Federated Analytics-Empowered Frequent Pattern Mining for Decentralized Web 3.0 Applications

Zibo Wang*, Yifei Zhu*‡, Dan Wang§, and Zhu Han¶
*UM-SJTU Joint Institute, Shanghai Jiao Tong University
‡Cooperative Medianet Innovation Center (CMIC), Shanghai Jiao Tong University
§Department of Computing, The Hong Kong Polytechnic University
¶Department of Electrical and Computer Engineering, University of Houston
Email: wangzibo@sjtu.edu.cn, yifei.zhu@sjtu.edu.cn, csdwang@comp.polyu.edu.hk, zhan2@uh.edu

IEEE INFOCOM 2024 - IEEE Conference on Computer Communications | 979-8-3503-8350-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/INFOCOM52122.2024.10621294

***Abstract*—The emerging Web 3.0 paradigm aims to decentralize existing web services, enabling desirable properties such as transparency, incentives, and privacy preservation. However, current Web 3.0 applications supported by blockchain infrastructure still cannot support complex data analytics tasks in a scalable and privacy-preserving way. This paper introduces the emerging federated analytics (FA) paradigm into the realm of Web 3.0 services, enabling data to stay local while still contributing to complex web analytics tasks in a privacy-preserving way. We propose FedWeb, a tailored FA design for important frequent pattern mining tasks in Web 3.0. FedWeb remarkably reduces the number of required participating data owners to support privacy-preserving Web 3.0 data analytics based on a novel distributed differential privacy technique. The correctness of mining results is guaranteed by a theoretically rigid candidate filtering scheme based on Hoeffding's inequality and Chebychev's inequality. Two response budget saving solutions are proposed to further reduce participating data owners. Experiments on three representative Web 3.0 scenarios show that FedWeb can improve data utility by ∼25.3% and reduce the participating data owners by ∼98.4%.**

***Index Terms*—Web 3.0, federated analytics, frequent pattern mining, differential privacy, decentralized applications**

I. Introduction

When federated analytics meets Web 3.0. Web 3.0 is believed to be the next generation of the Internet. Compared to previous Web 2.0 driven by dominant Internet application providers (sometimes termed *tech giants*), Web 3.0 starts a revolution in the storage, sharing, and profiting of personal data [1], [2]. In Web 2.0, these data are collected by the applications and utilized to create profit for the tech giants, resulting in privacy leakage and unfairness in data profiting. Web 3.0 advocates are wishing to resolve the issues by the decentralized nature of Web 3.0. In Web 3.0, only the user (data owner) can access and modify the private data, because these data are stored in the user devices, or encrypted by the users. Other entities can access the private data only when they are permitted by the data owner. In this setting, user privacy is properly handled, and the strongest privacy can be achieved when the data owner never grants permission to the data. In 2022, the market size of Web 3.0 reached $1.73 billion [3].

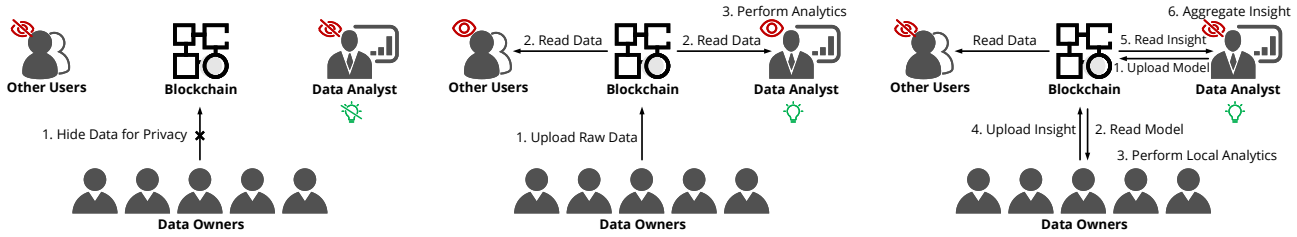
Although Web 3.0 has utilized its blockchain infrastructure to accomplish transparency, auditability, and data ownership management, privacy-preserving web data analytics is still a missing piece to complete the Web 3.0 ecosystem. In modern web applications, data analytics acts an important role in many tasks, like recommendation systems and service monitoring [18]. Analyzing data from vast data owners, if not misused, benefit both service providers and service users. These data analytics applications undoubtedly will continue to be a cornerstone for Web 3.0 applications.

However, to pursue the strongest privacy preservation, data owners may hide their sensitive data. It disables the data analytics and consequently degrades the web service and hurts both the service providers and users, as demonstrated in Fig. 1(a). Alternatively, data owners can share their data by providing data access to the data analysts. But it removes the privacy preservation of the Web 3.0 scheme, as demonstrated in Fig. 1(b).

Federated analytics (FA), a paradigm for privacy-preserving data analytics, has gained great success recently [19]. In FA, the raw data are prohibited to be transmitted to any other entities, which preserves the data privacy of data owners to some extent [20], [21]. In addition, it is able to apply additional privatization techniques, like cryptography or differential privacy (DP), to obtain a stronger and more formal privacy guarantee. FA has been applied in various data-driven scenarios, where data privacy is concerned by user awareness and regulations [22]–[25]. The decentralized nature of FA naturally matches the requirement of Web 3.0 applications.

To fill in the gap between the requirement for privacy preservation and the demand for high quality web data analytics in Web 3.0, in this paper, we introduce FA to the realm of Web 3.0 services. In our design, the data owners still grant data analysts permission to the local data. However, the data owners no longer let the data analysts access their raw data. *Instead, they offer FA services to the data analysts, so that the data analytics tasks can be properly completed, while the data privacy is preserved*. The FA-assisted Web 3.0 data analytics model is illustrated in Fig. 1(c). The local

This work is supported by the National Natural Science Foundation of China (Grant No. 62302292), the National Key R&D Program of China (Grant No. 2023YFB2704400), and the Fundamental Research Funds for the Central Universities. The work of D. Wang is supported by RGC-GRF 15209220, 15200321, 15201322, from ITC via project "Smart Railway Technology and Applications" (No. K-BBY1), RGC-CRF C5018-20G, ITC ITF-ITS/056/22MX. The work of Z. Han is supported by NSF CNS-2107216, CNS-2128368, CMMI-2222810, ECCS-2302469, US Department of Transportation, Toyota and Amazon. Corresponding author: Yifei Zhu.



(a) Complete data hiding sacrifices analytics (b) Direct data sharing sacrifices privacy (c) FA-based data analytics for Web 3.0

Fig. 1. Possible statuses of privacy preservation and data analytics in Web 3.0 applications. The marks of eye indicate the availability to learn private data. The marks of lamp indicate the availability to optimize the web service via data analytics.

TABLE I
WEB 3.0 SERVICES, THEIR FPM-RELATED APPLICATIONS, AND THE CORRESPONDING FPM TASKS.

