User Privacy in Collaborative Filtering Systems

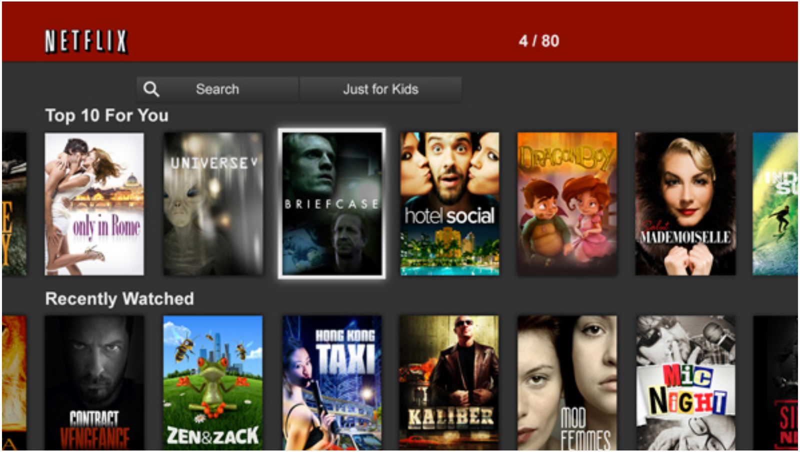
Antoine Rault

Supervision by *Anne-Marie Kermarrec* and *Davide Frey*

Recommendation System (RS)

Automatically recommend items by learning users' interest

Successful Recommendation Systems

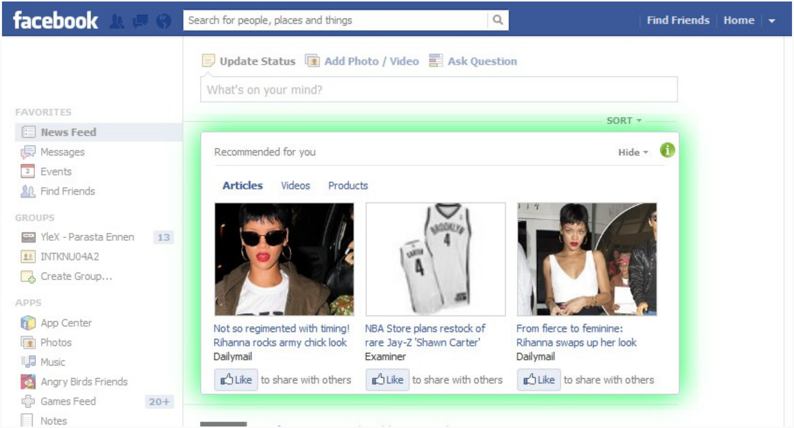


Netflix: 75% of views driven by recommendation

Successful Recommendation Systems

Amazon: +29% sales from recommendation

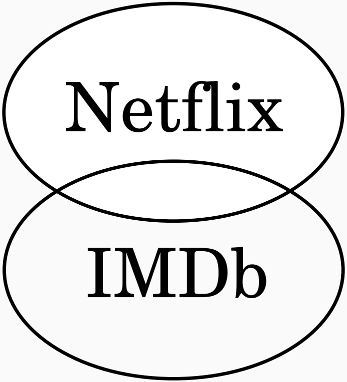
Successful Recommendation Systems



Facebook: News Feed is like a RS

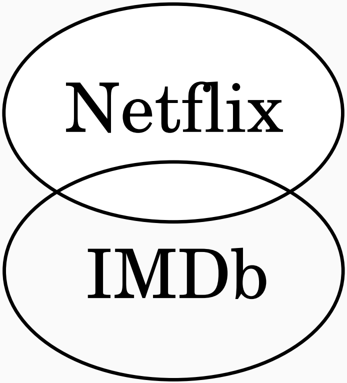
Recommendation System = Privacy Threat

Example: Netflix Prize De-anonymization



Recommendation System = Privacy Threat

Example: Netflix Prize De-anonymization



Lawsuit

Different Kinds of Privacy Threats

Source of threat: data collection

Threat: “Big Brother”

Solution: Decentralization

Different Kinds of Privacy Threats

Source of threat: data collection

Threat: “Big Brother”

Solution: Decentralization

Other sources of threats

Recommendation generation

Output of the RS

Collaborative Filtering (CF)

CF uses the preferences of users with similar interests in order to make recommendations

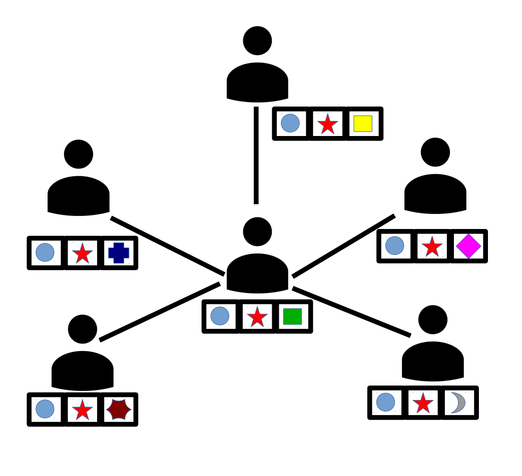
Collaborative Filtering (CF)

CF uses the preferences of users with similar interests in order to make recommendations

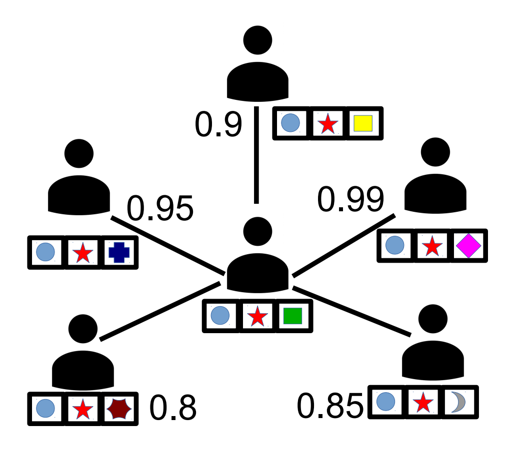
Variants of Collaborative Filtering (CF)

- ...
- User-based CF
- ...

User-based Collaborative Filtering

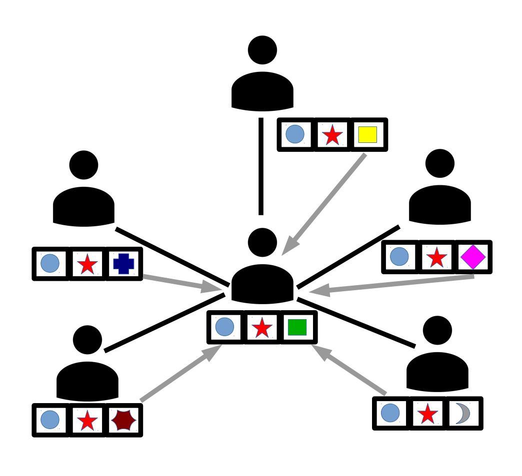


User-based Collaborative Filtering



1. Find most similar users (neighbors)

User-based Collaborative Filtering



1. Find most similar users (neighbors)
2. Take recommendations from neighbors' profile

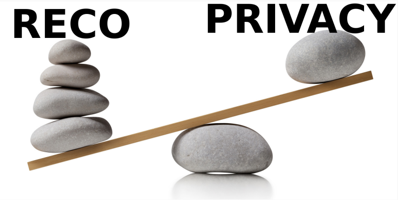
Other sources of threats

Recommendation generation (Similarity computation)

Output of the RS (recommendations themselves)

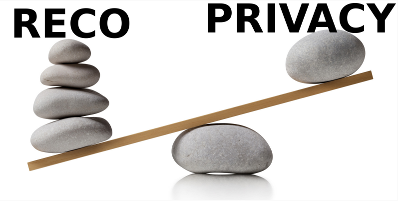
Collaborative-filtering systems are an underestimated threat to user privacy

Now



Collaborative-filtering systems are an underestimated threat to user privacy, and we propose privacy-preserving mechanisms for different stages of recommendation

Now



Our goal

Contributions

Other sources of threats

Recommendation generation (Similarity computation)

Output of the RS (recommendations themselves)

Contributions

Other sources of threats

Recommendation generation (Similarity computation)

Hide & Share
Conceal users profile' content during similarity computation

Output of the RS (recommendations themselves)

Contributions

Other sources of threats

Recommendation generation
(Similarity computation)

Hide & Share
Conceal users profile' content during similarity computation

Output of the RS
(recommendations themselves)

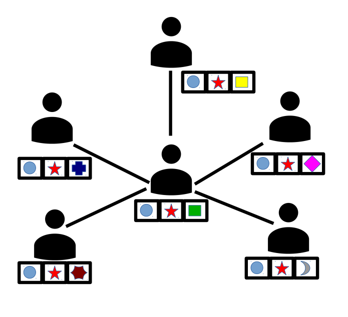
Attack analysis & 2-step
Prevent a type of privacy attack exploiting received recommendations

Contribution: Hide & Share

Decentralized User-based CF

User-based CF for user *U*:

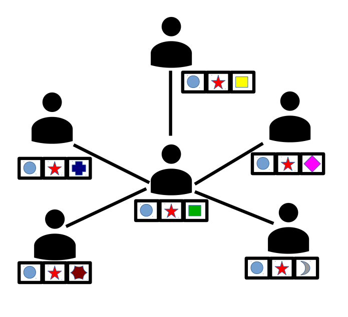
1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)



Decentralized User-based CF

User-based CF for user *U*:

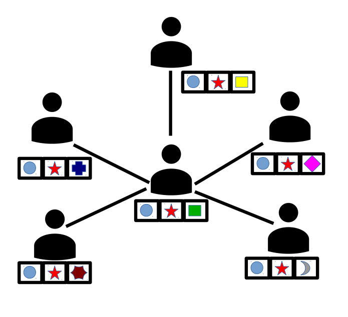
1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile



Decentralized User-based CF

User-based CF for user *U*:

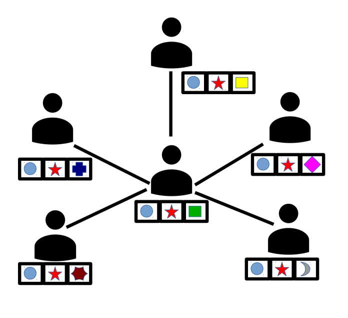
1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile
3. Rank candidate items with a predictor function



Decentralized User-based CF

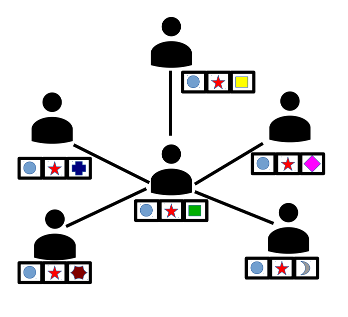
User-based CF for user *U*:

1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile
3. Rank candidate items with a predictor function
4. Recommend to *U* the top-*N* items



User-based CF for user *U*:

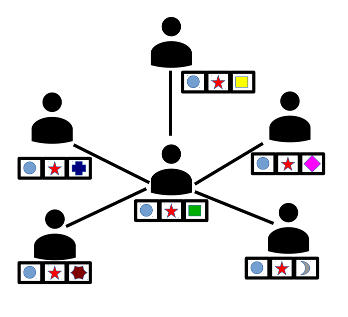
1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile
3. Rank candidate items with a predictor function
4. Recommend to *U* the top-*N* items



Decentralized User-based CF

User-based CF for user *U*:

1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile
3. Rank candidate items with a predictor function
4. Recommend to *U* the top-*N* items

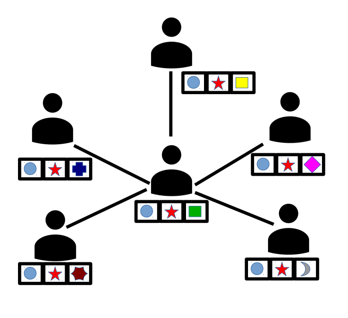


Decentralized version

Step 1: similarity computation = profile exchange

User-based CF for user *U*:

1. Find *U*'s K-Nearest-Neighbors (KNN) w/ similarity metric (*e.g.* cosine)
2. Gather candidate item set from neighbors' profile
3. Rank candidate items with a predictor function
4. Recommend to *U* the top-*N* items



Decentralized version

Step 1: similarity computation = profile exchange

Privacy threat by "Little Brothers", malicious users

“Little Brothers” Adversary Model

“Little Brothers” adversary

Goal: discover target user’s profile by reconstruction attack

“Little Brothers” Adversary Model

“Little Brothers” adversary

Goal: discover target user’s profile by reconstruction attack

Capabilities

- Passive information gathering
- Limited active steps:
  - Eavesdrop
  - Bias randomness
  - Unlimited similarity computations
- No collusion, no Sybil attack

Hide & Share (*H&S*): Landmark-based similarity

Goal

- Measure similarity
- Protect profiles

Hide & Share (*H&S*): Landmark-based similarity

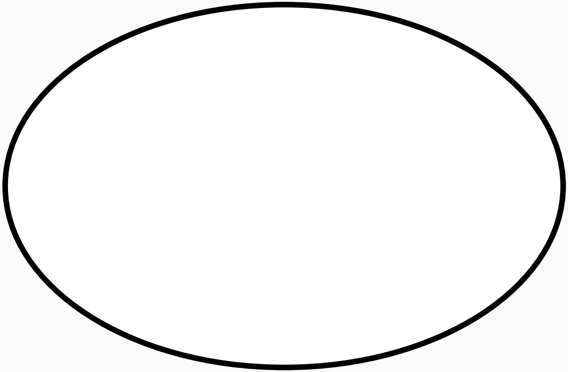
Goal

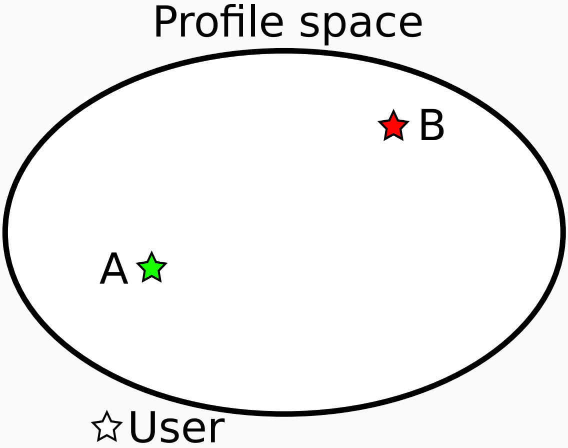
- Measure similarity
- Protect profiles

How?

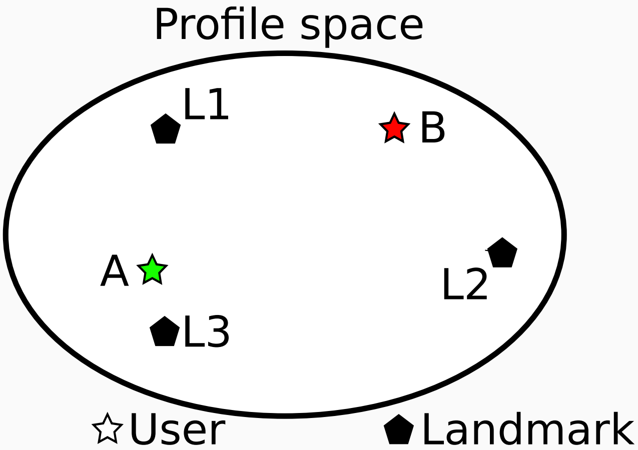
Measure indirectly 2 users' similarity by comparing their respective similarities with random profiles

Profile space

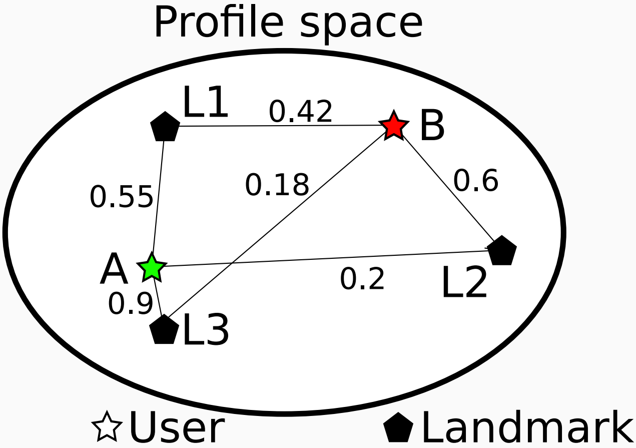



Profile space
B
A
User

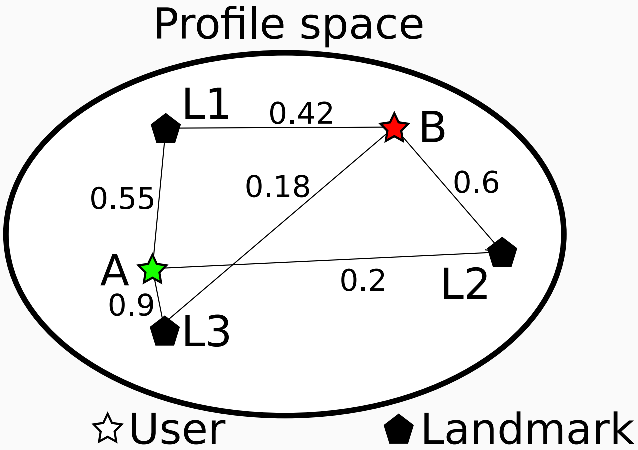
Hide & Share: Toy Example



Hide & Share: Toy Example



Hide & Share: Toy Example



	L1	L2	L3
A	0.55	0.2	0.9
B	0.42	0.6	0.18

Usual profile representation

List of $< itemID, rating >$

Usual profile representation

List of $< itemID, rating >$

Problem

Random profile (landmark) generation?

