Synthetically scaling a WiFi connection dataset

Jianxin Wei
National University of Singapore
Singapore
jianxinwe@gmail.com

Y.C. Tay
National University of Singapore
Singapore
dcstayyc@nus.edu.sg

ABSTRACT

Collecting real data is really a slow and expensive process. Therefore, synthetic data becomes more and more necessary to make research popular such as on machine learning, speeding up testing of AI algorithms. In this paper, we focus on generating synthetic data to scale real WiFi connection data provided by ALSET Educational Data Lake. Since WiFi connection data contains many personal privacy such as student account and mac address, firstly, we build an anonymization tool for ALSET to globally anonymize the private information. After that, we construct a Decomposition and Reorganization Model to generate preliminary synthetic data and maintain the joint distribution and density distribution of original data. In order to improve the authenticity of synthetic data and its similarity with real data, we then design a Student Tracing model adjusting the relationship among student, session time, and access point location. We evaluate our algorithm performance by comparing the difference between synthetic data and real data. The evaluation results show that our synthetic data maintains both joint and density distributions in high accuracy and well predicts students trajectory.

KEYWORDS

synthetic, anonymization, WiFi connection, trajectory

1 INTRODUCTION

Deep learning machines and artificial intelligence algorithms are expected to solve very difficult problems, and it is a huge data set that drives them. Unfortunately, data is still a issue for machine learning. Nowadays, companies often cannot get enough data to build highly accurate models in a given time frame. They tag the data by hand, which is slow, expensive, and low quality. However, if synthetic data can be used, it can help companies bypass these limits and democratize data [9]. If the company chooses to generate synthetic data on its own, it minimizes the need for third-party data sources and it can fast and cheaply generate lots of perfectly tagged data. Moreover, synthetic data avoid leaking privacy when collecting individual data.

Synthetic data refers to data generated by humans using computers, rather than data measured and collected from real-world environments. A generative model can learn from real data and create data sets that are very similar to real data. This data is anonymous and is created based on user-specified parameters, so it can have as many features as possible in real-world situations. As technology advances, the gap between synthetic data and real data is shrinking.

One way to create synthetic data is using real data. In this paper, we focus on the WiFi connection data provided by ALSET. ALSET Educational Data Lake, an exciting resource for education researchers, securely houses data gathered from across the university, including campus IT systems, student surveys, research study results, and much more [10]. Since WiFi connection data contains many personal privacy, we need to remove private attributes that identify personal information, such as IP address, student number, and mac address, to ensure that the data is anonymous.

Creating high-quality synthetic data is especially challenging when the system is complicated. If the synthetic data is very different from the real data, it will affect the quality of the decisions made based on the data. We list major challenges as follows:

- Lack of common methods: In reality, there are many areas that require synthetic data, such as education, autonomous driving, medical care, etc. Moreover, it involves a wide range of types, including text, images, voice, etc. Raw data from different sources has different content and relationships among its attributes. There is no useful common method that can be applied to all kinds of data. Especially, for WiFi connection data, we need to develop generating algorithms with few reference.
- Lack of evaluation indicators: Similarly, there is also no standard evaluation indicators. What we usually need is the authenticity of synthetic data, i.e., similarity between it and real data. However, it is hard and fuzzy to define such similarity in different areas. In terms of synthetic image, brightness(values of pixels), size of each object and relative location of objects all can be candidate indicators. In addition, the score of each indicator is often relevant to the order of attributes we generated. When our algorithm concentrate more on improving some indicators, the scores of other indicators may decrease. Thus, generating algorithm is very specific and sensitive to evaluation indicators.

Considering the challenges above, it is difficult to design algorithms and evaluation indicators. If we copy some of the raw data to scale up, when it is used to train classifiers, we can neither enhance the robustness of the classifier, nor maintain density distribution since we cannot know copy which part of data. In this paper, we mainly focus on the density distribution of access point, joint distribution among time, bytes, and access point, and the records of students that show their trajectories. We summarize the main contributions of this paper:

- First, we build an anonymization tool with GUI that can anonymize the same attribute in numerous tables or files globally.
- Second, we investigate the relation among 16 attributes in WiFi connection data, including their joint distributions and density distributions, and design an Decomposition and Reorganization Model to generate preliminary synthetic data. With contextual information, we divide the columns of the

original large table into various groups and form small tables. According to the synthetically scaling goal, we generate data in each small table independently to keep its density distribution, and combine them by equijoin under the constraint of joint distributions.

