PGPO: Prototype-Guided Progressive Obfuscation for Privacy-Preserving LLM-Enhanced Recommendation

Anonymous ACL submission

Abstract

LLM-enhanced recommender systems encode item side information into semantic embeddings, but these embeddings may leak sensitive textual attributes through inversion attacks. Existing input-side defenses rely on trusted collection or local token perturbation, often disrupting the inter-item semantic topology needed for recommendation. To address it, we introduce **relational fidelity** as the key objective, requiring semantic shift for privacy while preserving relative item relations for utility. Based on this principle, we propose **prototype-guided progressive obfuscation (PGPO)**, which learns topology-aware obfuscation anchors for high-influence prototype words and propagates them through a static semantic graph. PGPO is optimized with GRPO using individual dissimilarity, structural consistency, and semantic constraint rewards. Experiments on two benchmarks show strong inversion defense with near-upper-bound recommendation performance. The code is available at https://anonymous.4open.science/r/PGPO/.

1 Introduction

In recent years, large language models (LLMs) have leveraged their rich semantic understanding to provide high-quality embeddings for item side information (Wu et al., 2024). This approach effectively alleviates data sparsity and cold-start issues in recommendation. However, these embeddings may leak sensitive user information (Song and Raghunathan, 2020; Wang et al., 2025). Embedding inversion attacks can reconstruct the original text from public embeddings with high accuracy, and the risk increases when data is sent to uncontrolled third-party LLM APIs (Tsai et al., 2026).

Faced with this input-side threat, existing defenses struggle. Post-hoc recommendation protections such as differential privacy in recommender systems (Dwork et al., 2006; Müllner et al., 2023) and federated learning for recommendation (McMahan et al., 2017; Qin et al., 2021) assume a trustworthy collector and can hardly prevent leakage once data leaves the device. Traditional synonym-substitution obfuscation stays at the word level (Jin et al., 2020), treating items as isolated tokens and easily shattering semantics, thereby destroying semantic relationships between items anchored by side information.

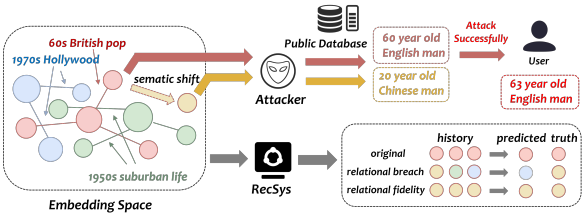


Figure 1: Motivation for relational-fidelity obfuscation in protecting item side information for LLM-enhanced recommendation.

To simultaneously protect privacy and maintain utility, input-side protection should achieve two semantic-level goals. The first goal is to **induce sufficient semantic shift** so that embedding inversion attacks cannot exploit the invertible mapping between embeddings and their source texts (Morris et al., 2023; Zhang et al., 2025). If obfuscated embeddings remain too close to the original, attackers can still approximate the text. Therefore, embeddings must move away from the original to break this invertible connection. The second goal is to **preserve local semantic neighborhood structure** that connect cold and popular items through shared attributes (Vasile et al., 2016; Liu et al., 2023). Maintaining these relative relationships preserves the semantic skeleton needed for recommendation without requiring global semantic fidelity.

Traditional discrete token-level substitution cannot finely control the magnitude or direction of changes in continuous embedding space. Such perturbations disrupt local semantic relations and

tightly couple deviation strength with order preservation, making joint optimization impossible. The *discrete→continuous→discrete* pipeline of LLM allows fine-grained manipulation of embeddings before decoding (Naparstek, 2026; Zhang et al., 2023), generating new words while decoupling deviation strength from relational structure.

However, recent LLM-based text privacy methods ,e.g., EmojiPrompt (Lin et al., 2025), LLM-Anonymization (Staab et al., 2025), and Incogni-Text (Frikha et al., 2025), all follow semantic fidelity, hiding private attributes while preserving literal meaning. This misaligns with recommender systems, which need relational fidelity, i.e., preserving the relative semantic skeleton across items rather than literal text. Under relational fidelity, obfuscated content can be arbitrary as long as a usable item similarity structure persists, turning the task from constrained masking into active topology reconstruction. LLMs naturally suit this, yet existing methods were never designed for it.

To realize **relational fidelity**, we propose a **prototype-guided progressive obfuscation (PGPO) framework** with three sequential stages: prototype-aware graph construction, prototype variant exploration, and anchor-oriented graph expansion. We first identify high-influence prototypes and build a static semantic neighbor graph to capture their structural impact. The LLM then explores suitable obfuscated variants for these prototypes, which serve as stable structural anchors. Finally, the framework propagates these anchored mappings through the static graph to cover the remaining vocabulary. This process is optimized using Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with three complementary rewards: i) **individual dissimilarity** maximizes embedding distance to enforce *semantic shift*, ii) **structural consistency** preserves local similarity order to maintain the semantic distribution necessary for recommendation, and iii) **semantic anchoring** constrains generation within category-specific semantic manifolds to prevent degenerate shortcuts. By jointly optimizing these rewards, the framework balances privacy and utility, protecting sensitive information while preserving the relative semantic structure crucial for recommendation.

Our contributions are as follows:

- We formulate input-side obfuscation for LLM-enhanced recommendation as a topology-preserving privacy problem, introducing **relational fidelity** to defend against embedding inversion while preserving relative item semantics.
- We propose a **prototype-guided progressive obfuscation framework** that scales topology-aware obfuscation from high-influence prototype words to the full vocabulary.
- We design three GRPO rewards, i.e., individual dissimilarity, structural consistency, and semantic constraint, to jointly enforce semantic shift and preserve item relations.
- Under aligned CosSim, experiments on two benchmarks show that our method achieves the highest recommendation utility at comparable defense strength.

2 Related Work

LLM-Enhanced Recommender Systems. Large language models (LLMs) have reshaped recommender systems, either as semantic enhancers enriching item metadata, user profiles, and interaction graphs (Wu et al., 2024; Wei et al., 2024), or as direct recommenders via language modeling, prompting, instruction tuning, or parameter-efficient fine-tuning, covering frameworks like P5 (Geng et al., 2022), M6-Rec (Cui et al., 2022), TALLRec (Bao et al., 2023), LLaRA (Liao et al., 2024) and ReLLa (Lin et al., 2024), and ranking or explanation models (Dai et al., 2023; Li et al., 2023b). However, encoding side information and interactions into continuous embeddings raises privacy concerns. User clicks can be inferred from exposure (Xin et al., 2023), and LLM recommenders are vulnerable to inversion attacks (Wang et al., 2025), underscoring the need for input-level privacy protection (Luo et al., 2024).

Embedding Inversion Attacks. Embedding inversion attacks aim to reconstruct original text or sensitive attributes from exposed embeddings, posing significant privacy risks in NLP. Early studies demonstrated that embedding models leak rich semantic (Song and Raghunathan, 2020; Fredrikson et al., 2015). Subsequent work further recovered full sentences using generative decoders (Li et al., 2023a) and inverted dense embeddings with high fidelity through iterative optimization and beam search (Morris et al., 2023). Recent attacks further expose practical vulnerabilities under cross-model (Huang et al., 2024), black-box (Chen et al., 2025), multilingual (Chen et al., 2024), and zero-shot settings (Zhang et al., 2025), motivating input-side protection for recommendation embeddings.

Text Obfuscation Defenses. Text obfuscation protects privacy by perturbing raw text while preserving downstream utility. Existing approaches include word- or character-level substitution and adversarial perturbation (Ebrahimi et al., 2018; Jia and Liang, 2017), homograph or typo-based replacement, and context-aware embedding perturbation (Zhou et al., 2023), as well as differential privacy-based sanitization such as SanText/SanText+ (Yue et al., 2021) and local DP token mapping (Feyisetan et al., 2020; Qu et al., 2021; Mai et al., 2023). Other methods exploit controlled paraphrasing, masking, or hybrid perturbations to hide sensitive attributes. Recent work explores LLM-based privacy protection, including prompt obfuscation (Lin et al., 2025), LLM-based anonymization (Staab et al., 2025), conditional attribute randomization (Frikha et al., 2025), utility-aware rewriting (Yang et al., 2025b), and privacy-preserving LLM inference (Tong et al., 2025). However, these studies mostly target general NLP text, user-generated content, or LLM prompts, leaving privacy-preserving obfuscation for item-side textual metadata in recommender systems largely underexplored.

3 Method

3.1 Problem Formulation

The obfuscated output z is generated from the original word x using its static neighbors and cached history samples. Let $\mathbf{u} = \phi(x)$ be the public top-K neighbor set retrieved from the static semantic graph, and let $\mathcal{H}$ denote the cached original-variant pairs selected from $\mathbf{u}$ and the global recent queue. The generation process is

$$z \sim P(\cdot \mid x, \mathbf{u}, \mathcal{H}; \xi), \tag{1}$$

where ξ denotes stochastic decoding and candidate exploration randomness.

We consider a white-box attacker who knows the LLM parameters, static graph, retrieval function ϕ, and original vocabulary. The attacker may observe the released obfuscated text or variant set, but not the hidden generation trajectory Γ, including processing order, cache and queue states, sampled candidates, and original-variant alignments in history samples. Let $\mathcal{O}$ denote all attacker-observable auxiliary information. The privacy objective is to maximize $H(X|Z, \mathcal{O})$, equivalently minimizing $I(X; Z|\mathcal{O})$.