Web service	Web 3.0 example	FPM-related application	Mined data	Pattern type
Internet browser	Brave [4]	User traversal analysis [5]	Web browsing history	Sequence
Operating system	WRIO [6]	Next used app prediction [7]	Mobile app usage logs	Sequence
Social network	Steemit [8]	Trend mining [9]	Timestamped social media posts	Itemset
Online market	OpenBazaar [10]	Market basket analysis [11]	Consumer baskets	Itemset
Computer security	Web3 Antivirus [12]	Ransomware detection [13]	System logs	Sequence
Micro-blogging	Mirror [14]	Emotion analysis [15]	Emotion classfication results of posts	Itemset
Media player	OPUS [16]	Music recommendation [17]	Historical music playlists	Sequence

privacy preservation of FA makes data owners safe to share their data, while the careful design of FA achieves the high quality of data analytics and downstream web services. Since FA algorithms are mostly task-specific, to support various data analytics demands, different tailored FA algorithms should be designed and deployed.

FA-based frequent pattern mining for Web 3.0 applications. Frequent pattern mining (FPM) is one of the most important tasks that discovers all substructures (items, subsets, or subsequences) with occurrence frequencies higher than a predefined threshold among the local data of data owners. It is applied in discovering and analyzing frequently occurring web events or objects. It has wide potential applications in Web 3.0 with some examples summarized in Table I.

As a classical and important data analytics task, many FA-based FPM solutions have been proposed [25]–[29]. Although these solutions manage to conduct FPM with privacy preservation, applying them to Web 3.0 encounters extra difficulties. One of the most critical challenges introduced by the Web 3.0 scenarios is the limited scale of participating data owners. Existing FA FPM solutions require tens of millions of participating data owners, and otherwise suffer from low data utility. Such requirements can be fulfilled in Web 2.0 applications as tech giants can gather mass participants. For example, Gboard (5 billion+ downloads in GooglePlay) users are utilized by Google as FA participants [20]. However, since the early-stage Web 3.0 applications have a relatively small scale of users (around millions, taking Steemit as an example [8], [30]), this makes the existing solutions no longer valid. New FA algorithms are needed to handle FPM with limited participating data owners.

In this paper, we follow the aforementioned model to apply FA in Web 3.0 ecosystems, and propose a novel FPM solution, named FedWeb, to fulfill the need for FPM in Web 3.0 services. It leverages the interactive FA structure to formulate a query-response scheme between the data analyst and data owners, and the iterative scheme to progressively update the candidate patterns with the confidence bounding and Apriori mechanism. To reduce the required participating data owners as restricted by Web 3.0, FedWeb utilizes the novel distributed differential privacy (DDP) techniques, which provide a formal privacy guarantee and reduce the noise added to the uploads. In addition, two flexible budget saving strategies, named candidate padding and data owner reusing, are proposed to further reduce the scale of participating data owners.

Our contributions. In summary, our contributions are:

- We introduce FA into the current Web 3.0 ecosystem, so that data analytics can be properly conducted without exposing the sensitive raw data.
- We design FedWeb for privacy-preserving FPM in Web 3.0 services. We incorporate the DDP privatization scheme and design two strategies to save the response budgets, overcoming the scale limitation in existing Web 3.0 services (Section IV).
- We rigorously prove the correctness of our candidate pattern filtering scheme. It formulates two confidence bounds using compounds from the Chebyshev's inequality and the Hoeffding's inequality, providing the theoretical guarantee of the FPM results of FedWeb (Section IV).
- Experiment results on various web service scenarios reveal that FedWeb obtains a high F1 score and requires few participating data owners. It verifies the capability of FedWeb in completing Web 3.0 FPM tasks with high quality under limited available data owners (Section V).

II. Preliminary: Privacy for Web 3.0

In this section, we discuss some concepts regarding privacy criteria which are related to our design.

DP is a gold standard in the field of private data publication, owing to its assumption of strong adversary knowledge and rigid statistical guarantee [31]. Central differential privacy (CDP) is the original and most widely used version of DP. CDP typically works in scenarios where the processor who runs the data publication mechanism can access the whole database. Therefore, CDP is no longer valid in Web 3.0 scenarios, where the data should be privatized before being gathered. Local differential privacy (LDP) are applied to handle the local privacy case [25]–[27]. However, LDP requires data owners to add significant noise to the uploads, which degrades the data utility and requires more data owners.

DDP is an emerging solution to provide (Web 3.0 available) local privacy preservation, while gaining high data utility equivalent to CDP [32]. DDP is realized via two key designs:

- **Distributed noise generation**: Each data owner adds a tailored noise to its local analytics result, where the noises from all data owners sum up to satisfy CDP.
- **Secure aggregation**: Data owners encrypt the upload of each individual data owner, and the data analyst then can only learn the sum of data owner uploads.

DDP inherits the privacy preservation of DP owing to the ingenious combination of the two designs. Distributed noise generation guarantees that the data statistically satisfy CDP after being aggregated. Secure aggregation prevents the data analyst from learning unaggregated uploads by transforming them into meaningless random values with encryption, as the added noise is not enough to solely satisfy DP (only LDP-level noise can preserve privacy for individual uploads).

Our distributed noise generation scheme is based on the geometric mechanism [33], which is described as follows.

Definition 1 (Geometric mechanism)**.** *Given a parameter $\alpha \in (0,1)$, a α-geometric mechanism M is a data publication mechanism possessing on dataset S that has integer output. Given the true query result $Q(S)$, M outputs $M(S) = Q(S) + G(\alpha)$, where $G(\alpha)$ is an integer two-sided geometric random variable with the following PDF:*

$$\mathbb{P}(G(\alpha) = x) = \frac{1-\alpha}{1+\alpha}\alpha^{|x|}. \tag{1}$$

When Q is defined as summation in S and the value of every record in S is binary, the *sensitivity* of Q becomes 1. α-geometric mechanism in this case satisfies $(-\ln\alpha)$-CDP.

To utilize the geometric mechanism in distributed noise generation, G should be decomposed into multiple random variables that are added by different users respectively. Pólya random variables are utilized to realize it, shown as follows.

Definition 2 (Pólya distribution)**.** *Let $\mathcal{X}$ be a random variable following $Pólya(r,p)$ distribution with $r \in \mathbb{R}$ and $p \in (0,1)$, its PDF is given as follows.*

$$\mathbb{P}(\mathcal{X} = x) = \binom{r+x-1}{r-1} p^x (1-p)^r. \tag{2}$$

Theorem 1. *Suppose there are n distributed data owners, each data owner adds $\mathcal{X}_i - \mathcal{Y}_i$ to its upload, where $\mathcal{X}_i$ and $\mathcal{Y}_i$ are i.i.d. $Pólya(1/n, \alpha)$ variables. These variables sum up to follow two-sided geometric distribution, i.e.,*

$$\sum_{i=1}^{n} (\mathcal{X}_i - \mathcal{Y}_i) = G(\alpha). \tag{3}$$

Proof. See Theorem 5.1 of [33]. □

Definition 1 and Theorem 1 show that, by individually adding the difference of two Pólya variables, data owners can collaboratively generate a Geometric noise on the aggregated data, which satisfies the requirement of DDP. A $Pólya(r,p)$ variable $\mathcal{X}$ is generated via a Poisson-Gamma mixture [34]: first draw γ from the $Gamma(r, p/(1-p))$ distribution, and then draw $\mathcal{X}$ from the Poisson distribution with parameter γ.

III. Threat Model and Problem Formulation

Threat model. Being consistent with many previous studies [25]–[29], we consider the data analyst and data owners to be semi-honest, *i.e.*, the data analyst or each data owner executes the protocol honestly, but try to learn the data from the (other) data owners with the information they received during the protocol. In addition, we consider that a limited number of participants may collude to learn the data. Particularly, we assume there exists collusion between, at most, the data analyst and one-third of participating data owners.

