xin Chen, Jinghan Jia, Yihua Zhang, Chongyu Fan, Jiancheng Liu, Mingyi Hong, Ke Ding, and Sijia Liu. "Defensive Unlearning with Adversarial Training for Robust Concept Erasure in Diffusion Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=dkpmfIydrF.

[58] Jiabao Ji, Yujian Liu, Yang Zhang, Gaowen Liu, Ramana Rao Kompella, Sijia Liu, and Shiyu Chang. "Reversing the Forget-Retain Objectives: An Efficient LLM Unlearning Framework from Logit Difference". In: *The Thirty-eighth Annual Conference on Neural*

Information Processing Systems. 2024. URL: https://openreview.net/forum?id=tYdR1lTWqh.

[59] Jinghan Jia, Jiancheng Liu, Yihua Zhang, Parikshit Ram, Nathalie Baracaldo, and Sijia Liu. "WAGLE: Strategic Weight Attribution for Effective and Modular Unlearning in Large Language Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=VzOgnDJMgh.

[60] Weilin Lin, Li Liu, Shaokui Wei, Jianze Li, and Hui Xiong. "Unveiling and Mitigating Backdoor Vulnerabilities based on Unlearning Weight Changes and Backdoor Activeness". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=MfGRUVFtn9.

[61] Martin Andres Bertran, Shuai Tang, Michael Kearns, Jamie Heather Morgenstern, Aaron Roth, and Steven Wu. "Reconstruction Attacks on Machine Unlearning: Simple Models are Vulnerable". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=i4gqCM1r3z.

[62] Leo Schwinn, David Dobre, Sophie Xhonneux, Gauthier Gidel, and Stephan Günnemann. "Soft Prompt Threats: Attacking Safety Alignment and Unlearning in Open-Source LLMs through the Embedding Space". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=CLxcLPfARc.

[63] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. "When Machine Unlearning Jeopardizes Privacy". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. CCS '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021,

pp. 896–911. ISBN: 9781450384544. DOI: 10.1145/3460120.3484756. URL: https://doi.org/10.1145/3460120.3484756.

[64] Hanlin Gu, Win Kent Ong, Chee Seng Chan, and Lixin Fan. "Ferrari: Federated Feature Unlearning via Optimizing Feature Sensitivity". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=YxyYTcv3hp.

[65] Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. "Tofu: A task of fictitious unlearning for llms". In: *arXiv preprint arXiv:2401.06121* (2024).

[66] Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. "Machine Unlearning". In: *2021 IEEE Symposium on Security and Privacy (SP)*. 2021, pp. 141–159. DOI: 10.1109/SP40001.2021.00019.

[67] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. "Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 6. 2023, pp. 7210–7217.

[68] Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. "Machine Unlearning: Solutions and Challenges". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2024), pp. 1–19. DOI: 10.1109/TETCI.2024.3379240.

[69] Lorenzo Bracciale, Marco Bonola, Pierpaolo Loreti, Giuseppe Bianchi, Raul Amici, and Antonello Rabuffi. *CRAWDAD roma/taxi*. IEEE Dataport, 2022. URL: https://dx.doi.org/10.15783/C7QC7M.

[70] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. "Understanding Mobility Based on GPS Data". In: *Proceedings of the 10th International Conference on Ubiquitous Computing*. UbiComp '08. Seoul, Korea: Association for Computing Machinery, 2008, pp. 312–321.

[71] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. "Mining Interesting Locations and Travel Sequences from GPS Trajectories". In: *TheWebConf*. 2009, pp. 791–800.

[72] Yu Zheng, Xing Xie, and Wei-Ying Ma. "GeoLife: A Collaborative Social Networking Service among User, location and trajectory". In: *IEEE Dat. Eng. B.* (2010).

[73] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. "Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45.1 (2015), pp. 129–142. ISSN: 2168-2216.

[74] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[75] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[76] Chongyu Fan, Jiancheng Liu, Alfred Hero, and Sijia Liu. "Challenging forgets: Unveiling the worst-case forget sets in machine unlearning". In: *arXiv preprint arXiv:2403.07362* (2024).

[77] Jacopo Bonato, Marco Cotogni, and Luigi Sabetta. "Is Retain Set All You Need in Machine Unlearning? Restoring Performance of Unlearned Models with Out-of-Distribution Images". In: *Computer Vision – ECCV 2024*. Ed. by Aleš Leonardis, Elisa

Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol. Cham: Springer Nature Switzerland, 2025, pp. 1–19. ISBN: 978-3-031-73232-4.

[78] Hritam Basak and Zhaozheng Yin. "Forget More to Learn More: Domain-Specific Feature Unlearning for Semi-supervised and Unsupervised Domain Adaptation". In: *Computer Vision – ECCV 2024*. Ed. by Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol. Cham: Springer Nature Switzerland, 2025, pp. 130–148. ISBN: 978-3-031-72920-1.

[79] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. "Membership inference attacks from first principles". In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2022, pp. 1897–1914.

[80] Meghdad Kurmanji, Eleni Triantafillou, and Peter Triantafillou. "Machine Unlearning in Learned Databases: An Experimental Analysis". In: *Proc. ACM Manag. Data* 2.1 (Mar. 2024). DOI: 10.1145/3639304. URL: https://doi.org/10.1145/3639304.

Appendix A

Detailed Results

A.1 Results Under Uniform Random Sampling

In this section, we present detailed results evaluating unlearning methods under the setting of uniform random sampling of users for deletion. Specifically, users are selected independently and uniformly at random from the dataset to simulate deletion requests. This evaluation setting provides a controlled and unbiased scenario to measure the unlearning performance across a representative subset of the user population. It allows us to assess the robustness, consistency, and utility preservation of the unlearning approach without introducing bias from specific user attributes or behaviors.