We learn a mapping $f : \mathcal{W} \to \mathcal{W}'$ such that each $w' = f(w)$ satisfies two criteria that jointly define relational fidelity:

Definition 1 (Individual Dissimilarity). Minimize the semantic similarity between a word and its obfuscated version to defeat inversion:

$$\min\ \mathbb{E}_{w\sim\mathcal{W}}\big[\mathrm{sim}\big(e(w), e(f(w))\big)\big], \tag{2}$$

where $e(\cdot)$ is the text encoder and sim is cosine similarity.

Definition 2 (Relational Consistency). Maximize the correlation between the similarity of any original word pair and that of their obfuscated counterparts, preserving the relative semantic topology:

$$\max\ \mathrm{Corr}\big(\mathrm{sim}(u, v),\ \mathrm{sim}(f(u), f(v))\big), \tag{3}$$

with Corr e.g. Spearman's rank correlation.

These objectives conflict, as semantic displacement improves privacy but may damage relational structure. We therefore optimize obfuscation in the continuous embedding space to balance privacy and relational fidelity.

3.2 Overall Framework

The core difficulty is that full-vocabulary joint optimization is challenging due to the large vocabulary size. We observe, however, that the global semantic topology is not uniformly distributed: a small set of **prototype words** exerts disproportionate structural influence. Based on this, we propose a **prototype-guided progressive obfuscation framework**, shown in Figure 2. The pipeline operates in three stages:

- **Identify** prototype words and build a static semantic neighbor graph.
- **Explore** suitable obfuscated variants for prototypes as structural anchors.
- **Propagate** these anchored mappings to the full vocabulary through the graph.

3.2.1 Prototype-Aware Graph Construction

Motivation. The global similarity topology is shaped predominantly by a few central words. On one hand, identifying these prototypes gives us high-leverage anchor points; on the other hand, precomputing a static neighbor graph decouples expensive similarity computation from generation, enabling scalable context retrieval.

Prototype Identification. The influence score of a word w is the total similarity received from other

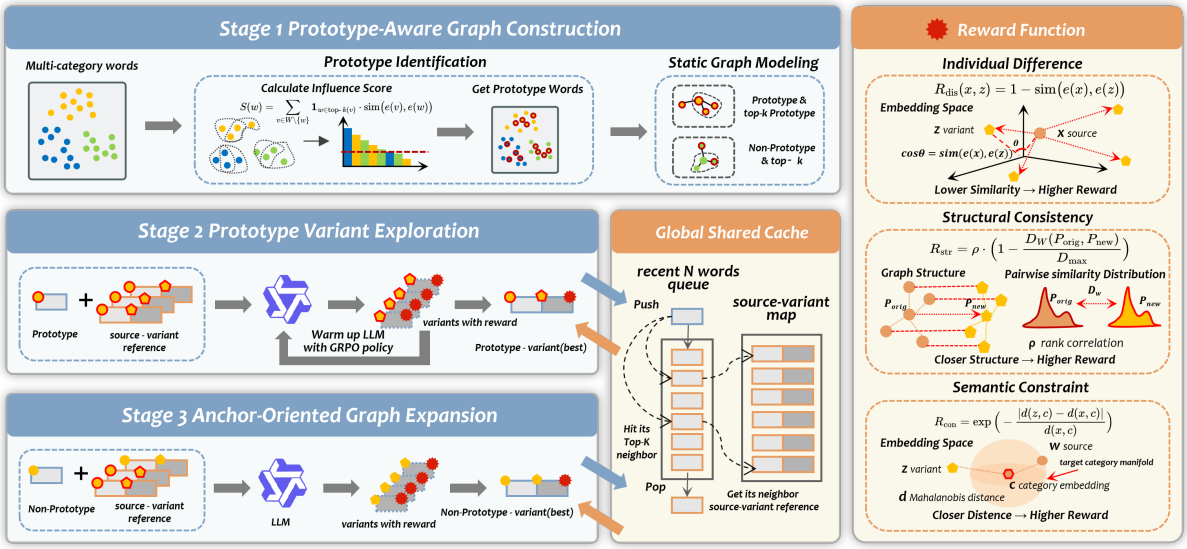


Figure 2: Overview of PGPO. High-influence prototype words are first identified, and a vocabulary-level static semantic graph is constructed. The LLM then explores prototype obfuscation anchors under GRPO-based reward optimization and propagates them through the graph to the remaining vocabulary.

words that rank w among their top-k neighbors:

$$S(w) = \sum_{v \in W \setminus \{w\}} \mathbf{1}_{w \in \text{top-}k(v)} \cdot \text{sim}\big(e(v), e(w)\big). \tag{4}$$

Sorting all words by $S(w)$ in descending order, we select the top M to form the prototype set $C_{\text{Prototype}}$. The hyperparameter k balances structural coverage and computational cost.

Static Graph Modeling. For each $w \in \mathcal{W}$, we precompute and cache its top-K semantic nearest neighbors based on cosine similarity. This graph later serves as an efficient index for retrieving semantically relevant context during generation.

3.2.2 Prototype Variant Exploration

To establish a reliable structural skeleton, we first learn obfuscation mappings for the prototype set $C_{\text{Prototype}}$ through a two-stage process.

1) Warm-up Phase. For T_{warmup} steps, we fine-tune the LLM on $C_{\text{Prototype}}$ using GRPO. This allows the model to rapidly acquire the obfuscation behavior and, guided by the structural consistency reward, begin shaping the relative semantic relationships among prototypes.

2) Prototype Generation Phase. After warm-up, the model is frozen. Each prototype $w \in C_{\text{Prototype}}$ is obfuscated conditioned on its top-K neighbors with cached variants from $C_{\text{Prototype}}$, producing a coherent set of prototype mappings that anchor the global topology.

3.2.3 Anchor-Oriented Graph Expansion

Once all prototypes are processed, the remaining words $\mathcal{W} \setminus C_{\text{Prototype}}$ are obfuscated. For each non-prototype w, we retrieve its top-K neighbors from the static graph and include those with an already finalized variant as few-shot examples. Since high-influence prototypes appear in many neighborhoods, the structural skeleton established earlier smoothly propagates outward. This hierarchical progression from core to periphery mirrors the underlying semantic influence structure, making large-scale, topology-aware obfuscation efficient and coherent.

3.3 Prompt Template and Caching Mechanism

We adopt a structured prompt template with a dynamic `{history_samples}` field. At generation time, each example is formatted as an original word and its finalized obfuscation.

```
Rewrite the term "{w}" (category: {c(w)
    }) into one new term.
Strict rules:
- Must strictly belong to the "{c(w)}"
    category and sound completely
    natural.
- Use exactly 2 to 4 words, matching the
     original term's style, length, and
    rhythm.
- Zero overlap: Do not use any synonyms,
     rephrasing, similar forms, or parts
     of the original term.
Output exactly and only: <answer>new
    term</answer>
```

```
{context_str}{history samples}
```

A two-level caching system maintains stability:

- **Per-word Cache:** stores the highest-reward variant from the latest group for each word.
- **Global Recent Queue:** tracks context-available words, with size N in training and no limit in inference.

Context assembly has three steps: (i) retrieve top-K static neighbors, (ii) filter by the global queue, and (iii) insert cached variants into the prompt.

3.4 Reward-Guided Relational Fidelity Optimization

We optimize the LLM with GRPO, whose group sampling enables diverse, critic-free training. To balance semantic shift and distribution preservation while avoiding degeneration, we use a weighted sum of three normalized rewards in $[0, 1]$:

$$R_{\text{total}} = \lambda_{\text{dis}} R_{\text{dis}} + \lambda_{\text{str}} R_{\text{str}} + \lambda_{\text{con}} R_{\text{con}}. \quad (5)$$

Individual Dissimilarity Reward. This reward operationalizes Criterion 1. It penalizes semantic proximity between the obfuscated output and the original word, pushing the embedding beyond the attacker's local reconstruction range:

$$R_{\text{dis}}(x, z) = 1 - \text{sim}\big(e(x), e(z)\big). \quad (6)$$

Structural Consistency Reward. This reward operationalizes Criterion 2 by evaluating whether the local similarity ordering of context words is preserved after obfuscation. Given a set of context pairs $\mathcal{H} = \{(x_i, z_i)\}_{i=1}^{|\mathcal{H}|}$, we compute pairwise similarity distributions:

$$\begin{aligned} P_{\text{orig}}(i,j) &= \frac{\exp(\text{sim}(x_i, x_j)/\tau)}{\sum_{k \neq l} \exp(\text{sim}(x_k, x_l)/\tau)}, \\ P_{\text{new}}(i,j) &= \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k \neq l} \exp(\text{sim}(z_k, z_l)/\tau)}. \end{aligned} \quad (7)$$

Let D_W denote the Wasserstein distance between these distributions (Villani et al., 2009), and ρ the Spearman correlation of the raw similarity matrices (Spearman, 1961). The reward is

$$R_{\text{str}} = \rho \cdot \left(1 - \frac{D_W(P_{\text{orig}}, P_{\text{new}})}{D_{\max}}\right), D_{\max} = 0.5. \quad (8)$$

This dual-term reward encourages both the preservation of similarity order (via ρ) and the overall distribution (via D_W).

Semantic Constraint Reward. While the previous rewards encourage semantic shift and relational preservation, the lack of spatial constraints may result in degenerate outputs. We therefore introduce a semantic anchor reward to constrain obfuscation within the category-specific embedding manifold. The semantic anchor reward is defined as

$$R_{\text{con}} = \exp\left(-\frac{|d(z,c) - d(x,c)|}{d(x,c)}\right), \quad (9)$$

where $d(z, c)$ denotes the Mahalanobis distance of embedding z to the category center μ_c with covariance Σ_c (Weinberger and Saul, 2009).

