# A Method to Anonymize Business Metrics to Publishing Implicit Feedback Datasets

YOSHIFUMI SEKI, Gunosy Inc., Japan
TAKANORI MAEHARA, RIKEN Center for Advanced Intelligence Project, Japan

This paper shows a method for building and publishing datasets in commercial services. Datasets contribute to the development of research in machine learning and recommender systems. In particular, because recommender systems play a central role in many commercial services, publishing datasets from the services are in great demand from the recommender system community. However, the publication of datasets by commercial services may have some business risks to those companies. To publish a dataset, this must be approved by a business manager of the service. Because many business managers are not specialists in machine learning or recommender systems, the researchers are responsible for explaining to them the risks and benefits.

We first summarize three challenges in building datasets from commercial services: (1) anonymize the business metrics, (2) maintain fairness, and (3) reduce the popularity bias. Then, we formulate the problem of building and publishing datasets as an optimization problem that seeks the sampling weight of users, where the challenges are encoded as appropriate loss functions. We applied our method to build datasets from the raw data of our real-world mobile news delivery service. The raw data has more than 1,000,000 users with 100,000,000 interactions. Each dataset was built in less than 10 minutes. We discussed the properties of our method by checking the statistics of the datasets and the performances of typical recommender system algorithms.

CCS Concepts: • **Information systems → Recommender systems**.

Additional Key Words and Phrases: datasets, recommender systems

**ACM Reference Format:**
Yoshifumi Seki and Takanori Maehara. 2020. A Method to Anonymize Business Metrics to Publishing Implicit Feedback Datasets. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 22–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3383313.3412256

## 1 INTRODUCTION

### 1.1 Background

Publicly available datasets have contributed to the development of machine learning and recommender systems research. Arguably, the most prominent example is ImageNet [9], which has made significant contributions in rapidly developing deep learning technology. The MovieLens dataset [12] played a similar role in the recommender system area. Every researcher interested in recommender systems would likely agree that MovieLens is one of the large factors supporting the development of recommendation system research. Building and publishing datasets are widely recognized as important contributions to the research community.

Publishing datasets benefits both researchers and publishers. Because researchers want to study more realistic data, they hope to use datasets from commercial services, such as e-commerce, video streaming, and social network sites.


Manuscript submitted to ACM

Commercial services can also receive benefits by publishing datasets. If the published datasets are used in research papers, the company that published the datasets will be able to deploy the latest, state-of-the-art algorithms to its service [3].

In this study, we focus on datasets consisting of implicit feedback [19], which is the collection of user interactions with items such as clicking on items, bookmarking a page, or listening to a song. Many practical problems in commercial services are targeted for implicit feedback. Also, explicit feedback, which includes explicit input by users, such as ratings and user reviews, is not always available. In recent years, some implicit feedback datasets by commercial services have been published (Section 5), but these are still not enough. There are a variety of services, and their trends change rapidly; thus, more diverse and fresh datasets are needed. We expect more implicit feedback datasets to be published by commercial services for the further development of recommendation system studies.

However, it is difficult to publish implicit feedback datasets in a commercial service as a public dataset because there are several business risks. Implicit feedback datasets are built using user behavior logs, which include confidential business information and users' personal information. Even though the privacy information is protected (e.g., by anonymization), leaking confidential business information and some reputation risks are still remaining. Note that explicit feedback datasets will also have similar issues, but, for now, these are often constructed by crawling public sources, such as user reviews and ratings available online [18]. Thus, these are currently recognized as less risky than implicit feedback datasets.[1]

Because of the existence of these business risks, before publishing an implicit feedback dataset in commercial services, researchers must get approval by a business manager of the service. Because many business managers are not specialists in machine learning or recommender systems, the researchers should be responsible for explaining the risks and benefits. However, the process of approval and an appropriate explanation is typically not known by those that have access to and use a public dataset. For example, many datasets from commercial services may be sampled according to a strategy to protect against those risks; but these strategies are not published. These challenges are important not only for publishing a dataset but also for providing a dataset in joint research with academics.

### 1.2 Contribution

In this study, we summarize the challenges of building and publishing a dataset in a commercial service and propose a method that resolves these challenges by solving an optimization problem. Note that existing studies for publishing datasets focused on protecting privacy information [7, 24]; however, we here focus on reducing business risks.

The challenges are the following. The first challenge is to keep business metrics confidential. Because user behavior logs may include some confidential business information, business managers are wary of potential information leaks by publishing the dataset. The second challenge is in the fairness of the dataset. In recent years, a problem emerges wherein products using machine learning technology might have embedded racism or stereotypes. A company needs to take care that their datasets do not promote such biases. A third challenge is the popularity bias. This challenge is not a risk, but it is important for commercial services to eventually generate more revenue by publishing a dataset. Recent studies point out that many recommendation algorithms have a popularity bias in that popular items are easily recommended. To address these challenges, we aim to publish a dataset securely and in a more appropriate manner.

The contributions of this study are as follows:

[1] This situation may change in the near future due to the GDPR (General Data Protection Regulation).

- We summarize the challenges of building and publishing datasets from commercial services: (1) anonymize the business metrics, (2) maintain fairness, and (3) reduce the popularity bias. (Section 2)
- We formulate the problem of building and publishing a dataset as an optimization problem that seeks the sampling weight of users, where the challenges are encoded as appropriate loss functions. (Section 3)
- We applied our method to build datasets from the raw data of our real-world mobile news delivery service, Gunosy[2], which is a popular news delivery service in Japan. The raw data has more than 1,000,000 users with 100,000,000 interactions. Each dataset was built in less than 10 minutes. We discussed the properties of our method by observing the statistics of the datasets and the results of typical recommender system algorithms. (Section 4)

The implementation of our proposed method and a dataset built by our proposed method are public in github[3].

