# Tracking You through DNS Traffic: Linking User Sessions by Clustering with Dirichlet Mixture Model

Mingxuan Sun
Louisiana State University
msun@csc.lsu.edu

Guangyue Xu
Louisiana State University
gxu3@lsu.edu

Junjie Zhang
Wright State University
Junjie.Zhang@wright.edu

Dae Wook Kim
Wright State University
kim.107@wright.edu

## ABSTRACT

The Domain Name System (DNS), which does not encrypt domain names such as "bank.us" and "dentalcare.com", commonly accurately reflects the specific network services. Therefore, DNS-based behavioral analysis is extremely attractive for many applications such as forensics investigation and online advertisement. Traditionally, a user can be trivially and uniquely identified by the device's IP address if it is static (i.e., a desktop or a laptop). As more and more wireless and mobile devices are deeply ingrained in our lives and the dynamic IP address such as DHCP has been widely applied, it becomes almost impossible to use one IP address to identify a unique user. In this paper, we propose a new tracking method to identify individual users by the way they query DNS regardless of dynamic changing IP addresses and various types of devices. The method is applicable based on two observations. First, even though users may update IP addresses dynamically during different sessions, their query patterns can be stable across these sessions. Secondly, domain name look ups in sessions are different from users to users according to their personal behaviors. Specifically, we propose the constrained Dirichlet multinomial mixture (CDMM) clustering model to cluster DNS queries of different sessions into groups, each of which is considered being generated by a unique user. Compared with traditional supervised and unsupervised models, our model does not acquire any labeled user information that is very hard to obtain in real networks or the specification of the number of clusters, and meanwhile enforces the maximum number of session data in each cluster, which fits the DNS tracking problem nicely. Experimental results on DNS queries collected from real networks demonstrate that our method accomplishes a high clustering accuracy and outperforms the existing methods.

## CCS CONCEPTS

• **Security and privacy → Network security**; • **Computing methodologies → Cluster analysis**;

*MSWiM '17, November 21–25, 2017, Miami, FL, USA*
© 2017 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
ACM ISBN 978-1-4503-5162-1/17/11…$15.00
https://doi.org/10.1145/3127540.3127567

## KEYWORDS

DNS behavior tracking, clustering, Dirichlet mixture model

## 1 INTRODUCTION

The Domain Name System (DNS), which maps between domain names and Internet Protocol (IP) addresses, is involved in almost all network interactions between users and network services such as web browsing, online shopping, instant messaging, and entertainments. DNS, even with security extensions (i.e., DNSSEC), does not encrypt domain names such as "yahoo.com", "bank.us", and "dentalcare.com" that commonly accurately reflect the specific network services offered by their corresponding IPs. As a result, analyzing DNS traffic shows great promise to reveal users' network activities, behaviors, and patterns.

DNS-based behavioral analysis is extremely attractive for many applications such as forensics investigation and online advertisement. Government agencies such as NSA would like to identify users with malicious activities given a collection of DNS query logs, in case adversaries change IP addresses frequently to cover their traces. Secondly, service providers such as Google and OpenDNS may keep track of user visiting behaviors and target personalized ads to each user. Meanwhile, significant privacy concerns are introduced to individual users. If users are aware of the existence of the types of privacy threats through DNS traffic, they can learn to control the amount of information leakage by possibly using different domain name servers or sharing online access with different people in a group.

Despite its promise and privacy concerns, using DNS to infer users' information requires that DNS queries/responses can be correctly attributed to their corresponding originators (i.e., network devices who generate them). This is a trivial task when devices' IP addresses are static (i.e., not changing over time) since we can directly use an IP address to represent a network device (i.e., a desktop or a laptop). Unfortunately, the dynamic IP address such as DHCP has been widely adopted to mitigate the saturation of IPv4 address usage. Specifically, a network device is usually assigned with different IP addresses across multiple days. This makes it impossible to use one IP address to identify a unique network device (or equivalently a user).

It is possible to identify individual users by the way they query DNS regardless of dynamic changing IP addresses and various types of devices based on two observations. First, even though users may update IP addresses dynamically from time to time, their query patterns are likely to be consistent across these periods [8, 10]. As a matter of fact, the online visiting behaviors of each individual user are recurrent and consistent during a short continuous period such as days or weeks. For example, users who have been highly frequent in some services such as "healthcare.com" are likely to remain frequent in the near future. Secondly, domain name look ups in a period are unique from users to users according to their personal behaviors. For example, some always prefer movies and shopping while others focus on stock market news and banking.

A few methods [8, 10] have been proposed to generate DNS-based fingerprints from DNS queries and use the extracted fingerprints to match and attribute DNS queries to their originators. Specifically, Herrmann et al. [8] employed top-visited domains as fingerprints whereas Kim et al. [10] leveraged temporal behaviors of domain queries (e.g., the order in which an arbitrary pair of domains are visited). Both methods follow the supervised learning paradigm, where a large set of DNS queries with labeled users are required to generate fingerprints for each user. Unfortunately, it is extremely challenging to acquire such pre-labeled dataset in real networks, which fundamentally limit the practical application of these types of research methods.

On the other hand, unsupervised learning such as clustering can be used to cluster DNS queries of different sessions into groups, each of which is considered being generated by a unique user. Specifically, we assume that queries issued from the same IP in a session are from the same user, where a session length is a continuous duration such as a day. As illustrated in Figure 1, there are assumed three IP addresses and each issues a set of DNS queries in four consecutive sessions. The goal of clustering is to group DNS query sessions into distinct groups, where sessions in the same group belong to the same user. For example, we believe that sessions labeled in orange belong to the same user since they frequently access particular types of domains such as "Yahoo" and "Learning". Another group linking all sessions labeled in blue tends to be a user who likes particularly shopping and online TV sites, and the last group labeled in green belongs to a user who likes news from "sina.com" and social media actives via "qq.com".