Hide & Share: Profile Representation

Usual profile representation

List of $< itemID, rating >$

Problem

Random profile (landmark) generation?

Solution: Compact profiles

- Compact profile = Bloom filter
- Containing only liked items

A & B first meeting

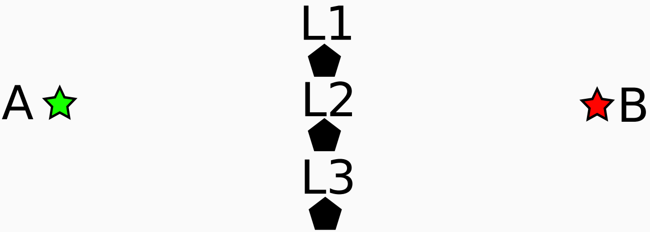
1. Setup a secure communication channel

A & B first meeting

1. Setup a secure communication channel
2. Common secret w/ bit-commitment scheme

A & B first meeting

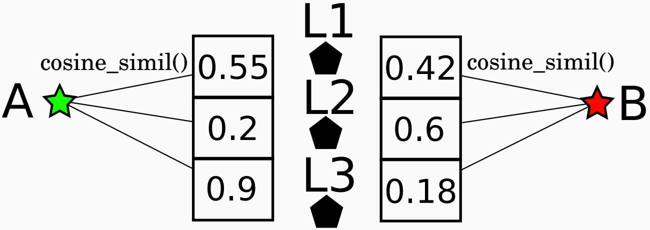
1. Setup a secure communication channel
2. Common secret w/ bit-commitment scheme
3. Derive L random profiles (landmarks) from the secret



Hide & Share: Protocol

A & B first meeting

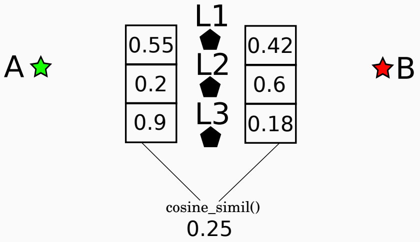
1. Setup a secure communication channel
2. Common secret w/ bit-commitment scheme
3. Derive L random profiles (landmarks) from the secret
4. Similarity computation w/ the landmarks



Hide & Share: Protocol

A & B first meeting

1. Setup a secure communication channel
2. Common secret w/ bit-commitment scheme
3. Derive L random profiles (landmarks) from the secret
4. Similarity computation w/ the landmarks
5. Cosine similarity of ”coordinates” vectors (aka landmark coordinates)



Hide & Share: Protocol

A & B first meeting

1. Setup a secure communication channel
2. Common secret w/ bit-commitment scheme
3. Derive L random profiles (landmarks) from the secret
4. Similarity computation w/ the landmarks
5. Cosine similarity of ”coordinates” vectors (aka landmark coordinates)

When A & B meet again

- Reuse the communication channel and landmarks
- Only steps 4 & 5 remain to be done.

H&S Evaluation

1. Recommendation quality
2. Privacy
3. Overhead

Evaluation: Datasets

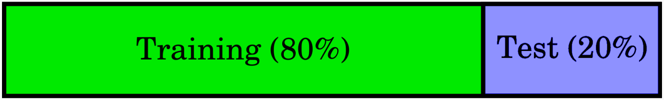
- **MovieLens**: movies recommendation datasets
- **Jester**: jokes recommendation dataset

	# users	# items	# ratings	rating range
ML-100k[1]	943	1,682	100,000	[1..5] (integers)
ML-1M[1]	6,040	3,900	1,000,000	[1..5] (integer)
Jester-1-1[2]	24,983	100	1,810,455	$[-10, 10]$ (reals)

[1]MovieLens: `http://grouplens.org/datasets/movielens/`
[2]Jester: `http://eigentaste.berkeley.edu/dataset/`

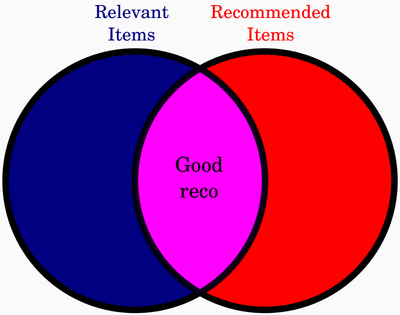
Evaluation: Methodology

1. Datasets split randomly



2. KNN graphs computation
3. Recommendations

Precision & Recall

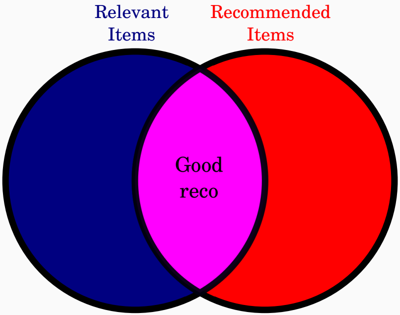


$$precision(user) = \frac{|good|}{|recommended|}$$

$$recall(user) = \frac{|good|}{|relevant|}$$

Evaluation: Recommendation Quality Metrics

Precision & Recall



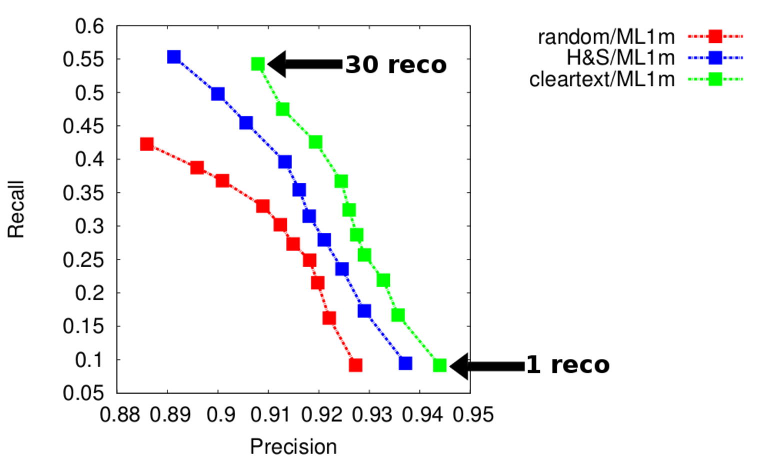
$$precision(user) = \frac{|good|}{|recommended|}$$

$$recall(user) = \frac{|good|}{|relevant|}$$

$$F1score = \frac{2 \times precision \times recall}{precision + recall}$$

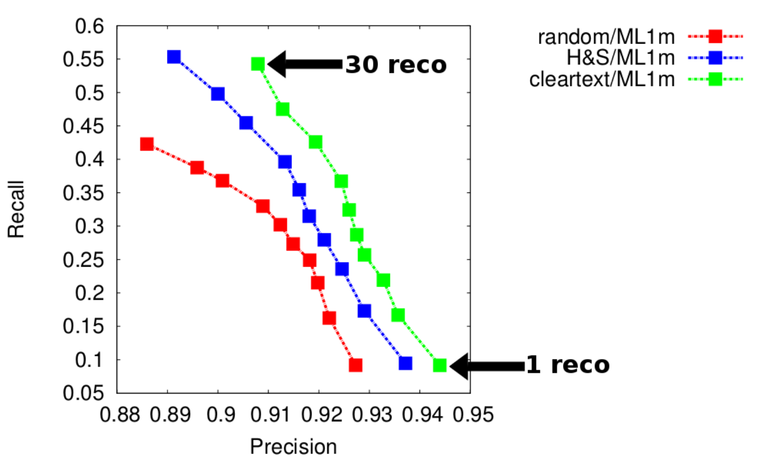
Evaluation: Recommendation Quality

Higher *Recall* & *Precision* = better



Evaluation: Recommendation Quality

Higher *Recall* & *Precision* = better



H&S better than random despite similarity approximation

Evaluation: Neighborhood Quality

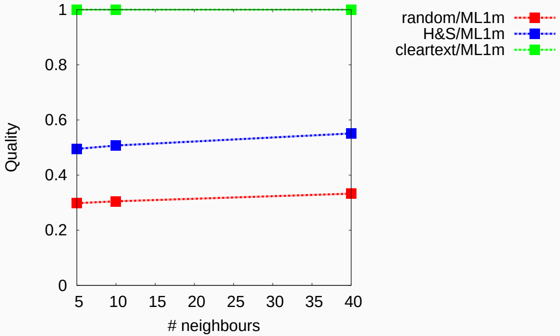
Normalized neighborhood quality

$$quality(user) = \frac{\overline{sim(user, neighborhood)}}{sim(user, idealNeighborhood)}$$

Normalized neighborhood quality

$$quality(user) = \frac{\overline{sim(user, neighborhood)}}{\overline{sim(user, idealNeighborhood)}}$$

