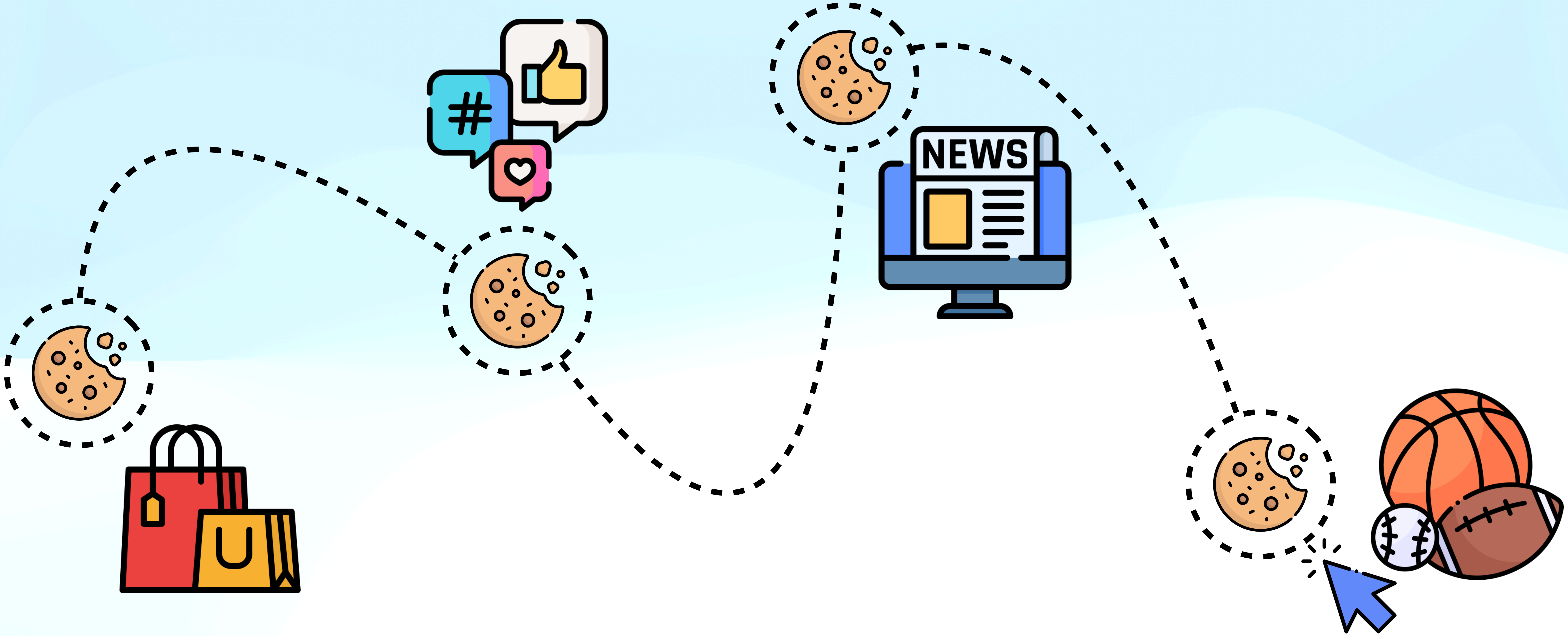
FP-Fed: Privacy-Preserving Federated Detection of Browser Fingerprinting

Meenatchi Sundaram Muthu Selva Annamalai (University College London)
Igor Bilogrevic (Google)
Emiliano De Cristofaro (University of California, Riverside)

What is Browser Fingerprinting?

Tracking on the Web

Third-party cookies



Tracking on the Web

Browsers block/restrict third-party cookies

Source: https://developer.mozilla.org/en-US/blog/goodbye-third-party-cookies/

What is Browser Fingerprinting?

Definition

- Collects set of information related to a user’s device that can be **uniquely identifiable**
    - Hardware (# CPU cores, screen size, etc.)
    - Software (Browser extensions installed, Version of flash installed, etc.)

What is Browser Fingerprinting?

Definition

- Collects set of information related to a user’s device that can be **uniquely identifiable**
  - Hardware (# CPU cores, screen size, etc.)
  - Software (Browser extensions installed, Version of flash installed, etc.)
- Deployed via **JS script** that runs in browser (e.g. `fingerprintjs`)

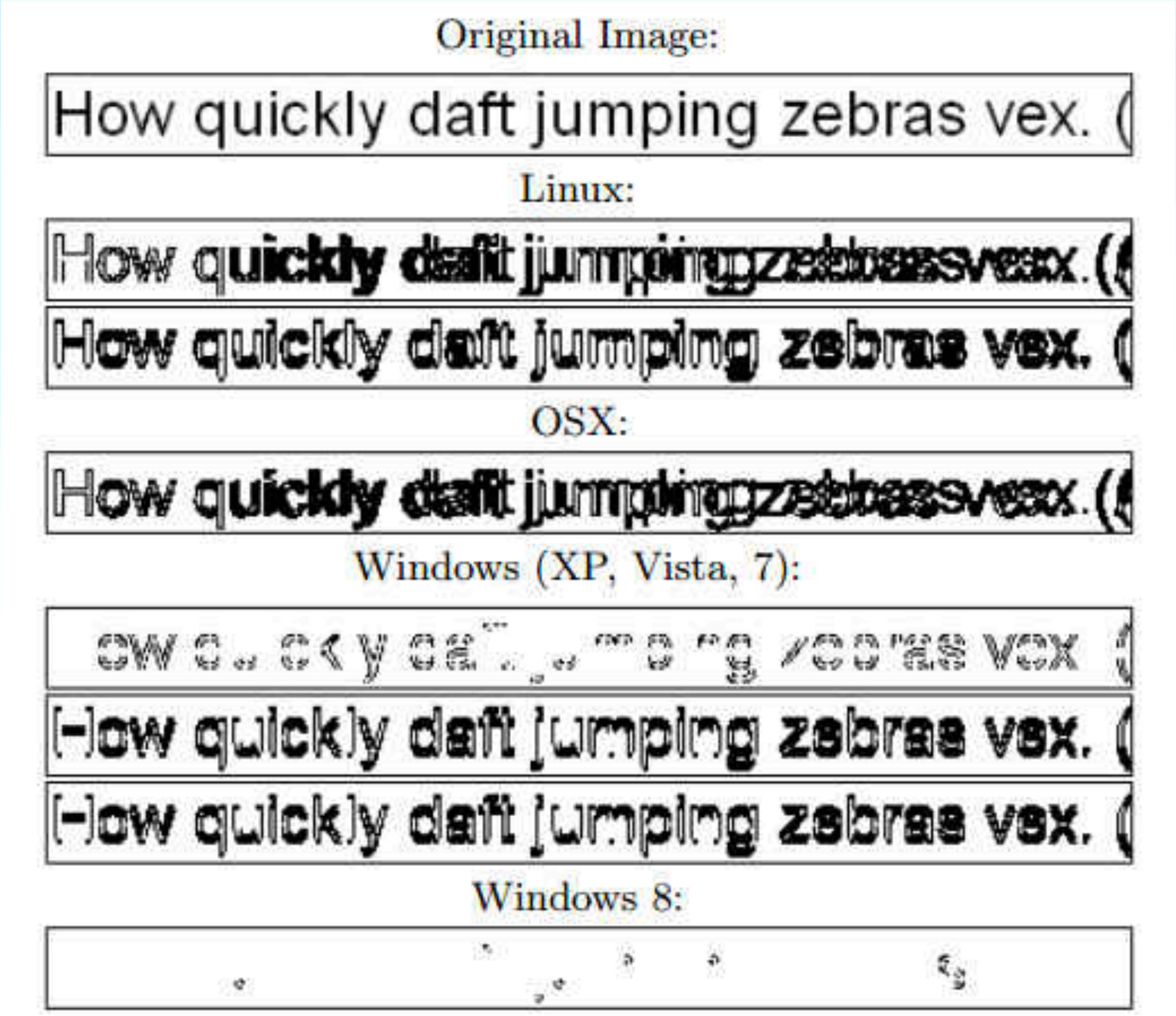
What is Browser Fingerprinting?

Definition

- Collects set of information related to a user's device that can be **uniquely identifiable**
    - Hardware (# CPU cores, screen size, etc.)
    - Software (Browser extensions installed, Version of flash installed, etc.)
- Deployed via **JS script** that runs in browser (e.g. `fingerprintjs`)
- **Invasive** tracking technique
    - Stateless: no information stored in browser (e.g., cookies)
    - Hard to prevent and less transparent

What is Browser Fingerprinting?

Example: Canvas Fingerprinting



Source: https://www.i-programmer.info/news/149-security/7583-the-canvas-fingerprint-how.html

Exploits differences in how different devices render images

Challenges of Browser Fingerprinting Detection

Prior Work

Browser Fingerprinting Detection

Manually curated blocklists / heuristics

- Hard to maintain
- Narrowly defined

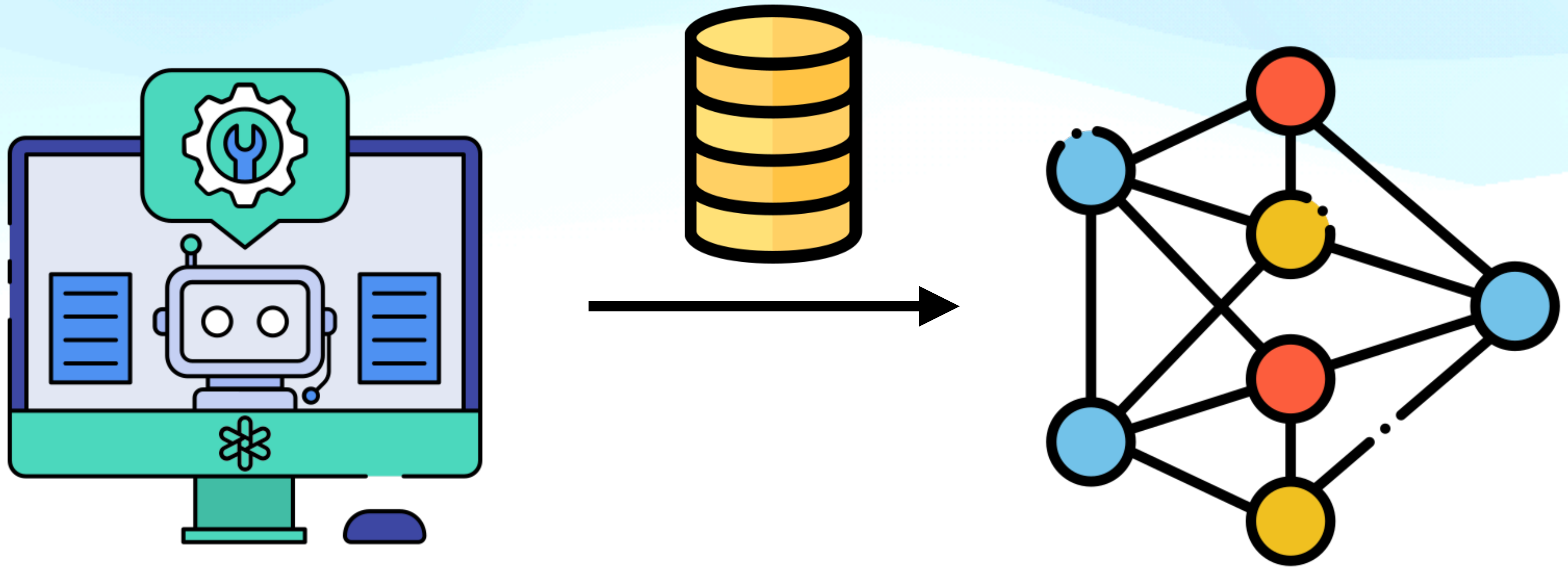
Prior Work

Browser Fingerprinting Detection

<u>Manually curated blocklists / heuristics</u>

- Hard to maintain
- Narrowly defined

<u>Centralized Machine Learning + Automated Web Crawl</u>



- Cannot replicate real human interactions
- Blocked by bot detectors, CAPTCHAs, login pages, and paywalls
- Large number of features (~2000)

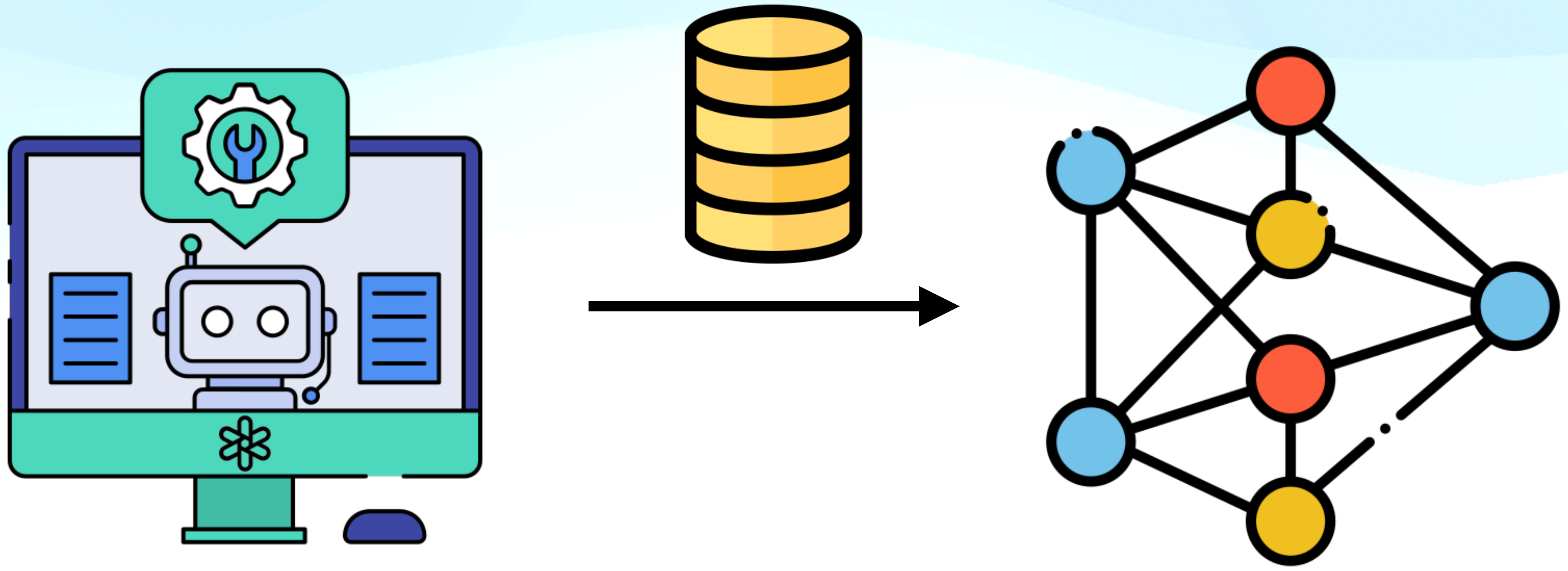
Prior Work

Browser Fingerprinting Detection

Manually curated blocklists / heuristics

- Hard to maintain
- Narrowly defined

Centralized Machine Learning + Automated Web Crawl



- Cannot replicate real human interactions
- Blocked by bot detectors, CAPTCHAs, login pages, and paywalls
- Large number of features (~2000)