## 2 TASK AND CHALLENGES

In this section, we pose the problem of building and publishing an implicit feedback dataset from a commercial service. To make the explanation concrete, we here use our mobile news delivery service, SERVICENAME, which is a popular news service in COUNTRYNAME, as an example. The service collects news articles from many news media and provides lists of selected articles to users, like Google news[4] and Bing news[5].

### 2.1 Task

Although real-world recommender systems have various kinds of data, we only focus on the following three data to simplify the situation.

**User behavior logs.** When user $u$ clicks article $a$ at time $t$, the triplet $(u, a, t)$ is recorded as a log.
**User attributes.** Each user has attributes, such as age and gender.
**Article category.** Each news article has a category, such as sports, entertainment, and politics.

During this paper, we always refer to "dataset" as the dataset built from the user behavior logs. Our task is to publish a subset of the user behavior logs as a dataset. We do not publish user attributes because, even if we anonymize user IDs, there is a risk of de-anonymizing the users from the attributes [23]. We also do not publish article categories to protect contents rights.

Here, we only consider the "sampling user" approach that first samples users and then collects all the user behavior logs associated with the sampled users. This is because if we take the "sampling behavior log" approach that directly samples logs, some parts of the consumption histories of the users will be missing. Such a dataset cannot be used for the task of predicting the next items that the users will consume, which is one of the typical tasks in recommender systems.

### 2.2 Challenges

We aim to publish a dataset by reducing business risks. To publish a dataset, this must be approved by a business manager of the service. Because many business managers are not specialists in machine learning or recommender systems, the researchers are responsible for explaining the risks and benefits.

[2]https://gunosy.com/
[3]https://github.com/gunosy/publishing-dataset-recsys20
[4]https://news.google.com/
[5]https://www.bing.com/news

We pose the following three challenges to the dataset: (1) anonymize the business metrics, (2) maintain the fairness, and (3) reduce the popularity bias. The following sections explain why we think these challenges are important. We will propose a mathematical formulation for these challenges in the next section.

*2.2.1 Anonymize the Business Metrics.* Companies who want to publish datasets, of course, do not want to disclose their confidential business metrics, such as the operating income, the average number of clicks, and the number of active users. The business manager of the services is afraid of leaking confidential business information. Hence, it must be ensured that these metrics cannot be estimated from the dataset.

A straight-forward way to build a dataset is to select users in a uniformly random manner and collect their user behavior logs (in the following, we refer to this approach as the "uniform sampling approach"). However, in this approach, some business metrics, such as the average number of clicks of users and the average active rate of users, could be easily estimated by aggregating the randomly sampled user behavior logs. These metrics suggest important information for understanding the profit structure of the business; thus, these should not be disclosed unintentionally.

According to the above observations, to conceal these business metrics, we must sample users with a non-uniform distribution. However, no existing studies exploit what distribution should be used to conceal the business metrics. Existing implicit datasets from commercial services may conceal these metrics by non-uniform sampling; however, they did not state what distributions have been employed.

*2.2.2 Maintain Fairness.* Recently, fairness of machine learning models is one of the important topics in the machine learning community [15]. Often, a real-world dataset contains some bias, and if we train a machine learning model with a biased dataset, the obtained model will also suffer from the bias. Therefore, many existing studies consider how to reduce the biases in such situations.

Here, we point out that publishing a fair dataset is very important, especially if the dataset comes from a commercial service. As described in Section 2.1, commercial service datasets hardly contain user attributes; hence the existing methods (e.g., [30]) that maintain fairness using user attributes that cannot be applied. This means that publishing an unfair dataset indirectly contributes to creating unfair machine learning models. In addition, any user of the dataset would not be aware if the model suffered from such a bias until it caused some serious problem in production. Such a risk will damage the company's reputation.

*2.2.3 Reduce Popularity Bias.* One motivation of dataset publication is making profit by developing good recommender systems. For this purpose we attempt to reduce the popularity bias. Popularity bias is based on the problem of the performance of a recommender system that depends on the accuracy of the popular items in a dataset [2]. Recommender systems are expected to match long-tailed items with users; thus, algorithms suffering the popularity bias cannot achieve their role. Although popularity bias is usually considered an algorithmic problem, we believe that it is also a problem in building datasets.

In this study, we focus on the popularity bias in the article category. Many services using recommender systems assign categories to items, and the popularity of these categories varies widely. For example, in a news service, the entertainment category is more popular than other categories. If the dataset is built by the uniform sampling, the items of unpopular categories are less frequently sampled. Some existing algorithm [29] tries to calibrate the evaluation of the popular items; however, the desired solution increases the number of interactions with unpopular items. Because researchers cannot increase the number of interactions, the publisher must keep a certain amount of interactions with unpopular category items.

## 3 MATHEMATICAL FORMULATION

In this section, we formulate our task and challenges, presented in Section 2, as an optimization problem. We then propose an algorithm to solve the problem.

We build our dataset as follows. Let $U$ be the set of users in the raw data, and we specify $m \ll |U|$ as the number of users in the building dataset. For each user $u \in U$, we assign a nonnegative weight $w(u) \in \mathbb{R}_+$. Then, we sample $m$ users without replacement, where user $u$ is included in the samples with probability proportional to $w(u)$. The dataset is now built by collecting all the behavior logs of the sampled users.

Under the above process, we formulate our task as a problem of finding the weight $w \in \mathbb{R}^U_{\geq 0}$ such that the dataset clears all the challenges described in Section 2.2 in expectation. We represent our three challenges of anonymizing business metrics, maintaining fairness, and removing popularity bias as loss functions $L_{\text{clicks}}(w)$, $L_{\text{attributes}}(w)$, and $L_{\text{categories}}(w)$, respectively, and minimize the weighted sum of the loss functions.

