Enhanced Feature Selection and Generation for 802.11 User Identification

Dingbang Xu
Department of Computer Science
Governors State University
University Park, Illinois, USA
Email: d-xu@govst.edu

Yu Wang
Department of Computer Science
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
Email: yu.wang@uncc.edu

Xinghua Shi
Department of Computer Science
University of Chicago
Chicago, Illinois, USA
Email: shi@uchicago.edu

***Abstract*— To provide user privacy, several anonymization techniques (e.g., pseudonyms applied to MAC addresses) have been proposed in 802.11 networks. However, recent research done by Pang et al. [1] has demonstrated that pseudonyms are not adequate to protect user privacy. The key idea of Pang et al.'s method is to locate implicit identifiers (e.g., IP addresses and port numbers a user frequently visits), build user profiles based on these implicit identifiers in the training data sets, and then apply classification techniques to identify unlabeled (testing) users.**

Our method proposed in this paper partly focuses on building user profiles. Compared with the method proposed by Pang et al. in [1], we propose a novel approach to selecting and generating features, which is critical to build user profiles. The feature selection and generation procedure can be dynamically controlled through setting a few important parameters. We did a series of simulations using 9.27GB SIGCOMM 2004 wireless data sets, and our simulation results demonstrate better classification rates compared with Pang et al.'s method.

I. Introduction

In recent years, various wireless networks have been deployed to support a wide variety of applications. The broadcast nature of wireless networks usually makes it challenging to protect a user's privacy including locations and identities. For example, in a wireless sensor network, the wireless signal strength may partially disclose a user's location, and in an 802.11 wireless network, the MAC address embedded in every frame may partially disclose the user's identity because of the uniqueness of each MAC address. To help protect users' privacy, extensive research on privacy issues has been conducted in different types of wireless networks. For example, Gruteser and Grunwald [2] propose "disposable" MAC addresses to prevent wireless users being continuously tracked. Jiang, Wang and Hu [3] propose obfuscation techniques applying to user identities and transmission time. Along with other techniques such as [4] and [5], they fall into a broader category of *pseudonym* techniques. The basic idea of these pseudonym techniques is to anonymize the user or device identity through frequently changing pseudonyms. For example, the MAC addresses of wireless devices can be periodically changed to new, unused pseudonyms to prevent users being tracked.

Though these pseudonym techniques are effective to certain extent, they cannot completely solve the privacy issues in wireless networks. To be more specific, Pang et al. [1] demonstrate that attackers may identify 64% of all users within one day at a spot where 25 users are served simultaneously in every hour. Pang et al.'s approach is based on the concept of *implicit identifiers*. The implicit identifiers are those identifiers that may potentially disclose a user's identity. For example, given a group of 100 users accessing wireless networks in a conference with only one user from University of Chicago, if the wireless trace recorded in the conference shows that certain University of Chicago email servers and web servers are frequently visited by some MAC addresses, then the activities from these seemingly unrelated MAC addresses are likely to be correlated to the only user from University of Chicago. Here the IP addresses of those frequently visited email servers and web servers along with their port numbers are implicit identifiers. After identifying certain implicit identifiers, Pang et al. further propose to build user profiles based on implicit identifiers in training data sets, and apply data classification techniques to identify users. More details of their method will be reviewed in Section III.

Our method proposed in Section IV is partially inspired by Pang et al.'s method [1]. In particular, we also use the concept of implicit identifiers. However, instead of generating one feature from one implicit identifier in Pang et al.'s method, we propose a novel way to dynamically select and generate multiple features from one implicit identifier, and this dynamic feature selection and generation procedure can be controlled through setting up a few important parameters. Our approach has three major steps: *feature selection*, *feature generation*, and *classification*. Feature selection decides how many features will be used, and what these features are. Feature generation algorithm processes sample data sets, generate feature values, and then further combine these feature values based on predefined parameters. In the last step, the combined feature values are fed into classification models for training/testing purposes. Since many classification models such as naive Bayes [6] are well defined, we will focus on feature selection and feature generation algorithms in this paper. We also did an extensive number of simulations using 9.27 GB SIGCOMM 2004 data sets [7] with $49,830,432$ wireless packets. Our simulation results in Section V show much improved classification rates compared with Pang et al.'s method [1].