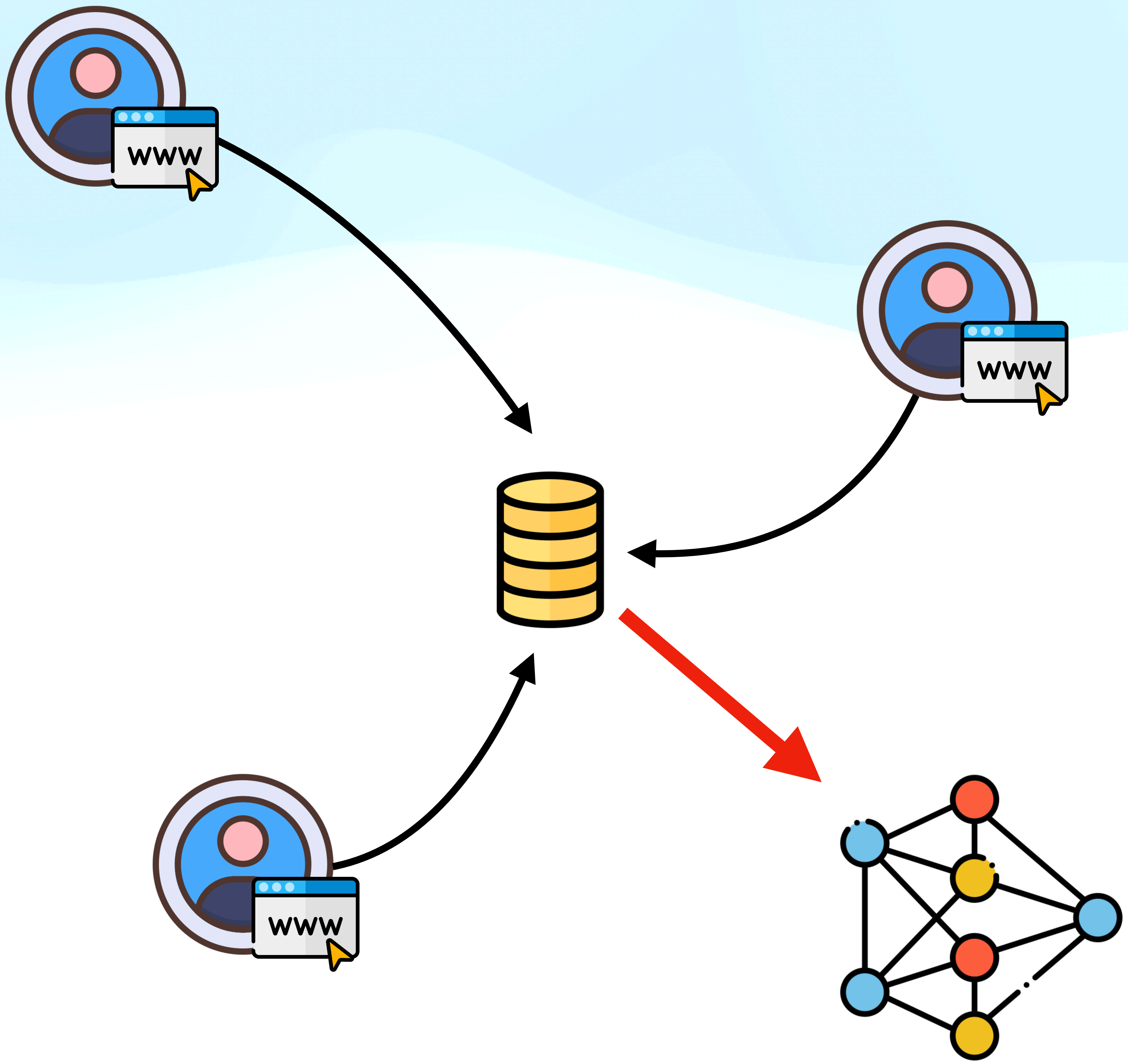
Might miss fingerprinting scripts

All icons are from www.flaticon.com

Naive solution

Browser Fingerprinting Detection

- Gather real-world observations from users as they browse websites

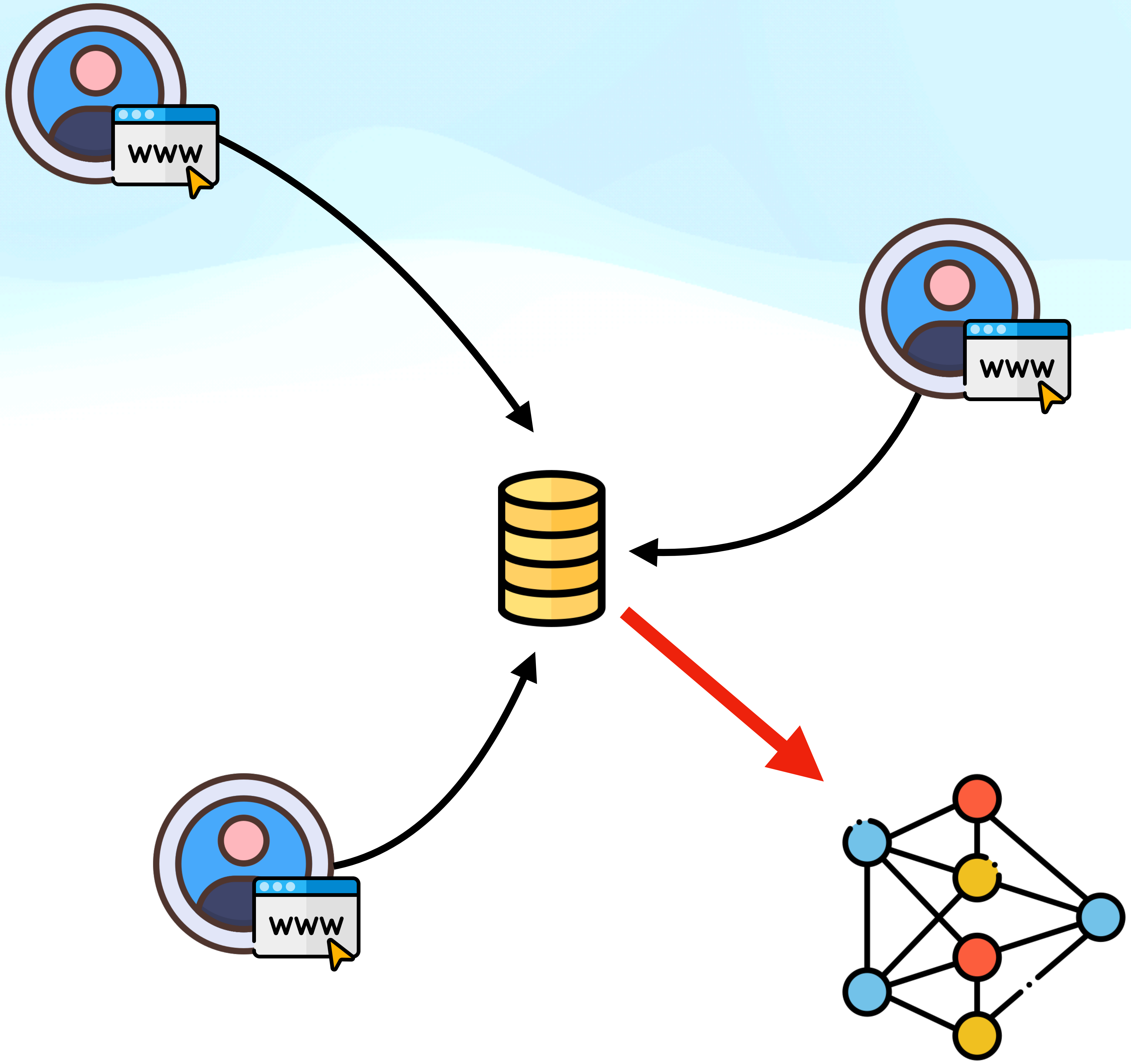


All icons are from www.flaticon.com

Naive solution

Browser Fingerprinting Detection

- Gather real-world observations from users as they browse websites
- Data collected from websites **might reveal sensitive information** (e.g., medical conditions)



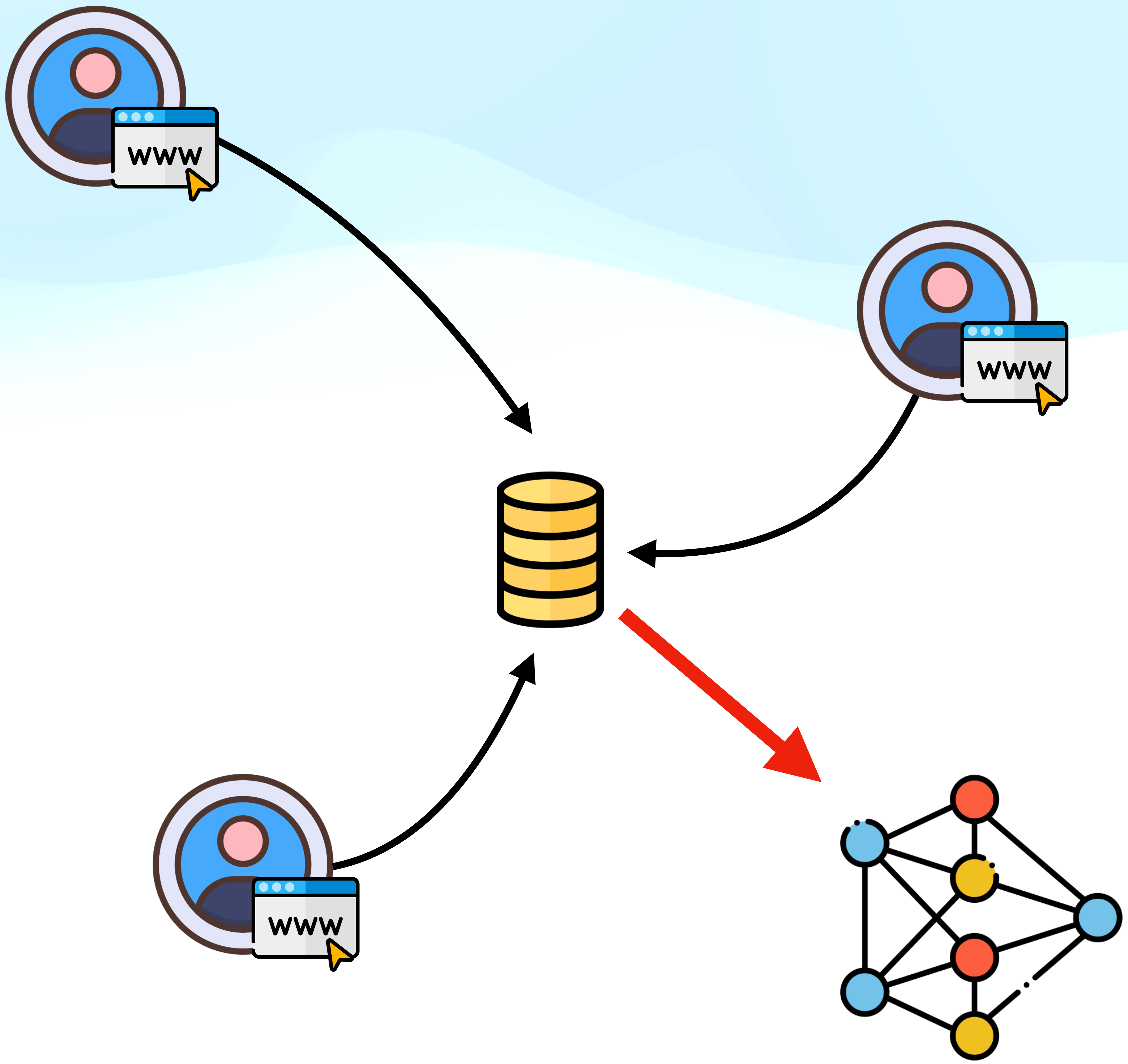
All icons are from www.flaticon.com

Naive solution

Browser Fingerprinting Detection

- Gather real-world observations from users as they browse websites
- Data collected from websites **might reveal sensitive information** (e.g., medical conditions)

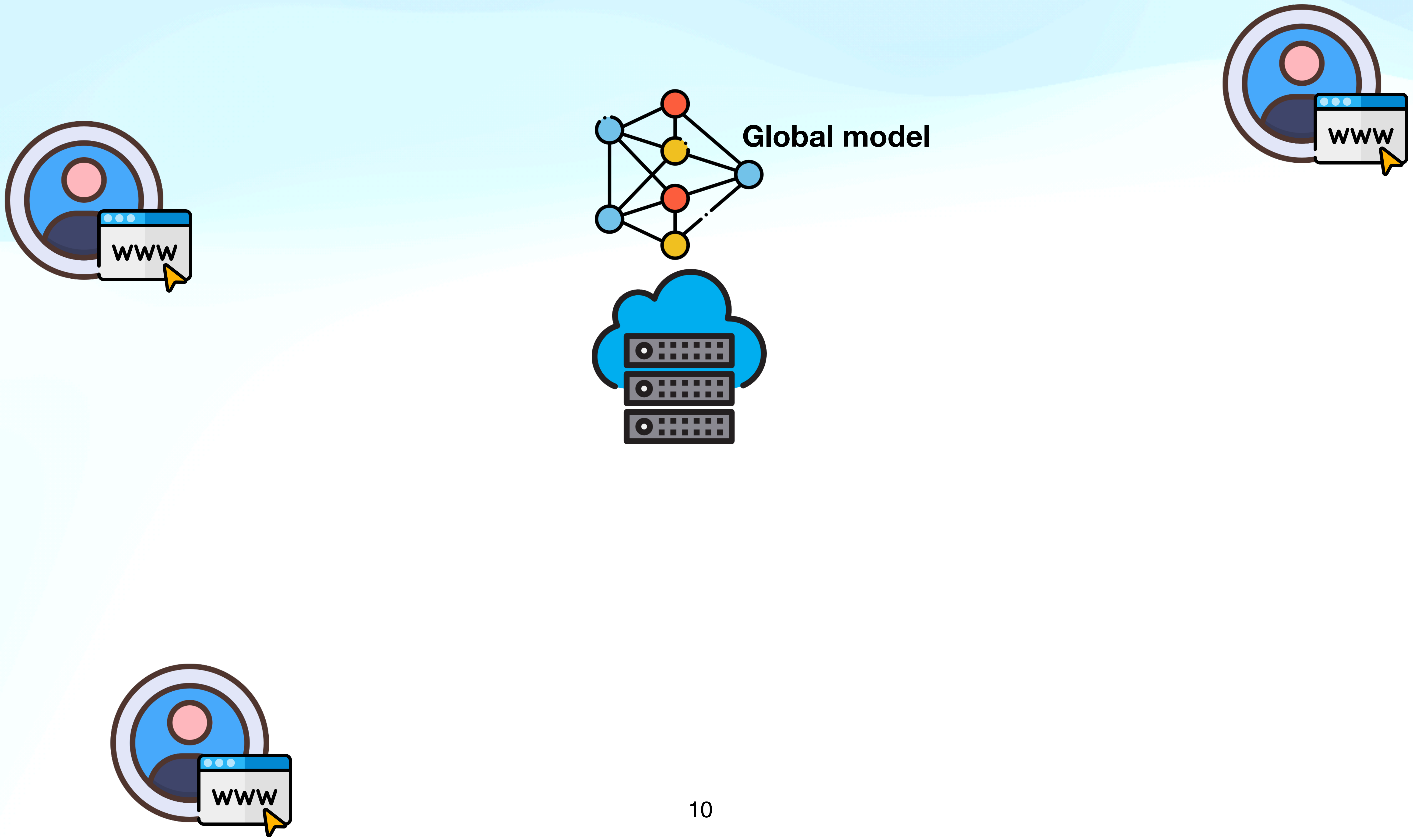
➡**Affect user's privacy**



All icons are from www.flaticon.com

Differentially Private Federated Learning (DP-FL)