- Third, for each student, we combine his/her connection records as trajectory and transform it into a vector. We construct Student Tracing Model to classify students by their vectors into different moving patterns and predict their trajectories by Markov Method. We apply this model to adjust data and improve its authenticity.
- Finally, we evaluate the performance of our algorithm through comparison with real data. The results show that our synthetic data maintains both joint and density distributions in high accuracy and well predicts students trajectory.

The rest of this paper is organized as follows: In Section 2, we investigate related work on generating different kinds of synthetic data and trajectory prediction methods. The purpose and function of our anonymization tool are presented in Section 3. In Section 4, we illustrate the design details of the Decomposition and Reorganization Model, and demonstrate the theory and application of Student Tracing Model in Section 5. The evaluation results are shown in Section 6. Finally, we conclude the paper in Section 7.

2 RELATED WORK

Synthetic data is artificially produced information, not information generated by actual events. Synthetic data is not limited to text data, but also exists in image, voice, and sensors such as radar and GPS.

In recent years, most of researches focus on generating synthetic image data for computer vision algorithms. Ankush Gupta et al. proposed an engine that overlays synthetic text to existing background images in a natural way, accounting for the local 3D scene geometry [5]. Handa et al. also showed comparable performance on NYUv2 dataset by carefully synthesizing training data with appropriate noise models [6]. In [16], Wong et al. investigated the benefit of augmenting data by data warping and synthetic over-sampling when training a machine learning classifier.

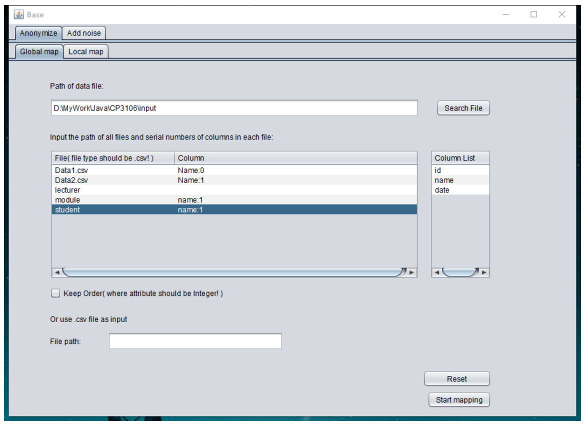
In terms of speech, S. Ronanki et al. presented an approach to duration modelling for statistical parametric speech synthesis [13]. Yuki Saito et al. proposed a method for statistical parametric speech synthesis incorporating generative adversarial networks(GANs) [14]. Winata et al. trained language model with the generated sentences, achieved state-of-the-art performance, and improved end-to-end automatic speech recognition [15].

Unlike speech and image data, the data in the table structure has no ubiquitous data synthesis method due to its numerous complex correlations. Ganapathi et al. presented a data driven approach to generating synthetic data matrices and probabilistic models by retrieving historical network traffic data [3]. In [19], Zhang and Tay proposed ASPECT, a framework for flexible application of tweaking tools to enforce target features in synthetic dataset. To ensure the synthetic data is similar to real data, [4] presented a new framework—Chronos, to support new demands on streaming data benchmarking and [11] presented a system—Synthetic Data Vault (SDV) that built generative models of relational databases. Their experiments showed that in some situation, synthetic data can successfully replace original data for data science.

WiFi positioning has always been a hot research topic. However, in our WiFi connection data, we do not focus on how to position a student but focus on a student's trajectory in one day so that we can generate a series of connection records and accurately simulate such trajectory. In [2], it partitioned individual trajectories obtained through WiFi into periods of stops and moves and opened the way for observing its visited locations or even social behavior. In [18], Markov method was applied in a periodically way to forecast the campus trajectory based on the data of student location and the experiment gave a satisfying prediction result. If we get the geographical information, we can do more accurate prediction. In [7], based on Constant Yaw Rate, Acceleration motion model, and maneuver recognition, it had a high success rate on trajectory prediction.

3 ANONYMIZATION TOOL

3.1 Anonymization



Path of data file: D:\MyWork\Java\CP3106\input

Input the path of all files and serial numbers of columns in each file:

File(file type should be .csv!)	Column
Data1.csv	Name:0
Data2.csv	Name:1
lecturer	
module	name:1
student	name:1

Column List
id
name
date

Or use .csv file as input

File path:

Figure 1: GUI of Anonymization Tool

Since WiFi connection data contains many personal privacy, we need to remove private attributes that identify personal information, such as IP address, student number, and mac address, to ensure that the data is anonymous. Here we provide two ways for ALSET to anonymize data, locally and globally. Global anonymization means that map the same text to the same hash value in all tables and files. It is necessary because local anonymization may come across the situation that one content is hashed to multiple values in different files, which will cause trouble or decrease accuracy when using this data afterwards.

