Black-Box Attacks on Sequential Recommenders via Data-Free Model Extraction

RecSys '21

Zhenrui Yue Huimin Zeng Zhankui He Julian McAuley

Alumnos: Benjamín Varela, Francisco Meza, Vicente Navarro

17 de junio de 2025

Contexto: Seguridad en recomendadores secuenciales

- Los **Sistemas de Recomendación Secuenciales (SRS)** predicen el *próximo ítem* a partir del historial ordenado de cada usuario (Netflix, Amazon, Steam).
- Se entrenan con modelos de aprendizaje profundo (RNN, SASRec, $\mathrm{BERT4Rec}$) sobre datos privados y costosos $\Longrightarrow$ alto valor comercial.
- Muchas plataformas exponen el SRS como servicio **black-box** vía API $\Longrightarrow$ sólo accesible mediante consultas.
- Preocupación emergente: Un atacante consultando la API, puede reconstruir un clon del modelo *sin* conocer datos ni código fuente.
- Esto abre la puerta a ataques como: manipulación de rankings, promoción de productos maliciosos o robo de propiedad intelectual.

Problema de recomendación:

¿Qué se recomienda?

Dado un historial ordenado de interacciones de un usuario $\mathbf{s}_t = (i_1, i_2, \ldots, i_t)$, el sistema atacado f_b devuelve el **próximo ítem** más probable.

$$\mathcal{I}_k = f_b(\mathbf{s}_t) \;=\; \left\{ i_{t+1}^{(1)},\, i_{t+1}^{(2)},\, \ldots,\, i_{t+1}^{(k)} \right\}$$

Ejemplo: $\mathbf{s}_t$ = [Toy Story, Nemo] $\rightarrow \mathcal{I}_3$ = [Toy Story 2, Frozen, Shrek]

Estado del arte: evolución de ataques por extracción

Línea de tiempo: extracción de modelos en IA

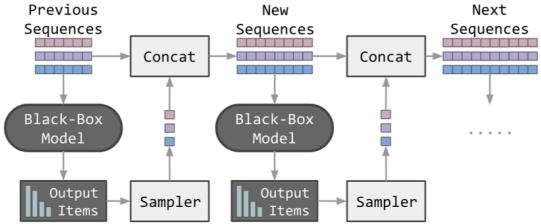
Año	Dominio	Hito clave
2018	Visión	*Knockoff Nets* (Orekondy): extracción de CNNs mediante consultas aleatorias.
2019	NLP	*BERT-Stealing* (Krishna): distilación de Transformers con pseudo-datos.
2021	RS secuenciales	**Este paper**: extracción *data-free* con generación autoregresiva y ataques transferibles.

Contribuciones del estudio

- **Primer análisis sistemático** de extracción de modelos en *recomendadores secuenciales* (NARM, SASRec y BERT4Rec) en tres datasets reales.
- **Estrategia data-free inédita:** Generación autoregresiva de secuencias + destilación hacia un modelo clonado, eliminando la necesidad de datos de usuarios.
- **Dos ataques transferidos**: Profile Pollution y Data Poisoning.

Solución – Paso 1: Generar datos sintéticos

1. **Generar secuencias**
   - **Objetivo**: imitar sesiones reales *sin* datos privados.
   - Elegir un *start item* al azar.
   - **Query**: enviar la secuencia parcial $\widehat{\mathbf{s}}_{1:t}$ al modelo f_b. $\rightarrow$ recibir top-k: $\hat{\mathbf{r}}_b^{(k)}$.
   - **Sample**: tomar 1 ítem del top-k (mayor score, mayor prob.).
   - **Append**: añadirlo y repetir hasta longitud T o agotar presupuesto B.



