876: Defending against Attribute Inference Attacks in Post-Training of Recommendation Systems via Unlearning

Wenhan Wu[1], Yili Gong[1], Jiawei Jiang[1], Chuang Hu[2,*], Xiaobo Zhou[2] and Dazhao Cheng[1,*]
1 School of School Science, Wuhan University, Wuhan, China
2 State Key Laboratory of Internet of Things for Smart City, University of Macau, Macau, Macau SAR
* Corresponding Authors

Introduction and Motivation

Privacy Issues in Recommender Systems.

Recommendation systems (RSs) utilize user-item interaction data to train models to provide personalized service recommendations. They are generally constructed based on user and item embeddings. However, existing studies have shown that these systems are vulnerable to Attribute Inference Attacks (AIAs), where attackers leverage threat models to infer sensitive user attributes like age or gender from embeddings generated through collaborative filtering (CF). These inferred attributes can expose sensitive user information, leading to a significant privacy breach, including unauthorized profiling and discriminatory practices supported by these characteristics of users.

TABLE I: Model Utility and Attribute Inference Attack Results via Data-Input Level Recommendation Unlearning.

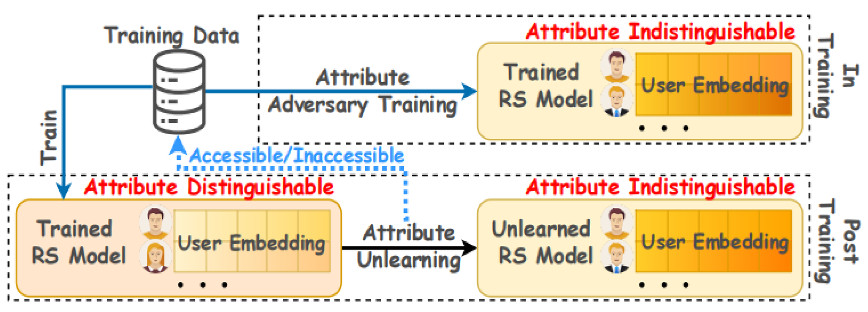
Stage	Utility		Attribute Privacy		
	NDCG@10↑	HR@10↑	Gender	Age	Location
Original	0.632	0.610	0.751	0.604	0.588
Unlearned	0.603	0.589	0.728	0.632	0.609
Random Attacker			0.500	0.143	0.167

Limitations of Existing In-training Protection Mechanisms.

Most existing attribute-wise protection falls into in-training methods, which means the protection is carried during training. The sensitive attributes must be specified in advance, making it hard to handle changing or new privacy needs. More specifically, the adversarial training-based methods adds an extra adversarial attribute-inference module such as discriminator network. The data modification-based methods alter the training data by adding dummy negatively correlated items to user profiles. It distorts the original user-item distribution and will decrease the recommendation accuracy.