In the following, we define the loss functions by the distance between some distributions in the dataset and some desired target distributions. Here, we make the following approximation:

$$\text{Weighted sampling } \textit{without} \text{ replacement} \approx \text{Weighted sampling } \textit{with} \text{ replacement.} \tag{1}$$

This is because the distribution of the weighted sampling with replacement is complicated [14], so it is not suitable as an objective function of the optimization problem. Theoretically, this approximation is valid when the sample size $m$ is much smaller than the whole size $|U|$, and the weight is $O(1/|U|)$ [4]; these conditions are satisfied in our experiments in Section 4.

In the following, for any $x \in \mathbb{R}^d_{\geq 0}$ with $x \neq 0$, we denote by $\overline{x} \in \mathbb{R}^d_{\geq 0}$ the vector whose entries are normalized to one, that is,

$$\overline{x} = \frac{x}{\sum_{i=1}^d x_i}. \tag{2}$$

### 3.1 Definition of $L_{\text{click}}$ — Anonymize the Business Metrics

We want to anonymize our business metrics, such as the average number of clicks and the number of active users. However, formulating this challenge is impossible because it needs to enumerate all the metrics that we should anonymize. Instead, we take the following approach.

We observe that several important metrics are strongly correlated with the distribution of the number of clicks. For example, the average number of clicks are immediately derived by taking the expectation over this distribution. Also, if this distribution is long-tailed then there are many active users. Thus, we assume that our business metrics are anonymized if the distribution of the number of clicks in the dataset is different from the one in the raw data. Under this assumption, our loss function is defined as follows.

Let $L$ be the maximum number of clicks of the users, and let $A_{\text{click}} \in \mathbb{R}^{L \times U}$ be the matrix such that

$$(A_{\text{click}})_{k,u} = \begin{cases} 1, & \text{user } u \text{ clicked articles } k \text{ times,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then, $\overline{A_{\text{click}}w} \in \mathbb{R}^L_{\geq 0}$ approximates the expected click distribution on the dataset. Let $p_{\text{click}} \in \mathbb{R}^L$ be the target distribution. Then, our loss function is given by

$$L_{\text{click}}(w) = W(\overline{A_{\text{click}}w}, p_{\text{click}}), \tag{4}$$

where $W$ is the Wasserstein distance on the real line with the $\ell_1$ distance, which is explicitly written as follows [27]:

$$W(p_1, p_2) = \sum_{k=1}^{V} \left| \sum_{i=1}^{k} (p_1(i) - p_2(i)) \right|. \tag{5}$$

### 3.2 Definition of $L_{\text{attributes}}$ — Maintain Fairness

We fix a sensitive user attribute $G$; for example, $G = \{\text{male}, \text{female}\}$ for gender. Let $A_{\text{attribute}} \in \mathbb{R}^{G \times U}$ be the matrix such that

$$(A_{\text{attribute}})_{g,u} = \begin{cases} 1, & \text{user } u \text{ has attribute } g, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Then, $\overline{A_{\text{attribute}}w}$ approximates the expected attribute distribution on the dataset. Let $p_{\text{attribute}} \in \mathbb{R}^G$ be the target distribution defined on $G$. To maintain the fairness, $p_{\text{attribute}}$ should be the uniform distribution on $G$. Then, our loss function is given by

$$L_{\text{attribute}}(w) = D(\overline{A_{\text{attribute}}w}, p_{\text{attribute}}), \tag{7}$$

where $D$ is the KL divergence defined by

$$D(p_1, p_2) = \sum_{g \in G} p_1(g) \log \frac{p_1(g)}{p_2(g)}. \tag{8}$$

### 3.3 Definition of $L_{\text{category}}$ — Reduce Population Bias

The loss function, $L_{\text{category}}$, is defined similarly to $L_{\text{attribute}}$. Let $C$ be the set of categories of the articles, e.g., $C = \{\text{sport}, \text{entertainment}, \dots\}$. Let $A_{\text{category}} \in \mathbb{R}^{C \times U}$ be the matrix such that

$$(A_{\text{category}})_{c,u} = \begin{cases} t, & t \text{ times user } u \text{ clicked articles of category } c, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Then, $\overline{A_{\text{category}}w}$ is the expected category distribution on the dataset. Let $p_{\text{category}} \in \mathbb{R}^C$ be the target distribution defined on $C$. To sample a certain amount of interactions from unpopular category items, $p_{\text{category}}$ should be the uniform distribution on $C$. Then, our loss function is given by

$$L_{\text{category}}(w) = D(\overline{A_{\text{category}}w}, p_{\text{category}}), \tag{10}$$

where $D$ is the KL divergence defined by (8).

### 3.4 Optimization

We find a weight vector $w \in \mathbb{R}^U_{\geq 0}$ at which all the loss functions have small values. To achieve this goal, we minimize the sum of these loss functions as follows:

$$\min_{w \in \mathbb{R}^U_{\geq 0}} L(w) := \alpha_{\text{click}} L_{\text{click}}(w) + \alpha_{\text{attribute}} L_{\text{attribute}}(w) + \alpha_{\text{category}} L_{\text{category}}(w), \tag{11}$$

where $\alpha_{\text{click}}$, $\alpha_{\text{attribute}}$, and $\alpha_{\text{category}}$ are hyperparameters that specify the balance of them; they are tuned by looking at the distributions on the dataset.

We apply the gradient descent-type algorithm (e.g., gradient descent with momentum [21]) to minimize the loss function. Here, the gradient of the loss function is easily obtained by the automatic differentiation. Note that this objective function is non-convex due to the normalization; thus, the gradient descent has no theoretical guarantee to converge on the optimal solution.

## 4 EXPERIMENTS

In this section, we apply our proposed method to our in-house dataset to demonstrate how to construct a public dataset. We implemented our algorithm in Python 3.8 with PyTorch [20][6]. We used a single Amazon EC2 t2.xlarge instance for the computation.