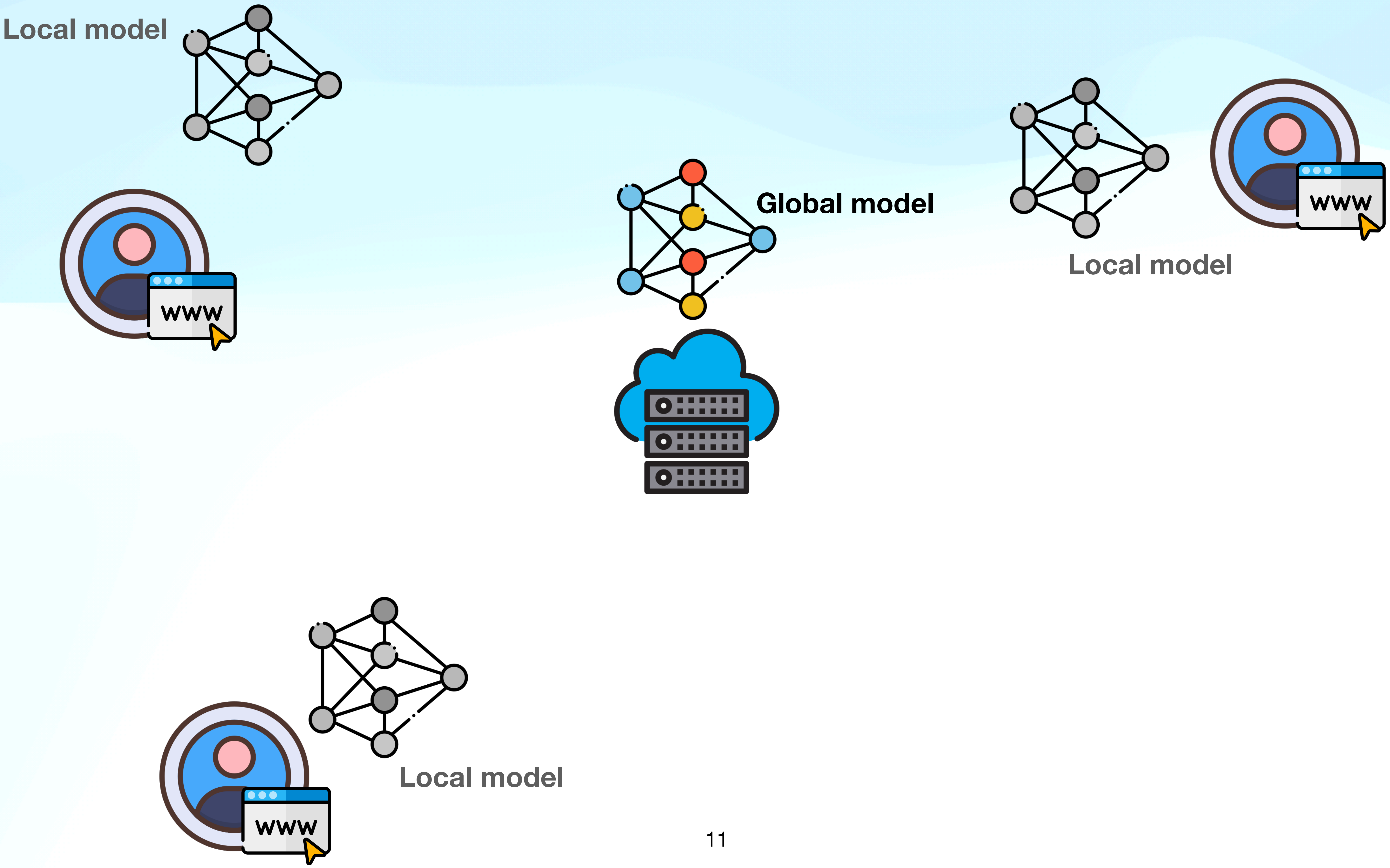
DP-FedAvg



All icons are from www.flaticon.com

DP-FedAvg

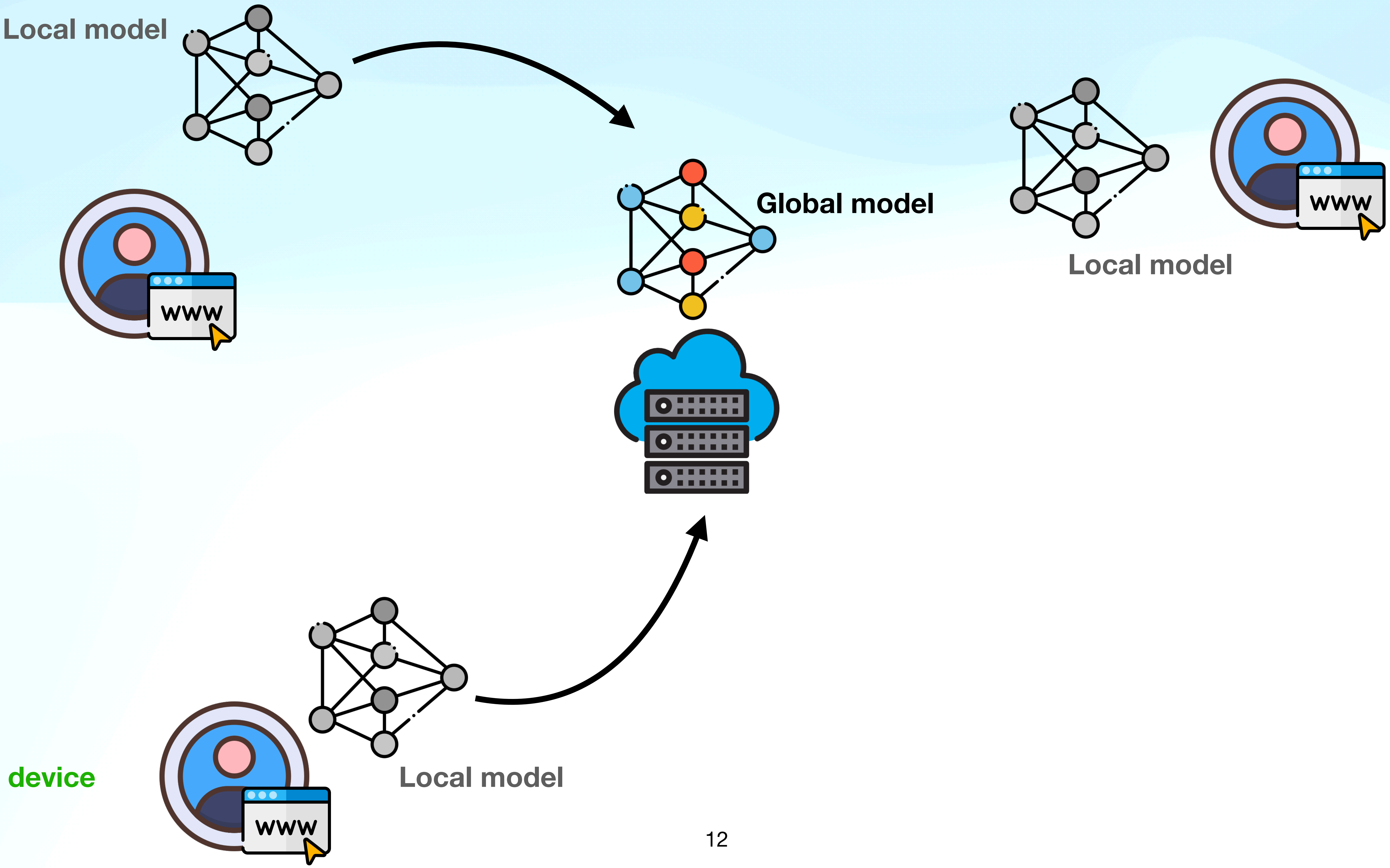
Each user trains local model



All icons are from www.flaticon.com

DP-FedAvg

Model updates are shared with the server

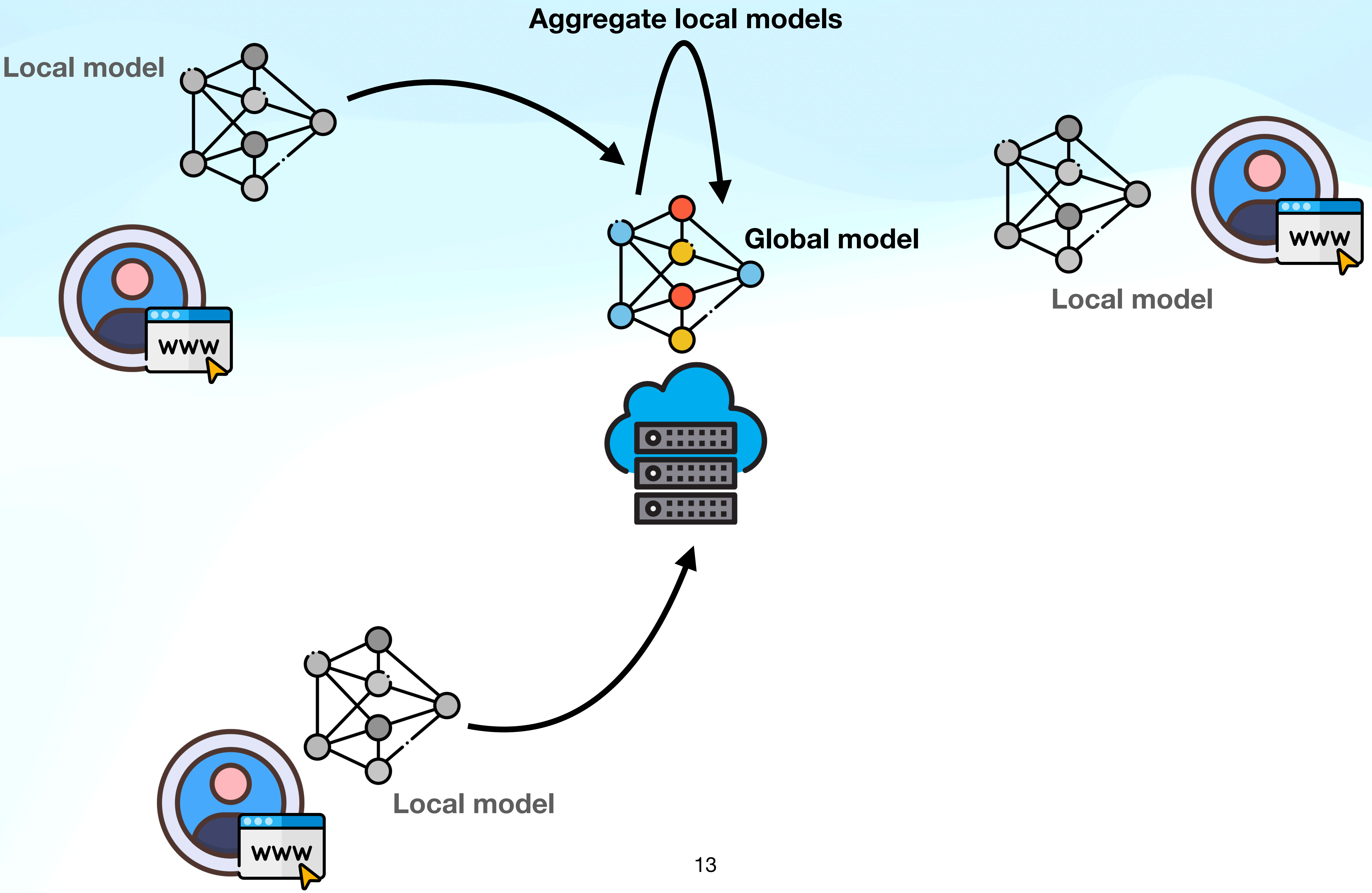


Raw data remains on device

All icons are from www.flaticon.com

DP-FedAvg

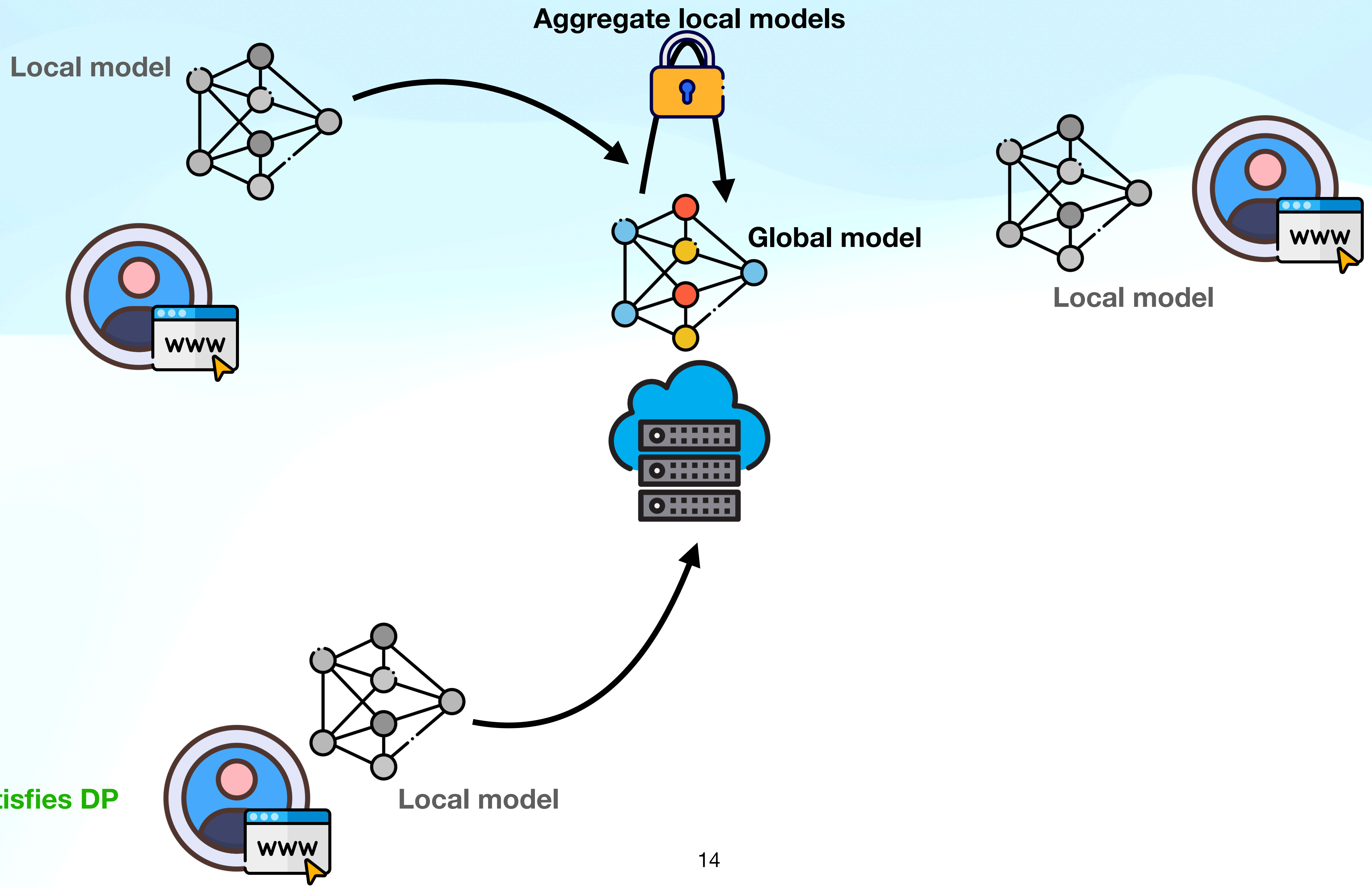
Server aggregates model updates



All icons are from www.flaticon.com

DP-FedAvg

Server adds statistical noise for privacy

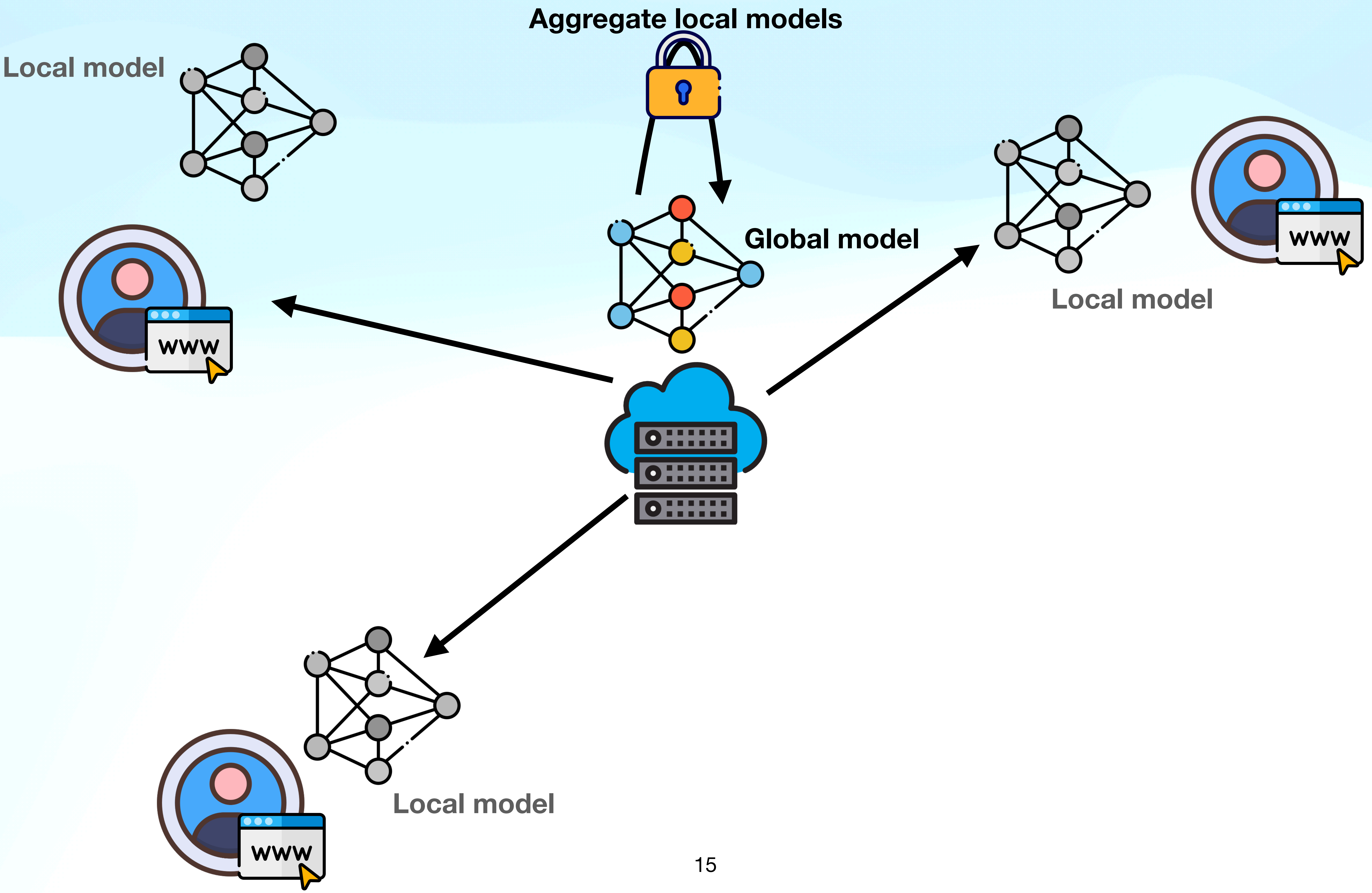


Aggregated model satisfies DP

All icons are from www.flaticon.com

DP-FedAvg

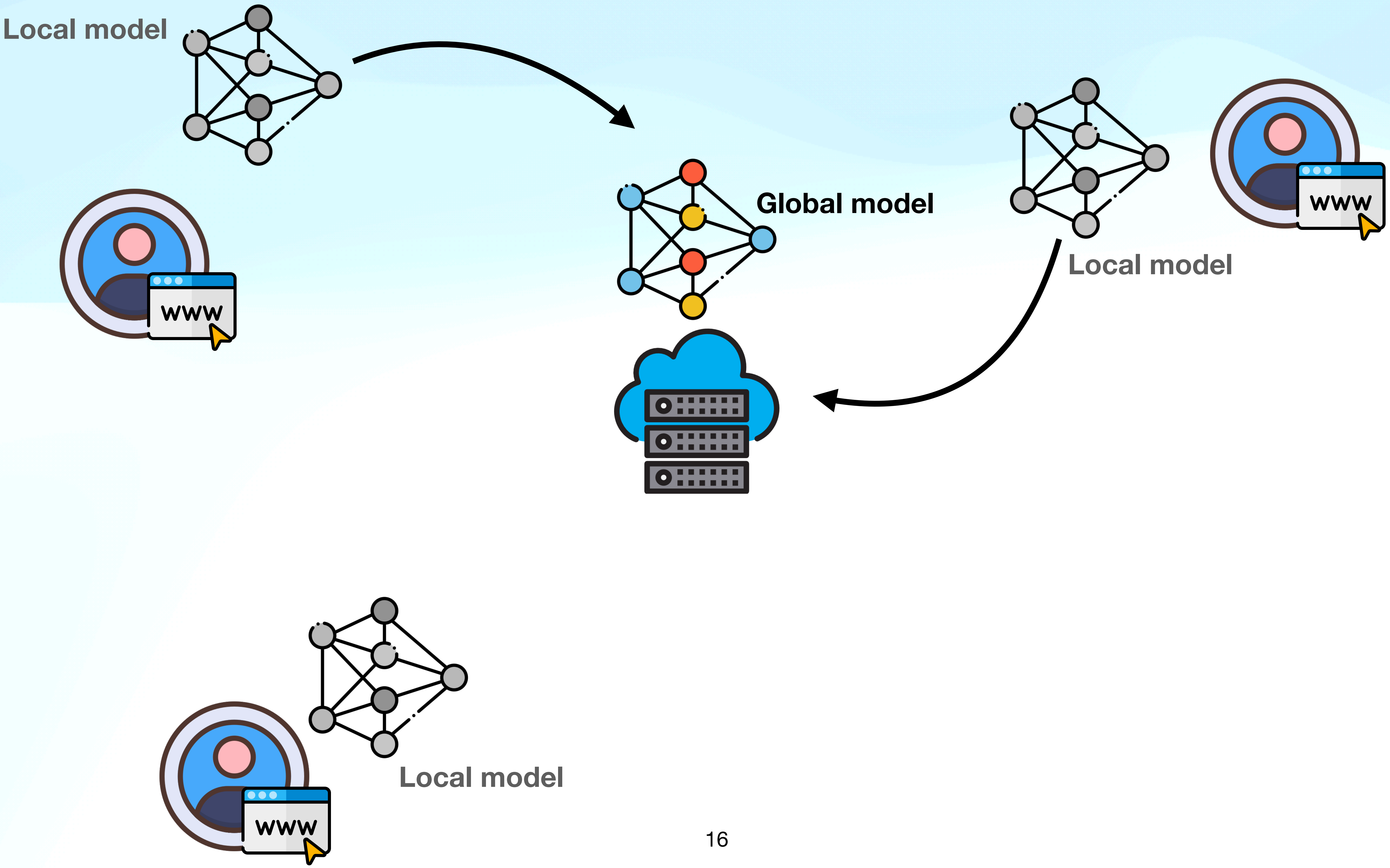
Updated model is shared with users



All icons are from www.flaticon.com

DP-FedAvg