Most existing work such as Kmeans and constrained Kmeans [11] has been proposed under this framework, where top-visited domains are used as features for clustering user sessions. These methods need the number of clusters to bootstrap, which is ideally the number of distinct users in the network. However, due to the bursty nature of human activities, such a design fundamentally undermines the applicability of this method since accurately estimating the number of users is extremely challenging in practice, if not entirely unrealistic. On the other hand, traditional Dirichlet multinomial mixture (DMM) and its variations [2, 7, 22] frequently used in Bayesian nonparametric modeling can automatically adjust the number of clusters. However, each cluster may contain many sessions generated from multiple users, which does not fit into the scenario of DNS tracking since we need to identify pure clusters and each corresponds to a particular user.

As a means towards systematically solving these challenges, we propose the constrained Dirichlet multinomial mixture (CDMM) clustering model, which is a generalization of the traditional Dirichlet multinomial mixture clustering (DMM) model. The key innovation of the proposed model is to sample a session's cluster label only from the qualified candidate clusters whose current sizes are smaller than a threshold, which is the maximum possible number of sessions a user can contribute during the observation period. For example, if session length is defined as a day and all session data is collected in a week, the threshold is 7 assuming each user only contributes one session each day. The constrained model fits the scenarios of DNS clustering better and can potentially improve the clustering performance. To our knowledge, this is the first research about imposing cluster size constraint on nonparametric Bayesian clustering methods. To summarize, we have made the following contributions:

- We have designed a new clustering model without requiring the specification of the number of clusters and meanwhile enforcing the maximum number of sessions in each cluster, which fits the DNS tracking problem nicely.
- Experimental results on DNS queries collected from real networks demonstrate that our method accomplishes a high clustering accuracy and outperforms the existing methods.

## 2 RELATED WORK

Active research has been conducted to infer users' behaviors through collected network traffic including DNS queries. The focuses of existing methods generally fall into two categories including i) attributing network events to individual network users and ii) inferring users' demographic information. Our method focuses on attributing sessions of DNS queries to their corresponding users (and thus falls into the first category).

A few methods have been proposed to attribute network events to individual users [6, 8–11, 17]. Most of these methods [6, 8–10, 17] first generate behavioral fingerprints for network users and then use these fingerprints to attribute network events whose users are unknown, which requires a significantly large set of events whose users are pre-labeled. This fundamentally limits the practical usage of these systems. In contrast, our method does not require any DNS sessions to be pre-labeled. In addition, methods including [6, 9, 17] generate user fingerprints using data sources (e.g., encrypted wireless traffic, HTTP traffic, and search engine traffic) that offer much finer-grained information compared to DNS traffic. In comparison with our method, Herrmann et al. [8] and Kim et al. [10] generate user fingerprints using the same data source (i.e., DNS queries). However, both methods follow the supervised learning paradigm, where a long period (multiple days) of DNS queries need to be labelled for bootstrapping. The work closest to ours is [11]. Similar to [11], our method uses clustering method to aggregate DNS sessions into clusters, where each cluster is expected to contain sessions generated by one user. In spite of such similarity, the method in [11] mandates the number of clusters. Unfortunately, it is extremely challenging, if not entirely impossible, to specify the number of clusters in practice, fundamentally limiting the application of [11]. In comparison, our method does not require the specification of the number of clusters.

A substantial body of studies [1, 3–5, 12–16, 19–21, 24, 25] have focused on inferring users' application-level activity information

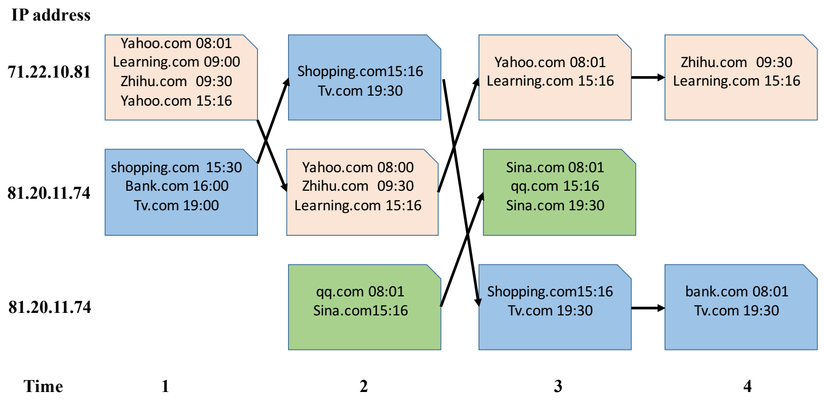

**Figure 1: Illustration: Linking user DNS query sessions regardless of changing IP addresses.**

from a variety of network resources such as HTTP traffic, search engine traffic, and encrypted skype traffic. For example, Chen et al. [3] used HTTP connection patterns to infer users' browsing activities. In [20, 21], Wright et al. used machine learning methods to reveal spoken language phases from encrypted VoIP traffic. Zhang et al. [23] inferred users' running applications from network-level traffic patterns. Sun et al. [19] also recovered webpages visited by users through encrypted network traffic by adopting carefully designed traffic signatures. Krishnan et al. [12] revealed search engine queries based on domain name correlations. Other types of online activities have also been employed for demographic inference such as friendship on Facebook [15, 24], search queries [1], location check-ins [25], and social network interests [16]. Although these methods have different objectives compared to ours, we expect our method can complement these methods by associating a user with more network events.

