Cookieless Fingerprints Across Devices

Michael Fang James Hong Seokho Hong
{mjfang, jamesh93, seokho}@stanford.edu
Stanford University

I. Introduction

Many people own and use several internet connected devices on a daily basis, including desktops, laptops, smartphones, tablets, game consoles and smart TVs. Identifying the same user across multiple devices and platforms can allow for improved ad targeting and content recommendations, leading to higher click-through-rate and ad revenue for online video ad providers such as YuMe, Inc.

Traditional cookie-based approaches to identify users work only for individual devices. Similarly, identifying users by device fingerprints is ineffective when a single user owns devices across many platforms. Each platform OS has its own unique policies that confound direct cross-platform comparisons. In many cases, it is more convenient for companies to avoid this problem by tracking users through account logins.

In this paper, we explore multiple approaches to pairing devices to users on an 8-day window of data provided by YuMe. We discuss categorical clustering of devices using the ROCK algorithm (Guha, Rastogi & Shim, 1999), identifying households and re-creating household ids, and our main experiment with cross-platform device similarity metrics.

II. Data

For this project, we performed experiments on data provided by YuMe, gathered from April 6-14, 2014. The full dataset is over 3tb uncompressed and contains over 1 billion video ad sessions worldwide. It is separated into request and beacon data, containing categorical features such as IP address, time stamp, device make/model and vendor. Beacon and request data also provide content-based features such as advertisement cost, advertisement id and referral site. Also, YuMe provides several of their own synthesized features including household id and number of devices in household, derived from IP address.

The number of informative fields is limited. We extracted 23 fields from the data for our analyses; these are the only fields that contained variability and were non-sparse. When comparing devices across multiple platforms, the number of useful fields drops further to fewer than 10, of which 5 are derivatives of IP address. These are the only fields that are logged in a consistent manner across all platforms.

session id	os name	sdk version
referrer site*	os version name	requested date
publisher id*	browser name	cookie
network id*	browser version name	**IP address***

is prefetch request	**country code**	placement id*
delivery point id	**state code***	advertisement id*
device make name	**city name***	click(through)
device model name	**service provider name***	

Figure 1: the “informative” features we extracted from the dataset. Derivatives of IP have been **bolded**. Fields we used for cross device comparison are denoted with *.

III. Main Challenges

In all of our experiments, we faced the following challenges:

1. *Lack of ground truth*

   The dataset is unlabeled, making it difficult to validate our results reliably. This means that it is impossible to verify that two devices indeed belong to the same individual; we must rely on fields such as IP address and Household ID to validate.

   Moreover, for many devices, we have no reliable method to verify that multiple sessions indeed belong to the same device. For instance, the information contained in the OS name/version, browser id/version, browser SDK, cookie, etc. fields, was often insufficient to distinguish multiple PC sessions at the same IP address. Without a definitive hardware id field or browser fingerprint data for PCs, we used a heuristic and treated all similar sessions at an IP address as a single device.

2. *Correlated data*

   Of the few data fields that are directly comparable (recorded consistently) across all platforms in the YuMe dataset, six were derived from IP address. These included country, state, city, service provider, census-data DMA id, and household id (HID). In particular as our later experiments demonstrate, YuMe’s synthesized HID field corresponds to a naïve grouping of devices by shared IP addresses. This severely limits the number of useful, independent fields that are available for comparing devices of different platforms.

3. *Data/schema are platform specific*

   This challenge manifests in two forms.

   First, understanding the schema is non-trivial as the meaning of each field varies wildly with platform and device. For instance, for iOS devices, the cookie field provided a reliable means of tracking a single device. However, for PCs the cookie field is unique to each request. We found also found these discrepancies to be true for a portion of android devices and, specifically, Roku players. Given that the dataset includes thousands of device models, each with its own OS specific policies for interfacing with YuMe, inconsistent schema and documentation is a possible experimental confound. For our experiments, we made the simplifying assumption that the schema is consistent for the majority of sessions.