Repeats until convergence

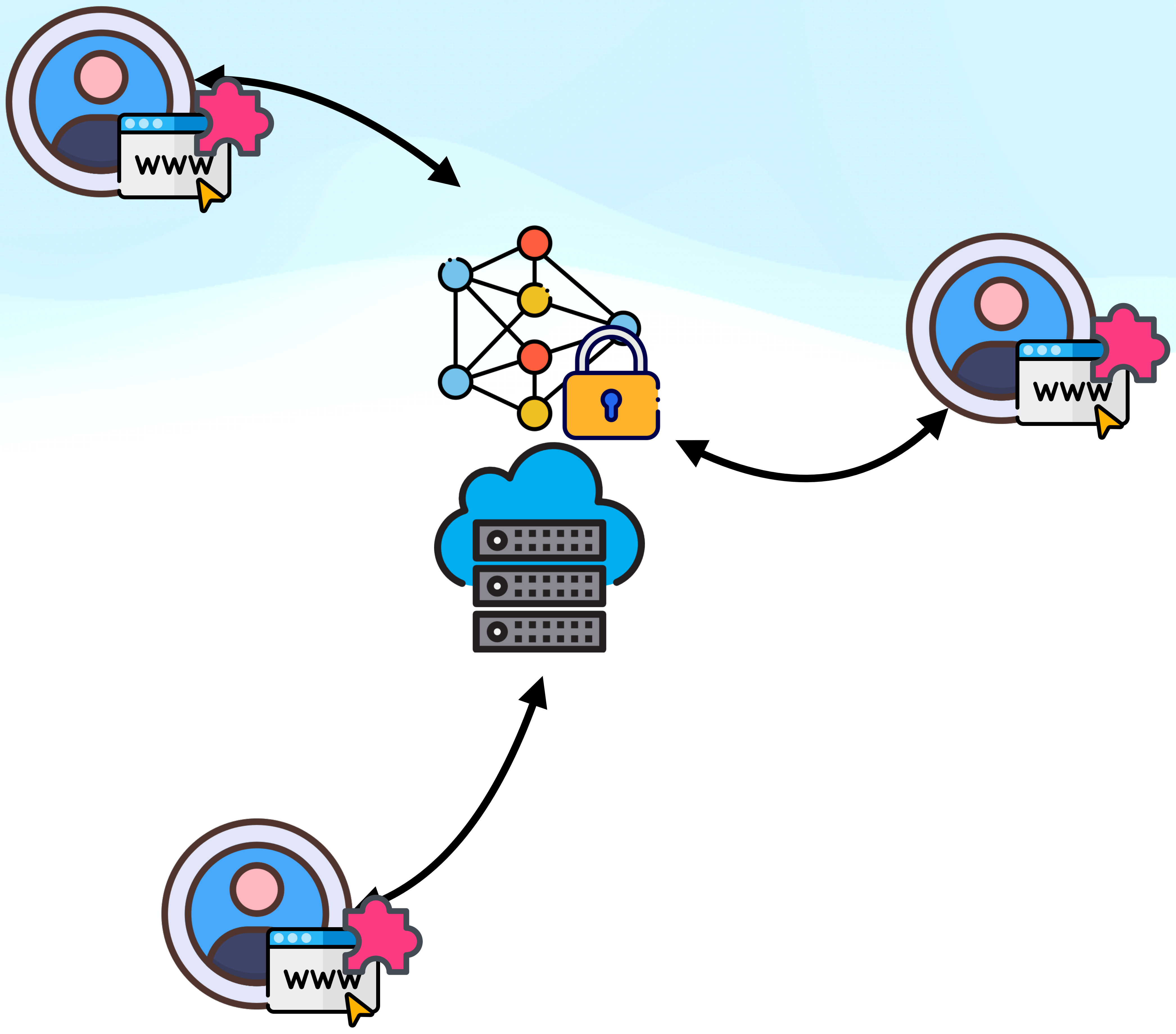


All icons are from www.flaticon.com

Applying DP-FL to Browser FP

Challenges

- Not trivial to federate existing classifiers **efficiently**

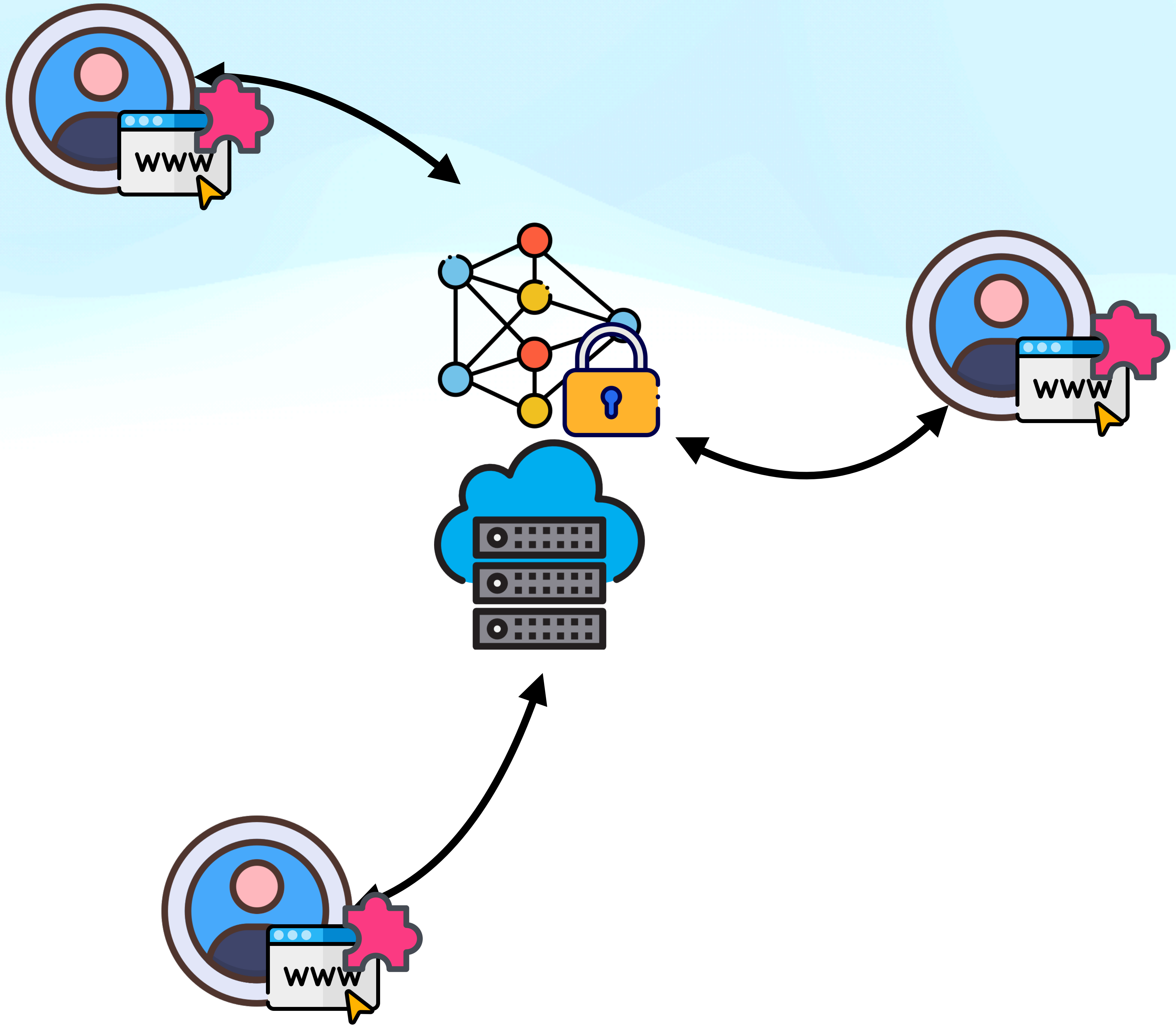


All icons are from www.flaticon.com

Applying DP-FL to Browser FP

Challenges

- Not trivial to federate existing classifiers **efficiently**
- **Intensive feature extraction** and **complex algorithms** may impact browser performance

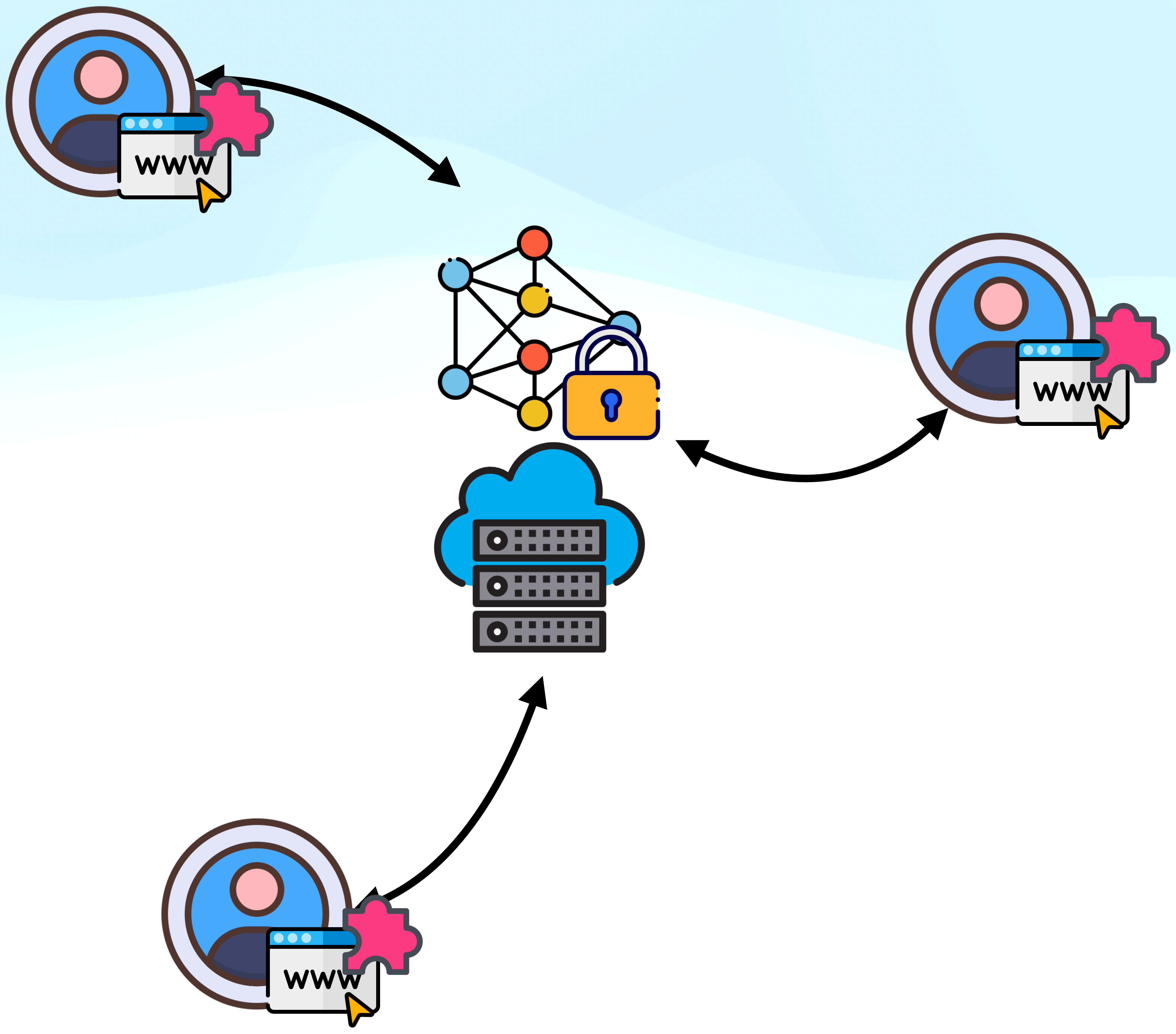


All icons are from www.flaticon.com

Applying DP-FL to Browser FP

Challenges

- Not trivial to federate existing classifiers **efficiently**
- **Intensive feature extraction** and **complex algorithms** may impact browser performance
- DP introduces **privacy-utility tradeoff**

