Nudge: A Private Recommendation Engine

Alexandra Henzinger
MIT

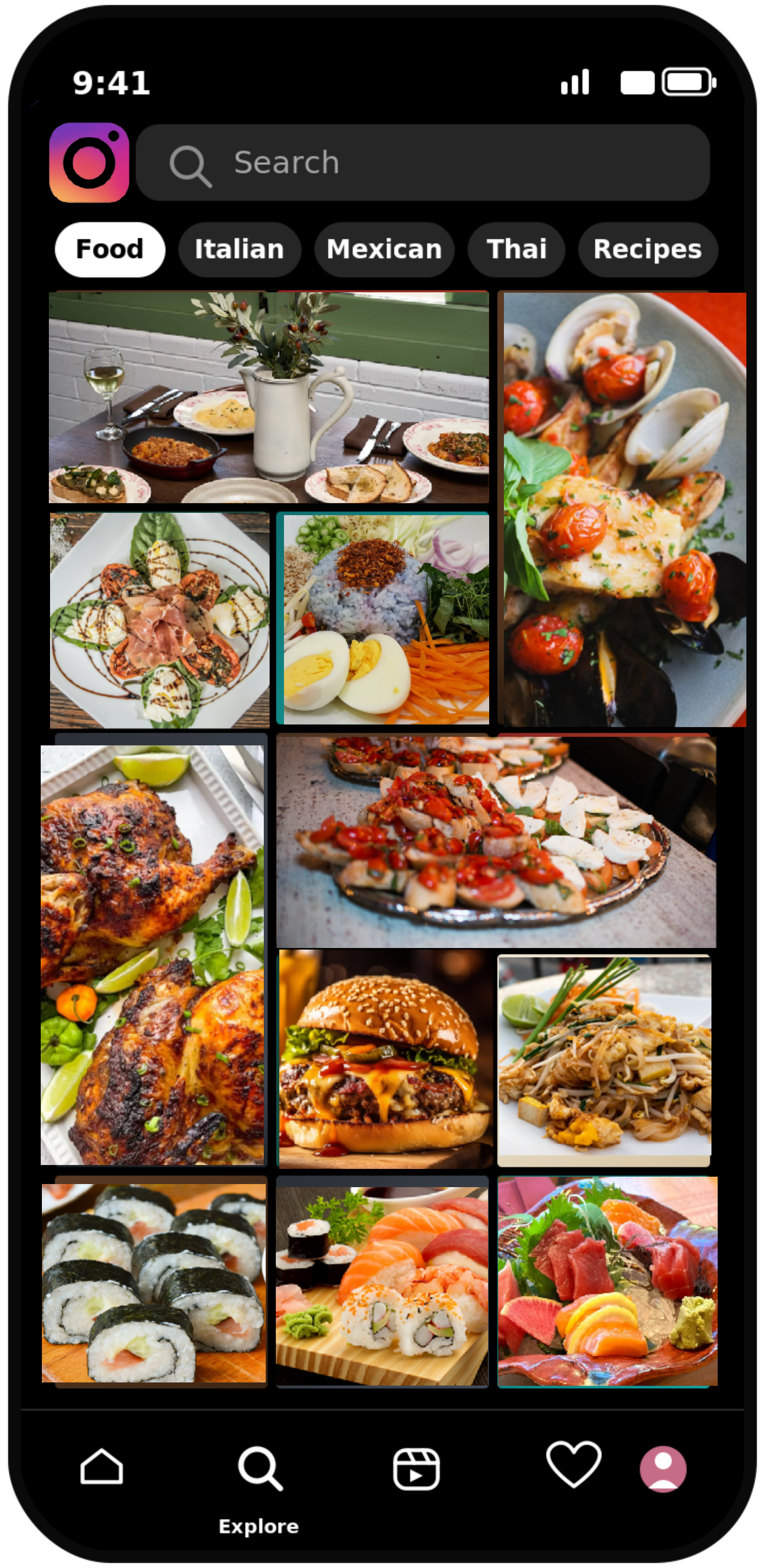
Emma Dauterman
Stanford

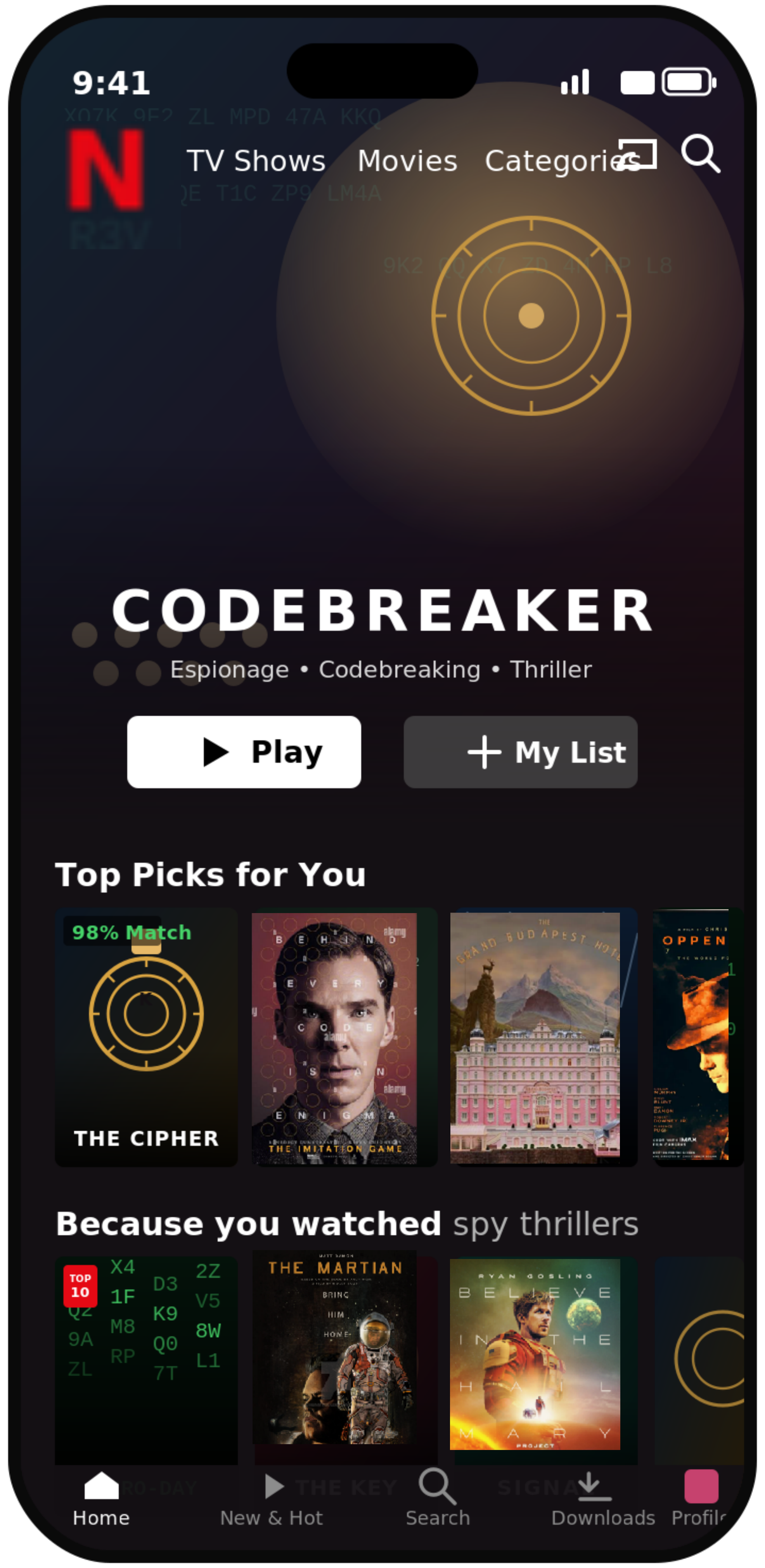
Henry Corrigan-Gibbs
UC Berkeley

Dan Boneh
Stanford

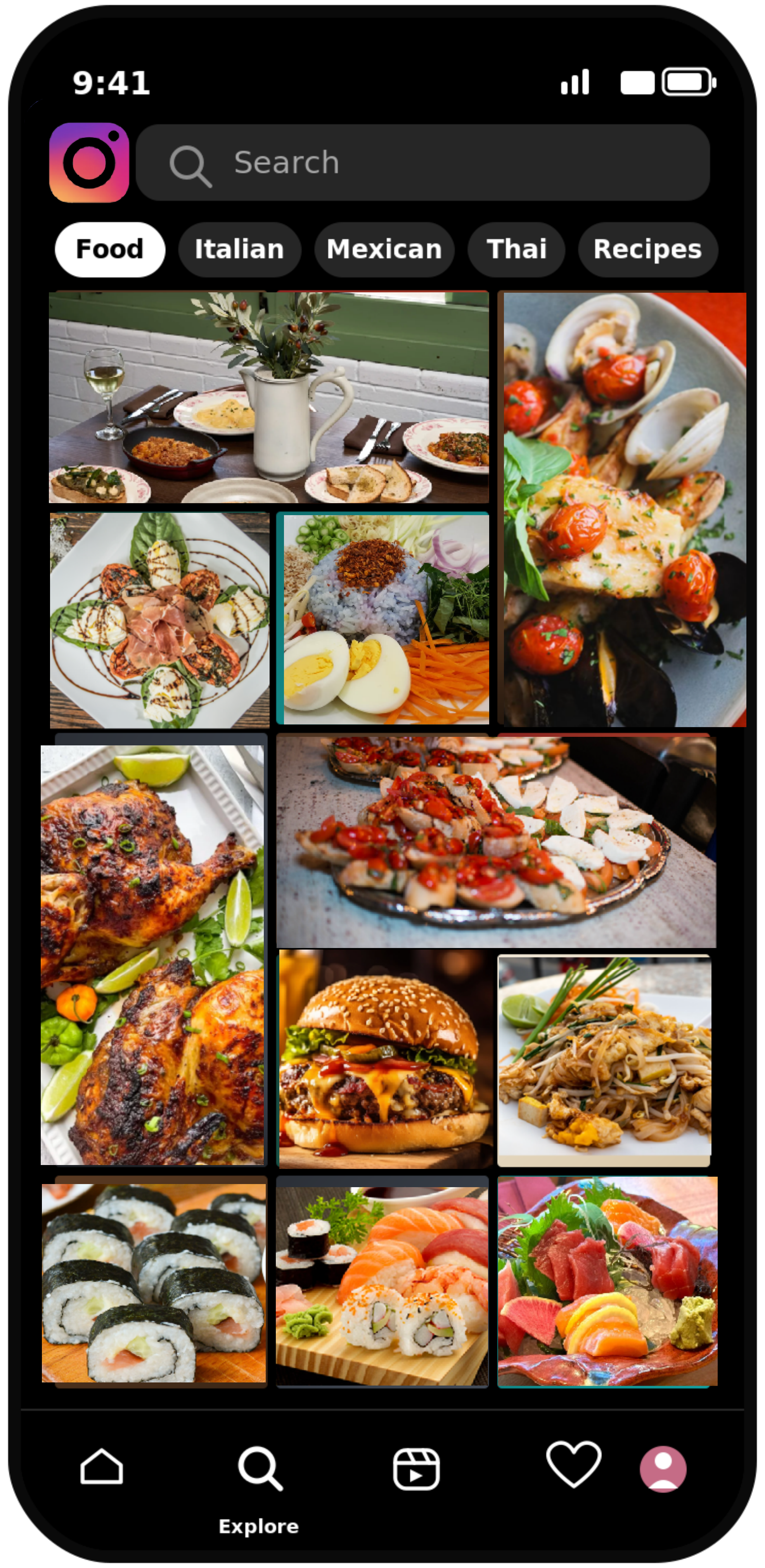
USENIX Security 2026

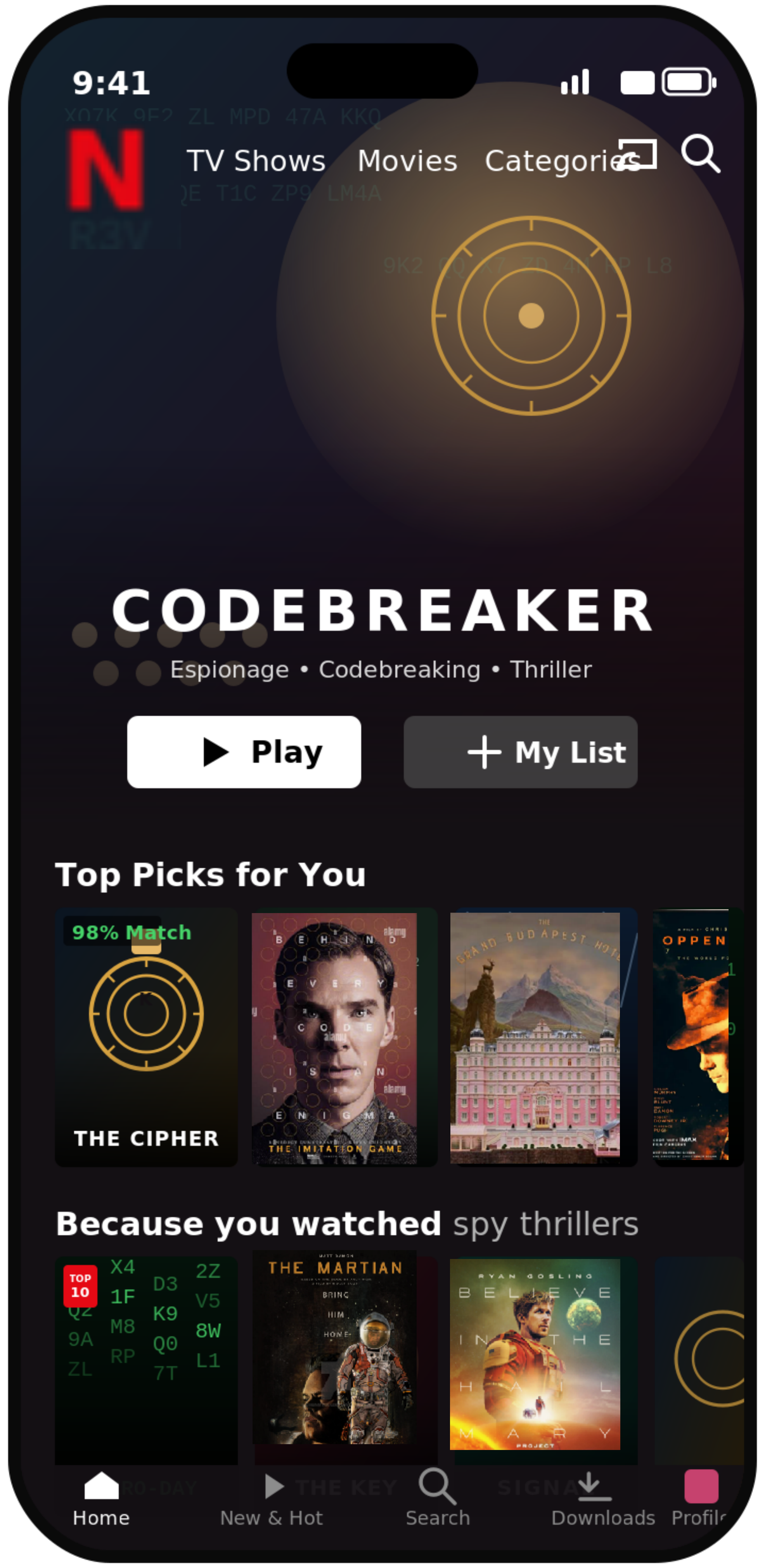
Today: Recommender systems learn from our every click