Higher *Neighborhood quality* = better

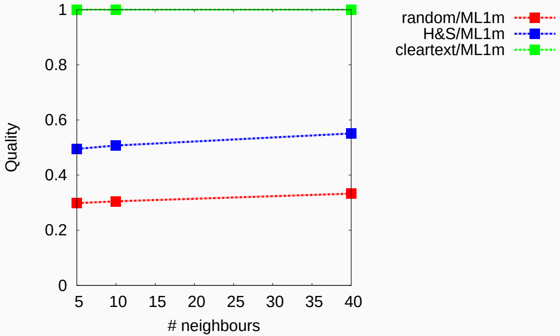


Evaluation: Neighborhood Quality

Normalized neighborhood quality

$$quality(user) = \frac{\overline{sim(user, neighborhood)}}{\overline{sim(user, idealNeighborhood)}}$$

Higher *Neighborhood quality* = better



lower neighborhood quality $\neq$ lower recommendation quality

Evaluation: Empirical Privacy

Profile Reconstruction Attack

1. Infer target's compact profile from landmark coordinates
2. Deduce items forming the compact profile

Evaluation: Empirical Privacy

Profile Reconstruction Attack

1. Infer target's compact profile from landmark coordinates
2. Deduce items forming the compact profile

Basic attack

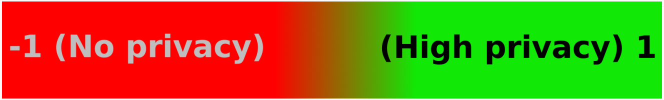
1. Compact profile inference: use the most similar landmark as guessed compact profile
2. Items inference:
    - Adversary knows: items $\leftrightarrow$ compact profile bits
    - Guesses all matching items

Evaluation: Privacy Metric

Set Score

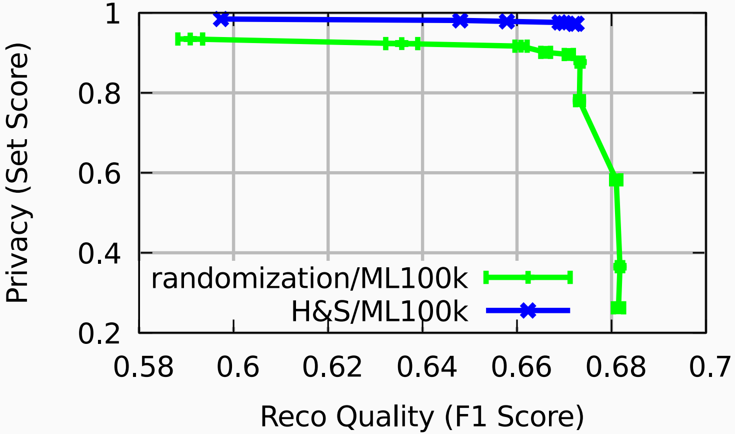
Given a guessed set of items, how much a profile remains private?

- Profiles = sets of items
- G: guessed profile, P: actual profile
- Range:



$$setScore(G, P) = \frac{|G \triangle P| - |G \cap P|}{|G \cup P|}$$

Higher *Set Score* & *F1 Score* = better

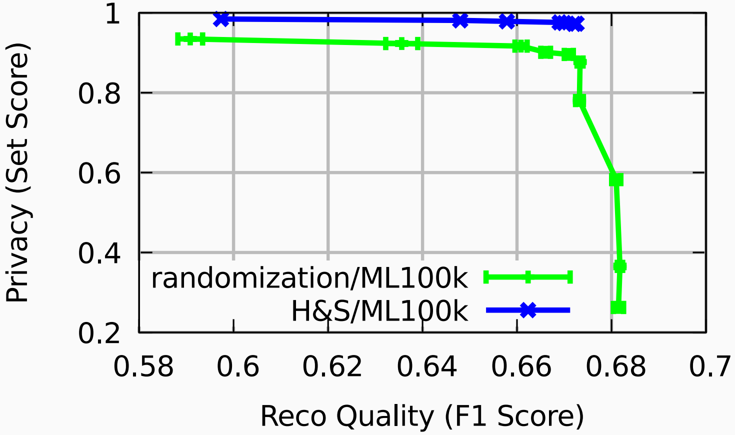


Randomization technique

Randomize a percentage of bits: $\frac{1}{2}$ chance of bit flip

Evaluation: Empirical Privacy

Higher *Set Score* & *F1 Score* = better



H&S: highest privacy & good recommendation quality

Evaluation: Formal Privacy

Upper-bound on leaked information

Knowing the landmarks and the associated coordinates, how much of the profile remains unknown

Upper-bound on leaked information
Knowing the landmarks and the associated coordinates, how much of the profile remains unknown

Conditional entropy: $H(W|V,M)$

where

- W: Compact profile: $\vec{w} = \frac{\vec{c}}{||\vec{c}||}$, $\vec{w} \in \mathbb{R}^n_{[0,1]}$, uniformly distributed
- V: Landmark-based coordinates: $\vec{v} \in \mathbb{R}^m_{[0,1]}$, $\vec{v} = \vec{w}M$
- M: Landmarks: $M \in \mathbb{Z}_2^{n \times m}$, binomial distribution $p = 0.05$

Formal Privacy Evaluation

Manipulating the formula: $H(W|V, M) = H(W) - \mathcal{L}$

$\mathcal{L}$: Upper bound on recoverable information about $\vec{w}$

Formal Privacy Evaluation

Manipulating the formula: $H(W|V, M) = H(W) - \mathcal{L}$

$\mathcal{L}$: Upper bound on recoverable information about $\vec{w}$

Numerical values

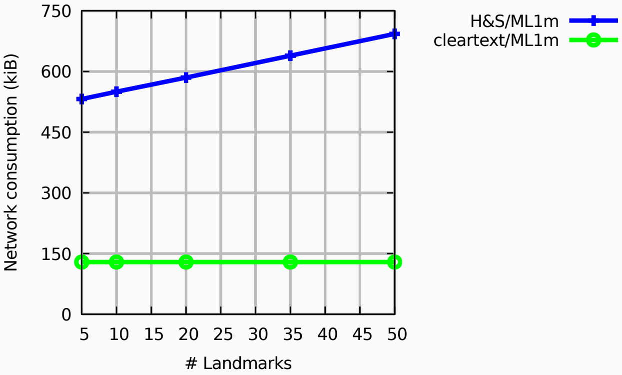
# landmarks	profile size	$\mathcal{L}$	F1 score
$m = 25$	660	660	0.6690
$m = 10$	660	505	0.6602
$m = 7$	660	399	0.6567
$m = 5$	660	338	0.6480
$m = 3$	660	283	0.6360