Training and Inference. During training, R_{total} drives the GRPO policy updates. During inference, the best candidate from each group is selected. Joint optimization automatically balances semantic shift and distribution preservation, maintaining the relational fidelity essential for recommendation.

4 Theoretical Analysis

Detailed derivations are deferred to Appendix A. Let Γ be the residual hidden generation trajectory and $\mathcal{O}$ the attacker-observable information.

Lemma 1 (Trajectory Leakage Decomposition) *For the obfuscated output $Z \sim p(\cdot|X, \mathcal{O}, \Gamma)$,*

$$\begin{aligned} I(X; Z|\mathcal{O}) = H(Z|\mathcal{O}) - \mathbb{E}\big[H(Z|X, \mathcal{O}, \Gamma)\big] \\ - I(Z; \Gamma|X, \mathcal{O}). \end{aligned} \quad (10)$$

Lemma 2 (Semantic-Mode Entropy) *Partition the output space into semantic modes with weights α_m and intra-mode distributions p_m. Then*

$$\begin{aligned} \mathbb{E}\big[H(Z|X, \mathcal{O}, \Gamma)\big] = \mathbb{E}\left[H(\alpha) + \sum_m \alpha_m H(p_m)\right] \\ - \epsilon. \end{aligned} \quad (11)$$

where $\epsilon = \mathbb{E}[H(M|Z, X, \mathcal{O}, \Gamma)]$ measures residual mode ambiguity.

With the reward-shaped policy p_λ induced by Eq. (5), these lemmas lead to the main privacy guarantee.

Theorem 1 (Attack Success Bound for PGPO) *For a finite original-word set $\mathcal{X}$ and an approximately uniform prior,*

$$\begin{aligned} P_{\text{succ}} \leq \frac{1}{\log_2 |\mathcal{X}|} \Bigg(\epsilon^\lambda - \mathbb{E}_\lambda \left[H(\alpha^\lambda) + \sum_m \alpha_m^\lambda H(p_m^\lambda) \right] \\ + H_\lambda(Z|\mathcal{O}) - I_\lambda(Z; \Gamma|X, \mathcal{O}) + 1 \Bigg). \end{aligned} \quad (12)$$

Here R_{dis} promotes mode-weight entropy $H(\alpha^{\lambda})$, R_{con} preserves valid intra-mode uncertainty, R_{str} controls ϵ^{λ}, and $I_{\lambda}(Z;\Gamma|X,\mathcal{O})$ captures trajectory-induced uncertainty.

For the final item representation $\mathbf{h}_{\text{item}}$, data processing gives

$$I(X;\mathbf{h}_{\text{item}}|C,\mathcal{O}) \leq I(X;e(Z),S|C,\mathcal{O}). \quad (13)$$

If collaborative signals contain limited residual information about X, we further have

$$I(X;\mathbf{h}_{\text{item}}|C,\mathcal{O}) \lesssim \rho_{\alpha} I(X;e(Z)|C,\mathcal{O}) + \eta + \delta, \quad (14)$$

suggesting that reducing text-level leakage also improves representation-level privacy.

5 Experiments

5.1 Experimental Setup

Datasets and Side Information. We evaluate on two public recommendation benchmarks, MovieLens-1M[1] and Amazon-Book[2]. Ratings no less than 3 are treated as positive for MovieLens-1M, and ratings no less than 4 for Amazon-Book. We use Qwen2.5-14B (Yang et al., 2024) to generate multi-category item attributes from original titles and construct the side information vocabulary $\mathcal{W}$. Dataset statistics are shown in Table 1, with preprocessing and attribute construction details in Appendix B.1 and Appendix B.2.

Recommendation Models. We evaluate six representative recommenders: LightGCN (He et al., 2020), SASRec (Kang and McAuley, 2018), FREEDOM (Zhou and Shen, 2023), LightGT (Wei et al., 2023), TALLRec (Bao et al., 2023), and LLaRA (Liao et al., 2024). More details are provided in Appendix B.7.2.

Baselines. We compare with six baselines: None, raw Description, InferDPT (Tong et al., 2025), Text2Text (Feyisetan et al., 2020; Qu et al., 2021), SanText+ (Yue et al., 2021), and TextObfuscator (Zhou et al., 2023). Detailed baseline descriptions are provided in Appendix B.3.

Privacy Evaluation. We use Vec2Text (Morris et al., 2023) as the embedding inversion attacker to recover sensitive category labels from item embeddings (Fredrikson et al., 2015). For fair comparison, we align the average cosine similarity between original and obfuscated embeddings across methods, and evaluate attack performance under the same aligned CosSim (Morris et al., 2023; Li et al., 2023a; Huang et al., 2024). Attack training details are provided in Appendix B.4.

[1] https://grouplens.org/datasets/movielens/1m/
[2] https://jmcauley.ucsd.edu/data/amazon_v2/

Dataset	MovieLens-1M	Amazon-Book
Users	6,038	5,921
Items	3,629	5,658
Interactions	337,228	101,305
Sparsity	98.46%	99.66%

Table 1: Dataset statistics.

Implementation Details. Texts are encoded with BERT-base-uncased (Devlin et al., 2019). For traditional recommenders, embeddings are reduced to 128 dimensions by PCA (Abdi and Williams, 2010); for multimodal and LLM-based models, 768-dimensional embeddings are retained. Feature processing details, default hyperparameters, and reward schedules are reported in Appendix B.5 and Appendix B.6.

5.2 Overall Performance

We evaluate our method across three key dimensions: obfuscated variants quality, recommendation performance, and embedding inversion defense.

(i) Obfuscated Variants Quality. As shown in Table 2, our method produces valid variants for nearly all words and achieves the highest Spearman correlation on both datasets (0.198 on MovieLens-1M, 0.181 on Amazon-Book), substantially outperforming all baselines in preserving local semantic topology. In contrast, SanText+ yields low NLL but suffers from reduced diversity due to frequent special tokens, while InferDPT, Text2Text, and TOF generate noisy outputs with weak neighborhood consistency. These results confirm that our approach better balances obfuscation, fluency, and relational semantic preservation under comparable perturbation strength.

(ii) Recommendation Performance. Table 3 shows that our method consistently achieves the best or near-best downstream recommendation results across all model types on both datasets. Compared to the upper bound using raw descriptions, the performance drop is minimal, typically below 2% for traditional collaborative filtering models, indicating that key semantic relations in the side information are largely retained. Baseline token-level methods, however, degrade embedding quality and

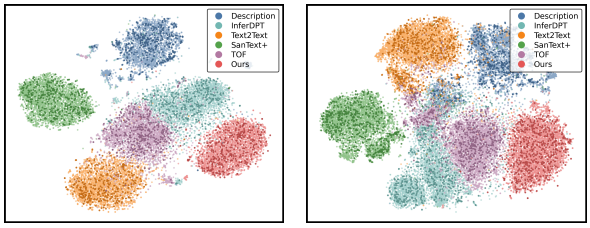
Dataset	Method	Valid†	CosSim†	KL‡	Spearman†	NLL‡	D-2‡	D-1†	LR†
MovieLens-1M	Description	–	1.0000	–	1.0000	1.9858	0.3958	0.1959	1.0000
	InferDPT	0.8276	0.6378	6.7429	0.0546	61.7860	0.5677	0.2708	0.7379
	Text2Text	0.9659	0.6246	6.1386	0.0857	39.8507	0.4446	0.3661	0.9674
	SanText+	0.9912	0.6266	8.7132	0.1612	3.5427	0.2668	0.1118	1.2536
	TOF	0.1000	0.6297	6.2628	0.0341	42.7631	0.5455	0.2585	1.0944
	Ours	0.9986	0.6293	7.9264	0.1981	5.7050	0.4257	0.1960	0.8758
Amazon-Book	Description	–	1.0000	–	1.0000	2.9403	0.3310	0.1539	1.0000
	InferDPT	0.8038	0.6762	6.2585	0.0538	50.2972	0.4773	0.2466	0.7777
	Text2Text	0.9719	0.6509	6.3891	0.0951	34.3471	0.3698	0.3242	0.9343
	SanText+	0.9899	0.6455	8.9465	0.1572	3.9354	0.2258	0.0992	0.9585
	TOF	0.9999	0.6668	6.2941	0.0359	42.5229	0.4624	0.2277	1.0812
	Ours	0.9983	0.6519	6.9642	0.1808	10.0452	0.3378	0.1661	0.9628

† For word-level metrics, ‡ For item description sentences.

Table 2: Obfuscated variants quality evaluation on MovieLens-1M and Amazon-Book.