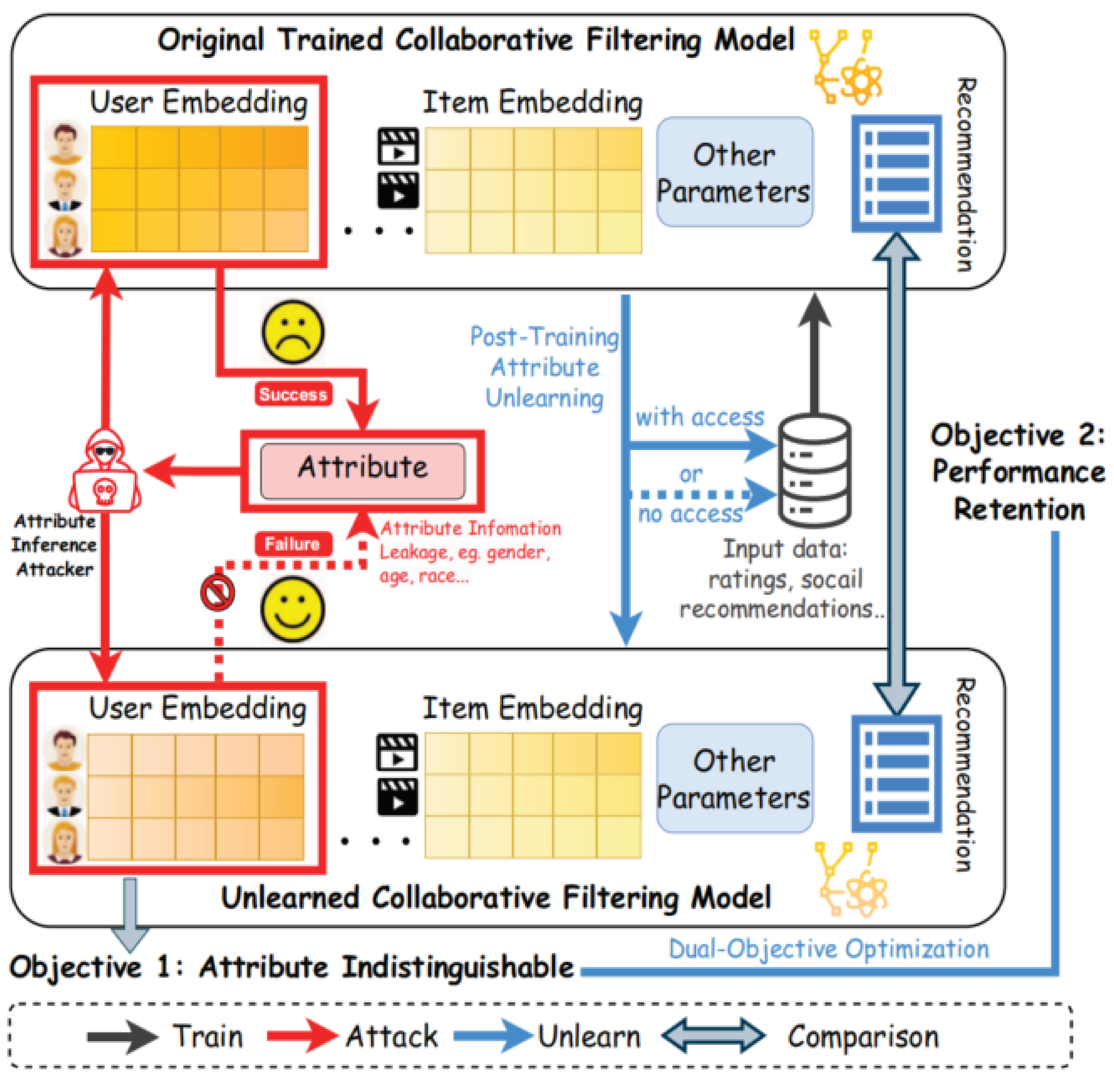
Why Attribute Unlearning?

- AU, as a post-training method, is more flexible without the modification to the training phase. Also, it is adaptable to dynamic user privacy requests.



- Current machine unlearning methods struggle to decouple sensitive attributes from the model. Attribute unlearning (AU) can effectively remove such attributes, thus defending against AIAs.

System Overview



- **Objective 1:** Attribute Indistinguishable.
  Effectively removing the association between user-marked attributes for deletion and user embeddings to prevent privacy leakage.
- **Objective 2:** Recommendation Knowledge Retention.
  Ensuring the recommendation performance of the RS is maintained post-unlearning.
- **Dual-Objective Optimization.**
  Balance the attribute-privacy and recommendation performance

Problem-Solving Approach

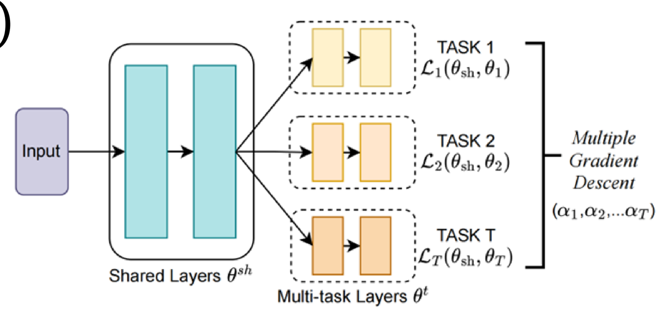
Objective 1: Minimizing the mutual information between the user embedding em_i' and the attributes to be forgotten $a_j \in AU$:

$$\min_{em_i'} \sum_{au_j \in A_u} I(em_i'; a_j)$$

Objective 2: Ensuring recommendation performance remains consistent post-unlearning

$$\min_{em_i'} Dist(M(em), M(em_i'))$$

Dual-Objective Optimization:
Changing hard parameter sharing for Multi-task Learning to Multiple Gradient Descent Algorithm



Compositional Attribute Unlearning

Private Attributes Information Loss.

