

Forgetting by Pruning: Data Deletion in Join Cardinality Estimation

Chaowei He¹, Yuanjun Liu¹, Qingzhi Ma^{1*}, Shenyuan Ren², Xizhao Luo¹, Lei Zhao¹, An Liu^{1*}

¹Soochow University

²Beijing Jiaotong University

cwhe@stu.suda.edu.cn, yjliu1@stu.suda.edu.cn, qzma@suda.edu.cn, syren@bjtu.edu.cn, xzluo@suda.edu.cn, zhaol@suda.edu.cn, anliu@suda.edu.cn

Abstract

Machine unlearning in learned cardinality estimation (CE) systems presents unique challenges due to the complex distributional dependencies in multi-table relational data. Specifically, data deletion, a core component of machine unlearning, faces three critical challenges in learned CE models: attribute-level sensitivity, inter-table propagation and domain disappearance leading to severe overestimation in multi-way joins. We propose Cardinality Estimation Pruning (CEP), the first unlearning framework specifically designed for multi-table learned CE systems. CEP introduces Distribution Sensitivity Pruning, which constructs semi-join deletion results and computes sensitivity scores to guide parameter pruning, and Domain Pruning, which removes support for value domains entirely eliminated by deletion. We evaluate CEP on state-of-the-art architectures NeuroCard and FACE across IMDB and TPC-H datasets. Results demonstrate CEP consistently achieves the lowest Q-error in multi-table scenarios, particularly under high deletion ratios, often outperforming full retraining. Furthermore, CEP significantly reduces convergence iterations, incurring negligible computational overhead of 0.3%-2.5% of fine-tuning time.

Code — <https://github.com/heriec/CEP>

Introduction

Machine unlearning (Cao and Yang 2015; Guo et al. 2019; Bourtole et al. 2021), the task of removing the influence of specific data points from trained models, has become increasingly important for data privacy and dynamic model updates. Regulations such as GDPR (Voigt and Von dem Bussche 2017) and CCPA (Goldman 2020) mandate the right to data deletion, while real-world systems often require models to adapt to deletions from data expiration or correction. In database (DB) systems, machine learning (ML) models are increasingly being explored to enhance core components. Among them, cardinality estimation (CE), a fundamental task in query optimization, must reflect data deletions accurately to maintain query performance. This necessitates applying unlearning techniques to ensure the model’s predictions remain reliable after data removal.

*Corresponding author

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Unfortunately, current learned CE systems lack effective mechanisms for handling data deletions. Existing approaches either require expensive full model retraining for every deletion or resort to naive fine-tuning that ignores the distributional shifts in relational data. Recent work like Kurmanji, Triantafillou, and Triantafillou (2024) applies unlearning to downstream DB tasks but focuses only on single-table scenarios, neglecting the complexities of multi-table join estimation. Moreover, this work relies on existing unlearning methods without introducing new strategies tailored to the specific challenges of cardinality estimation.

The fundamental challenge lies in CE’s unique characteristics that distinguish it from traditional machine learning scenarios. First, **attribute-level sensitivity** varies dramatically based on value rarity and distribution patterns. For instance, deleting 1,000 “Action” movies from a large dataset causes minimal distributional impact since many similar records remain, while deleting just 5 “Film-Noir” movies may eliminate the entire genre, drastically altering the model’s learned representations. Second, **inter-table propagation** compounds this complexity, as deletions cascade through foreign key relationships, affecting not only the target table but also related tables and their join distributions. A single actor deletion impacts actor-specific queries, movie-actor join cardinalities, and potentially multi-way joins involving genres, directors, and ratings.

Another critical challenge is **domain disappearance**, where deletions remove all instances of certain attribute values, shrinking the model’s input space. This phenomenon is particularly problematic in multi-table scenarios where join operations amplify estimation errors. Without proper handling, models continue to assign non-zero probabilities to vanished domains, leading to severe overestimation in complex multi-way joins. Existing unlearning methods, designed for independent data points, cannot address such domain-level changes inherent in structured relational data.

To address these challenges that remain unaddressed by current unlearning methods in learned CE, we propose **Cardinality Estimation Pruning (CEP)**, an unlearning framework specifically designed for multi-table CE tasks. CEP introduces two key components: (1) **Distribution Sensitivity Pruning** addresses the core challenge of attribute-level sensitivity and inter-table propagation. It constructs semi-join deletion results using join keys and computes sensitiv-

ity scores from distributional shifts to guide fine-grained parameter pruning. (2) **Domain Pruning** tackles domain disappearance by removing deleted value domains from the model’s input space, ensuring zero probability mass is assigned to obsolete values. Following the pruning phase, CEP performs fine-tuning on the retained data to restore model performance and ensure accurate CE on the retained dataset.

In summary, our contributions are as follows:

- We present the first unlearning framework for multi-table learned CE, identifying key challenges in relational data deletion and proposing CEP with specialized pruning techniques to enable efficient removal of training records without full retraining.
- We design two specialized components: (i) Distribution Sensitivity Pruning computes sensitivity scores from distributional shifts between full and retained join results to guide fine-grained parameter pruning; (ii) Domain Pruning removes support for value domains entirely eliminated by deletion, ensuring no residual probability mass.
- We evaluate CEP on NeuroCard and FACE across IMDB and TPC-H datasets, demonstrating CEP consistently achieves the lowest Q-error in multi-table scenarios, particularly under high deletion ratios, often outperforming even full retraining. Furthermore, CEP significantly reduces convergence iterations, incurring negligible computational overhead (0.3%-2.5% of fine-tuning time).