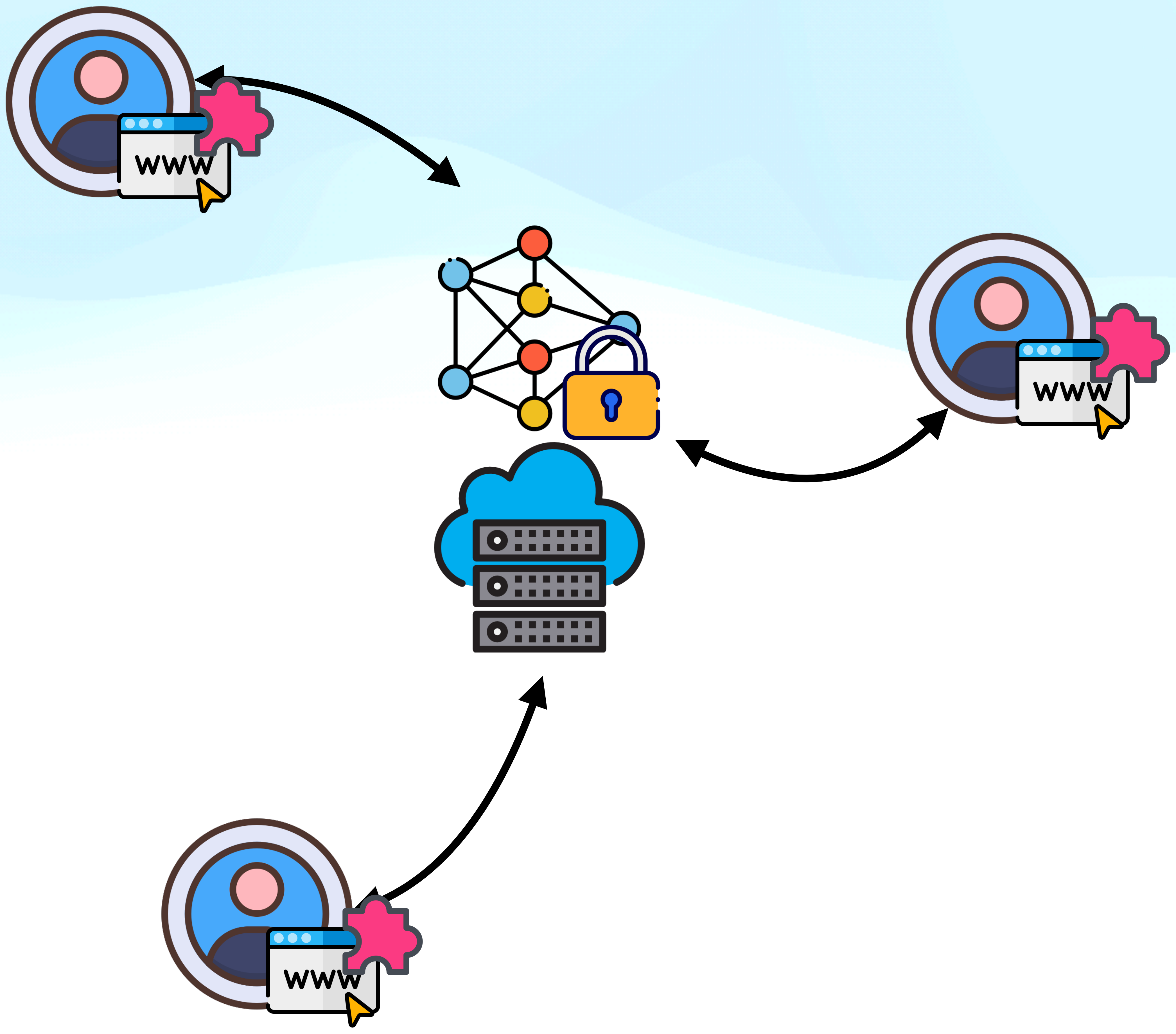


All icons are from www.flaticon.com

Our Work

FP-Fed

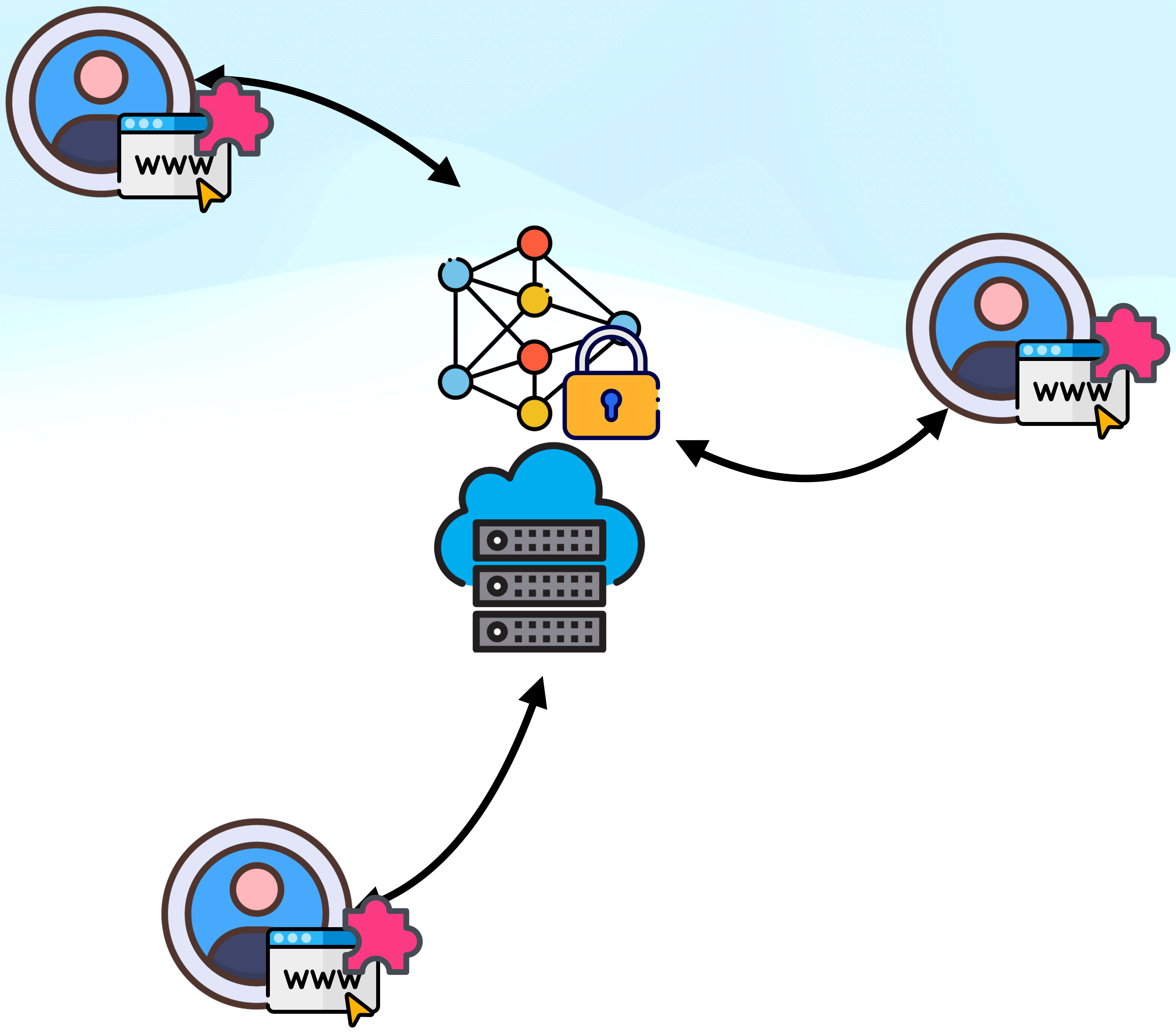
- **Distributed system (DP-FL)** for detecting browser fingerprinting in the wild



All icons are from www.flaticon.com

Our Work
FP-Fed

- **Distributed system (DP-FL)** for detecting browser fingerprinting in the wild
- Requires **minimal features** (~150)

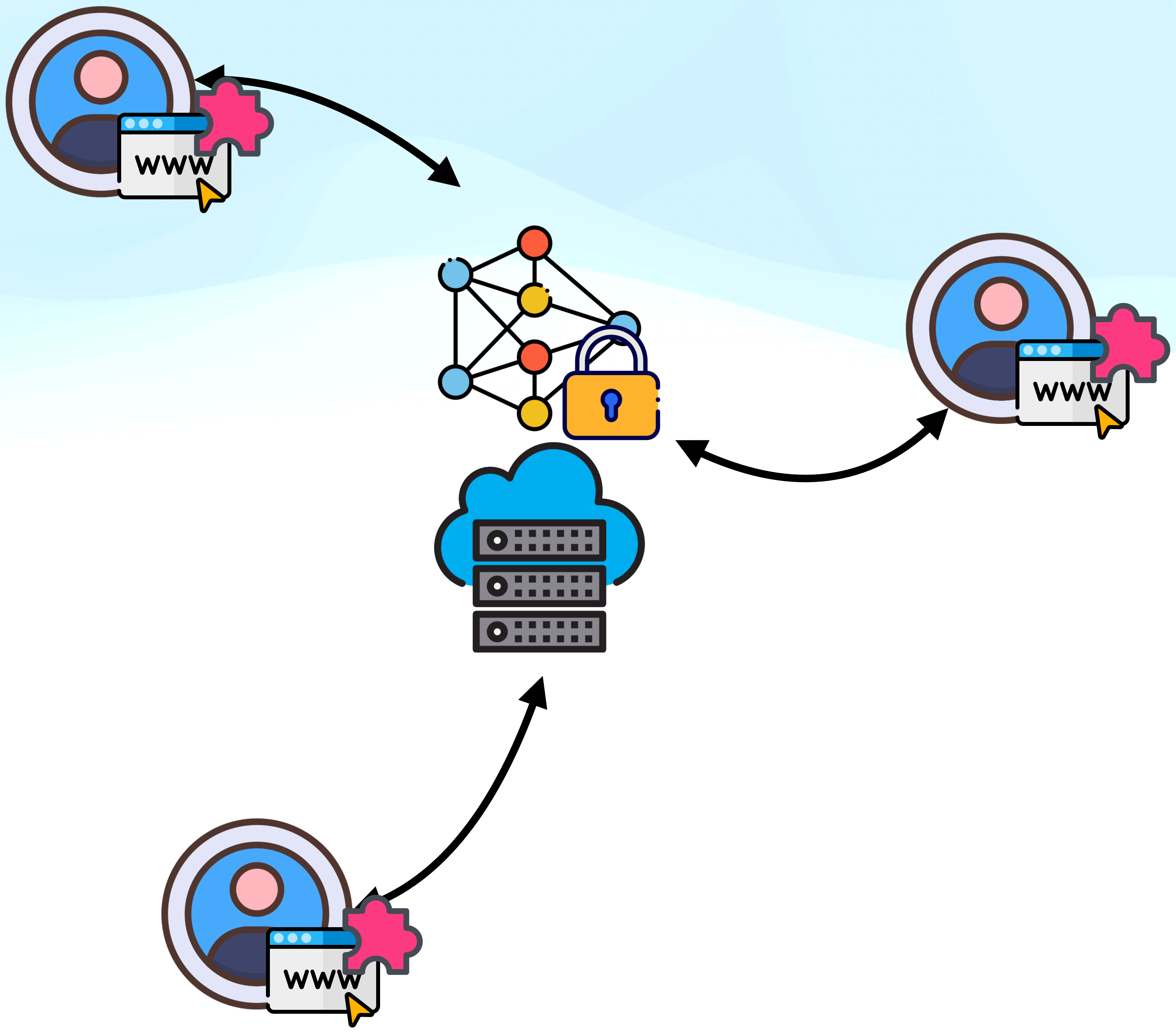


All icons are from www.flaticon.com

Our Work

FP-Fed

- **Distributed system (DP-FL)** for detecting browser fingerprinting in the wild
- Requires **minimal features** (~150)
- Achieves **high accuracy** with **minimal false positives** even with **formal privacy guarantees**



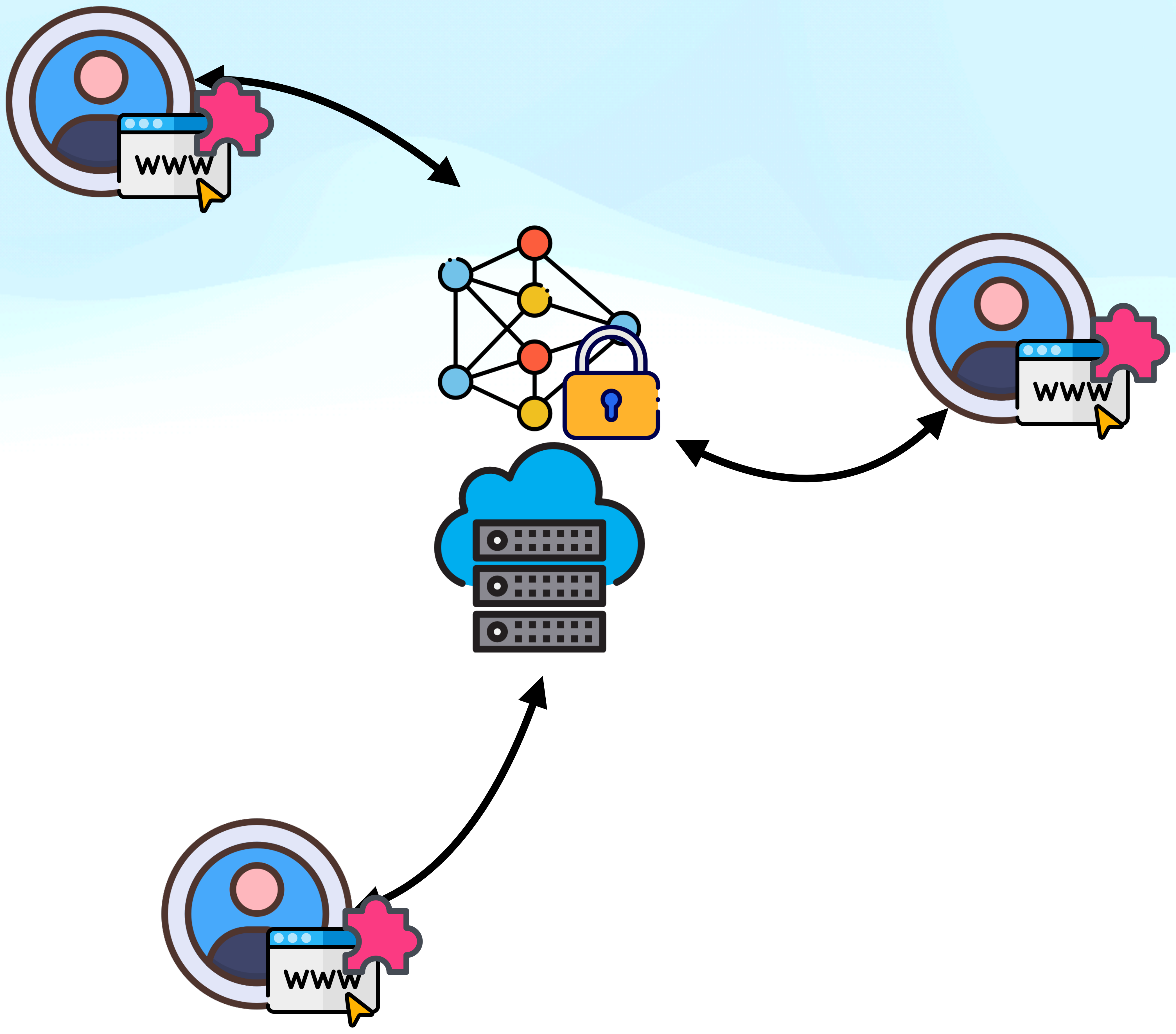
All icons are from www.flaticon.com

Our Work

FP-Fed

- **Distributed system (DP-FL)** for detecting browser fingerprinting in the wild
- Requires **minimal features** (~150)
- Achieves **high accuracy** with **minimal false positives** even with **formal privacy guarantees**

➡ Enables use of **real-world browsing patterns** instead of automated crawls

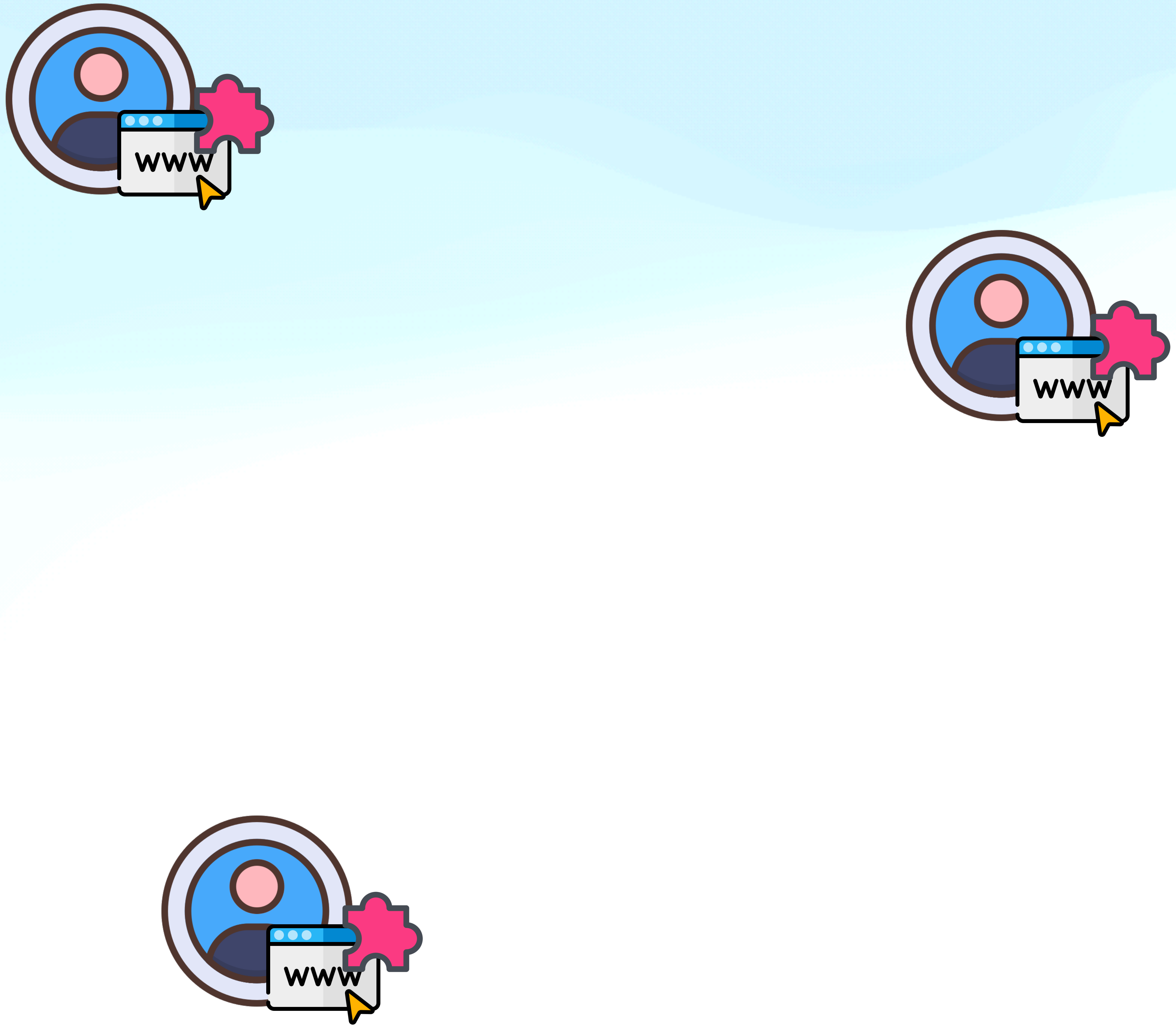


All icons are from www.flaticon.com

FP-Fed

FP-Fed

Step 1: Participants build local dataset



Step 1: Participants build local dataset

a) Feature Extraction

- API Call Counts (684)
  - # times **monitored APIs** are called
  - e.g., `CanvasRenderingContext2D.measureText` is called 50 times ⇒ Canvas Font fingerprinting

Step 1: Participants build local dataset
a) Feature Extraction

- API Call Counts (684)
  - # times **monitored APIs** are called
  - e.g., `CanvasRenderingContext2D.measureText` is called 50 times ⇒ Canvas Font fingerprinting
- Custom features (830)
  - Processed from **arguments and return values of API calls**

Step 1: Participants build local dataset

a) Feature Extraction

- API Call Counts (684)
  - # times **monitored APIs** are called
  - e.g., `CanvasRenderingContext2D.measureText` is called 50 times ⇒ Canvas Font fingerprinting
- Custom features (830)
  - Processed from **arguments and return values of API calls**

➡ Total features: 1514 (684 + 830)

Step 1: Participants build local dataset

b) Assign Ground Truth

- High-precision ground-truth heuristic defined by Iqbal et al.[1]
- Types of fingerprinting: Canvas, Canvas Font, WebRTC & Audio Context

[1] *Iqbal, U., Englehardt, S., & Shafiq, Z. (2021, May). Fingerprinting the Fingerprinters: Learning to Detect Browser Fingerprinting Behaviors. In IEEE S&P*

Step 1: Participants build local dataset
c) DP Federated Feature Pre-processing

- Feature normalization
  - Normalize each feature to have **mean 0 and variance 1**