## 3 CONSTRAINED DMM MODEL

In this part, we introduce constrained Dirichlet multinomial mixture clustering (CDMM) model in the context of linking user DNS query activities regardless of dynamically updating IP addresses. A session of a particular IP can be represented as a vector of dimension $V$, where $V$ is the size of unique domain names and each element of the vector is the query frequency for each domain from the IP during a day. Given a collection of session data $\vec{S} = \{s_i\}_{i=1}^{N}$, we would like to find the corresponding cluster indicators $\vec{Z} = \{z_i\}_{i=1}^{N}$, where $z_i \in \{1, 2, \ldots, K\}$ and $K$ is the total number of clusters, e.g., the number of unique users. Since we do not know the actual number of clusters, we can initialize $K$ with a large number, and the clusters will be learned automatically to fit to the data.

Specifically, our model CDMM is a probabilistic generative process, in which each session is generated from a mixture of several components where each component corresponds to a specific user. Session vectors $\{s_i\}_{i=1}^{N}$ are observed variables and the component indicators $\vec{Z} = \{z_i\}_{i=1}^{N}$ are latent variables. In addition, our model extends DMM by adding a size constraint to each cluster, that is the number of sessions in each cluster should be less than *maxSize*. In the following sections, we present the details of the probabilistic model, the generative process, and the collapsed Gibbs sampling method for parameter inference for CDMM. The math notations of the model and parameters are listed in Table 1.

**Table 1: NOTATIONS**

| Notation | Description |
|---|---|
| $\alpha, \beta$ | concentration parameter for Dir-Mult distribution |
| $\vec{S}$ | all session vectors |
| $\vec{Z}$ | cluster label vectors for corresponding sessions |
| $s_i$ | $i^{th}$ session vector from $\vec{S}$ |
| $z_i$ | cluster label for $i^{th}$ session |
| $j$ | $j^{th}$ domain name in dictionary |
| $\vec{\Phi}_k$ | multinomial parameters for cluster $k$ over domain dictionary |
| $\vec{\theta}$ | multinomial parameters for the weights of each cluster |
| $V$ | the size of domain name dictionary |
| $N$ | the number of sessions |
| $K$ | the number of initialized clusters |
| $m_k$ | the number of sessions in cluster $k$ |
| $N_k$ | the occurrences of all domains in cluster $k$ |
| $N_k^j$ | the occurrence of $j^{th}$ domain in cluster $k$ |
| $n_i$ | the occurrence of all domains in session $s_i$ |
| $n_i^j$ | the occurrence of $j^{th}$ domain in session $s_i$ |
| $maxSize$ | maximum cluster size |
| $maxIter$ | maximum number of iterations |

### 3.1 Generative Process

The CDMM model is parameterized by two Dirichlet multinomial distributions and corresponding positive scaling parameters $\alpha$ and $\beta$. The session vectors $\{s_i\}_{i=1}^{N}$ are generated with symmetric Dirichlet prior:

$$\theta|\alpha \sim Dir(\alpha|K, \ldots, \alpha|K), \tag{1}$$

$$z_i|\theta \sim Mult(\theta_1, \ldots, \theta_k) \quad i = 1, \ldots, N, \tag{2}$$

$$\Phi_k|\beta \sim Dir(\beta) \quad k = 1, \ldots, K, \tag{3}$$

$$s_i|z_i, \{\Phi_k\}_{k=1}^{K} \sim Mult(s_i|\Phi_{z_i}). \tag{4}$$

The process first samples one out of $K$ clusters, e.g., $k$, based on the multinomial distributions with parameter $\theta$, and then generates session $s_i$ according to the distribution of cluster $k$ parameterized by $\Phi$. In order to enforce size constraint on each cluster, we need to modify the above generative procedure as follows:

- For the current session, constructing the candidate cluster (user) label list, where all the candidate clusters have fewer than *maxSize* DNS queries.
- Sampling the cluster label from the candidate list as in Equation (2).
- Choosing the corresponding multinomial distribution of a candidate cluster from Dirichlet distribution parameterized by $\beta$.
- Generating DNS queries for the session based on the chosen cluster label and multinomial distribution parameterized by $\Phi_{z_i}$ as defined in Equation (4).

More intuitively, we can use the Chinese Restaurant process (CRP) to illustrate the CDMM model where sessions are customers and users are represented as tables in a Chinese restaurant. Assume there are $K$ fixed tables and $N$ customers. At the very beginning, each user is randomly assigned to a table. If they are not satisfied with the initial assignment, they can change tables dynamically. We assume this reallocating process is guided by the following rules. Firstly, users are more willing to join a table with more people in order to feel comfortable, which means rich clusters gets richer. In addition, users prefer to sit with their friends, that is users like to join a table where they share more similar features with existing table users. Most importantly, different from traditional CRP, we have a size constraint for each table. If the table is full, users has to choose a second preferable table instead.

## 3.2 Parameter Learning

Given sessions and their initial label assignments, we employ the Gibbs sampling method to infer the hidden cluster for each session. The key concept of Gibbs sampling is to sample the cluster label using the conditional posterior probability $p(z_i|\vec{Z}_{\neg i},\vec{S})$ as shown in Equation (5):

$$p(z_i=k|\vec{Z}_{\neg i},\vec{S},\alpha,\beta) \propto p(z_i=k|\vec{Z}_{\neg i},\alpha)\cdot p(s_i|z_i=k,\vec{Z}_{\neg i},\vec{S}_{\neg i},\beta), \tag{5}$$

where $\vec{Z}_{\neg i}$ is the user assignments for all sessions except for session $s_i$, The intuition of this formula is that removing the effect of session $s_i$ from the DNS session corpus and using the rest information to infer the user label for session $s_i$.