Problem formulation. Our system consists of a data analyst hosting an FPM task, and a number of data owners D holding private data. The data analyst wishes to discover the frequent patterns within the local data of data owners, *i.e.*, the patterns whose frequencies are higher than a user-defined frequency threshold f. The FA design consists of two functions: an insight derivation function $\mathcal{I}$ which executes on one data owner $d \in D$, and an aggregation function $\mathcal{A}$ which aggregates the insights to derive the final output[1]. The problem of privacy-preserving FPM in Web 3.0 scenarios we try to resolve can be described as follows.

Problem 1. *Design the functions $\mathcal{I}$ and $\mathcal{A}$, where $\mathcal{I}$ executes on each participating data owner $d \in D'$, where the set of participating data owners D' is a subset of available data owners, i.e., $D' \subseteq D$. $\mathcal{A}$ takes the outputs of $\mathcal{I}$ as input, and outputs a set of discovered frequent patterns $\mathcal{F} = \mathcal{A}(\{\mathcal{I}(d) | d \in D'\})$. The algorithm design should achieve three goals:*

- *$\mathcal{F}$ should be closed to the ground truth frequent patterns $\mathcal{F}'$, i.e., high F1 score $F1(\mathcal{F}, \mathcal{F}')$ should be achieved.*[2]
- *The algorithm should execute on a small scale of participating data owners, i.e., the size of D' should be low.*
- *The output of $\mathcal{I}$ should satisfy DDP. In detail, the upload of individual data owner $\mathcal{I}(d)$ should be encrypted, and the aggregated uploads should statistically satisfy CDP.*

[1] We use d to denote both a data owner and its local data

[2] F1 score is defined by $2pr/(p+r)$, where p is precision and r is recall.

IV. Privacy-Preserving FPM Design for Web 3.0

In this section, the Web 3.0-oriented FA-based privacy-preserving FPM scheme, FedWeb, is described in detail.

The FedWeb design is iterative, where the data analysts and data owners communicate for multiple rounds until the FPM task is completed. Each round consists of three phases: 1) *candidate pattern distribution*, where the data analyst distributes possible frequent patterns to the data owners; 2) *DDP-based private response*, where the data owners respond on their received candidates based on the established DDP mechanism; 3) *response analysis*, where the data analyst gathers the uploads, analyzes the uploads, filters candidates, and generates new candidates. In addition, two flexible strategies are proposed to reduce the required data owners.

A. Phase 1: Candidate pattern distribution

In FedWeb, the data analyst maintains a pool of candidate patterns. In the first round, the "most basic" candidates, *i.e.*, those who cannot be generated by other patterns with the Apriori property, are placed into the candidate pool. For example, in frequent itemset mining tasks, all itemsets with only one item are placed into the candidate pool in the first round. The candidate pool is updated in each response aggregation phase. When the candidate pool becomes empty, the FPM task is completed and the algorithm terminates.

In the candidate distribution phase, the data analyst activates data owners to become participants in this round, and each data owner is assigned a set of candidates to respond to. Two user-defined parameters affect the procedure: the maximal number of candidates each data owner can respond to (response budget) K, and the number of responses each candidate receives in one round P. Let $\mathcal{C}_t$ denotes the candidate pool in round t, and $|\mathcal{C}_t|$ denotes its size. The data analyst should activate sufficient data owners so that each candidate in $\mathcal{C}_t$ has been assigned P data owners, while each data owner can respond to at most K candidates but cannot respond to one candidate more than once. We use N_t to denote the set of participating data owners in round t, with size $|N_t|$.

Each candidate in the candidate pool is given a unique index from 1 to $|\mathcal{C}_t|$. The index is valid throughout this round, and the participating data owners are informed of the indexes of their received candidates. We use $\mathcal{C}_t(i)$ to denote the i-th candidate.

B. Phase 2: DDP-based private response

After receiving at most K candidates from the data analyst, each data owner first checks whether the received candidates are within its local data. Then, it constructs a response vector encoding their response to all the candidates. Next, distributed noise is added to the upload to satisfy DP. Finally, the response vector is uploaded to the data analyst with secure aggregation.

Consider a data owner with index j, its local data is denoted d_j. The data owners check whether the candidates are within their local data, following the specification of FPM subproblems. If a candidate $\mathcal{C}_t(i)$ is within its local data, it is written as $\mathcal{C}_t(i) \in d_j$. Otherwise, it is $\mathcal{C}_t(i) \notin d_j$.

Algorithm 1 FedWeb: Data owner procedure

Input: Maximal candidates of each data owner: K; DDP parameter: ϵ; List of candidates (and their indexes) it should respond $\mathcal{C}$, number of global candidates: $|\mathcal{C}_t|$.
Output: Response: R

function RESPOND($K, \epsilon, \mathcal{C}$)
  $R \leftarrow |\mathcal{C}_t|$-length all-zero vector
  for $c \in \mathcal{C}$ (i_c is the global index of c) **do**
    if c is within local data **then**
      $R[i_c] \leftarrow 1$
    end if
    $R[i_c] \leftarrow R[i_c] + (\mathcal{X} - \mathcal{Y})$ ▷ Eq. (4) defines $\mathcal{X}, \mathcal{Y}$
  end for
  $\mathcal{N} \leftarrow$ randomly picked encryption neighbors
  for $x \in \mathcal{N}$ **do**
    $\mathcal{M}_x \leftarrow$ pairwise $|\mathcal{C}_t|$-length mask
      ▷ Data owner x gets an inversed mask
    $R \leftarrow R + \mathcal{M}_x$
  end for
  return R
end function

After the checking, each data owner constructs a response vector with length $|\mathcal{C}_t|$, where each entry represents the data owner's response to a candidate (with the corresponding index). Particularly, the i-th entry of the response vector will be 1 when the data owner received candidate i from the data analyst and candidate i is within its local data, or 0 otherwise.

After the construction of the raw response vector, distributed noise is added to all the received candidates, no matter whether it is within its local data. Finally, denoting the final response from data owner j as R_j, its i-th entry is calculated as follows.

$$R_j[i] = \begin{cases} 0, & \text{if } j \text{ does not receive } \mathcal{C}_t(i), \\ \mathcal{X} - \mathcal{Y}, & \text{if } j \text{ receives } \mathcal{C}_t(i) \wedge \mathcal{C}_t(i) \notin d_j, \\ 1 + \mathcal{X} - \mathcal{Y}, & \text{if } j \text{ receives } \mathcal{C}_t(i) \wedge \mathcal{C}_t(i) \in d_j, \end{cases} \tag{4}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are i.i.d. $Pólya(1/P, e^{-\epsilon/K})$ variables.

After the generation of the response vector, the data owners upload the vectors to the data analyst with secure aggregation. In this paper, we utilize the secure aggregation scheme in [35]. Each data owner communicates with other $O(\log N_t)$ data owners, and shares a pairwise mask and a secret share of the self mask with each other data owners. After that, each data owner adds the pairwise masks and the self mask to their uploads. The data analyst can learn the sum of the uploads by summing their encrypted uploads to eliminate the pairwise masks, requesting the secret shares from other data owners to eliminate the self mask, and requesting the pairwise masks from other data owners when a data owner drops out. The solution defends collusion among one-third of participating data owners and data analyst, which meets our threat model. Many other options to realize DDP are surveyed in [33].