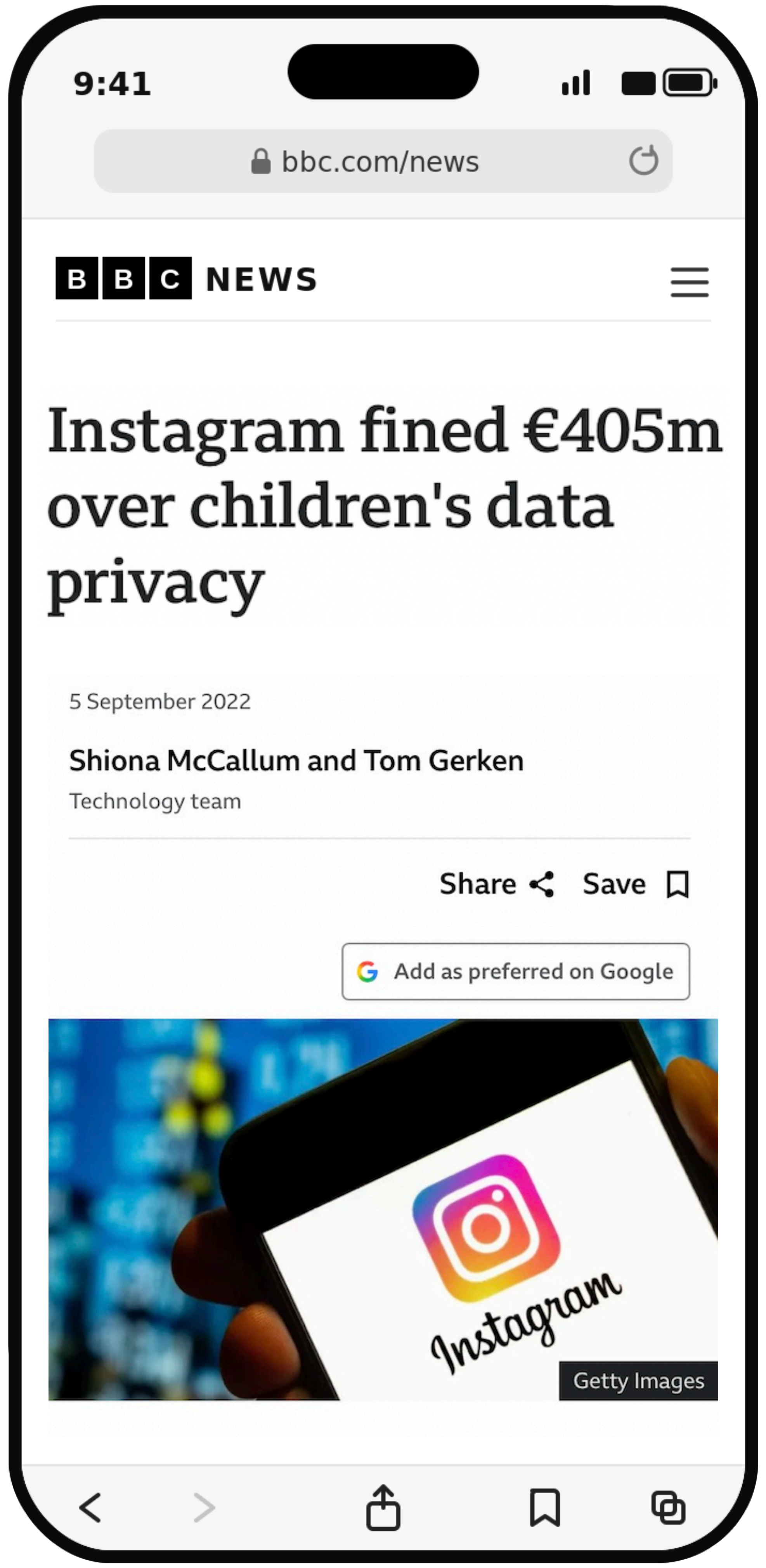


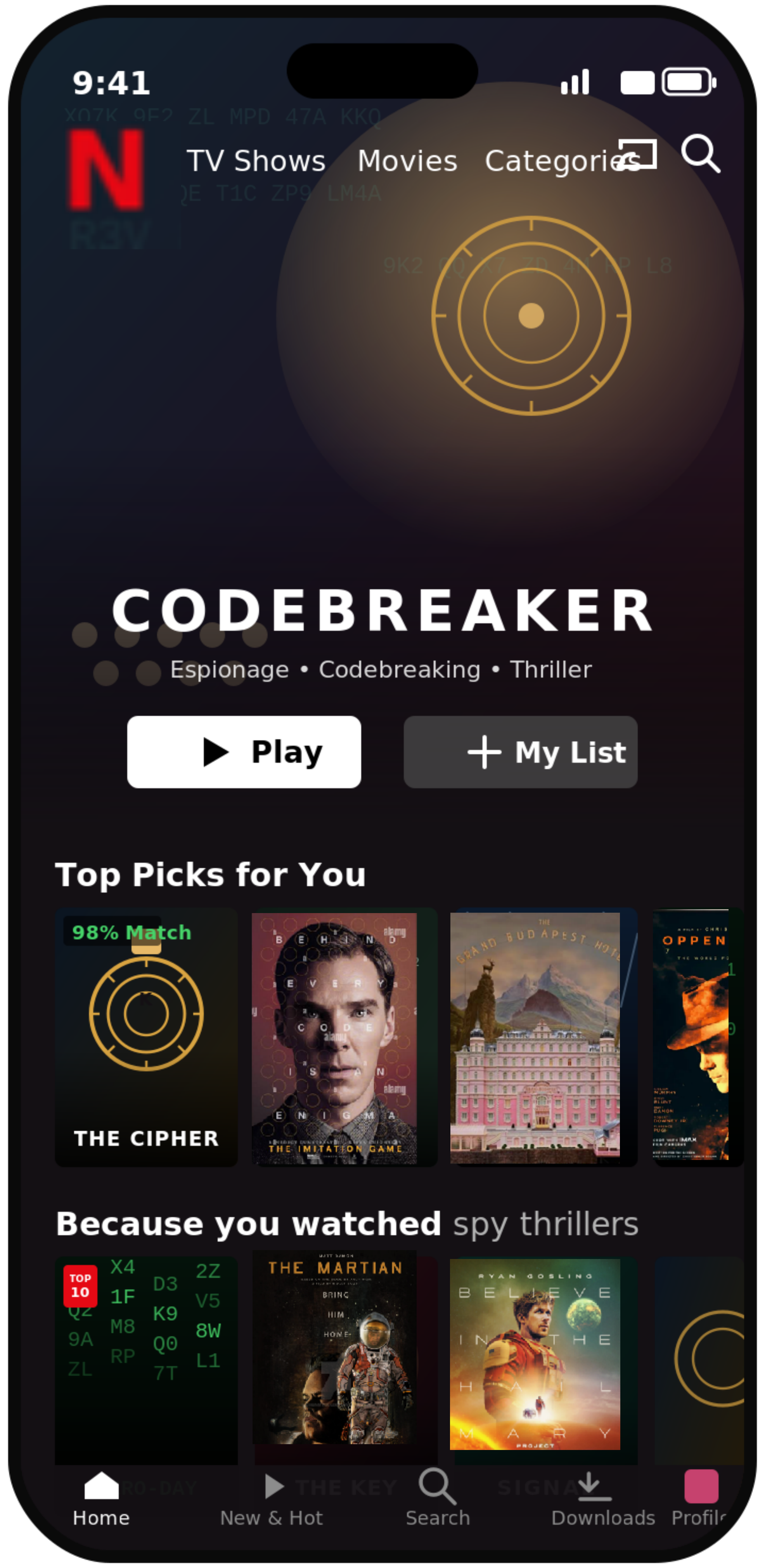
Today: Recommender systems learn ~~from~~ our every click



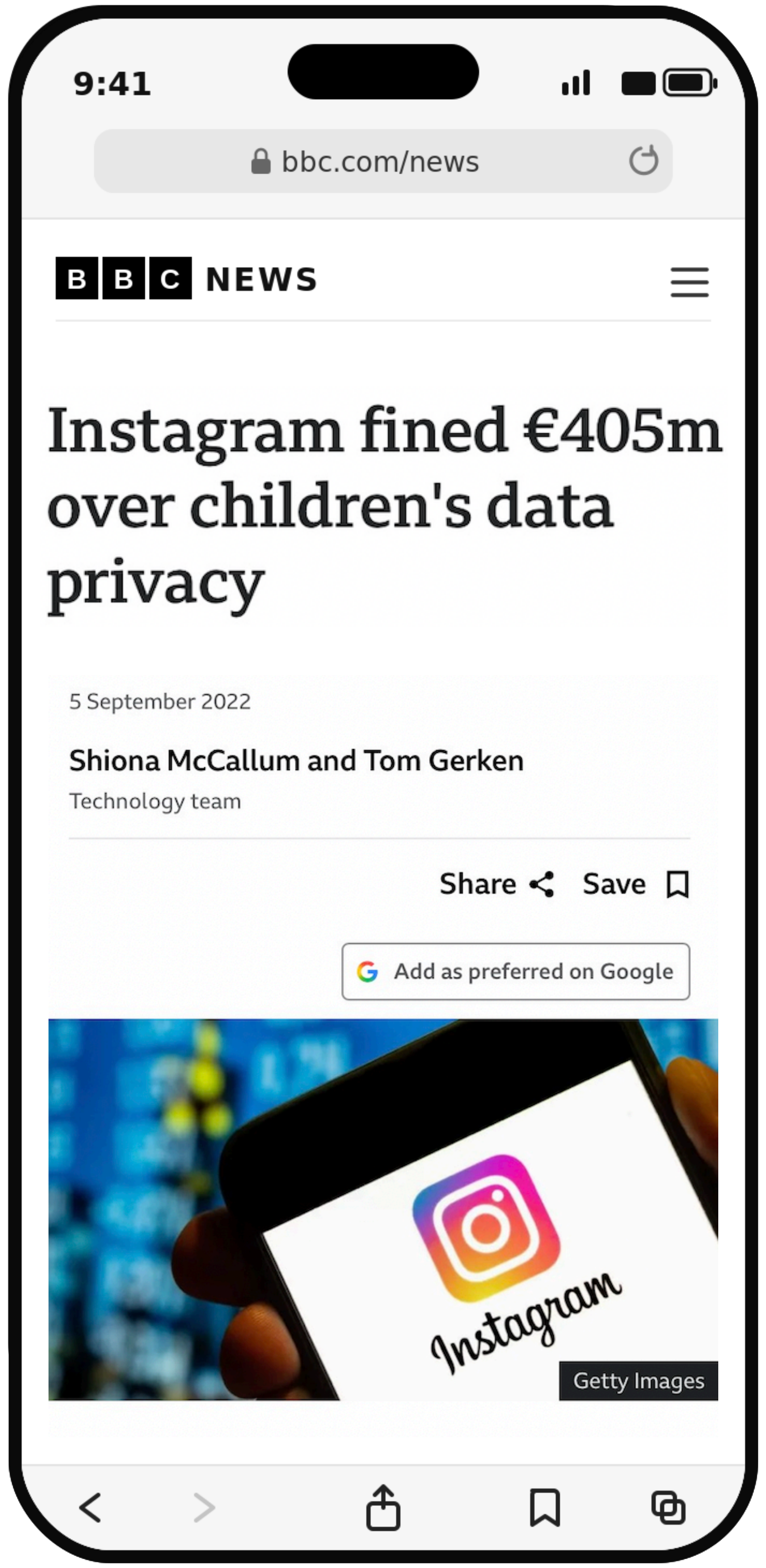


Today: Recommender systems learn ~~from~~ our every click





Today: Recommender systems learn ~~from~~ our every click



Two approaches to private recommenders

Scale

1. Efficient but leaky

- Anonymization: reversible via attacks
- Trusted hardware: vulnerable to side channels
- Federated learning: leaks intermediate states

[GKS08,NS08,DDKP22,LBVKH23,MWAPR23,RSWP23,BGLRS25,…]

2. Cryptographically private but small-scale

- Garbled circuits or bespoke 4PC: run at thousand-user scale

[NIWJTB13, GraphSC15, KKKYSK18, PIRSONA21]

Privacy

Two approaches to private recommenders

Scale

1. Efficient but leaky

- Anonymization: reversible via attacks
- Trusted hardware: vulnerable to side channels
- Federated learning: leaks intermediate states

[GKS08,NS08,DDKP22,LBVKH23,MWAPR23,RSWP23,BGLRS25,…]

?

2. Cryptographically private but small-scale

- Garbled circuits or bespoke 4PC: run at thousand-user scale

[NIWJTB13, GraphSC15, KKKYSK18, PIRSONA21]

Privacy

This talk:

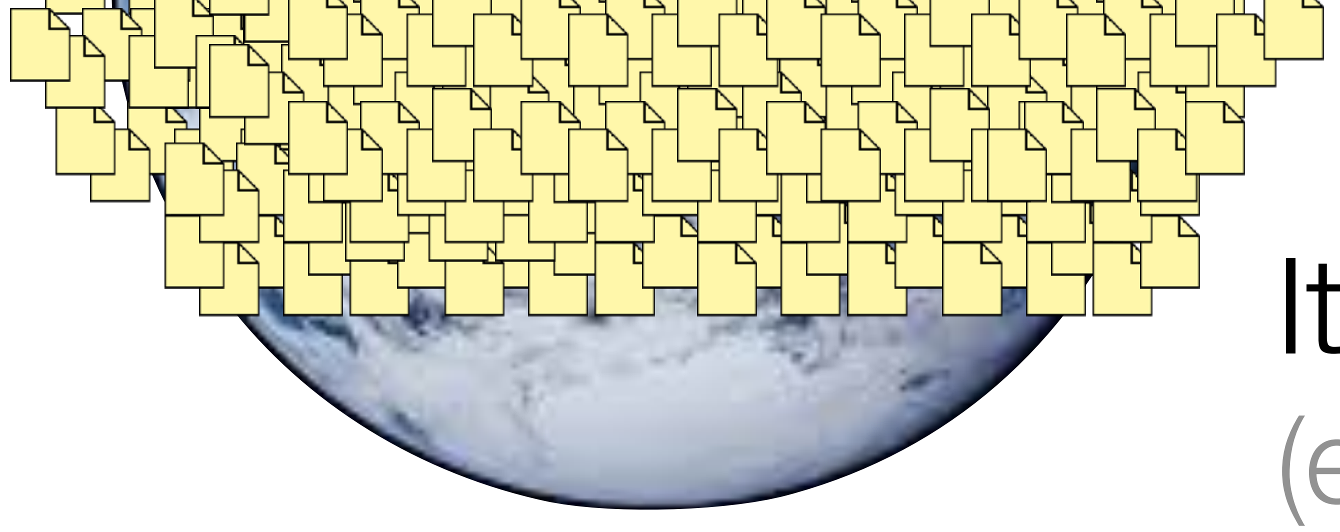
Nudge, a recommender with cryptographic privacy

+ Learns & serves recommendations **without seeing any single user's data**
  using three non-colluding service providers

+ Matches popular matrix-factorization algorithms in quality

+ Runs at **million-user scale**
  e.g., with > 75x more users and 50x more ratings than prior systems

- Needs other tools to privately fetch content (iCloud private relay, Tor, or PIR)