The work of Y. Wang is supported in part by the US National Science Foundation (NSF) under Grant No. CNS-0721666.

II. Related Work

Privacy has become a big concern in recent years. Privacy issues in different types of wireless networks such as wireless LANs [2], [3], RFID [8], Bluetooth networks [5], sensor networks [9], [10], or general pervasive computing [4] have been extensively studied. The most common solution for preserving location privacy in wireless networks is pseudonym techniques, which are proposed by several researchers such as those in [2]–[5]. The basic idea is to anonymize the user or device identity with frequently changed pseudonyms.

However, recent research [1], [11], [12] demonstrates that pseudonyms are not adequate to provide anonymity in certain wireless networks. Even without a unique MAC address, attackers may still use different *implicit identifiers* to identify devices or users (wireless fingerprinting techniques). There are two kinds of implicit identifiers: hardware-based or software-based. Hardware-based fingerprinting (also called RF fingerprinting) [13], [14] uses physical characteristics of RF transmission (such as signal power attenuation, modulation accuracy, or waveform accuracy) to differentiate messages from different radios or different hardware devices. On the other hand, software-based fingerprinting [1], [11], [12], [15] uses the differences in software configurations to distinguish network nodes. For example, Kohno *et al.* [11] estimate the clock skew using TCP and ICMP timestamps to fingerprint network devices. Franklin et al. [12] and Desmond et al. [15] use statistical analysis of the inter-frame timing of transmitted probe request frames to detect devices. Pang et al. [1] study techniques to identify users based on the patterns of the wireless traffic such as network destinations (IP addresses and port numbers) a user frequently visits. In this paper, we focus on improving Pang et al.'s method by proposing new, enhanced feature selection and generation algorithms.

Some applications may apply machine learning technique to build implicit identifiers, for example, Internet traffic classification [16]–[18] and web user identification [19] (based on timing and sizes of web transfer). Due to the space constraint, we do not discuss them here.

III. 802.11 User Identification by Pang et al. [1]

Our approach on identifying 802.11 users is partially inspired by the method proposed by Pang et al. [1]. Though we use the similar implicit identifiers such as *network destinations*, we use a totally different approach to selecting and generating features before classification models are applied. To help clarify the difference between our approach and Pang et al.'s approach [1], we give an overview of their method.

Pang et al.'s method [1] assumes that pseudonyms are applied to wireless devices, which means users can change the MAC addresses of their wireless devices at a reasonable time interval (e.g., once an hour). Under this assumption, it is usually not feasible to track/identify users through discovering the correlation among those frequently changed MAC addresses. Instead, to identify different users, Pang et al. propose to locate implicit identifiers (e.g., the IP addresses and port numbers a user frequently visits), build users profiles based on these implicit identifiers in the training data sets, and then apply classification techniques to differentiate users.

There are four implicit identifiers proposed in Pang et al.'s method [1]: *Network Destination*, *SSID*, *Broadcast Packet Size*, and *MAC Protocol Field*. A network destination is a pair of ⟨IP, Port⟩ that a user visits. SSID is Service Set Identifier, which is a name broadcast by wireless access points to differentiate wireless LANs. Broadcast packet sizes are the sizes of 802.11 broadcast packets. For example, 276 is the broadcast packet size related to NETBIOS service (UDP port 137). MAC protocol fields represent the fields in MAC frame headers. In 802.11 wireless LANs, there are several protocol fields, for example, *more data*, *order flag*, and *protected flag*.

All these four implicit identifiers are used in classification. More specifically, each implicit identifier will be transformed into one feature, and then assuming that four features are independent, a naive Bayes classifier is applied to identify users. The procedure to generate one feature from one implicit identifier is as follows. Given an implicit identifier $\mathcal{I}$ and a user u, assume that the set of all possible values of $\mathcal{I}$ for u in a training set is S. Then for a sample S_u, the corresponding feature f is computed as

$$f = \frac{\sum_{t \in (S \cap S_u)} weight(t)}{\sum_{t \in (S \cup S_u)} weight(t)},$$

where $weight(t) = 1/$(the number of users in S with value t).