Method	SASRec		LightGCN		FREEDOM		LightGT		TALLRec	LLaRA
	H@10	N@10	H@10	N@10	H@10	N@10	H@10	N@10	AUC	H@1
MovieLens-1M										
None	0.1116	0.0515	0.0814	0.0199	0.0793	0.0201	0.0758	0.0183	0.6828	0.21635
Description	0.1147	0.0524	0.0864	0.0212	0.0842	0.0205	0.0799	0.0199	0.6983	0.23509
InferDPT	0.1125	0.0516	0.0831	0.0201	0.0834	**0.0204**	0.0778	0.0194	0.6932	0.23422
Text2Text	0.1133	0.0522	0.0834	0.0201	0.0826	0.0202	0.0788	0.0195	0.6935	0.23250
SanText+	0.1139	0.0528	0.0837	0.0204	0.0829	0.0196	0.0796	0.0191	0.6928	0.22904
TOF	0.1134	0.0518	0.0824	0.0200	0.0816	0.0199	0.0753	0.0190	0.6877	0.23049
Ours	**0.1144**	**0.0529**	**0.0861**	**0.0215**	**0.0837**	0.0202	**0.0806**	**0.0201**	**0.6999**	**0.23753**
Amazon-Book										
None	0.1792	0.0877	0.0890	0.0399	0.0749	0.0319	0.0888	0.0394	0.7799	0.44844
Description	0.1866	0.0915	0.0937	0.0414	0.0781	0.0327	0.0920	0.0411	0.7963	0.52039
InferDPT	0.1821	0.0882	0.0924	0.0411	0.0747	0.0324	0.0895	0.0396	0.7928	0.52278
Text2Text	0.1772	0.0875	0.0914	0.0406	0.0722	0.0305	0.0905	0.0401	0.7903	0.51558
SanText+	0.1815	0.0893	0.0901	0.0406	0.0743	0.0312	0.0891	0.0402	0.7932	0.48681
TOF	0.1821	0.0885	0.0899	0.0407	0.0734	0.0312	0.0901	**0.0404**	0.7922	0.51079
Ours	**0.1883**	**0.0921**	**0.0939**	**0.0412**	**0.0762**	**0.0328**	**0.0910**	0.0403	**0.7950**	**0.52518**

Table 3: Recommendation performance on MovieLens-1M and Amazon-Book.



(a) MovieLens-1M (b) Amazon-Book

Figure 3: T-SNE visualization of item embeddings from original descriptions and method-specific obfuscated variants for each item.

disrupt implicit item–item relations, leading to inferior accuracy.

(iii) Embedding Inversion Defense. Table 4 reports that all obfuscation methods cut attack success rates by over 50% relative to the raw Description baseline, confirming that semantic perturbation effectively mitigates inversion attacks. Although baseline token-level methods offer partial protection, they degrade word quality and recommmend utility, as Tables 2 and 3 show. In contrast, our method achieves the SOTA defense performance while preserving word quality to effectively support recommendation.

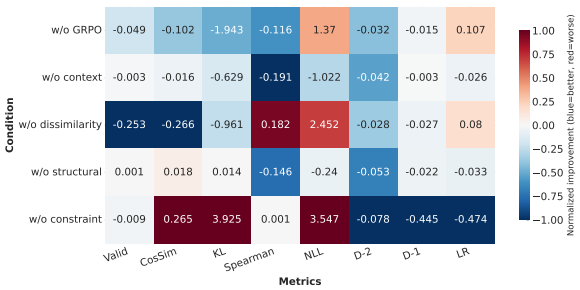
5.3 Ablation Study

We conduct ablation studies to evaluate the contribution of each component, where we set variants as: (1) **w/o GRPO**, directly generating outputs with the LLM; (2) **w/o context**, removing context examples and generating each word independently; (3) **w/o dissimilarity**, removing the individual dissimilarity reward; (4) **w/o structural**, removing the structural consistency reward; and (5) **w/o constraint**, removing the semantic constraint reward.

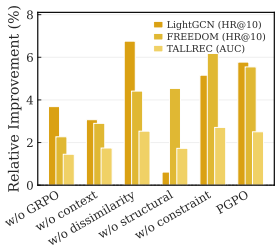
Results in Fig. 4 show that all components are important. The **w/o GRPO** variant reduces recommendation performance and increases attack success, confirming the benefit of group-based

Method	MovieLens-1M					Amazon-Book				
	EM	Precision	Recall	F1	Jaccard	EM	Precision	Recall	F1	Jaccard
Description	0.0641	0.2413	0.3638	0.2720	0.2116	0.0988	0.1829	0.1836	0.1832	0.1551
InferDPT	0.0309	0.0676	0.0848	0.0703	0.0583	0.0480	0.0950	0.0955	0.0952	0.0795
Text2Text	0.0276	0.0593	0.0808	0.0646	0.0523	0.0324	0.0727	0.0730	0.0728	0.0594
SanText+	0.0276	0.0839	0.1252	0.0941	0.0711	0.0226	0.0599	0.0605	0.0601	0.0477
TOF	0.0301	0.0672	0.0912	0.0722	0.0586	0.0384	0.0887	0.0896	0.0890	0.0722
Ours	0.0309	0.0662	0.0913	0.0721	0.0593	0.0314	0.0925	0.0933	0.0928	0.0724

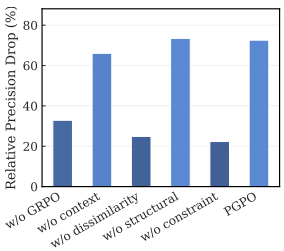
Table 4: Embedding inversion defense comparison across MovieLens-1M and Amazon-Book.



(a) Normalized variant quality metric changes relative to the full model.

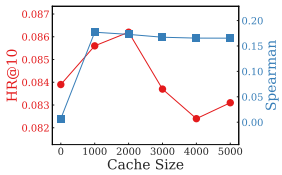


(b) Recommendation utility.

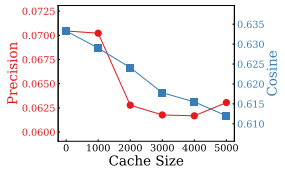


(c) Inversion defence.

Figure 4: Ablation study of each component.

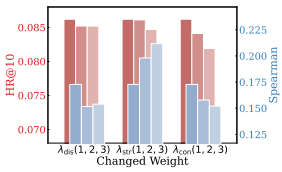


(a) Recommendation utility.

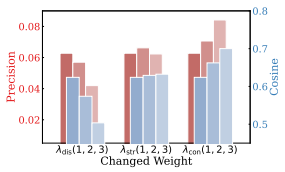


(b) Inversion defence.

Figure 5: Cache Size Sensitivity.



(a) Recommendation utility.



(b) Inversion defence.

Figure 6: Reward Weights Sensitivity.

policy optimization. The **w/o context** variant sharply weakens semantic preservation, showing that neighborhood-aware generation is essential for maintaining local topology. The **w/o dissimilarity** variant raises CosSim, indicating insufficient semantic shift and weaker privacy protection. The **w/o structural** variant harms both semantic correlation and recommendation utility, confirming the need to preserve local semantic order. The **w/o constraint** variant degenerates into just translation, causing high attack success and semantic inconsistency, which highlights the constraint reward's role in preventing degenerate shortcuts.

5.4 Parameter Sensitivity Analysis

Cache Size. Figs. 5, 7 and Table 6 show that a very small cache destroys topology, a moderate cache improves Spearman correlation to preserve local semantics, and further increases slightly degrade both, indicating diminishing returns and minor perturbations from early low-quality samples.

Prototype Word Count. Figs. 8, 9 and Table 7 show few prototype words suffice to preserve semantic relations, while adding more increases structure learning difficulty, degrades performance, and biases the model toward emphasizing dissimilarity.

Reward Weight Configuration. Figs. 6, 10 and Table 8 show that reward weights steer the model to generate diverse variants, thereby controlling privacy strength via similarity and recommendation performance via structure preservation, with both balanced through semantic constraint.

Detailed analyses are provided in Appendix C.

6 Conclusion

In this paper, we addressed input-level privacy for LLM-enhanced recommender systems. By shifting from semantic fidelity to **relational fidelity**, we proposed PGPO, a prototype-guided progressive obfuscation framework. Both theoretical analysis and experiments show that PGPO achieves a stronger privacy–utility trade-off than baselines while preserving recommendation utility. Future work may extend the framework to other modalities and decentralized deployment.

Limitations

Our framework currently treats side-information words with a unified obfuscation strategy, without explicitly assigning higher protection priority to more sensitive or rare words over generic ones. Future work may incorporate sensitivity-aware weighting to better allocate privacy budgets across words. Additionally, PGPO introduces extra computational cost due to LLM-based variant generation, which may be mitigated in future work through offline caching, batch generation, lightweight policy distillation, or retrieval-based reuse of high-quality obfuscation mappings. A risk is that the obfuscation could be misused to bypass content moderation by replacing prohibited items with innocuous.

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM conference on recommender systems*, pages 1007–1014.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Yiyi Chen, Heather Lent, and Johannes Bjerva. 2024. Text embedding inversion security for multilingual language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7808–7827.

Yiyi Chen, Qiongkai Xu, and Johannes Bjerva. 2025. Algen: Few-shot inversion attacks on textual embeddings via cross-model alignment and generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24330–24348.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM conference on recommender systems*, pages 1126–1132.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Robert M Fano and David Hawkins. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29(11):793–794.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. In *Proceedings of the 13th international conference on web search and data mining*, pages 178–186.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333.

Ahmed Frikha, Nassim Walha, Krishna Kanth Nakka, Ricardo Mendes, Xue Jiang, and Xuebing Zhou. 2025. Incognitext: Privacy-enhancing conditional text anonymization via llm-based private attribute randomization. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 2490–2501.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM conference on recommender systems*, pages 299–315.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648.

Yu-Hsiang Huang, Yuche Tsai, Hsiang Hsiao, Hong-Yi Lin, and Shou-De Lin. 2024. Transferable embedding inversion attack: Uncovering privacy risks in text embeddings without model queries. In *Proceedings of the 62nd Annual Meeting of the Association*

for Computational Linguistics (Volume 1: Long Papers), pages 4193–4205.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2021–2031.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Haoran Li, Mingshi Xu, and Yangqiu Song. 2023a. Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14022–14040.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 110–119.