Related Work

Machine Learning for CE

Recent research integrates ML into database systems, with CE being a prominent focus. Traditional methods based on histograms (Matias, Vitter, and Wang 1998; Poosala et al. 1996) and sampling (Olken 1993; Haas et al. 1995) often perform poorly on complex queries such as multi-table joins due to attribute independence assumptions. Neural networks, known for their expressive power, have been proposed to overcome this. Naru (Yang et al. 2019) uses autoregressive modeling for conditional distributions; NeuroCard (Yang et al. 2020) extends this to multi-table joins by modeling joint distributions; and FACE (Wang et al. 2021) adopts normalizing flows to learn invertible transformations, improving estimation accuracy and generalization. However, these models inherently assume static data distributions, making them ill-equipped to handle dynamic data changes, particularly data deletions, vital for compliance and real-world database management.

To handle dynamic data, recent efforts have explored adapting CE models to evolving data. Query-driven methods like Warper (Li, Lu, and Kandula 2022) and CardOOD (Li et al. 2024) adapt to evolving workloads through continual query feedback. While valuable, these approaches rely on specific query patterns or extensive historical logs. Data-driven approaches for dynamic CE have also begun to emerge: Kurmanji and Triantafillou (2023) investigate learning under data insertions via transfer learning. More specifically concerning data deletion and unlearning, Kurmanji, Triantafillou, and Triantafillou (2024) directly addresses data deletions in CE. Crucially, however, this work

focuses exclusively on single-table scenarios, leaving the complex challenges of multi-table join estimation for unlearning entirely unaddressed. This multi-table limitation is precisely what our work tackles.

Machine Unlearning

Machine unlearning in deep models aims to remove the influence of specific data points from a trained model without full retraining. Early approaches include Wu, Dobriban, and Davidson (2020), which leveraged stored gradient information for efficient approximation of model updates after data deletion. Influence function-based methods, pioneered by Guo et al. (2019), estimate individual data point impact, but often incur high computational costs due to expensive second-order Hessian computations (Basu, Pope, and Feizi 2020) and perform poorly on non-convex models. To address computational overhead, several works approximate the Hessian. Golatkar, Achille, and Soatto (2020) approximated it using the Fisher information matrix (FIM), enabling more tractable updates. Building on this, Liu et al. (2023) further leveraged FIM to derive theoretical bounds on model divergence via influence functions. Another notable FIM-based approach, Selective Synaptic Dampening (SSD) by Foster, Schoepf, and Brintrup (2024), identifies and suppresses parameters most influenced by forgotten samples by comparing FIMs from retain and deletion data.

Beyond gradient and influence-based techniques, other unlearning strategies involve structural changes or knowledge distillation. Jia et al. (2023) showed that model sparsity can improve unlearning efficacy. SCRUB (Kurmanji et al. 2023) adopts a teacher-student framework to minimize alignment with deleted data while maximizing retention of retained data. While these diverse methods advance machine unlearning, they largely focus on classification and give limited attention to regression settings. Tarun et al. (2023) proposed Blindspot Unlearning by fine-tuning with Gaussian noise replaced deletions, and Chen et al. (2025) extended influence function from linear to logistic regression. Crucially, none of these existing unlearning techniques address the intricate attribute-level sensitivity, inter-table propagation, or domain disappearance unique to multi-table relational data in cardinality estimation. Their general-purpose nature lacks specialized mechanisms for accurate and efficient unlearning in CE models, motivating our novel CEP framework.

Preliminaries

We consider a relational database \mathcal{D} consisting of multiple tables T^1, T^2, \dots, T^n , where each table T contains tuples defined over a set of attributes A^1, A^2, \dots, A^m . Each attribute A^i has an associated domain $Dom(A^i)$, representing the set of distinct values it can take.

Given a pre-trained model with parameters θ^o on the full (i.e. original) dataset \mathcal{D} , we aim to remove the influence of a deleted dataset $\mathcal{D}_d \subset \mathcal{D}$ while preserving performance on the retain set $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_d$. In multi-table scenarios, deletions occur per table: for each table T^i , we have retained and deleted subsets T_r^i and T_d^i respectively, where $\mathcal{D}_r = T_r^1, \dots, T_r^n$ and $\mathcal{D}_d = T_d^1, \dots, T_d^n$. We denote the