   Second, a more difficult challenge arises from the observation that the content based fields in the dataset cleave according to device type. Our categorical clustering and device pairing experiments affirm our suspicions that YuMe advertisements are targeted based on device type. Clustering by advertisement id, domain id, placement id, and publisher id, etc. results in a neat partition of the data into mobile/tablet and PC cohorts.

4. *Uninteresting/uninformative measures of user behavior*

   In our analyses, we are forced to make the assumption that user behavior (e.g. click-through, percent of ad watched) is comparable across different platforms. This assumption is clearly false; in fact, we approach any assumptions about user behavior with great skepticism. According to YuMe, mobile ads requests, representing a quarter of the dataset come from mobile apps using the YuMe ad delivery SDK. These requests are all labeled as "prefetch" in the dataset, meaning their request times and content cannot be considered user behavior or preferences. By contrast, YuMe's video ads on PC are requested when a user loads a video in a browser.

IV. Preliminary Experiments

Our project attempted to link devices to users in the YuMe dataset. To achieve this, we adopted two preliminary approaches: [1] a coarse household grouping algorithm by IP address and [2] clustering of devices based on categorical features in the data.

A. Households by IP Address

Our first intuition involved generating households using IP address, matching YuMe's approach for household id. To do this, we used a naïve algorithm. First, we bucketed ad sessions by IP address. Then, we iterated through the sessions assigned to each IP address and created device profiles. Finally, we outputted lists of devices at each IP based on whether the number of sessions at the IP exceeded a threshold (e.g. 10 sessions). In an improved version, we filtered devices at an IP address using timestamp variance, assuming that devices that connect regularly at varied times throughout the 8-day window are more likely to belong to that particular household.

While this experiment was simple, it aided our preliminary analysis tremendously. Our naïve threshold approach closely matched YuMe's HID upon manual inspection. Using timestamps improved results, lowering the number of devices that were falsely assigned to multiple households. In addition, an issue that we found was is that households tended to be small, containing 1.433 devices on average. With so few devices, in the majority of cases it's not possible to assume that individual users use YuMe services on multiple devices.

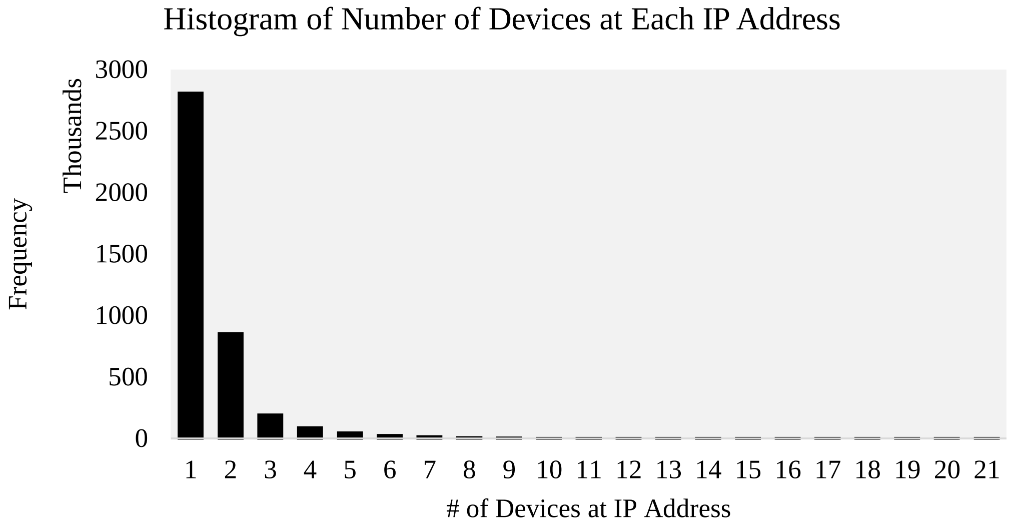

Figure 2: Frequency histogram of # of devices at an IP address. Most IP addresses contain only one device in the 8-day window. The maximum number of devices observed at a single IP address was 703.

