2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

User Privacy Leakage in Text-based Recommendation

Zhiyong Wu[0009-0002-1224-1744], Hongguang Chen[0009-0005-8843-195X], and Yin Chen(✉)

College of Artificial Intelligence, South China Normal University, Guangzhou 510631, China
ychen@scnu.edu.cn

Abstract. With the breakthrough development of large language models, the recommendation system is undergoing a transformation from the traditional recommendation model based on the unique identity (ID) of users/items (pure ID-based model, IDRec) to the recommendation model integrated with pre-trained modality encoder (modality-based recommendation model, MoRec). This paradigm overcomes the cold start problem and enables the recommendation model to achieve cross-platform migration through pretraining. However, the vector encoded by the modality encoder always contains more information. Given this, a natural question arises: would MoRec suffer more security issue which could cause serious leakage of user historical behavior data compared to IDRec? We aim to explore this question and study modality-based attack model in textual field. Specifically, we study several subquestions: (i) which recommendation paradigm, T-MoRec (Textual-MoRec) or IDRec, performs worse in protecting user privacy against attacks by the attack model? (ii) can the latest technical advances from NLP translate into attack improvement for T-MoRec? (iii) are there other factors affect textual recommendation attack model? What's the proper setting to conduct the attack? To answer all this questions, we purpose Text-based Recommendation Attack Model (TRAM) and conduct rigorous experiments with textual modality. We provide the first empirical on two public datasets, MIND and EB-NeRD, demonstrating that T-MoRec leads to serious leakage of user historical behavior data compared with IDRec under the same conditions. Additionally, we show that the leakage is consistently influenced by the hyperparameters and training cost in textual recommendation attack model.

Keywords: Information Security, Recommendation System, Data Mining.

1 Introduction

Recommendation systems nowadays provide personalized item recommendations based on user interactions, with deep learning and graph neural networks playing a crucial role. However, it is important to note that both these methods heavily rely on ID mapping information during training. Specially, the training data consists solely of mapped user/item indices, and their interactions are represented in the interaction matrix using binary values (1 for interaction and 0 for no interaction). While this data has proven effective, it overlooks other valuable data, such as rich textual information related to items. The absence of this additional information can lead to a reduction in the amount of information captured in the learned representations.

In recent years, there has been a growing interest in leveraging various data modalities to enhance recommendation systems. In particular, large language models (LLMs) such as GPT-4 [1] and LLaMA [2] have shown remarkable performance in natural language understanding tasks. Current studies [3] have already combined recommendation methods with the characteristics of language models. The shift from pure ID-based model (IDRec) to modality-based recommendation model (MoRec) has revealed that under state-of-the-art LLM encoders, MoRec can achieve or surpass the recommendation effectiveness of IDRec.

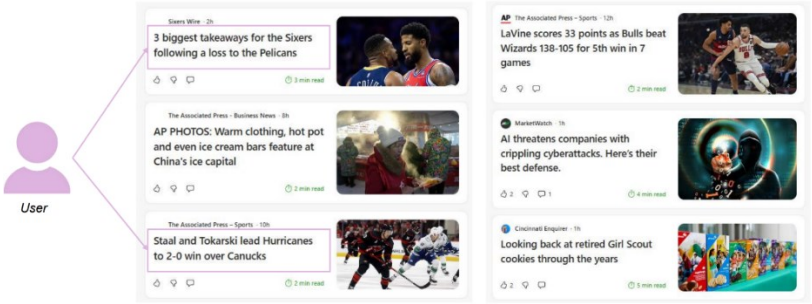


Fig. 1. Two illustrative examples of recommender exposure data are shown. The left illustrates that user interacts with two news items in the exposed item slate. And the right depicts a case where no interactions are observed between the exposed items and the user.