Table A.1: The results for the HO-Rome dataset using uniform sampling. All numbers are presented as percentages.

HO-Rome Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	0.72	0.73	0	100	0.66	0.64	0.61	100	1.33	1.2	0.73	100	0.83	0.61	3.14
Finetuning	98.52	5.74	6.17	24.3	92.28	5.61	6.23	12.84	94.2	5.72	6.23	6.07	94.9	5.82	6.17	4.58
NegGrad	100	5.58	6.05	24.1	99.84	4.84	5.12	12.15	98.09	5.15	5.12	5.99	96.38	5.92	5.91	4.71
NegGrad+	100	5.51	5.56	24.32	99.84	4.9	5.23	11.95	97.44	5.59	5.64	6.08	96.5	5.97	6.08	4.45
Bad-T	98.52	5.71	6.14	24.29	92.12	5.56	6.2	12.83	94.13	5.65	6.35	6.1	94.9	5.78	6.26	4.58
SCRUB	99.26	5.72	6.14	24.28	92.28	5.57	6.2	12.9	94.12	5.67	6.23	5.96	94.97	5.8	6.17	4.41
TraceHiding (Ent.)	99.26	5.69	6.17	23.67	95.19	5.45	5.88	11.94	94.83	5.63	5.85	5.45	95.14	5.8	6.02	4.42
TraceHiding (C.D.)	99.26	5.71	6.08	23.73	95.5	5.57	5.99	11.44	95.35	5.71	6.2	4.86	95.22	5.77	5.99	4.15
TraceHiding (Uni.)	99.26	5.73	6.14	22.6	95.98	5.63	6.05	12.71	94.98	5.82	6.14	5.87	95.18	5.78	5.99	4.06
LSTM																
Retraining	100	6.37	3.98	4.25	100	0.73	0.7	5.89	100	0.79	0.7	2.7	100	0.85	0.67	5.85
Finetuning	98.52	11.1	6.84	22.42	90.56	11.08	6.84	14.57	89.08	11.02	6.73	7.69	90.14	11.35	6.81	8.24
NegGrad	99.26	11.18	6.93	20.69	96.85	11.26	6.75	14.52	95.22	11.18	6.78	7.93	94.72	11.73	6.35	8.46
NegGrad+	99.26	11.24	6.9	22.23	97.01	11.3	6.99	14.3	94.92	11.49	6.64	8	94.53	11.89	6.26	8.42
Bad-T	98.52	11.1	6.87	22.41	90.4	11.13	6.93	14.53	89.16	11.16	6.99	7.65	89.96	11.53	6.99	8.29
SCRUB	98.52	11.1	6.73	22.42	90.56	11.07	6.84	14.59	89.08	11.05	6.73	7.67	90.07	11.38	6.81	8.26
TraceHiding(Ent.)	98.52	11.08	6.75	22.3	94.47	11.19	6.61	13.68	91.21	11.17	6.7	6.61	95.6	11.92	6.05	7.09
TraceHiding (C.D.)	98.52	11.13	6.78	22.55	94.49	11.17	6.67	12.16	91.52	11.38	6.81	6.76	95.15	11.65	6.23	8.43
TraceHiding (Uni.)	98.52	11.08	6.78	21.57	95.26	11.2	6.75	12.02	91.72	11.04	6.78	6.27	94.64	11.63	6.08	7.86
BERT																
Retraining	100	88.74	39.65	31.35	100	87.76	38.42	26.96	100	84.55	36.61	20.7	100	85.36	32.34	18.12
Finetuning	35.65	81.54	38.04	48.9	60.63	82.44	38.1	43.6	58.48	82.29	37.66	36.82	54.16	82.92	37.02	35.39
NegGrad	86.2	76.09	36.87	48.35	80.6	52.58	29.77	39.77	97.59	7.54	4.74	36.75	99.28	0.64	0.64	27.5
NegGrad+	69.72	80.49	38.04	49.23	72.05	73.06	36.32	42.1	82.58	60.91	31.08	32.03	88.72	41.01	21.67	31.7
Bad-T	75.19	79.36	37.98	49.25	65.36	78.78	37.25	42.84	62.31	78.82	36.64	40.65	53.69	78.19	35.67	42.86
SCRUB	79.54	77.72	37.08	47.77	82.94	67.38	34.27	42.74	70.4	67.86	34.33	38.36	86.53	53.7	25.94	32.48
TraceHiding (Ent.)	86.39	78.17	37.08	45.6	85.59	68.25	34.36	38.79	89.56	55.98	29.94	36.3	94.91	43.64	21.73	33.55
TraceHiding (C.D.)	78.61	80.49	37.81	48.11	72.24	76.4	37.16	42.85	83.77	71.14	34.33	33.48	82.5	70.85	31.29	31.03
TraceHiding (Uni.)	95.28	79.97	37.92	48.1	76.05	74.77	35.67	43.12	75.8	70.13	34.36	34.1	80.55	64.91	30.41	31.83
ModernBERT																
Retraining	100	69.38	39.94	34.27	100	67.02	38.77	26.36	100	63.88	36.78	13.08	100	57	32.4	18.76
Finetuning	62.22	59.56	37.49	4.89	65.61	59.16	37.72	25.28	64.51	61.21	36.9	28.43	64.04	61.07	35.76	19.5
NegGrad	72.59	62.25	40.38	45.43	77.91	62.5	39.47	15.67	79.66	63.03	38.25	27.06	78.83	62.54	36.11	11.43
NegGrad+	76.67	61.98	40.12	13.7	75.29	61.91	39.27	28.12	77.07	61.92	37.6	16.97	75.31	62.18	36.32	29.04
Bad-T	75.28	62.55	40.2	37.3	70.39	62.97	38.68	34.97	68.8	63.88	37.95	17.74	66.37	64.04	35.38	27.45
SCRUB	63.24	61.21	39.56	48.14	65.74	59.98	38.36	35	67.97	60.12	37.72	19.16	68.93	59.19	36.23	17.7
TraceHiding (Ent.)	47.59	58.63	38.25	34.36	70.78	56.69	36.87	25.83	77.64	55.9	35.03	26.79	76.23	53.72	32.63	11.9
TraceHiding (C.D.)	62.13	58.54	38.36	34.72	67.93	58.59	37.92	25.38	70.03	57.23	35.56	20.43	70.44	58.36	34.74	28.12
TraceHiding (Uni.)	44.91	58.37	38.48	46.08	66.75	57.36	36.67	32.71	69.8	57.38	35.94	10.34	73.95	57.1	34.18	11.02