As for now, the workloads of data owners in this round are all completed, and the procedure of data owners in each

round (insight derivation function $\mathcal{I}$) is demonstrated in Algo. 1, with part of the secure aggregation scheme omitted for the sake of clarity. The details of the omitted part to construct the response vector can be checked in [35]. The private response scheme provides a formal privacy guarantee, as shown below.

Theorem 2. *The private response scheme demonstrated in Algo. 1 satisfies ϵ-DDP for every data owner.*

Proof. The conclusion of Theorem 1 and Definition 1 shows that, the response on each candidate satisfies (ϵ/K)-CDP after it aggregates. Since each data owner responds to at most K candidates, a data owner then satisfies ϵ-DDP according to the composition theorem of DP, which concludes the proof. □

C. Phase 3: Response analysis

For a candidate c in the candidate pool, the data analyst maintains a profile consisting of three values: the sum of historical response values r_c (represented by the corresponding entry of the response vector), the number of data owners that have responded to the candidate n_c, and the number of rounds the candidate stayed in the candidate pool m_c (there is a constant relationship that $n_c = Pm_c$). After receiving the aggregated response vector, the data analyst updates the profile of every candidate in the pool. Then, the data analyst performs a two-stage analysis of the candidates: filtering existing candidates and generating new candidates.

For filtering existing candidates, the data analyst evaluates each candidate: whether the candidate can be accepted or rejected as a frequent pattern, and removed from the candidate pool. Since the scheme involves many randomized procedures, we can not guarantee the correctness of every decision. Notice that r_c/n_c is exactly an unbiased estimate of candidate frequency, and the estimation is getting more accurate when the candidate receives more responses from data owners. The decision procedure is driven by the confidence bounds: we accept/reject a candidate as a frequent pattern when there is sufficient probabilistic confidence that its true frequency is higher/lower than the FPM threshold. Otherwise, the candidate remains in the pool to receive more responses in later rounds.

We derive the final bounds as compounds concluded from the Chebyshev's inequality and the Hoeffding's inequality.

Theorem 3 (Confidence bound of frequent patterns)**.** *A candidate c with profile (r_c, n_c, m_c) is a frequent pattern (target frequency f) with confidence $(1-\eta_g)(1-\eta_s)$ when*

$$\frac{r_c}{n_c} - \sqrt{\frac{2e^{-\epsilon/K}}{2(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}} - \sqrt{\frac{\ln \eta_s}{-2n_c}} \geq f. \quad (5)$$

Proof. See Appendix A. □

Theorem 4 (Confidence bound of non-frequent patterns)**.** *A candidate c with profile (r_c, n_c, m_c) is not a frequent pattern (target frequency f) with confidence $(1-\eta_g)(1-\eta_s)$ when*

$$\frac{r_c}{n_c} + \sqrt{\frac{2e^{-\epsilon/K}}{2(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}} + \sqrt{\frac{\ln \eta_s}{-2n_c}} \leq f. \quad (6)$$

Algorithm 2 Filtering candidate procedure

Input: A candidate to be filtered: c.
Output: Response: Operation on c.

```
function FILTER(c)
    if (r_c, n_c, m_c) satisfies Theorem 3 then
        return "Remove c from P and add c to F"
    else if (r_c, n_c, m_c) satisfies Theorem 4 then
        return "Remove c from P"
    else if n_c ≥ τ and r_c/n_c ≥ f then
        return "Remove c from P and add c to F"
    else if n_c ≥ τ and r_c/n_c < f then
        return "Remove c from P"
    end if
    return "Hold c in P"
end function
```

Proof. See Appendix B. □

A decision with three options is made for each candidate pattern in the candidate pool: if its profile meets the expression in Theorem 3, it will be removed from the candidate pool and recorded as a mined frequent pattern; if its profile meets the expression in Theorem 4, it will be just removed from the candidate pool; otherwise, the candidate will remain in the candidate pool to receive more response from data owners. In particular, in order to save the response resource, a threshold τ of the maximum response each candidate can receive is set. If a candidate has $n_c \geq \tau$ in any round, it will be forced to be filtered: it will be considered as a frequent pattern if $r_c/n_c \geq f$, or a non-frequent pattern otherwise. Algo. 2 demonstrates the procedure of filtering one candidate.

After the filtering of existing candidates, the data analyst generates new candidates and places them into the candidate pool to push forward the FPM task. The generation of new candidates leverages the Apriori property of FPM, that the subpatterns of a frequent pattern must be all frequent. To generate new candidates, the data analyst checks the patterns that have been filtered as frequent patterns, and generates a new unexplored pattern as a candidate if all of its subpatterns have been filtered as frequent patterns. For example, in frequent itemset mining tasks, the data analyst will generate a new itemset $\{a, b, c\}$ when the three itemsets $\{a, b\}$, $\{a, c\}$, and $\{b, c\}$ are all filtered as frequent patterns. In frequent sequence mining tasks, a new candidate $a \to b \to c$ will be generated when $a \to b$ and $b \to c$ are filtered as frequent patterns. For frequent item mining tasks, since all the investigated items are placed in the candidate pool in the first round, no new candidate will be generated in later rounds.

D. Further enhancement on saving response budget

Although the previously presented mechanism is working, we can observe that some response budgets of data owners are still wasted. Specifically, as each data owner provides K response budgets, when the total number of candidates $|\mathcal{C}_t|$ is less than K, the participating data owners cannot use all their budgets within that round. As some budgets are wasted, the

system needs more data owners to participate, which harms the effectiveness of FedWeb in Web 3.0 scenarios. Therefore, we enhance our original design with the following two strategies, candidate padding and data owner reusing, to further reduce the data owner usage.

Candidate padding. The candidate padding strategy fills the candidate pool with the patterns that are likely to be candidates in the future, until the number of candidates becomes K. To realize it, the data analysts sort the existing candidates with their r_c/n_c profiles in decreasing order. Then, the candidates are virtually accepted as frequent patterns one by one. Once a candidate is virtually accepted, some new candidates may be generated based on the Apriori property. These new candidates are put into the candidate pool virtually. When a padding candidate becomes a real candidate in the future, its profile is synchronized.

Data owner reusing. For a data owner that fails to use up its budgets in one round, the data owner reusing strategy asks the data owner to wait for future instructions, instead of completing its participation. Since a data owner must spend its response budgets on different candidates, the awaiting data owners will be reused in later rounds when there is any new candidate that is never responded to by the data owner. The data owner will be reused to respond to more candidates until its budget is used out.

Comparison of two strategies. Data owner reusing performs better in saving response budget, which leads to less data owner usage than candidate padding. But it requires data owners to communicate with the data analyst for many rounds, and keep waiting for the instructions during the procedure. On the other hand, candidate padding only requires one-shot communication between the data analyst and data owners, where the data owner can complete their workload in one round. But it performs less effectively in saving participating data owners. Data owner reusing is promising in cases like cryptocurrency-related applications where data owners are likely to be always online, but unwilling to share their data freely. Candidate padding is desirable when the data owners frequently drop out, or cannot afford long awaiting, such as the mobile web applications.

The whole procedure of FedWeb has been discussed. The procedure of the data analyst (aggregation function $\mathcal{A}$) is presented in Algo. 3.