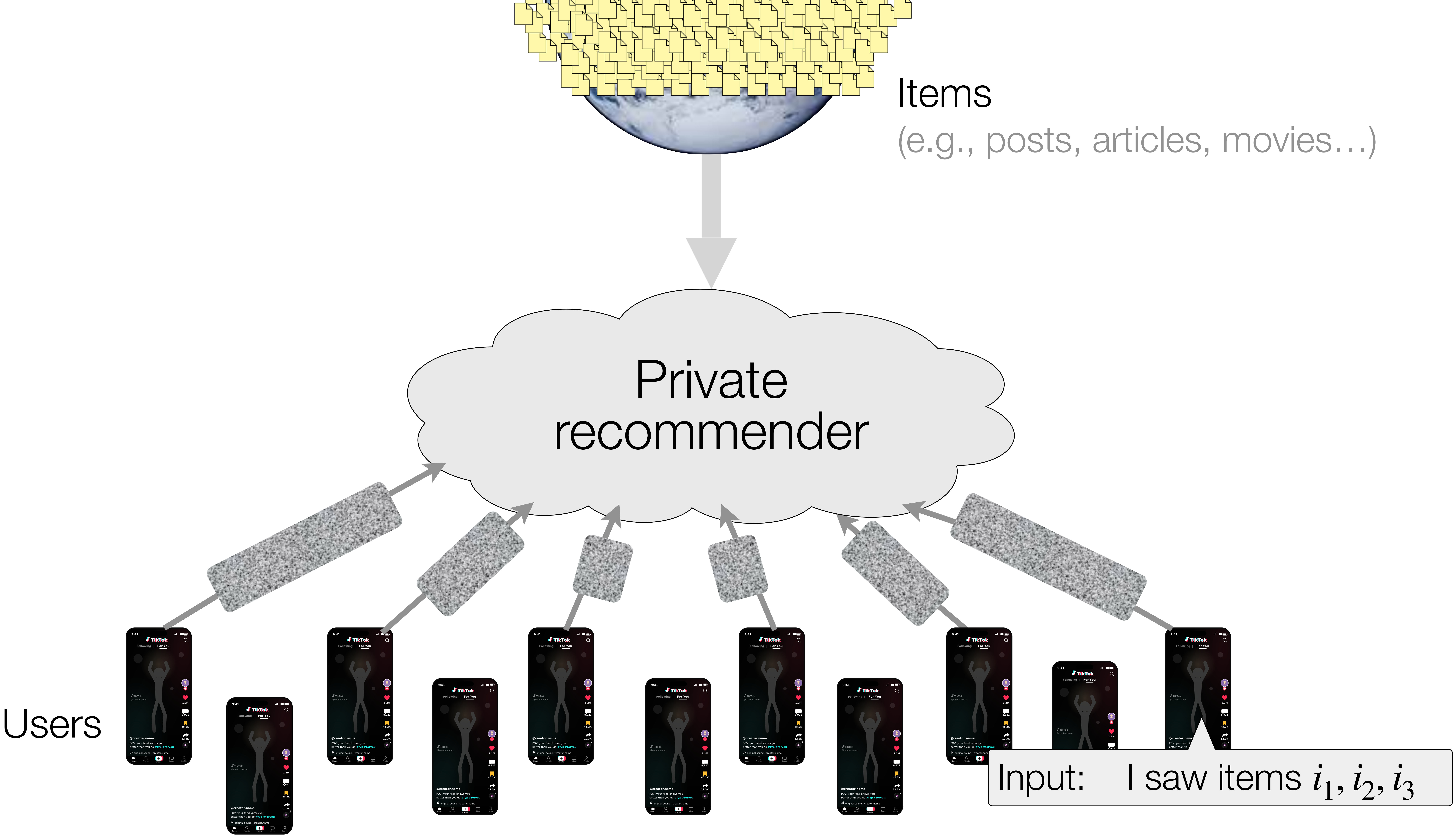


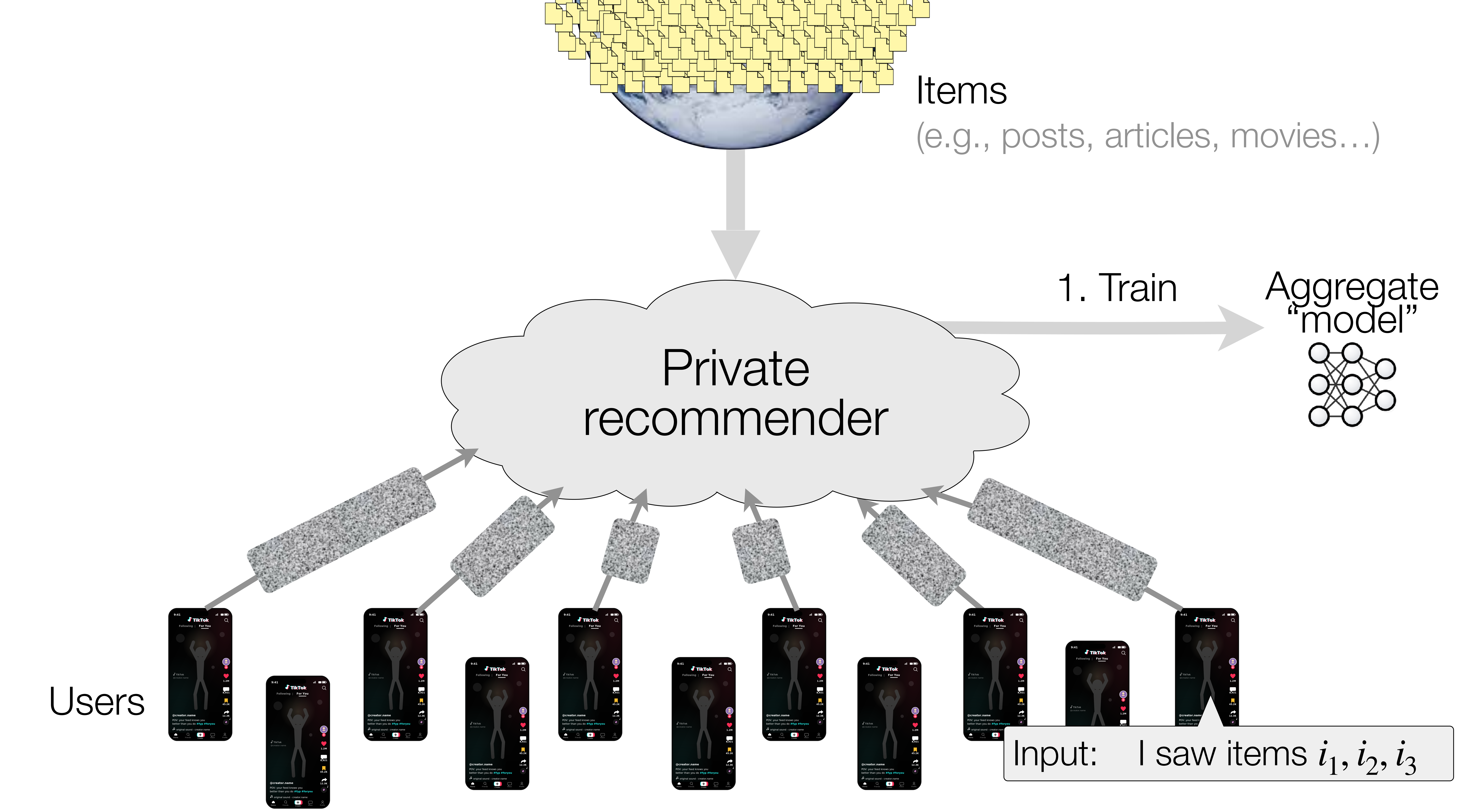
Items

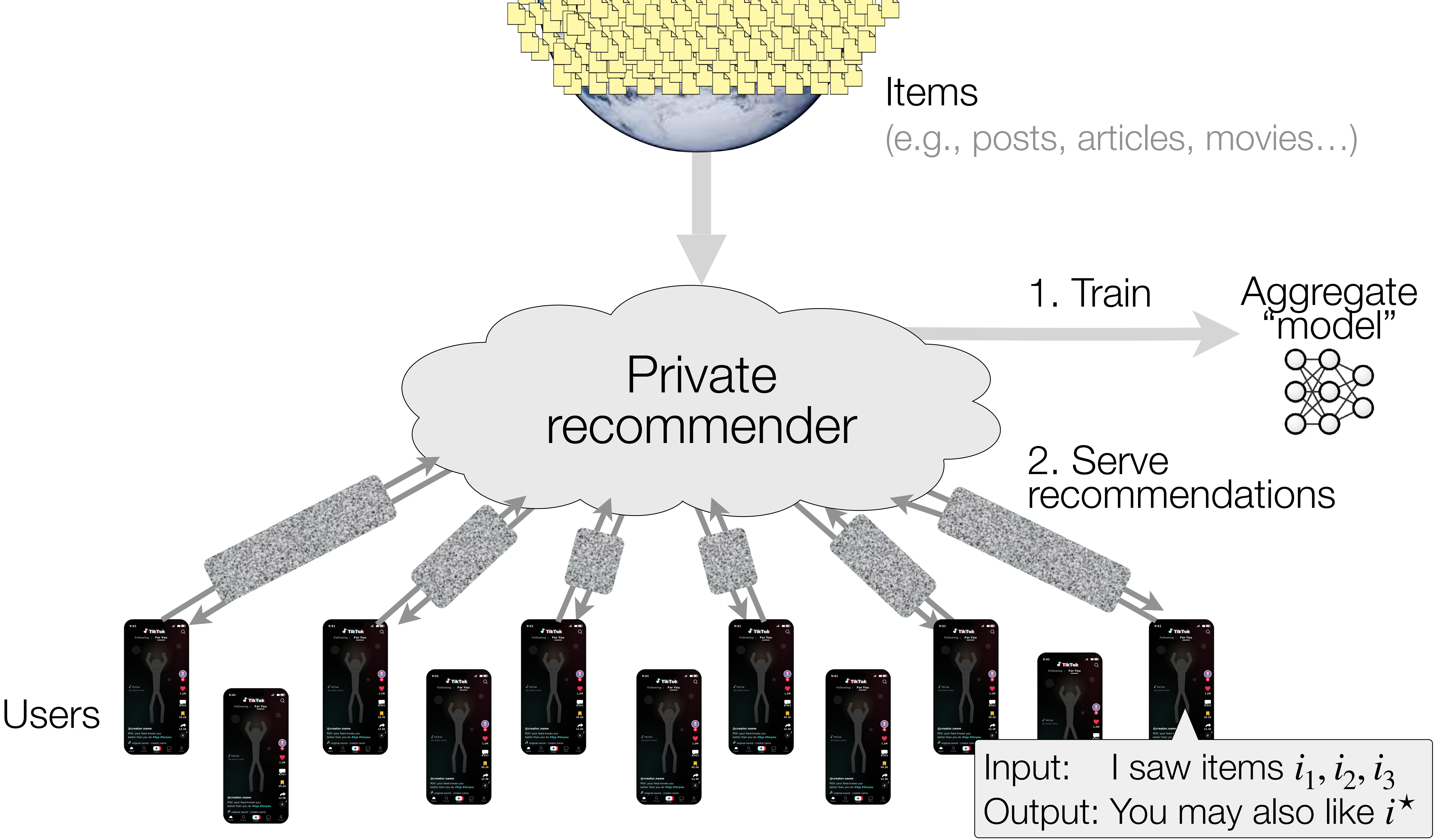
(e.g., posts, articles, movies…)