Table A.2: The results for the HO-Geolife dataset using uniform sampling. All numbers are presented as percentages.

HO-Geolife Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	66.17	54.8	35.9	100	72.61	60.85	36.15	100	52.13	41.21	25.43	100	50.56	37.58	23.71
Finetuning	44.73	84.96	64.98	47.99	38.12	85.47	65.2	46.58	25.82	84.88	65.34	46.73	16.19	84.88	65.48	46.12
NegGrad	79.15	83.94	62.56	48.02	58.58	84.96	64.7	45.99	45.64	83.14	62.28	46.01	41.47	83.46	62.42	45.41
NegGrad+	56.83	84.63	64.27	48.04	54.73	85.23	65.41	46.08	46.59	85.52	64.13	46.16	37.29	85.36	63.99	45.55
Bad-T	47.88	84.08	64.84	47.98	38.24	84.38	65.34	46.46	28.81	83.95	64.41	46.42	21.73	83.96	64.06	45.82
SCRUB	50.29	84.37	64.77	48.03	38.33	84.53	65.2	46.48	34.73	84.34	64.2	46.38	30.21	84.03	63.42	45.55
TraceHiding (Ent.)	60.61	84.37	64.41	47.98	45.58	84.59	65.05	46.47	50.46	85.08	63.2	46.25	39.52	84.25	62.85	45.63
TraceHiding (C.D.)	62.42	84.45	64.06	48.05	45.32	84.53	65.2	46.41	48.85	84.43	62.49	46.25	39.07	84.33	62.28	45.64
TraceHiding (Uni.)	50.68	84.39	64.98	47.98	44.64	84.48	65.27	46.31	46.16	85.03	63.2	46.3	42.05	84.22	62.28	45.37
LSTM																
Retraining	100	62.36	50.46	29.48	100	66.42	53.67	32	100	31.08	26.05	17.08	100	31.62	23.56	11.03
Finetuning	42.84	79.64	60.64	46.46	38.24	80.49	61.14	42.74	28.7	79.46	60.57	43.95	21.33	79.11	61	43.6
NegGrad	43.68	79.27	60.57	46.17	42.8	79.78	60.36	42.45	48.53	78.18	58.65	43.36	41.37	78.05	59.64	42.95
NegGrad+	45.63	79.27	60.57	46.24	44.48	80.01	60.57	42.42	39.97	79.87	60	43.76	33.27	79.18	60.93	43.1
Bad-T	40.5	79.16	60.57	46.33	37.86	79.71	60.85	42.51	29.05	78.75	60.78	43.87	22.28	78.04	60.71	43.65
SCRUB	42.84	79.23	60.57	46.37	37.09	79.95	60.78	42.56	30	78.97	60.5	43.79	23.22	78.53	60.64	43.48
TraceHiding (Ent.)	43.9	79.23	60.71	46.19	39.61	80.08	60.93	42.5	42.15	79.08	59.22	43.48	43.26	78.79	59.15	43.06
TraceHiding (C.D.)	43.35	79.31	60.57	45.92	41.39	79.9	60.78	41.77	39.18	79.47	59.5	43.67	48.86	78.85	56.87	43.4
TraceHiding (Uni.)	43.9	79.33	60.28	46.38	40.62	79.97	60.85	41.91	39.69	79.32	59.64	43.3	51.65	78.8	57.79	43.11
BERT																
Retraining	100	86.93	65.12	42.88	100	87.3	67.76	36.9	100	86.52	59.36	39.18	100	84.32	56.51	39.8
Finetuning	63.07	87.69	66.98	48.59	49.69	87.89	66.9	48.28	40.6	88.17	65.27	47.78	36.8	88.65	64.98	47.1
NegGrad	93.14	85.85	63.35	47.67	78.3	87.15	64.63	48.36	79.8	69.52	50.25	45.49	88.9	59.6	41.21	45.89
NegGrad+	95.43	86.68	63.84	48.39	77.81	87.04	65.05	47.69	66.98	81.43	57.86	45.82	83.23	77.66	52.81	45.16
Bad-T	70.07	86.26	63.35	48.64	69.22	86.52	65.05	48.33	82.52	83.34	54.66	47.06	83.41	84.01	52.67	46.89
SCRUB	89	86.61	63.2	48.28	76.01	84.9	64.34	47.39	71.6	81.62	56.87	46.25	77.86	78.78	53.38	45.48
TraceHiding (Ent.)	99.13	83.17	61.14	48.64	100	80.31	60.93	47.43	93.77	75.23	52.46	45.07	93.86	74.93	48.68	44.44
TraceHiding (C.D.)	74.61	86.31	64.41	49.32	83.11	85.62	65.05	48.11	78.11	85.2	58.65	47.05	75.06	86.01	56.65	45.9
TraceHiding (Uni.)	92.86	86.36	62.28	49.13	90.59	86.77	64.91	47.34	70.27	83.01	57.72	47.01	68.99	84.81	58.36	46.03
ModernBERT																
Retraining	100	83.5	60	26.73	100	82.35	62.35	16.46	100	81.12	53.67	35.57	99.84	77.29	48.9	25.01
Finetuning	73.87	81.75	61.85	30.19	73.51	83.01	61.14	12.62	51.11	83.32	61.35	39.52	51.97	84.02	59.15	24.73
NegGrad	92.97	80.21	58.65	28.73	78.17	82.03	61.42	21.31	62.5	81.77	57.22	45.87	67.82	81.1	54.31	43.45
NegGrad+	93.16	81.59	58.86	23.07	77.53	82.31	61.49	20.51	61.41	82.89	58.22	29.14	62.23	83.83	57.15	44.07
Bad-T	70.59	80.89	58.72	16.28	67.86	81.46	60.5	30.77	63.69	79.98	55.16	33.28	57.87	81.92	54.59	44.69
SCRUB	90.74	80.88	58.79	47	76.09	82.34	61.07	21.1	52.72	80.99	59.07	28.64	66.37	81.09	54.95	34.4
TraceHiding (Enr.)	93.12	80.72	59	47.63	73.3	80.4	60.07	37.34	85.76	79.1	54.59	43.73	85.53	80.46	50.96	38.32
TraceHiding (C.D.)	92.43	81.2	59.22	28.08	68.74	80.85	60	29.85	72.19	79.94	55.44	38.11	66.42	80.55	53.81	26.97
TraceHiding (Uni.)	91.18	80.37	59.29	23.72	67.38	80.96	61.07	46.02	65.11	81.69	56.65	38.1	77.75	81.77	52.67	38.69