IV. Enhanced Feature Selection and Generation

Before we present the details of our method, we first informally define a few terms. An *implicit identifier* is a characteristic or a combination of characteristics that may potentially be used to differentiate users. A *feature* is a variable used in the classification model. Given n features f_1, f_2, $\cdots$, f_n, and a special class variable f_c representing the class label, classification, informally speaking, decides the value of f_c based on the values of f_1, f_2, $\cdots$, f_n. Features may represent feature names or feature values depending on the context. Compared with implicit identifiers or features, *attributes* may represent a broader range of concepts. For example, attributes may represent features in classification, or certain field data in network packets, depending on the context.

Our method is partially inspired by Pang et al.'s method [1]. We also use those implicit identifiers such as network destinations and SSIDs for wireless networks. However, when building user profiles, instead of transforming each implicit identifier into one feature in the classification model, we propose a novel way to dynamically generate multiple features from one implicit identifier, and this dynamic feature generation process can be controlled through setting up a few important parameters. Our simulation result shows improved classification rates compared with the method in [1].

Roughly speaking, there are two critical algorithms in our method: one is feature selection, and the other is feature generation. Feature selection decides how many features will be used, and what these features are. Based on these selected features, the feature generation algorithm processes sample data sets, generate feature values (i.e., attribute values), and then further group these feature values to reduce the number of feature dimensions. Finally, these grouped feature values will be fed into classification models for training/testing purposes.

To select features for any implicit identifier (e.g., network destination), we first locate the attributes (or attribute combinations) corresponding to the implicit identifier. For example, implicit identifier network destination has an corresponding attribute combination $\langle$IP, Port$\rangle$, where these attribute values can be extracted from packet data sets. Next we list all different attribute values for this implicit identifier, and sort them in terms of certain criteria. Lastly, certain number of these attribute values in the sorted list are selected, and each attribute value is transformed into one feature. The details of this feature selection procedure is given in Algorithm 1.

In Algorithm 1, Lines 1 to 3 are used to select candidate features. Specifically, Line 2 extracts all different attribute values in the data set corresponding to the implicit identifier, and Line 3 further transforms these attribute values into the same number of features. For example, in a data set, for the attribute combination $\langle$IP, Port$\rangle$, suppose there are two pairs of different values $\langle 123.123.123.123, 80\rangle$ and $\langle 156.156.156.156, 445\rangle$, we can transform them into two features with the names "$IP = 123.123.123.123, Port = 80$" and "$IP = 156.156.156.156, Port = 445$", respectively.

Lines 4 to 9 in Algorithm 1 sort the candidate feature list based on index values. Our design for this index value is that it should represent a user's consistent behavior, and also is able to differentiate users. With this design goal, there may exist several different ways to compute index values. As an example, Algorithm 1 shows one approach. Specifically, given any attribute value v'_i, the index value is computed based on the number of the packets having v'_i and the number of users accessing v'_i. Intuitively, if a user's behavior is consistent (e.g., frequently visiting certain web sites), we should be able to observe the corresponding attribute values repeatedly. Additionally, in certain ranges, the lesser the number of users having the same attribute values, the easier the users may be differentiated. This explains why we let index value $Ind(v'_i) = \frac{PC(v'_i)}{UC(v'_i)}$. Nevertheless, other factors may be incorporated into index value computation, for example, the frequency of attribute values in an unit time duration (e.g., 1 hour). After these index values are computed for all candidate features, Line 9 further sorts them based on index values.

In Algorithm 1, the actual feature selection is performed from Lines 10 to 12. The basic idea here is to select top n features in term of index values. However, we also notice that there may exist some extreme cases, for example, based on the analysis to SIGCOMM 2004 wireless data sets, we observe that most users access the wireless network with SSID *sigcomm-nat*. Though there are huge numbers of packets with SSID *sigcomm-nat*, and the index value for this SSID is also large, this feature does not differentiate users well, because almost every user accesses it. To rule out these extreme cases, we further require that $UC(v_i) < T_{uc}$, where T_{uc} is a pre-set threshold. Notice that the selection of T_{uc} may be critical. If T_{uc} is too high, the selected features may be too common to identify users, but if T_{uc} is too low, the number of selected features may be too small.