### 4.1 Experimental Setting

We built a dataset from the raw data in our news delivery service. The raw data consisted of the user behavior logs in a certain duration, users' genders (male or female) as their attributes, and the articles' categories (11 categories including sports, entertainment, and so on.)[7] The number of users in the raw data was more than 1,000,000, and the number of clicks in the raw data was more than 100,000,000. From this raw data, we built datasets that consisted of the behavior logs of 60,000 users. Note that we only publish a subset of the user behavior logs as a dataset, and the users' genders and the articles' categories are not published as parts of the dataset.

We built eight datasets, 1CC, 1UC, 1CU, 1UU, 2CC, 2CU, 2UC, and 2UU, as shown in Table 1. "Zipf($s$)" in the #Clicks column indicates that the target distribution $p_{\text{click}}$ for the click distribution follows Zipf's distribution of exponent $s$, in other words, $p_{\text{click}}(k) \propto 1/k^s$. Note that the click distribution of the raw data is closer to Zipf(1) than Zipf(2). For both cases, we set $\alpha_{\text{click}} = 1$. The symbol "o" in *Attributes* column indicates $\alpha_{\text{attribute}} = 5$ and $p_{\text{attribute}}$ follows the uniform distribution; otherwise $\alpha_{\text{attribute}} = 0$, (i.e., there is no loss for the attributes). Similarly, "o" in *Categories* column indicates $\alpha_{\text{category}} = 10$ and $p_{\text{population}}$ follows the uniform distribution; otherwise $\alpha_{\text{category}} = 0$. Table 1 also contains the statistics of the datasets in its four rightmost columns, which are discussed below.

### 4.2 Properties of Datasets

*4.2.1 Overview.* The statistics of the datasets are shown in Table 1. #Users is the number of users, #Interactions is the number of interactions, and #Items is the number of items. These datasets were built so that #Users is 60,000. In this study, #Interactions show the number of clicks, and #Items shows the number of news articles. Density is represented by #Interactions/(#Item × #Users), and this metric shows the sparsity of each dataset.

[6] https://github.com/gunosy/publishing-dataset-recsys20

[7] User IDs and article IDs were changed from the master databases by a hashing algorithm, and the timestamp was normalized by some criteria. Thus, no one could match these interactions in the published datasets to the master databases.

Table 1. Statistics of constructed datasets.

| Name | #Clicks | Attributes | Category | #Users | #Interactions | #Items | Density |
|---|---|---|---|---|---|---|---|
| 1UU | Zipf(1) | | | 60,000 | 5,406,392 | 29,219 | 0.00308 |
| 1CU | Zipf(1) | o | | 60,000 | 5,372,623 | 29,054 | 0.00308 |
| 1UC | Zipf(1) | | o | 60,000 | 4,211,982 | 29,671 | 0.00236 |
| 1CC | Zipf(1) | o | o | 60,000 | 4,255,355 | 29,740 | 0.00238 |
| 2UU | Zipf(2) | | | 60,000 | 301,293 | 13,377 | 0.00037 |
| 2CU | Zipf(2) | o | | 60,000 | 318,525 | 13,406 | 0.00039 |
| 2UC | Zipf(2) | | o | 60,000 | 148,337 | 11,940 | 0.00020 |
| 2UU | Zipf(2) | o | o | 60,000 | 146,711 | 11,965 | 0.00020 |

The volume of #Interactions is extremely different among Zipf(1) and Zipf(2) — #Interactions of Zipf(1) is 10 times more than that of Zipf(2). This difference was caused because Zipf(2) samples more users with fewer clicks. Each metric did not change significantly in the datasets constrained by gender. The datasets constrained by the article categories (which were 1UC, 1CC, 2UC, and 2CC) had a smaller #Interactions than the other datasets, and the decrease of #Interactions was larger in Zipf (2) than in Zipf (1). Also, #Items in Zipf(1) did not change significantly with the category constraint, whereas Zipf(2) decreased. Under the constraint of Zipf(2), the articles that could be sampled to satisfy the category constraint were limited.

*4.2.2 Business Metrics.* Figure 1 shows the cumulative densities of the number of clicks of the datasets. All the datasets with the same target distribution, in other words, (i.e., Zipf($s$), $s \in \{1, 2\}$), followed the same distribution; these are completely overlapped. This means that we successfully controlled the click distribution of the datasets, and hence, the business metric can be anonymized.

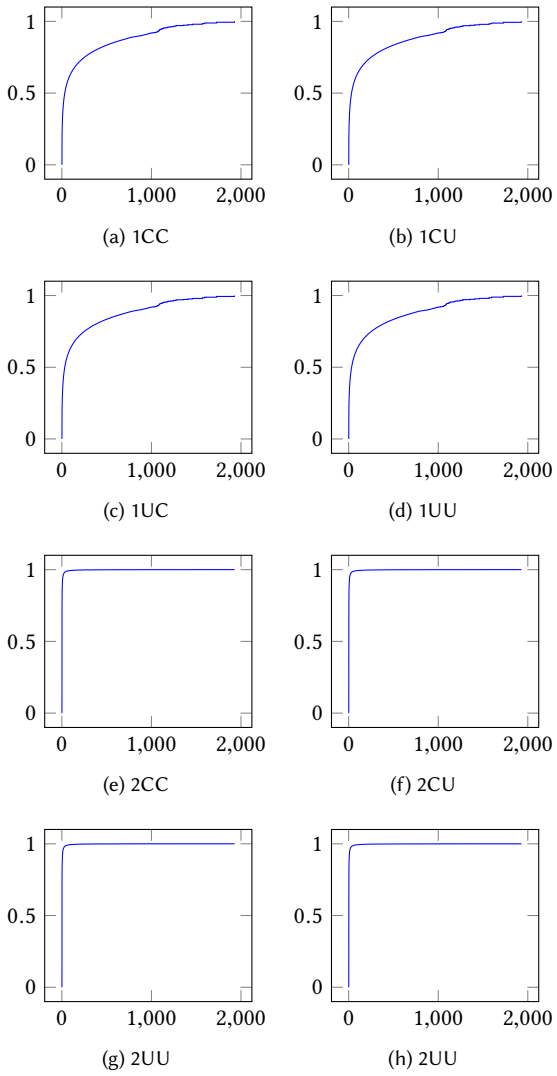
*4.2.3 Fairness.* Figure 2 shows the fraction of the genders in the dataset. All four gender-controlled datasets had balanced distributions of genders, whereas the remaining four gender-uncontrolled datasets did not. This means that, in our raw data, gender was easy to balance.