AttrCloak approximate the mutual information $\mathcal{L}_{i,j}^{AU}$ of attribute au_j ($|U_{au_j}|$ classes, for class $c_k \in |U_{a_j}|$, the number of users is $|S_{a_j=c_k}|$) using a variational upper bound based on the KL divergence

$$\mathcal{L}_j^{AU} = I(em_i'; a_j) \le \sum_{k=1}^{|U_{a_j}|} \frac{|S_{a_j=c_k}|}{|S_{a_j}|} D_{KL}(q_\phi(em_i' | Em_{au_j=c_k}) \,|\, p(em_i'))$$

- Computing $em's$ Gaussian distribution for each：

$$\mu_{j,k} = \frac{1}{|S_{a_j=c_k}|} \sum_{em_i' \in S_{a_j=c_k}} em_i', \qquad \Sigma_{j,k} = \frac{1}{|S_{a_j=c_k}|} \sum_{em_i' \in S_{a_j=c_k}} (em_i' - \mu_{j,k})(em_i' - \mu_{j,k})^T$$

$$\mu_{global} = \frac{1}{|S_{a_j}|} \sum_{em_i' \in S_{a_j}} em_i', \qquad \Sigma_{global} = \frac{1}{|S_{a_j}|} \sum_{em_i' \in S_{a_j}} (em_i' - \mu_{global})(em_i' - \mu_{global})^T$$

- Calculating the KL divergence between each class embedding distribution and the global embedding distribution：

$$D_{KL}(N(\mu_{j,k}, \Sigma_{j,k}) || N(\mu_{global}, \Sigma_{global})) = \frac{1}{2}[\log \frac{\det(\Sigma_{global})}{\det(\Sigma_{j,k})} - d + Tr(\Sigma_{global}{}^{-1}\Sigma_{j,k}) + (\mu_{global} - \mu_{j,k})^T \Sigma_{global}{}^{-1}(\mu_{global} - \mu_{j,k})]$$

- Calculating the unlearning loss function $\mathcal{L}_j^{AU}$ for au_j as below:

$$\mathcal{L}_j^{AU} = \sum_{k=1}^{|U_{a_j}|} \frac{|S_{a_j=c_k}|}{|S_{a_j}|} D_{KL}(N(\mu_{j,k}, \Sigma_{j,k}) || N(\mu_{global}, \Sigma_{global})), \ \mathcal{L}^{AU} = \frac{1}{|A|} \sum_{j=1}^{|A|} \mathcal{L}_j^{AU}$$

Recommendation Knowledge Retention Loss.

An intuitive approach is to directly use the recommendation loss function $\mathcal{L}^{Rec}$ from the RS training phase. To accelerate the execution process, we only update user embeddings during unlearning:

$$\mathcal{L}^{Rec} = \mathcal{L}_{BPR,BPR,...}(s_\psi(f_{\varphi,p}(u), f_{\varphi,p}(i)), R)$$

We additionally propose the use of a regularization loss $\mathcal{L}^{Reg}$ to restrict the range of user embedding updates:

$$\mathcal{L}^{Reg} = \sum_{i=1}^{|U|} ||em_i - em_i'||^2 = \sum_{i=1}^{|U|} \sum_{j=1}^{d} ||em_{i,j} - em_{i,j}'||^2$$

Combining the loss above, we get a dual-objective optimization problem:

$$\min_{em_i'} \mathcal{L}^{All} = \min_{em_i'} \alpha_u \mathcal{L}^{AU} + \alpha_r \mathcal{L}^R, \text{ where } \mathcal{L}^R = \mathcal{L}^{Rec} \text{ or } \mathcal{L}^R = \mathcal{L}^{Reg}$$

Dual-objective Optimization with Parameter Self-sharing.