Evaluation: Communication Overhead

Average communication overhead/round

For 1 peer, over 50 rounds,

Lower *Network consumption* = better

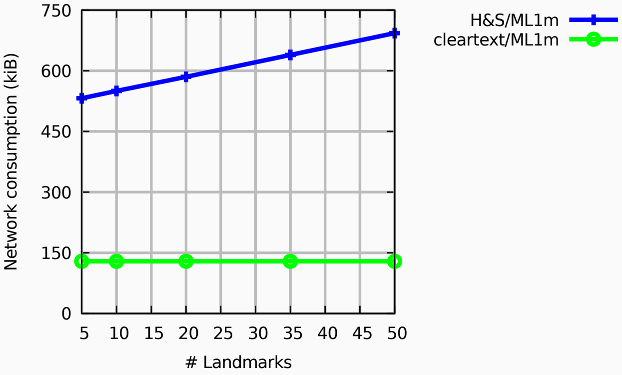


Evaluation: Communication Overhead

Average communication overhead/round

For 1 peer, over 50 rounds, 1 round $\simeq$ 30 sec. $\rightarrow$ 20-25 kiB/s

Lower *Network consumption* = better

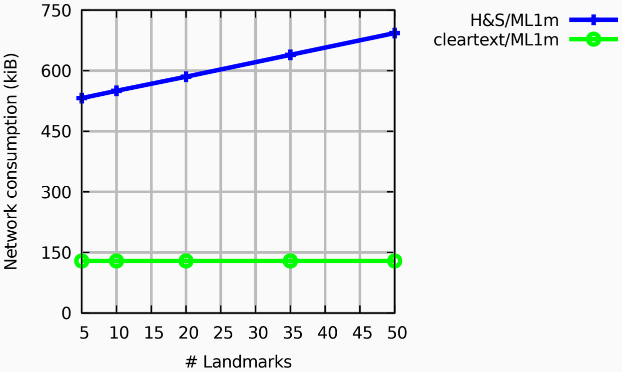


Evaluation: Communication Overhead

Average communication overhead/round

For 1 peer, over 50 rounds, 1 round $\simeq$ 30 sec. $\rightarrow$ 20-25 kiB/s

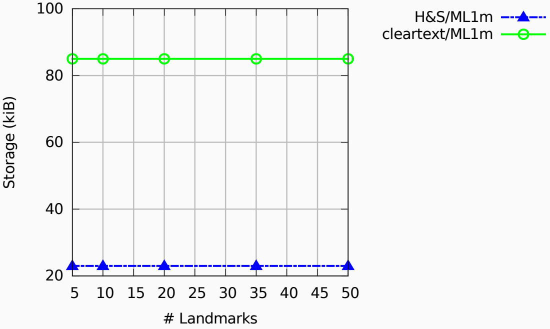
Lower *Network consumption* = better



H&S: reasonable bandwidth by today's standards

Average storage overhead

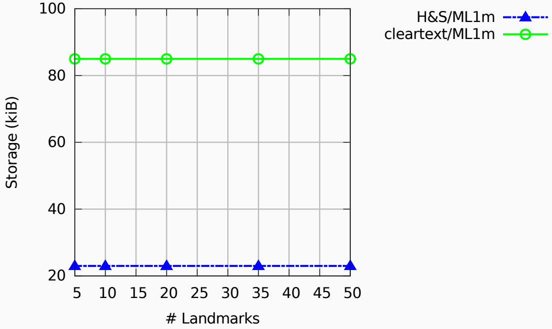
Lower *Storage* = better



Evaluation: Storage & Computational Overhead

Average storage overhead

Lower *Storage* = better

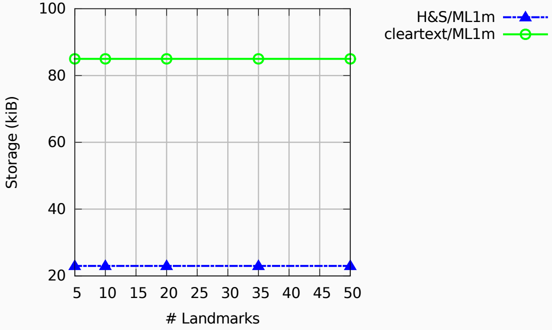


H&S: no profile caching →lower storage requirement

Evaluation: Storage & Computational Overhead

Average storage overhead

Lower *Storage* = better



H&S: no profile caching →lower storage requirement

Computational overhead/peer

Negligible from user's perspective (think HTTPS websites)

Contribution: Hide & Share

Conclusion

Conclusion

Good trade-off

- Reasonable recommendation performance
- Reversing H&S is not trivial (empirical privacy)
- Quantified max. information leak (formal privacy)

Contributions

Other sources of threats

Recommendation generation
(Similarity computation)

Hide & Share
Conceal users profile' content during similarity computation

Output of the RS
(recommendations themselves)

Attack analysis & 2-step
Prevent a type of privacy attack exploiting received recommendations

Contribution: Attack Analysis & $2-step$

Studied Threat

Attack on user privacy

Using:

- The RS's output
- Auxiliary (a priori) information about target

Sybil Attack Overview

Attack against user-based CF, proposed in [CKNFS11][1] but not evaluated

[1] "You Might Also Like:" Privacy Risks of Collaborative Filtering ; by Calandrino, Kilzer, Narayanan, Felten, Shmatikov ; in S&P 2011

Sybil Attack Overview

Attack against user-based CF, proposed in [CKNFS11][1] but not evaluated

Attack Rationale

If you know all but one of your neighbors' profile, unknown items recommended = remaining neighbor

[1] "You Might Also Like:" Privacy Risks of Collaborative Filtering ; by Calandrino, Kilzer, Narayanan, Felten, Shmatikov ; in S&P 2011

Adversary Model

Goal

Discover items from the target's profile

Adversary Model

Goal

Discover items from the target's profile

Capabilities

- Active attack
- Can create fake identities (Sybils)

Adversary Model

Goal

Discover items from the target's profile

Capabilities

- Active attack
- Can create fake identities (Sybils)

Knowledge

- Value of K (of KNN)
- Auxiliary info. about target's profile (*i.e* subset of items)

Adversary Model

Goal

Discover items from the target's profile

Capabilities

- Active attack
- Can create fake identities (Sybils)

Knowledge

- Value of K (of KNN)
- Auxiliary info. about target's profile (*i.e* subset of items)

Sources of Auxiliary Information

Public profile, item/product reviews, free in decentralized systems

Attack Detailed

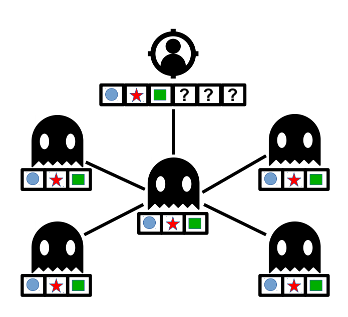
1) Inject Sybils

- Create κ fake identities (Sybils) using Aux. Info.

Attack Detailed

1) Inject Sybils

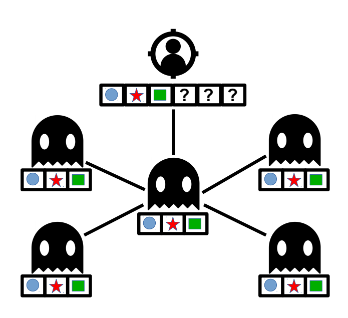
- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →



Attack Detailed

1) Inject Sybils

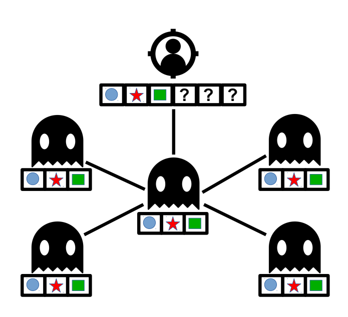
- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →
- In [CKNFS11]: $O(log\ n)$ aux. items = target singled out



Attack Detailed

1) Inject Sybils

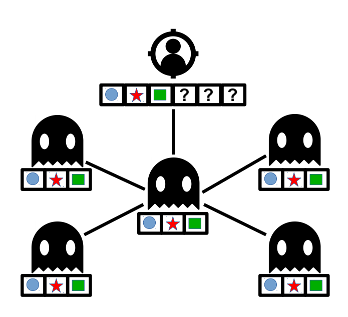
- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →
- In [CKNFS11]: $O(log\ n)$ aux. items = target singled out



Attack Detailed

1) Inject Sybils

- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →
- In [CKNFS11]: $O(log\ n)$ aux. items = target singled out



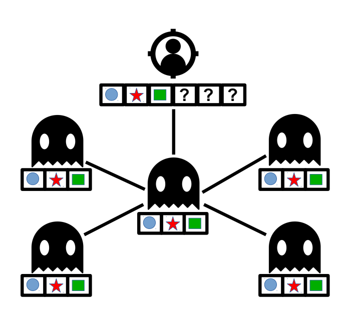
2) Guess Using Recommendations

- Sybil asks RS for recommendations

Attack Detailed

1) Inject Sybils

- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →
- In [CKNFS11]: $O(log\ n)$ aux. items = target singled out



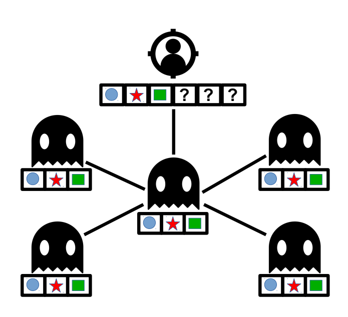
2) Guess Using Recommendations

- Sybil asks RS for recommendations
- Sybil users pool their recommendations

Attack Detailed

1) Inject Sybils

- Create κ fake identities (Sybils) using Aux. Info.
- **Success criterion**: Sybil neighborhood is →
- In [CKNFS11]: $O(log\ n)$ aux. items = target singled out



2) Guess Using Recommendations

- Sybil asks RS for recommendations
- Sybil users pool their recommendations
- If **success criterion** met: recommendations come from target's profile

Attack Evaluation: Methodology

Software

Apache Mahout's user-based collaborative filtering

[3] MovieTweetings: `https://github.com/sidooms/MovieTweetings`

Software

Apache Mahout's user-based collaborative filtering

Datasets

	# users	# items	# ratings	rating range
ML-100k	943	1,682	100,000	[1..5]
Jester-1-1	24,983	100	1,810,455	$[-10, 10]$
MovieTweetings[3]	24,921	15,142	212,835	[0..10]

[3] MovieTweetings: `https://github.com/sidooms/MovieTweetings`

Software

Apache Mahout's user-based collaborative filtering

Datasets

	# users	# items	# ratings	rating range
ML-100k	943	1,682	100,000	[1..5]
Jester-1-1	24,983	100	1,810,455	$[-10, 10]$
MovieTweetings[3]	24,921	15,142	212,835	[0..10]

Impact of similarity metrics

Attack depends on neighborhoods → similarity metrics

[3] MovieTweetings: `https://github.com/sidooms/MovieTweetings`

Attack Evaluation: Similarity Metrics I

$$Cosine(A,N) = \frac{r_A \cdot r_N}{\|r_A\|\|r_N\|}$$

$$Jaccard(A,N) = \frac{|r_A \cap r_N|}{|r_A \cup r_N|}$$

$$Pearson(A,N) = \frac{cov(r_A, r_N)}{\sigma_A \times \sigma_N} = \frac{\sum\limits_{i \in I_{AN}} (r_{A,i} - \overline{r_A})(r_{N,i} - \overline{r_N})}{\sqrt{\sum\limits_{i \in I_{AN}} (r_{A,i} - \overline{r_A})^2 \sum\limits_{i \in I_{AN}} (r_{N,i} - \overline{r_N})^2}}$$

$$Cos\text{-}overlap(u,n) = \frac{u \cdot n}{\sqrt{\sum\limits_{i \in I_{un}} (u_i)^2} \times \sqrt{\sum\limits_{i \in I_{un}} (n_i)^2}}$$