Algorithm 3 FedWeb: Data analyst procedure

Input: Maximal candidates of each data owner: K; DDP parameter: ϵ; Round-level responder for each candidate: P; Response number threshold: τ; Target frequency: f.
Output: Frequent patterns: $\mathcal{F}$

$\mathcal{F} \leftarrow$ an empty set
$\mathcal{P} \leftarrow$ an empty set ▷ Set of candidate patterns
Add basic patterns to $\mathcal{P}$, initialize profile r_c, n_c, m_c
while $\mathcal{P}$ is not empty (round t) **do**
  $\mathcal{C}_t \leftarrow \mathcal{P}$
  if *candidate_padding_flag* and $|\mathcal{C}_t| < K$ **then**
    Generate new candidates based on Apriori property and place them into $\mathcal{C}_t$, until $|\mathcal{C}_t| = K$
  end if
  Generate indexes for candidates in $\mathcal{C}_t$
  for $c \in \mathcal{C}_t$ **do**
    $D_c \leftarrow$ an empty set ▷ Data owners responding c
  end for
  if *data_owner_reusing_flag* **then**
    Remove saved data owners that used out budgets
    Add available saved data owners to D_c
  end if
  Add new data owners to D_c until $|D_c| = P$ for all c
  $\mathcal{R} \leftarrow |\mathcal{C}_t|$-length all-zero vector
  for each data owner d (new or saved) **do**
    $\mathcal{C}^{<d>} \leftarrow \{c \in \mathcal{C}_t | d \in D_c\}$
    $R_d \leftarrow \text{RESPOND}(K, \epsilon, \mathcal{C}^{<d>})$
    $\mathcal{R} \leftarrow \mathcal{R} + R_d$
    if *data_owner_reusing_flag* and $|\mathcal{C}^{<d>}| < K$ **then**
      Save d with its response history
    end if
  end for
  for $c \in \mathcal{C}_t$ with index i_c **do**
    $r_c \leftarrow r_c + \mathcal{R}[i_c]$; $n_c \leftarrow n_c + P$; $m_c \leftarrow m_c + 1$
  end for
  for $c \in \mathcal{P}$ **do**
    Operate on c following $\text{FILTER}(c)$
  end for
  Generate new candidates form $\mathcal{F}$ with Apriori property
  Add new candidates to $\mathcal{P}$, initialize profile r_c, n_c, m_c
end while
return $\mathcal{F}$

V. Evaluation

We evaluate our design FedWeb with existing local privacy FPM solutions in several web FPM service scenarios, to validate the capacity and effectiveness of FedWeb.

A. Experiment setting

Datasets. We evaluate FedWeb in three representative web scenarios based on real-world web datasets.

- **SteemOps (frequent item mining)**: The SteemOps dataset [30] is collected from Steemit, a representative decentralized social network application following the Web 3.0 paradigm. In our formulation, each data owner is a Steemit user, and the items in the local data are the other users the data owner has ever reposted to. By completing the frequent item mining task, the data analyst is able to discover these popular users.
- **MSNBC (frequent sequence mining)**: The MSNBC dataset [36] includes users' browsing trace of news in *msnbc.com*. In our formulation, each data owner holds the browsing history of one MSNBC user, which is essentially a sequence of websites. Mining frequent sequential patterns in the browsing trace, also called user traversal analysis, is a key application for web service analysis.
- **MovieLens (frequent itemset mining)**: The MovieLens

dataset [37] includes user's rating in *movielens.org*. In our evaluation, we formulate the local itemset data of users as the themes of movies that a MovieLens user has ever rated full score on them, and completion of the frequent itemset mining helps data analysts to analyze the correlation between different themes of movies.

Methods. In addition to our designed FedWeb, we use three benchmarks for comparison, named FedFPM, SFP, and RAPPOR. We introduce these methods respectively as follows.

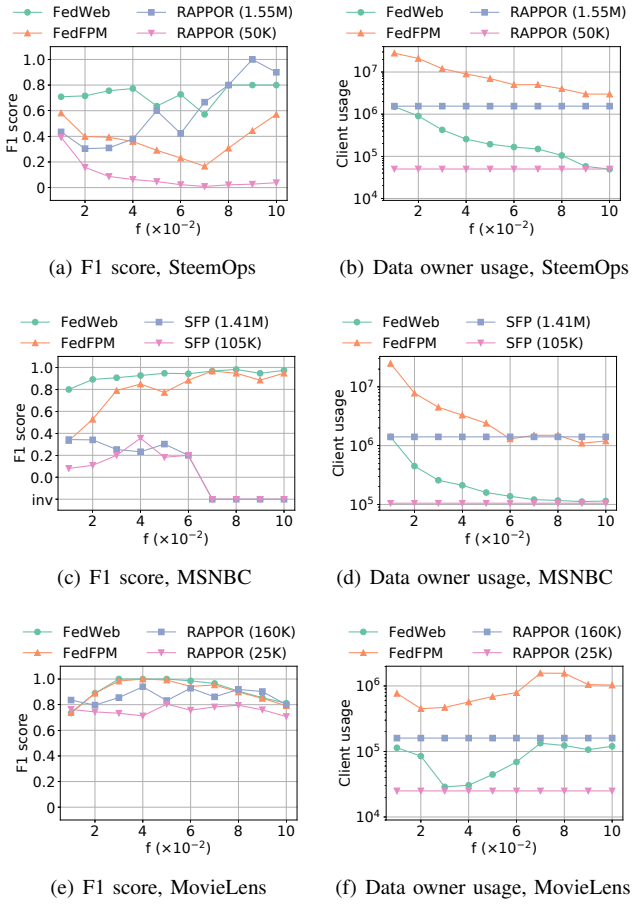
- **FedWeb** is the proposed design in this paper. By default, we apply the *data owner reusing strategy* to further reduce the number of participating data owners.
- **FedFPM** [25] is a pioneering work in the field of FA-based FPM tasks. It completes multiple FPM tasks under a unified framework. It enforces LDP via a one-bit randomized response, which provides a local privacy guarantee, but leads to a high loss of data utility.
- **SFP** [27] is proposed by Apple to discover frequently used words (sequences of letters) from user keyboard inputs. SFP can only handle the sequence data, and is therefore evaluated in the MSNBC dataset only.
- **RAPPOR** [26] is proposed by Google to gather crowd user uploads with an LDP guarantee. It is originally designed for frequent item mining. We modify RAPPOR by encoding all the itemsets as distinct items to solve the frequent itemset mining task. RAPPOR is evaluated in SteemOps and MovieLens datasets.

Metrics. Under specific DP requirement ϵ, the performance of an FPM algorithm is determined by the data utility (correctness of FPM result, represented by F1 score) and the total number of participating data owners. An effective algorithm should gain high F1 scores using few data owners.

Parameters. We set the target frequencies of FPM to be between 0.01 and 0.1. The number of responses each candidate receives in each round is set to $P = 1,000$. The maximal candidate of each data owner is set to $K = 50$. The DP parameters of the schemes are all set to $\epsilon = 2.0$. The confidence levels used in response analysis are set to $\xi_g = 0.01, \xi_s = 0.01$. The remaining parameters of FedFPM, SFP, and RAPPOR all follow the default settings in their original papers. As SFP and RAPPOR should predefine the total data owners, we set two values of the data owner number for each RAPPOR/SFP setting: one is slightly higher than the maximal usage of FedWeb, and another is slightly lower than the minimum usage of FedWeb. The first setting represents the case when FedWeb does not gain an unfair advantage by utilizing more data owners, and the latter setting investigates whether the first setting makes the data owner numbers unreasonably high.