Algorithm 1 Feature Selection for One Implicit Identifier

```
Input: A packet dataset $\mathcal{S}$, an implicit identifier $\mathcal{I}$, a user count threshold $T_{uc}$, and an integer $n$ (i.e., the number of features to be selected).
Output: $n$ features $\mathcal{F}^*_n$ for $\mathcal{I}$.
 1: {Candidate Features:}
 2: In $\mathcal{S}$, select a complete list $\mathcal{L}'$ of all different values of the attribute (or attributes) corresponding to implicit identifier $\mathcal{I}$. Let $\mathcal{L}' = \{v'_1, v'_2, \cdots, v'_m\}$ where $m$ is the number of different values of that attribute in $\mathcal{S}$.
 3: For each $v'_i \in \mathcal{L}'$, we generate a candidate feature $f'_i$ which is an indicator whether the value of $v'_i$ exists in the dataset for implicit identifier $\mathcal{I}$. Let $\mathcal{F}'$ be the set of all candidate features, i.e., $\mathcal{F}' = \{f'_1, f'_2, \cdots, f'_m\}$.
 4: {Sorting of Features:}
 5: for all value $v'_i$ in $\mathcal{L}'$ do
 6:     Count $PC(v'_i)$, which is the number of packets with value $v'_i$ in $S$.
 7:     Count $UC(v'_i)$, which is the number of users in $S$ who have packets with value $v'_i$.
 8:     Compute index value $Ind(v'_i) = \frac{PC(v'_i)}{UC(v'_i)}$.
 9: Decreasingly sort all values $v'_i$ in $\mathcal{L}'$ and their corresponding features $f'_i$ in $\mathcal{F}'$ based on the index values $Ind(v'_i)$. Let the ordered lists be $\mathcal{L} = \{v_1, v_2, \cdots, v_m\}$ and $\mathcal{F} = \{f_1, f_2, \cdots, f_m\}$ respectively.
10: {Selection of Features:}
11: Select a list $\mathcal{F}^*_n$ of top $n$ features in the ordered list $\mathcal{F}$, where for each feature $f_i$, its corresponding value $v_i$ in $\mathcal{L}$ need to satisfy $UC(v_i) < T_{uc}$.
12: Return $\mathcal{F}^*_n$.
```

Algorithm 1 selects those features that are frequently accessed by users. To identify individual users based on these features, we further apply classification techniques in data mining. However, before classification, we first need to generate feature values for the features selected in Algorithm 1. Algorithm 2 is proposed to generate features (i.e., feature values) based on a user's packet data set. In Algorithm 2, we first generate a bit vector based on the packet data in a unit time period (e.g., 1 hour), then reduce the dimensions of bit vectors through combining multiple bits into decimal values.

In Algorithm 2, Steps 1 through 5 are used to compute feature values in a bit vector format. This bit vector is initialized to all 0s in Step 2. Next, each feature is examined based on packet data. If there are some packets having the attribute values corresponding to a feature, then this feature's bit in the bit vector will be set to 1. For example, suppose we have two features "$IP = 123.123.123.123, Port = 80$" and "$IP = 156.156.156.156, Port = 445$." We examine the packet data set, and find some packets with IP address 123.123.123.123 and port number 80, but there are no packets with IP address "156.156.156.156 and port number 445, then the corresponding bit vector will be set to $\langle 10 \rangle$. In our implementation of this part of Algorithm 2, we build a hash table, which provides mapping between features selected in Algorithm 1 and their indices in the bit vector. This hash table greatly improves the performance of bit setting.

Steps 6 through 10 in Algorithm 2 are used to group the bits in the bit vectors so that the large number of bits can be combined into fewer number of bit groups, and these bit groups are further transformed into decimal numbers. The purpose of these steps is to reduce data dimension, which can facilitate data classification [20]. As an example, let us assume that we have a bit vector with 15 bits $\langle 001010011000011 \rangle$. Further assume that we want to group these 15 bits into 3 groups, with 5 bits each. Bit vector $\langle 001010011000011 \rangle$ is grouped into $\langle 00101, 00110, 00011 \rangle$, which will be further transformed into a decimal vector $\langle 5, 6, 3 \rangle$.