*4.2.4 Population Bias.* Figure 3 shows the fraction of the article categories in the dataset. In general, the category-controlled datasets were more uniform than the category-uncontrolled datasets. However, the two category-controlled datasets with Zipf(1) still had unbalanced category distributions. This may be because Zipf(2) tended to have more balanced distributions — this was observed by comparing {1CU, 1UU} and {2CU, 2UU}.

### 4.3 Timestamp Distributions

Our method manipulated the distributions of the clicks, user attributes, and the categories. Hence, we are interested in how other statistics were modified from the raw data. Here, we observe the distributions of the timestamp of the clicks, shown in Figure 4.

In general, the distributions of the datasets did not differ significantly from the raw data. In particular, the datasets with Zipf(1) had more similar distributions to the raw data than the datasets with Zipf(2). This may because the distribution of the raw data was more similar to Zipf(1).




(a) 1CC (b) 1CU

(c) 1UC (d) 1UU

(e) 2CC (f) 2CU

(g) 2UU (h) 2UU

Fig. 1. Cumulative densities of the number of clicks; the $x$-axis is for the number of clicks, and $y$-axis is the cumulative densities. The first row is for Zipf(1). and the second row is for Zipf(2).

### 4.4 Comparison of Algorithm Evaluations

Next, we confirm how the evaluations of our eight datasets differed. In this experiment, our purpose was not to maintain the evaluation results of the recommendation algorithm. If the results were very different for each dataset, we should choose the sampling conditions carefully. Also, if the results had some patterns or trends, we might be able to select sampling conditions to achieve a specific purpose.

*4.4.1 Comparing Algorithms.* We compared the results with six well-established algorithms.

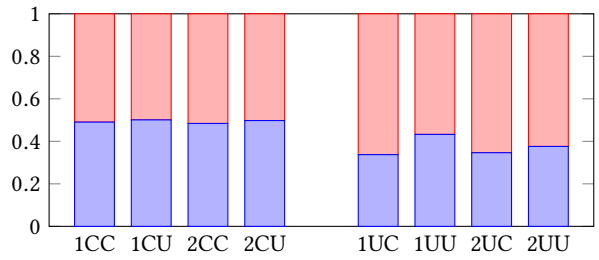



Fig. 2. Distribution of user gender in the datasets. The colors correspond to gender (male or female); we do not disclose the correspondence. The left four datasets were gender-controlled, and the right four datasets were not.

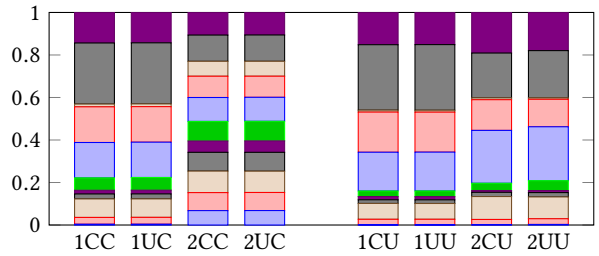



Fig. 3. Distribution of the categories of the clicked articles in the datasets. The colors correspond to the categories (sport, entertainment, etc); we do not disclose the correspondence. The left four datasets were category-controlled, and the right four datasets were not.

- **Random**: A method that selected items randomly.
- **TopPopular**: A method that selected the most popular items. Popularity is the number of clicks.
- **Item-KNN** [5]: A traditional collaborative filtering (CF) approach based on k-nearest-neighbor (KNN) and item–item similarities. The method contains two parameters, neighborhood size $\in \{50, 100, 200\}$ and a regularization term $\in \{16, 32, 64\}$[8].
- **BPR-MF** [22]: A pairwise ranking model based on matrix factorization. his is used as a strong CF-based algorithm in many studies [8, 28].
- **Item2vec** [1]: A model inspired from word2vec. We train the model using the SGNS (Skip-Gram Negative Sampling) [16]. The method contains a negative sampling size $\in \{5, 10, 15, 20\}$, and window size $\in \{3, 7, 12, 15\}$[6].
- **GRU4REC** [13]: A sequence prediction model using GRU (Gated Recurrent Unit). This model is expected to capture long-term user preferences.

In addition, BPR-MF, Item2Vec, and GRU4REC have the same parameters, which are vector dimension $\in \{10, 100, 300\}$, learning ratio $\in \{0.001, 0.01, 0.1, 1.0\}$, and batch size $\in \{128, 256, 512\}$.

*4.4.2 Experiment Settings and Metrics.* This experiment followed the setting of one-leave-out, which is one of the popular settings in recommender algorithm evaluation. This setting splits the history of item consumption to the last item and others; the model predicts the last consumption item using other items. To evaluate this setting, 20 items were selected randomly, excluding items which were not consumed by the user, and these items along with the test item of the user were ranked by the scoring of the model. Also, we evaluated this generated ranking by nDCG@10 (normalized