Table A.3: The results for the HO-NYC dataset using uniform sampling. All numbers are presented as percentages.

HO-NYC Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	89.41	75.42	46.98	100	87.43	72.22	42.42	100	81.72	65.22	40.5	100	80.84	58.67	39.72
Finetuning	15.64	92.18	77.55	49.61	15.22	92.82	77.54	47.6	18.22	93.15	77.24	43.67	14.79	92.86	77.21	40.24
NegGrad	32.77	87.46	74.24	49.38	34.1	87.18	73.61	47.08	29.92	87.49	73.28	43.06	28.85	86.35	71.19	39.68
NegGrad+	23.64	87.67	74.64	49.35	31.69	88.06	74.15	47.15	27.65	88.7	74.06	43.14	27.9	89	72.74	39.8
Bad-T	26.07	87.61	74.57	49.42	32.13	87.34	73.66	47.11	31.78	87.81	73.02	43.04	31.29	87.08	71.12	39.56
SCRUB	26.8	87.66	74.59	49.47	28.99	87.67	74.1	47.26	33.41	88.16	73.15	43.07	33.98	87.54	70.55	39.72
TraceHiding (Ent.)	33.38	87.68	74.43	49.37	31.08	87.53	73.9	47.23	38.88	88.25	72.46	42.95	45.71	86.66	68.17	39.63
TraceHiding (C.D.)	29.99	87.61	74.55	49.4	42.78	87.41	73.13	46.88	37.36	88.12	72.69	43.29	54.47	86.2	66.54	39.52
TraceHiding (Uni.)	33.9	87.73	74.44	49.35	36.81	87.56	73.55	47.13	38.57	88.05	72.57	42.8	43.57	87.06	69.16	39.86
LSTM																
Retraining	100	91.23	76.35	48.4	100	87.34	71.98	42.25	100	85.96	66.54	40.62	100	66.33	48.72	36.24
Finetuning	13.12	93.14	79.63	49.59	13.05	93.79	79.61	47.73	14.31	93.89	79.64	44.76	12.07	93.84	79.19	39.91
NegGrad	25.28	90.58	77.42	49.35	24.77	90.47	77.05	47.42	26.74	90.62	76.42	44.39	24.17	90.02	75.28	39.44
NegGrad+	22.24	90.74	77.6	49.35	22.84	90.99	77.56	47.47	24.71	91.44	77.14	44.31	23.96	91.56	76.15	39.41
Bad-T	19.3	90.68	77.63	49.59	24.03	90.84	77.2	47.39	24.35	90.91	76.51	44.33	24.42	90.78	75	39.26
SCRUB	25.53	90.69	77.46	49.57	26.77	90.77	77.1	47.37	31.48	90.98	76.08	44.28	28.15	90.62	74.83	39.11
TraceHiding (Ent.)	28.05	90.69	77.6	49.5	28.66	90.71	77.02	47.33	34.86	90.94	75.64	44.51	36.88	90.26	73.13	38.91
TraceHiding (C.D.)	26.16	90.71	77.64	49.54	26.91	90.71	77.14	47.34	36.17	90.88	75.6	44.1	39.62	90.29	72.7	38.68
TraceHiding (Uni.)	37.8	90.66	77.32	49.48	36.8	90.7	76.79	47.22	28.05	91.06	76.32	44.38	40.77	90.03	72.63	38.97
BERT																
Retraining	100	99.76	85.62	42.25	100	99.75	82.83	37.4	100	99.62	78.22	33.35	100	99.4	71.52	32.94
Finetuning	10.58	99.35	86.33	49.99	8.38	99.37	86.2	49.73	7.39	99.44	86.11	49.23	5.96	99.45	85.42	47.81
NegGrad	46.01	98.66	84.72	50	90.69	46.42	39.62	49.44	100	0.55	0.5	45.9	100	1.1	0.92	38.44
NegGrad+	42.53	99.69	85.82	49.99	63.82	91.08	76.35	49.57	94.18	78.26	61.82	48.52	98.63	71.23	51.35	47.47
Bad-T	55.44	99.65	85.46	49.77	77.86	98.28	81.19	49.76	52.03	99.23	80.56	48.87	46.95	99.49	77.33	48.21
SCRUB	54.15	99.42	85.55	49.49	90.32	93.99	77.69	49.79	97.86	90.4	70.36	49.27	98.98	93.94	66.22	48.89
TraceHiding (Ent.)	71.95	99.41	84.99	50	96.93	97.75	80.32	49.16	97.12	96.36	75	48.17	98.92	97.28	69.2	48.21
TraceHiding (C.D.)	63.45	99.58	85.57	49.26	79.25	99.11	82.55	49.19	89.33	99.05	77.8	47.92	91.36	98.71	70.8	46.79
TraceHiding (Uni.)	68.76	99.47	85.21	49.82	94.93	98	80.58	49.39	98.25	97.84	75.63	48.59	98.36	97.33	69	47.91
ModernBERT																
Retraining	100	99.86	81.42	38.66	100	99.81	78.5	36.57	100	99.79	73.25	30.64	100	99.63	66.62	27.09
Finetuning	2.11	99.76	82.07	50	2.89	99.66	81.87	48.55	2.38	99.79	81.74	45.84	2.27	99.71	80.86	41.62
NegGrad	39.72	99.92	81.92	49.84	33.36	99.85	80.78	47.93	48.12	87.89	69.75	44.97	35.23	93.56	71.57	40.19
NegGrad+	37.35	99.92	81.92	49.97	31.94	99.69	80.9	47.6	43.36	97.36	76.1	44.71	33.24	97.92	74.73	40.81
Bad-T	43.31	99.93	81.81	49.76	34.41	99.93	81.26	47.62	48.92	99.91	78.01	43.77	55.37	99.88	73.49	40.08
SCRUB	36.59	99.9	82.06	49.99	47.27	98.63	78.94	48.16	47.03	97.84	75.99	45.6	76.38	93.55	64.78	44.55
TraceHiding (Ent.)	51.6	99.8	81.71	49.97	23.25	99.25	80.84	48.39	80.71	96.5	71.75	45.99	94.84	96.68	65.22	45.09
TraceHiding (C.D.)	54.88	99.85	81.61	49.99	42.55	99.03	79.66	47.94	60.14	98.5	75.42	45.72	88.69	97.37	66.2	42.53
TraceHiding (Uni.)	40.65	99.77	81.61	49.99	39.35	99.31	80.28	47.97	76.08	97.9	72.95	45.5	94.78	96.52	65	44.65