(a) Autoregressive Data Generation

Solución: extracción del modelo black-box

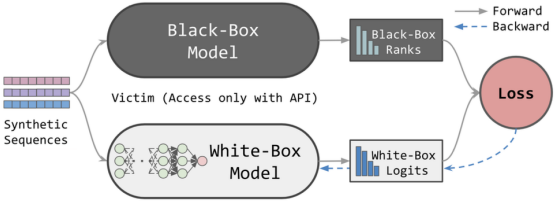
Destilación

- *Input*: pares $\left(\hat{\mathbf{s}}_{1:t}, \hat{\mathbf{r}}_b^{(k)}\right)$.
- *Objetivo*:

$$f_w^* \;=\; \arg\min_{f_w} \sum_{i=1}^{|\mathcal{X}|} \mathcal{L}_{\text{dis}}\left(\hat{\mathbf{r}}_b^{(k)(i)},\; f_w\left(x^{(i)}\right)\right).$$

- *Pérdida*:

$$\mathcal{L}_{\text{dis}} = \frac{1}{k-1}\sum_{i=1}^{k-1} \max\left(0, \hat{s}_w^k[i{+}1] - \hat{s}_w^k[i] + \lambda_1\right) + \frac{1}{k}\sum_{i=1}^{k} \max\left(0, \hat{s}_{\text{neg}}^k[i] - \hat{s}_w^k[i] + \lambda_2\right)$$



(b) Model Extraction via Distillation

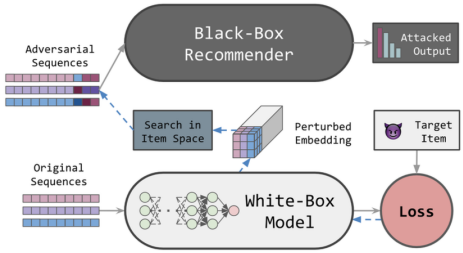
Ataques downstream

★ Profile Pollution

- El atacante usa el clon $\widehat{f}$ para generar una *secuencia* que incluya el ítem objetivo ℓ.
- Se calculan gradientes / similitud de embeddings para construir secuencias más efectivas.
- Esa secuencia se inyecta en el perfil del usuario, elevando la probabilidad de recomendar ℓ.
- Intuición: "contaminar" el historial para que f_b crea que el usuario desea ℓ.
- **Objetivo formal**:

$$z^* = \arg\max_z E_t(f_b([\mathbf{x}; z]))$$

Maximizar la exposición esperada del ítem objetivo ℓ.



(a) Profile pollution attack with white-box model

Ataques downstream

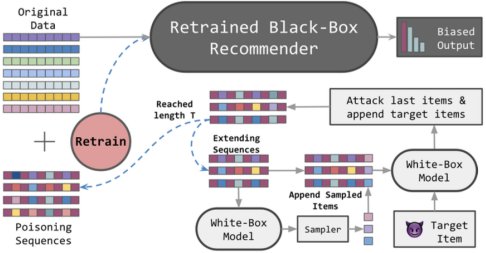
★ Data Poisoning

- Crea *usuarios ficticios* con historiales sintéticos obtenidos mediante $\widehat{f}$.
- Inyecta estos historiales en el dataset de entrenamiento degrada NDCG y Recall globales.
- Intuición: sembrar ruido estructurado que empuje al modelo hacia malas representaciones.
- **Objetivo formal**:

$$Z^* = \arg\max_{Z} \sum_{i=1}^{|\mathcal{X}|} E_t\big(f_b'(x^{(i)})\big)$$

$$\text{s. a. } f_b' = \arg\min_{f_b} \mathcal{L}_{\text{rec}}\big(f_b,\ Z \cup \mathcal{X}\big)$$

Maximizar la exposición del objetivo tras el *re-training*.