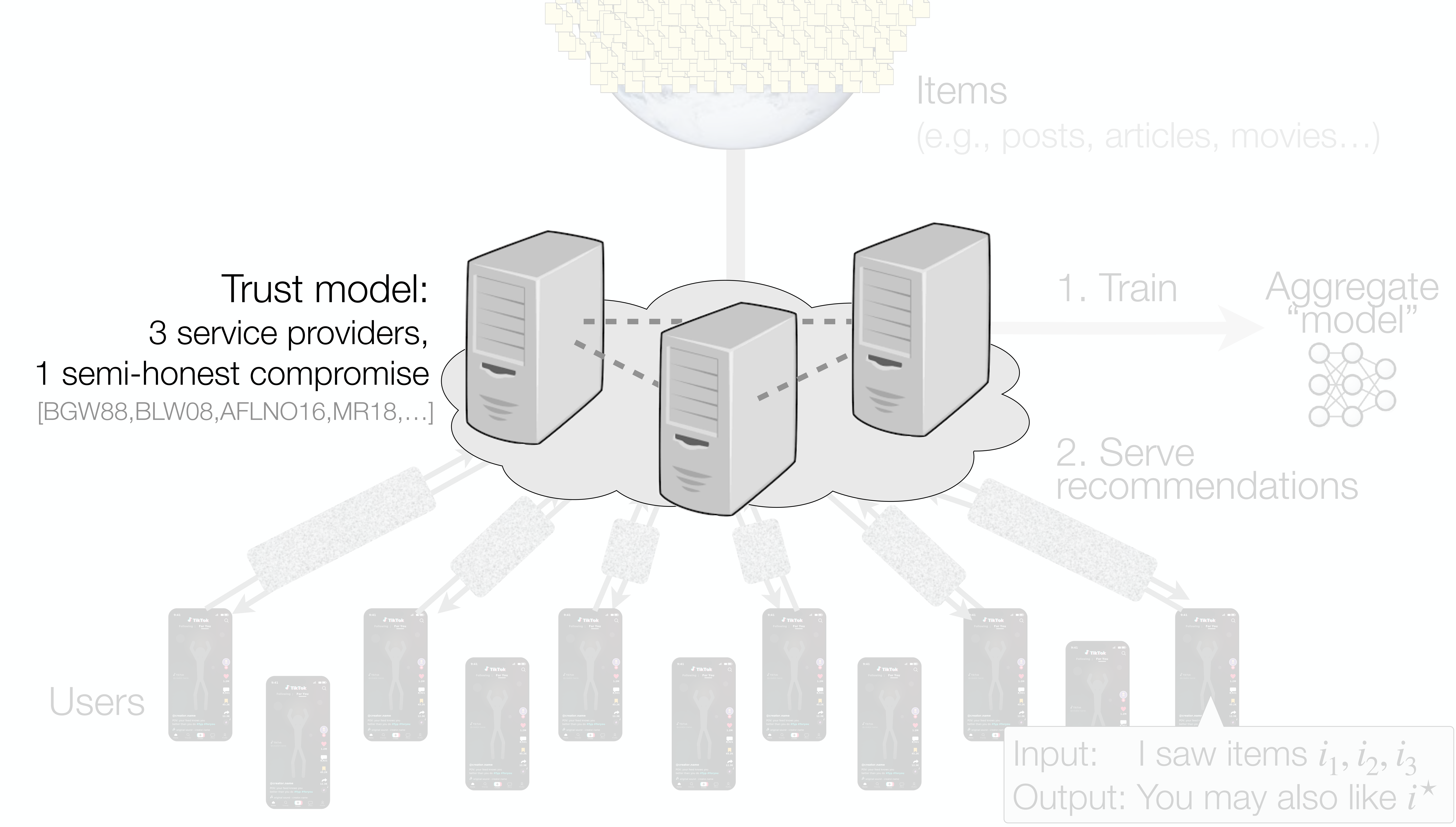
Private recommender

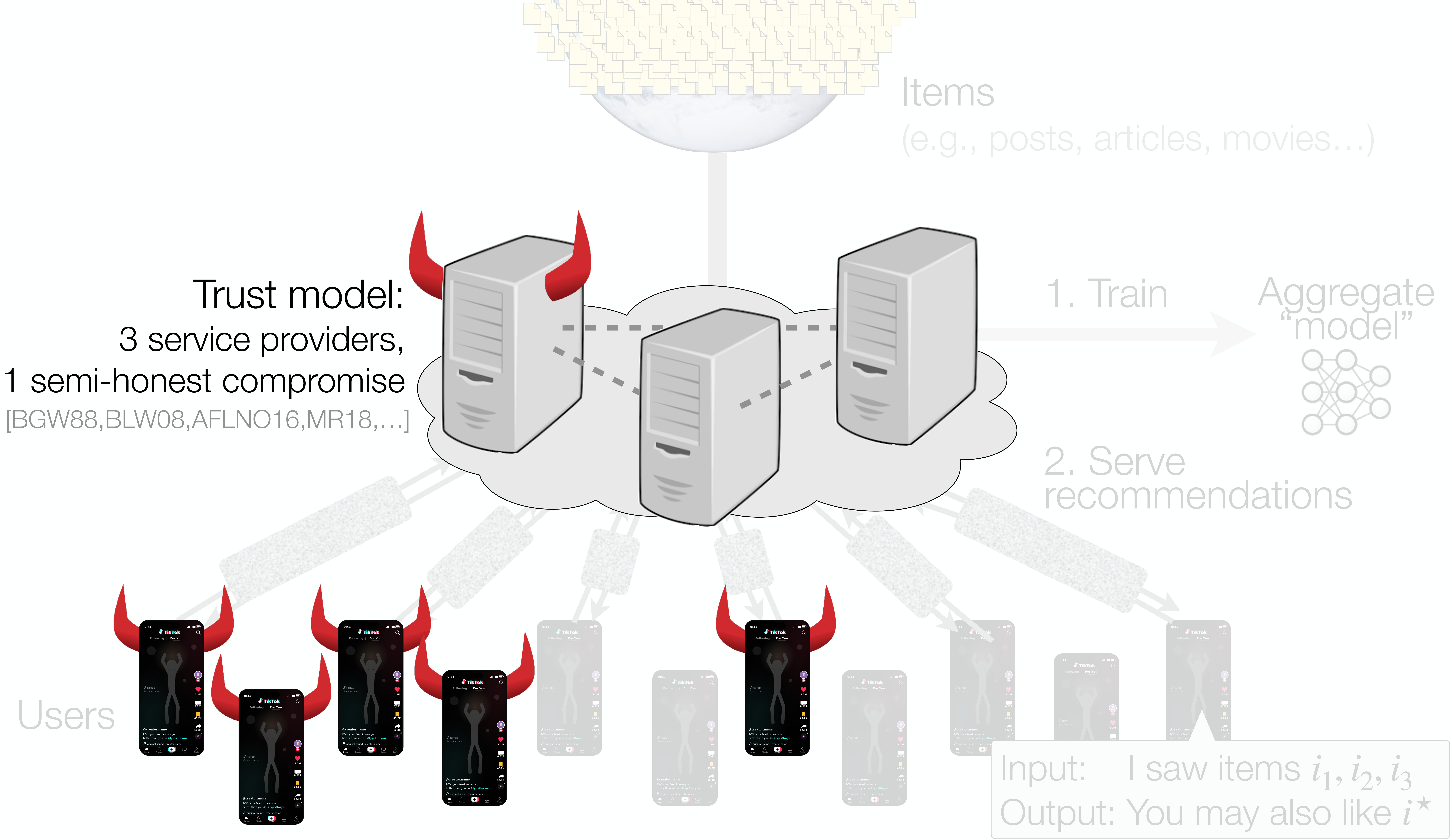
Users


Items
(e.g., posts, articles, movies…)
Private recommender
Users
Input: I saw items i_1, i_2, i_3


Items
(e.g., posts, articles, movies…)
Private recommender
1. Train
Aggregate “model”
Users
Input: I saw items i_1, i_2, i_3


Items
(e.g., posts, articles, movies…)
Private recommender
1. Train
Aggregate "model"
2. Serve recommendations
Users
Input: I saw items i_1, i_2, i_3
Output: You may also like $i^\star$


Items
(e.g., posts, articles, movies…)
Trust model:
3 service providers,
1 semi-honest compromise
[BGW88,BLW08,AFLNO16,MR18,…]
1. Train
Aggregate "model"
2. Serve recommendations
Users
Input: I saw items i_1, i_2, i_3
Output: You may also like $i^\star$


Items
(e.g., posts, articles, movies…)
Trust model:
3 service providers,
1 semi-honest compromise
[BGW88,BLW08,AFLNO16,MR18,…]
1. Train
Aggregate
"model"
2. Serve
recommendations
Users
Input: I saw items i_1, i_2, i_3
Output: You may also like $i^\star$

System goal: Privacy

This talk:

Cryptographic input privacy. No party — not users, not the servers — learns anything more about any user's input, beyond

(a) what the aggregate model reveals

(b) how many ratings each user submitted.

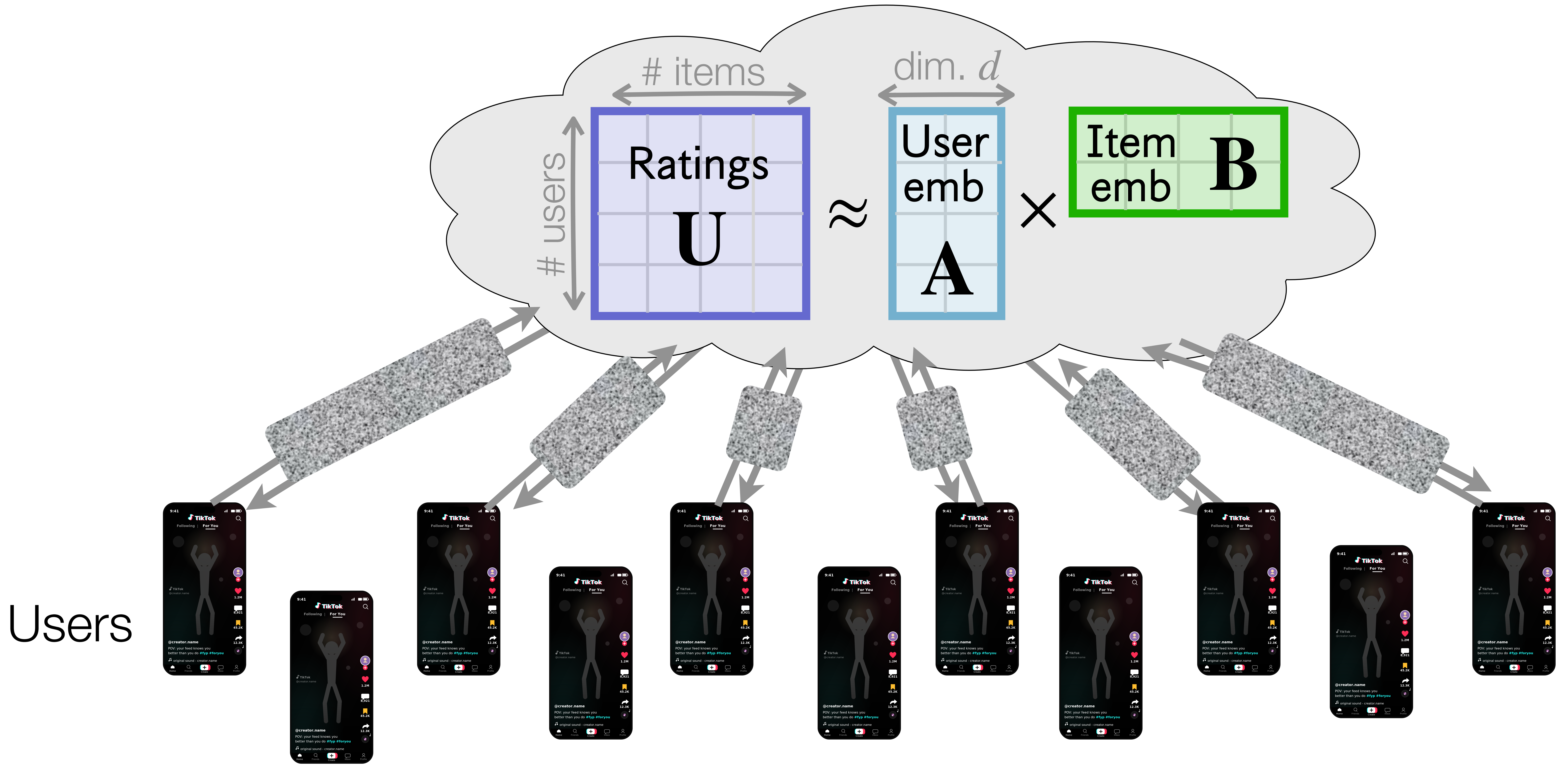
More goals (see paper):

Differential privacy of outputs (i.e., the model & ensuing recommendations).

Input validation to protect against malicious users.

System goal: Recommend via classic Matrix Factorization

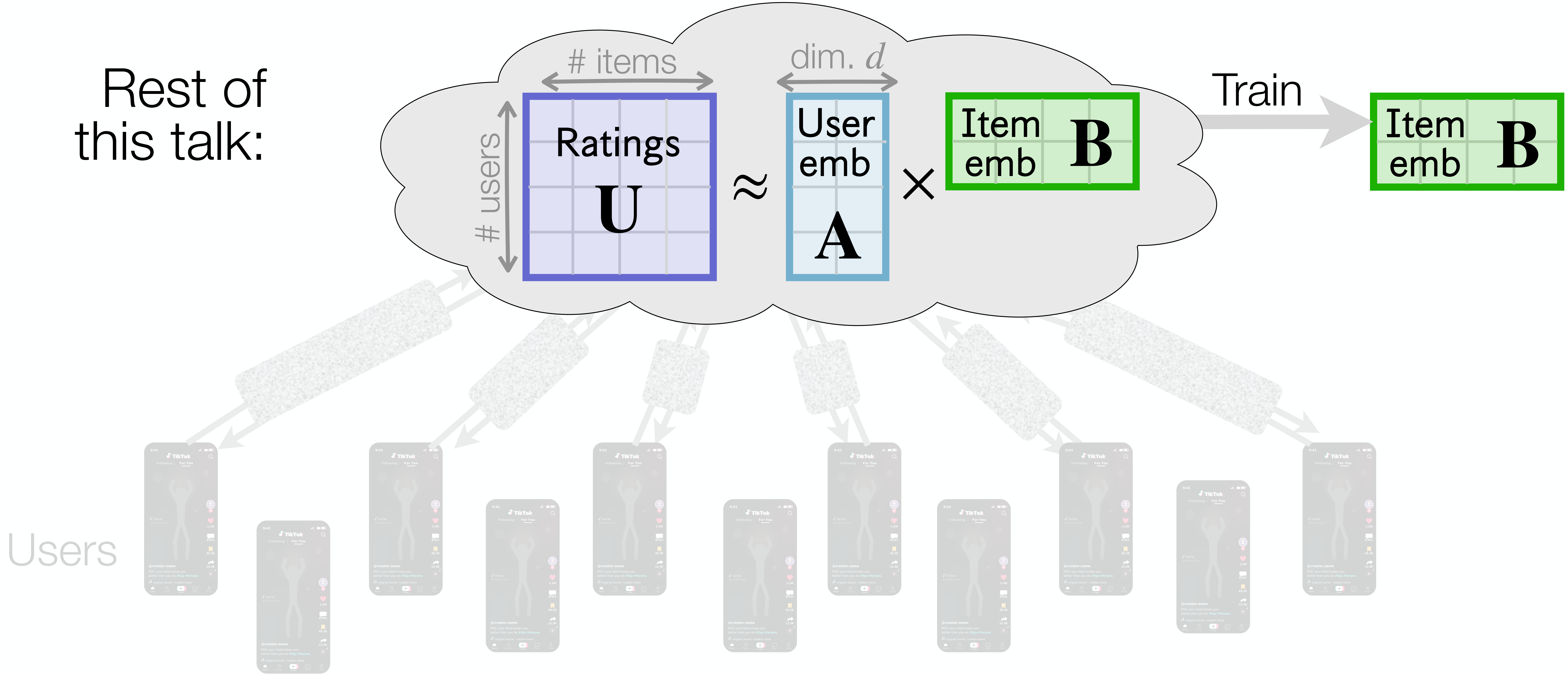
Has been used at Netflix, Yelp, Twitter, Spotify, and YouTube [KBV09]



System goal: Recommend via classic Matrix Factorization

Has been used at Netflix, Yelp, Twitter, Spotify, and YouTube [KBV09]

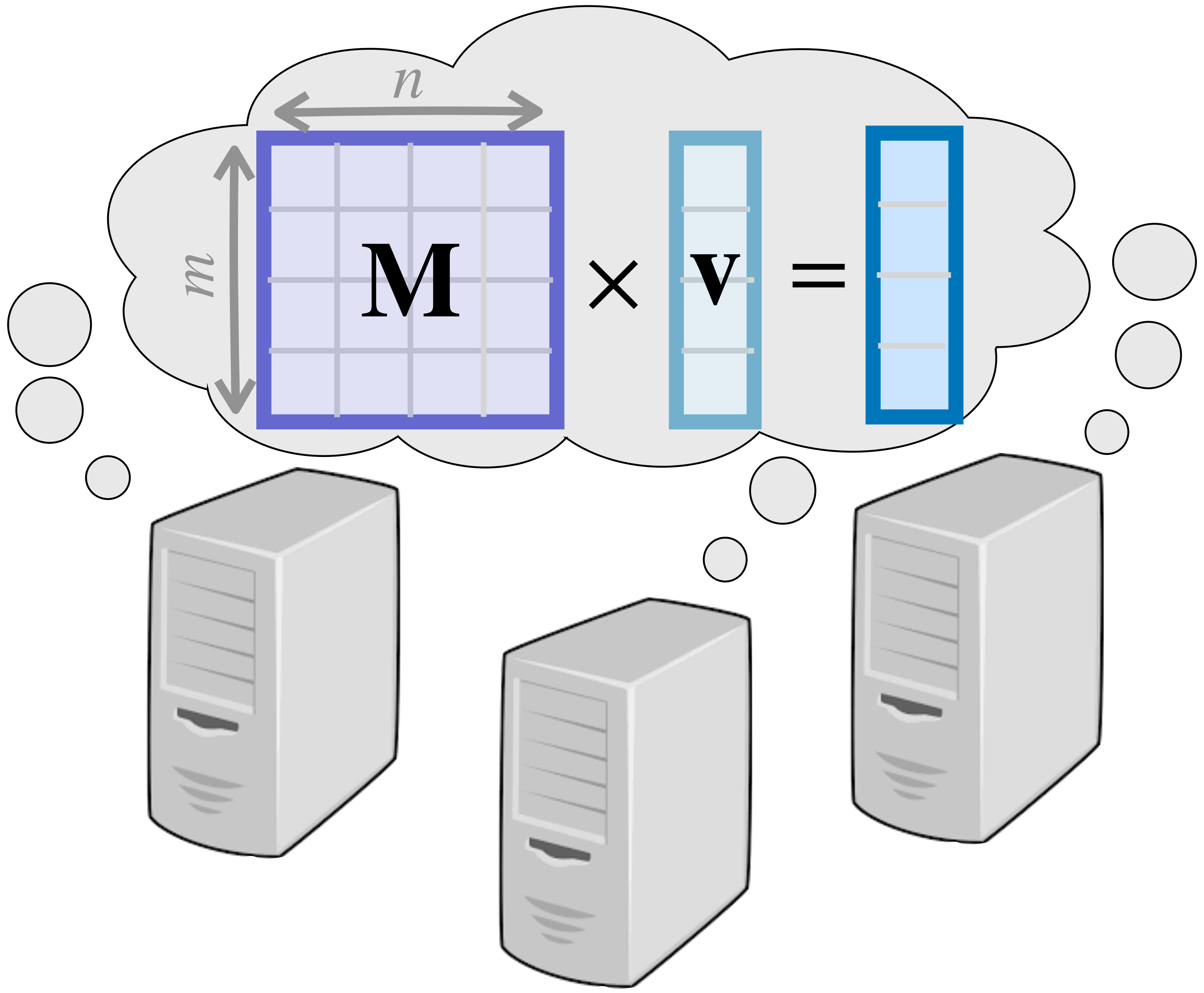
Rest of this talk:



Strategy: Lean on 3PC's unusually cheap operations

Replicated secret sharing

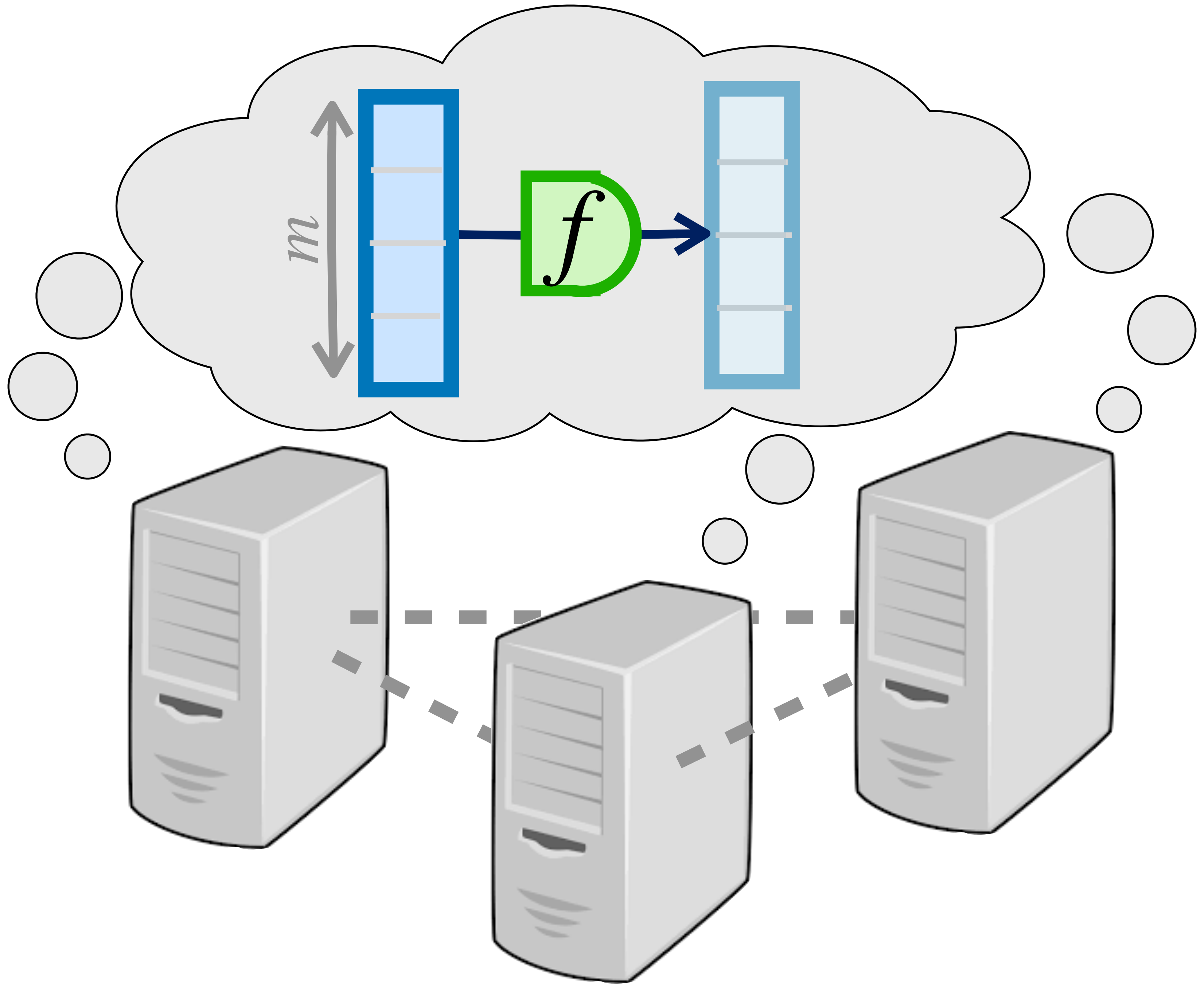
[CDI05,AFLNO16,MR18]



Can multiply secret-shared $\mathbf{M}$ by secret-shared $\mathbf{v}$ with no interaction ($3mn$ operations)

Function secret sharing

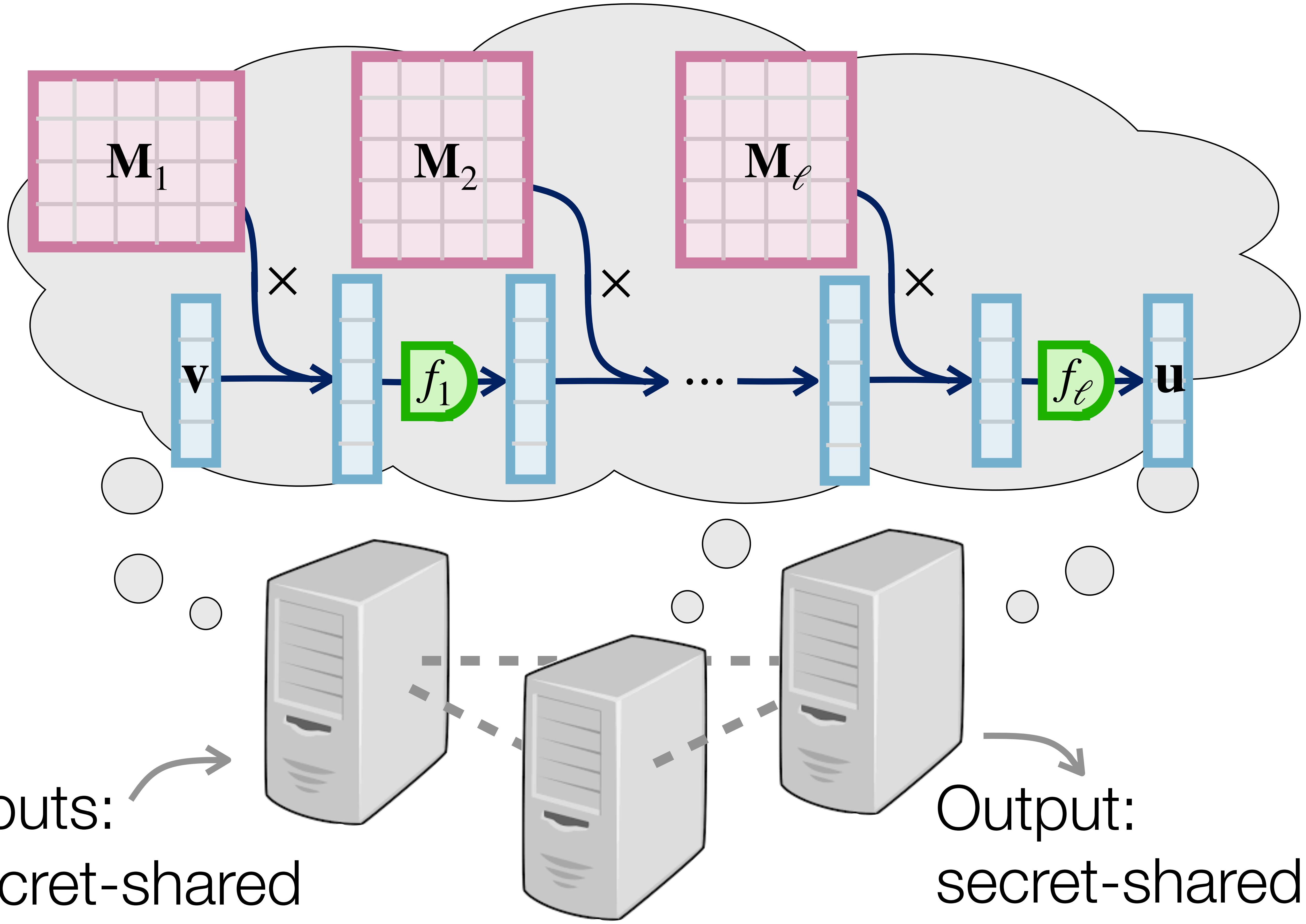
[BGI15,BGI16,BGI19,BCGGIKR21,Grotto23,Orca24,…]



Can evaluate certain non-linear functions on a secret-shared vector ($O(1)$ rounds, $O(m)$ comm.)

Strategy: Lean on 3PC's unusually cheap operations

Abstraction: "Matrix-vector programs"

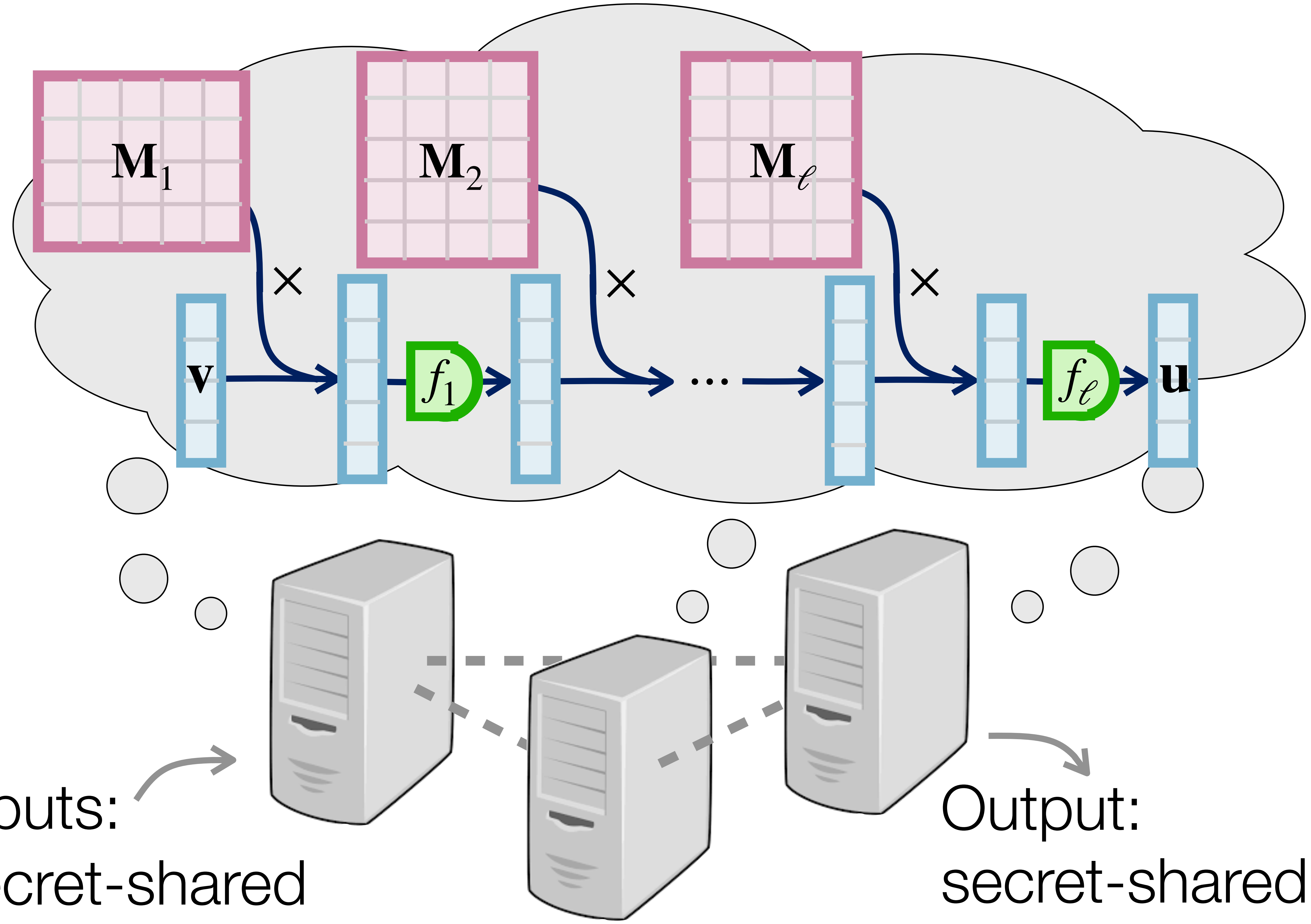


Inputs:
secret-shared
$\mathbf{v}, \mathbf{M_1}, \ldots, \mathbf{M}_\ell$

Output:
secret-shared $\mathbf{u}$

Strategy: Lean on 3PC's unusually cheap operations

Abstraction: "Matrix-vector programs"



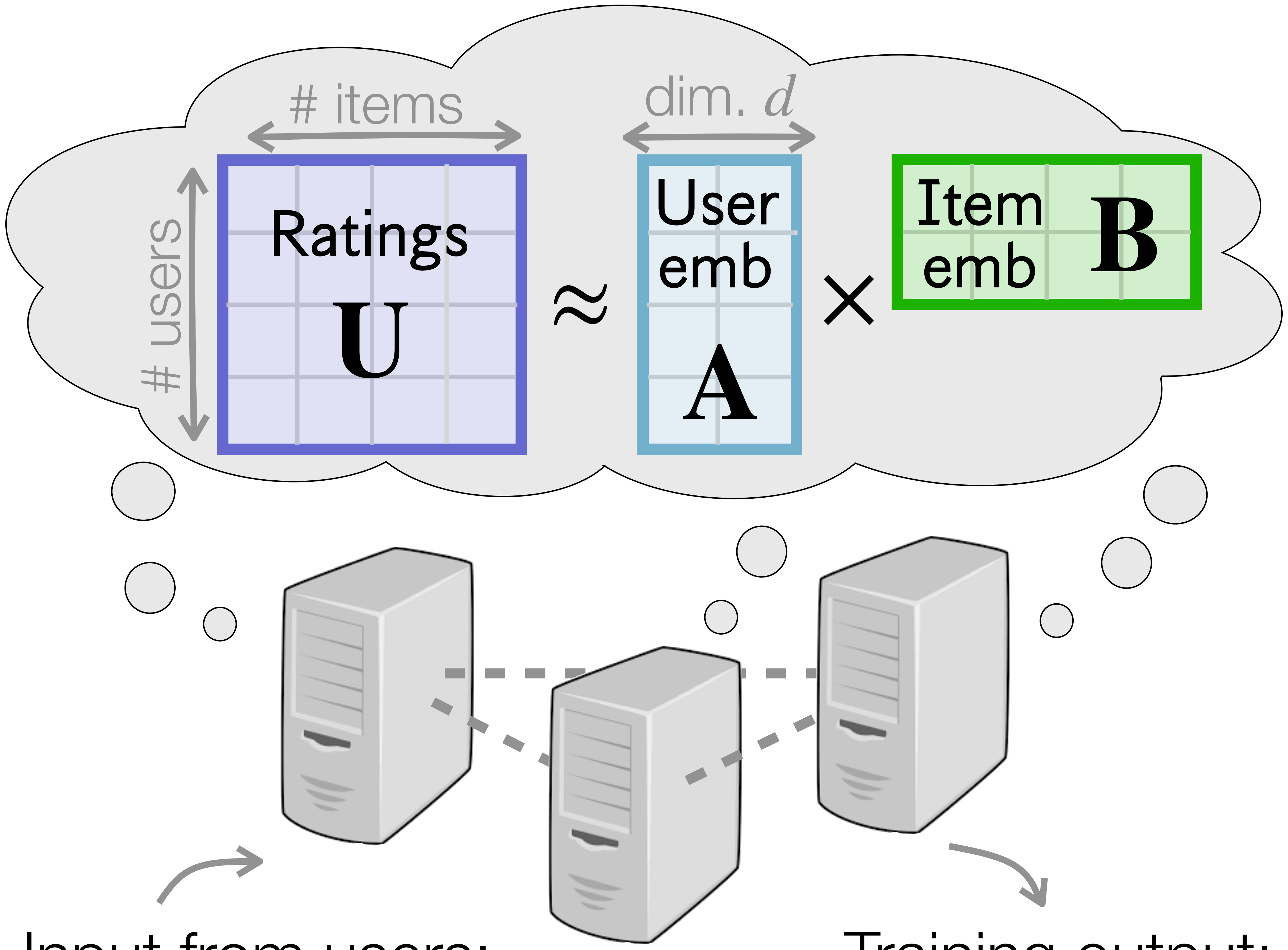
Inputs: secret-shared $\mathbf{v}, \mathbf{M}_1, \ldots, \mathbf{M}_\ell$