In this paper, we intend to explore a potential security risk of Textual Modality-based Recommendation (T-MoRec) and investigate a key question: Whether there are security risks in T-MoRec that affect its development? We conduct experiments using recommender system exposure data since this kind of data has been demonstrated to be capable of causing user behavior leakage [4]. Fig. 1 gives illustrative examples of the exposure data. To be concise, we attempt to address the following subquestions:

RQ1: Equipped with modality encoders (MEs), would T-MoRec perform worse in protecting user privacy against attacks by the attack model? To answer this question, we propose a new Textual modality-based Recommendation Attack Model (TRAM). We conduct empirical studies using both IDRec-based Attack Model (IAM) and TRAM, evaluated on two real-world recommendation datasets: MIND [5] and EB-NeRD [6].

Novelty clarification: Previous studies on T-MoRec [7-9] have not provided rigorous comparison with IDRec in terms of security. A fair comparison here means that T-MoRec and IDRec should be compared using the same backbone network and experimental settings, such as training setups and optimization techniques. Security comparison means that when facing the same attack model, would T-MoRec perform worse in protecting user privacy against attacks by the attack model compared to IDRec?

RQ2: Can the recent technical advances in the field of NLP leads to significant leakage of user's historical behavioral data in T-MoRec? We investigate this question through two experimental approaches. First, we access T-MoRec by comparing smaller vs larger ME (Modality Encoder), as prior research indicates that pre-trained

ME with larger model sizes typically produce richer modality features than their smaller counterparts in various downstream tasks. Second, we evaluate T-MoRec by comparing weaker vs stronger ME, where weaker and stronger is determined based on their performance in NLP tasks.

RQ3: Are there any factors that affect the TRAM? What's the proper setting to conduct the attack? A key goal of T-MoRec research is to explore the framework and factors influencing the attack model. We aim to identify and analyze the key elements that affect the effectiveness of the attack model. To ensure a fair comparison, we use control variables, including components such as the self-attention based encoder and decoder.

2 Preliminary

We use U to denote user set and I to denote item set. In the recommender system, user u generates a series of user behaviors (e.g., watching, rating, or liking) based on the items exposed by the system. We denote the set of user behavior sequences for user u as B^u, which is considered as user privacy in this work. More specifically, in the news recommendation, B^u could represent the user complete browsing history across multiple session, where B_i^u corresponds to a sequence of news articles clicked during the i-th session (e.g., B_1^u = ("Politics News", "Tech Review") for morning browsing, B_2^u = ("Sports Update") for afternoon browsing). The set of system recommendation item slates for user u is represented by E_i^u, which serves as the input data for the attack model. As shown in Fig. 2, the task of the attack model is to infer B_i^u from the observation of E_i^u, which can be formulated as Eq. 1

$$p(b_1, b_2, \ldots, b_M \mid e_1, e_2, \ldots, e_N). \tag{1}$$

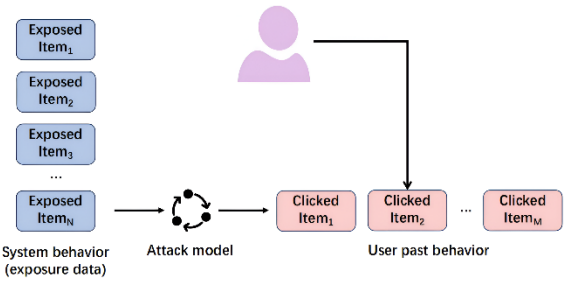


Fig. 2. The attack model aims to infer the privacy of user's past behavior from the system behavior data.

The input and output of our attack model come from the systems and users, respectively. As the input, the system exposure data E_i^u is provided solely by the system, meaning that the attack model is unaware of which items in E_i^u were clicked or interacted with by the user. Additionally, the recommender system usually exposes a slate

of items simultaneously, implying that the items in E_i^u could have no strong sequential orders. Table 1 summarizes the important notations used in this paper.

Table 1. The glossary table.