The Graphical User Interface(GUI) of our anonymization tool is shown in Figure 1. There are three steps to use it. First, we input the directory of the folder where contains all files of data. Second, we click on the **Search file** button and it will display all the files' name on the left big table. When we click on each filename, it will display all the columns(data attributes) of that file on the right small table. Third, we can select the columns we want to anonymize by clicking in the right table, or delete them after a wrong click in the left table. In each file, we can select more than one column like

Table 1: Attributes of WiFi connection data

student token	mac token	session start time	session end time
XXXX	ZZZZ	2018-08-12 22:13:59.231	2018-08-13 00:16:14.900
ip address token	**day**	**bytes received**	**bytes sent**
YYYY	2018-08-13	66204	66408
eap type	**ap name**	**ap location**	**ap location mapping**
PEAP	RC2-L21-AP06	RC2 Level 21 Rm. RC2-21-A-02	UTown > UTown-YNC-RC2 > RC2-L21
protocol	**rssi**	**snr**	**ssid**
DOT11AC	-32	32	NUS_STU

session start time and *session end time*. Finally, when completing the configuration, we click **Start mapping** to finish anonymization.

Our tool will also output the mapping relations between values before and after anonymization. In addition, we provide the function of keeping the order of content before and after anonymization. For example, if we want to anonymize some attributes like date, time, or student's score, we must maintain the relative order among values, otherwise and the data will be meaningless after anonymization such as *session start time* will be later than *session end time*.

3.2 Noise fill

Besides anonymization, we also provide the functions of generalization and filling noise to data. Generalization function has one parameter—range size and it converts specific values like student's score into ranges. In terms of filling noise, there is one parameter—percent of noise. According to the specific percent of noise, we randomly select some tuples in original table, change its key value, slightly modify the values of their attributes, and add these new artificial tuples in to raw data.

4 DECOMPOSITION AND REORGANIZATION MODEL

In this section, we first list obvious and absolute constraints in subsection 4.1. In step of Decomposition, we analysis the relationship among attributes, rationally assume their dependencies(joint distributions), and decompose the large table with 16 attributes into multiple small tables. We independently generate small tables and combine them into integral synthetic data in the step of Reorganization.

4.1 Constraints

The 16 attributes of WiFi connection data, including *student token*, *mac token*, *session start time*, *session end time*, *ip address token*, *day*, *bytes received*, *bytes sent*, *eap type*, *ap name*, *ap location*, *ap location mapping*, *protocol*, *rssi*, *snr*, and *ssid*, are shown in Table 1. Before building model and generating synthetic data, we figure out the constraints on content and value in this large table. The main constraints are listed as follows:

- The *session start time* should be before *session end time*.
- One *ip address token* cannot be assigned to different *mac token* at the same time.
- One *student token* or *mac token* mac cannot connect to different *ap name* at the same time.
- The range of *bytes received*, *bytes sent*, *rssi*, and *snr* are constrained. For example, *rssi* varies from -120 to 0.

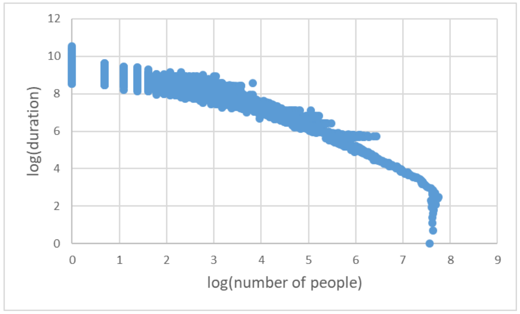


Figure 2: Distribution of connection duration

4.2 Decomposition

For network datasets, people tend to be more concerned with traffic and congestion control in the network. Additionally, after carefully observation and leveraging our context knowledge from real life, most of attributes are more or less associated with the *ap name*—name of access point. Consequently, we start decomposition from *ap name*. We focus on two main joint distributions. One is among *ap name*, *session start time&session end time*, and number of connections, which represents the network congestion. Another is among *ap name*, *session start time&session end time*, and *bytes received&bytes sent*, which represents the network traffic at specific time and each access point.