B. Experiment results

Performance of FedWeb and benchmarks. We execute the FPM algorithms in the three scenarios, and record their F1 score of finally mined patterns, and the total number of participating data owners. The results in three datasets are shown in Fig. 2. FedWeb outperforms RAPPOR, SFP, and



(a) F1 score, SteemOps (b) Data owner usage, SteemOps

(c) F1 score, MSNBC (d) Data owner usage, MSNBC

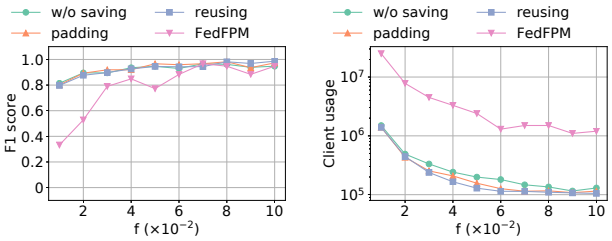
(e) F1 score, MovieLens (f) Data owner usage, MovieLens

Fig. 2. Performance of FedWeb and the benchmarks in three datasets. The legends of RAPPOR and SFP mark their numbers of participating data owners. We mark invalid ("inv") if no frequent pattern is output from an algorithm.

FedFPM in F1 scores, even if they utilize more data owners than FedWeb. By averaging the F1 score over the ten target frequencies, FedWeb achieved an improvement in F1 score of at least 25.3%, 17.3%, and 1.2% in SteemOps, MSNBC, and MovieLens datasets, respectively. FedWeb also performs outstandingly in reducing the required data owners. Compared to FedFPM, FedWeb saved 81.1%∼98.4% of participating data owners. Lastly, in Figs. 2(e) and 2(a), the decrease of RAPPOR performance with the data owner numbers shows that RAPPOR is still data hungry, and the comparison we make is fair for RAPPOR. On the other hand, the performance of SFP is extremely low, no matter what the data owner number is. It indicates the ineffectiveness of SFP in our tasks.

Performance of response budget saving strategies. To test the effects of the two response budget saving strategies in Section IV-D, we evaluate the vanilla FedWeb, candidate padding, and data owner reusing in the MSNBC scenario.[3] The results are shown in Fig. 3. Compared to the vanilla FedWeb without further response budget saving, the candidate padding strategy can reduce the total participating data owners

[3]The experiment results similar to Figs. 3 and 4 conducted in other datasets, which also support our analysis, are omitted due to space limit.



(a) F1 score (b) Data owner usage

Fig. 3. Performance of FedWeb with different response budget saving strategies in MSNBC dataset. The candidate padding and data owner reusing strategies are abbreviated as "padding" and "reusing", respectively.

(a) F1 score (b) Data owner usage ($\times 10^5$)

Fig. 4. Performance of FedWeb under different ϵ and K in MSNBC dataset.

by 6.3%∼30.1% (16.2% on average); the data owner reusing strategy can reduce by 6.9%∼36.9% (21.6% on average); and neither of the strategies affects F1 score. Data owner reusing strategy can obtain a better result in saving participating data owners, but it requires data owners to await for multiple rounds. In our experiments, a data owner under the data owner reusing strategy should participate in 3.60 rounds on average. In addition, FedWeb is able to remarkably outperform FedFPM, even without any response budget saving strategy.

Parametric study. The DDP parameter ϵ, and the maximum candidate of each data owner K are a pair of important parameters that greatly affect the performance of FedWeb. Theoretically, a larger ϵ/K leads to a better data utility, because it determines the noise added to each candidate response, and a larger K leads to fewer data owner usage, because fewer data owners are required when each data owner responds to more candidates. We conduct experiments on the MSNBC datasets with target frequency $f = 0.01$ under different ϵ and K settings, and summarize the resulting F1 scores and data owner usages in Fig. 4. By analyzing the F1 score results in Fig. 4(a), we verify our theoretical analysis, that a similar F1 score can be achieved when the value of ϵ/K is fixed, and the F1 score is higher when ϵ/K is higher. In Fig. 4(b), we found that when ϵ is fixed, a higher K can reduce the total required data owners. The parameter K is an important handler for data analysts: when the privacy requirement ϵ is given, K can be used to tradeoff data utility and data owner usage, where a higher data utility can be achieved by reducing K, and a smaller data owner usage can be achieved by increasing K.

VI. Related Work

Data analytics in Web 3.0. While the Web 3.0 paradigm emerges, researchers start to place interest in digesting great value from the data generated by Web 3.0. In these works, data from decentralized applications [30], [38] or cryptocurrency blockchains [39]–[41] are gathered for further data analytics or public dataset release. However, they exactly fetch the public data stored in the blockchain, following the least privacy scheme shown in Fig. 1(b). Compared to the previous works, this paper presents the first work to introduce FA as a building block in privacy-preserving Web 3.0 data analytics, achieving formal privacy guarantees for web data analytics tasks.

Federated analytics. FA is first introduced by Google to perform data analytics without exposing raw data [20]. As for now, FA algorithms are task-specific, where each FA algorithm can be applied to one (or a class of) data analytics problems. The FA model has been applied in various classical and important data analytics tasks, including heavy hitter discovery [22], [42], set computation [43], sample mean/median estimation [24], [44], clustering [23], and FPM [25]. In this paper, we also advance the federated FPM studies by proposing a scalable FA design. The idea of utilizing FA for data analytics for Web 3.0 also sheds light on new application scenarios for FA studies.

Privacy-preserving FPM. There exists a line of works studying FPM under privacy preservation. The early study focuses on resolving one subproblem of FPM, including frequent item mining [26], [28], frequent itemset mining [29], and frequent sequence mining [27]. FedFPM [25] proposes the first FA framework to unify all FPM subproblems and is the most related work to our problem. However, compared with FedFPM, FedWeb is Web 3.0-oriented with a special focus on reducing the participating data owners. It utilizes DDP that achieves a formal privacy guarantee without significant data utility loss, derives complex confidence bounds to filter candidates to adapt DDP, and proposes flexible budget saving strategies to further reduce participating data owners.

VII. Conclusion

In this paper, we address the problem of privacy-preserving data analytics on local sensitive data in Web 3.0 systems. We propose a reform to the current Web 3.0 data publication scheme, namely, instead of transmitting raw sensitive data, data owners provide FA service to the data analyst. Based on the task-specific FA paradigm, we further design a novel solution for privacy-preserving FPM in Web 3.0 scenarios. The proposed FedWeb mechanism judiciously incorporates DDP into its response scheme with theoretically-sound candidate generation/filtering mechanisms and flexible response budget saving strategies, to achieve reliable privacy preservation, high data utility, and, what is important for Web 3.0, fewer participating data owners. Experiment results show that, our solution can achieve $\sim 25.3\%$ higher F1 score and $\sim 98.4\%$ fewer consumed data owners compared to the benchmarks.