A.2 Results Under Targeted Sampling

In this section, we present results from evaluating unlearning methods under a targeted sampling strategy, where users with high informational content are selected for deletion. To identify such users, we compute the entropy of their trajectories based on token level representations, capturing the degree of variability and unpredictability in their behavior. Lower entropy values indicate users whose data contributes more significantly to the model's knowledge. By focusing on these low-entropy users, we simulate a more challenging unlearning scenario where the model must forget influential or informative data. This targeted approach provides insight into the performance of unlearning methods when facing deletion requests that potentially have a larger impact on model utility and generalization.

Table A.4: The results for the HO-Rome dataset using targeted sampling. All numbers are presented as percentages.

HO-Rome Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	17.54	14.12	12.04	100	28.16	20.18	14.24	100	3.23	2.89	10.72	100	1.27	0.88	21.41
Finetuning	98.1	5.7	6.05	19.76	98.42	5.8	6.11	12.64	96.77	5.98	6.17	6.64	92.69	5.66	6.14	0.95
NegGrad	100	3.18	3.19	17.99	100	1.69	1.96	9.27	99.92	1.42	1.52	2.49	99.26	0.96	1.23	2.81
NegGrad+	100	3.57	3.63	13.07	100	3.64	3.16	6.12	99.67	3.87	3.57	4.57	98.52	3.11	2.84	3.55
Bad-T	98.1	5.69	5.99	19.61	99.05	5.67	5.82	13.36	96.91	5.73	5.94	7.05	95.37	5.04	5.09	5.42
SCRUB	99.31	5.63	5.58	20.64	98.89	5.63	5.7	11.18	98.35	5.65	5.38	6.74	96.59	4.59	4.68	3.06
TraceHiding (Ent.)	98.79	5.7	6.05	18.61	98.57	5.87	6.05	11.82	96.93	5.8	6.02	6.71	94.6	4.71	5.26	3.07
TraceHiding (C.D.)	97.49	5.68	6.11	20.37	98.58	5.79	6.14	12.57	96.69	5.88	5.96	6.24	94.99	4.85	5.38	3.56
TraceHiding (Uni.)	98.79	5.71	6.14	20.35	98.26	5.74	6.08	12.67	97.08	5.81	5.99	6.51	94.79	4.93	5.41	2.8
LSTM																
Retraining	100	8.66	5.56	9.69	100	5.84	3.95	11.97	100	0.86	0.67	26.18	100	0.95	0.7	40.94
Finetuning	97.93	11.14	6.75	21.68	86.44	11.23	6.96	16.89	92.2	11.53	7.08	10.55	87.79	11.11	6.96	7.79
NegGrad	100	8.46	5.85	10.62	99.83	5.23	3.27	14.16	99.92	3.54	2.19	5.75	99.81	2.75	0.99	6.41
NegGrad+	100	9.47	6.14	17.47	99.84	8.99	5.47	11.6	99.85	7.94	4.77	11.05	98.86	9.08	4.5	4.36
Bad-T	95.03	11.14	6.78	22.64	95.38	10.98	6.75	18.21	94.66	11.38	6.4	11.95	93.18	10.5	6.08	14.33
SCRUB	96.41	11.22	6.96	21.54	97.55	10.63	6.32	15.31	95.03	10.67	6.17	10.3	95.41	10.14	5.47	6.33