The first term on the right side of Equation (5) is the component weight measured by the number of data points in cluster $k$. The more sessions a user already have, the more likely session $s_i$ is assigned to this user. Our model also imposes size constraint for each cluster in case the cluster becomes increasingly large. The concrete calculation for the component weight is shown in Equation (6):

$$p(z_i=k|\vec{Z}_{\neg i},\alpha)=\int_\theta p(z_i|\theta)p(\theta|\alpha)=\frac{m_{k,\neg i}+\alpha}{N-1+K\alpha}. \tag{6}$$

The second term of Equation (6) in the middle is the likelihood measuring how likely the current cluster generates session $s_i$. The more features session $s_i$ and the existing sessions of the cluster share, the more likely $s_i$ will fall into this cluster. In CDMM, we adopt multinomial distribution to model each cluster, which can tackle the sparsity problem and is more suitable to our actual session clustering circumstance. And the corresponding formula is shown in Equation (7):

$$p(s_i|z_i=k,\vec{Z}_{\neg i},\vec{S}_{\neg i},\beta)\propto\frac{\prod_{j\in s_i}\prod_{l=1}^{n_i^j}(N_{k,\neg i}^j+\beta+l-1)}{\prod_{t=1}^{n_i}(N_{k,\neg i}+V\beta+t-1)}, \tag{7}$$

where $N_{k,\neg i}^j$ and $N_{k,\neg i}$ are the cluster-level statistics, with $N_{k,\neg i}^j$ as the total occurrences of domain $j$ in cluster $k$ by removing the effect of session $s_i$, and $N_{k,\neg i}$ as the number of sessions in cluster $k$ after removing the effect of session $s_i$. In addition, $n_i$ is the total occurrences of all domains in session $s_i$, and $n_i^j$ is the occurrences of domain $j$ in session $s_i$. Since not all domains occur in a specific session $s_i$, we only take into account the domain names that appear in the session, that is $j\in s_i$, when calculating the probability.

Combining Equations (6) and (7) into Equation (5), we can rewrite the full posterior distribution as follows:

$$p(z_i=k|\vec{Z}_{\neg i},\vec{S})\propto\frac{m_{k,\neg i}+\alpha}{N-1+K\alpha}\frac{\prod_{j\in s_i}\prod_{l=1}^{n_i^j}(N_{k,\neg i}^j+\beta+l-1)}{\prod_{t=1}^{n_i}(N_{k,\neg i}+V\beta+t-1)}. \tag{8}$$

Gibbs sampling works in a similar way as Expectation-Maximization (EM). For each iteration, after determining the cluster label $z_i$ for a session $s_i$, e.g., $z_i=k$, we need to update the parameters of the $k^{th}$ corresponding cluster. Since we assume each cluster follows a multinomial distribution, the updating rule for each cluster in each iteration is shown in Equation (9):

$$p(\vec{\Phi}_k|\vec{S},\vec{Z},\beta)=Dir(\vec{\Phi}_k|\vec{n}_k+\beta)\quad and\quad \Phi_{k,j}=\frac{N_k^j+\beta}{\sum_{j=1}^V N_k^j+V\beta}, \tag{9}$$

where $\vec{n}_k=[N_k^1,N_k^2,\ldots,N_k^V]$ and each entry $N_k^j$ is the occurring number of the $j^{th}$ query in the $k^{th}$ cluster. Different clusters have different weights on the query frequency of domains in the dictionary, meaning users have their own preference when browsing the Internet. The discriminative distribution information can help group the sessions into clusters.

## 3.3 Hyper-parameters

In CDMM, there are two hyper-parameters, $\alpha$ and $\beta$, which balance between the priority knowledge and the observed data. The prior knowledge can be treated as pseudo experimental results before we observe the real data.

The hyper-parameter $\alpha$ controls the component weight. We give $\alpha$ a large value when we are confident about our prior knowledge compared with the real data. A large $\alpha$ means that each cluster has equal probability to generate each session data at the very beginning and these probability needs more observed data to be adjusted. In a similar way, $\beta$ controls the multinomial distribution of domain name occurrences for each cluster. A small $\beta$ means that each user has equal preference over all domains, which however can be easily updated by the observed data. In our practice, $\alpha$ with smaller values such as 0.001 and $\beta$ with smaller values such as 0.05 usually work fine.

## 3.4 Algorithm

In this part, we list the pseudo code of constrained Gibbs sampling for our CDMM model, as shown in Algorithm 1. It follows the common Gibbs sampling framework: 1) scan session vector iteratively and remove the effect of the current session; 2) construct the qualified cluster list for the session; 3) using rest information to calculate the posterior probability of a session belonging to each qualified cluster and sample cluster label; 4) update multinomial distribution based on the selected cluster label.

Compared with the traditional DMM model, the most significant modification of CDMM is *Step 3*. Before sampling the cluster label for session $s_i$, we need to construct the qualified cluster list first. The list is dynamic during each iteration and contains clusters who have fewer than *maxSize* sessions currently. We only sample cluster label from the qualified list because some clusters are already full and cannot take any more sessions.