➡ Improve convergence of model

Step 1: Participants build local dataset

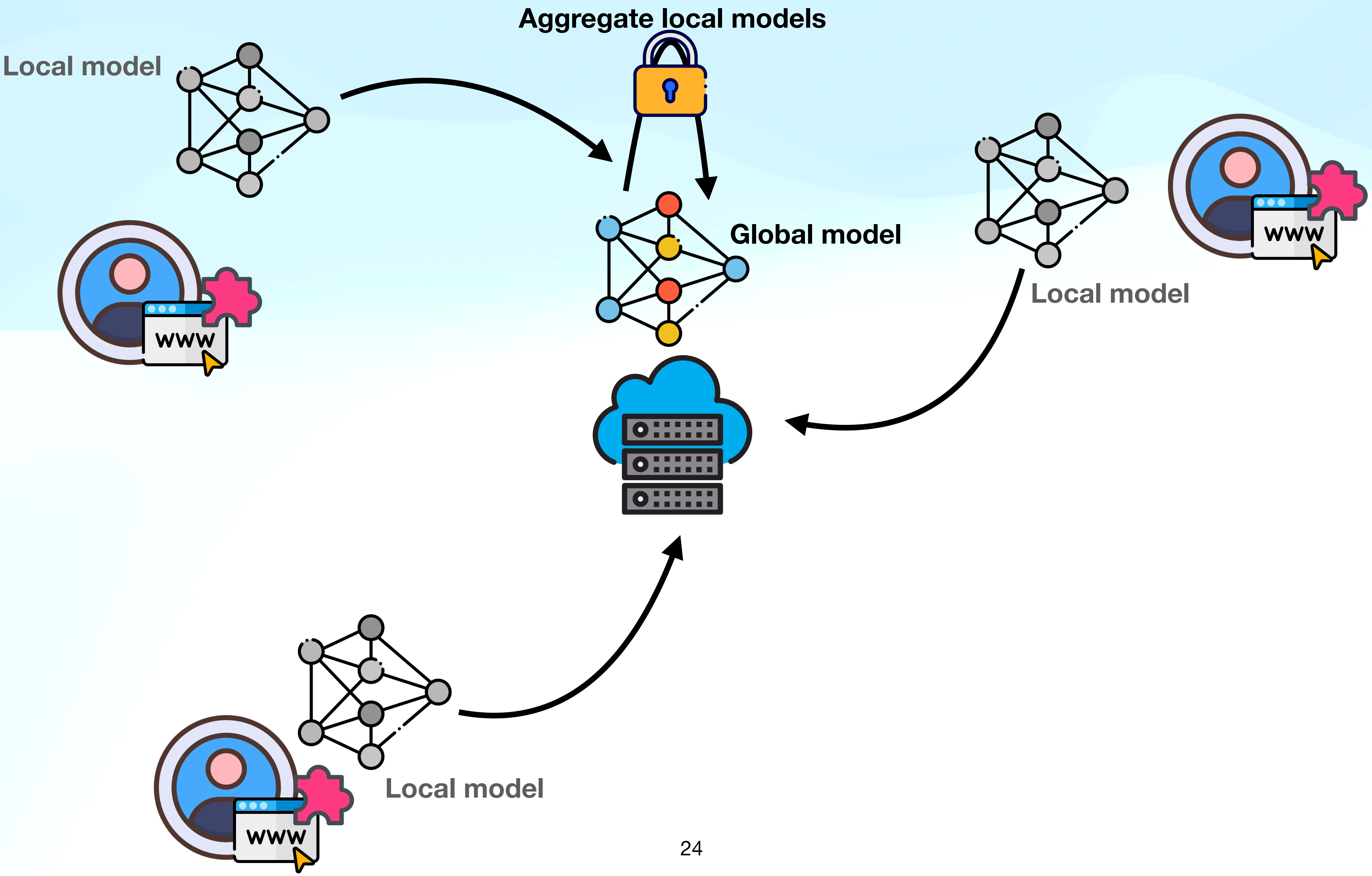
c) DP Federated Feature Pre-processing

- Feature normalization
  - Normalize each feature to have **mean 0 and variance 1**

  ➡ Improve convergence of model
- **DP-FedNorm:** Federate + Add DP noise to normalisation

FP-Fed

Step 2: Participants run DP-FedAvg



All icons are from www.flaticon.com

Experimental Setup

Experimental Setup
Dataset

- Ideally real-world browsing sessions will be collected
- For the purposes of this work, **automated crawl of 20k popular websites** sampled from Chrome User Experience Reports
- 18k successfully visited, 181k unique scripts, **752 fingerprinting**

Experimental Setup

Simulating Participants

- Each participant **samples 100 domains** according to Zipf's law over popularity of websites from Tranco ranking
- Assign **scripts loaded by domain** to participant

Experimental Setup
Model

- Logistic Regression, LBFGS solver
- Report Area-Under-Precision-Recall-Curve (AUPRC)
  - Threshold can be adjusted, allowing precision-recall to be tuned
  - AUPRC summarizes model performance across variety of thresholds

Experimental Setup

Minimal Feature Set

- Collecting all 1514 features may be **computationally intensive** and raise **privacy concerns**
- Experiment with smaller feature sets (e.g., APIs that are natively tracked by Google Chrome — High Entropy APIs)

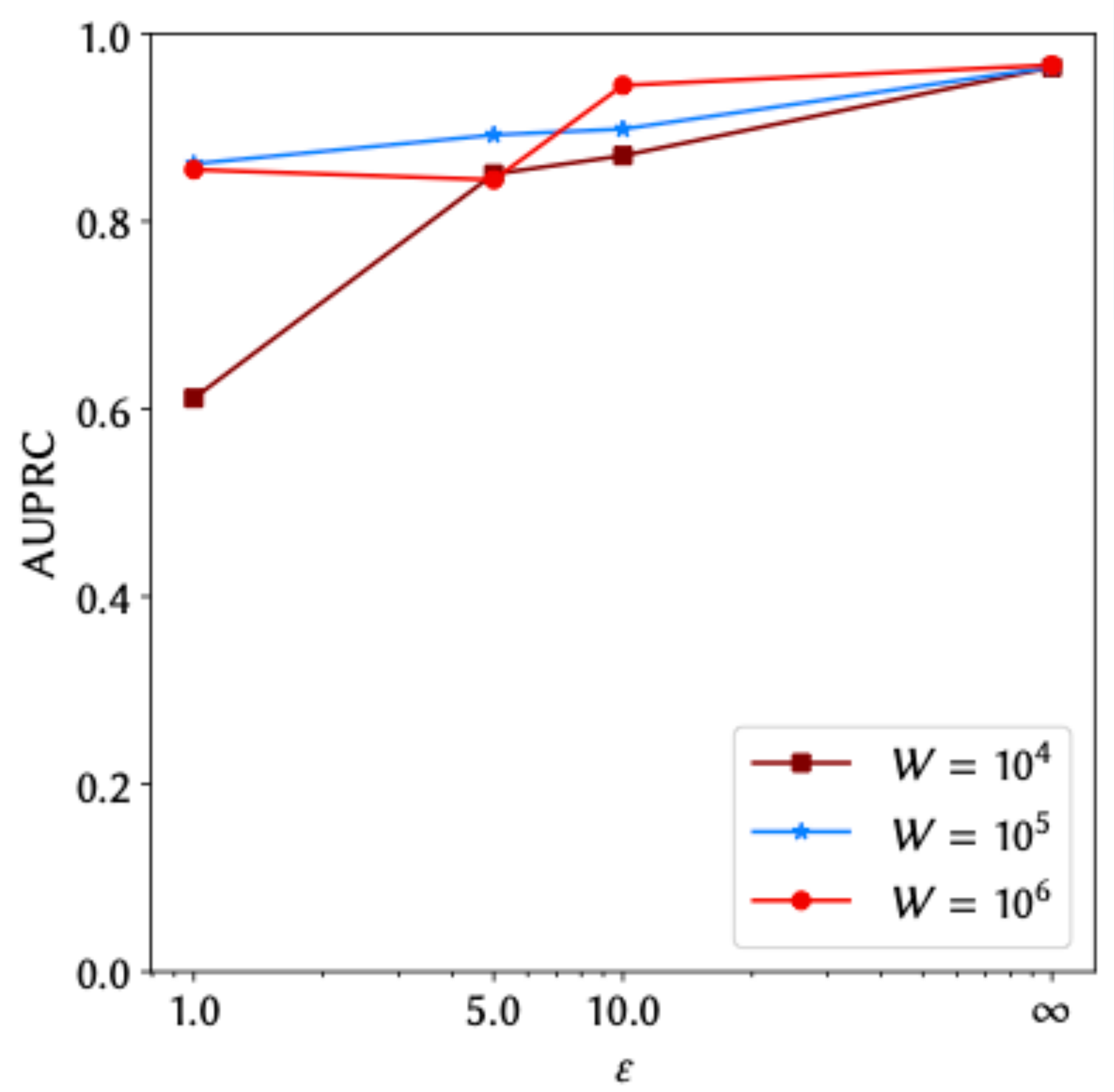
Feature Set	# API Call Counts	# Custom
API	684	830
FP-Inspector	500	830
JShelter	96	492
High Entropy APIs	109	0

Results

FP-Fed

Impact of ε (privacy parameter)

- *All* feature set
- 100 participants sampled per round

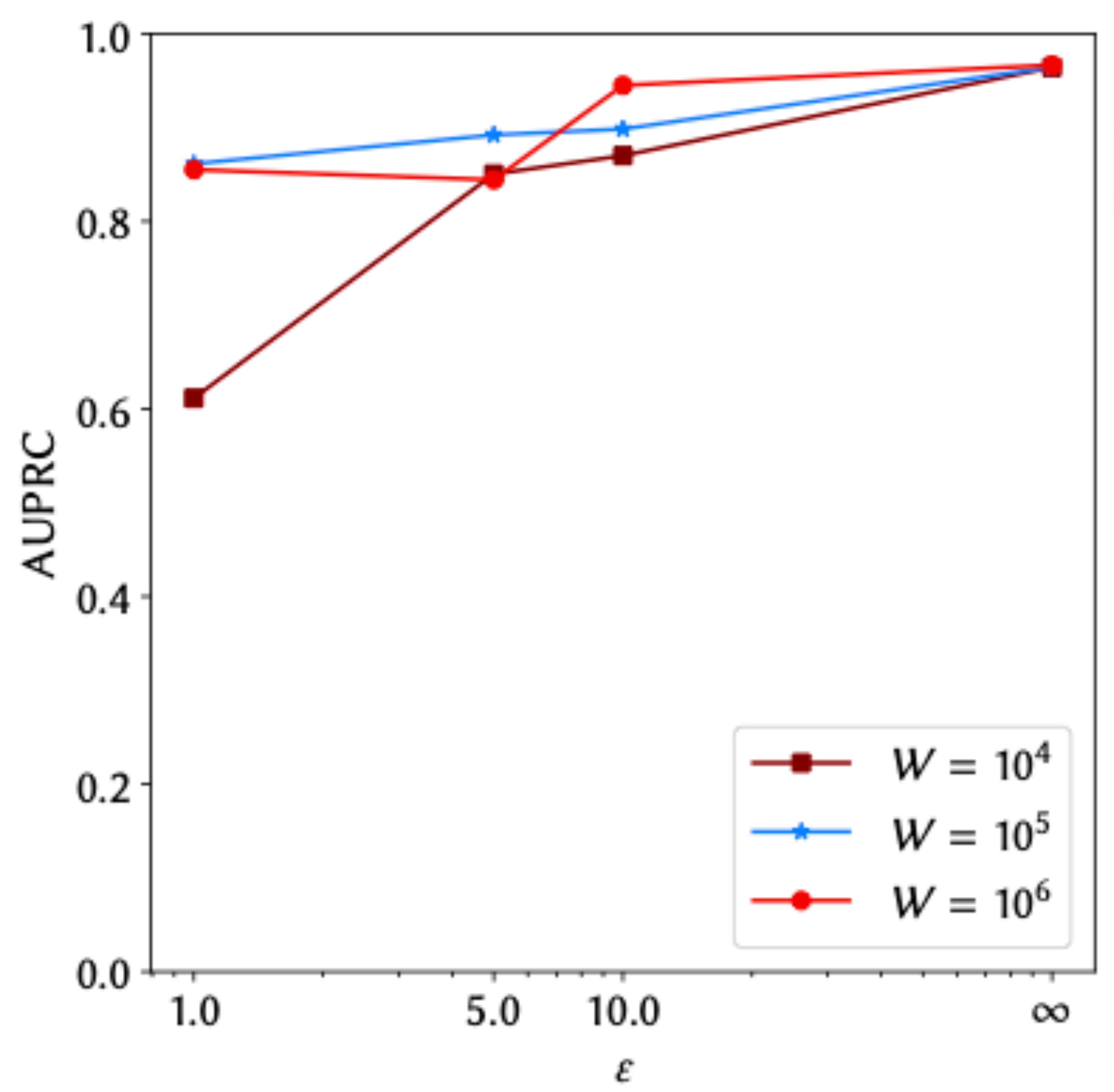


FP-Fed

Impact of ε (privacy parameter)

- *All* feature set
- 100 participants sampled per round

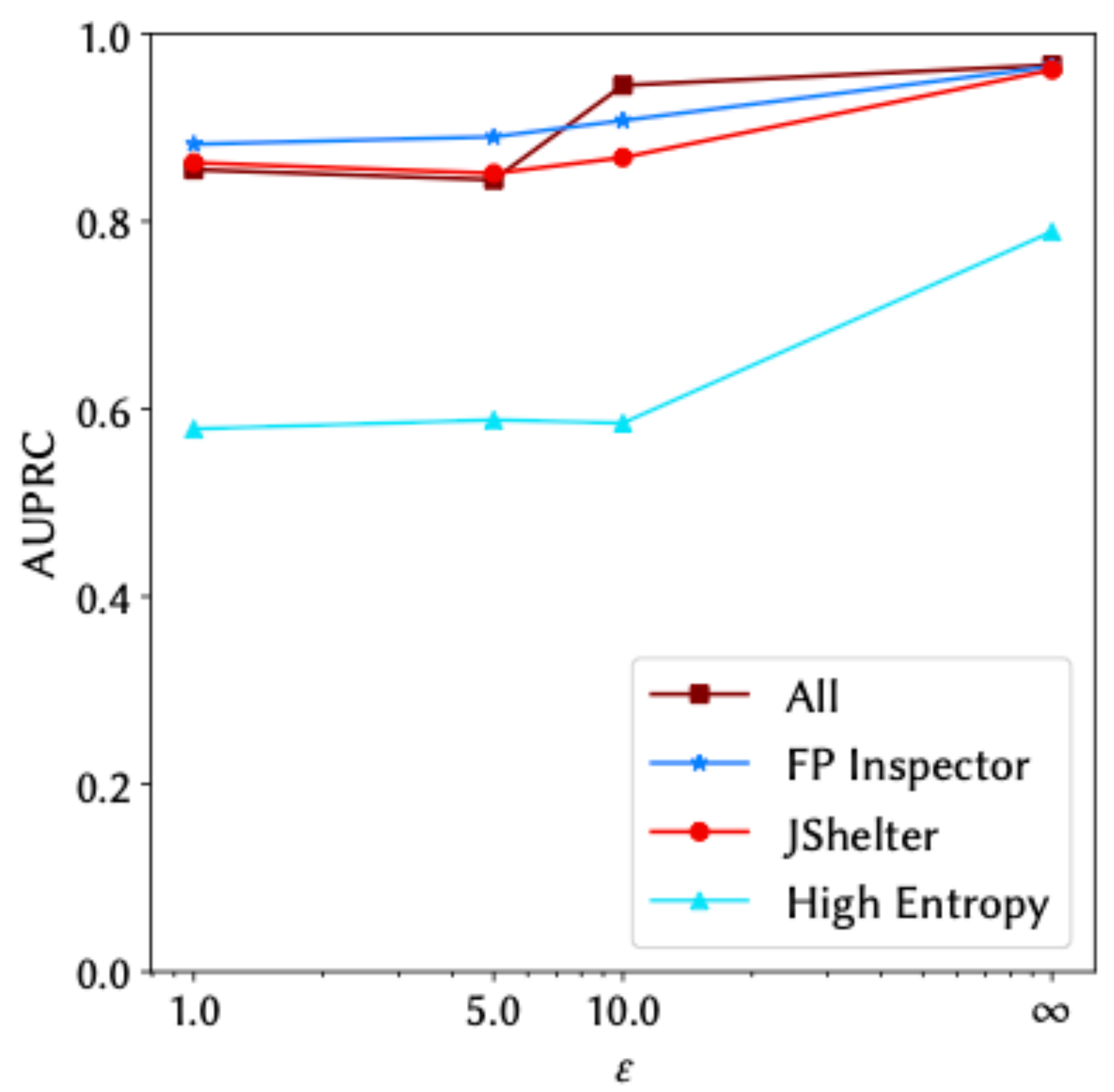
➡ **Having large number of participants is beneficial** even if not all are participating in each round



FP-Fed

Impact of feature set (*All*, *FP-Inspector*, *JShelter*, *High Entropy*)