Notations	Description
U, I	The user and item set
B^u	The user behavior sequence set of user u, $B^u = \{B_1^u, B_2^u, \ldots, B_{\lvert B^u \rvert}^u\}$
B_i^u	A specific user behavior sequence of user u, $B_i^u = (b_1, b_2, \ldots, b_M)$ with $b_j \in I$
E^u	The set of exposed item slates for user u, $E^u = \{E_1^u, E_2^u, \ldots, E_{\lvert E^u \rvert}^u\}$
E_i^u	A specific exposed item slate for user u, $E_i^u = (e_1, e_2, \ldots, e_N)$ with $e_j \in I$
M	The length of user behavior sequence
N	The size of exposed item slate
d	The item embedding size

3 Methodology

3.1 Text-based Recommendation Attack Model

Our proposed Text-based Recommendation Attack Model (TRAM) focuses on inferring users' private iteration sequences from observed system exposure data. As illustrated in Fig. 3, the IDRec-based Attack Model (IAM) represents items using unique IDs and retrieves embeddings from the item ID embedding matrix, which are then fed into the attack model to infer user privacy. In contrast, TRAM transforms the input format of E_i^u from item IDs to textual representations. Specifically, it leverages various pre-trained text encoders followed by a dimension transformation (DT) layer to obtain the required embeddings for the attack model. Subsequently, the attack model processes these embeddings in the same manner to infer user privacy. To enhance the adaptability and effectiveness of TRAM, we employ pre-trained text encoders of varying sizes that represent current mainstream architectures.

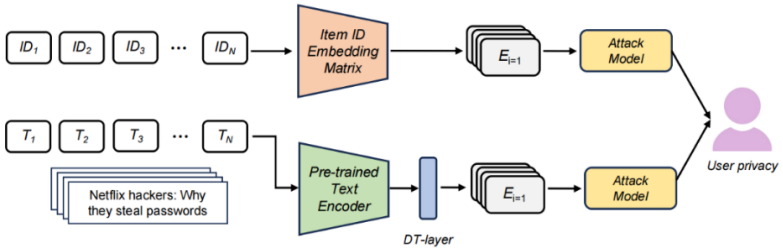


Fig. 3. An overview of TRAM and IAM. T_i denotes text features, and E_i represents item embedding. TRAM utilizes pre-trained text encoders, while IAM uses item ID embedding matrix.

The attack model in TRAM and IAM is designed to reconstruct users' historical behavior sequences by leveraging encoded item representations obtained from system exposure data. As shown in Fig. 4, it consists of two key components: a self-attention based encoder and an attention-based transformer decoder. The encoder processes the sequence of item embeddings to capture the latent contextual information within the exposed items, while the decoder utilizes this latent representation to predict the most likely sequence of user interactions. To ensure the model effectively captures both short-term and long-term dependencies in user behavior, we incorporate position embeddings and masked attention in the decoder, enabling it to generate more possible predictions of past user actions. The overall architecture facilitates a robust privacy inference process by progressively refining the extracted information through multi-head attention and feed-forward networks.

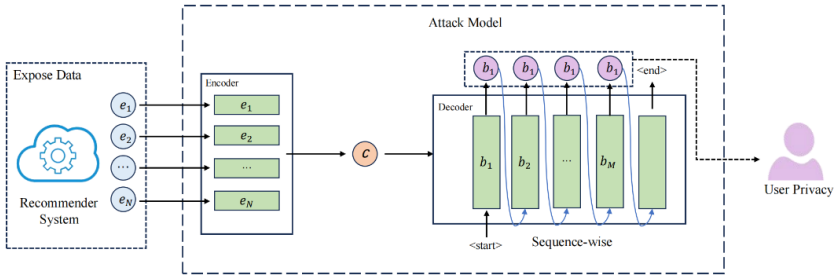


Fig. 4. The overall attack model structure. The encoder mapping the system exposure E_i^u to a representation c. Then the user privacy B_i^u could be inferred through sequence-wise decoding.