**Algorithm 1:** Constrained Dirichlet Multinomial Mixture Model (CDMM)

```
Input:
    • Sessions in the DNS query logs, $\vec{S}$
    • Hyper-parameter, $\alpha$ and $\beta$
    • Estimated initial number of clusters, $K$
    • Maximal size of each cluster size, $maxSize$
Output: Cluster label vector for each session, $\vec{Z}$
begin
    Step 1: Set $m_k$, $N_k$ and $N_k^j$ to zero for each cluster $k$
    Step 2: Randomly assign cluster label for each session and
    update cluster information based on assignments
    for each session $i \in [1, N]$: do
        $z_i \leftarrow k \sim Random[1, K]$
        $m_k \leftarrow m_k + 1$ and $N_k \leftarrow N_k + n_i$
        for domain $j \in s_i$ do
            $N_k^j \leftarrow N_k^j + n_i^j$
        end
    end
    Step 3: Constrained Collapsed Gibbs Sampling
    for iter ← 1 to maxIter do
        for each session $i \in [1 : N]$ do
            ①save cluster label for the current session: $k = z_i$
            ②remove session effect
            $m_k \leftarrow m_k - 1; N_k \leftarrow N_k - n_i$ for each domain
            name $j \in s_i$ do
                $N_k^j \leftarrow N_k^j - n_i^j$
            end
            ③construct qualified cluster list with fewer than
            maxSize sessions
            ④sample a cluster $k$ for $s_i$ from the candidate
            cluster list by Equation (8):
            $z_i \leftarrow k \sim p(z_i = k|\vec{Z}_{\neg}i, \vec{S})$
            ⑤recover this session's effect
            $m_k \leftarrow m_k + 1$ and $N_k \leftarrow N_k + n_i$
            for each domain name $j \in s_i$ do
                $N_k^i \leftarrow N_k^i + n_i^j$
            end
        end
    end
end
```

## 3.5 Complexity Analysis

- **Space complexity**:
  Following Algorithm 1, we need to store all session vectors $\vec{S}$, which essentially is a session-domain matrix of size $N \cdot V$, where $N$ is the number of sessions and $V$ is the size of domain name dictionary. Compared with Dirichlet Process Mixture model, we predefine a fixed number of maximum clusters $K$. Therefore, for each iteration, we need to store the information about $\vec{Z}$ with size $N$, $m_k$ and $N_k$ with size $K$, and $N_k^j$ with size $K \cdot V$. Since the number of clusters is always smaller than the number of data points, the total space complexity of CDMM is $O(N \cdot V)$. In practice, the matrix will be really sparse, thus the actual storage is much less than $O(N \cdot V)$.
- **Time complexity**: The most time-consuming part of CDMM is the collapsed Gibbs sampling procedure. Given the number of iterations and session-Vocabulary matrix, Gibbs sampling alternatively samples user label for each session from the conditional posterior distribution computed with other user labels fixed. For one session, we need to compute $K$ probabilities and each represents the likelihood that the $k^{th}$ cluster generates the session. To compute this probability, we need to go through every domain occurrence in each session. We assume the average number of domains in a session is $L$. Then given $maxIter$, the complexity of CDMM is $O(maxIter \cdot N \cdot K \cdot L)$.

## 4 EXPERIMENTS

To evaluate the performance of our proposed CDMM model in clustering DNS sessions, we carry out experiments on real-world dataset and compare results with several state-of-the-art clustering techniques.

## 4.1 Dataset

Our DNS dataset contains 1M domain queries covering totally 89,009 distinct domain names over 7 days from September 23 to September 31 in 2013. The dataset was collected at a Chinese university where all DNS queries are originated from the student housing network. The log keeps the records of a large number of students with different querying behaviors. For example, each student accesses domains of different types both in their studying and leisure times. Each student is uniquely assigned a static IP address that connects a device in the dorm to the broadband network, so the dataset contains the ground truth that maps between user IP address and DNS queries. The labeled information allows us to evaluate the effectiveness of our proposed clustering algorithms.

We preprocess the raw DNS query log and format it in a matrix as described in the previous section. Empirical studies show that a session time can be a fixed duration of 24 hours since a large number of users do not change IP addresses during a day. The data is sparse. The number of queries per user follows a long tail distribution, where 80% users only query fewer than 50 domains and 20%users query more than 50 domains. Similarly, the number of sessions per domain also follows a long tail distribution. To overcome data sparsity, following the approach in [8], we select the most frequent 200 domains with at least 100 queries from different sessions as the dictionary for feature representation, which largely reduces the dimension without dramatically hurting clustering performance. After feature selection, we construct our testing subset of 4K sessions from 478 distinct users with different activity levels.

## 4.2 Evaluation Metrics

We evaluate the results against the traditional gold standard using several classic metrics, including adjusted Rand Index (ARI) [18], Normalized Mutual Information (NMI), homogeneity, completeness, and V-score. In addition, we would like to evaluate the clustering performance in the context of tracking individual users with respect to changes of user activity levels, so we also adopt the Traceability metric. We give more information about these metrics below.

**ARI**: Rand Index (RI) measures the cluster similarity by counting pairs that are assigned correctly or wrong. ARI is an "adjusted for chance" measure which is independent of the number of clusters. We use ARI to compare the clustering performance.

**NMI**: Mutual Information (MI) measures the information shared between two clusters. The more information they share, the more efficient the clustering algorithm is. NMI is the normalized mutual information by scaling the MI results between 0 and 1, where 0 means sharing no information and 1 means perfect correlation. Given the true labels and the cluster labels, NMI is a symmetric indicator to measure the clustering results.