TraceHiding (Ent.)	97.93	11.1	6.84	21.39	87.6	11.07	6.96	16.84	94.42	11.33	6.58	8.12	92.34	11.05	5.91	5.59
TraceHiding (C.D.)	96.55	11.19	6.81	21.84	87.79	11.12	7.11	16.09	93.74	11.36	6.96	10.32	91.27	11.14	6.02	6.42
TraceHiding (Uni.)	96.55	11.09	6.81	22.25	87.73	11.14	6.96	16.46	93.98	11.35	6.81	8.92	91.8	11.03	5.94	4.95
BERT																
Retraining	100	88.89	39.18	28.56	100	86.7	37.37	24.4	100	86.91	36.67	21.38	100	81.34	32.66	20.12
Finetuning	63.03	81.87	38.54	48.86	55.95	81.01	37.63	43.21	58.48	82.51	38.19	37.62	55.65	83.21	36.35	33.82
NegGrad	77.28	74.93	36.75	46.45	75.64	52.88	28.36	41.34	97.42	10.98	7.16	35.09	100	1.21	0.73	27.29
NegGrad+	69.13	77.67	37.51	45.72	75.75	70.64	34.77	39.26	80.16	58	30.15	32.72	79.33	50.08	26.17	30.5
Bad-T	71.9	78.71	37.66	48.11	60.75	77.99	35.94	40.61	52.21	79.7	37.02	41.03	54.15	78.92	35.38	43.29
SCRUB	79.46	79.87	38.27	47.53	83.21	66.19	31.73	43.57	82.9	65.24	32.02	37.01	76.68	63.94	29.24	32.93
TraceHiding (Ent.)	82.58	75.71	37.28	46.19	81.45	66.26	33.33	40	86.84	58.58	29.77	35.86	94.18	49	23.39	30.22
TraceHiding (C.D.)	84.56	79.01	37.92	47.11	77.55	76.01	36.58	39.89	85.77	72.97	34.47	33.77	82.9	71.68	31.49	30.89
TraceHiding (Uni.)	91.14	78.09	37.72	47.59	83.22	73.7	35.82	38.44	80.84	72.09	34.68	34.29	77.98	70.16	31.17	30.04
ModernBERT																
Retraining	100	67.59	40.03	34.75	100	67.12	38.36	25.44	100	62.76	37.13	24.57	100	56.81	31.52	17.6
Finetuning	66.14	59.72	37.49	40.88	61.12	59.67	35.82	33.65	63.41	59.95	36.32	30.33	65.09	60.55	35.67	26.55
NegGrad	76.76	62.51	40.23	40.3	74.91	62.78	39.5	34.59	74.13	62.63	38.92	30.75	78.91	61.97	35.67	24.89
NegGrad+	74.6	62.29	40.23	40.98	70.8	61.83	39.39	34.92	72.25	61.56	37.78	27.21	77.25	62.36	34.82	13.49
Bad-T	82.38	62.61	40.29	14.22	63.24	62.96	39.21	36.29	68.97	63.23	37.4	29.99	64.8	64.72	35.47	13
SCRUB	55.8	59.38	38.65	41.72	69.03	59.64	38.89	35.6	66.33	59.42	37.31	31.35	70.26	60.44	35.58	9.56
TraceHiding (Ent.)	48.27	57.52	37.49	41.92	63.65	58.81	37.81	35.29	69.27	55.27	35.35	30.09	81.05	51.4	30.99	9.51
TraceHiding (C.D.)	62.54	58.47	38.65	41.9	68.23	57.56	37.4	31.49	65.31	57.72	36.35	29.88	70.97	58.5	35	9.66
TraceHiding (Uni.)	71.83	59.25	38.45	41.26	66.12	57.59	37.49	27.82	67.5	56.33	36.4	31.53	74.2	57.16	33.27	10.84

Table A.5: The results for the HO-Geolife dataset using targeted sampling. All numbers are presented as percentages.