Lei Li, Yongfeng Zhang, and Li Chen. 2023b. Personalized prompt learning for explainable recommendation. *ACM Transactions on Information Systems*, 41(4):1–26.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1785–1795.

Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024*, pages 3497–3508.

Sam Lin, Wenyue Hua, Zhenting Wang, Mingyu Jin, Lizhou Fan, and Yongfeng Zhang. 2025. Emojiprompt: Generative prompt obfuscation for privacy-preserving communication with cloud-based llms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 12342–12361.

Siwei Liu, Zaiqiao Meng, Craig Macdonald, and Iadh Ounis. 2023. Graph neural pre-training for recommendation with side information. *ACM Transactions on Information Systems*, 41(3):1–28.

Sichun Luo, Wei Shao, Yuxuan Yao, Jian Xu, Mingyang Liu, Qintong Li, Bowei He, Maolin Wang, Guanzhi Deng, Hanxu Hou, and 1 others. 2024. Privacy in llm-based recommendation: Recent advances and future directions. *arXiv preprint arXiv:2406.01363*.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. 2023. Split-and-denoise: Protect large language model inference with local differential privacy. *arXiv preprint arXiv:2310.09130*.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. Pmlr.

John Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M Rush. 2023. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12448–12460.

Peter Müllner, Elisabeth Lex, Markus Schedl, and Dominik Kowald. 2023. Differential privacy in collaborative filtering recommender systems: a review. *Frontiers in big Data*, 6:1249997.

Roman F Nalewajski. 2011. Elements of information theory. In *Perspectives in Electronic Structure Theory*, pages 371–395. Springer.

Oshri Naparstek. 2026. Token maturation: Autoregressive language generation via continuous token dynamics. *arXiv preprint arXiv:2601.04854*.

Jiangcheng Qin, Baisong Liu, and Jiangbo Qian. 2021. A novel privacy-preserved recommender system framework based on federated learning. In *Proceedings of the 2021 4th International Conference on Software Engineering and Information Management*, pages 82–88.

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. 2021. Natural language understanding with privacy-preserving bert. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1488–1497.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Congzheng Song and Ananth Raghunathan. 2020. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 377–390.

Charles Spearman. 1961. The proof and measurement of association between two things.

Robin Staab, Mark Vero, Mislav Balunovic, and Martin Vechev. 2025. Language models are advanced anonymizers. In *The Thirteenth International Conference on Learning Representations*.

Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi, Weiming Zhang, Nenghai Yu, Tianwei Zhang, and Zhikun Zhang. 2025. Inferdpt: Privacy-preserving inference for black-box large language models. *IEEE Transactions on Dependable and Secure Computing*.

Yu-Che Tsai, Hsiang Hsiao, Kuan-Yu Chen, and Shou-De Lin. 2026. Concept-aware privacy mechanisms for defending embedding inversion attacks. *arXiv preprint arXiv:2602.07090*.

Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM conference on recommender systems*, pages 225–232.

Cédric Villani and 1 others. 2009. *Optimal transport: old and new*, volume 338. Springer.

Yubo Wang, Min Tang, Nuo Shen, Shujie Cui, and Weiqing Wang. 2025. Privacy risks of llm-empowered recommender systems: An inversion attack perspective. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pages 812–821.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM international conference on web search and data mining*, pages 806–815.

Yinwei Wei, Wenqi Liu, Fan Liu, Xiang Wang, Liqiang Nie, and Tat-Seng Chua. 2023. Lightgt: A light graph transformer for multimedia recommendation. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 1508–1517.

Kilian Q Weinberger and Lawrence K Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research*, 10(2).

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, and 1 others. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

Xin Xin, Jiyuan Yang, Hanbing Wang, Jun Ma, Pengjie Ren, Hengliang Luo, Xinlei Shi, Zhumin Chen, and Zhaochun Ren. 2023. On the user behavior leakage from recommender system exposure. *ACM Transactions on Information Systems*, 41(3):1–25.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Tianyu Yang, Xiaodan Zhu, and Iryna Gurevych. 2025b. Robust utility-preserving text anonymization based on large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28922–28941.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman SM Chow. 2021. Differential privacy for text analytics via natural text sanitization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3853–3866.

Collin Zhang, John X Morris, and Vitaly Shmatikov. 2025. Universal zero-shot embedding inversion. *arXiv preprint arXiv:2504.00147*.

Yingji Zhang, Danilo S Carvalho, Ian Pratt-Hartmann, and André Freitas. 2023. Llamavae: Guiding large language model generation via continuous latent sentence spaces. *arXiv preprint arXiv:2312.13208*.

Xin Zhou, Yi Lu, Ruotian Ma, Tao Gui, Yuran Wang, Yong Ding, Yibo Zhang, Qi Zhang, and Xuan-Jing Huang. 2023. Textobfuscator: Making pre-trained language model a privacy protector via obfuscating word representations. In *Findings of the association for computational linguistics: ACL 2023*, pages 5459–5473.

Xin Zhou and Zhiqi Shen. 2023. A tale of two graphs: Freezing and denoising graph structures for multimodal recommendation. In *Proceedings of the 31st ACM international conference on multimedia*, pages 935–943.

A Theoretical Analysis

This appendix provides detailed derivations for Section 4. Let X denote the original side-information word, $\mathbf{U} = \phi(X)$ denote its public neighbor set retrieved from the static semantic graph, and Z denote the generated obfuscated variant. We consider

a strong attacker who knows the LLM parameters, the static semantic graph, the retrieval function ϕ, and the original vocabulary. The attacker may also observe the released obfuscated text or the set of released variants.

We define $\mathcal{O}$ as all attacker-observable or inferable information, including the released variants, the public graph, the original vocabulary, and any final-position information if available. The hidden trajectory Γ contains only residual stepwise generation factors that remain unobserved after conditioning on $\mathcal{O}$, such as unreleased cache states, queue states, sampled candidate groups, source-variant alignments inside history samples, and stochastic decoding paths. Thus, the privacy objective is to reduce $I(X; Z|\mathcal{O})$, equivalently increasing $H(X|Z, \mathcal{O})$.

A.1 Path-dependent Generation Trajectory

PGPO does not rely on hiding the released variants themselves. Instead, its privacy effect comes from the path-dependent generation process that determines how each prompt context is assembled and how each finalized variant is selected. At step t, let the processing order be π and let $X_t = W_{\pi_t}$ be the word currently being obfuscated. Its static neighbor set is public and fixed by the precomputed graph:

$$\mathbf{U}_t = \phi(X_t). \tag{15}$$

Let $\mathcal{C}_{t-1}$ denote the per-word cache before step t, and let $\mathcal{Q}_{t-1}$ denote the global recent queue. The actual history samples injected into the prompt are

$$\mathcal{H}_t = \{(x_j, z_j) \in \mathcal{C}_{t-1} : x_j \in \mathbf{U}_t,\ x_j \in \mathcal{Q}_{t-1}\}. \tag{16}$$

Although $\mathbf{U}_t$ is public, the realized $\mathcal{H}_t$ is path-dependent. Different processing orders, cache states, and queue states may cause the same word and the same static neighbor set to receive different history samples.

Given $\mathcal{H}_t$, the LLM samples a group of candidates:

$$\tilde{Z}_t^{(g)} \sim p_{\theta_t}(\cdot|X_t, \mathbf{U}_t, \mathcal{H}_t; \xi_t), \quad g = 1, \ldots, G, \tag{17}$$

where ξ_t denotes stochastic decoding randomness. The finalized variant is selected according to the total reward in Eq. (5):

$$Z_t = \text{Select}\left(\{\tilde{Z}_t^{(g)}\}_{g=1}^{G}, R_{\text{total}}\right). \tag{18}$$

Then the cache and queue are updated:

$$\mathcal{C}_t = \mathcal{C}_{t-1} \cup \{(X_t, Z_t)\}, \tag{19}$$

$$\mathcal{Q}_t = \text{UpdateQueue}(\mathcal{Q}_{t-1}, X_t). \tag{20}$$

The recursively coupled process can be summarized as

$$\pi \to \mathcal{C}_{t-1}, \mathcal{Q}_{t-1} \to \mathcal{H}_t \to p_{\theta_t}(Z_t|X_t, \mathbf{U}_t, \mathcal{H}_t) \to Z_t \to \mathcal{C}_t, \mathcal{Q}_t. \tag{21}$$

We denote the residual hidden trajectory for the target step by

$$\Gamma_t = (\pi_{1:t}, \mathcal{C}_{0:t-1}, \mathcal{Q}_{0:t-1}, \mathcal{H}_{1:t}, \xi_{1:t}). \tag{22}$$

For notational simplicity, we omit the step index and write

$$Z \sim p(\cdot|X, \mathcal{O}, \Gamma). \tag{23}$$

This definition avoids treating released variants themselves as hidden. The attacker may observe the released variants, but this does not reveal the ordered trajectory of source-variant pairs used in the cache and prompt. Therefore, the attacker must marginalize over possible residual trajectories.