B. Clustering of Categorical Data

We considered clustering as a method for finding groups of similar devices. The motivation behind this is to use an alternative method of finding similarity patterns which cannot be found in just simple device-to-device comparisons, but rather in the neighborhood of similar devices.

Our data was predominantly categorical, which presented difficulties for clustering devices since there is no notion of distance which can be used. We approached this difficulty with the ROCK algorithm by Guha, Rastogi and Shim (1999), a method of hierarchical clustering, which we will describe below.

Once we have assigned sessions to devices, we can consider its values in a particular field as items in a market basket. We can then compute similarity of devices, e.g. Jaccard. We call two devices neighbors if they have a similarity above a certain threshold t. Then, we define the link of two devices to be the number of neighbors the two devices have in common. So, the idea in the ROCK clustering algorithm is to, at each step, merge the two clusters which would maximize the number of links within each cluster while minimizing the number of links between clusters.

However, the efficacy of this method is based on the assumption that values within a field are not disjoint across devices. An additional concern is that ROCK does not scale to large data, $O(n^{2.37})$, making it useful only on small representative samples of data.

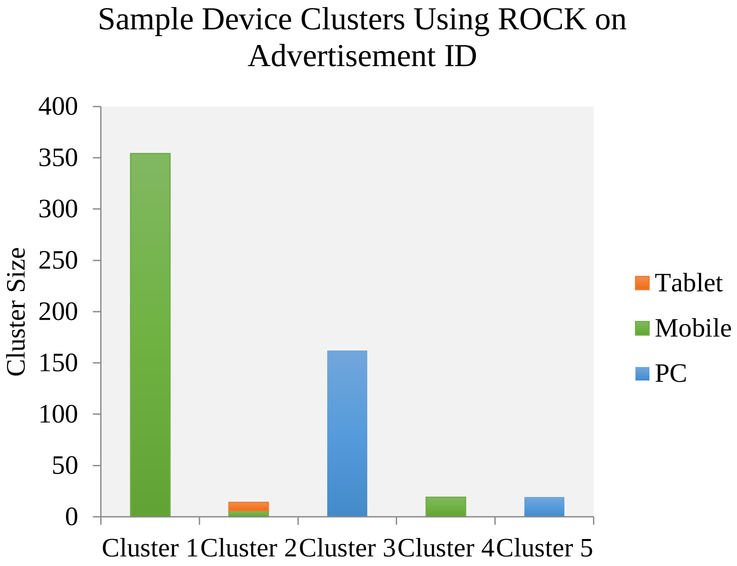

Figure 3. Results from clustering based on the ROCK algorithm using advertisement id**.** Sample data input contained 610 mobile devices, 240 PCs and 17 tablets, from a representative random sampling of devices. Because the clusters by advertisement id are homogenous by device type, advertisement id is not useful for cross device type comparisons. Considering that we expect few users to own multiple devices of the same type, this means advertisement id is not useful device comparisons in our main experiment (section V).

V. Main Experiment

To deal with the lack of ground truth, we define our evaluation method to provide a relative measurement of the effectiveness of any algorithm.

A. Method framework

We consider pairs of devices and attempt to classify the pair as 'owned by the same user' or 'owned by different users'. Since we have no ground truth, we define 'owned by the same user' in very simple ways. There are two ways we have defined it: the pair of devices are owned by one user if they overlap in at least one IP address, or the pair of devices are owned by one user if the Jaccard similarity between the sets of IP addresses exceeds a threshold. The rationale for these methods of labeling is they're the most obvious heuristics for whether a pair of devices is owned by one user and thus they are the safest basis for evaluating analysis of less straightforward data.

B. Algorithm

We define algorithm here as a method to classify a pair of devices as owned by the same user or not. Our algorithms produce feature vectors based on the data about the two devices and train a Support Vector Machine to optimize the classification. The better the SVM classifies on the training set, the more informative the features are, indicating a better algorithm and model.

C. Similarity metrics

Each algorithm consists of a vector of similarity metrics, each of which is a function of certain data fields for two devices. For example, the Naive IP comparator takes the set of IP addresses of the two devices and returns the Jaccard distance between the sets. An algorithm could use this Naive IP comparator as one of the similarity metrics in the feature vector alongside other similarity metrics.