Output: secret-shared $\mathbf{u}$

Communication: sublinear in the input size (only ever reshare vectors)

Computation: 3x more operations on each party than if computed in cleartext (if matrices sufficiently large)

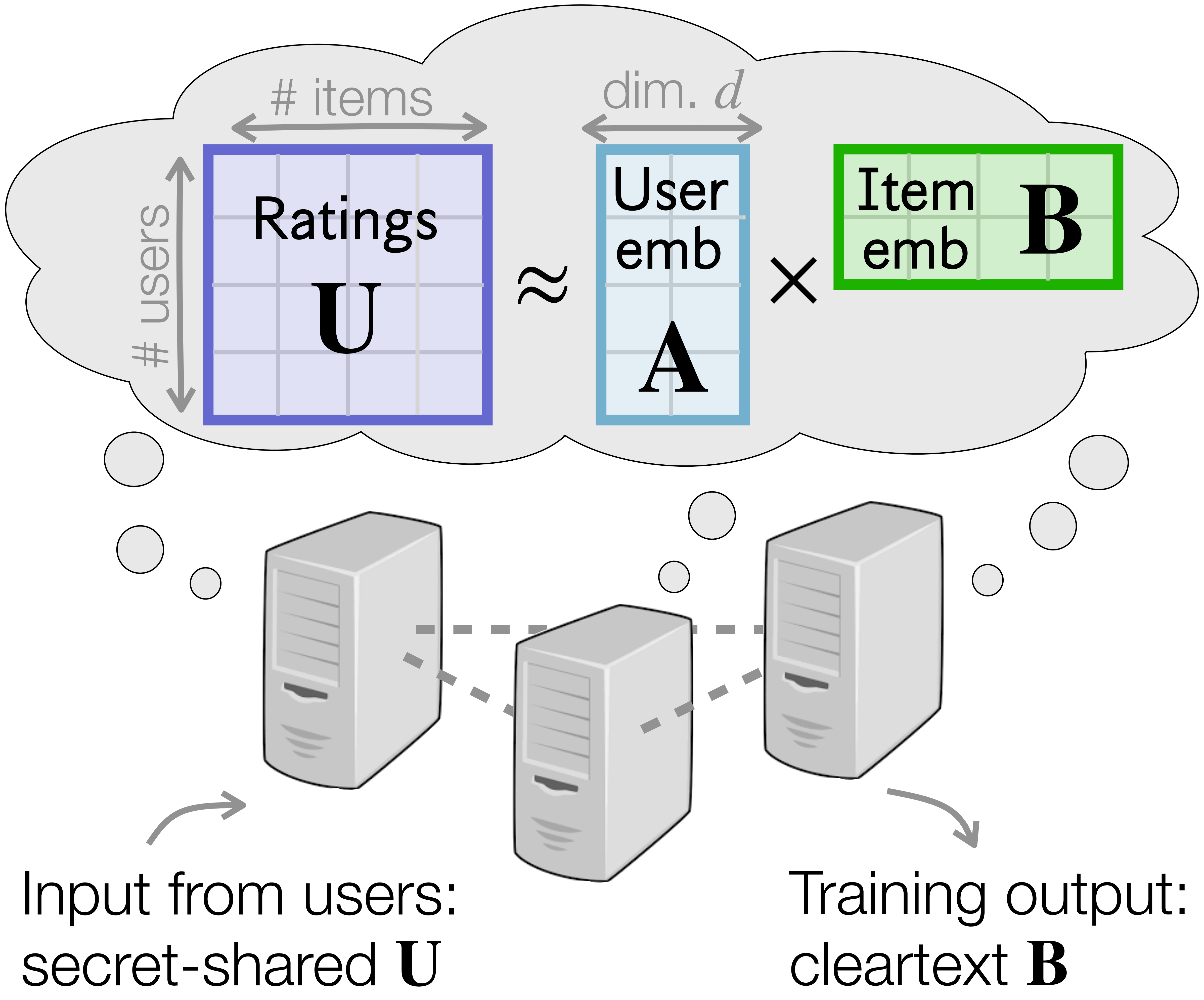
Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Input from users:
secret-shared $\mathbf{U}$

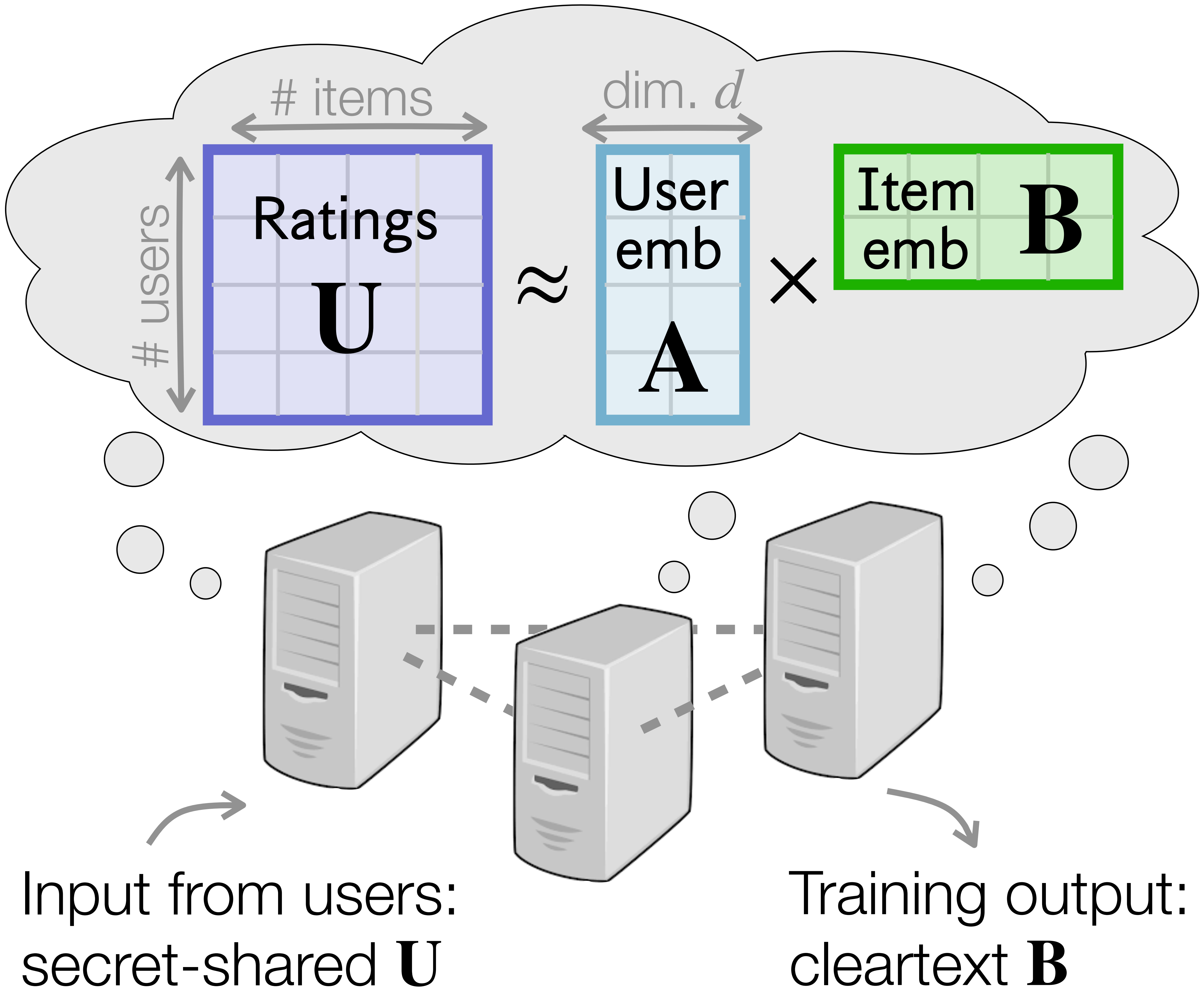
Training output:
cleartext $\mathbf{B}$

Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Fact: can set the rows of $\mathbf{B}$ to be the top eigenvectors of $\mathbf{U}^T\mathbf{U}$ (made orthonormal)

Strategy: Rewrite matrix factorization to fit the 3PC abstraction



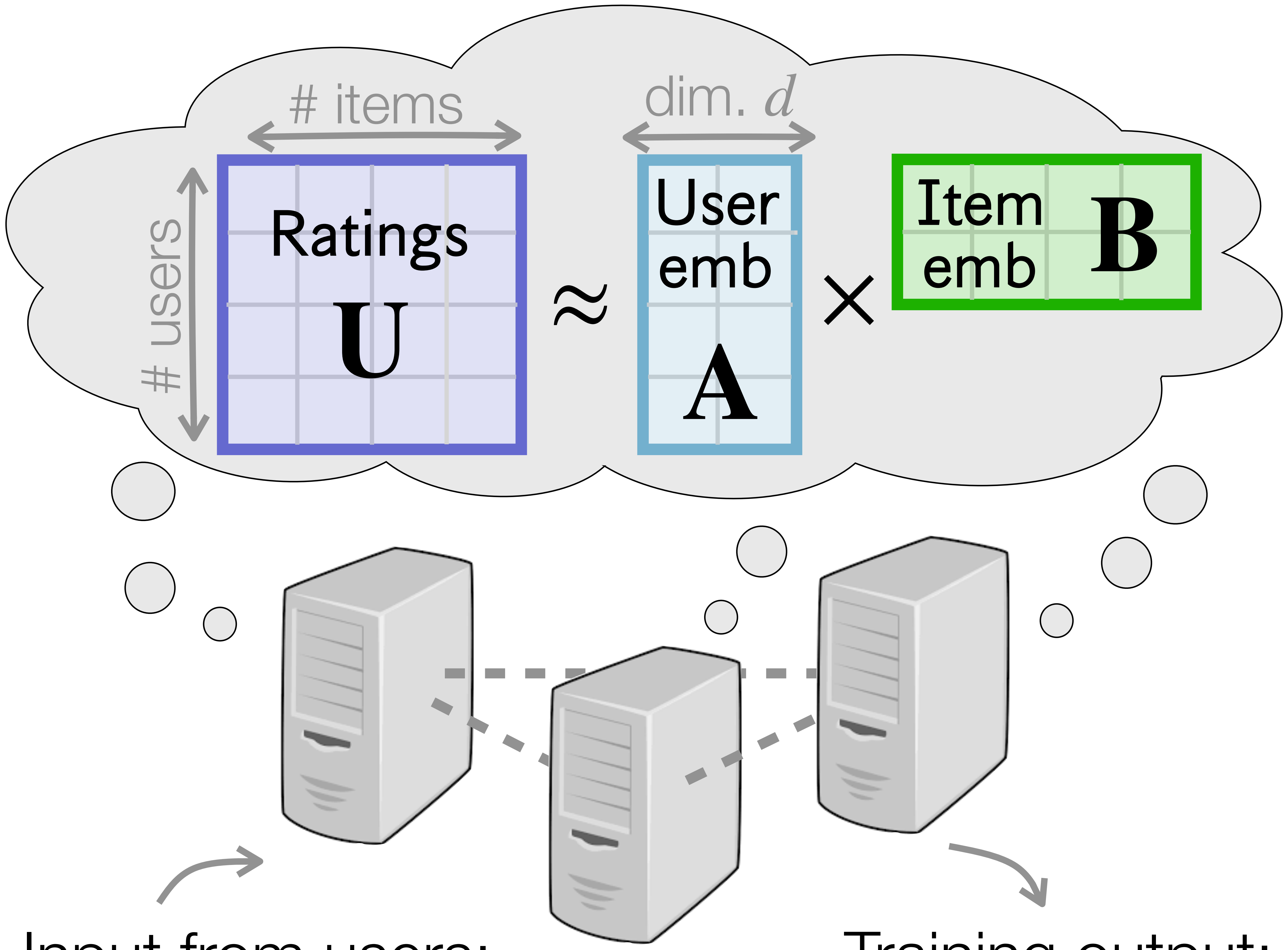
Fact: can set the rows of $\mathbf{B}$ to be the top eigenvectors of $\mathbf{U}^T\mathbf{U}$ (made orthonormal)

$\Downarrow$

Fact [Power iteration]: can compute each eigenvector as $(\mathbf{U}^T\mathbf{U})^{10} \cdot \mathbf{r}$, for a random vector $\mathbf{r}$ (orthogonal to those so far)

[MP1929]

Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Input from users: secret-shared $\mathbf{U}$

Training output: cleartext $\mathbf{B}$

Fact: can set the rows of $\mathbf{B}$ to be the top eigenvectors of $\mathbf{U}^T\mathbf{U}$ (made orthonormal)

$\Downarrow$

Fact [Power iteration]: can compute each eigenvector as $(\mathbf{U}^T\mathbf{U})^{10} \cdot \mathbf{r}$, for a random vector $\mathbf{r}$ (orthogonal to those so far)

[MP1929]