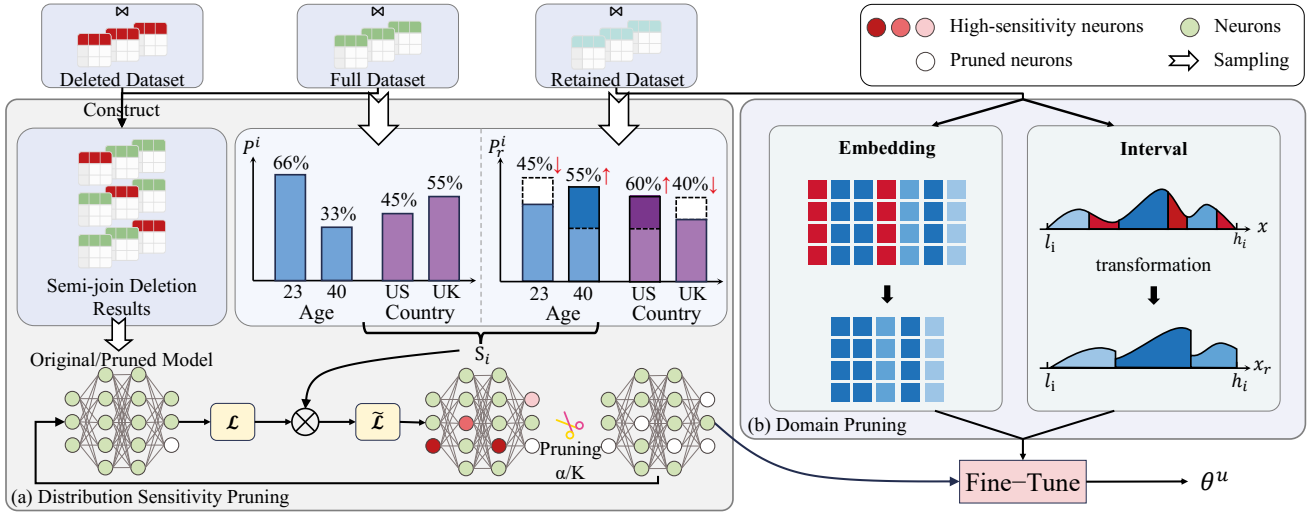


Figure 1: Overview of our proposed method **Cardinality Estimation Pruning (CEP)**, (a) **Distribution Sensitivity Pruning**: constructs semi-join deletion results and computes sensitivity scores from distributional shifts between the full and retained join results. These weights are used to identify and prune highly sensitive parameters. (b) **Domain Pruning**: removes support for value domains entirely erased by the deletion. Pruned models are then fine-tuned on the retained dataset to restore performance.

model’s cardinality estimate of a query q under original parameters θ^o as $|C| = f(q; \theta^o) \cdot |T|$, where $f(\cdot; \theta^o)$ is a selectivity estimator learned by minimizing a loss function \mathcal{L} and $|T|$ is the cardinality of the full outer join. The goal of unlearning is to update parameters θ^o to θ^u , such that it behaves as if \mathcal{D}_d never exists.

Cardinality Estimation Pruning

In this section, we present CEP, which integrates distribution sensitivity pruning and domain pruning to effectively remove the influence of deleted data from CE models. The overall pipeline is shown in Figure 1.

Distribution Sensitivity Pruning

To effectively prune models in response to data deletions, we propose Distribution Sensitivity Pruning, which measures each model parameter’s sensitivity to deleted data and prunes highly sensitive ones.

Sensitivity Scores. Our goal is to compute sensitivity scores that accurately reflect the impact of deleted data on model parameters. While methods like the Hessian matrix or its first-order approximation, the Fisher Information Matrix (FIM), are commonly used to assess parameter importance, their computational cost can be prohibitive in practice. To ensure scalability, we adopt the diagonal approximation of the FIM. This significantly reduces both computational and memory overhead by approximating the importance of parameter θ with its squared gradient magnitude over the deleted dataset \mathcal{D}_d :

$$\mathcal{I}(\mathcal{L}, \mathcal{D}_d) = \mathbb{E}_{x \in \mathcal{D}_d} \left[\left(\frac{\partial \mathcal{L}(x)}{\partial \theta} \right)^2 \right] \quad (1)$$

While $\mathcal{I}(\mathcal{L}, \mathcal{D}_d)$ provides a general measure of parameter importance, they don’t inherently capture the attribute-level

sensitivity crucial for data deletion. To address this, we introduce a novel attribute sensitivity measure S_i , designed to quantify how deletions impact the distributions of individual attributes. From an information-theoretic perspective, S_i quantifies the distributional shift between the original and retained attribute values, revealing significant information changes. For each attribute A^i , we define S_i as:

$$S_i = \frac{|P^i - P_r^i|}{P_r^i} \quad (2)$$

where P^i and P_r^i denote the empirical probability mass functions (or histograms for categorical attributes) of attribute A^i in the full and retained datasets, respectively. S_i measures the relative distributional shift of each attribute, guiding the model to focus pruning on parameters most affected by deletion. We incorporate S_i into the loss function to prioritize attributes with larger shifts, yielding a modified loss $\tilde{\mathcal{L}}(x)$ tailored to the model architecture.

- **Autoregressive Models:** Since AR models learn conditional probability distributions $p(A^i | A^{<i})$, we apply S_i directly to each conditional term: $\tilde{\mathcal{L}}(x) = -\sum_{i=1}^D S_i \cdot \log p(A^i | A^{<i})$.
- **Normalizing Flow Models:** For NF models that learn joint probability distributions through invertible transformations, we aggregate column shifts across all attributes present in a sample: $\tilde{\mathcal{L}}(x) = \mathcal{L}(x) \cdot \left(\sum_{i \in \text{Cols}(x)} S_i \right)$

The final sensitivity scores is computed as:

$$\mathcal{I}(\tilde{\mathcal{L}}, \mathcal{D}_d) = \mathbb{E}_{x \in \mathcal{D}_d} \left[\left(\frac{\partial \tilde{\mathcal{L}}(x)}{\partial \theta} \right)^2 \right] \quad (3)$$

Multi-Table Pruning Strategy. In multi-table scenarios, individual table statistics may not accurately reflect the

Algorithm 1: Distribution Sensitivity Pruning

1: **Input:** $\theta^o, \{J_d^k\}_{k=1}^K, \{P^i\}_{i \in A}, \{P_r^i\}_{i \in A}, \alpha, N_s$
2: **Output:** Pruned parameters θ^u
3: Set per-iteration pruning thresholds $\alpha_k \leftarrow \alpha/K$
4: Compute $S_i = \frac{|P^i - P_r^i|}{P^i}$ for each $i \in A$
5: Initialize sensitivity scores $\mathcal{I}_k \leftarrow 0$ for all k
6: **for** each deleted join J_d^k where $k = 1$ to K **do**
7: **for** iteration $t = 1$ to N_s **do**
8: Sample mini-batch $\mathbf{x}_t \sim J_d^k$
9: Compute loss $\tilde{\mathcal{L}}(\mathbf{x}_t)$ using S_i
10: Update sensitivity scores: $\mathcal{I}_k \leftarrow \mathcal{I}_k + \left(\frac{\partial \tilde{\mathcal{L}}(\mathbf{x}_t)}{\partial \theta}\right)^2$
11: **end for**
12: Update θ^o by pruning with \mathcal{I}_k and α_k
13: **end for**
14: **return** pruned parameters $\theta^u \leftarrow \theta^o$