Attack Evaluation: Similarity Metrics II

$$CosineAvg(u, n) = \frac{\sum\limits_{i \in I_u \cup I_n} u_i \times n_i}{\|u\| \|n\|}$$

$$WUP\text{-}u(u, n) = \frac{\sum\limits_{i \in I_{un}} u_i \times n_i}{\sqrt{\sum\limits_{i \in I_{un}} (u_i)^2} \times \sqrt{\sum\limits_{i \in I_n} (n_i)^2}}$$

$$WUP\text{-}n(u, n) = \frac{\sum\limits_{i \in I_{un}} u_i \times n_i}{\sqrt{\sum\limits_{i \in I_u} (u_i)^2} \times \sqrt{\sum\limits_{i \in I_{un}} (n_i)^2}}$$

Attack Evaluation: Success Metrics

Attack Success

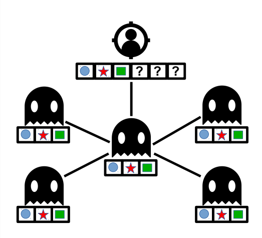
- Yield
  - Number of guesses
- Accuracy
  - Fraction of correct guesses
- Expected neighborhoods
  - Out of the κ Sybils, how many meeting **success criterion**?

Attack Evaluation: Success Metrics

Attack Success

- Yield
  - Number of guesses
- Accuracy
  - Fraction of correct guesses
- Expected neighborhoods
  - Out of the κ Sybils, how many meeting **success criterion**?

Criterion met

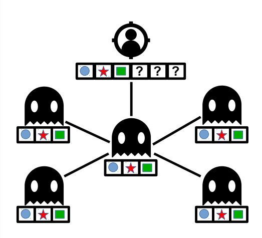


Attack Evaluation: Success Metrics

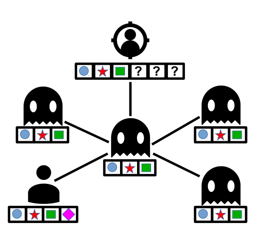
Attack Success

- Yield
  - Number of guesses
- Accuracy
  - Fraction of correct guesses
- Expected neighborhoods
  - Out of the κ Sybils, how many meeting **success criterion**?