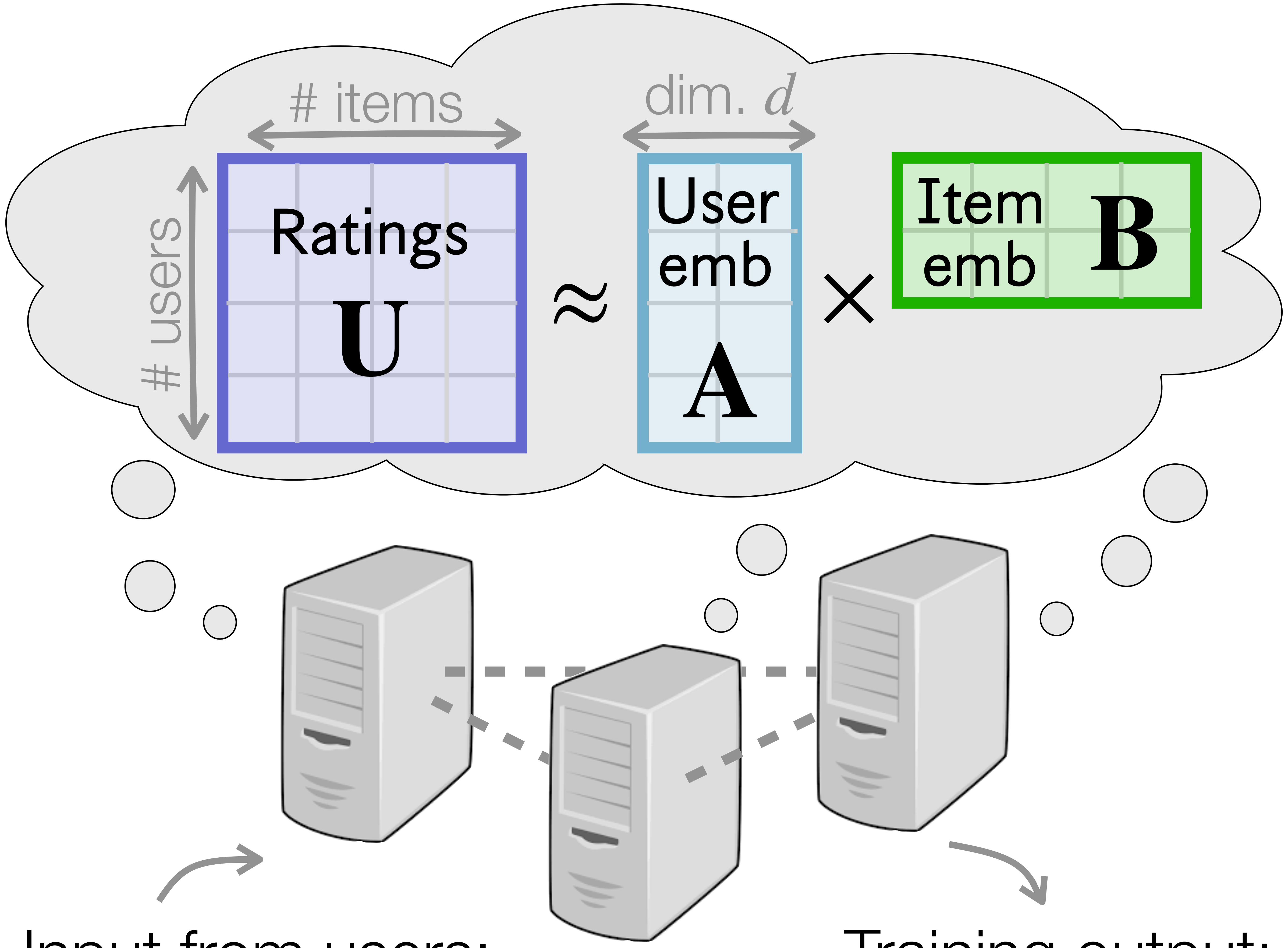
$\Downarrow$

Instead: compute each eigenvector as

$$\mathbf{U}^T \cdot \ldots \cdot \mathbf{U} \cdot (\mathbf{U}^T \cdot (\mathbf{U} \cdot \mathbf{r}))$$

(with appropriate floating-point arithmetic)

Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Input from users:
secret-shared $\mathbf{U}$

Training output:
cleartext $\mathbf{B}$

Fact: can set the rows of $\mathbf{B}$ to be the top eigenvectors of $\mathbf{U}^T\mathbf{U}$ (made orthonormal)

$\Downarrow$

Fact [Power iteration]: can compute each eigenvector as $(\mathbf{U}^T\mathbf{U})^{10} \cdot \mathbf{r}$, for a random vector $\mathbf{r}$ (orthogonal to those so far)

[MP1929]

$\Downarrow$

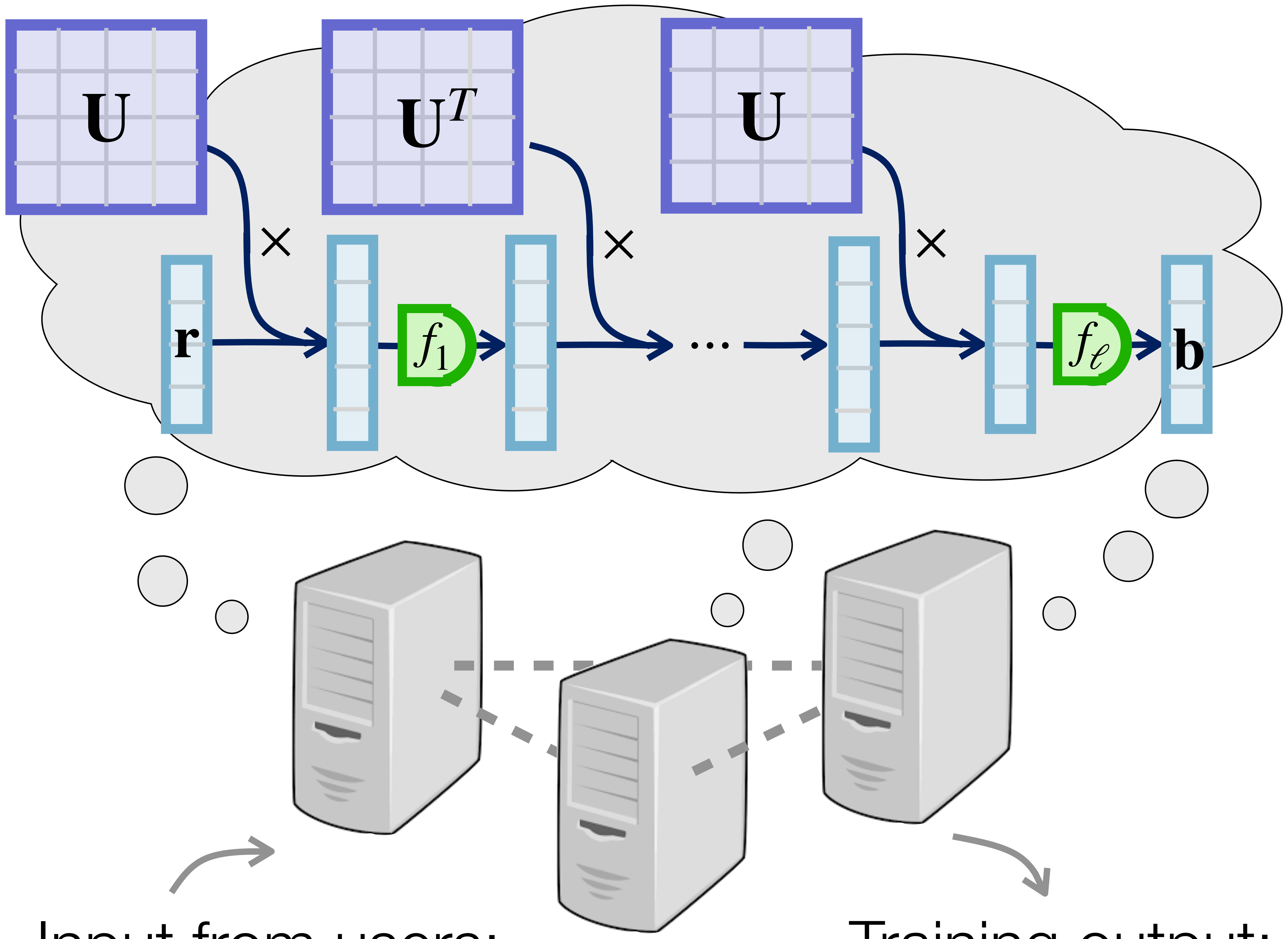
Instead: compute each eigenvector as

$$\mathbf{U}^T \cdot \ldots \cdot \mathbf{U} \cdot (\mathbf{U}^T \cdot (\mathbf{U} \cdot \mathbf{r}))$$

(with appropriate floating-point arithmetic)

Matrix-vector program!

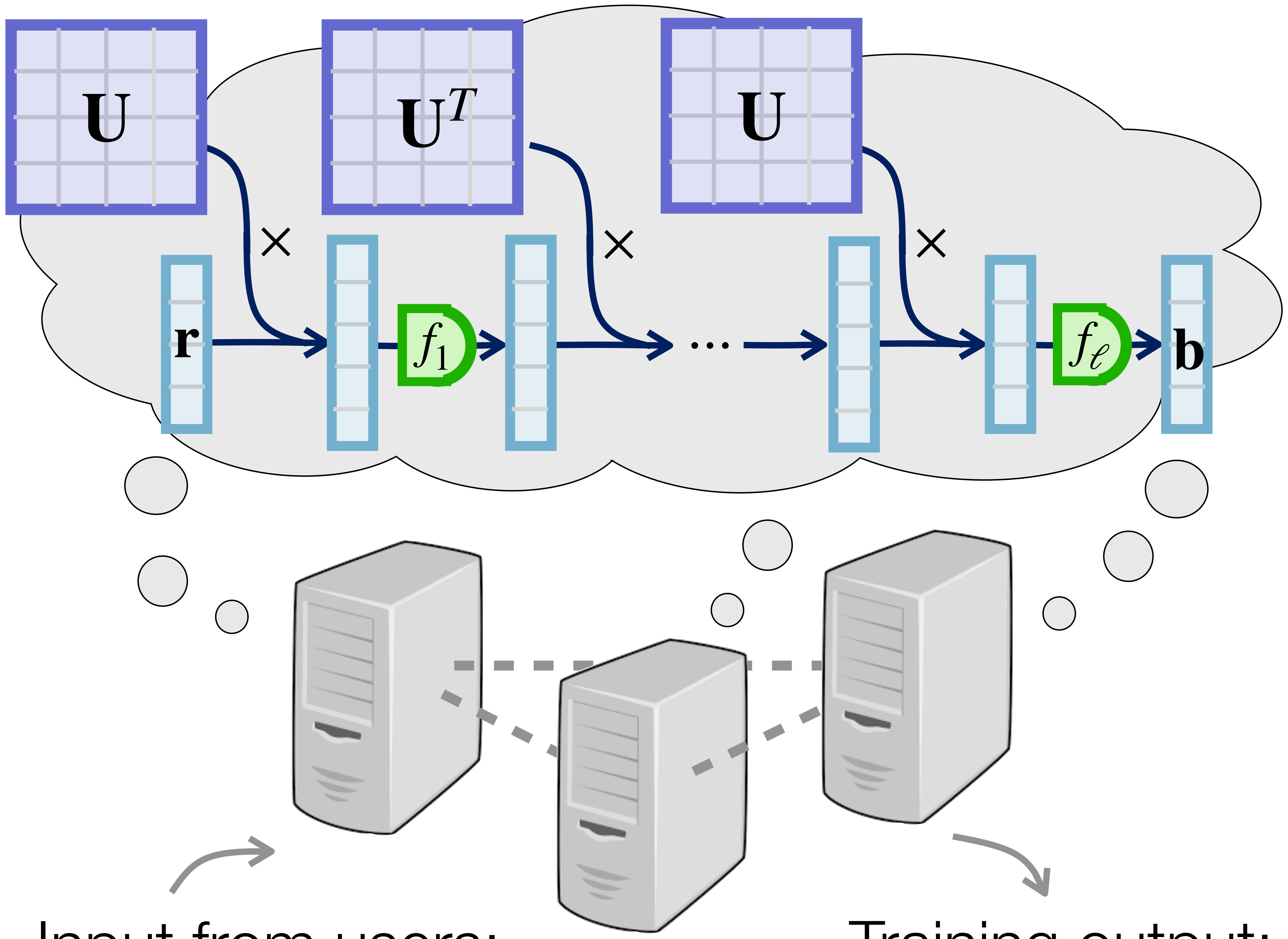
Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Input from users:
secret-shared $\mathbf{U}$

Training output:
cleartext $\mathbf{B}$

Strategy: Rewrite matrix factorization to fit the 3PC abstraction



Input from users:
secret-shared $\mathbf{U}$

Training output:
cleartext $\mathbf{B}$

In paper: new 3PC protocols for vector truncation & normalization (e.g., shrinking communication by 6x)

Total communication:
$O(d \cdot \text{users} + d \cdot \text{items})$ bits

Per-party computation:
$\approx 60 \cdot d \cdot \text{users} \cdot \text{items}$ operations

Nudge Implementation

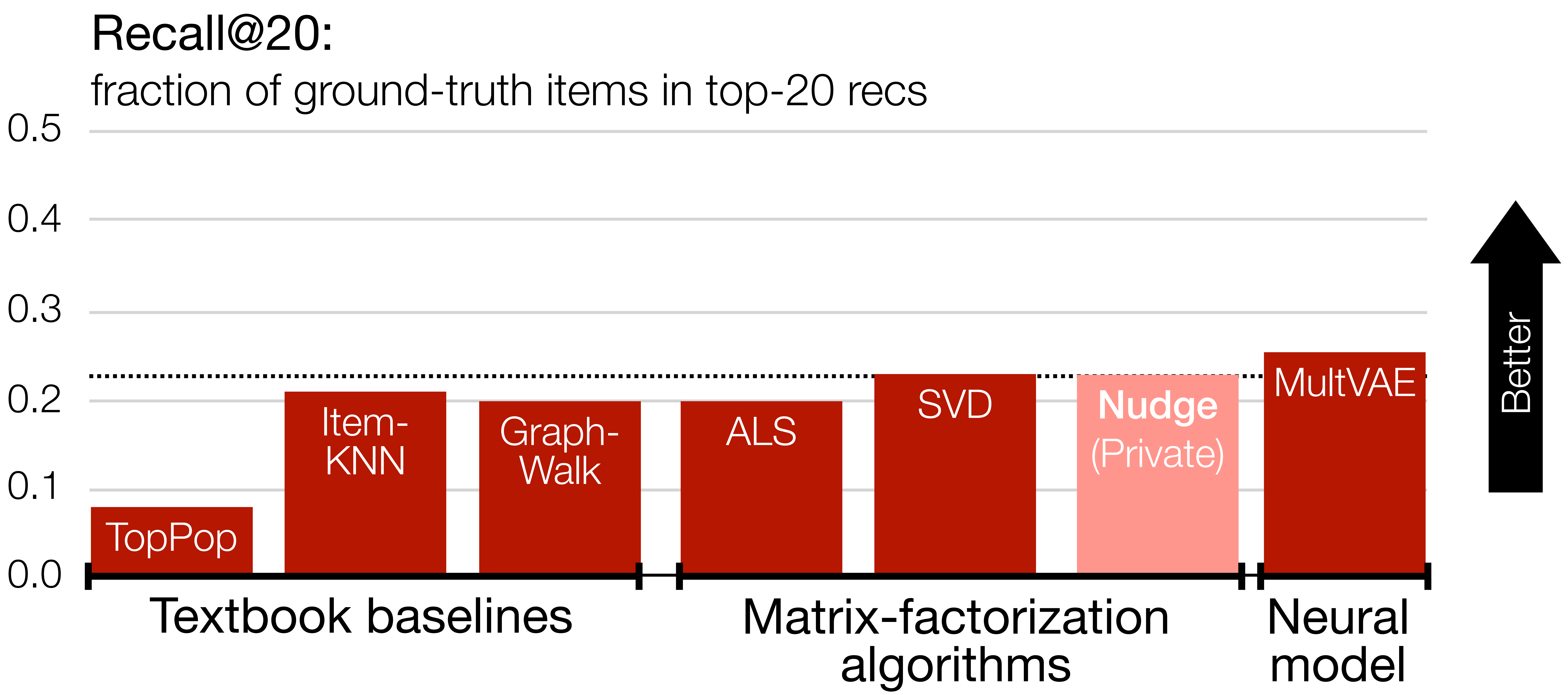
Open-source code at `github.com/NudgeArtifact/private-recs`

- 7,200 lines of Go
- 2,500 lines of Go for function-secret-sharing tools (truncation, normalization)
- For speed: C for matrix-multiplication core, SIMD, optimized memory locality