- 1M total participants

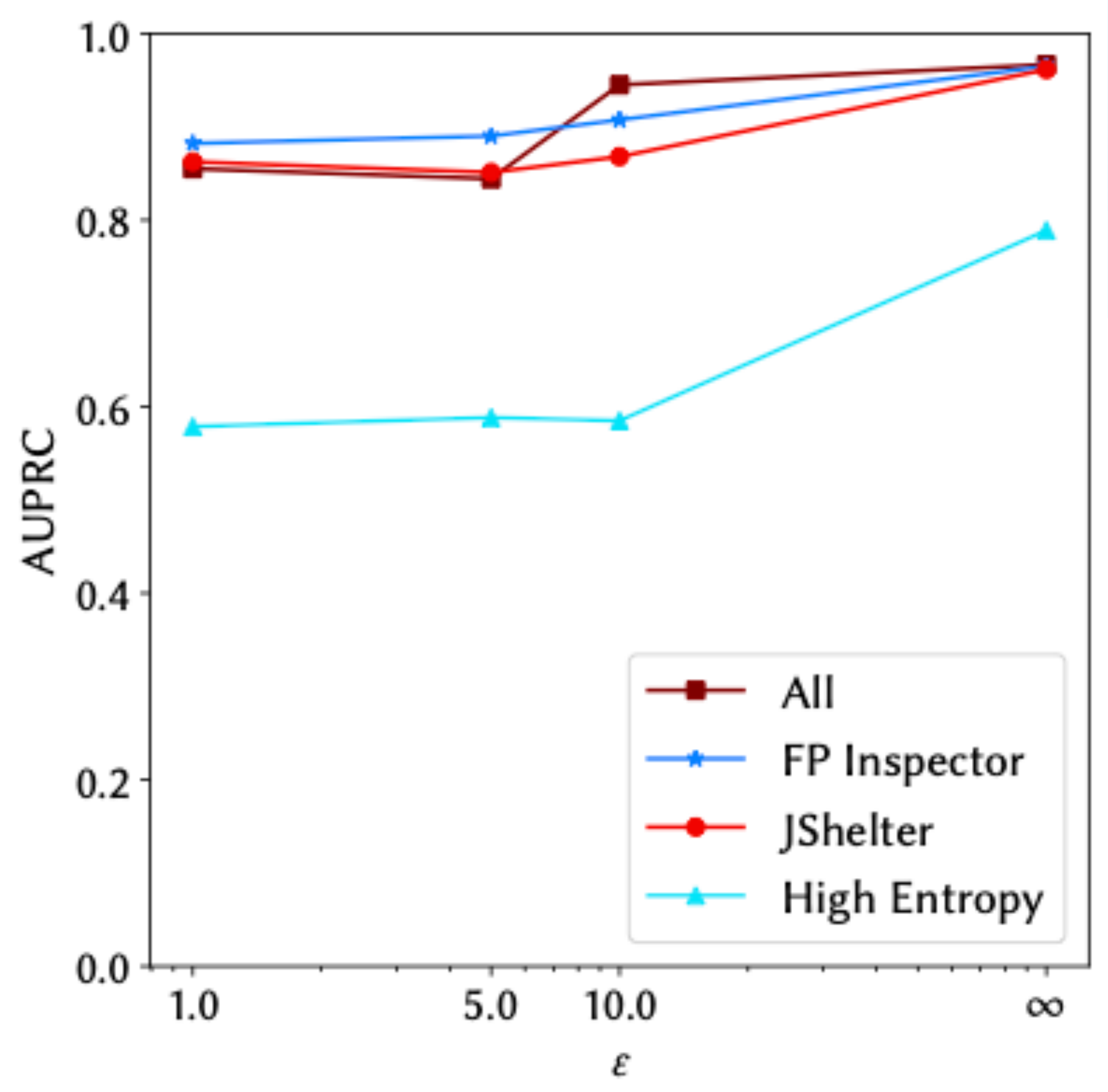


FP-Fed

Impact of feature set (*All*, *FP-Inspector*, *JShelter*, *High Entropy*)

- 1M total participants

➡ *JShelter* **performs similarly** to *All* and *FP-Inspector* even though it **only contains 40% of features**



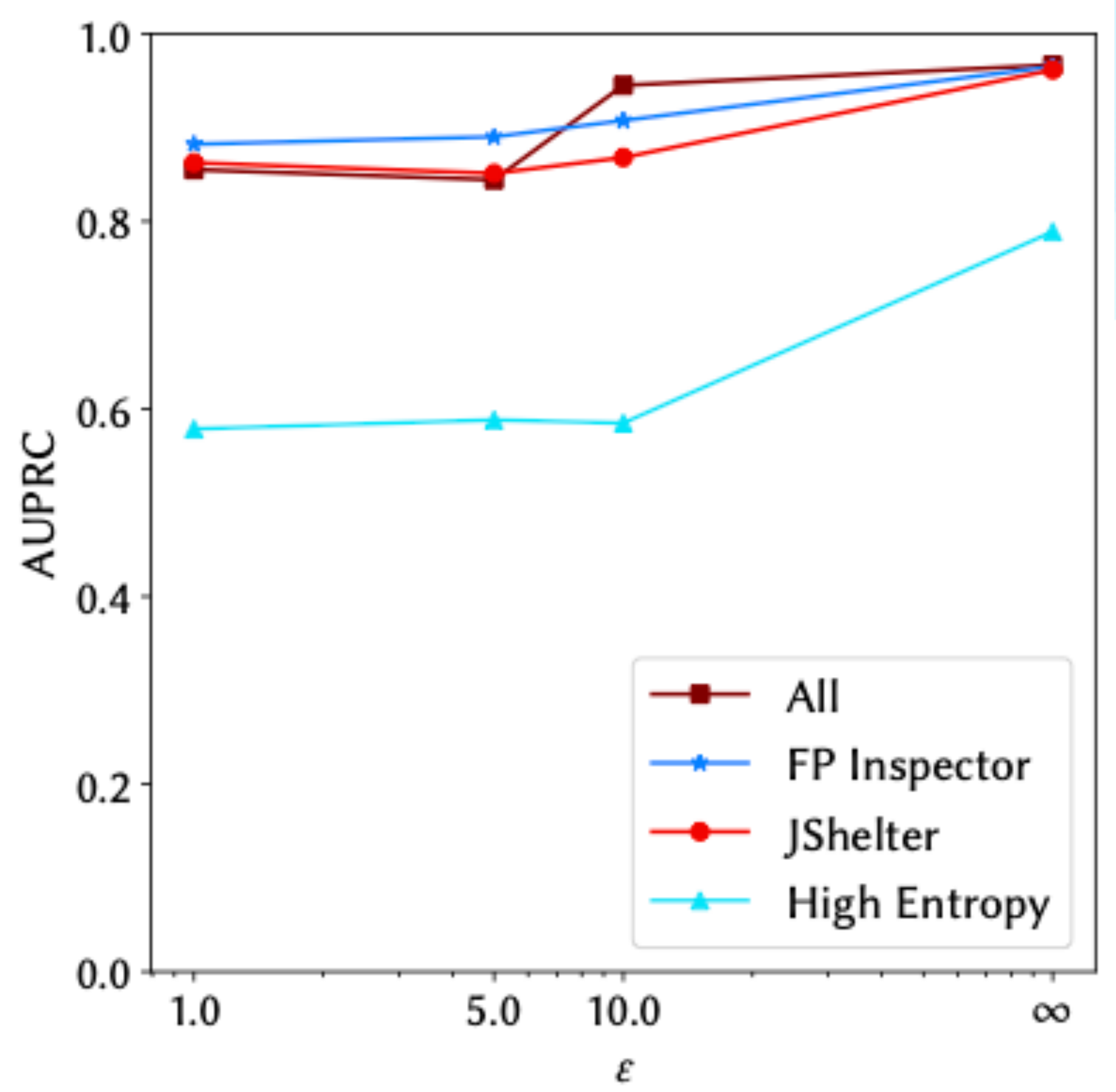
FP-Fed

Impact of feature set (*All*, *FP-Inspector*, *JShelter*, *High Entropy*)

- 1M total participants

➡ *JShelter* **performs similarly** to *All* and *FP-Inspector* even though it **only contains 40% of features**

➡ *High Entropy* is **ineffective even without DP**



Working towards a “minimal” feature set

Research Questions

- High Entropy APIs **have too little features** to reliably detect Browser FP

Working towards a “minimal” feature set

Research Questions

- High Entropy APIs **have too little features** to reliably detect Browser FP
- RQ: **How many features** does FP-Fed need to perform well?
  - RQ1: How many API call counts?
  - RQ2: How many custom features?

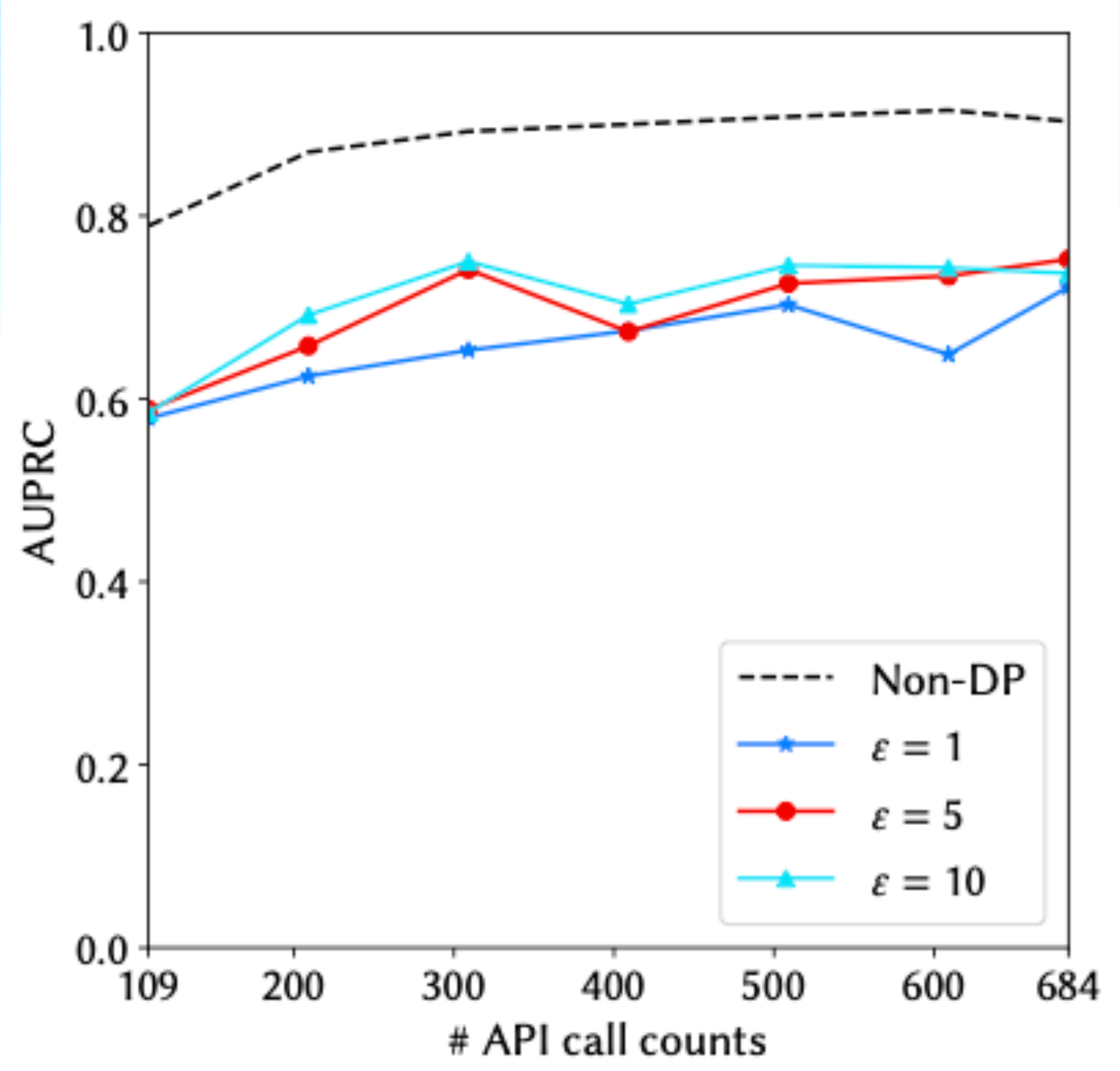
Working towards a “minimal” feature set

Methodology

- Model parameters are used as **“feature importance” score**
- Sort features according to score and **add features to *High Entropy* incrementally**

Working towards a “minimal” feature set

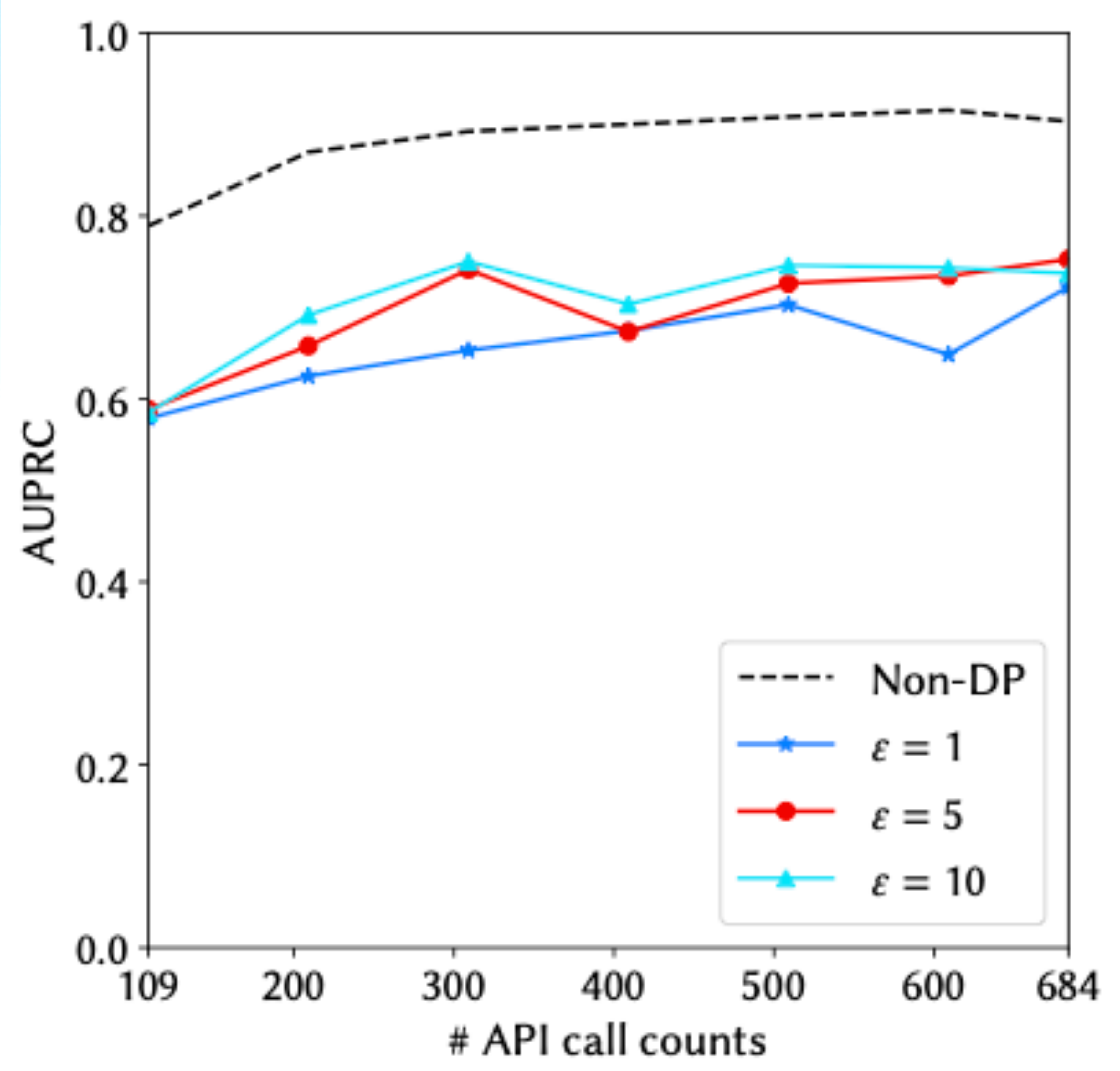
RQ1: How many API Call Counts necessary?



Working towards a “minimal” feature set

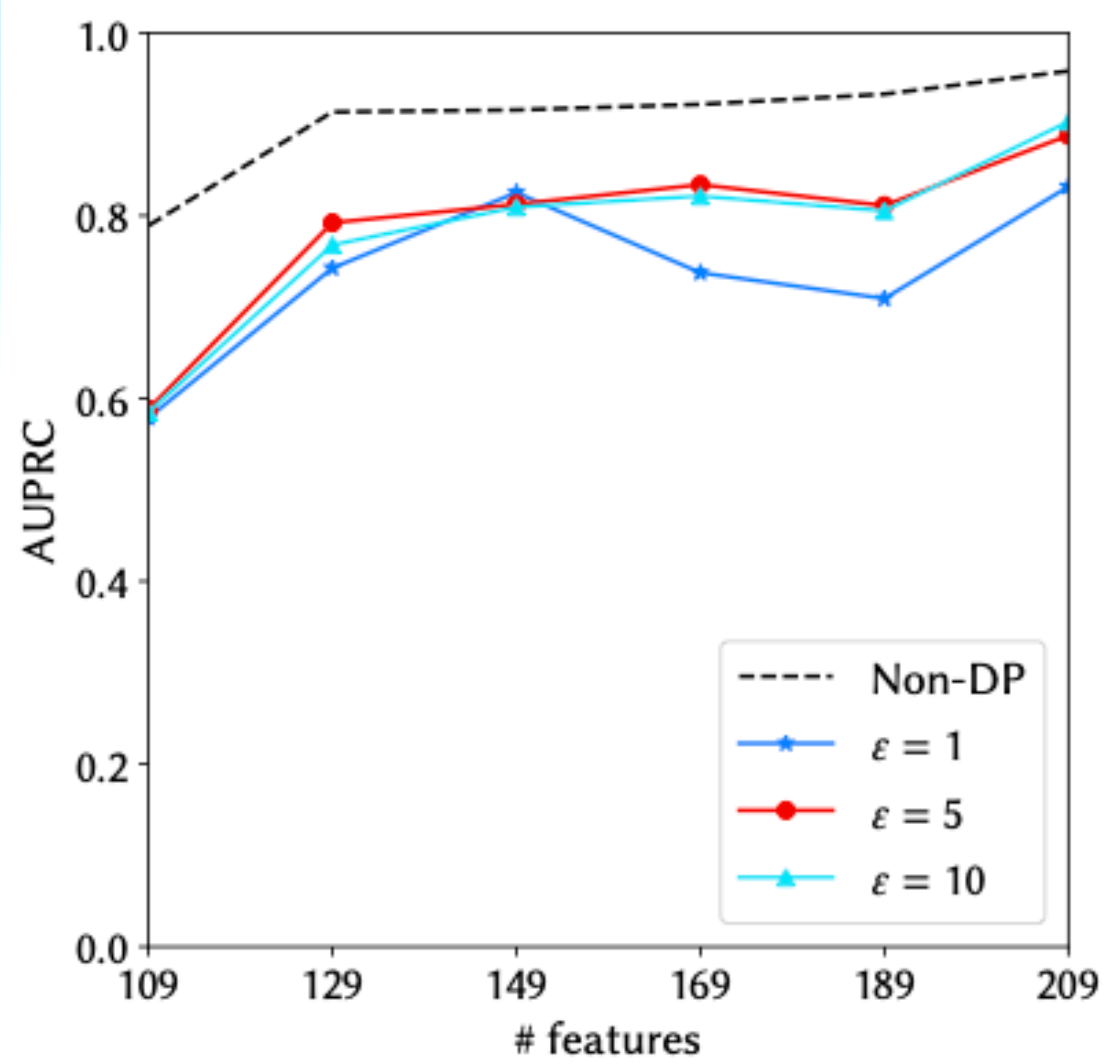
RQ1: How many API Call Counts necessary?