global distributional changes after join operations. To address this challenge, we need to estimate how data deletion from each table affects the final joined result. We construct **semi-join deletion results** $\{J_d^k\}_{k=1}^K$ to simulate the impact of deleted data on the overall join distribution. For each table T^k , we compute:

$$J_d^k = T^1 \bowtie \dots \bowtie T^{k-1} \bowtie T_d^k \bowtie T^{k+1} \bowtie \dots \bowtie T^n \quad (4)$$

where T_d^k represents the deleted subset from table k , and other tables T^j (for $j \neq k$) remain complete. As fully traversing these joined results to extract distributional features would be computationally expensive, we employ a **sampling-based** approach (Zhao et al. 2018) that captures key distributional characteristics through limited sampling iterations N_s . The sampled results from each J_d^k are then used to compute parameter importance scores that reflect the true impact of deletions under full-schema semantics.

The detailed procedure for Distribution Sensitivity Pruning is outlined in Algorithm 1. The threshold α is distributed evenly across tables ($\alpha_k \leftarrow \alpha/K$). Instead of one-shot pruning all deletions simultaneously, we adopt iterative pruning for each J_d^k as iterative magnitude pruning (IMP) better preserves model performance than one-shot magnitude pruning (OMP) by allowing gradual adaptation and finer sensitivity estimation (Lee, Ajanthan, and Torr 2018; Frankle and Carbin 2018). This removes parameters influenced by deletions and preserves those vital for retained data.

Domain Pruning

To ensure that the model does not retain information about deleted values, we propose **Domain Pruning** that explicitly removes model support for completely eliminated values.

Formally, for a column A^i , let $Dom(A^i)$ denote the original domain and $Dom(A_r^i)$ the retained domain after deletion. The deleted domain values $Dom(A_d^i)$ are:

$$Dom(A_d^i) = Dom(A^i) \setminus Dom(A_r^i) \quad (5)$$

The pruning strategy varies by column type:

- **Categorical columns.** Each categorical attribute is represented via an embedding matrix E^i . We prune the embedding vectors corresponding to deleted domain values:

$$E_{\text{pruned}}^i = E^i[:, Dom(A_d^i)] \quad (6)$$

- **Numerical columns.** After deletion, $Dom(A_r^i)$ may consist of disjoint subranges $\{[a_j, b_j]\}_{j=1}^k$ within the original range $[l, h]$. We transform these valid subranges into a compact continuous space:

$$x_{\text{pruned}} = \frac{x - a_j + \text{offset}_j}{\sum_{n=1}^k (b_n - a_n)} \cdot (h - l) + l' \quad (7)$$

where offset_j is the cumulative length of preceding intervals and l' is the new lower bound. This preserves ordering within $Dom(A_r^i)$ while eliminating gaps from $Dom(A_d^i)$. Query ranges spanning deleted domains are adjusted by clamping to the nearest valid boundaries.

This input-level approach is model-agnostic and ensures eliminated domains cannot influence cardinality estimates.

Experiments

Experimental Setup

Datasets and Workloads. We evaluate on two standard CE benchmarks: (1) IMDB (Leis et al. 2015) dataset with JOB-light workload containing 70 queries across 6 relational tables, with full join resulting in approximately 2×10^{12} tuples; (2) TPC-H (Transaction Processing Performance Council 2014) with 4 tables at scale factor 10 (10GB data) and 100 randomly generated queries.

Baselines & Models. We conduct comparisons against several representative unlearning baselines, including: (a) *Stale*, which continues using the original model without any modification; (b) *Retrain*, which retrains the model on the retained data; (c) *Fine-Tune(FT)*, which adapts the original model by training it on the retained data for a few epochs. We evaluate these methods on two state-of-the-art CE models: the autoregressive **NeuroCard** (Yang et al. 2020) and the normalizing flow model **FACE** (Wang et al. 2021).

Evaluation Measures. We use *Q-error* ($\max(\hat{c}/c, c/\hat{c})$) to measure estimation accuracy, where \hat{c} and c are estimated and true cardinalities. Following prior work (Kurmanji, Triantafillou, and Triantafillou 2024), we do not evaluate membership inference attacks (MIAs) as generative models lack explicit memorization signals. We evaluate on two query types: (i) **Original Queries (OQ)** from the workload to verify successful data removal; (ii) **Complement Queries (CQ)** with inverted range predicates to test selective forgetting. CQ prevent the model from simply redistributing probability mass from deleted regions to unrelated areas. We report Q-error at 50th, 75th, 95th and 99th percentiles.

Unlearning Tasks. We evaluate two unlearning scenarios: (i) **Attribute deletion (A)**: We remove tuples based on column values, e.g., $\text{year} \in [1999, 2010]$. (ii) **Random deletion (R)**: Tuples are randomly selected for removal. We vary affected tables from one to all, and apply deletion ratios ranging from 0.1 to 1. Each task is denoted as [Type]-[Scope]-[Ratio], where Type indicates the