Table 2. Comparison of the algorithm NDCG@10 evaluations for each datasets

| | #Click | Attributes | Category | Random | TopPopular | Item-KNN | Item2vec | BPR-MF | GRU4REC |
|---|---|---|---|---|---|---|---|---|---|
| uniform | | - | | 0.219 | 0.338 | 0.58 | 0.581 | 0.43 | 0.429 |
| 1UU | Zipf(1) | | | 0.218 | 0.289 | 0.552 | 0.583 | 0.418 | 0.392 |
| 1CU | Zipf(1) | o | | 0.221 | 0.291 | 0.549 | 0.592 | 0.442 | 0.451 |
| 1UC | Zipf(1) | | o | 0.219 | 0.223 | 0.518 | 0.584 | 0.415 | 0.423 |
| 1CC | Zipf(1) | o | o | 0.219 | 0.205 | 0.52 | 0.583 | 0.402 | 0.415 |
| 2UU | Zipf(2) | | | 0.219 | 0.283 | 0.413 | 0.333 | 0.349 | 0.437 |
| 2CU | Zipf(2) | o | | 0.213 | 0.306 | 0.412 | 0.331 | 0.364 | 0.337 |
| 2UC | Zipf(2) | | o | 0.201 | 0.260 | 0.408 | 0.315 | 0.344 | 0.344 |
| 2CC | Zipf(2) | o | o | 0.217 | 0.264 | 0.438 | 0.332 | 0.335 | 0.346 |

Table 3. Comparison of the algorithm HR@10 evaluations for each datasets

| | #Click | Attributes | Category | Random | TopPopular | Item-KNN | Item2vec | BPR-MF | GRU4REC |
|---|---|---|---|---|---|---|---|---|---|
| uniform | | - | | 0.479 | 0.420 | 0.792 | 0.874 | 0.780 | 0.863 |
| 1UU | Zipf(1) | | | 0.478 | 0.359 | 0.771 | 0.893 | 0.787 | 0.778 |
| 1CU | Zipf(1) | o | | 0.487 | 0.362 | 0.770 | 0.903 | 0.800 | 0.843 |
| 1UC | Zipf(1) | | o | 0.485 | 0.284 | 0.736 | 0.882 | 0.765 | 0.823 |
| 1CC | Zipf(1) | o | o | 0.480 | 0.258 | 0.743 | 0.885 | 0.769 | 0.812 |
| 2UU | Zipf(2) | | | 0.485 | 0.446 | 0.671 | 0.601 | 0.690 | 0.736 |
| 2CU | Zipf(2) | o | | 0.484 | 0.473 | 0.678 | 0.614 | 0.731 | 0.737 |
| 2UC | Zipf(2) | | o | 0.454 | 0.433 | 0.648 | 0.656 | 0.673 | 0.695 |
| 2CC | Zipf(2) | o | o | 0.479 | 0.457 | 0.683 | 0.589 | 0.674 | 0.711 |

discounted cumulative gain) and HR@10 (hit ratio), which are well-established ranking metrics for recommendation tasks.

In this experiment, we selected interactions from datasets by a certain time range. Because the value of items rapidly decreases according to times in news domain. If the time range of the dataset is broad, it is difficult to predict a user's next click item because it depends on when the user launches the application.

To train and tune each model, the dataset was split into a training set and a validation set. As a validation set, we placed a penultimate item of the userâĂŹs history as a target item set and items before that as input items[28]. As a training set, we placed the third item from the end of the userâĂŹs history as a target item, and the items before that as input items. Since four or more items are required for training in user consumption history, users who had four or more clicks were selected from the dataset. Many studies set this threshold to 20, so this setting was closer to the problem of cold-start settings.

*4.4.3 Result.* Table 2 and Table 3 shows the result of algorithm comparison. The result shows that Zipf(1) and Zipf(2) have different trends; so we describe the difference in detail.

In Zipf(1), Item2vec records the best performance of all the algorithms in both metrics. Item-KNN was second in NDCG, and GRU4REC was second in HR. When a category constraint was added, the performance of Item-KNN is decreased in both metrics, whereas the performance of Item2vec is not. Surprisingly, in both metrics, TopPopular lost against Random under the category and gender constraints. When the gender constraint was added, the performance did not change significantly. Although there was a small difference, the trends were roughly similar to the uniform.

The datasets under Zipf(2) had different results. In NDCG, GRU4REC achieved the best performance except both the attribute and category are constrained, whereas Item-KNN achieved the best if both are constrained. In HR, GRU4REC was the best for all the datasets with Zipf(2). In Item2Vec, the performance of Zipf(2) was worse than that of Zipf(1). This may because the datasets of Zipf(2) were very sparse. In Item-KNN and GRU4REC, the impact of deterioration was small, so these algorithms may have some resistance to sparsity. In Zipf(2), the constraint of gender also made a big difference.

These results suggest that the performance of the algorithms differed in how the datasets were built. In this study, we do not imply which parameter is better or worse. Considering these properties, it is necessary to select appropriate conditions.

### 4.5 Discussion

Our proposal method determines sampling probability under three challenges, which are business metric anonymization, maintaining fairness, and popularity bias reduction. In this experiment, business metric anonymization is most affected by the property of the dataset. It is unclear that this result is common in other datasets; however, we think there may be no significant difference.

The difference between the datasets strongly affects the performances of algorithms. The performance on Zipf(1) is similar to the uniform. This may be because Zipf(1) has more similar distribution to raw data than Zipf(2). The datasets with Zipf(2) are more sparse than those with Zifp(1). In other words, Zipf(2) will sample more cold-start users, so the dataset with Zipf(2) may contribute to improving the cold-start problem. It is necessary to select sampling settings according to the purpose, and it may be important to publish datasets with various settings.

## 5 RELATED WORKS

### 5.1 Dataset Publications

The challenge publishing datasets have been discussed mainly anonymization. These studies focus on how defense the de-anonymization attack that combines published datasets with other resources, such as other public datasets, crawled web pages, and social hackings[8] [7, 23]. For example, [17] demonstrated that an adversary who knows only a little bit about an individual subscriber could easily identify this subscriber's record in the dataset using IMDb[9]. Because our dataset consisted only of interactions excluding user attributes and item information, these de-anonymization techniques cannot be applied. However, besides the de-anonymization risks, there are some challenges publishing datasets, and we summarize and solve these challenges.