3.2 Encoder

The self-attention based encoder consists of multi-head attention and feed-forward network. The multi-head attention mechanism uses scaled dot-product attention in each head to capture the relevance of input items. The dot-product attention can be expressed as Eq. 2

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d}})V, \tag{2}$$

where Q, K, V represent the queries, keys and values, respectively. The scaling factor $\sqrt{d}$ is employed to normalize the computed correlations, preventing excessively large inner products. Then the multi-head attention on the input item embeddings is formulated as Eq. 3 and Eq. 4

$$MHA(\mathbf{E}) = \text{concat}\{head_1, head_2, \ldots, head_h\}\mathbf{W}^O \tag{3}$$

$$head_i = \text{Attention}(\mathbf{W}_i^Q\mathbf{E}, \mathbf{W}_i^K\mathbf{E}, \mathbf{W}_i^V\mathbf{E}), \tag{4}$$

where $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, ...\mathbf{e}_N] \in \mathbb{R}^{N\times d}$ is the stacked embedding matrix of the exposed items. $\mathbf{W}_i^Q\mathbf{E}, \mathbf{W}_i^K\mathbf{E}, \mathbf{W}_i^V\mathbf{E}$ and $\mathbf{W}^O$ are trainable parameters, where h denotes the number of attention heads.

To mitigate overfitting and enable a more stable learning without vanishing or exploding gradient issues, here we also introduce residual connection, dropout layers and layer normalization. The representation following the multi-head attention mechanism is formulated as Eq. 5

$$\widetilde{\boldsymbol{E}} = \boldsymbol{E} + dropout(MHA(Layernorm(\boldsymbol{E}))) \in \mathbb{R}^{N\times d}. \tag{5}$$

Subsequently, a two-layer of feed-forward network (FFN) is utilized to enhance the capacity of encoder. The latent representation after self-attention based encoding is formulated as Eq. 6

$$\boldsymbol{C}_{att} = \widetilde{\boldsymbol{E}} + dropout(FFN(LayerNorm(\widetilde{\boldsymbol{E}}))) \in \mathbb{R}^{N\times d}. \tag{6}$$

Additionally, a CLS token is inserted into the original input sequence $(\mathbf{e}_1, \mathbf{e}_2, ...\mathbf{e}_N)$. The corresponding vector of the CLS token in $\boldsymbol{C}_{att}$ can then be regarded as the final encoded vector representation of the system exposure.

3.3 Decoder

Based on the encoded latent representation of system behavior, we propose attention-based transformer decoder. The mathematical details are not reiterated here, as the transformer decoder shares a similar calculation process to the attention-based encoder. However, it is important to highlight the follow key differences.

- Unlike the encoder, position embeddings are incorporated in the decoder to differentiate the order of the user's past behaviors and to introduce sequential signals.
- Given the nature of the attack on user past behavior sequences, the model should only consider the last j items when inferring the (j-1)-th item. To address this, a casual mask is introduced in the decoder to modify the attention mechanism, thereby implementing masked multi-head attention.

In attention-based transformer decoding, we define cross-entropy loss for each output position to perform the classification. The final loss function for decoding can thus be formulated as Eq. 7

$$\mathrm{L}(\mathbf{Q},\mathbf{Y}) = -\sum_{m=1}^{M}\sum_{j=1}^{|\mathrm{I}|} y_{mj} \log q_{mj}, \tag{7}$$

where y_{mj} and q_{mj} are ground-truth probability and the (m, j)-th entry of the computed probability matrix $\mathbf{Q}$ (i.e., the stack of $\mathbf{q}_t$). It is important to note that during the inference stage, we cannot obtain the ground-truth user behaviors like the training process. Therefore, the inference results from previous steps are used as the input for subsequent inference steps.