A.2 Trajectory Leakage Decomposition

Lemma 1 (Trajectory Leakage Decomposition). For the obfuscated output $Z \sim p(\cdot|X, \mathcal{O}, \Gamma)$,

$$I(X; Z|\mathcal{O}) = H(Z|\mathcal{O}) - \mathbb{E}\big[H(Z|X, \mathcal{O}, \Gamma)\big] - I(Z; \Gamma|X, \mathcal{O}). \tag{24}$$

Proof. By definition,

$$I(X; Z|\mathcal{O}) = H(Z|\mathcal{O}) - H(Z|X, \mathcal{O}). \tag{25}$$

For the residual trajectory Γ,

$$I(Z; \Gamma|X, \mathcal{O}) = H(Z|X, \mathcal{O}) - \mathbb{E}\big[H(Z|X, \mathcal{O}, \Gamma)\big]. \tag{26}$$

Thus,

$$H(Z|X, \mathcal{O}) = \mathbb{E}\big[H(Z|X, \mathcal{O}, \Gamma)\big] + I(Z; \Gamma|X, \mathcal{O}). \tag{27}$$

Substituting this equality into the definition of $I(X; Z|\mathcal{O})$ gives the result. □

The term $I(Z; \Gamma|X, \mathcal{O})$ measures how much the output distribution changes across residual generation trajectories. For a fixed pair $(x, \mathcal{O})$, define the trajectory-mixture distribution as

$$p_{x,\mathcal{O}}(z) = \sum_{\gamma} \pi(\gamma|x, \mathcal{O}) p(z|x, \mathcal{O}, \gamma). \tag{28}$$

Then

$$I(Z;\Gamma|X=x,\mathcal{O}) = \sum_{\gamma} \pi(\gamma|x,\mathcal{O}) \cdot \mathrm{KL}\Big(p(\cdot|x,\mathcal{O},\gamma)\|p_{x,\mathcal{O}}(\cdot)\Big). \tag{29}$$

Therefore, if different residual trajectories induce different output distributions, the attacker faces additional uncertainty from marginalizing over Γ. If the exact processing order, cache states, queue states, source-variant alignments, candidate groups, and sampling path are all revealed or inferable, then they are absorbed into $\mathcal{O}$ and the remaining trajectory term vanishes.

A.3 Semantic-Mode Entropy

The following analysis does not assume that the LLM explicitly samples a discrete latent variable. Instead, semantic modes are introduced as an analytical partition of the output space. For a fixed condition $(x,\mathcal{O},\gamma)$, let $\{\mathcal{Z}_1,\dots,\mathcal{Z}_K\}$ be a hard or soft partition of possible outputs according to distinguishable semantic directions of obfuscation. Let $M \in \{1,\dots,K\}$ denote the semantic-mode assignment of Z.

For each fixed condition $(x,\mathcal{O},\gamma)$, define

$$\alpha_m = P(M=m|x,\mathcal{O},\gamma), \tag{30}$$

and

$$p_m(\cdot) = P(Z=\cdot|M=m,x,\mathcal{O},\gamma). \tag{31}$$

Here α_m is the weight of semantic mode m, and p_m is the intra-mode output distribution.

Lemma 2 (Semantic-Mode Entropy). Partition the output space into semantic modes with weights α_m and intra-mode distributions p_m. Then

$$\mathbb{E}\big[H(Z|X,\mathcal{O},\Gamma)\big] = \mathbb{E}\Big[H(\alpha) + \sum_m \alpha_m H(p_m)\Big] - \epsilon, \tag{32}$$

where

$$\epsilon = \mathbb{E}[H(M|Z,X,\mathcal{O},\Gamma)]. \tag{33}$$

Proof. For fixed $(x,\mathcal{O},\gamma)$, applying the chain rule to (M,Z) gives

$$\begin{aligned} H(M,Z|x,\mathcal{O},\gamma) &= H(M|x,\mathcal{O},\gamma) + H(Z|M,x,\mathcal{O},\gamma), \\ H(M,Z|x,\mathcal{O},\gamma) &= H(Z|x,\mathcal{O},\gamma) + H(M|Z,x,\mathcal{O},\gamma). \end{aligned} \tag{34}$$

Since M follows the categorical distribution α,

$$H(M|x,\mathcal{O},\gamma) = H(\alpha), \tag{35}$$

and

$$H(Z|M,x,\mathcal{O},\gamma) = \sum_m \alpha_m H(p_m). \tag{36}$$

Combining the two decompositions yields

$$\begin{aligned} H(Z|x,\mathcal{O},\gamma) = H(\alpha) &+ \sum_m \alpha_m H(p_m) \\ &- H(M|Z,x,\mathcal{O},\gamma). \end{aligned} \tag{37}$$

Taking expectation over $(X,\mathcal{O},\Gamma)$ gives the result. □

If the semantic partition is deterministic, then M is determined by Z and $H(M|Z,x,\mathcal{O},\gamma) = 0$. For soft or overlapping semantic modes, the residual uncertainty is positive and is captured by ϵ. Therefore, ϵ measures the ambiguity of assigning a generated output to a semantic mode.

A.4 Attack Success Bound

The rewards do not directly optimize the entropy terms in Lemma 2. Instead, they shape the generation policy and affect the leakage decomposition through semantic diversity, intra-mode uncertainty, residual mode ambiguity, and trajectory uncertainty. Let p_λ denote the generation distribution induced by the reward-shaped policy optimized with Eq. (5). Under p_λ, the corresponding semantic-mode weights, intra-mode distributions, residual ambiguity, and trajectory divergence are denoted by $\alpha_m^\lambda, p_m^\lambda, \epsilon^\lambda, I_\lambda(Z;\Gamma|X,\mathcal{O})$.

Theorem 1 (Attack Success Bound for PGPO). For a finite original-word set $\mathcal{X}$ and an approximately uniform prior

$$P_{\text{succ}} \le \frac{1}{\log_2|\mathcal{X}|}\Bigg(\epsilon^\lambda - \mathbb{E}_\lambda\Big[H(\alpha^\lambda) + \sum_m \alpha_m^\lambda H(p_m^\lambda)\Big] + H_\lambda(Z|\mathcal{O}) - I_\lambda(Z;\Gamma|X,\mathcal{O}) + 1\Bigg). \tag{38}$$

Proof. Substituting Lemma 2 into Lemma 1 gives

$$\begin{aligned} I_\lambda(X;Z|\mathcal{O}) =& H_\lambda(Z|\mathcal{O}) + \epsilon^\lambda - I_\lambda(Z;\Gamma|X,\mathcal{O}) \\ &- \mathbb{E}_\lambda\left[H(\alpha^\lambda) + \sum_m \alpha_m^\lambda H(p_m^\lambda)\right]. \end{aligned} \tag{39}$$

Let $\widehat{X}(Z,\mathcal{O})$ be the attacker's estimator of the original input X, and let

$$P_{\text{succ}} = \Pr[\widehat{X}(Z,\mathcal{O}) = X]. \tag{40}$$

For a finite candidate space $\mathcal{X}$ and an approximately uniform prior over X after conditioning on $\mathcal{O}$, Fano's inequality (Fano and Hawkins, 1961; Nalewajski, 2011) gives

$$P_{\text{succ}} \leq \frac{I_\lambda(X; Z|\mathcal{O}) + 1}{\log_2 |\mathcal{X}|}. \tag{41}$$

Substituting Eq. (39) into the above inequality gives the result. □

The bound shows that PGPO can reduce attack success by increasing semantic-mode diversity, preserving valid intra-mode uncertainty, controlling residual mode ambiguity, and forcing the attacker to marginalize over residual generation trajectories. When $H_\lambda(Z|\mathcal{O})$ is controlled by the output format and category constraints, increasing the negative terms in the bound tightens the attack success upper bound.

A.5 Reward Interpretation

The reward-induced bound explains the roles of the three rewards. The individual dissimilarity reward in Eq. (6) encourages variants to move away from the original word along diverse semantic directions. This tends to increase the empirical coverage of semantic modes and hence the mode-weight entropy $H(\alpha^\lambda)$.

The structural consistency reward in Eq. (8) preserves local relational order among history samples. By making semantic modes more consistently assignable to relationally meaningful regions, it helps control the residual ambiguity ϵ^λ, which appears as a positive term in the attack success bound.

The semantic constraint reward in Eq. (9) anchors generated variants to category-valid semantic manifolds. It prevents diversity from drifting into invalid or out-of-category regions, thereby preserving meaningful intra-mode uncertainty $\sum_m \alpha_m^\lambda H(p_m^\lambda)$ rather than producing degenerate noise.

Overall, the reward effects can be summarized as

$$\begin{aligned} R_{\text{dis}} &\Rightarrow H(\alpha^\lambda), \\ R_{\text{str}} &\Rightarrow \epsilon^\lambda, \\ R_{\text{con}} &\Rightarrow \sum_m \alpha_m^\lambda H(p_m^\lambda). \end{aligned} \tag{42}$$

These arrows indicate mechanism-level influence rather than mathematical equivalence.

A.6 Representation-level Leakage

Finally, we connect text-level obfuscation to the final item representation used by recommender models. Let

$$\mathbf{h}_{\text{item}} = g(e(Z), S), \tag{43}$$

where $e(Z)$ is the text embedding of the obfuscated variant and S denotes collaborative signals. Let C denote category or contextual information used in the representation process. Since $\mathbf{h}_{\text{item}}$ is obtained from $(e(Z), S)$ through the representation function g, the data processing inequality gives Eq. (13).

By the chain rule,

$$\begin{aligned} I(X; e(Z), S|C, \mathcal{O}) = I(X; e(Z)|C, \mathcal{O}) \\ + I(X; S|e(Z), C, \mathcal{O}). \end{aligned} \tag{44}$$

If collaborative signals contain limited residual information about X, we assume

$$I(X; S|e(Z), C, \mathcal{O}) \leq \eta. \tag{45}$$

If the downstream representation introduces at most δ additional approximation slack and the text branch contributes with effective coefficient ρ_α, then Eq. (14) follows.