APPENDIX A
PROOF OF THEOREM 3

Noticing that $\mathcal{X}-\mathcal{Y}$ is with expectation 0, r_c is expected to be increased by 1 if the responding data owner possesses the candidate c, and 0 otherwise. Therefore, r_c/n_c is an unbiased estimation of the true frequency of c (denoted f_c).

There are exactly two sources of random error on r_c/n_c: the **sampling error** caused by drawing the data owners to respond on a candidate, and the **geometric error** caused by the geometric noise of the uploads. Denote the two errors as $\mathcal{E}_s$ and $\mathcal{E}_g$, respectively. The relationship between our estimation and the true candidate frequency can be modeled as follows.

$$\frac{r_c}{n_c} = f_c + \mathcal{E}_s + \mathcal{E}_g. \tag{7}$$

To start with, we bound the magnitude of $\mathcal{E}_g$. $\mathcal{E}_g$ is introduced by the two-sided geometric noise (constructed by distributed Pólya noises) which is added once in each round. Therefore, m_c noises are added into r_c, where each of them follows the distribution in (1) with $\alpha = e^{-\epsilon/K}$.

The variance of the added two-sided geometric noise added in each round is calculated.

$$Var(G(\alpha)) = \frac{2\alpha}{(1-\alpha)^2}. \tag{8}$$

By analyzing the essence of $\mathcal{E}_g$, we know that it is

$$\mathcal{E}_g = \frac{\sum^{m_c} G(\alpha)}{n_c} = \frac{\sum^{m_c} G(\alpha)}{Pm_c} = \frac{\sum^{m_c} \frac{G(\alpha)}{P}}{m_c}, \tag{9}$$

exactly the average of m_c random variables following the distribution $\frac{G(\alpha)}{P}$. The variance of $\frac{G(\alpha)}{P}$ is, obviously, $\frac{2\alpha}{P^2(1-\alpha)^2}$. We then utilize the Chebyshev's inequality [45] to bound $\mathcal{E}_g$, the average of i.i.d. random variables with known variance.

Theorem 5 (Chebyshev's inequality for deviation of random variables average)**.** *Let $X_1, X_2, ..., X_n$ be i.i.d. random variables with expectation μ and variance $Var(X)$, we have*

$$\mathbb{P}\left(\frac{\sum_{i=1}^n X_i}{n} - \mu \geq x\right) \leq \frac{Var(X)}{2nx^2} \tag{10}$$

for any $x > 0$.

By applying $\mathcal{E}_g$ into (10), we have

$$\mathbb{P}(\mathcal{E}_g \geq x) \leq \frac{2\alpha}{2(1-\alpha)^2 P^2 m_c x^2}. \tag{11}$$

We set η_g as the allowed error rate for geometric error bounding, and recall $\epsilon/K = -\ln\alpha$ to satisfy ϵ-DP, (11) can be transformed into

$$\mathbb{P}\left(\mathcal{E}_g \geq \sqrt{\frac{2e^{-\epsilon/K}}{2(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}}\right) \leq \eta_g. \tag{12}$$

Then, we try to bound $\mathcal{E}_s$, the error caused by the sampling effect. The bounding is based on an important observation that, $f_c + \mathcal{E}_s$, is equivalent to the frequency of candidate c on a sample of n_c data owners. As a result, $f_c + \mathcal{E}_s$ is an average of n_c random variables, where each variable takes two values 0 or 1 (Bernoulli).

Based on the observation, we use the Hoeffding's inequality [45] to bound the divergence between $f_c + \mathcal{E}_s$ and f_c.

Theorem 6 (Hoeffding's inequality)**.** *Let $X_1, X_2, ..., X_n \in [0,1]$ be i.i.d. random variables with expectation μ, we have*

$$\mathbb{P}\left(\frac{\sum_{i=1}^n X_i}{n} - \mu \geq x\right) \leq \exp(-2x^2 n) \tag{13}$$

for any x.

By considering $f_c + \mathcal{E}_s$ as the average of n_c 0-1 bounded variables, we have

$$\mathbb{P}(f_c + \mathcal{E}_s - f_c \geq x) = \mathbb{P}(\mathcal{E}_s \geq x) \leq \exp(-2x^2 n_c). \tag{14}$$

We set η_s as the allowed error rate for sampling error bounding. Eq. (14) can be transformed into

$$\mathbb{P}\left(\mathcal{E}_s \geq \sqrt{\frac{\ln \eta_s}{-2n_c}}\right) \leq \eta_s. \tag{15}$$

By summarizing (12) and (15), we can derive a confidence of $(1-\eta_s)(1-\eta_g)$ that both the expression in (12) and (15) do not hold. After that, we can use (7) derive a bound of f_c with confidence $(1-\eta_s)(1-\eta_g)$ that

$$\begin{aligned}\mathbb{P}\Bigg(f_c \geq \frac{r_c}{n_c} - \sqrt{\frac{2e^{-\epsilon/K}}{(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}} - \sqrt{\frac{\ln \eta_s}{-2n_c}}\Bigg) \\ \geq (1-\eta_s)(1-\eta_g).\end{aligned} \tag{16}$$

Eq. (16) can be transformed into a confidence bound of c being a frequent pattern when the bound of f_c (RHS of the first line of (16)) is larger than f, which concludes the proof.

APPENDIX B
PROOF OF THEOREM 4

The proof of Theorem 4 follows the same procedure as Appendix A. A reversed version of (12) can be derived by applying the reversed version of the Chebyshev's inequality.

$$\mathbb{P}\left(\mathcal{E}_g \leq -\sqrt{\frac{2e^{-\epsilon/K}}{2(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}}\right) \leq \eta_g. \tag{17}$$

Similarly, a reversed version of (15) can be derived by applying the reversed version of the Hoeffding's inequality.

$$\mathbb{P}\left(\mathcal{E}_s \leq -\sqrt{\frac{\ln \eta_s}{-2n_c}}\right) \leq \eta_s. \tag{18}$$

By summarizing (17) and (18), considering the joint case that both expressions do not hold, and substituting into (7), we can derive a bound similar to (16):

$$\begin{aligned}\mathbb{P}\Bigg(f_c \leq \frac{r_c}{n_c} + \sqrt{\frac{2e^{-\epsilon/K}}{(1-e^{-\epsilon/K})^2 P^2 m_c \eta_g}} + \sqrt{\frac{\ln \eta_s}{-2n_c}}\Bigg) \\ \geq (1-\eta_s)(1-\eta_g).\end{aligned} \tag{19}$$

Eq. (19) can be transformed into a confidence bound of c not being a frequent pattern when the bound of f_c is smaller than f, which concludes the proof.

REFERENCES

[1] Liberal Democratic Party of Japan, "Japan's nft strategy for the web 3.0 era," https://www.taira-m.jp/Japan\%27s\%20NFT\%20Whitepaper_E_050122.pdf, 2023.

[2] Ethereum Organization, "Introduction to web3," https://ethereum.org/en/web3/, 2023.

[3] Grand View Research, "Web 3.0 blockchain market size, share & trends analysis report by blockchain type, by application (cryptocurrency, conversational ai, data & transaction storage), by end use, by region, and segment forecasts, 2023 - 2030," https://www.grandviewresearch.com/industry-analysis/web-3-0-blockchain-market-report#, 2023.

[4] Brave Software, Inc., "Brave," https://brave.com/, 2023.