4 Experiment

4.1 Dataset Description

We evaluate these experiments in two real-world news datasets: MIND [5] and EB-NeRD [6]. Both of the two datasets contain user behavior data (e.g., clicks) and system behavior data (e.g., exposed impressions). Besides, news articles are published continuously and are prone to expiration, which introduces a significant cold-start problem [10], contributing to the notable comparison results. Table 2 summarizes the statistics of these two datasets. To make a fair comparison between IDRec and T-MoRec, the dataset should ensure that user's decisions to click on an item are determined purely by the item's textual modality content features. Both in MIND and EB-NeRD we represent items by their news article titles.

Table 2. Dataset statistics. The impressions can be seen as system behavior data while the clicks can be regarded as the user behavior privacy.

Dataset	EB-NeRD	MIND
#users	30,485	96,584
#items	20,746	101,527
#impressions	3,795,120	74,791,274
#clicks	477,534	8,716,100
#Avg. title len.	6.6	11.52

4.2 Evaluation Protocols

We use cross-validation to evaluate the performance of TRAM, setting the training, validation, and test data split ratio to 8:1:1. We use user-based data splitting strategy because it avoids the potential information shortcuts between dataset. For validation and test part, the evaluation is to provide the attack model with item exposed by the system. Then checking the rank of the M ground-truth items in the results. We use Recall to evaluate the attack performance. Let $\tilde{B}_i^u @k$ denote the top-k inference output

of the attack model[1]. Recall@k measures how many ground-truth user behaviors are included in $\tilde{B}_i^u @k$, which is formulated as Eq. 8

$$Recall@k = \frac{|\ \tilde{B}_i^u @k \bigcap B_i^u|}{M} \tag{8}$$

We then report the average Recall@k across the entire test user set as the final results. In addition, we also report Normalized Discounted Cumulative Gain (NDCG), which are weighted versions of Recall assigning higher weights to the top-ranked positions of the inference result lists.

4.3 Hyperparameter Settings

We conduct our experiments with a batch of system exposure data and user behavior data (i.e., $[\ B_i^u \mid E_i^u\ ]$). For fair comparison, TRAM and IAM use the same hyperparameter settings. The sizes of B_i^u and E_i^u are set as $M = 5$ and $N = 10$ respectively without special mention. The item embedding size d from $\{128,512,768\}$ according to the pretrained model. We train all models with the Adam optimizer [11]. The dropout rate is tuned to 0.1. The learning rate γ is set as 0.001. For attention-based encoder and transformer-based decoder, the hyperparameters of the multi-head self-attention are set as 2 heads with a total of 128 hidden neurons without further specification. And for the MLP (multi-layer perception) we set l to 2. We utilize the weight sharing technique [12, 13] to tie the weights of the encoder item embedding and SoftMax layer item embedding in the decoder. For label smoothing, the ε is set to the $1/|\ I\ |$. Each experiment is conducted 3 times and the average result is reported.

4.4 IDRec vs. T-MoRec Attack Model (RQ1)

According to existing literature, IDRec has been identified as a significant threat to user privacy in recommender systems [4]. As MoRec gains increasing attention in the research community, our focus shifts to comparing these two paradigms to determine which one poses a greater risk of privacy leakage. To the best of our knowledge, such a direct comparison has not been explicitly addressed in prior studies.

As shown in Table 3, we observe that in the MIND dataset, TRAM achieves the highest score when utilizing the BERT_{base} encoder. Similarly, in the EB-NeRD dataset, TRAM outperforms IAM when employing the $\text{DistilBERT}_{multi}$ encoder. This suggests that, compared to IDRec, TRAM leads to more severe leakage of users' historical behavior across both datasets. We attribute this to the fact that text-based encoders inherently capture richer contextual information, making them more susceptible to privacy

[1] $\tilde{B}_i^u @k$ contains $k \times M$ items. It composed of the top-k items of all M inference positions.

leakage. This finding raises a critical security concern for T-MoRec research, highlighting the potential risks associated with using stronger text encoders in recommendation models.