Recently, there have been some public implicit feedback datasets from commercial services. Several data science competitions, such as Kaggle[10], the Recsys Challenges[11], and the KDD Cup[12], held using implicit feedback in commercial

[8] e.g. Acquire some information by asking directly.
[9] https://www.imdb.com/
[10] https://www.kaggle.com/
[11] https://recsys.acm.org/recsys20/challenge/
[12] https://www.kdd.org/kdd2020/kdd-cup

services, such as clicks on web pages and advertisements[13] and music playback[14], and their datasets have been published. Also, a research group from Grenoble University has published two implicit feedback datasets, KASANDR[26] and PANDOR[25], from commercial services. However, they did not disclose how to build these datasets.

The most famous case of dataset publication by a commercial service is the Netflix Prize [3]. The Netflix Prize is a competition offering one million dollars to the first individual or team to develop a recommendation system capable of predicting movie ratings with at least 10% greater accuracy than Cinematch, the companyâĂŹs existing system. In this competition, to protect users' privacy and confidential business information, Netflix made this statement [11]:

> to prevent certain inferences being drawn about the Netflix customer base, some of the rating data for some customers âĂę [has] been deliberately perturbed in one or more of the following ways: deleting ratings; inserting alternative ratings and dates; and modifying rating dates. (Netflix Prize Rules, n.d.)

Thus, Netflix was trying to avoid business and privacy risks.

Competitions in Kaggle have similar concerns as demonstrated, for example, in the following statement for the Outbrain Click Prediction[15]:

> Please remember that participants are prohibited from de-anonymizing or reverse engineering data or combining the data with other publicly available information.

Although not specified, the dataset itself was probably processed to avoid risks.

Thus, when the company publishes the dataset, they take care of business and privacy risks. However, this process is not shared as common knowledge, so each dataset has to be carefully considered on an individual basis. In this study, we aim to define requirements to build an appropriate public dataset and to promote recommender system research in this area.

### 5.2 Mathematical Formulation

Our mathematical formulation has been motivated by Fukuchi et al. [10]. They considered the problem of finding fair subsets of (possibly) unfair datasets such that the sampled subset is indistinguishable from the original dataset for the purpose of deceiving auditors. They formulated their problem as a minimization problem of Wasserstein distance between the samples and the original dataset subject to fairness criteria.

We also formulated our dataset building problem as Wasserstein and KL-divergence minimization problem. There are some significant differences. First, they minimized the general Wasserstein distance, whereas we only minimized the Wasserstein distance on the real line with the $\ell_1$ distance. Second, they represented fairness as a hard constraint, whereas we represented it as a soft constraint by the KL divergence from the uniform distribution. These differences affect the scalability of the algorithms. Their algorithm cannot be applied to the input of a size of more than 10,000, whereas our algorithm successfully applied to the raw data of a size of more than 1,000,000.

## 6 CONCLUSION

Datasets contribute to the development of research in machine learning and recommender systems. Implicit datasets in commercial services are more important to recommender system studies; however, publishing datasets is difficult due

[13] https://www.kaggle.com/c/outbrain-click-prediction
[14] https://www.aicrowd.com/challenges/spotify-sequential-skip-prediction-challenge
[15] https://www.kaggle.com/c/outbrain-click-prediction

to some business risks. Contrary to this importance, this difficulty has not been discussed and its solution has not been proposed.

We first summarized three challenges in building datasets from commercial services: (1) anonymize the business metrics, (2) maintain fairness, and (3) remove the popularity bias. Then, we formulated the problem of building and publishing the dataset as an optimization problem that seeks the sampling weight of users, where the challenges are encoded as appropriate loss functions. We applied our method to build datasets from the raw data of our real-world mobile news delivery service. The raw data has more than 1,000,000 users with 100,000,000 interactions. Each dataset was built in less than 10 minutes. We discussed the properties of our method by checking the statistics of the datasets and the performances of typical recommender system algorithms.

Because this study is the first attempt to reduce business risks in publishing datasets, there are several important future directions. First, this study proposed a method to anonymize the business metrics; however, we did not give a theoretical guarantee if the impossibility of the estimation. Proving such an impossibility is an important problem. Second, this study only considered the user-item interactions as the behavior logs. However, real-world services may have different types of behavior logs, such as dwell time in web pages and playtime in video streaming. The extension of our method to such a situation is a promising future work.

Previously, researchers has not disclosed how to build the dataset and has not shared the knowledge with the community. We hope that our work will lead to more discussions on the process of building and publishing datasets and that many datasets will be published.

## REFERENCES


[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *Proceedings of the IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP'16)*. IEEE, 1–6.

[2] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical Biases in Information Retrieval Metrics for Recommender Systems. *Information Retrieval Journal* 20, 6 (2017), 606–634.

[3] James Bennett, Stan Lanning, et al. 2007. The Netflix Prize. In *Proceedings of KDD Cup and Workshop*, Vol. 2007. 35.

[4] Vladimir Braverman, Rafail Ostrovsky, and Gregory Vorsanger. 2015. Weighted sampling without replacement from data streams. *Inform. Process. Lett.* 115, 12 (2015), 923–926.

[5] John S. Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*. 43âĂŞ52.

[6] Hugo Caselles-Dupré, Florian Lesaint, and Jimena Royo-Letelier. 2018. Word2vec applied to recommendation: Hyperparameters matter. In *Proceedings of the 2018 ACM Conference on Recommender Systems*. 352–356.

[7] Chih-Cheng Chang, Brian Thompson, Hui (Wendy) Wang, and Danfeng Yao. 2010. Towards Publishing Recommendation Data with Predictive Anonymization. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS'10)* (Beijing, China) *(ASIACCS âĂŹ10)*. Association for Computing Machinery, New York, NY, USA, 24âĂŞ35. https://doi.org/10.1145/1755688.1755693

[8] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys'19)*. 101–109.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*. IEEE, 248–255.