After we get these decimal vectors (i.e., feature values) with reduced dimension, we put the data into classification models for either training or testing. The identification of individual users will be performed through classification algorithms, which have been extensively studied with a rich literature.

Algorithm 2 Feature Generation for One Implicit Identifier

```
Input: A packet dataset $\mathcal{S}_u$ from a user $u$ in a unit time period, an implicit identifier $\mathcal{I}$, $n$ features $\mathcal{F}_n^*$ (corresponding to $\mathcal{I}$), and a group number $g$.
Output: A set of feature values $\mathcal{F}_g^* = \{f_1^*, f_2^*, \cdots, f_g^*\}$ .
1:  {Calculation of Feature Values:}
2:  Initialize a $n$-dimensional vector $X$ with every dimension being 0, i.e., $X = \langle x_1 x_2 \cdots x_n \rangle$ where all $x_i = 0$.
3:  for all feature $f_i$ in $\mathcal{F}_n^*$ do
4:      if any packet in $S_u$ has the attribute value $v_i$ which is corresponding to $f_i$ then
5:          Set $x_i$ be 1.
6:  {Grouping of Feature Values:}
7:  Partition the vector $X$ into $g$ groups, $X_1, X_2, \cdots, X_g$, where each group $X_i$ can be treated as a $\lceil \frac{n}{g} \rceil$-bit binary number. (The last group may have less bits).
8:  for $i = 1$ to $g$ do
9:      Transform the binary number $X_i$ into a decimal value $f_i^*$, i.e., $f_i^* = int(X_i)$.
10: Return $\mathcal{F}_g^*$.
```

V. Simulations

Simulation Setup. To evaluate the effectiveness of our method, we did a series of simulations using SIGCOMM 2004 802.11 network data sets [7] (around 9.27 GB wireless packet dump files were processed in the simulations), which were collected through wireless monitors using tcpdump during SIGCOMM 2004 conference. We summarize the data set being used in Table I. In addition, we use Wireshark (`http://www.wireshark.org/`) to pre-process wireless packets, and use Microsoft SQL Server to store packet data as well as data processing results. We implement both Pang et al.'s method [1] and our method using JAVA. We use two implicit identifiers: network destination and SSID. To help us perform data classification, we use WEKA, a data mining software package implemented in JAVA [6] (`http://www.cs.waikato.ac.nz/ml/weka/`). Similar as in Pang et al.'s method, we also assume that pseudonyms are applied to 802.11 devices, which means each user's wireless MAC address can be changed at a reasonable time interval. In our simulation, we assume that pseudonyms are applied once an hour, and we partition each user's wireless trace into multiple subsets with one hour time duration for each subset. We then process each subset to generate feature values for classification purpose (identifying users).

TABLE I
Simulation data set summary

Data source	Wireless traces tcpdumped in SIGCOMM 2004
Data size	9.27GB
# Packets processed	$49, 830, 432$
Implicit identifiers	Network Destination & SSID
Timestamp of first packet	9AM 08/31/2004
Timestamp of last packet	7PM 09/01/2004
Training time period	9AM 08/31/2004 to 7PM 08/31/2004 & 9AM 09/01/2004 to 7PM 09/01/2004
Testing time period	9AM 09/01/2004 to 7PM 09/01/2004
# Training/Testing users	10 most active users

We notice that in user classification, training/testing approaches have significant impact on our simulation results, so before we present simulation results, we first clarify three types of training/testing being applied in our simulations: Type A, Type B, and Type C training/testing.

- Type A training/testing. In this case, we only differentiate two class labels: *Label u* denoting that the instances are from *User u*, and *Label* $\bar{u}$ denoting that the instances are *not* from *User u*. During training/testing, Label u instances are from user u, and Label $\bar{u}$ instances are *randomly* selected instances from non-u users.
- Type B training/testing. Similar to Type A training/testing, Type B training/testing also only differentiates two class labels: *Label u* and *Label* $\bar{u}$. During training/testing, Label u instances are from user u, however, label $\bar{u}$ instances are from *one* non-u user.
- Type C training/testing. In this case, we differentiate multiple class labels: *Label* u_1 denoting the instances are from *user* u_1, *Label* u_2 denoting the instances are from *user* u_2, $\cdots$, and *Label* u_k denoting the instances are from *user* u_k. During training/testing, label u_i ($1 \le i \le k$) instances are from user u_i.