Deletion Ratio		Neurocard					Face					
		0.1	0.3	0.5	0.8	1	0.1	0.3	0.5	0.8	1	
State	OQ	50th	1.42	1.46	1.66	1.77	2.50	1.23	1.30	1.41	1.68	2.58
		75th	2.81	2.76	2.89	2.93	5.92	2.14	2.16	2.45	3.17	8.55
		95th	6.17	6.75	8.13	8.94	1.31e+5	8.26	8.66	10.18	11.52	2.60e+9
		99th	9.52	10.30	12.63	15.71	2.86e+7	35.37	33.49	402.32	63.36	3.18e+10
	CQ	50th	1.38	1.51	1.58	1.84	2.48	1.20	1.24	1.40	1.61	2.57
		75th	2.77	2.34	2.89	2.65	5.63	2.26	2.22	2.48	3.10	8.15
		95th	4.91	5.11	5.91	6.49	1.30e+5	8.29	8.70	9.96	11.35	2.61e+8
		99th	7.44	10.30	8.91	10.01	2.89e+7	33.93	32.52	437.10	63.13	3.19e+10
Retrain	OQ	50th	1.35	1.33	1.46	1.49	1.43	1.18	1.18	1.15	1.13	1.14
		75th	2.33	2.28	2.42	2.33	2.47	2.08	2.00	2.08	2.16	5.86
		95th	3.92	4.31	5.31	5.62	4.86	7.42	7.84	8.39	7.18	2.56e+5
		99th	7.21	5.76	8.18	9.74	6.59	42.32	53.83	247.46	31.18	9.17e+7
	CQ	50th	1.32	1.48	1.48	1.59	1.61	1.20	1.17	1.15	1.13	1.20
		75th	2.19	2.16	2.50	2.43	2.76	2.09	2.06	2.11	2.17	6.13
		95th	4.74	4.11	3.39	3.73	5.02	7.77	7.75	8.79	7.57	2.89e+5
		99th	6.69	5.45	4.67	7.68	6.64	44.16	54.19	262.39	31.59	1.38e+8
Fine-Tune	OQ	50th	1.37	1.34	1.35	1.47	1.42	1.18	1.18	1.17	1.19	1.21
		75th	2.48	2.23	2.48	2.31	2.75	2.09	2.03	2.13	2.15	6.81
		95th	4.81	4.71	5.63	4.83	42.01	6.95	7.08	8.59	8.14	5.98e+4
		99th	6.89	6.75	8.58	8.13	5142	20.51	18.81	169.19	32.31	5.28e+7
	CQ	50th	1.49	1.49	1.50	1.52	1.65	1.18	1.18	1.19	1.17	1.18
		75th	2.19	2.26	2.18	2.85	3.15	2.06	2.08	2.11	2.13	5.38
		95th	3.72	4.23	3.56	4.62	269.69	6.96	6.73	8.18	7.86	1.41e+5
		99th	5.54	7.37	5.06	6.49	3.93e+4	20.88	19.23	152.54	27.63	3.06e+8
CEP(Ours)	OQ	50th	1.30	1.31	1.33	1.33	1.21	1.09	1.16	1.16	1.15	1.11
		75th	2.28	2.19	2.14	2.00	1.94	1.53	1.67	1.68	1.83	1.77
		95th	4.50	4.51	5.79	3.83	4.52	6.56	7.37	8.44	7.90	8.75
		99th	6.49	7.75	6.77	6.69	6.41	13.58	14.05	67.42	26.03	65.8
	CQ	50th	1.30	1.31	1.31	1.35	1.30	1.08	1.18	1.15	1.16	1.08
		75th	2.17	2.15	2.08	2.14	2.07	1.55	1.63	1.68	1.82	1.79
		95th	4.21	4.28	4.77	4.14	6.13	5.40	6.72	8.45	7.56	8.59
		99th	7.58	6.82	5.67	5.82	7.39	13.66	14.00	70.68	26.65	69.53

Table 1: Q-error performance under varying deletion ratios (0.1 to 1.0) on A-1 task in JOB-light, evaluated on both Original Queries (OQ) and Complement Queries (CQ).

deletion strategy (Attribute or Random), `Scope` refers to the number of affected tables, and `Ratio` denotes the proportion of tuples deleted per table. For example, `A-2-0.5` applies column-based deletion conditions to 2 tables, removing 50% of tuples from each, while `R-3-0.3` randomly deletes 30% of tuples from 3 tables.

Main Results

Effect of Deletion Ratios. Table 1 presents Q-error results for A-1 (attribute deletion on 1 table) task across varying deletion ratios (0.1 to 1.0) on NeuroCard and FACE models. Fine-Tune performs comparably to Retrain at low deletion ratios but fails dramatically under full deletion (deletion ratio of 1), with 99th reaching 5142 on OQ and 3.93e+4 on CQ, indicating parameter instability in sparse regions. Retrain avoids this by removing embeddings of deleted values; however, it exhibits fundamental limitations in Face models due to their continuous nature. Since normalizing flows operate over continuous spaces, they lack the granularity to

explicitly exclude specific value sets. Our method, CEP, consistently achieves the lowest Q-error across all percentiles and deletion ratios, maintaining stable performance even under full deletion (50th at 1.21, 99th at 6.41). Interestingly, CEP occasionally outperforms full retraining. We hypothesize this is due to the model sparsification effect of distribution sensitivity pruning, echoing the lottery ticket hypothesis (Frankle and Carbin 2018). Furthermore, our experiments show that CEP enables effective and selective unlearning across different model architectures and deletion ratios, maintaining stable performance as deletion intensity increases while properly excluding deleted data regions.