HO-Geolife Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	75.84	60.78	34.49	100	66.53	53.81	37.99	100	54.74	42.35	30.34	100	41.42	32.03	27.56
Finetuning	50.86	84.7	65.27	48.15	26.93	84.84	65.27	47.83	10.61	83.3	65.55	48.87	23.22	85.93	65.55	45.24
NegGrad	68.65	83.66	64.13	48.07	54.66	84.13	63.56	47.64	27.98	81.61	63.84	48.42	40.71	85.29	64.2	44.42
NegGrad+	68.16	83.96	64.06	48.17	49.58	84.69	64.2	47.7	19.34	83.3	65.2	48.62	38.65	86.79	65.05	44.58
Bad-T	50.44	83.86	64.98	48.11	27.51	83.82	64.56	47.79	12.33	82.1	65.05	48.75	24.95	84.99	63.99	45.09
SCRUB	54.06	83.82	64.91	48.15	29.83	84	65.12	47.81	14.17	82.34	65.12	48.69	32.04	85.2	64.41	44.92
TraceHiding (Ent.)	58.8	83.84	64.77	4.78	51.83	84.35	64.06	10.69	49.85	82.01	60.85	19.98	44.25	86.51	63.84	23.59
TraceHiding (C.D.)	60.13	83.92	64.91	4.8	54.94	84.54	63.77	10.6	44.15	81.68	61.42	20.22	41.49	85.27	64.27	23.43
TraceHiding (Uni.)	58.03	83.84	64.84	48.14	48.48	84.23	64.7	47.59	25.93	81.98	64.48	48.4	41.3	86.04	64.34	44.36
LSTM																
Retraining	100	65.94	55.66	31.27	100	60.07	50.46	33.5	100	39.87	33.24	20.64	100	33.34	26.33	13.31
Finetuning	33.56	79.31	60.5	47.59	36.4	79.77	61.21	46.57	19.12	78.3	60.85	45.88	27.47	80.54	61.21	41.1
NegGrad	34.05	78.89	60.36	47.66	48.2	78.61	60.36	46.2	33.69	77.55	59.79	45.81	38.15	80.2	59.57	40.76
NegGrad+	34.05	79.01	60.5	47.64	42.26	79.18	60.71	46.36	36.99	78.12	58.51	45.8	38.06	81.18	60.07	40.98
Bad-T	33.56	78.84	60.5	47.71	36.2	78.9	60.85	46.6	20.47	77.68	60.78	45.73	27.48	79.74	60.78	41.07
SCRUB	33.8	78.91	60.64	47.74	37.35	79.13	61	46.53	20.59	77.83	60.64	46.29	29.83	80.12	60.85	41.12
TraceHiding (Ent.)	44.57	78.9	60	5.43	47.03	79.12	59.79	10.16	34.06	77.76	59	19.73	51.89	80.54	57.86	19.76
TraceHiding (C.D.)	41.82	79.01	59.86	5.56	55.29	78.97	60	10.1	38.24	77.73	58.58	19.82	48.69	80.57	57.94	19.96
TraceHiding (Uni.)	32.8	78.97	60.57	47.64	46	79.12	60.64	46.47	32.4	77.65	59.72	46.2	43.04	80.56	59.72	40.91
BERT																
Retraining	100	86.05	67.19	33.2	100	85.7	64.91	42.02	100	83.82	58.72	41.12	100	86.44	59.79	37.87
Finetuning	46.26	87.24	66.83	49.5	44.62	87.35	65.48	48.48	24.14	85.53	65.41	49.37	37.02	89.55	66.12	47.67
NegGrad	67.37	86.02	65.77	49.07	79.43	80.19	59.86	48.26	81.68	69.91	49.11	47.7	90.26	64.67	46.48	45.37
NegGrad+	67.25	86.43	65.62	49.39	81.16	83.16	61.21	47.95	84.88	79.11	55.8	47.41	89.6	79.08	54.23	44.69
Bad-T	55.55	86.2	65.27	49.54	67.34	82.8	60.71	48.46	78.39	80.95	55.02	48.01	82.63	84.04	53.81	46.41
SCRUB	65.78	86.16	65.05	49.61	70.23	82.33	60.57	47.34	74.92	79.51	55.02	48.21	92.78	78	52.31	45.3
TraceHiding (Ent.)	90.8	84.76	65.2	48.91	84.4	80.36	59.72	46.99	83.93	78.71	54.73	46.95	99.32	72.97	49.89	43.37
TraceHiding (C.D.)	74.41	86.14	65.62	48.67	60.74	85.74	63.63	48.7	75.79	83.07	57.15	47.96	68.88	86.05	59.36	44.63
TraceHiding (Uni.)	91.82	84.93	65.2	49.01	68.03	85	61.99	47.83	82.47	81.08	55.87	47.7	80.05	83.83	56.87	44.5
ModernBERT																
Retraining	99.76	82.9	62.56	8.49	100	82.74	59.57	25.29	99.68	79.98	53.52	14.69	100	81.04	53.88	27.09
Finetuning	47.87	81.13	61.42	11.28	58.5	82.74	61.64	25.22	33.83	80.86	60	34.05	44.77	84.17	61.42	32.26
NegGrad	58.53	80.34	60.71	18.58	67.29	81.38	59.36	17.17	50.51	78.62	56.23	23.72	60.59	82.5	57.65	35.17
NegGrad+	54.63	81.13	61.49	24.93	64.07	82.01	60.64	33.97	47.65	80.78	58.43	23.32	59.48	83.78	58.79	43.65
Bad-T	56.64	80.81	60.78	29.42	61.76	80.49	59.22	24.69	57.8	78.7	55.87	29.28	53.26	81.63	56.87	41.85
SCRUB	61.61	80.09	60.85	10.36	61.27	80.96	59.86	24.76	63.67	78.81	55.3	41.54	62.79	82.78	57.37	26.96
TraceHiding (Ent.)	58.48	79.01	60.64	39.56	65.08	79.21	58.86	24.14	70.24	77.61	54.66	22.39	69.07	80.73	57.51	40.74
TraceHiding (C.D.)	45.39	79	61	39.69	54.41	80.96	60.71	41.15	52.33	77.59	56.16	22.42	61.04	82.47	58.79	28.3
TraceHiding (Uni.)	47.41	79.6	60.28	10.49	70.38	79.82	58.93	25.18	64.88	77.76	55.37	47.14	66.09	81.51	56.23	42.7

Table A.6: The results for the HO-NYC dataset using targeted sampling. All numbers are presented as percentages.