(Naïve) IP	Returns Jaccard distance between the sets of IPs. (This metric was used only to evaluate.)
(Naïve) City	Returns Jaccard distance between the sets of cities.
Tracker Comparator	Compares the user’s response to advertisements. If two devices have watched advertisements for a similar duration of time, a higher number is returned.
Service Provider	Returns Jaccard distance between the sets of service providers.
Domain	Returns Jaccard distance between the sets of domains visited.
Time & IP	Similar to Naive IP, except it weights night-time IPs, presumably because devices are more likely to be at home at night-time so IP overlaps should more strongly correlate with the same user owning the two devices.
Time & City	Also weights sessions at night more strongly.
Exclude Mobile	Returns 0 if both devices are mobile, presuming that few people own two mobile devices.
Cosine Similarity IP	Returns the cosine similarity between the sets of IPs.
Cosine Similarity Referrer Site	Returns the cosine similarity between the sets of referrer sites.

Figure 4. Sample device-similarity functions.

Most of these similarity metrics do poorly on their own (single feature in the feature vectors), but performance increases in composite similarity metrics.

We also constructed similarity metrics on fields such as advertisement id, placement id, publisher id, and referrer site. These proved to be non-informative. In addition, in another experiment we applied the ROCK clustering algorithm within IP subnets and returned whether two devices were in the same cluster. Since we based ROCK clustering on advertisement id, placement id, etc., this too was non-informative.

D. Pairing

Comparing each device to every other device would require computation time quadratic to the number of devices considered, which is unfeasible. We therefore hash each device to a bucket before comparing all devices to each other within the bucket. The hash function is based on the subnet mask of one of the IP addresses of the devices. Using the subnet mask gives geographic locality to the buckets greatly increasing the chances that any pair of devices considered are owned by the same user, and eliminating much computation time. (subnet masking also does not invalidate using the full IP address for evaluation.) This pairing method operates in expected linear time, with acceptable tradeoffs in accuracy.

E. Scalability

The work on this paper was done with only devices that connected from within California. Within the California data, we subsampled buckets (as hashed by subnet masking). It is not necessary to use all the data since the end result is a classification accuracy measurement, which requires only so much data to achieve a reasonable level of precision.

With that said, our methods are still scalable, and could be run on every device in the dataset. As described previously, the subnet hashing reduces the number of pairs to a quantity approximately linear with respect to the number of devices. The algorithm for classification is also linear as the creation of the feature vectors and labeling is linear with respect to the number of pairs of devices. The SVM uses stochastic gradient descent for training and thus is linear with respect to the number of pairs of devices in the training subsample. Overall, our methods require expected linear time.

F. Results

Since our evaluation metric is based on a threshold, our final results will vary as the threshold moves. If two devices need to have a higher overlap in IP address to be considered as owned by the same user, fewer device pairs will be considered as belonging to one user. This makes it easier to classify and improves classification accuracy, but then we will incorrectly label true pairs as false negatives.

Figure 5 shows two similarity metrics on their own, alongside our best composite metric. Comparing the two devices' service providers and sites visited are two of the best standalone similarity metrics, but combined together and along with Tracker comparator, Exclude Mobile comparator, and Cosine similarity of Referrer site, the composite metric with learned weights achieves a higher f1 score than any individual metric.