**Homogeneity, Completeness, and V-score**: Homogeneity measures the purity of the cluster. If most data points in the cluster have the same label, the homogeneity of this cluster will be high. Otherwise, the cluster is not pure and will have a lower homogeneity. Given ground truth, high completeness means more data points with the same true label fall into the same cluster. V-score measures how well the clustering result satisfies homogeneity and completeness.

**Traceability**: Adopting the same notation from [11], we define that a user is completely traceable, if all his or her sessions across different days are assigned to a single cluster. It is possible that the session data of completely traceable users are grouped together with other users' session data and the clusters are not necessarily pure. Thus we define a user is perfectly traceable if all his or her sessions are assigned to a single cluster and the cluster is pure without other users' sessions. We would like to measure the percentage of completely linked users and perfectly linked users, respectively.

## 4.3 Baselines for Comparison

- **Kmeans**: Kmeans is one of the most popular clustering algorithms. Kmeans iteratively clusters $N$ samples into $K$ clusters by assigning samples to the nearest centroid and recomputes the centroid for all clusters. Kmeans quits its iteration when reaching the maximum number of iterations or no cluster reassignment for samples occurs. The drawback of Kmeans clustering is that the number of clusters should be pre-defined. Moreover, measuring similarity by Euclidean distance is not quite appropriate for some clustering problems. And initial centroid will affect the final clustering performance.
- **Constrained Kmeans (c-Kmeans)** [11]: c-Kmeans is an extension to the existing K-means with an additional constraint that the number of points in each cluster should be smaller than a certain threshold. The core idea of this method is to add a reassignment step after the traditional Kmeans method. Given the cluster size constraint, after recomputing the centroid for each cluster, c-Kmeans sorts the data points within a cluster by distance from the centroid. Based on the distance, c-Kmeans removes data points far from the centroid until this cluster satisfies the size constraint. Each removed point will be assigned to the closest cluster with size smaller than the constraint. The method fits our scenario of DNS clustering. C-Kmeans is also sensitive to initialization methods. Moreover, C-Kmeans requires the number of clusters to be pre-defined.
- **Dirichlet Multinomial Mixture Model (DMM)** [22]: In contrast to K-means methods, DMM can dynamically cluster data to $k$ groups where the exact $k$ does not need to be predefined. DMM is a generative model and its core idea is to sample the cluster label based on the conditional posterior distribution. The posterior probability of a sample assigning to a cluster is determined by cluster size and the similarity between the sample and the existing cluster samples. The method fits our scenario of DNS clustering, where the number of users changes every day. However, this generative process will make big clusters get bigger and this trend will violate the cluster size constraint.
- **Constrained Dirichlet Multinomial Mixture Model (CDMM)**: With the similar relationship between Kmeans and C-Kmeans, our CDMM model is an extension to DMM with an additional constraint that the number of points in each cluster should be smaller than a certain threshold. The core change of this method to DMM is the cluster sample step. In this step, we do not sample the label from all clusters. We only sample the cluster label from the qualified user list based on the posterior distribution.

## 4.4 Clustering Results

*4.4.1* **Clustering performance using classic metrics:** We first briefly summarize the experimental results and discuss the effects of the maximum number of clusters and the initialization approaches on the clustering accuracy.

To compare the model robustness with respect to the cluster size, we conduct two types of experiments, where the first assumes that the number of users is very close to the ground truth size, and the second assumes that the number of users is 200% of the ground truth to simulate the burstiness in complex online systems.

For each experiment, we further discuss the performance of all four clustering models with two different types of initialization strategies: random initialization (R) and smart initialization (S),

Table 2: **Evaluate the clustering performance of eight methods using five classic metrics. Suffix "R" indicates random initialization and suffix "S" indicates smart initialization.**

| clusterNum | Metrics | Methods | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Kmeans-R | CKmeans-R | DMM-R | CDMM-R | Kmeans-S | CKmeans-S | DMM-S | CDMM-S |
| True User Number | ARI | 0.3024 | 0.4802 | 0.2051 | **0.5021** | 0.3812 | 0.6444 | 0.2744 | **0.7539** |
| | NMI | 0.4381 | 0.5564 | 0.4089 | **0.5610** | 0.5206 | 0.6909 | 0.5099 | **0.7730** |
| | HOMO | 0.8192 | **0.8719** | 0.7319 | 0.8654 | 0.8517 | 0.9113 | 0.8063 | **0.9320** |
| | Complete | 0.8549 | 0.8791 | **0.8888** | 0.8885 | 0.8773 | 0.9171 | 0.9134 | **0.9499** |
| | V-Score | 0.8367 | 0.8755 | 0.8028 | **0.8768** | 0.8643 | 0.9142 | 0.8565 | **0.9409** |
| 200% True User Number | ARI | 0.3393 | 0.4591 | 0.2197 | **0.5082** | 0.3793 | 0.5071 | 0.2802 | **0.7332** |
| | NMI | 0.3931 | 0.4642 | 0.4441 | **0.5655** | 0.4305 | 0.4993 | 0.5267 | **0.7520** |
| | HOMO | 0.9075 | **0.9302** | 0.7490 | 0.8670 | 0.9138 | **0.9417** | 0.8111 | 0.9248 |
| | Complete | 0.8418 | 0.8607 | 0.9076 | **0.8898** | 0.8510 | 0.8700 | 0.9231 | **0.9463** |
| | V-Score | 0.8734 | 0.8941 | 0.8207 | **0.8782** | 0.8813 | 0.9044 | 0.8635 | **0.9354** |

which configure the initial $k$ cluster centroids in different ways. In random initialization, the centroids of each cluster are initialized randomly from one of the session data, which fits the scenario where the algorithm is deployed on a new DNS service with no prior knowledge. In smart initialization, we assume that the ground truth labels are available from $k$ unique users. Specifically, for Kmeans and C-Kmeans, using smart initialization, centroids are initialized as session vectors from $k$ different users as in [8]. For DMM and CDMM, using smart initialization, sessions of the same users are assigned to the same cluster with higher probability, which is an extremely ideal case. After initialization, clustering models will re-assign each point to clusters iteratively to fit to data until the resulting clustering assignments will not change any more.