- Why Parameter Self-sharing: To accelerate the training, the layers following the user embedding, including item embedding, do not participate in multi-task optimization and are therefore not classified as "task-specific" layers.
- Pareto Stationary Point in Attribute Unlearning:

$$\alpha_u(\nabla_{em_i'}(\mathcal{L}^{AU}) + (1 - \alpha_u)(\nabla_{em_i'}(\mathcal{L}^R)) = 0, \alpha_u \in [0,1]$$

reformulate as the following optimization

$$\min_{\alpha_u} \left\| \alpha_u(\nabla_{em_i'}(\mathcal{L}^{AU}) + (1 - \alpha_u)(\nabla_{em_i'}(\mathcal{L}^{AU})) \right\|_2^2$$

solving the following optimization

$$\widehat{\alpha_u} = \left[\frac{(\nabla_{em_i'}(\mathcal{L}^R) - \nabla_{em_i'}(\mathcal{L}^{AU}))^T \nabla_{em_i'}(\mathcal{L}^R)}{\left\| \nabla_{em_i'}(\mathcal{L}^{AU}) - \nabla_{em_i'}(\mathcal{L}^R)) \right\|_2^2} \right]_{+,1}^T$$

where $[\cdot]_{+,1}^T$ denotes clipping to $[0,1]$ for $\mathcal{L}^R = \mathcal{L}^{Rec}$, for $\mathcal{L}^R = \mathcal{L}^{Reg}$, to avoid stagnation of gradient updates caused by regularization, it is clipped to $[0, 0.1]$, i.e., $[x]_{+,1}^T = \max(\min(x,1), 0/0.1)$.

- By adjusting α_u dynamically and continuously during the AU updates, a balance between the performance and privacy objectives is achieved.

Evaluation Results

Attribute-wise Privacy Performance against AIA.

Attacker	Type	Dataset / Sensitive Attributes	MovieLens-100K Gender	MovieLens-100K Age	MovieLens-100K Occupation	MovieLens-1M Gender	MovieLens-1M Age	MovieLens-1M Occupation	ModCloth Body Shape	Last.FM-1K Gender	Last.FM-1K Age	Last.FM-1K Location
XGBoost Attacker		Original	0.7626	0.4496	0.2281	0.7955	0.4164	0.1995	0.7648	0.7513	0.6040	0.5876
	DD	AttrCloak-DD	0.5822	0.1989	0.0995	0.5608	0.1692	0.1024	0.5566	0.5485	0.3531	0.3493
		RAP [6]	0.9310	0.8143	0.6034	0.9917	0.9611	0.8079	0.9903	0.9660	0.8789	0.9281
		BlurMe [26]	0.6300	0.3170	0.1525	0.6641	0.2957	0.1482	0.6606	0.5813	0.4023	0.4161
		LDP-SH [22]	0.6698	0.2692	0.1472	0.6854	0.3171	0.1217	0.6289	0.5977	0.4351	0.3783
	DF	AttrCloak-DF	0.5995	0.2401	0.1419	0.6203	0.2503	0.0772	0.5663	0.5612	0.1892	0.3279
		U2U-R [2]	0.9987	0.9947	0.9788	0.9999	0.9992	0.9999	0.9999	0.9937	0.9823	0.9987
		D2D-R [2]	0.6538	0.2798	0.1989	0.7113	0.3180	0.1551	0.6386	0.5927	0.3405	0.3960
MLP Attacker		Original	0.7414	0.3289	0.2454	0.7243	0.3526	0.1660	0.7653	0.6179	0.5359	0.5612
	DD	AttrCloak-DD	0.5928	0.1724	0.0358	0.5217	0.1656	0.0406	0.5216	0.5549	0.1501	0.1803
		RAP [6]	0.6286	0.2056	0.0995	0.5235	0.1829	0.0257	0.5610	0.5549	0.1197	0.2421
		BlurMe [26]	0.6605	0.2162	0.0902	0.6177	0.1573	0.0803	0.5637	0.5776	0.2245	0.2711
		LDP-SH [22]	0.6976	0.2586	0.0690	0.6475	0.1838	0.0828	0.6787	0.5422	0.3266	0.2699
	DF	AttrCloak-DF	0.6088	0.1950	0.0528	0.6169	0.1821	0.0555	0.5589	0.5132	0.0567	0.2106
		U2U-R [2]	0.6300	0.2003	0.1056	0.6036	0.2036	0.0927	0.5640	0.5536	0.3783	0.4288
		D2D-R [2]	0.6340	0.2162	0.0531	0.6537	0.2045	0.0348	0.5645	0.4918	0.2686	0.2863
Random Attacker			0.5000	0.1429	0.0476	0.5000	0.1429	0.0476	0.5000	0.5000	0.1469	0.1667