(b) Data poisoning attack via adversarial co-visitation

Datasets y Modelos

Datasets	Users	Items	Avg. len	Max. len	Sparsity
ML-1M	6,040	3,416	166	2277	95.16%
Steam	334,542	13,046	13	2045	99.90%
Beauty	40,226	54,542	9	293	99.98%

Table 1. Data Statistics

Model	Basic Block	Training Schema
NARM [25]	GRU	Autoregressive
SASRec [17]	TRM	Autoregressive
BERT4Rec [37]	TRM	Auto-encoding

Table 2. Sequential Model Architecture

Preguntas de investigación

1. **RQ1 – Viabilidad de la extracción**
   ¿Es posible obtener los pesos del modelo black-box sin los datos reales?
2. **RQ2 – Factores que influyen en la extracción del modelo black-box**
   ¿Como afecta la data sparsity y el presupuesto de consultas en la extracción del modelo?
3. **RQ3 – Viabilidad del ataque Profile Pollution**
   ¿Podemos realizar ataques de Profile Pollution mediante el modelo extraído?
4. **RQ4 – Viabilidad del ataque Data Poisoning**
   ¿Podemos realizar ataques de Data Poisoning mediante el modelo extraído?

Métricas utilizadas en la evaluación

★ Ranking Performance

- **Recall@K (Hit Rate HR@K)**
- **NDCG@K (Normalized Discounted Cumulative Gain)**

★ Agreement Measure – Agr@K

$$\mathrm{Agr@K} \;=\; \frac{|\,B_{\mathrm{top}K} \,\cap\, W_{\mathrm{top}K}|}{K}, \qquad 0 \le \mathrm{Agr@K} \le 1$$

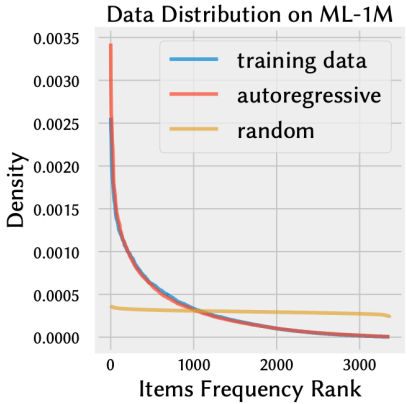
- $B_{\mathrm{top}K}$: lista top-K del modelo **black-box** (víctima).
- $W_{\mathrm{top}K}$: lista top-K del modelo **white-box** extraído.

Resultados RQ1 - Viabilidad de la extracción

		Black-Box		White-Box-Random				White-Box-Autoregressive			
Dataset	Option	N@10	R@10	N@10	R@10	Agr@1	Agr@10	N@10	R@10	Agr@1	Agr@10
ML-1M	NARM	**0.625**	**0.820**	0.598	0.809	0.389	0.605	0.615	0.812	**0.571**	**0.747**
	SASRec	**0.625**	**0.817**	0.578	0.796	0.270	0.516	0.602	0.802	**0.454**	**0.662**
	BERT4Rec	**0.602**	**0.806**	0.565	0.794	0.241	0.488	0.571	0.791	**0.339**	**0.593**
Steam	NARM	**0.628**	0.848	0.625	**0.849**	0.679	0.642	0.601	0.806	**0.743**	**0.722**
	SASRec	**0.627**	**0.850**	0.579	0.802	0.434	0.556	0.593	0.805	**0.668**	**0.702**
	BERT4Rec	**0.622**	**0.846**	0.609	0.838	0.199	0.490	0.585	0.793	**0.708**	**0.667**
Beauty	NARM	**0.356**	**0.518**	0.319	0.477	**0.356**	**0.511**	0.272	0.380	0.344	0.425
	SASRec	0.344	0.494	0.304	0.459	0.251	0.213	**0.347**	**0.505**	**0.343**	**0.357**
	BERT4Rec	**0.349**	**0.515**	0.200	0.352	0.026	0.043	0.300	0.454	**0.178**	**0.291**

Table 4. Extraction performance under identical model architecture and 5k budget, with Black-box original performance.

Resultados RQ1 – Viabilidad de la extracción



(a) Data Distribution for original and generated data.