Therefore, reducing text-level leakage also reduces the information available through the final item representation, except for residual collaborative information and model approximation slack.

B Experimental Settings

B.1 Dataset Preprocessing

MovieLens-1M. We keep ratings ≥ 4 as positive feedback and retain only the latest interaction for duplicate user-item pairs. Each item keeps one anchor interaction, while the remaining interactions are randomly kept with probability 0.4. Users with fewer than 3 interactions are fully assigned to the training set.

Amazon-Book. Interactions from January to March 2018 are first filtered to retain users and items with at least 3 interactions. We then randomly sample 10,000 items and keep their associated interactions. After sampling, users and items with fewer than 3 interactions are removed again. Ratings ≥ 4 are treated as positive feedback.

B.2 Item-side Attribute Construction

We use Qwen2.5-14B to generate fine-grained, multi-category textual attributes from item titles. For each item i, the LLM yields an attribute set $\mathcal{A}_i$, where each attribute word is assigned to a category $c \in \mathcal{C}$. Attributes are organized per category as $\text{ItemEdges}[i] = \{c : \mathcal{W}_{i,c}\}$, and the initial vocabulary is $\mathcal{W}_{\text{init}} = \bigcup_i \mathcal{A}_i$.

B.2.1 Prompt Templates

```
# MovieLens-1M Movie Prompt
You are a movie database curator.

Introduce the movie titled "{item_name}"
     in valid JSON format with exactly
    these 9 keys:
"Genre", "Actors", "Country", "Character
    ", "PlotAndTheme",
"Mood", "Critical Acclaim", "Production
    Quality", "Soundtrack"

For each key, output 3-5 concise,
    distinctive English keywords.
Each keyword should be 2-5 words long
    and discriminative.

Output ONLY the JSON object.
```

```
# Amazon-Book Prompt
You are a book metadata curator.

Introduce the book titled "{item_name}"
    in valid JSON format with exactly
    these 9 keys:
"Genre", "Author", "PublicationContext",
     "CharacterOrFocus", "PlotAndTheme",
"ToneAndStyle", "AudienceAndUseCase", "
    DistinctiveElements", "RelatedTopics
    "

For each key, output 3-5 concise,
    distinctive English keywords.

Output ONLY the JSON object.
```

B.2.2 Category and Keyword Pruning

After generating attributes, we prune redundant categories and retain discriminative keywords by: (1) computing IDF (Salton and Buckley, 1988) for each keyword, (2) scoring each category by average keyword IDF and pairwise embedding similarity, (3) iteratively removing categories with high redundancy and low score, and (4) keeping the top-τ keywords per retained category for each item. The resulting retained categories and keyword counts are reported in Table 5.

Dataset	Category	#Keywords
MovieLens-1M	PlotAndTheme	6384
	Actors	5122
	Soundtrack	4039
	Production Quality	2893
	Total	18398
Amazon-Book	PlotAndTheme	7637
	Author	6242
	PublicationContext	4513
	ToneAndStyle	3725
	Total	21678

Table 5: Retained categories and keyword counts after category pruning. The total number is not equal to the sum of individual categories because some keywords appear in multiple categories.

B.3 Baselines

We compare our method with the following baselines:

- **None:** Item embeddings constructed without using side information, serving as the lower bound.
- **Description:** Raw side information, serving as the utility upper bound.
- **InferDPT:** A differentially private text perturbation method that injects Laplacian noise during text encoding (Tong et al., 2025), used directly without LLM processing.
- **Text2Text:** A sequence-to-sequence text obfuscation method that perturbs embeddings under local differential privacy and decodes them back (Feyisetan et al., 2020; Qu et al., 2021).
- **SanText+:** An improved text obfuscation method that optimizes substitution word selection to enhance obfuscation quality (Yue et al., 2021).
- **TextObfuscator (TOF):** A rule-based text obfuscation tool that replaces sensitive words for privacy protection (Zhou et al., 2023).

B.4 Attack Model and Fair Comparison

We adopt the Vec2Text embedding inversion attack (Morris et al., 2023) to recover sensitive category labels from item embeddings. The attack training data does not overlap with the items used in the main recommendation experiments. For MovieLens-1M, we sample 80,000 movie tags from MovieLens-25M[3] and MovieLens-32M[4]. For Amazon-Book, we sample 10,000 disjoint items. Embeddings are generated using the same pipeline as in the main experiments.

Since different obfuscation methods induce different degrees of semantic displacement, directly comparing attack success rates can be misleading. For fair evaluation, we tune the hyperparameters of each method, such as dissimilarity reward and DP noise, so that the average cosine similarity between

[3] https://grouplens.org/datasets/movielens/25m/
[4] https://grouplens.org/datasets/movielens/32m/

original and obfuscated embeddings is aligned. All inversion attacks are evaluated under this aligned CosSim setting.

B.5 Feature Processing Pipeline

All original and obfuscated texts are encoded into 768-dimensional embeddings using BERT-base-uncased. For traditional recommendation models, PCA is applied to reduce the dimension to 128. For multimodal and LLM-based models, the full 768-dimensional embeddings are retained and integrated through learnable projection layers.

B.6 Training Configuration

For MovieLens-1M, the reward weights are $(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}}) = (1, 2, 1)$. For Amazon-Book, the reward weights are $(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}}) = (2, 3, 2)$.

For both datasets, the prototype word count is set to 5000, the cache size is set to 2000, and the warm-up length is

$$T_{\text{warmup}} = 2 \left\lceil \frac{|C_{\text{Prototype}}|}{\text{batch_size}} \right\rceil. \tag{46}$$

We adopt a phased reward schedule. During the first $\lceil |C_{\text{Prototype}}|/\text{batch_size} \rceil$ steps, only the semantic constraint and individual dissimilarity rewards are optimized by setting $\lambda_{\text{str}} = 0$. After sufficient high-quality anchors are cached, the structural consistency reward is restored to its default weight.

All ablation and parameter sensitivity experiments are conducted on MovieLens-1M. For ablation studies, we use the same hyperparameter settings as the MovieLens-1M main experiment unless otherwise specified. For parameter sensitivity analysis, we vary one factor at a time while keeping the others fixed. In cache size sensitivity, we set $(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}}) = (1, 1, 1)$ and vary the cache size in $\{0, 1000, 2000, 3000, 4000, 5000\}$. In prototype word count sensitivity, we use $(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}}) = (1, 2, 1)$ and vary the prototype word count in $\{1250, 2500, 5000, 7500, 10000\}$. In reward weight sensitivity, we fix the cache size to 2000 and start from $(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}}) = (1, 1, 1)$. We then vary λ_{dis}, λ_{str}, and λ_{con} individually, with each weight taking values from $\{1, 2, 3\}$.

The batch size is set to 2, and all PGPO training and generation experiments are conducted on a single NVIDIA GeForce RTX 4090 GPU with 24GB memory.

B.7 Evaluation Metrics

We evaluate our method from three perspectives: obfuscated variants quality, recommendation performance, and embedding inversion defense

B.7.1 Obfuscated Variants Quality

We evaluate generation quality at both the word and context levels.

For word-level evaluation, we encode original words and generated variants with BERT-base-uncased embeddings. We report Distinct-1 (Li et al., 2016) to measure token-level diversity, and use Spearman correlation to evaluate whether the local semantic structure is preserved after transformation. In addition, we define **Valid**, **Spearman** and **LR** for our task.

Valid measures the proportion of generated variants that are not identical to their original words:

$$\text{Valid} = 1 - \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(v_i = o_i), \tag{47}$$

where o_i and v_i denote the original word and its generated variant, respectively.

Spearman measures whether the local semantic structure is preserved after transformation. For each original word, we retrieve its top-k nearest neighbors within the same category in the original BERT embedding space, and compare the similarity rankings before and after transformation using Spearman rank correlation (Spearman, 1961). We average the correlation over all valid samples:

$$\text{Spearman} = \frac{1}{N} \sum_{i=1}^{N} \rho_i, \tag{48}$$

where ρ_i denotes the rank correlation for the i-th sample. A higher value indicates better preservation of the local neighborhood structure.

LR measures the average length ratio between generated variants and original words:

$$\text{LR} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{len}(v_i)}{\text{len}(o_i)}. \tag{49}$$

For context-level evaluation, we replace the original words in downstream contexts with their generated variants and evaluate the resulting texts. We report Distinct-2 (Li et al., 2016) to measure contextual diversity. We use Qwen3-4B (Yang et al., 2025a) to compute KL divergence (Nalewajski, 2011) and NLL (Bengio et al., 2003), where

Cache Size	Valid†	CosSim†	KL‡	Spearman†	NLL‡	D-2‡	D-1†	LR†
0	0.9982	0.6333	8.1727	0.0070	4.3177	0.3828	0.1430	0.9021
1000	0.9990	0.6290	7.9774	0.1771	4.3254	0.3996	0.1668	0.8787
2000	0.9991	0.6241	7.9014	0.1728	5.8615	0.4112	0.1783	0.8706
3000	0.9990	0.6178	7.7505	0.1672	6.7148	0.4327	0.2079	0.8439
4000	0.9994	0.6155	7.7191	0.1655	7.2066	0.4469	0.2278	0.8571
5000	0.9996	0.6120	7.7077	0.1653	8.0208	0.4456	0.2258	0.8619

† For word-level metrics, ‡ For item description sentences.