Multi-Table Deletion Analysis. We evaluate model robustness under column deletions from A-2 (attribute deletion on 2 tables) or A-6 (attribute deletion on 6 tables), with deletion ratios of 0.5 and 1. On Job-light (Table 2), our method achieves the lowest Q-error across all deletion settings using NeuroCard and FACE. For example, in the most challenging case (A-6-1), NeuroCard achieves a 99th percentile

			OQ				CQ			
			50th	75th	95th	99th	50th	75th	95th	99th
NeuroCard	A-2-0.5	stale	1.46	2.74	6.72	10.28	1.43	2.67	5.44	7.97
		Retrain	1.55	2.24	4.09	9.47	1.47	2.46	3.35	7.20
		Fine-Tune	1.34	2.22	4.46	8.95	1.34	2.42	3.59	6.84
		CEP(Ours)	1.34	2.17	4.56	8.87	1.25	2.27	3.60	6.74
	A-2-1	stale	1.86	3.47	1.15e+4	4.70e+7	1.77	3.06	1.05e+4	4.75e+7
		Retrain	1.76	2.58	6.17	8.31	1.67	2.83	5.92	9.45
		Fine-Tune	1.42	2.93	10.56	1.15e+4	1.46	2.38	12.00	1.14e+4
		CEP(Ours)	1.29	2.13	5.26	6.34	1.28	2.09	4.86	6.81
	A-6-0.5	stale	2.15	4.23	10.53	22.96	2.30	4.20	10.94	18.65
		Retrain	1.38	2.12	3.63	6.06	1.30	2.31	4.02	6.20
		Fine-Tune	1.32	2.07	3.55	5.76	1.33	2.26	3.75	6.64
		CEP(Ours)	1.24	1.84	3.66	5.77	1.33	1.85	3.54	6.13
A-6-1	stale	30.77	5.63e+4	4.38e+6	1.83e+7	22.37	5.48e+4	4.20e+6	1.83e+7	
	Retrain	1.24	1.70	4.18	21.84	1.27	1.64	5.29	23.55	
	Fine-Tune	1.61	3.42	24.7	4168	1.47	3.20	24.7	4208	
	CEP(Ours)	1.02	1.63	3.72	4.84	1.03	1.53	3.62	4.52	
FACE	A-2-0.5	stale	1.32	2.86	9.09	33.72	1.32	2.82	8.74	36.08
		Retrain	1.20	2.10	7.56	33.91	1.19	2.12	7.17	32.00
		Fine-Tune	1.17	1.99	7.04	20.79	1.16	1.98	7.04	21.97
		CEP(Ours)	1.15	1.79	7.02	16.26	1.13	1.78	7.13	15.92
	A-2-1	stale	1.83	4.86	1.71e+8	5.18e+10	1.83	4.74	1.70e+8	5.33e+10
		Retrain	1.18	3.30	5.63e+5	2.85e+8	1.21	3.19	4.92e+5	3.17e+8
		Fine-Tune	1.23	3.15	2.64e+5	2.01e+8	1.23	3.03	2.44e+5	2.26e+8
		CEP(Ours)	1.08	1.33	8.20	56.77	1.09	1.33	8.37	59.58
	A-6-0.5	stale	1.66	3.38	11.00	209.72	1.68	3.29	11.04	209.96
		Retrain	1.16	2.04	7.54	205.02	1.16	2.03	7.06	222.23
		Fine-Tune	1.18	2.12	8.52	135.54	1.18	2.08	8.07	135.82
		CEP(Ours)	1.14	1.66	8.55	101.03	1.14	1.62	8.66	104.63
A-6-1	stale	12.42	2.04e+8	1.30e+10	2.84e+10	12.28	2.06e+8	1.35e+10	2.90e+10	
	Retrain	6.34	964	3.15e+6	4.11e+7	6.31	917.5	3.96e+6	5.40e+7	
	Fine-Tune	5.52	1333	8.01e+5	1.21e+7	5.22	1459	4.77e+5	7.93e+6	
	CEP(Ours)	1.02	1.13	8.54	24.70	1.01	1.11	9.25	24.80	

Table 2: Q-error results on JOB-light with deletion ratios 0.5 and 1 on A-2 and A-6 tasks.

Q-error of 4.84, significantly outperforming Retrain (21.84) and Fine-Tune (4168). On TPC-H, which features more uniform data distributions, our method also maintains low Q-error across all percentiles and deletion levels, while effectively mitigating the tail error spikes observed in Fine-Tune. The consistent performance across datasets with distinct distribution characteristics, such as the skewness of Job-light and the uniformity of TPC-H, demonstrates the robustness and generalizability of our method under large-scale structural changes, while retaining better efficiency than full retraining.

Overall Comparison. By comparing Table 1 and Table 2, we find that CEP exhibits increasing advantages as the deletion scope expands. This demonstrates CEP’s robustness across varying deletion sizes. While Fine-Tune and Retrain struggle with accuracy degradation under more extensive deletions, CEP consistently maintains low Q-error and stable performance. Notably, the relative gains of CEP are more significant in the multi-table setting, indicating its strong scalability and robustness when handling larger and more

complex unlearning scenarios.

Training Efficiency Analysis. To assess the efficiency and effectiveness of our proposed CEP method, we compared it against standard fine-tuning (FT) regarding total training time and convergence. Figure 2 shows the training time breakdown for FT and CEP under both A-1 (attribute deletion on 1 table) and A-6 (attribute deletion on 6 tables) tasks, using FACE and NeuroCard models. The pruning phase in CEP adds negligible overhead. For A-1, pruning is only 0.3% of the FT duration. Even for A-6, pruning is just 2.5% of the FT cost for the computationally intensive FACE model. Figure 3 contrasts the training curves across different Q-error quantiles on A-6-0.5. Due to the differing training times between FACE and NeuroCard models, we normalized the x-axis to training progress (0-100%) for a unified comparison. The results show that CEP consistently achieves faster convergence than FT, indicating that CEP facilitates early stopping.