➡ API call counts by themselves are not enough to reliably detect fingerprinting



Working towards a “minimal” feature set

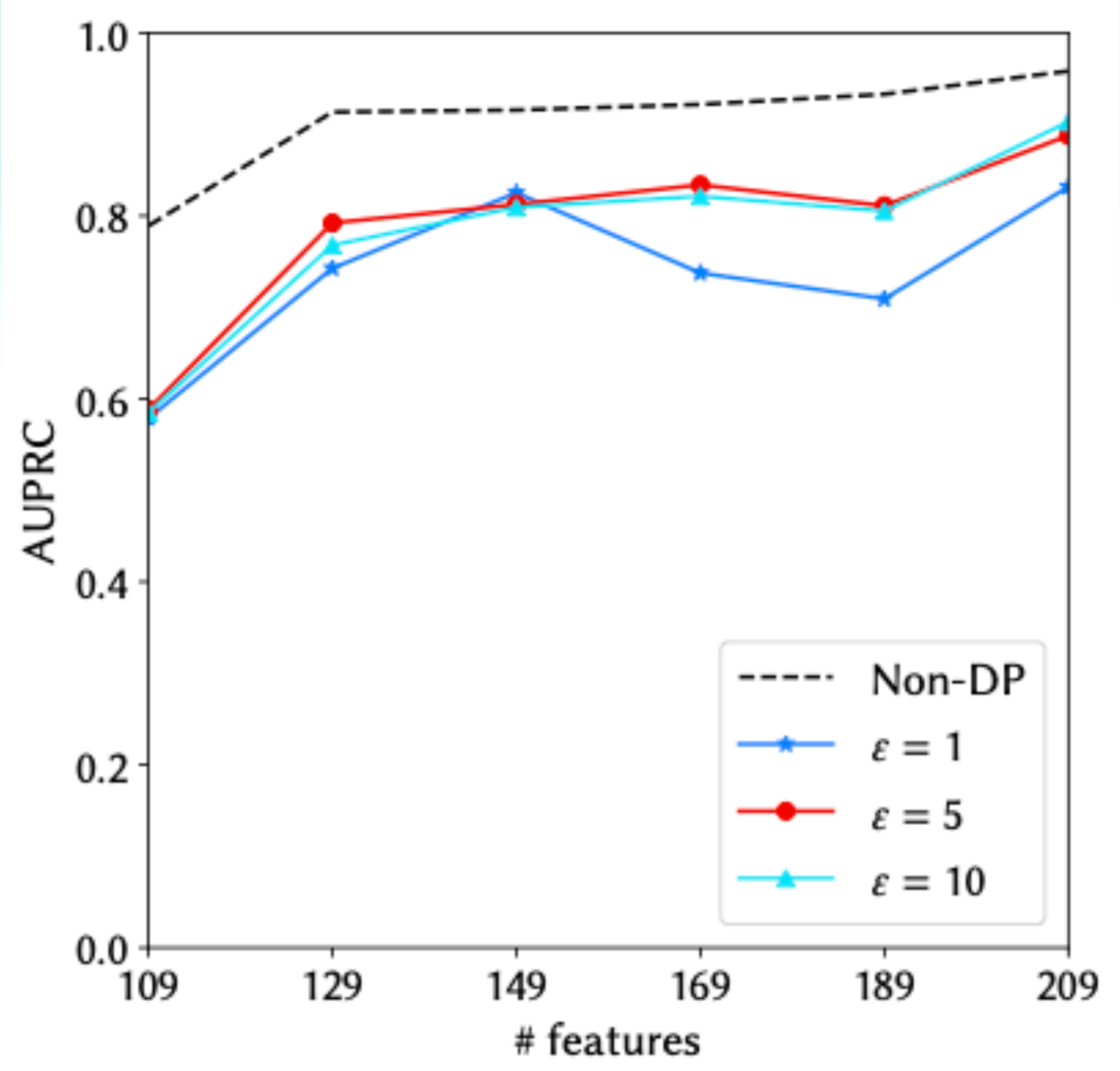
RQ2: How many custom features necessary?



Working towards a “minimal” feature set

RQ2: How many custom features necessary?

➡ DP-FedNorm might cause **momentary drops** even with **more features** especially at **high privacy levels**

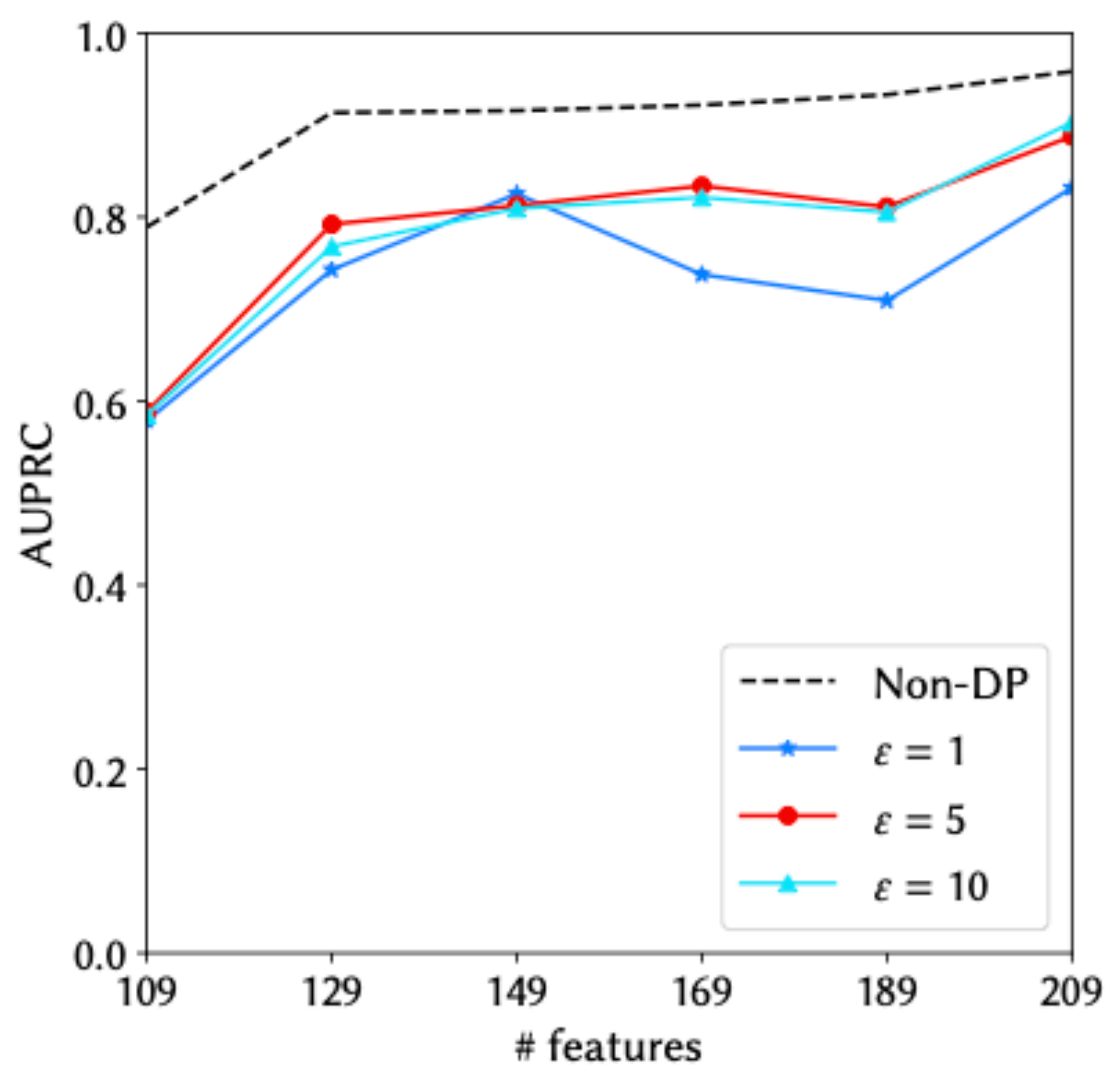


Working towards a “minimal” feature set

RQ2: How many custom features necessary?

➡ DP-FedNorm might cause **momentary drops** even with **more features** especially at **high privacy levels**

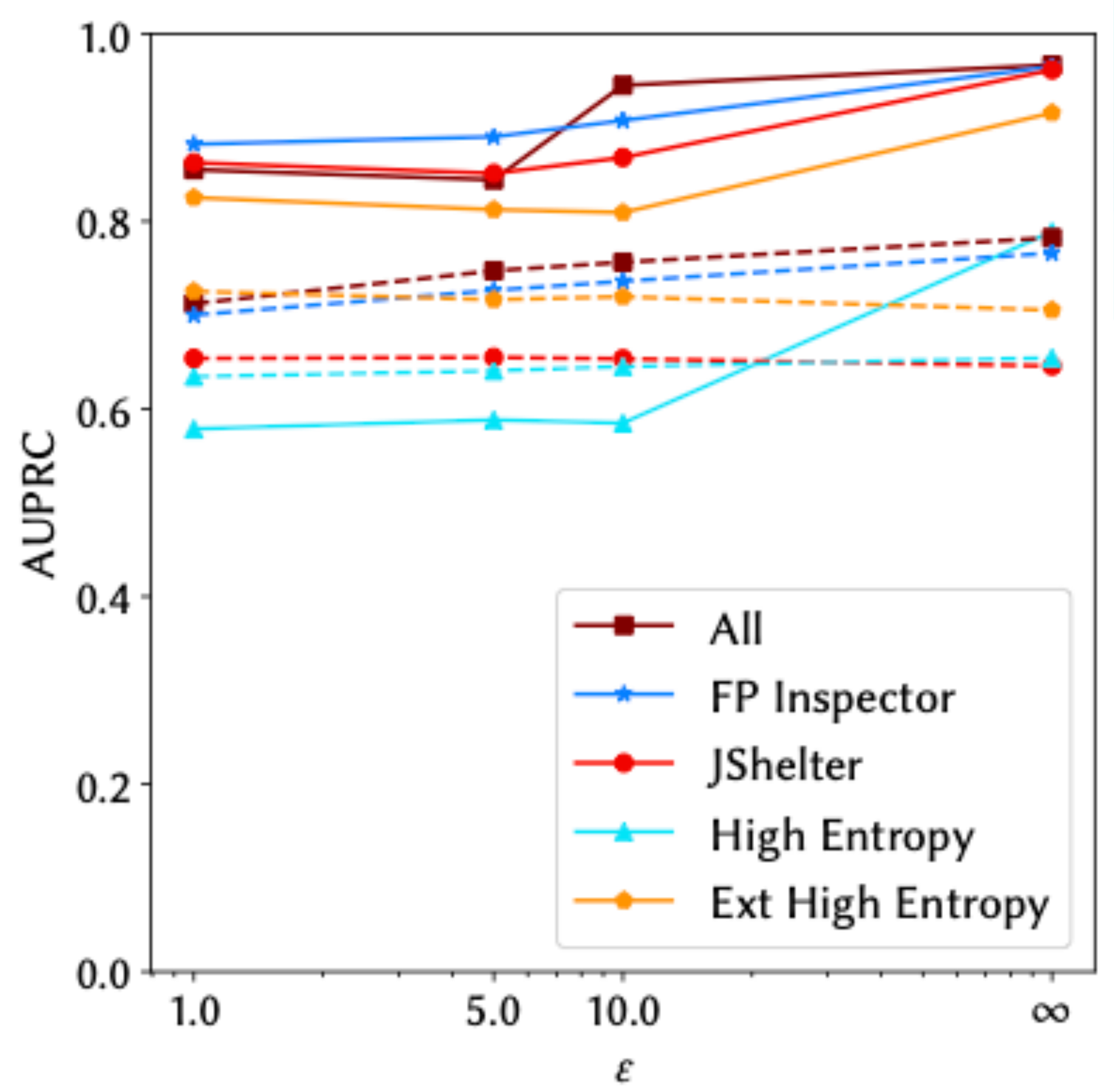
➡ **40 additional features (API call counts + custom) are optimal**



FP-Fed

Impact of DP-FedNorm

- Dotted line ⇒ Train with DP-FedAvg but **without DP-FedNorm**
- First work to consider using **differentially private feature pre-processing** in federated setting



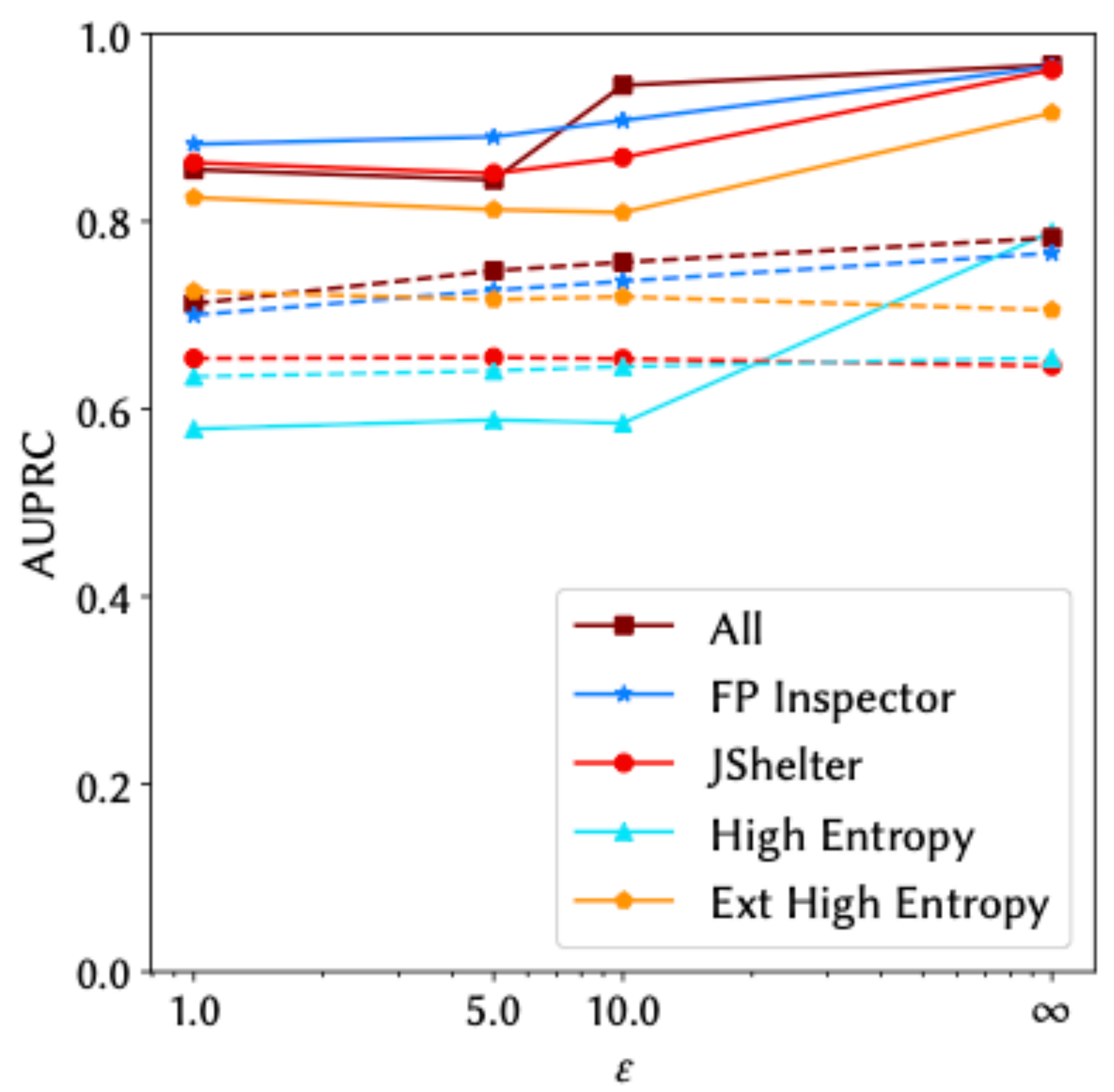
FP-Fed

Impact of DP-FedNorm

- Dotted line ⇒ Train with DP-FedAvg but **without DP-FedNorm**

- First work to consider using **differentially private feature pre-processing** in federated setting

➡ **DP-FedNorm improves model-performance by up to 20.8%**



Conclusion

Conclusion
Limitations

Conclusion

Limitations

1. Simulated distributed setting

Conclusion
Limitations

1. Simulated distributed setting

2. Large performance drop at high levels of privacy

Conclusion

Limitations

1. Simulated distributed setting

2. Large performance drop at high levels of privacy

3. Real world considerations

Conclusion

Limitations

1. Simulated distributed setting

2. Large performance drop at high levels of privacy

3. Real world considerations

4. Ground truth heuristic

Conclusion

Summary

- Introduced FP-Fed: Applying DP-FL to solve problem of detecting browser fingerprinting
  - Does not require automated crawl
  - Efficient system: Minimal feature set (149 API Call counts + custom features)
  - Acceptable utility with strong privacy guarantees (AUPRC > 0.8 at $\varepsilon = 1$)

Credits

- Icons are made by Andrean Probowo, iconixar, Freepik, Jesus Chavarria, Iconfromus, Uniconlabs, Paul J., zafdesign, setiawanap, srip, manshagraphics, Good Ware, and IdeaGrafc from www.flaticon.com

Thank you!