Furthermore, we observe that TRAM's performance aligns closely with IAM across various datasets, including MIND and EB-NeRD, suggesting that the primary factor influencing privacy leakage is the dataset itself. Compared to EB-NeRD, MIND demonstrates a higher risk of historical behavior leakage. This discrepancy is likely due to the difference in average title length—11.52 words in MIND vs 6.6 words in EB-NeRD. The longer and richer vocabulary in MIND makes it easier for the attack model to capture semantic patterns, thereby increasing the likelihood of privacy leakage.

Table 3. Attack performance comparison between TRAM and IAM. TRAM results are obtained using different pre-trained MEs. Boldface indicates the highest score within the same dataset. "Rec" denotes Recall. Since the EB-NeRD dataset includes Danish, its encoder is a multilingual BERT model.

Dataset	Metrics	IAM	BERT_{tiny}	BERT_{base}	RoBERTa_{base}	Improv.
MIND	Rec@10	**0.7387**	0.7285	**0.7705**	0.7579	+4.29%
	NDCG@10	**0.3904**	0.3769	**0.4207**	0.4026	+7.76%
		IAM	$\text{DistilBERT}_{multi}$	BERT_{multi}	XML-R_{large}	Improv.
EB-NeRD	Rec@10	**0.7106**	**0.7387**	0.7298	0.7015	+3.95%
	NDCG@10	**0.3678**	**0.4035**	0.4002	0.3731	+9.71%

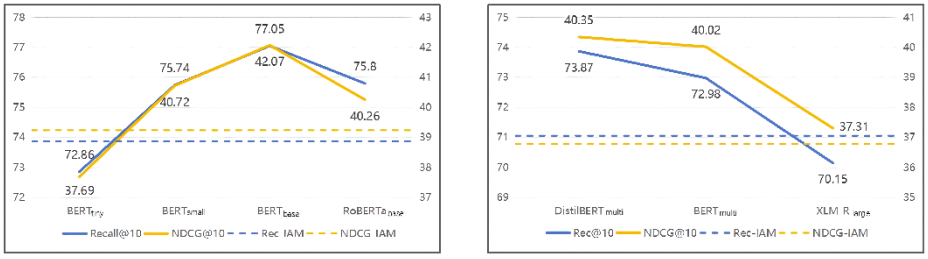
4.5 Inherit Advance in Modality Encoder (RQ2)

Intuitively, TRAM's performance also depends on the choice of textual modality encoder and the size of the encoder itself. This raises a critical question: Can recent advancements in NLP lead to severe leakage of users' historical behavioral data in T-MoRec? We aim to address this from the following perspectives.

First, we evaluate whether a larger pre-trained ME increases the risk of user behavioral data leakage in T-MoRec. As shown in Fig. 5, our evaluation on the MIND dataset reveals a clear trend: larger NLP models tend to heighten security risks, i.e., BERT_{tiny}-based TRAM < BERT_{small}-based TRAM < BERT_{base}-based TRAM. However, when evaluating TRAM on the EB-NeRD dataset, we observe a different trend: BERT_{multi}-based TRAM < $\text{DistilBERT}_{multi}$-based TRAM. This is likely because $\text{DistilBERT}_{multi}$ and BERT_{multi} share the same hidden dimensions, but $\text{DistilBERT}_{multi}$ undergoes a distillation process that removes redundant noise and focuses on the most relevant features. This enhanced representational efficiency allows $\text{DistilBERT}_{multi}$-based TRAM to achieve better attack performance than its BERT_{multi} counterpart. Thus, while larger MEs generally pose a greater security risk, this trend does not hold in all cases.

Besides, we investigate whether a more powerful encoder network leads to improved attack performance. For instance, RoBERTa is widely recognized as outperforming BERT. However, as shown in Fig. 5, TRAM with RoBERTa_{base} does not surpass TRAM with BERT_{base} on the MIND dataset, despite RoBERTa being considered a stronger model. A similar pattern is observed in EB-NeRD, where the weaker $\text{DistilBERT}_{multi}$,

paradoxically, achieves the highest Recall@10 and NDCG@10 score. These findings suggest that a more powerful NLP model does not necessarily increase security risks.