[5] L. Sun and X. Zhang, "Efficient frequent pattern mining on web logs," in *Proc. Asia-Pacific Web Conf.*, Hangzhou, China, Apr. 2004, pp. 533–542.

[6] WRIO Ltd., "WRIO Internet OS," https://webrunes.com/, 2022.

[7] E. H.-C. Lu, Y.-W. Lin, and J.-B. Ciou, "Mining mobile application sequential patterns for usage prediction," in *Proc. IEEE Int. Conf. Granular Comput.*, Hokkaido, Japan, Oct. 2014, pp. 185–190.

[8] Steemit, Inc., "Steemit," https://steemit.com/, 2023.

[9] P. N. Nohuddin, F. Coenen, R. Christley, C. Setzkorn, Y. Patel, and S. Williams, "Finding "interesting" trends in social networks using frequent pattern mining and self organizing maps," *Knowl.-Based Syst.*, vol. 29, no. 1, pp. 104–113, May 2012.

[10] OpenBazaar Team, "OpenBazaar," https://github.com/OpenBazaar/openbazaar-desktop/, 2021.

[11] J. P. B. Saputra, S. A. Rahayu, and T. Hariguna, "Market basket analysis using fp-growth algorithm to design marketing strategy by determining consumer purchasing patterns," *J. Appl. Data Sci.*, vol. 4, no. 1, pp. 38–49, Jan. 2023.

[12] Web3 Antivirus, "Web3 Antivirus: Protecting your digital assets," https://web3antivirus.io/, 2022.

[13] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, and R. Khayami, "Know abnormal, find evil: frequent pattern mining for ransomware threat hunting and intelligence," *IEEE Trans. Emerg. Topics Comput.*, vol. 8, no. 2, pp. 341–351, Sep. 2017.

[14] Reflective Technologies, Inc., "Mirror," https://mirror.xyz/, 2022.

[15] M. P. Skenduli, M. Biba, C. Loglisci, M. Ceci, and D. Malerba, "Mining emotion-aware sequential rules at user-level from micro-blogs," *J. Intell. Inf. Syst.*, vol. 57, no. 2, pp. 369–394, Jun. 2021.

[16] Opus Foundation, "Web 3.0 designed for artists," https://opusfoundation.medium.com/web-3-0-designed-for-artists-f33e5d5036fb, 2022.

[17] N. Hariri, B. Mobasher, and R. Burke, "Context-aware music recommendation based on latenttopic sequential patterns," in *Proc. ACM Conf. Recommender Syst.*, New York, USA, Sep. 2012, pp. 131–138.

[18] J. Han, J. Pei, and H. Tong, *Data mining: concepts and techniques*. Morgan kaufmann, 2022.

[19] D. Wang, S. Shi, Y. Zhu, and Z. Han, "Federated analytics: Opportunities and challenges," *IEEE Network*, vol. 36, no. 1, pp. 151–158, 2021.

[20] Google AI, "Federated analytics: Collaborative data science without data collection," https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html, 2020.

[21] P. Kairouz, B. McMahan, and V. Smith, "Federated learning and analytics: Industry meets academia," https://sites.google.com/view/fl-tutorial/home, 2021.

[22] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li, "Federated heavy hitters discovery with differential privacy," in *Proc. Int. Conf. Artif. Intell. Statist.*, Palermo, Italy, Jun. 2020, pp. 3837–3847.

[23] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Int. Conf. Mach. Learn.*, virtual event, Jul. 2021, pp. 2611–2620.

[24] G. Cormode and I. L. Markov, "Bit-efficient numerical aggregation and stronger privacy for trust in federated analytics," *arXiv*, 2021.

[25] Z. Wang, Y. Zhu, D. Wang, and Z. Han, "Fedfpm: A unified federated analytics framework for collaborative frequent pattern mining," in *Proc. IEEE Int. Conf. Comput. Commun.*, virtual event, May 2022.

[26] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proc. ACM Conf. Comput. Commun. Secur.*, Scottsdale, USA, Nov. 2014, pp. 1054–1067.

[27] Apple differential privacy team, "Learning with privacy at scale," https://machinelearning.apple.com/research/learning-with-privacy-at-scale, 2017.

[28] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Vienna, Austria, Oct. 2016, pp. 192–203.

[29] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *Proc. IEEE Symp. Secur. Privacy*, San Francisco, CA, May 2018, pp. 127–143.

[30] C. Li, B. Palanisamy, R. Xu, J. Xu, and J. Wang, "Steemops: Extracting and analyzing key operations in steemit blockchain-based social media platform," in *Proc. ACM Conf. Data Appl. Secur. Privacy*, virtual event, 2021, pp. 113–118.

[31] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory Cryptography Conf.*, New York, USA, Mar. 2006, pp. 265–284.

[32] E. Bagdasaryan, P. Kairouz, S. Mellem, A. Gascón, K. Bonawitz, D. Estrin, and M. Gruteser, "Towards sparse federated analytics: Location heatmaps under distributed differential privacy with secure aggregation," *Proc. Privacy Enhancing Technol.*, vol. 4, no. 1, pp. 162–182, 2022.

[33] S. Goryczka and L. Xiong, "A comprehensive comparison of multiparty secure additions with differential privacy," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 5, pp. 463–477, Sep. 2015.

[34] N. L. Johnson, S. Kotz, and A. W. Kemp, *Univariate discrete distributions*. John Wiley & Sons, 2005.

[35] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly) logarithmic overhead," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, virtual event, Nov. 2020, pp. 1253–1269.

[36] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, "Visualization of navigation patterns on a web site using model-based clustering," in *Proc. ACM Int. Conf. Knowl. Discovery Data Mining*, Boston, MA, Aug. 2000, pp. 280–284.

[37] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 1–19, Jan. 2015.

[38] M. Li, J. Zhu, T. Zhang, C. Tan, Y. Xia, S. Angel, and H. Chen, "Bringing decentralized search to decentralized services," in *Proc. USENIX Symp. Oper. Syst. Des. Implementation*, virtual event, Jul. 2021, pp. 331–347.

[39] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A fistful of bitcoins: characterizing payments among men with no names," in *Proc. Conf. Internet Meas. Conf.*, Barcelona, Spain, Oct. 2013, pp. 127–140.

[40] M. Spagnuolo, F. Maggi, and S. Zanero, "Bitiodine: Extracting intelligence from the bitcoin network," in *Proc. Financial Cryptography Data Secur.*, Christ Church, Barbados, Mar. 2014, pp. 457–468.

[41] M. Möser, R. Böhme, and D. Breuker, "Towards risk scoring of bitcoin transactions," in *Proc. Financial Cryptography Data Secur.*, Christ Church, Barbados, Mar. 2014, pp. 16–32.

[42] J. Acharya and Z. Sun, "Communication complexity in locally private distribution estimation and heavy hitters," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, USA, Jun. 2019, pp. 51–60.

[43] B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai, "Spot-light: lightweight private set intersection from sparse ot extension," in *Proc. Annu. Int. Cryptology Conf.*, Barbara, USA, Aug. 2019, pp. 401–431.

[44] J. Böhler and F. Kerschbaum, "Secure multi-party computation of differentially private median," in *Proc. USENIX Conf. Secur. Symp.*, virtual event, Aug. 2020, pp. 2147–2164.

[45] D. Wang and Z. Han, *Sublinear algorithms for big data applications*. Springer, 2015.