Resultados RQ2 – Factores que influyen en la extracción del modelo

Model	k-core	Black-Box		Ours		
		N@10	R@10	N@10	R@10	Agr@10
NARM	5	0.360	0.536	0.331	0.488	0.559
	6	0.372	0.562	0.352	0.539	0.614
	7	0.386	0.630	0.369	0.597	**0.726**
	8	0.347	0.597	0.362	0.643	0.690
SASRec	5	0.351	0.514	0.332	0.479	0.651
	6	0.380	0.558	0.373	0.547	0.744
	7	0.424	0.640	0.427	0.648	0.782
	8	0.415	0.675	0.410	0.672	**0.791**
BERT4Rec	5	0.346	0.509	0.351	0.520	0.561
	6	0.366	0.547	0.374	0.555	0.652
	7	0.402	0.643	0.399	0.650	0.682
	8	0.403	0.694	0.383	0.659	**0.717**

Table 5. Influence of data sparsity. Model extraction on k-core Beauty

Resultados RQ2 – Factores que influyen en la extracción del modelo

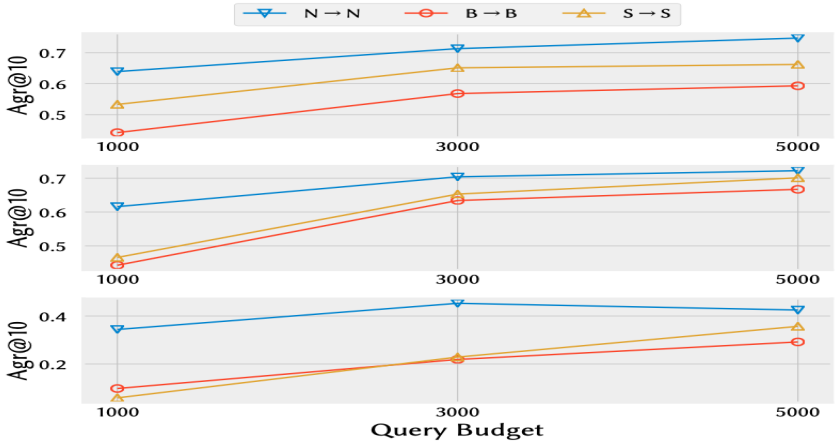


Table 6. Influence of query budgets on ML-1M (top), Steam (middle) and Beauty (bottom).

Resultados RQ3 – Viabilidad del ataque Profile Pollution

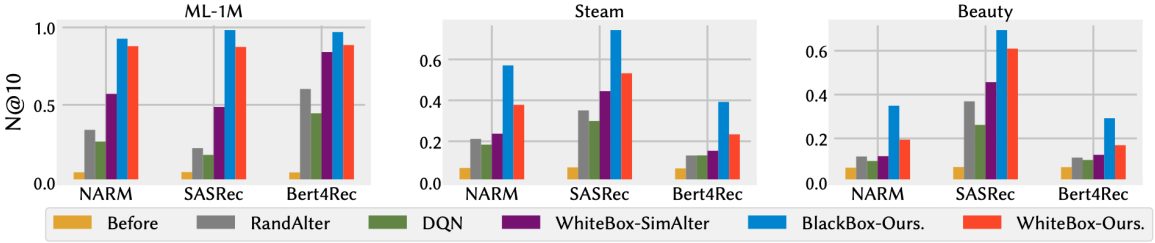


Fig. 5. Profile pollution attacks performance comparisons with different methods.