(a) MIND (b) EB-NeRD

Fig. 5. Evaluation of attack performance with different MEs in TRAM on the MIND and EB-NeRD dataset. Rec-IAM and NDCG-IAM denote the Rec@10 and NDCG@10 of IAM.

4.6 TRAM Study (RQ3)

In this subsection, we conduct experiments to explore the factors that influence the performance of TRAM. Specifically, we examine the impact of the number of attention heads in the multi-head self-attention mechanism and analyze the training cost associated with different model architectures.

Multi-head self-attention. As shown in Fig. 6(a), we evaluate the attack performance of TRAM using BERT$_{tiny}$ and BERT$_{base}$, where the number of attention heads is set to {1, 2, 4, 8}. We observe that the attack performance steadily decreases as the number of heads increases in BERT$_{tiny}$. Specifically, when the number of heads is set to 8, the attack performance decreases by 2.37% compared to when the number of heads is set to 1. However, when using BERT$_{base}$, we observe a different phenomenon that attack performance peaks when the number of heads is set to 2, instead of following a consistent decline.

Besides, we also show the results on EB-NeRD under the same experimental settings. As shown in Fig. 6(b), the attack performance of TRAM with DistilBERT$_{multi}$ and BERT$_{multi}$ follows a similar trend to that observed with BERT$_{tiny}$ on the MIND dataset. Specifically, when DistilBERT$_{multi}$ is used as the modality encoder, attack performance decreases steadily as the number of heads increases. Conversely, for BERT$_{multi}$, the attack performance peaks when the number of heads is set to 1, mirroring the behavior observed with BERT$_{tiny}$ in the earlier experiment.

The observed performance trends can be attributed to the varying representational capacities of the modality encoders. On the MIND dataset, BERT$_{base}$ exhibits significantly larger hidden dimensions and a more complex architecture, enabling its multi-head attention mechanism to capture a wide range of semantic subspaces. In contrast, BERT$_{tiny}$, with smaller hidden dimensions and fewer layers, is more constrained in its expressive power. As the number of heads increases, the dimensions of each head in smaller models decreases, limiting their ability to capture complex features. Conversely, encoders like BERT$_{base}$, with sufficient representational capacity, are better

equipped to leverage multi-head attention even as the number of heads increases. As a result, when the number of heads in $BERT_{base}$ -based TRAM increases to N = 2, each head maintains a sufficient dimension to capture more complex features.

On the EB-NeRD dataset, $DistilBERT_{multi}$ and $BERT_{multi}$ have the same hidden dimensions and a similar number of parameters, so both these models have similar representational capacity. Explained why both of them have the same result, achieving the highest performance when the number of heads is set to 1 in TRAM.

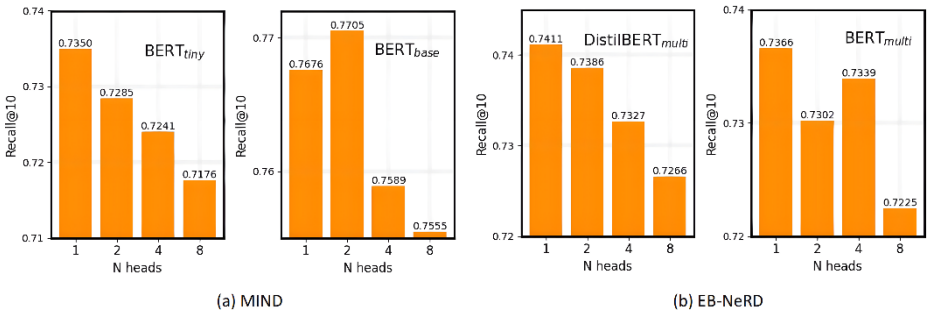


Fig. 6. Attack performance with different N heads on MIND and EB-NeRD.