In addition, to compare the classification results, we also define *classification rate of a user u* as CR_u= $\frac{\#\text{correctly classified instances}}{\#\text{total instances classified}}$, which is a value between 0 to 1.

As mentioned in Table I, the data set in our simulations spans two days (08/31/2004 and 09/01/2004). Our training uses both days, but only the second day is used for testing.

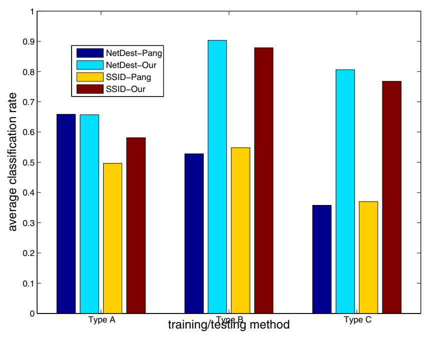


Fig. 1. Comparison between Pang et al.'s method and our method.

In addition, for all our simulations, we use all available packet data for feature selection (Algorithm 1), and then select 10 most active users (in terms of wireless activities) for training/testing (features generated by Algorithm 2) and compute their average classification rates. For Type B (or Type C) training/testing, we use 10 groups of 2 (or 3) users from the 10 most active users.

Simulation Results. We did five sets of simulations to evaluate the performance of our method. For all these simulations, average classification rates related to those 10 most active users are plotted in a series of figures (Figures 1, 2(a), 2(b), 2(c), and 3). All three types of training/testing are executed, and we observe consistent trends in Type B and Type C training/testing. However, the results from Type A training/testing may or may not demonstrate the same trends as those in Type B and Type C training/testing, largely because of the random instance selection performed in Type A training/testing. In the following, we report the results for all these five sets of simulations, mainly focusing on the observations from Type B and Type C training/testing. However, for the completeness of the simulations, Type A training/testing results are also plotted.

Our first set of simulations is to compare the effectiveness of Pang et al.'s method and our method. For our method, we set the number of features $n = 100$, the user count threshold $T_{uc} = 36$, and the number of groups $g = 10$. The result is shown in Figure 1. In Figure 1, we observe that our method outperforms Pang et al.'s method in Type A, Type B, and Type C training/testing for both implicit identifiers (The only exception is Type A training for network destination, where our result is similar to the one in Pang et al.'s method). Particularly, For network destination implicit identifier in Type C training/testing, our method has the average classification rate of 0.806. Compared with 0.358 from Pang et al.'s method, our method increases the average classification rate by 125.1%.

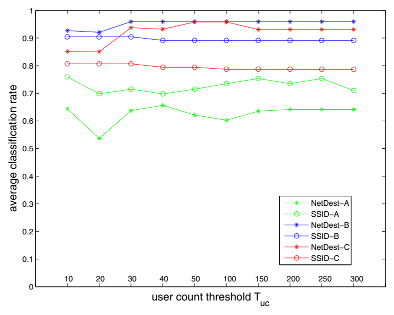
After the first set of simulations confirms that our method is effective in identifying users, we would like to explore our method more deeply. Specifically, we did another four sets of simulations to evaluate our method by setting different parameters.

Our second set of simulations focuses on feature selection (Algorithm 1) with different threshold T_{uc}. We set the number of features $n = 100$, the number of groups $g = 10$, and change the user count thresholds T_{uc} with different values from 10 to 300. The average classification rates for these different T_{uc} values are shown in Figure 2(a). We observe that different user count thresholds do affect the classification rates. Setting an appropriate T_{uc} value may increase the rate of user identification. For example, in Type C training/testing for network destination, setting T_{uc} to either 50 or 100 may bring a higher classification rate.