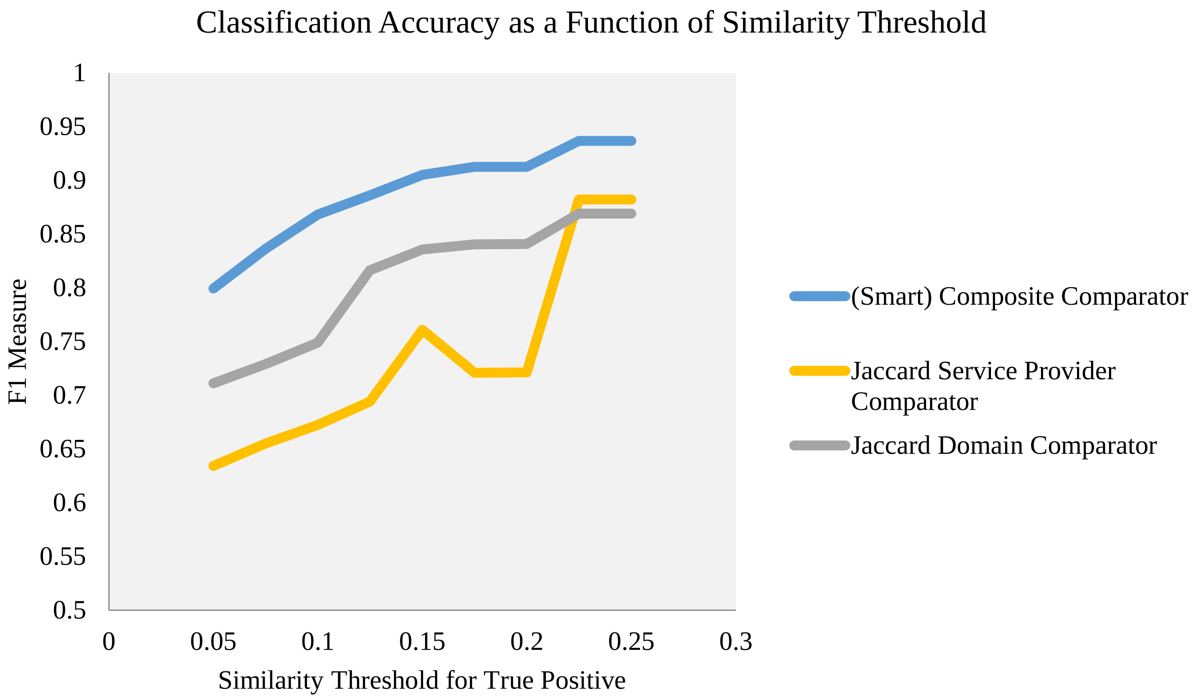

Figure 5. By increasing the overlap (Jaccard Distance) required to classify a device pair as owned by the same user, the classification accuracy increases. Our results also demonstrate that our composite comparator with learned weights performs above individual comparators.

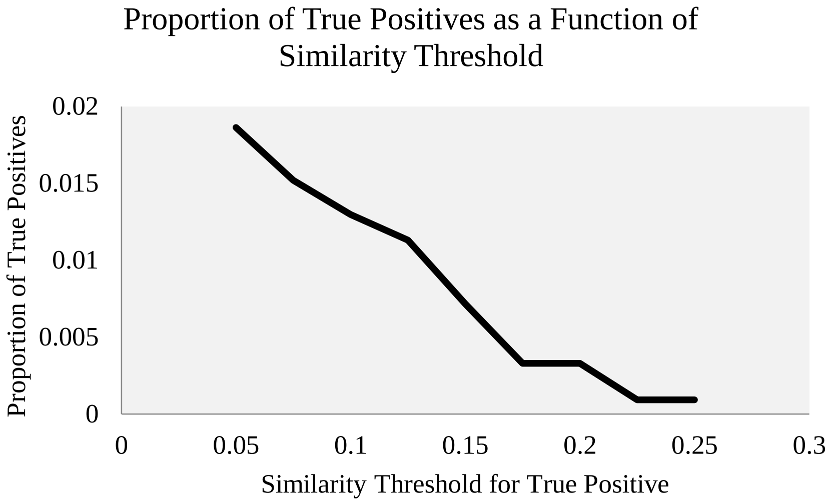

Figure 6. Increasing the device similarity threshold for a true positive on the evaluation metric reduces the proportion of true positives.

Similarity Metric	**F1-Score**
City	0.888
Cosine Referrer Site	0.664
Jaccard Service Provider	0.640
Jaccard Domain	0.782
Jaccard Ad ID	0.574
Cosine IP	0.952

Figure 7. Performance of some individual similarity metrics with the Similarity Threshold held constant at 0.1. As expected IP is the most informative feature of IP. But our besides IP, city is the next most informative field in the dataset.