Random Deletion Evaluation. Table 3 presents the Q-error under 0.3 random deletions for R-2 (random deletion on 2

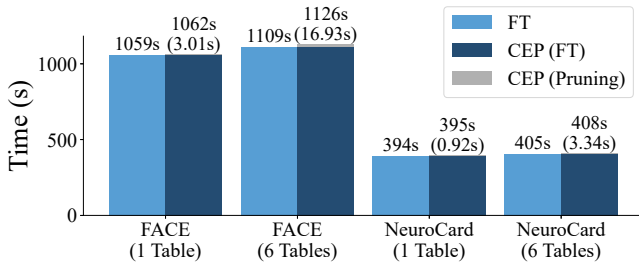


Figure 2: Training time comparison between FT and CEP on FACE and NeuroCard under 1 and 6 tables deletion. Each bar shows the total training time. For CEP, the upper bar represents the combined time of fine-tune and pruning, with the pruning time shown in parentheses.

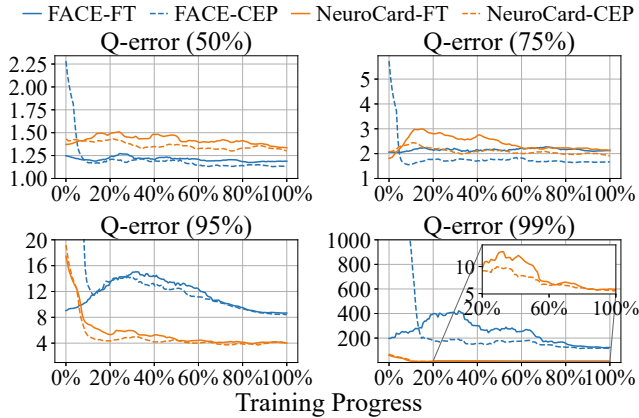


Figure 3: Training convergence curves of FT and CEP across different Q-error on A-6-0.5 task. The x-axis is normalized to training progress (0-100%) to enable aligned comparison between FACE and NeuroCard models.

tables) and R-3 (random deletion on 3 tables) task on JOB-light. Our method consistently delivers the lowest 50th percentile across both NeuroCard and FACE models. At higher percentiles, since random deletion lacks targeted selectivity, distinctions between methods become less apparent. Nevertheless, our approach maintains competitiveness, attaining 99th of 6.66 on NeuroCard and 9.94 on FACE in the R-2 configuration, demonstrating overall robust performance. These results indicate that our method exhibits resilience against random deletion scenarios, effectively mitigating the degradation introduced by non-selective data removal.

Ablation Study. We conduct an ablation study to evaluate the individual contributions of Domain Pruning (D) and Distribution Sensitivity Pruning (S) within our CEP framework. Table 4 presents results for JOB-light with various component combinations on A-1 (attribute deletion on 1 table) task. Our complete method (CEP) consistently delivers superior performance across all quantiles. Notably, disabling domain pruning (CEP-D) causes catastrophic degradation in tail errors: NeuroCard’s 99th deteriorates from 6.41 to 2155, while FACE degrades from 65.8 to 4.10e+5, demonstrating that domain pruning is essential for handling

		NeuroCard				FACE			
		50th	75th	95th	99th	50th	75th	95th	99th
R-2-0.3	stale	1.60	3.25	6.84	13.13	1.20	2.20	8.04	18.23
	Retrain	1.39	2.12	4.54	6.21	1.15	2.01	8.09	13.82
	Fine-Tune	1.46	2.30	4.70	6.78	1.17	2.14	6.74	12.93
	CEP(Ours)	1.13	2.05	5.22	6.66	1.09	1.84	7.95	9.94
R-3-0.3	stale	1.74	4.02	6.87	13.14	1.18	2.34	8.18	18.47
	Retrain	1.50	2.22	4.36	8.14	1.17	2.08	7.04	15.29
	Fine-Tune	1.55	2.22	4.44	7.49	1.19	2.10	6.79	15.43
	CEP(Ours)	1.29	2.11	5.03	6.56	1.17	2.13	6.32	15.38

Table 3: Q-error under 30% random deletion on JOB-light, evaluated on A-1 and A-6 tasks.

Method	NeuroCard				FACE			
	50th	75th	95th	99th	50th	75th	95th	99th
CEP	1.21	1.94	4.52	6.41	1.11	1.77	8.75	65.8
CEP-D	1.41	2.49	9.92	2155	1.19	6.03	51801	4.10e5
CEP-S	1.25	2.10	5.10	7.01	1.15	2.21	12.20	71.1
CEP-D-S	1.42	2.50	21.6	2817	1.20	6.67	87212	6.12e5
FT	1.42	2.75	42	5142	1.21	6.81	59757	5.29e7
FT+D	1.26	2.15	5.55	9.29	1.13	3.15	15.22	73.21
Retrain			/		1.14	5.86	2.56e5	9.17e7
Retrain+D			/		1.09	1.43	10.12	53.37

Table 4: Ablation results on JOB-light with ratio = 1.0 on A-1 task. **D**: Domain Pruning; **S**: Distribution Sensitivity Pruning. Variants (e.g., CEP-D, FT+D) represent ablations or combinations.

eliminated value domains. Distribution sensitivity pruning (CEP-S) provides moderate yet consistent improvements, particularly under high-error conditions. When integrated with baselines, domain pruning proves remarkably effective: FT+D reduces FACE’s 99th from 5.29e+7 to 73.21, and Retrain+D achieves 99th of 53.37. These findings validate that both pruning components are indispensable for our lightweight CEP framework.

Conclusion

We present CEP, the first unlearning framework for multi-table learned CE that enables efficient data deletion without full retraining. Our method combines distribution sensitivity pruning, which leverages sensitivity scores derived from distributional shifts to guide parameter pruning, with domain pruning that eliminates support for entirely removed value domains. Experiments on NeuroCard and FACE with IMDB and TPC-H datasets show that CEP matches or exceeds retraining accuracy, especially under high deletion ratios, while significantly reducing computational cost. These results establish CEP as a practical solution for ML systems in database environments with frequent data deletions, opening promising directions for adaptive unlearning mechanisms in real-world deployments. Future work may extend CEP to support insertions and updates, explore finer-grained unlearning at the tuple or predicate level, and integrate CEP into full query optimizers to assess system-wide impact.