Table 6: Parameter sensitivity analysis on cache size.

KL measures the average divergence between token probability distributions of original and transformed contexts, and NLL (from LM cross-entropy loss) evaluates the fluency of the transformed contexts.

B.7.2 Recommendation Performance

For traditional recommendation models, including collaborative filtering and multimodal recommendation methods, we follow a chronological 8:1:1 data split and report HR@10 and NDCG@10. For SASRec, the latest 20 interacted items are used as the input sequence. For LLM-based recommendation models, we adopt the dataset splits and evaluation metrics from their original papers: TALLRec (Bao et al., 2023) is evaluated with AUC, while LLaRA (Liao et al., 2024) is evaluated with HR@1. For all experiments, we select the best-performing model on the validation set for testing, and all results are averaged over 5 repeated runs.

B.7.3 Embedding Inversion Defense

To evaluate embedding inversion defense, we formulate the problem as multi-label attribute prediction. Given the predicted attribute set P_i and the ground-truth attribute set T_i for each sample, we evaluate attack and defense effectiveness with standard multi-label metrics, including EM, Precision, Recall, F1, and Jaccard. EM measures whether the predicted attribute set exactly matches the ground-truth set. Precision, Recall, F1, and Jaccard measure prediction correctness, coverage, their harmonic mean, and set-level similarity, respectively.

B.8 Artifact Use

We use publicly available datasets, pretrained models, recommendation models, attack tools, and baseline implementations only for academic research and evaluation, following their corresponding licenses, terms of use, and intended research purposes. The derived side-information attributes and obfuscated variants are used solely for privacy-preserving recommendation experiments and are not intended for commercial deployment, user profiling, or redistribution outside research contexts. Our released code will include license information and reproduction instructions under the access conditions of the original datasets and models.

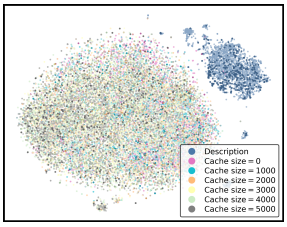


Figure 7: T-SNE of embeddings across cache sizes.

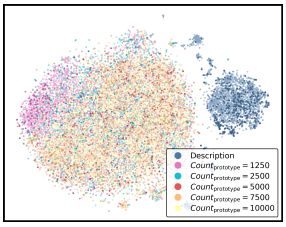


Figure 8: T-SNE of embeddings across prototype word counts.

C Additional Analysis

C.1 Cache Size

Fig. 7 shows that the global embedding distribution remains largely stable across cache sizes, while individual samples shift locally. As shown in Table 6 and Fig. 5, removing the cache severely weakens local structure preservation and leads to poor recommendation performance. A moderate cache substantially improves Spearman correlation and recommendation accuracy while reducing attack effectiveness, indicating that cached history samples provide useful contextual anchors for preserving local topology and promoting sufficient semantic shift. However, further increasing the cache size slightly decreases structural preservation and embedding similarity, while increasing NLL. This suggests that an overly large cache may bias generation toward excessively low-similarity variants, leading to diminishing returns for the overall privacy–utility trade-off.

C.2 Prototype Word Count

Fig. 8 shows that the overall embedding layout remains relatively stable across different prototype

Count	Valid†	CosSim†	KL‡	Spearman†	NLL‡	D-2‡	D-1†	LR†
1250	0.9951	0.6309	7.3785	0.2352	8.3621	0.4705	0.3270	0.7863
2500	0.9970	0.6442	7.5392	0.2353	6.9372	0.4261	0.2240	0.8422
5000	0.9986	0.6293	7.9264	0.1981	5.7050	0.4257	0.1960	0.8758
7500	0.9976	0.6228	7.7439	0.1650	6.8084	0.4333	0.2117	0.8518
10000	0.9986	0.6209	7.7050	0.1308	7.2492	0.4576	0.2323	0.8692

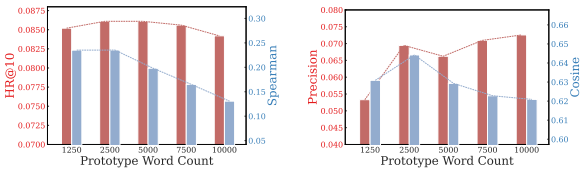
† For word-level metrics, ‡ For item description sentences.

Table 7: Parameter sensitivity analysis on prototype count.

$(\lambda_{\text{dis}}, \lambda_{\text{str}}, \lambda_{\text{con}})$	Valid†	CosSim†	KL‡	Spearman†	NLL‡	D-2‡	D-1†	LR†
$(1, 1, 1)$	0.9991	0.6241	7.9014	0.1728	5.8615	0.4112	0.1783	0.8706
$(2, 1, 1)$	0.9995	0.5740	7.8329	0.1517	7.3188	0.4650	0.2788	0.8191
$(3, 1, 1)$	0.9986	0.5036	8.3222	0.1539	7.9570	0.5864	0.6248	0.7775
$(1, 2, 1)$	0.9986	0.6293	7.9264	0.1981	5.7050	0.4257	0.1960	0.8758
$(1, 3, 1)$	0.9976	0.6324	7.6278	0.2118	6.9764	0.4309	0.2107	0.8537
$(1, 1, 2)$	0.9945	0.6624	7.6043	0.1579	6.2802	0.4120	0.1855	0.8589
$(1, 1, 3)$	0.9797	0.7001	7.4114	0.1521	6.1095	0.4171	0.1913	0.8755

† For word-level metrics, ‡ For item description sentences.

Table 8: Parameter sensitivity analysis on reward weight configuration.



(a) Recommendation utility. (b) Inversion defence.

Figure 9: Prototype Word Count Sensitivity.

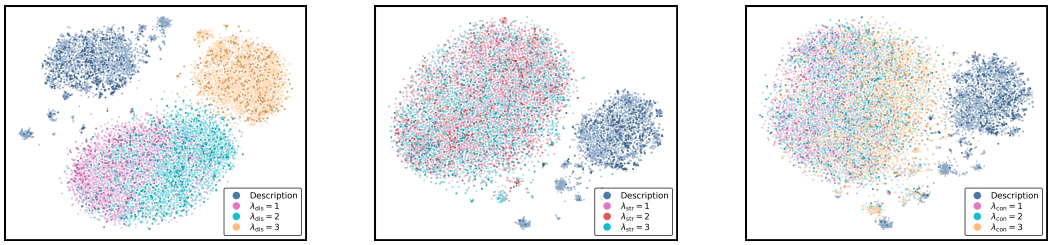
counts, although individual samples exhibit local shifts. As reported in Table 7 and Fig. 9, a small set of prototypes is already sufficient to establish effective structural anchors, preserving local semantic relations while producing variants with clear semantic displacement. Increasing the prototype count moderately improves fluency and slightly benefits recommendation performance. However, using too many prototypes makes the structure-learning process harder and encourages stronger semantic deviation, which lowers Spearman correlation and may slightly harm downstream utility. These results indicate that prototype selection should emphasize high-influence anchors rather than exhaustive coverage.

C.3 Reward Weight Configuration

Individual Dissimilarity Reward. Fig. 10a shows that increasing the weight of the individual dissimilarity reward noticeably shifts the embedding distribution. As shown in Table 8 and Fig. 10, larger dissimilarity weights substantially reduce CosSim, indicating stronger semantic displacement and improved privacy protection. The accompanying increase in D-1 and D-2 further suggests that this reward promotes more diverse variants. However, excessive emphasis on dissimilarity slightly increases NLL and reduces Spearman correlation, reflecting a mild loss in fluency and local structural preservation. Overall, this reward mainly controls privacy strength by pushing variants away from their originals while keeping downstream performance relatively stable.

Structural Consistency Reward. Fig. 10b shows that the structural consistency reward mainly affects local sample positions rather than the global embedding layout. As reported in Table 8 and Fig. 10, increasing its weight improves Spearman correlation, confirming that this reward helps preserve local semantic topology. Meanwhile, D-1 and D-2 increase slightly, and NLL rises only moderately, suggesting that the model can strengthen relational consistency without severely sacrificing diversity or fluency. Although CosSim increases slightly because the relative contribution of dissimilarity is reduced, the overall privacy–utility balance remains stable. Thus, this reward primarily controls relational fidelity by maintaining neighborhood structure.

Semantic Constraint Reward. Fig. 10c shows that the semantic constraint reward affects both global layout and local sample positions. As shown in Table 8 and Fig. 10, increasing its weight encourages generated variants to stay closer to the category-specific semantic manifold, improving semantic validity and preventing degenerate outputs. However, overly strong semantic constraints



(a) Varying R_{dis} weight. (b) Varying R_{str} weight. (c) Varying R_{con} weight.

Figure 10: T-SNE visualizations of embeddings under different reward weight configurations.

make the variants closer to the original words, as reflected by higher CosSim and reduced diversity. This weakens privacy protection because the individual dissimilarity objective becomes relatively less influential. Therefore, the semantic constraint reward is most effective when used as a regularizer that preserves validity without overly suppressing semantic shift.

C.4 Further Observation

Across the above analyses, we observe that once the local semantic skeleton is sufficiently preserved, further improving fine-grained semantic fidelity brings limited gains to recommendation performance. This supports our relational fidelity principle: effective recommendation does not require literal semantic preservation, but rather a stable relative structure among item attributes. The remaining utility is likely supported by recommender model and complementary collaborative signals.