Training cost. As shown in Fig. 5, when facing attack T-MoRec with larger ME tend to pose greater security risk. However, the training time also increase since the architecture of attack model is self-attention based with a long interaction sequence. We show the training cost on MIND and EB-NeRD in Table 4. It's easy to imagine that RoBERTabase and XML-Rlarge have the highest training time because both have the most parameters. This is consistent with existing similar research findings [4]. Therefore, while utilizing text modality information can improve attack effectiveness, it often comes with a higher training cost.

Table 4. The training cost. Param: number of tunable parameters, Time/E: averaged training time for one epoch, 'm' means minutes. We use A10 GPU with 24G memory.

Dataset	Method	Param.	Time/E	Dataset	Method	Param.	Time/E
MIND	IAM	40M	3.8m	EB-NeRD	IAM	5M	0.14m
	$BERT_{tiny}$	4M	7.3m		$DistilBERT_{multi}$	135M	0.35m
	$BERT_{base}$	109M	16.5m		$BERT_{multi}$	178M	0.33m
	$RoBERTa_{base}$	125M	17.8m		$XML\text{-}R_{large}$	559M	0.61m

5 Conclusions and Future Work

In this study, we propose a new recommendation attack model, TRAM, which explores the risk of textual modality-based recommender system in terms of user privacy leakage. The results show that T-MoRec would suffer more serious security risk, leading to

leakage of users' historical behavior data, compared to the traditional IDRec. In addition, we found that the risk of privacy leakage generally increases with ME parameters size and is consistently influenced by the hyperparameters and training cost in the attack model. Therefore, how to further reduce the risk of privacy leakage while improving the recommendation effect is still an important direction for future research. Future work can further optimize the architecture of modal encoders and explore more privacy-protecting methods, especially in large-scale datasets and diverse scenarios. In addition, further research is needed to investigate the portability of T-MoRec across different platforms and its impact on privacy protection.

Acknowledgments. The authors would like to thank all the anonymous reviewers for their valuable comments to improve this paper.

References

1. Achiam, J., Adler, S., Agarwal, S. et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774(2023).
2. Touvron, H., Lavril, T., Izacard, G., Martinet, X., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971(2023)
3. Yuan, Z., Yuan, F., Song, Y., Li, Y., et al.: Where to go next for recommender systems? id- vs. modality-based recommender models revisited. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2639–2649(2023)
4. Xin, X., Yang, J., Wang, H., Ma, J., et al.: On the user behavior leakage from recommender system exposure. In: ACM Transactions on Information Systems 41(3),1-15(2023)
5. Wu, F., Qiao, Y., Chen, J.H., Wu, C., et al.: Mind: A large-scale dataset for news recommendation. In: Proceedings of the 58th annual meeting of the association for computational linguistics. 3597–3606(2020)
6. Kruse, J., Lindskow, K., Kalloori, S., Polognano, M., et al.: EB-NeRD a large-scale dataset for news recommendation. In: Proceedings of the Recommender Systems Challenge. 1–11(2024)
7. Li, J., Zhu, J., Bi, Q., Cai, G., et al.: MINER: Multi-interest matching network for news recommendation. In: Findings of the Association for Computational Linguistics. 343–352(2022)
8. Wu, C., Wu, F., Ge, S., Qi, T., et al.: Neural news recommendation with multi-head self-attention. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP). 6389–6394(2019)
9. Wu, C., Wu, F., Qi, T., Huang, Y.: Empowering news recommendation with pre-trained language models. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval. 1652–1656(2021)
10. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: Proceedings of the 16th international conference on World Wide Web. 271–280(2007)
11. Diederik, P.K.: Adam: A method for stochastic optimization. CoRR,abs/1412.6980(2014)
12. Inan, H., Khosravi, K., Socher, R.: Tying word vectors and word classifiers: A loss framework for language modeling. arXiv preprint arXiv:1611.01462 (2016)

13. Press, O., Wolf, L., Using the output embedding to improve language models. arXiv preprint arXiv:1608.05859 (2016)