[10] Kazuto Fukuchi, Satoshi Hara, and Takanori Maehara. 2020. Faking Fairness via Stealthily Biased Sampling. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20), Special Track on AI for Social Impact (AISI)*. AAAI, 8.

[11] Blake Hallinan and Ted Striphas. 2016. Recommended for you: The Netflix Prize and the production of algorithmic culture. *New media & society* 18, 1 (2016), 117–137.

[12] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS)* 5, 4 (2015), 1–19.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the 2016 International Conference on Learning Representations (ICLR'16)*. 10.

[14] Daniel G Horvitz and Donovan J Thompson. 1952. A generalization of sampling without replacement from a finite universe. *J. Amer. Statist. Assoc.* 47, 260 (1952), 663–685.

[15] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2019. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635* (2019).

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Conference on Neural Information Processing Systems (NIPS'13)*. 3111–3119.

[17] Arvind Narayanan and Vitaly Shmatikov. 2006. How To Break Anonymity of the Netflix Prize Dataset. *arXiv* abs/cs/0610105 (2006).

[18] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*. 188–197.

[19] Douglas W Oard, Jinmook Kim, et al. 1998. Implicit feedback for recommender systems. In *Proceedings of the 1998 AAAI Workshop on Recommender Systems*. 81–83.

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Proceedings of the 33th Conference on Neural Information Processing Systems (NeurIPS'19)*. Curran Associates, 8024–8035.

[21] Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural Networks* 12, 1 (1999), 145–151.

[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*. 452–461.

[23] Luc Rocher, Julien M Hendrickx, and Yves-Alexandre De Montjoye. 2019. Estimating the success of re-identifications in incomplete datasets using generative models. *Nature Communications* 10, 1 (2019), 1–9.

[24] Pierangela Samarati and Latanya Sweeney. 1998. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. (1998).

[25] Sumit Sidana, Charlotte Laclau, and Massih-Reza Amini. 2018. Learning to Recommend Diverse Items over Implicit Feedback on PANDOR. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys âĂŹ18)*. Association for Computing Machinery, New York, NY, USA, 427âĂŞ431. https://doi.org/10.1145/3240323.3240400

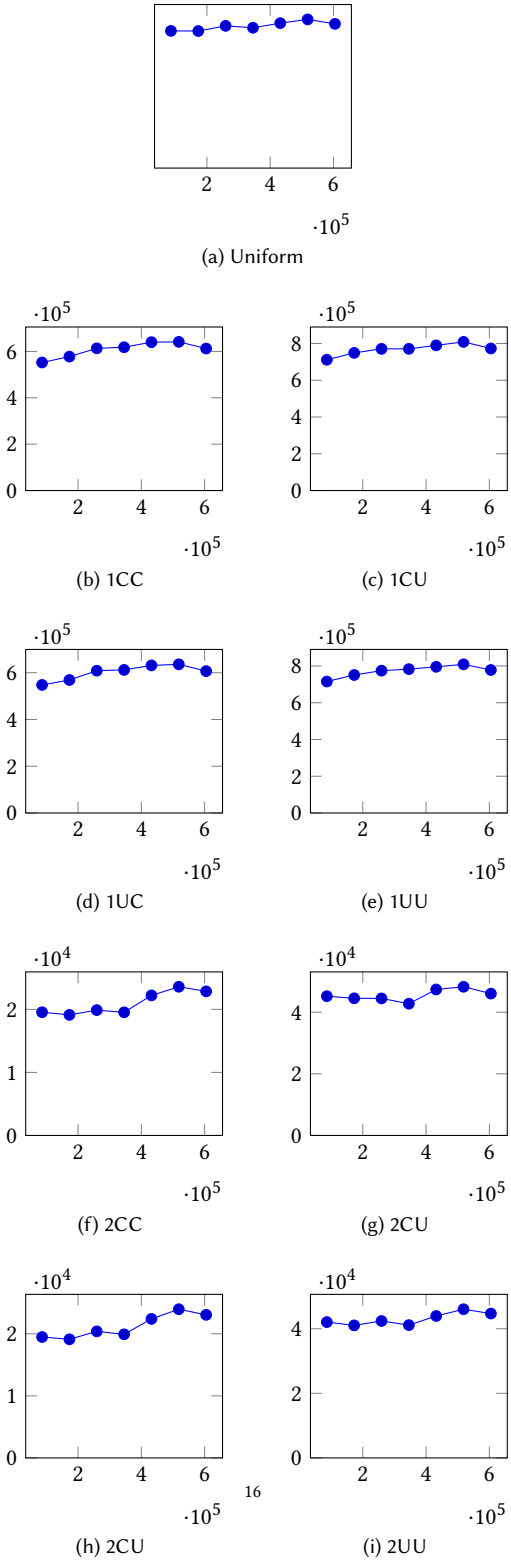
[26] Sumit Sidana, Charlotte Laclau, Massih R Amini, Gilles Vandelle, and André Bois-Crettez. 2017. KASANDR: A Large-Scale Dataset with Implicit Feedback for Recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1245–1248.

[27] SS Vallender. 1974. Calculation of the Wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications* 18, 4 (1974), 784–786.

[28] Lucas Vinh Tran, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. 2020. HyperML: A boosting metric learning approach in hyperbolic space for recommender systems. In *Proceedings of the 2020 International Conference on Web Search and Data Mining (WSDM'20)*. 609âĂŞ617.

[29] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys'18)*. 279–287.

[30] Sirui Yao and Bert Huang. 2017. Beyond parity: Fairness objectives for collaborative filtering. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS'17)*. 2921–2930.




(a) Uniform

(b) 1CC (c) 1CU

(d) 1UC (e) 1UU

(f) 2CC (g) 2CU

(h) 2CU (i) 2UU

Fig. 4. Timestamp distributions. The $x$-axis is for the time [sec] from some epoch, and The $y$-axis is for the number of clicks in each day.