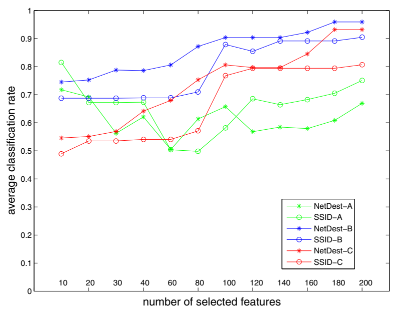
Our third set of simulations is still related to feature selection, but this time we concentrate on the number of features to be selected (i.e., the integer n in Algorithm 1). We set the user count threshold $T_{uc} = 36$, and the number of groups $g = 10$. And then we change the number of features n with different values from 10 to 200. The average classification rates for these different n values are shown in Figure 2(b). We observe that different numbers of features can affect classification rates. General speaking, the more features we choose, the better classification results we obtain. This observation is consistent with our intuition. More features selected means that user behaviors can be more precisely (more fine-grained) captured, which usually may bring better results. However, we also notice that Type A training/testing may not necessarily follow this trend. This is because that some instances are randomly selected in Type A training/testing.

Our fourth set of simulations focuses on feature generation. We want to study how group number g (in Algorithm 2) affects classification results. In this set of simulations, we set the number of features $n = 100$, and the user count threshold $T_{uc} = 36$. Then we set g into different values ($4, 5, 6, 8, 10, 12, 13,$ and 15). The average classification rates for these different group numbers are shown in Figure 2(c). We observe that classification rates are affected by group number g. In certain settings, higher values of group number g may result in higher classification rates. For example, in Type C training/testing for network destination, setting g to 15 has a better average classification rate compared with setting g to 10. However, we also need to notice that this may not always be the case, and it largely may be data set dependent.

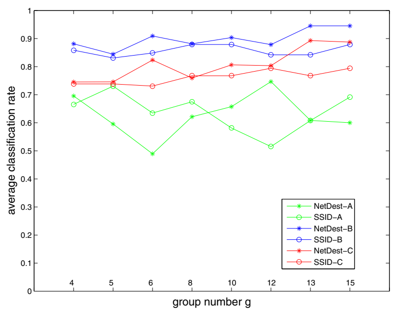
Our last set of simulations studies how different data classification methods affect user identification. Similar to the previous simulations, we set the number of features $n = 100$, the user count threshold $T_{uc} = 36$, and the number of groups $g = 10$. Then we select eight classification methods in WEKA: NaiveBayes, BayesNet, AdaBoostM1, MultilayerPerceptron, DecisionTable, IBk, J48, and SMO (The details of these classification methods can be found in [6]). The average classification rates for these different classification methods are shown in Figure 3. We observe that different classification methods may bring different average classification rates under the same data set. For example, in Type B and Type C training/testing for network destination, the classification method IBk outperforms all the other seven classification methods. But in Type B training/testing for SSID, we do not see big differences among different classification methods. This tells us that classification rates are dependent on classification methods as well as data sets.



(a) threshold T_{uc}

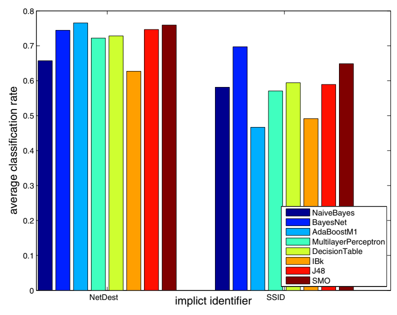


(b) number of features n

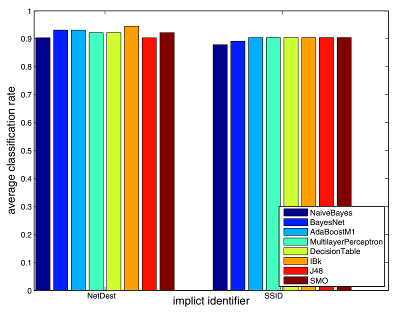


(c) group numbers g

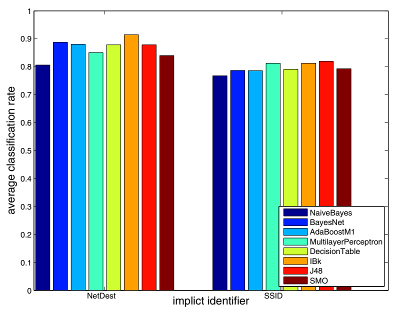
Fig. 2. Results using different threshold T_{uc}, number of features n, or group numbers g.