Table 2 shows the clustering results of Kmeans, C-Kmeans, DMM, and CDMM with both random initialization and smart initialization strategies, a total of 8 methods.

First of all, by comparing two batch experiments, model CDMM-R/S and DMM-R/S are stable even if the maximum number of clusters is very different from the ground truth size, e.g., twice of the ground truth. This is because DMM and CDMM can adjust the number of clusters dynamically to fit the data and the results are less sensitive to the initialized size of clusters. However, Kmeans and C-Kmeans are more sensitive to the initialized size of clusters. For example, the NMI score for Kmeans-R drops from 0.4381 to 0.3931, the NMI score for Kmeans-S drops from 0.5206 to 0.4305, the ARI score for CKmeans-R drops from 0.4802 to 0.4591, and the ARI score of CKmeans-S drops from 0.6444 to 0.5071 when the cluster size doubles.

Furthermore, the performance of clustering methodologies heavily depends on the initial cluster assignments. The results of all four models with smart initialization "-S" are much better than those with random initialization "-R". This indicates that clustering with prior knowledge such as user session patterns from previous days or weeks can largely increase the accuracy.

Finally, our CDMM outperforms all other models consistently with both types of initialization. Specifically, from Table 2, we can see that DMM-R/S have lower metric scores except for homogeneity. From the definition of homogeneity, clusters with large sizes are likely to generate high homogeneity score. However, all session data with different ground truth (e.g., user labels) may fall in the same cluster thus all other metrics are extremely low. Compared with the traditional DMM, our Model CDMM-R/S add the constraint to cluster size, which results in smaller and purer clusters. Similar conclusion can be drawn when comparing C-Kmeans and Kmeans. In summary, due to both advantages of dynamic allocation and size constraints, CDMM achieves better performance compared with other methods.

*4.4.2* **Clustering performance in tracking DNS users:** We would like to measure the traceability of users with respect to different activity levels. Specifically, we divide users into two groups: active users who appear more than five days in a week, and less active users otherwise. Figure 2 compares the clustering results of all eight models measured by traceability in different groups. As long as all the sessions of a user is in a cluster, this user is completely traceable. A user is perfectly traceable if all the sessions are in one cluster and the cluster does not have other users' sessions. By definition, perfect traceable rate (displayed on right) is always lower than complete traceable rate (displayed on left).

Generally, the accuracy in terms of traceability is much higher for active users than for less active users. For example, the complete traceability using CDMM-R increases from 26% to 36%, and the perfect traceability increases from 10% to 19% when users are more active.

Further, our CDMM model also gets the best results compared with other models in terms of traceable rate. If users are active on more than 5 days in a week, it is likely that the algorithm can completely identify the users at the rate of 19% with random initialization and 49% with smart initialization.

In summary, our model is very effective to trace active users. From the user perspective, in case to avoid these types of privacy threats through DNS traffic, they may try different domain name servers thus to spread activities in each domain server, or share online access with different people in a group to disguise their unique query patterns.

## 5 CONCLUSIONS

DNS-based behavioral analysis is extremely attractive for many applications such as forensics investigation and online advertisement. In this paper, we have proposed the constrained Dirichlet multinomial mixture (CDMM) clustering model without requiring the specification of the number of clusters and meanwhile enforcing

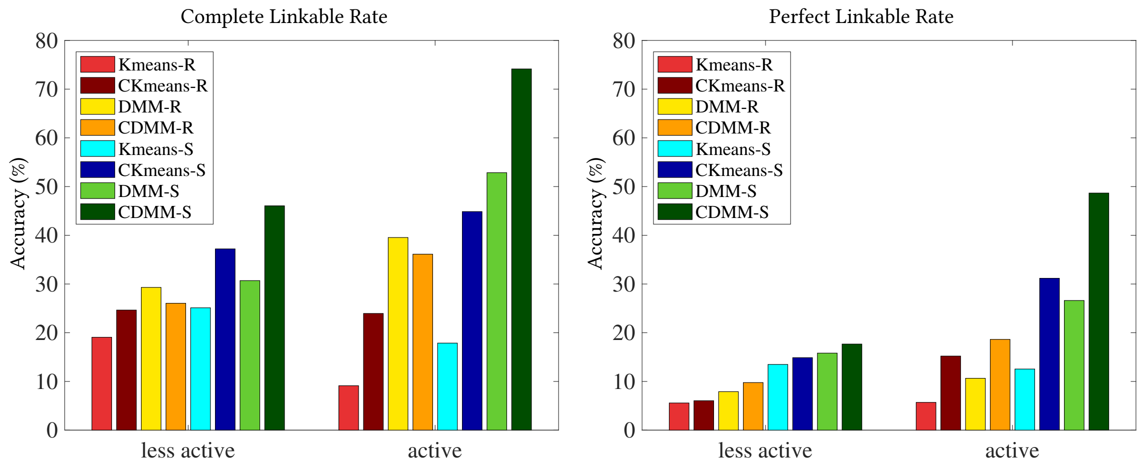

**Figure 2: Clustering performance of eight methods measured by complete linkable rate (left) and perfect linkable rate (right) with respect to different user activity groups in the observation period.**

the maximum number of data in each cluster, which fits the DNS tracking problems nicely. We have performed extensive evaluation based on DNS queries collected from real networks. Experimental results have demonstrated that our method accomplishes a high clustering accuracy and outperforms the existing methods.