An important problem to note is maintaining the distribution of connection duration of students. We plot the *log* value of connection duration and number of people in Figure 2, which shows the relation between them is consistent with Power Law—$y = ax^{-k}$. If we only focus on the number of online people and generate *session start time* and *session end time* independently, we will definitely disrupt the duration distribution.

There are many other dependencies that can enlighten us. DHCP controls a range of IP addresses and automatically assign an IP address to each network device. IP addresses on the same LAN

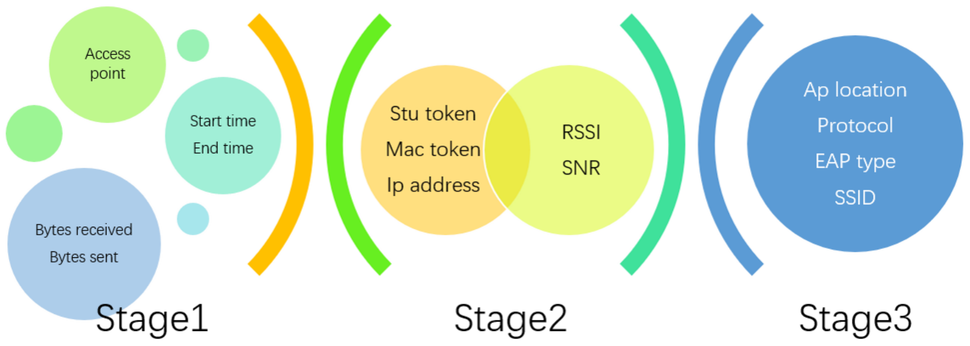


Figure 3: Order of data generation in Reorganization

have the same gateway. Thus, each *ap name* has a fixed range of *ip address token*.

We notice that *student token* and *mac token* are constrained by the *ap name*. Desktop in lab cannot connect to some access point far away from the lab. In addition, students have few probability to use others' personal device and public computers can be used by many students. So, when filling in a synthetic tuple with real *student token* and *mac token* instead of randomly generating some hash value, we must treat *student token* and *mac token* as a whole and only use the combination of *student token*, *mac token*, and *ap name* that appeared in connection records before.

The relationship among *ap name*, *ap location*, and *ap location mapping* is obvious. Each *ap name* has its fixed *ap location* and *ap location mapping*. Using multiple SSIDs allows users to access different networks, each with different policies and functions, increasing the flexibility and efficiency of the network infrastructure. So, there is an one-to-many relationship between *ap name* and *ssid*. Also, each SSID has a specific communication protocol such as DOT11N2_4GNZ and DOT11N5GHZ resulting in the one-to-one correspondence between *ssid* and *protocol*.

In the experiments of M. N. Borenovic[1] and Hüseyin Yiğitler[17], RSSI and SNR are nearly distributed as multivariate normal distribution. We select some access point with more than 50 connection records and observe the fluctuation of their RSSI and SNR. As shown in Figure 4&5, except for some abnormal values, the distribution of both RSSI and SNR looks like Gaussian distribution. Thus, we assume that *rssi* and *snr* of each *ap name* are multivariate normal distribution.

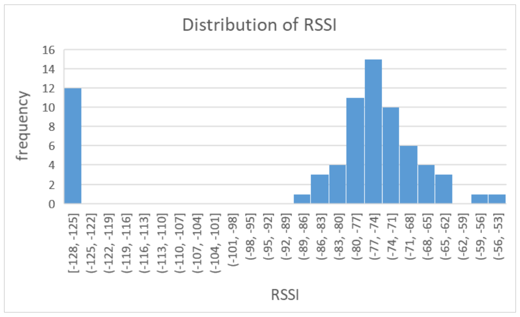


Figure 4: Distribution of connection duration

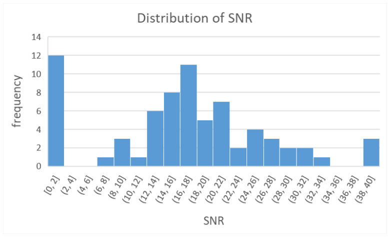


Figure 5: Distribution of connection duration

4.3 Reorganization

We reorganize the 16 attributes following the pipeline in Figure 3. We use a parameter—length of time interval and divide one day into intervals like 00:00-00:20 and 00:20-00:40 when the length is 20 minutes. In each interval i, we count the number of connection records $C_{i,j}$ of every *ap name* P_j and get the number of total connection records C_i in interval i. We suppose the total connection records in one day as C, the size of synthetic data as $\hat{C}$, and the number of connection at interval i and *ap name* P_j in synthetic data as $\hat{C}_{i,j}$. Then $\hat{C}_{i,j}$ can be computed by the following equation:

$$\hat{C}_j = C_j \cdot \frac{\hat{C}}{C}$$

Then we scan the original data. For each *ap name* P_j, we construct a queue Q_j and put the pair of *session start time* and *session end time* in its queue. When assigning *session start time* and *session end time* to $\hat{C}_{i,j}$ records, we use the Algorithm 1. We shuffle the order of Q_j and pop the pairs. If $\hat{C}_{i,j}$ is greater than Q_j, we again shuffle the Q_j and pop the time pairs till the number of pairs we pop is $\hat{C}_{i,j}$.

We do not directly use the same time as original data otherwise the data will be useless. Instead, we add appropriate noise. For each pair of *session start time* and *session end time*, we firstly generalize them into the form of interval and use the start of the interval to represent it, e.g., 09:33-12:54 will be transformed into 09:20-12:40

Algorithm 1 Time allocation

```
1: Shuffle $Q_j$;
2: if $\hat{C}_{i,j} < len(Q_j)$ then
3:     Output the first $\hat{C}_{i,j}$ pairs in $Q_j$;
4: else
5:     $C\_new = \hat{C}_{i,j}$;
6:     while $C\_new > len(Q_j)$ do
7:         Output all pairs in $Q_j$;
8:         $C\_new = C\_new - len(Q_j)$
9:     end while
10: end if
```

when length of interval is 20 minutes. Secondly, we add them with two random numbers ranging from 0 to 1200s.

As we have analyzed in Decomposition, when allocating time to synthetic data, there are two problems to stress. One is the distribution of online students in each interval. Another is maintaining the time duration distribution. Since the above processes are all stochastic, the distribution of online people would be maintaining. Moreover, we only slightly change the *session start time* and *session end time* so the duration distribution would not be affected. Using a queue to store and output the time pairs instead of randomly select one pair to output is to increase the probability of picking a time pair that rarely appeared in original data. Our experiment showed that even if we randomly select numerous times, the pairs rarely appeared before can hardly be selected because the random number generated by computer is not really random.

After assembling *session start time*, *session end time*, and *ap name*, we allocate *student token*, *mac token*, and *ip address token* as shown in Algorithm 2. We extract stu&mac&ip token from real records and store them in a table T_j belonging to P_j and put some fake tokens in T_j for instance that someone wants to research on the change of WiFi connection data when University enrolls much more students. For each synthetic tuple in $\hat{C}_{i,j}$, we assign stu&mac&ip token that not being allocated at that time(from *session start time* to *session end time*) to it.

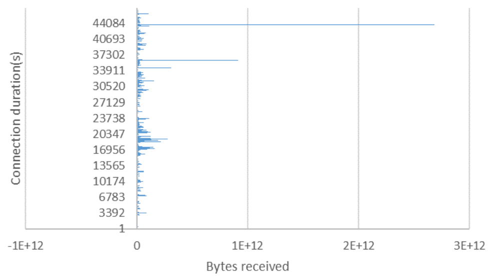


Figure 6: Connection duration with Bytes received

We consider the distributions of *bytes received* and *bytes sent* of each access point and each interval independently according to student habits. One way to generate is multiplying time with average speed(Bytes/s). We assume a person's network traffic is constant over time and add his/her average speed on every interval

Algorithm 2 Stu&Mac&IP token allocation

```
1: while scan each tuple in original data do
2:     $P_j$ : current access point
3:     $T_j$ : token warehouse of $P_j$;
4:     Store the pair of stu&mac token in $T_j$;
5:     Store ip address token in $T_j$;
6: end while
7: Put fake stu&mac&ip token in $T_j$ if need;
8: for $\hat{C}_{i,j} \in \hat{C}$ do
9:     for each tuple $\in \hat{C}_{i,j}$ do
10:        while true do
11:            Find one pair of stu&mac token in $T_j$;
12:            Check whether the pair being allocated at that time;
13:        end while
14:        while true do
15:            Find one ip address token in $T_j$;
16:            Check whether the ip being allocated at that time;
17:        end while
18:        if stu&mac or ip are all occupied then
19:            Generate a fake one;
20:        end if
21:    end for
22: end for
```

in his/her connection duration; e.g., if a connection lasted from 18:55-19:10, it contributes to the intervals 18:40-19:00 and 19:00-19:20. In each interval, we compute the average speed per capita and assume that everyone connecting at this interval will have such speed. However, in our experiment, this bytes assignment method lacks of data variance. In real data, it seems that values of *bytes received* and *bytes sent* vary dramatically as shown in Figure 6 and have no obvious correlation with the length of connection time. There should be a implicit distribution in the speeds of students in each interval. To simplify, we assume it as a uniform distribution ranging from minimum to maximum speed in records, as shown in Algorithm 3.