VI. Discussion

The topic of cookie-less fingerprints across multiple devices remains relatively unexplored in academic literature. Related topics include de-anonymization and device fingerprinting for a single device.

Narayanan and Shmatikov (2008) present a statistical approach to de-anonymizing large sparse datasets. To achieve this, Narayanan and Shmatikov exploit the sparsity of “micro-data” such as user preferences, video recommendations and transaction records of IMDB profiles to deduce user profiles in the Netflix Prize Dataset. In principle, our task of identifying the same user across devices is similar. However, the features in the YuMe dataset failed to reflect user behavior ad preferences, and were incomparable across device platforms. Also, compared to Netflix’s recommendation system, the ad targeting systems used by YuMe are either more rudimentary or obfuscated by the lack of meta-information associated with fields such as advertisement id, referrer site, etc.

Existing work on device fingerprinting has focused on device features such as OS fonts, browser plugins, browser performance, detecting proxies, and hardware-specific features

(Nikiforakis et al. 2013). These approaches are generally inapplicable to fingerprinting across multiple devices, especially devices of varying platforms. Results in this sector are confounded by different browser and OS specific fonts, application, and plugin policies. Furthermore, with the exception of PC data, YuMe's requests come from within apps, with limited permissions. Ultimately, fingerprinting by hardware and OS features allowed us to guess individual devices at a particular IP when concrete hardware id fields were not available or unreliable.

VII. Conclusion

Unfortunately we cannot draw strong conclusions because of the lack of ground truth. However, the framework we created can still measure the relative information content of different fields and its potential relevance by using them to predict IP, our approximated ground truth. We also have confirmed that a limited subset of the different fields add non-overlapping information because composite similarity metrics are more informative than individual metrics.

VIII. Future Directions

In our evaluation, we assumed IP address to be ground truth for pairing multiple devices. For an average size household, this is a reasonable assumption as less than 1.6% of IP addresses have more than 5 devices associated to them. Unfortunately, for households of this size, we cannot guarantee that the devices are indeed owned by the same user within the household.

To improve results correlating two users to the same device, we require better documentation of the existing data and more informative fields. Some possible improvements are:

1. *Provide a mapping from advertisement id to the actual advertisement*

   Because advertisement ids are correlated with type of device, they are not useful in their present form for cross platform similarity measurements. Providing information or a means to obtain information about advertising ids such as genre, age target, etc. would improve modeling of user behavior.

2. *For mobile sessions, provide the name of the app performing a request*

   Similarly, basic information about the app making pre-fetch requests would go a long way to identifying users by their media consumption habits. Since the number of devices per household is small, knowing features such as gender and age group would allow for effective disambiguation of mobile users at an IP address.

3. *Provide a mapping from content id to content meta-information*

   This is the equivalent to the above except for PC devices. Logging information such as the site from which the request originated, would allow for device generalized feature extraction.

Ultimately, hardware features and IP address are useful for detecting devices and households, but more device-type independent features are crucial to discerning individual users.

IX. Acknowledgements

We would like to acknowledge our mentor, Professor Anand Rajaraman, for his role in guiding our project; Vijay Kaushik and Sachin Gupta for their role in helping us understand the dataset; and YuMe for providing the dataset.

X. References

[1] Boriah, S., Chandola, V., & Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. *red*, *30*(2), 3.

[2] Guha, S., Rastogi, R., & Shim, K. (1999, March). ROCK: A robust clustering algorithm for categorical attributes. In *Data Engineering, 1999. Proceedings., 15th International Conference on* (pp. 512-521). IEEE.

[3] Narayanan, A., & Shmatikov, V. (2008, May). Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on* (pp. 111-125). IEEE.

[4] Nikiforakis, N., Kapravelos, A., Joosen, W., Kruegel, C., Piessens, F., & Vigna, G. (2013, May). Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Security and Privacy (SP), 2013 IEEE Symposium on* (pp. 541-555). IEEE.