## 6 ACKNOWLEDGEMENT

This work was supported in part by the Louisiana Board of Regents under Grant LEQSF(2017-20)-RD-A-29 and National Science Foundation under Grant 1560437.

## REFERENCES

[1] Bin Bi, Milad Shokouhi, Michal Kosinski, and Thore Graepel. 2013. Inferring the demographics of search users: social data meets search queries. In *Proceedings of the 22nd International Conference on World Wide Web*. 131–140.

[2] Gang Chen, Haiying Zhang, and Caiming Xiong. 2016. Maximum margin Dirichlet process mixtures for clustering. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 1491–1497.

[3] Shuo Chen, Rui Wang, XiaoFeng Wang, and Kehuan Zhang. 2010. Side-channel leaks in web applications: a reality today, a challenge tomorrow. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*. 191–206.

[4] Mauro Conti, Luigi V Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2015. Can't you hear me knocking: identification of user actions on Android apps via traffic analysis. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. 297–304.

[5] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. 2016. User profiling in the time of HTTPS. In *Proceedings of the 2016 ACM Internet Measurement Conference*. 373–379.

[6] Xiaodan Gu, Ming Yang, Jiaxuan Fei, Zhen Ling, and Junzhou Luo. 2015. A novel behavior-based tracking attack for user identification. In *Proceedings of the Third International Conference on Advanced Cloud and Big Data*. 227–233.

[7] Jinjin Guo and Zhiguo Gong. 2016. A nonparametric model for event discovery in the geospatial-temporal space. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*. 499–508.

[8] Dominik Herrmann, Christian Banse, and Hannes Federrath. 2013. Behavior-based tracking: exploiting characteristic patterns in DNS traffic. *Computers & Security* 39 (2013), 17–33.

[9] Sakshi Jain, Mobin Javed, and Vern Paxson. 2016. Towards mining latent client identifiers from network traffic. In *Proceedings of Privacy Enhancing Technologies Symposium*. 100–114.

[10] Dae Wook Kim and Junjie Zhang. 2015. You are how you query: deriving behavioral fingerprints from DNS traffic. In *Security and Privacy in Communication Networks*, Bhavani Thuraisingham, Xiaofeng Wang, and Vinod Yegneswaran (Eds.). Springer International Publishing, 348–366.

[11] Matthias Kirchler, Dominik Herrmann, Jens Lindemann, and Marius Kloft. 2016. Tracked without a trace: linking sessions of users by unsupervised learning of patterns in their DNS traffic. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. 23–34.

[12] Srinivas Krishnan and Fabian Monrose. 2010. DNS prefetching and its privacy implications: When good things go bad. In *Proceedings of the Third USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More*. 10–10.

[13] Marc Liberatore and Brian Neil Levine. 2006. Inferring the source of encrypted HTTP connections. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*. 255–263.

[14] Takashi Matsunaka, Akira Yamada, and Ayumu Kubota. 2013. Passive OS fingerprinting by DNS traffic analysis. In *Proceedings of the IEEE 27th International Conference on Advanced Information Networking and Applications*. 243–250.

[15] David Stillwell Michal Kosinski and Thore Graepel. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences* 110, 15 (2013), 5802–5805.

[16] Alan Mislove, Bimal Viswanath, Krishna P. Gummadi, and Peter Druschel. 2010. You are who you know: inferring user profiles in online social networks. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*. 251–260.

[17] Jeffrey Pang, Ben Greenstein, Ramakrishna Gummadi, Srinivasan Seshan, and David Wetherall. 2007. 802.11 user fingerprinting. In *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*. 99–110.

[18] William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *J. Amer. Statist. Assoc.* 66, 336 (1971), 846–850.

[19] Qixiang Sun, Daniel R. Simon, Yi-Min Wang, Wilf Russell, Venkata N. Padmanabhan, and Lili Qiu. 2002. Statistical identification of encrypted web browsing traffic. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*. 19–30.

[20] Charles V Wright, Lucas Ballard, Scott E Coull, Fabian Monrose, and Gerald M Masson. 2008. Spot me if you can: uncovering spoken phrases in encrypted VoIP conversations. In *Proceedings of the IEEE Symposium on Security and Privacy*. 35–49.

[21] Charles V Wright, Lucas Ballard, Fabian Monrose, and Gerald M Masson. 2007. Language identification of encrypted VoIP traffic: Alejandra y Roberto or Alice and Bob. In *Proceedings of the 16th USENIX Security Symposium*. 1–12.

[22] Jianhua Yin and Jianyong Wang. 2014. A Dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 233–242.

[23] Fan Zhang, Wenbo He, Xue Liu, and Patrick G. Bridges. 2011. Inferring users' online activities through traffic analysis. In *Proceedings of the 4th ACM Conference on Wireless Network Security*.

[24] Elena Zheleva and Lise Getoor. 2009. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th International Conference on World Wide Web*. 531–540.

[25] Yuan Zhong, Nicholas Jing Yuan, Wen Zhong, Fuzheng Zhang, and Xing Xie. 2015. You are where you go: inferring demographic attributes from location check-ins. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*. 295–304.