Experimental setup

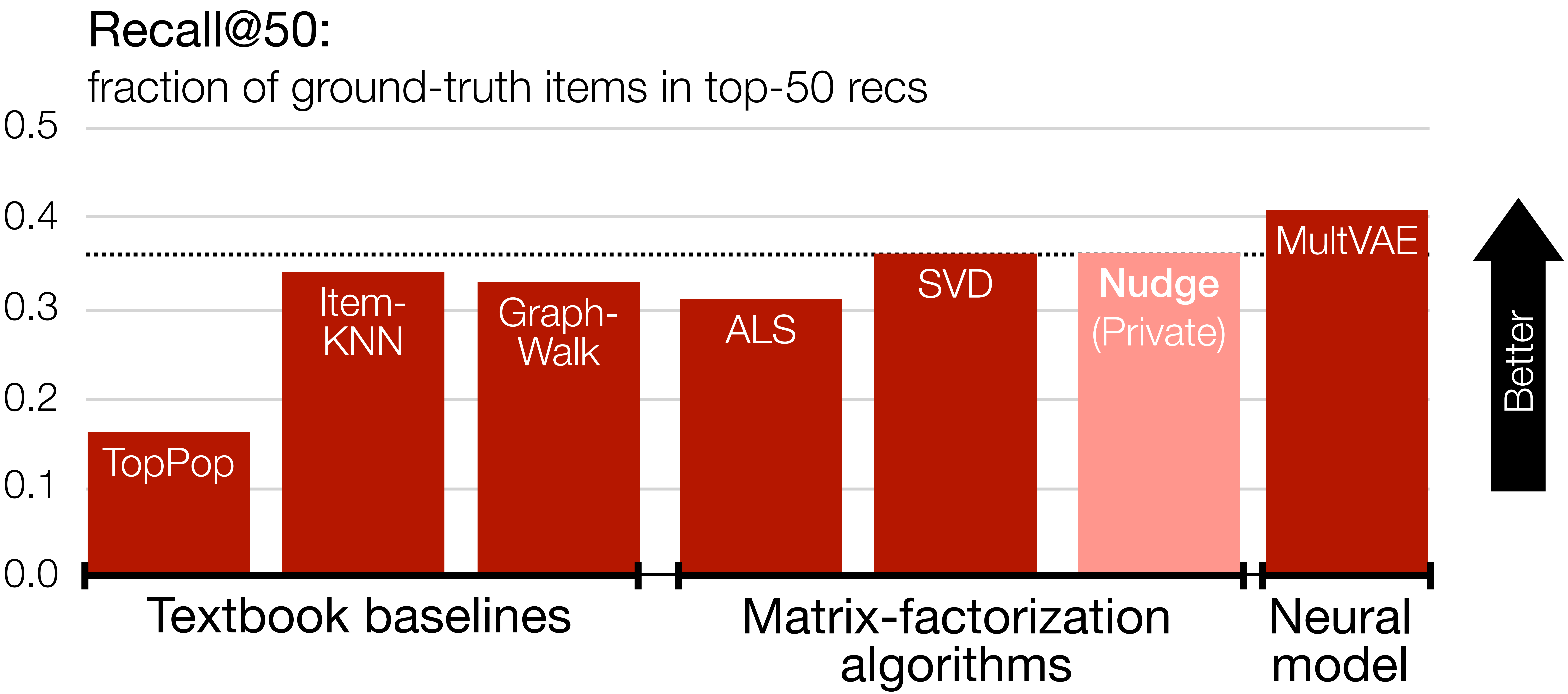
- Each server runs on a 192-core machines (1.5TB of RAM)
- Connected via a local-area network (50 GBps link)
- Factor matrices as large as fit in RAM

Nudge matches the quality of cleartext matrix factorization



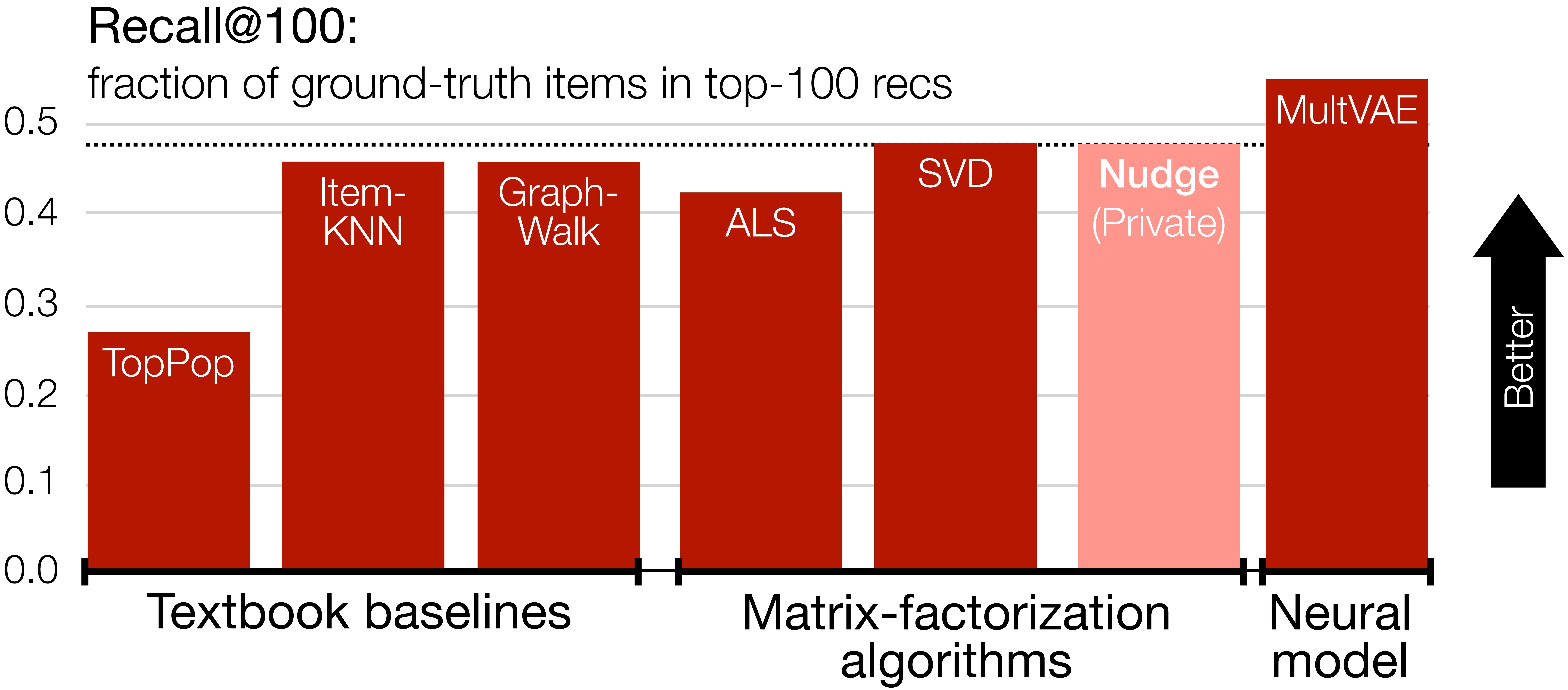
Dataset: Netflix prize (400k users, 18k items, 56 million ratings)

Nudge matches the quality of cleartext matrix factorization



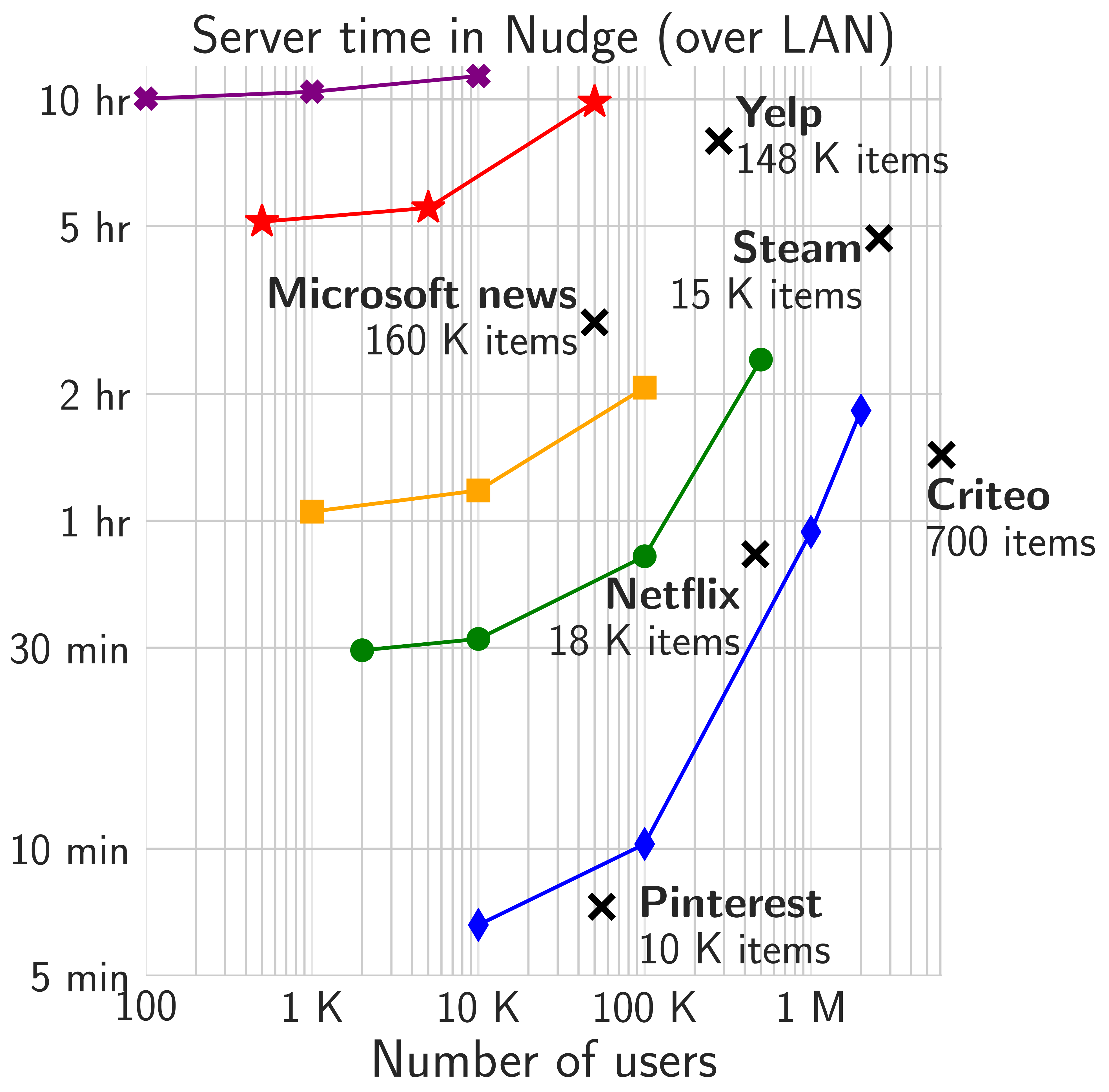
Dataset: Netflix prize (400k users, 18k items, 56 million ratings)

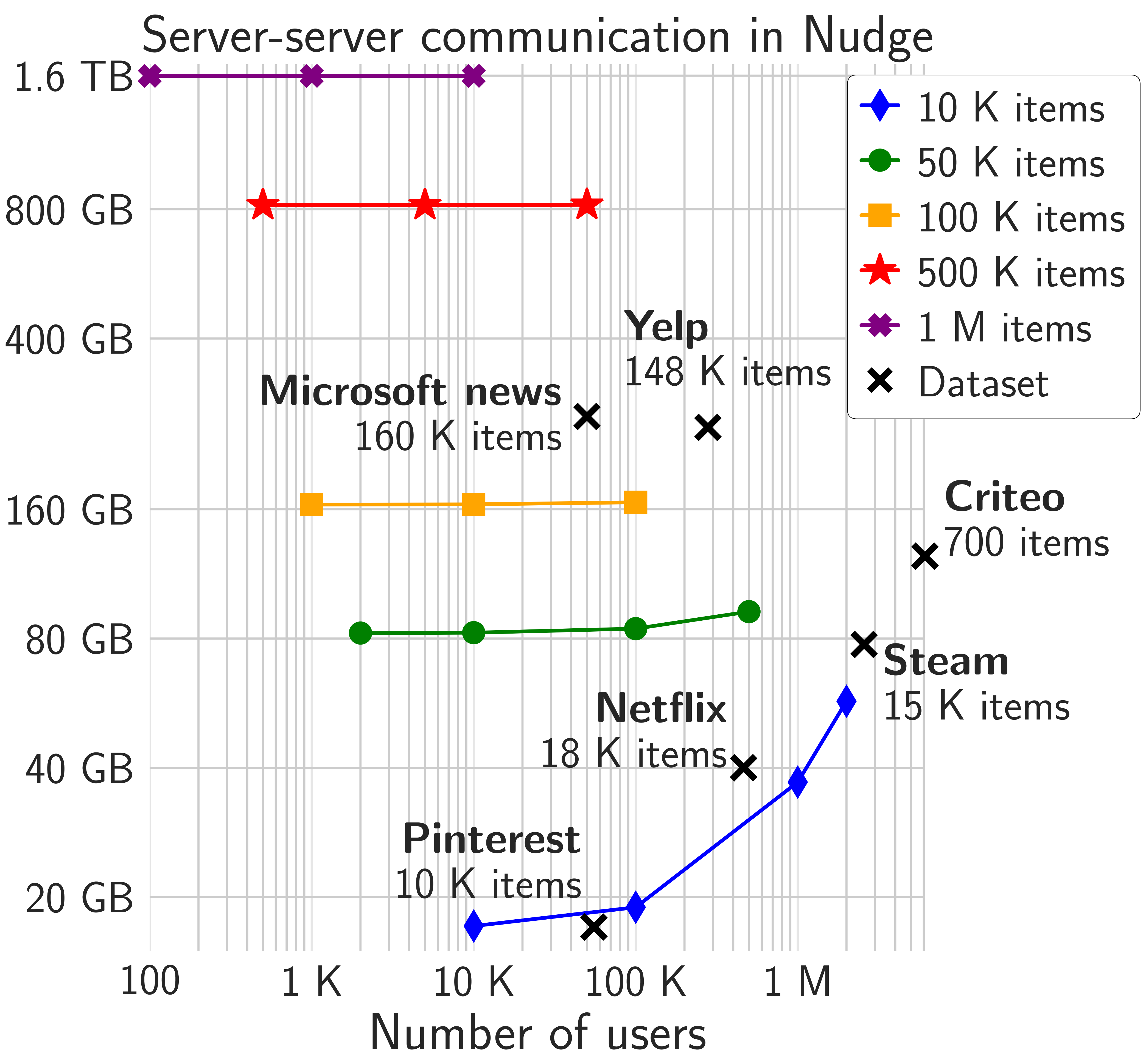
Nudge matches the quality of cleartext matrix factorization



Dataset: Netflix prize (400k users, 18k items, 56 million ratings)

Nudge factors matrices with 10 billion entries in hours





Largest dataset used by prior private recommender: 6k users / 4k items, taking 78 TB [GraphSC, a 2PC]

Matrix approximation can be fast & private, and it’s everywhere

- [x] - This talk: Private recommendations via matrix factorization
- [ ] - Compressing secret-shared matrices?
- [ ] - Clustering secret-shared graphs?
- [ ] - Learning low-dimensional embeddings on secret-shared data?
- [ ] - PageRank on secret-shared networks?

… what’s next?

Matrix approximation can be fast & private,
and it's everywhere

- ☑ - This talk: Private recommendations via matrix factorization
- ☐ - Compressing secret-shared matrices?
- ☐ - Clustering secret-shared graphs?
- ☐ - Learning low-dimensional embeddings on secret-shared data?
- ☐ - PageRank on secret-shared networks?

… what's next?

`github.com/NudgeArtifact/private-recs`
`eprint.iacr.org/2026/179`