Acknowledgments

This work is supported by projects funded by the Talent Fund of Beijing Jiaotong University (2025JBMCO18) and Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

- Basu, S.; Pope, P.; and Feizi, S. 2020. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*.
- Bourtole, L.; Chandrasekaran, V.; Choquette-Choo, C. A.; Jia, H.; Travers, A.; Zhang, B.; Lie, D.; and Papernot, N. 2021. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, 141–159. IEEE.
- Cao, Y.; and Yang, J. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, 463–480. IEEE.
- Chen, J.; Shi, W.; Lin, W.; Wang, C.; Liu, W.; Sun, H.; and Liu, G. 2025. Unlearning Attacks for Regression Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.
- Foster, J.; Schoepf, S.; and Brintrup, A. 2024. Fast Machine Unlearning without Retraining through Selective Synaptic Dampening. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(11): 12043–12051.
- Frankle, J.; and Carbin, M. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.
- Golatkar, A.; Achille, A.; and Soatto, S. 2020. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9304–9312.
- Goldman, E. 2020. An introduction to the california consumer privacy act (ccpa). *Santa Clara Univ. Legal Studies Research Paper*.
- Guo, C.; Goldstein, T.; Hannun, A.; and Van Der Maaten, L. 2019. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*.
- Haas, P. J.; Naughton, J. F.; Seshadri, S.; and Stokes, L. 1995. Sampling-based estimation of the number of distinct values of an attribute. In *VLDB*, volume 95, 311–322.
- Jia, J.; Liu, J.; Ram, P.; Yao, Y.; Liu, G.; Liu, Y.; Sharma, P.; and Liu, S. 2023. Model sparsity can simplify machine unlearning. *Advances in Neural Information Processing Systems*, 36: 51584–51605.
- Kurmanji, M.; Triantafillou, E.; and Triantafillou, P. 2024. Machine unlearning in learned databases: An experimental analysis. *Proceedings of the ACM on Management of Data*, 2(1): 1–26.
- Kurmanji, M.; and Triantafillou, P. 2023. Detect, distill and update: Learned DB systems facing out of distribution data. *Proceedings of the ACM on Management of Data*, 1(1): 1–27.
- Kurmanji, M.; Triantafillou, P.; Hayes, J.; and Triantafillou, E. 2023. Towards unbounded machine unlearning. *Advances in neural information processing systems*, 36: 1957–1987.
- Lee, N.; Ajanthan, T.; and Torr, P. H. 2018. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.
- Leis, V.; Gubichev, A.; Mirchev, A.; Boncz, P.; Kemper, A.; and Neumann, T. 2015. How good are query optimizers, really? *Proceedings of the VLDB Endowment*, 9(3): 204–215.
- Li, B.; Lu, Y.; and Kandula, S. 2022. Warper: Efficiently adapting learned cardinality estimators to data and workload drifts. In *Proceedings of the 2022 International Conference on Management of Data*, 1920–1933.
- Li, R.; Zhao, K.; Yu, J. X.; and Wang, G. 2024. CardOOD: Robust Query-driven Cardinality Estimation under Out-of-Distribution. *arXiv preprint arXiv:2412.05864*.
- Liu, Y.; Sun, C.; Wu, Y.; and Zhou, A. 2023. Unlearning with fisher masking. *arXiv preprint arXiv:2310.05331*.
- Matias, Y.; Vitter, J. S.; and Wang, M. 1998. Wavelet-based histograms for selectivity estimation. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, 448–459.
- Olken, F. 1993. *Random sampling from databases*. Ph.D. thesis, Citeseer.
- Poosala, V.; Haas, P. J.; Ioannidis, Y. E.; and Shekita, E. J. 1996. Improved histograms for selectivity estimation of range predicates. *ACM Sigmod Record*, 25(2): 294–305.
- Tarun, A. K.; Chundawat, V. S.; Mandal, M.; and Kankanhalli, M. 2023. Deep regression unlearning. In *International Conference on Machine Learning*, 33921–33939. PMLR.
- Transaction Processing Performance Council. 2014. TPC Benchmark H (Decision Support). <http://www.tpc.org/tpch/>. Accessed: 2025-07-06.
- Voigt, P.; and Von dem Bussche, A. 2017. The eu General Data Protection Regulation (GDPR). *A Practical Guide*.
- Wang, J.; Chai, C.; Liu, J.; and Li, G. 2021. FACE: A normalizing flow based cardinality estimator. *Proceedings of the VLDB Endowment*, 15(1): 72–84.
- Wu, Y.; Dobriban, E.; and Davidson, S. 2020. Deltagrad: Rapid retraining of machine learning models. In *International Conference on Machine Learning*, 10355–10366. PMLR.
- Yang, Z.; Kamsetty, A.; Luan, S.; Liang, E.; Duan, Y.; Chen, X.; and Stoica, I. 2020. NeuroCard: one cardinality estimator for all tables. *arXiv preprint arXiv:2006.08109*.
- Yang, Z.; Liang, E.; Kamsetty, A.; Wu, C.; Duan, Y.; Chen, X.; Abbeel, P.; Hellerstein, J. M.; Krishnan, S.; and Stoica, I. 2019. Deep unsupervised cardinality estimation. *arXiv preprint arXiv:1905.04278*.
- Zhao, Z.; Christensen, R.; Li, F.; Hu, X.; and Yi, K. 2018. Random sampling over joins revisited. In *Proceedings of the 2018 International Conference on Management of Data*, 1525–1539.