Criterion met



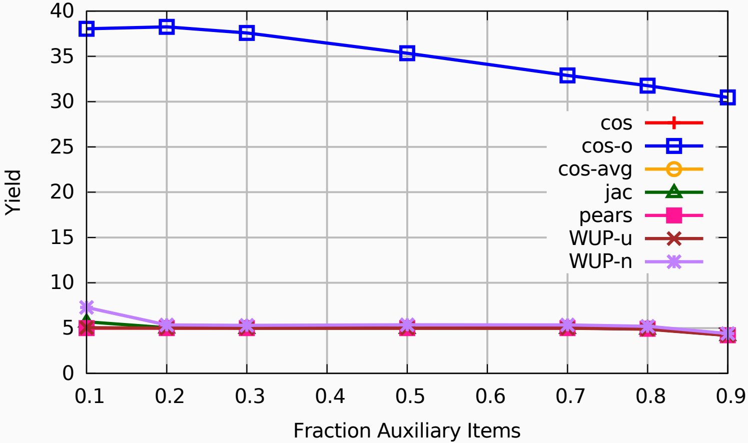
Criterion not met



Attack Success Evaluation I

Sybils ask 5 recommendations each

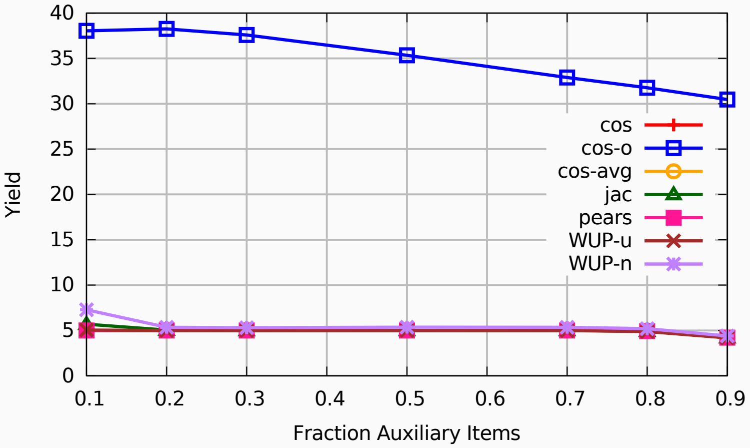
Lower *Yield* = better privacy



Attack Success Evaluation I

Sybils ask 5 recommendations each

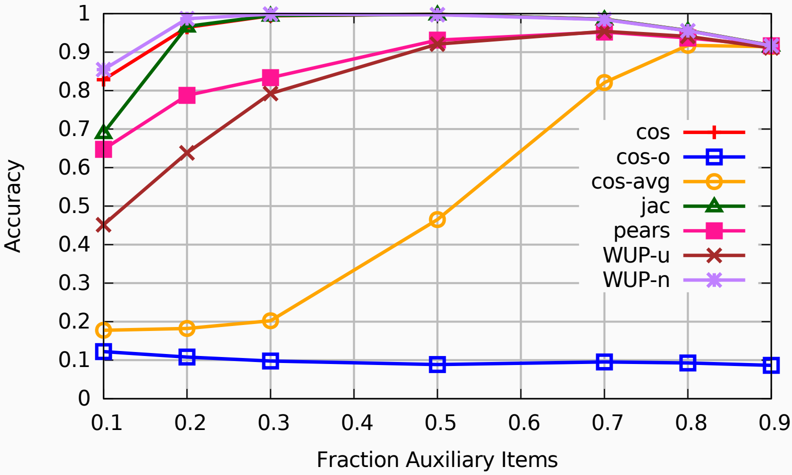
Lower *Yield* = better privacy



2 behaviors: *Cos-overlap* and the other metrics

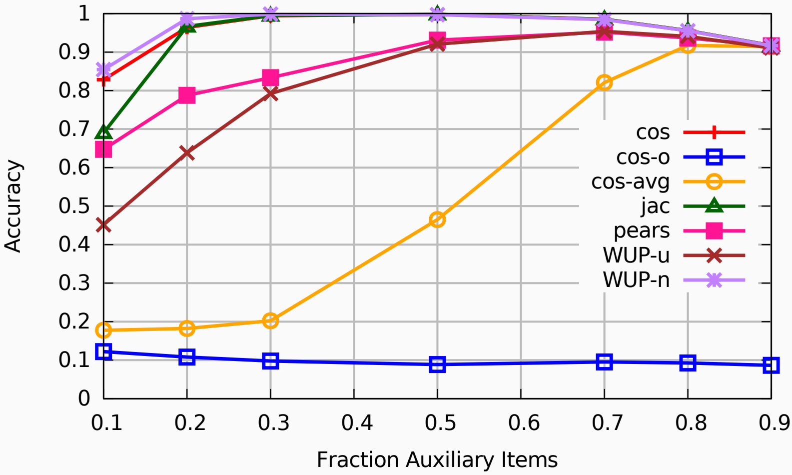
Attack Success Evaluation II

Lower *Accuracy* = better privacy



Attack Success Evaluation II

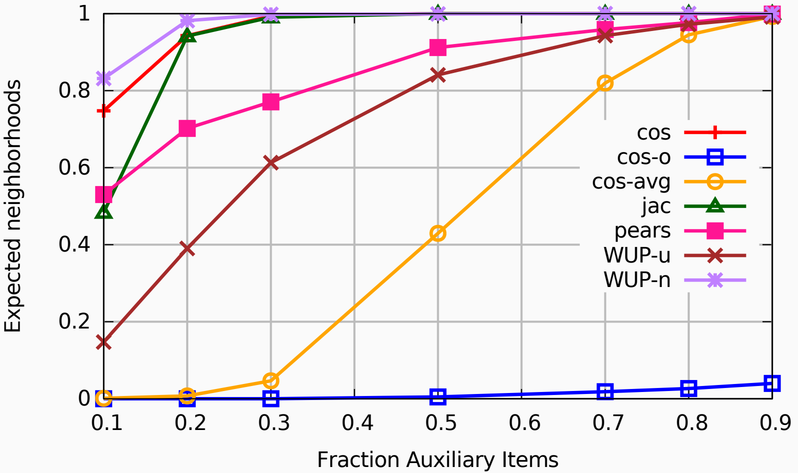
Lower *Accuracy* = better privacy



Only *Cos-overlap* resists the attack w/ many auxiliary items

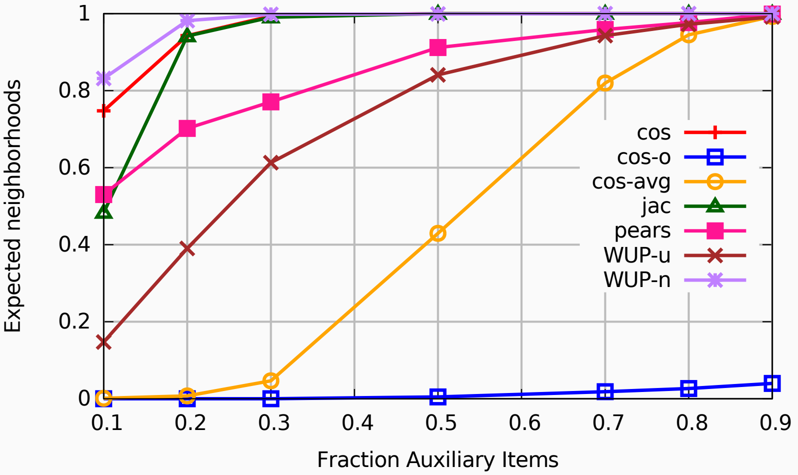
Attack Success Evaluation III

Lower *Expected neighborhoods* = better privacy



Attack Success Evaluation III

Lower *Expected neighborhoods* = better privacy



Coarse similarity by *Cos-overlap* defeats the attack

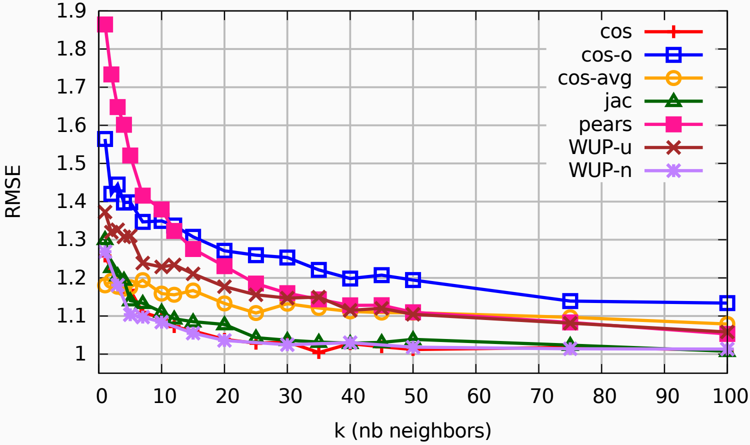
Similarity Metrics Evaluation: Recommendation Quality

Root Mean Square Error $RMSE(A) = \sqrt{\frac{\sum_{i=1}^{n}(pred_{A,i} - r_{A,i})^2}{n}}$

Similarity Metrics Evaluation: Recommendation Quality

Root Mean Square Error $RMSE(A) = \sqrt{\frac{\sum_{i=1}^{n}(pred_{A,i} - r_{A,i})^2}{n}}$

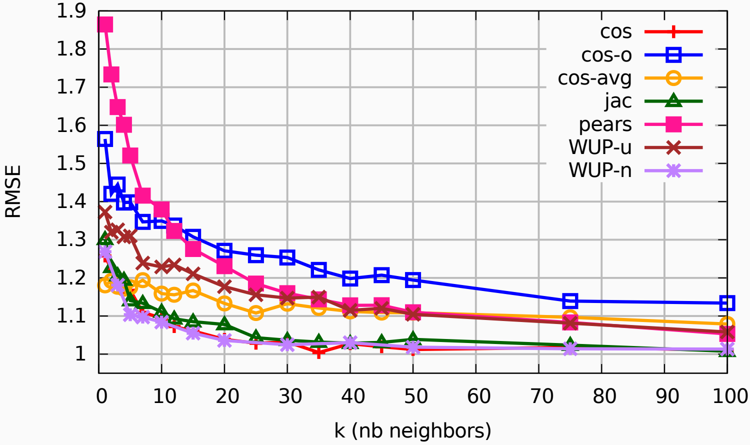
Lower *RMSE* = better recommendations



Similarity Metrics Evaluation: Recommendation Quality

Root Mean Square Error $RMSE(A) = \sqrt{\frac{\sum_{i=1}^{n}(pred_{A,i} - r_{A,i})^2}{n}}$

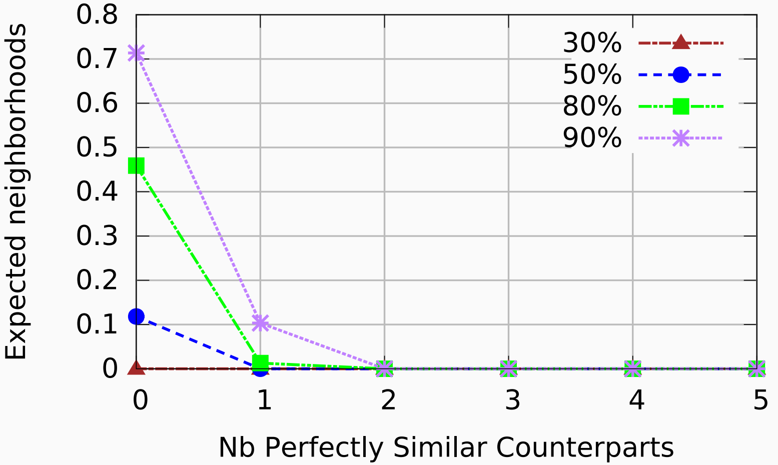
Lower $RMSE$ = better recommendations



Correlation of recommendation quality & attack resilience

Understanding *Cos-overlap*'s Resiliency

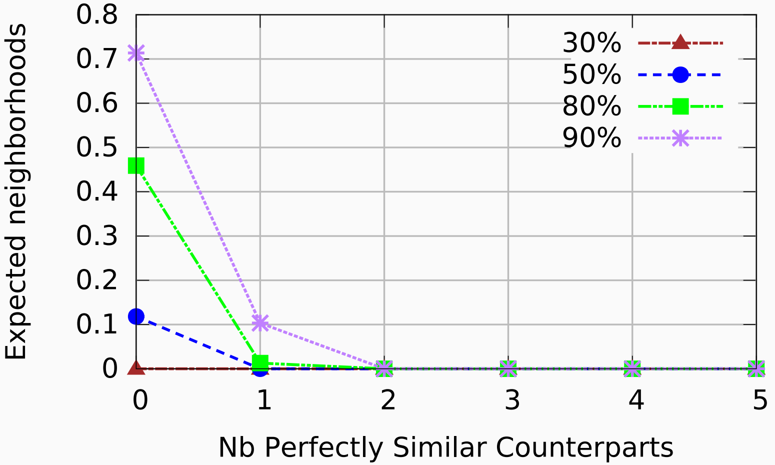
Lower *Expected neighborhoods* = better privacy



Perfecty Similar Counterpert (PSC): Users 100% similar to target

Understanding *Cos-overlap*'s Resiliency

Lower *Expected neighborhoods* = better privacy



1+ PSCs prevent Sybil from having an expected neighborhood

Attack Resiliency: Perfectly Similar Counterparts

The attack fails when the target has PSCs

Attack Resiliency: Perfectly Similar Counterparts

The attack fails when the target has PSCs

Counter-measure Idea

Combine:

- *Cos-overlap*'s coarse similarity approximation (creating PSCs)
- good recommendation quality

1. Make similar enough users indistinguishable from each other

2−*step* Overview

1. Make similar enough users indistinguishable from each other
    - Users with $Cosine \geq th$, similarity capped th

2−*step* Overview

1. Make similar enough users indistinguishable from each other
    - Users with $Cosine \geq th$, similarity capped th

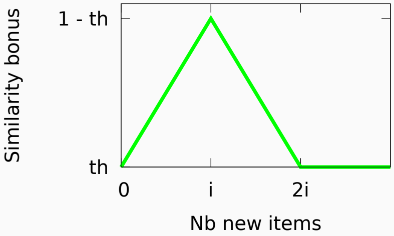
2. Select among them the most useful ones for recommendation

2–*step* Overview

1. Make similar enough users indistinguishable from each other
    - Users with $Cosine \geq th$, similarity capped th
2. Select among them the most useful ones for recommendation
    - Similarity bonus depending on the number of “new” items

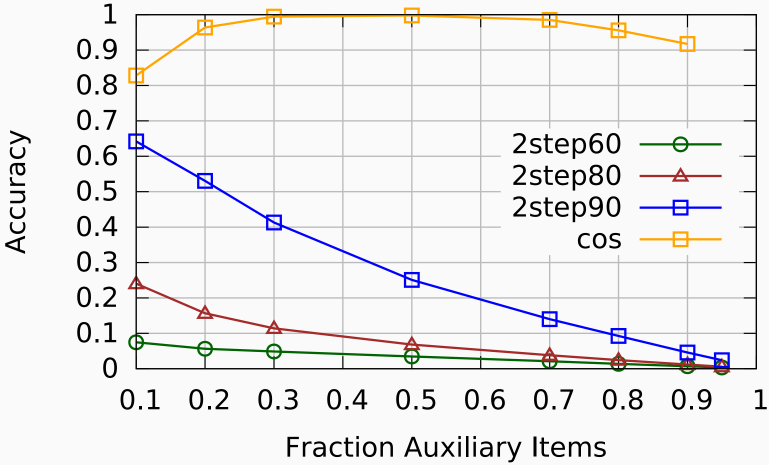
$2-$*step* Overview

1. Make similar enough users indistinguishable from each other
    - Users with $Cosine \geq th$, similarity capped th
2. Select among them the most useful ones for recommendation
    - Similarity bonus depending on the number of “new” items
    - Users with i “new” items get a similarity bonus of $1 - th$