HO-NYC Dataset																
Sample Size	1 %				5 %				10 %				20 %			
Methods	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA	UA	RA	TA	MIA
GRU																
Retraining	100	88.83	75.07	45.92	100	84.15	68.96	43.72	100	84.54	66.36	41.76	100	78.79	56	39.73
Finetuning	15.51	91.97	77.28	49.95	13.74	92.59	77.33	48.11	14.43	92.8	77.02	45.04	15.18	93.18	76.96	39.78
NegGrad	31.33	87.45	74.52	49.9	31.1	87.09	73.39	47.84	30.97	87.02	72.49	44.63	28.17	86.56	71.11	39.37
NegGrad+	36.67	87.64	74.64	49.91	28.3	88.01	74.01	47.9	30.95	88.21	73.48	44.68	27.19	89.06	72.76	39.43
Bad-T	26.65	87.45	74.52	49.9	33.25	87.2	73.02	47.76	33.59	87.31	72.39	44.77	35.34	87.2	70.71	39.13
SCRUB	29.28	87.55	74.49	49.87	34.13	87.52	73.41	48.02	34.52	87.61	72.61	44.7	33.34	87.72	70.95	39.18
TraceHiding (Ent.)	37.16	87.66	74.46	49.92	42.01	87.3	72.83	47.9	41.32	87.4	71.7	44.73	38.38	86.97	69.42	39.04
TraceHiding (C.D.)	33.98	87.7	74.6	49.87	38.22	87.34	72.97	47.89	43.38	87.39	71.56	44.65	36.91	87.47	69.93	39.06
TraceHiding (Uni.)	31.23	87.61	74.46	49.91	36.96	87.45	73.1	47.97	36.65	87.47	72.31	44.85	43.86	87.27	68.55	38.79
LSTM																
Retraining	100	91.78	76.31	47.19	100	88.41	71.1	42.18	100	85.84	66.33	40.46	100	84.21	58.23	39.79
Finetuning	6.14	93.6	79.97	49.49	9.46	93.48	79.94	48.36	11.72	93.65	79.5	44.75	12.86	93.75	78.99	40.71
NegGrad	20.9	90.54	77.44	49.2	26.35	90.07	76.65	47.87	23.46	90.21	76.33	44.33	24.24	90.16	74.91	40.07
NegGrad+	18.39	90.7	77.82	49.2	28.15	90.83	76.83	47.94	24.49	91.23	76.74	44.25	24.78	91.81	75.71	40.04
Bad-T	11.92	90.62	77.76	49.32	28.09	90.5	76.6	47.94	24.46	90.55	76.33	44.12	25.87	90.87	74.8	39.75
SCRUB	19.47	90.72	77.65	49.27	24.75	90.45	77.06	48.14	26.46	90.54	75.86	44.35	31.06	90.84	74.01	39.81
TraceHiding (Ent.)	25.4	90.64	77.52	49.26	34.09	90.31	76.11	48.13	34.65	90.29	74.97	44.25	39.56	90.48	72.38	39.78
TraceHiding(C.D)	28.53	90.62	77.58	49.28	38.41	90.28	75.94	47.9	29.81	90.45	75.67	44.25	42.24	90.28	71.67	39.75
TraceHiding (Uni.)	30.23	90.64	77.41	49.26	30.25	90.37	76.46	48.05	30.69	90.43	75.46	44.44	40.66	90.32	72.15	39.57
BERT																
Retraining	100	99.79	85.73	41.64	100	99.56	81.74	37.82	100	99.64	78.16	33.72	100	99.44	69.77	32.4
Finetuning	6.84	99.33	86.52	50	6.31	99.46	86.29	49.73	5.19	99.46	86.12	49.38	6.38	99.51	85.48	29.45
NegGrad	63.19	99.73	85.95	49.99	89.09	54.19	46.45	49.58	100	1.17	0.99	45.51	100	0.49	0.37	37.5
NegGrad+	15.79	99.79	86.62	50	72.17	90.83	73.93	49.48	91.77	82.35	62.04	48.92	98	74.28	51.87	48.07
Bad-T	64.05	99.67	85.95	49.96	77.55	98.31	79.96	49.86	54.02	99.28	80.03	49.24	48.08	99.55	76.26	34.82
SCRUB	61.54	99.45	85.72	49.99	91.33	91.41	74.14	49.68	98.66	92.89	71.52	49.25	99.42	91.47	63.23	48.72
TraceHiding (Ent.)	90.78	98.77	84.45	0.99	96.66	98.89	80.24	9.28	98.93	97.79	75.72	18.36	98.88	97.47	67.59	25.56
TraceHiding (C.D.)	80.24	99.17	85.03	49.99	80.91	99.03	81.35	49.52	88.56	98.94	77.74	48.31	94.62	98.7	69.27	32.98
TraceHiding (Uni.)	90.78	98.77	84.45	49.99	96.66	98.89	80.24	49.54	98.93	97.79	75.72	48.43	98.88	97.47	67.59	30.03
ModernBERT																
Retraining	100	99.89	81.53	40.85	100	99.81	76.89	36.08	100	99.74	73.05	31.21	100	99.72	65.03	22.26
Finetuning	3.15	99.74	81.8	50	1.88	99.71	81.91	49.16	2.82	99.7	81.48	44.25	2.86	99.83	81.23	41.66
NegGrad	42.44	99.9	82.09	50	37.48	99.46	79.66	48.33	26.88	99.37	79.38	44.55	47.58	80.08	60.54	32.77
NegGrad+	40.36	99.92	82.1	50	34.43	99.74	80.49	48.88	36.99	98.91	77.93	43.79	38.75	96.59	72.64	24.47
Bad-T	41.54	99.93	82.2	49.99	43.44	99.9	79.94	48.6	44.11	99.88	78.32	43.38	60.67	99.84	71.47	28.31
SCRUB	54.14	99.82	81.81	50	35.91	99.65	80.06	49.03	31.85	99.04	78.23	45.49	75.94	94.15	64.41	45.19
TraceHiding (Ent.)	38.78	99.84	81.97	49.92	51.43	98.28	77.61	48.41	78.1	96.82	71.84	45.46	96.54	96.6	62.81	44.97
TraceHiding (C.D.)	46.58	99.85	81.91	50	40.99	99.23	79.09	48.39	67.96	98.05	73.86	44.82	89.57	97.38	64.72	30.4
TraceHiding (Uni.)	45.85	99.88	81.98	50	47.52	98.93	78.85	48.8	72.95	97.32	73.15	44.94	96.29	96.65	63.15	45.23