(a) Type A



(b) Type B



(c) Type C

Fig. 3. Results using different classification methods.

VI. Conclusions and Future Work

In this paper, based on the concept of implicit identifiers, we propose a novel approach to selecting and generating features, and thus users can be identified through data classification. Our extensive simulations on 9.27 GB SIGCOMM 2004 wireless trace with $49,830,432$ packets demonstrate improved classification rates compared with Pang et al.'s method [1].

There are several directions we will further explore in our future work. Our simulation results preliminarily demonstrate the relationships among different parameters such as *the number of selected features* n, *user count threshold* T_{uc}, and *group number* g. We will look for more data sets for simulations to study the relationships among them. Secondly, since our method is based on the concept of implicit identifiers, we will explore and evaluate other implicit identifiers in our future work. Thirdly, we will investigate methods for improving the user privacy even though the implicit identifier based identification is applied.

References

[1] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall, "802.11 user fingerprinting," in *Proc. of ACM MobiCom*, 2007.

[2] M. Gruteser and D. Grunwald, "Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis," *Mob. Netw. Appl.*, vol. 10, no. 3, pp. 315–325, 2005.

[3] T. Jiang, H.J. Wang, and Y.-C. Hu, "Preserving location privacy in wireless LANs," in *Proc. of ACM MobiSys*, 2007.

[4] A.R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 46–55, 2003.

[5] F.-L. Wong and F. Stajano, "Location privacy in bluetooth," in *Proc. of ESAS*, 2005.

[6] I.H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, Morgan Kaufmann, San Francisco, USA, 2005.

[7] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, J. Zahorjan, and E. Lazowska, "CRAWDAD data set uw/sigcomm2004 (v. 2006-10-17)," http://crawdad.cs.dartmouth.edu/uw/sigcomm2004, 2006.

[8] S.L. Garfinkel, A. Juels, and R. Pappu, "RFID privacy: An overview of problems and proposed solutions," *IEEE Security and Privacy*, vol. 3, no. 3, pp. 34–43, 2005.

[9] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source-location privacy in sensor network routing," in *Proc. of ICDCS*, 2005.

[10] K. Mehta, D. Liu, and M. Wright, "Location privacy in sensor networks against a global eavesdropper," in *Proc. of IEEE ICNP*, 2007.

[11] T. Kohno, A. Broido, and K.C. Claffy, "Remote physical device fingerprinting," in *Proc. of IEEE Symp. on Security and Privacy*, 2005.

[12] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J.V. Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting," in *Proc. of USENIX-SS*, 2006.

[13] D.B. Faria and D.R. Cheriton, "Detecting identity-based attacks in wireless networks using signalprints," in *Proc. of ACM WiSe*, 2006.

[14] R. Gerdes, T. Daniels, M. Mina, and S. Russell, "Device identification via analog signal fingerprinting: A matched filter approach," in *Proc. of Network & Distributed System Security Symp.*, 2006.

[15] D.C.C. Lou, C.Y. Cho, C.P. Tan, and R.S. Lee, "Identifying unique devices through wireless fingerprinting," in *Proc. of ACM WiSec*, 2008.

[16] T. Karagiannis, A. Broido, M. Faloutsos, and K.C. Claffy, "Transport layer identification of P2P traffic," in *Proc. of ACM SIGCOMM conf. on Internet measurement*, 2004.

[17] Q. Sun, D.R. Simon, Y.-M. Wang, W. Russell, V.N. Padmanabhan, and L. Qiu, "Statistical identification of encrypted web browsing traffic," in *Proc. of IEEE Symp. on Security and Privacy*, 2002.

[18] A.W. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *Proc. of ACM SIGMETRICS*, 2005.

[19] B. Padmanabhan and Y. Yang, "Clickprints on the web: Are there signatures in web browsing data?," Downloaded from http://knowledge.wharton.upenn.edu/papers/1323.pdf, 2006.

[20] A. Choppin, "Unsupervised classification of high dimensional data by means of self-organizing neural networks," M.Sc. thesis, Universit catholique de Louvain (Belgium), Computer Science Dept., 1998.