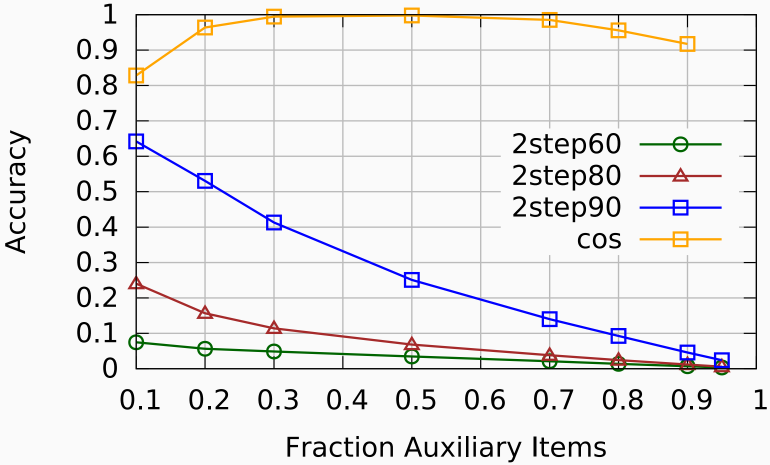
Attack Success Evaluation with $2-step$ I

Lower *Accuracy* = better privacy



Attack Success Evaluation with $2-step$ I

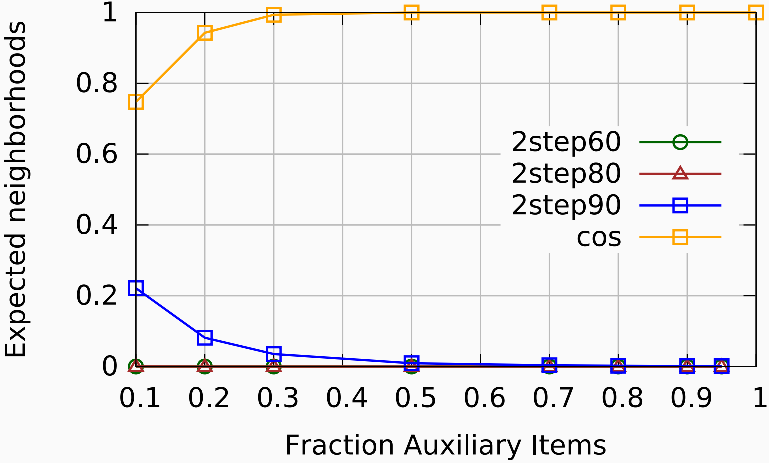
Lower *Accuracy* = better privacy



$2-step$: good attack resiliency, esp. with low *th*

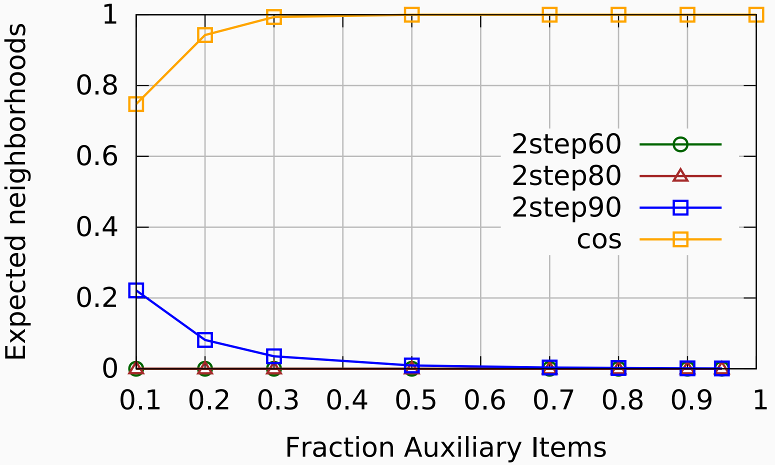
Attack Success Evaluation with $2-step$ II

Lower *Expected neighborhoods* = better privacy



Attack Success Evaluation with 2−*step* II

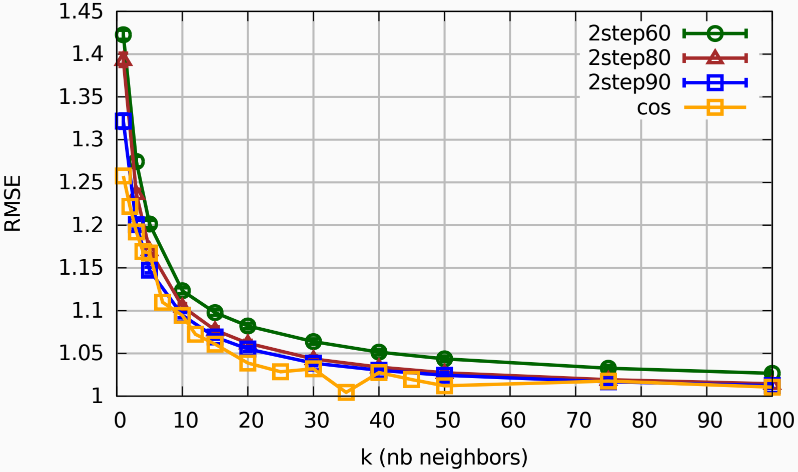
Lower *Expected neighborhoods* = better privacy



2−*step*: Expected neighborhoods rarely obtained

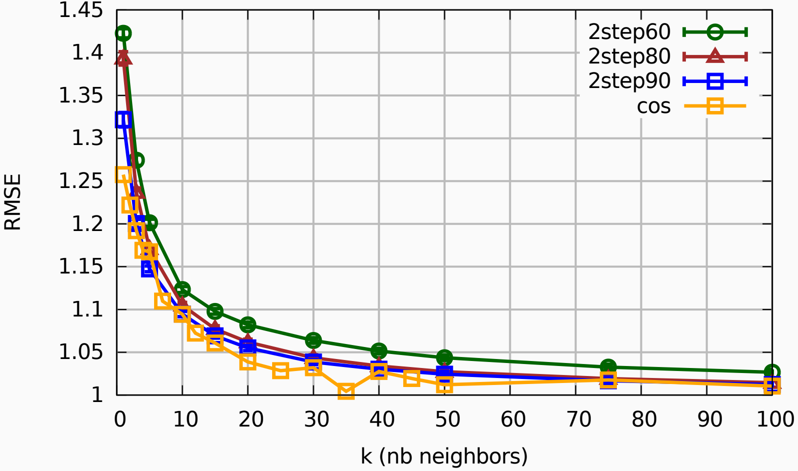
Recommendation Quality of $2-step$

Lower *RMSE* = better recommendations



Recommendation Quality of $2\text{−}step$

Lower *RMSE* = better recommendations



$2\text{−}step$: Recommendation quality close to *Cosine*'s

Contribution: Attack Analysis & $2-step$

Conclusion

Conclusion

We addressed a privacy threat via recommendations

Conclusion

We addressed a privacy threat via recommendations

Sybil Attack Study

- Generally effective attack w/o PSCs
- Higher than expected required level of auxiliary knowledge

Conclusion

We addressed a privacy threat via recommendations

Sybil Attack Study

- Generally effective attack w/o PSCs
- Higher than expected required level of auxiliary knowledge

Counter-measure: $2-step$
Promising preliminary evaluation:

- Good attack resiliency (better than *Cos-overlap*'s)
- Good recommendation quality (close to *Cosine*'s)

Conclusion

Summary

- Recommendation = useful but need more privacy
- Addressed 2 types of threat:
  - During recommendation generation
  - From recommendations themselves

Conclusion

Summary

- Recommendation = useful but need more privacy
- Addressed 2 types of threat:
  - During recommendation generation
  - From recommendations themselves

Contributions

- Privacy during similarity computation, *Hide & Share*
- Twofold contribution:
  - Evaluation of a Sybil attack on user privacy
  - Privacy-preserving counter-measure, 2—*step*

Perspectives: Short-term

Hide & Share

- Stronger adversary (*e.g.* collusion)
- Privacy-preservation after KNN computation

Perspectives: Short-term

Hide & Share

- Stronger adversary (*e.g.* collusion)
- Privacy-preservation after KNN computation

2−*step*

- Knowledge of k for the adversary
- Test on a real-world RS
- Further evaluation of 2−*step* in progress

Perspectives: Long-term

- Privacy impact heuristics in RSs
- Do-Not-Track-like mechanisms for RSs
- Study more attacks to raise awareness about privacy threats of RSs
- User data monetization

Thank You