Experiments were conducted on four datasets with MLP/XGboost attackers.

Recommendation Performance.

Datasets	Type	Methods	NDCG@5	NDCG@10	HR@5	HR@10
MovieLens-100K		Original	0.8116	0.7721	0.7979	0.7479
	DD	AttrCloak-DD	0.8172	0.7769	0.8034	0.7524
		RAP	0.5819	0.5590	0.5784	0.5473
		BlurMe	0.8053	0.7685	0.7941	0.7464
		LDP-SH	0.6183	0.5838	0.6174	0.5695
	DF	AttrCloak-DF	0.7728	0.7531	0.7661	0.7409
		U2U-R	0.6952	0.6735	0.6895	0.6619
		D2D-R	0.6685	0.6439	0.6614	0.6294
MovieLens-1M		Original	0.6473	0.6095	0.6324	0.5853
	DD	AttrCloak-DD	0.6554	0.6176	0.6404	0.5934
		RAP	0.6518	0.6224	0.6285	0.5979
		BlurMe	0.6468	0.6089	0.6306	0.5840
		LDP-SH	0.4907	0.4691	0.4928	0.4609
	DF	AttrCloak-DF	0.6523	0.6217	0.6277	0.5958
		U2U-R	0.6750	0.6243	0.6467	0.5904
		D2D-R	0.6318	0.6017	0.5981	0.5668
ModCloth		Original	0.7297	0.7483	0.7678	0.8203
	DD	AttrCloak-DD	0.7928	0.8093	0.8292	0.8734
		RAP	0.3842	0.3860	0.3956	0.4148
		BlurMe	0.7168	0.7345	0.7512	0.8013
		LDP-SH	0.5647	0.6004	0.6262	0.7307
	DF	AttrCloak-DF	0.5470	0.5523	0.5589	0.5781
		U2U-R	0.5084	0.5053	0.5162	0.5321
		D2D-R	0.5082	0.5053	0.5164	0.5322
Last.FM-1K		Original	0.6696	0.6318	0.6556	0.6097
	DD	AttrCloak-DD	0.6655	0.6281	0.6498	0.6049
		RAP	0.5693	0.5297	0.5556	0.5062
		BlurMe	0.6452	0.6008	0.6106	0.5694
		LDP-SH	0.6113	0.5703	0.5943	0.5433
	DF	AttrCloak-DF	0.6060	0.5707	0.5888	0.5477
		U2U-R	0.5135	0.4788	0.4972	0.4564
		D2D-R	0.5657	0.5313	0.5498	0.5096

Efficiency.

Type	Time(s)	ML-100K	ML-1M	ModCloth	Last.FM-1K
DD	AttrCloak-DD	235.78	1322.18	97.05	9580.37
	RAP	580.35	4317.50	672.63	76398.79
	BlurMe	678.69	4266.94	506.78	196695.70
	LDP-SH	414.76	2972.77	474.38	51290.93
DF	AttrCloak-DF	12.81	39.20	167.52	10.03
	U2U-R	8.98	111.36	169.03	88.34
	D2D-R	7.45	56.98	88.54	6.38

Comprehensive evaluations show that AttrCloak effectively safeguards user privacy (with AIA accuracy closer to that of a random attacker) while maintaining high-quality recommendations (without great performance drop, even better), in an efficient post-training way (with low running time).

Contact Us　Paper