Resultados RQ3 – Viabilidad del ataque Profile Pollution

		ML-1M			Steam			Beauty		
Popularity	Attack	NARM	SASRec	BERT4Rec	NARM	SASRec	BERT4Rec	NARM	SASRec	BERT4Rec
head	before	0.202	0.217	0.201	0.313	0.327	0.311	0.261	0.246	0.260
	ours	**0.987**	**0.981**	**0.968**	**0.850**	**0.745**	**0.714**	**0.650**	**0.825**	**0.521**
middle	before	0.037	0.034	0.036	0.012	0.012	0.009	0.023	0.030	0.027
	ours	**0.902**	**0.876**	**0.901**	**0.341**	**0.585**	**0.152**	**0.106**	**0.556**	**0.105**
tail	before	0.005	0.008	0.009	0.000	0.000	**0.000**	**0.002**	0.009	0.002
	ours	**0.701**	**0.760**	**0.760**	**0.017**	**0.160**	**0.000**	0.001	**0.557**	**0.010**

Table 7. Profile pollution attacks to different sequential models for items with different popularity, reported in N@10.

Resultados RQ4 – Viabilidad del ataque Data Poisoning

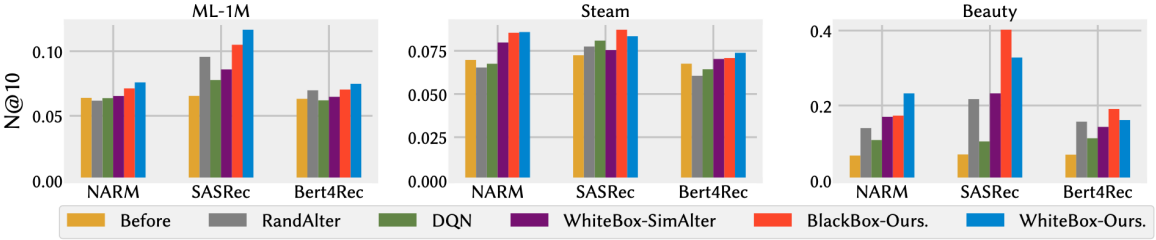


Fig. 6. Data poisoning attacks performance comparisons with different methods

Resultados RQ4 – Viabilidad del ataque Data Poisoning

		ML-1M			Steam			Beauty		
Popularity	Attack	NARM	SASRec	BERT4Rec	NARM	SASRec	BERT4Rec	NARM	SASRec	BERT4Rec
head	before	0.202	0.217	0.201	0.313	0.327	0.311	0.261	0.246	0.260
	ours	**0.205**	**0.351**	**0.213**	**0.347**	**0.352**	**0.324**	**0.425**	**0.477**	**0.364**
medium	before	0.037	0.034	0.036	0.012	0.012	0.009	0.023	0.030	0.027
	ours	**0.053**	**0.067**	**0.048**	**0.026**	**0.021**	**0.014**	**0.217**	**0.309**	**0.135**
tail	before	0.005	0.008	0.009	0.000	0.000	0.000	0.002	0.009	0.002
	ours	**0.016**	**0.032**	**0.017**	**0.004**	**0.004**	**0.004**	**0.086**	**0.235**	**0.036**

Table 8. Data poisoning attacks to different sequential models for items with different popularity, reported as N@10.

Conclusiones principales

Hallazgos esenciales

1. **La extracción data-free es factible.** Con un número moderado de consultas se obtiene un clon muy similar al modelo original.
2. **El clon habilita ataques efectivos.** Inyectar perfiles o datos sintéticos sesga de forma apreciable la calidad de las recomendaciones.
3. **La vulnerabilidad varía según arquitectura y densidad.** Modelos más simples y datasets más densos se clonan con mayor fidelidad.

Referencias

- Yue et al. *Black-Box Attacks on Sequential Recommenders via Data-Free Model Extraction*. RecSys '21.
- Orekondy et al. *Knockoff Nets*. CVPR '19.
- Krishna et al. *Thieves on Sesame Street! Model Extraction of BERT-based APIs*. ICLR '20.
- Zhang et al. *LOKI: Practical Data Poisoning Attack against Next-Item Recommendation*. WWW '20.
- Tramèr et al. *Stealing ML Models via APIs*. USENIX '16.

github.com/Yuceeeeeee/RecSys-Extraction-Attack

Fin

¡Gracias por su atención!

¿Preguntas?