For each access point, we learn its multivariate normal distributions of RSSI and SNR from historical data. When scaling dataset, we generate two groups of numbers conforming to these two distributions as *rssi* and *snr*. Additionally, we add some abnormal values as they were in real data. We attach less importance on *protocol* and *ssid* and just keep the proportion of their frequency in original data. As *eap type*, *ap location*, and *ap location mapping* are bound to *ap name*, we obtain all the components of WiFi connection data now and can generate an integral synthetic dataset.

5 STUDENT TRACING MODEL

In Section 4, our method largely keep the joint distributions and density distributions among attributes without violating constraints by centering on *ap name* and splicing data. But the trajectory of each student may be fragmented since we only guarantee the *student token* and *mac token* will not conflict and allocate randomly. In this section, we propose Student Tracing Model to improve the algorithm of allocating *student token* and *mac token*. We represent the student trajectory by a vector and cluster them into different

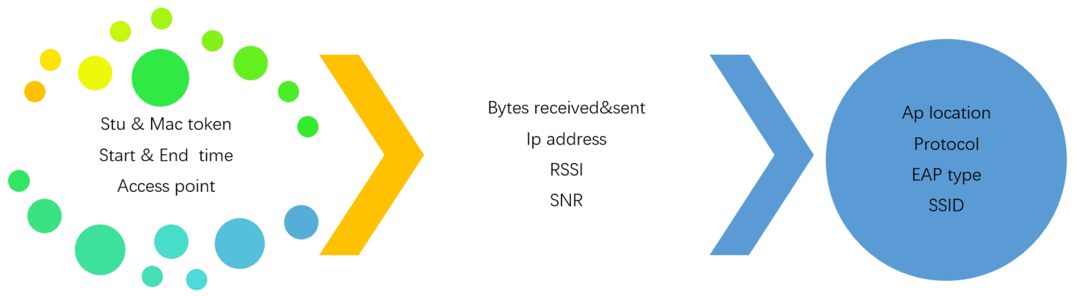


Figure 7: Order of data generation in Student Tracing Model

Algorithm 3 Bytes computation

```
1:  while scan each tuple in original data do
2:      $P_j$ : Current access point;
3:      $S_b r$=bytes received / duration;
4:      $S_b s$=bytes sent / duration;
5:      for all intervals in duration do
6:          Update $U_{received}[minSpeed, maxSpeed]$;
7:          Update $U_{sent}[minSpeed, maxSpeed]$;
8:      end for
9:  end while
10: for $\hat{C}_{i,j} \in \hat{C}$ do
11:     for each tuple $\in \hat{C}_{i,j}$ do
12:         for each interval in duration do
13:             Randomly generate two speeds s. t. $U_{received}\&U_{sent}$;
14:         end for
15:         compute bytes received and bytes sent;
16:     end for
17: end for
```

moving patterns. For each pattern, we construct their transition probability matrix. When generating synthetic data, we start from *student token* and concatenate other attributes as shown in Figure 7.

5.1 Pattern Clustering

As shown in Table 1, *ap location mapping* has 3 labels, *UTown, UTown-YNC-RC2, RC2*, identifying location from large area to precise area. We encode labels of access point P_j with three-dimension vectors, $a_{j,1}$, $a_{j,2}$, and $a_{j,3}$. We assign different weights, w_1, w_2, and w_3, to three vectors, where $w_1 \gg w_2 \gg w_3$. We use the equation 1 to compute difference $d(j_1, j_2)$ between two access point j_1 and j_2, where $I(a_{j1,n} = a_{j2,n})$ is explained in equation 2.

$$d(j_1, j_2) = \sum_{n=1}^{3} w_n \cdot I(a_{j1,n} = a_{j2,n}) \tag{1}$$

$$I(a_{j1,n} = a_{j2,n}) = \begin{cases} 1, a_{j1,n} = a_{j2,n} \\ 0, a_{j1,n} \neq a_{j2,n} \end{cases} \tag{2}$$

We observe that most of students keep connecting to one access point every one or two hours since their lives are regular. They go to auditorium in the morning, go to canteen at noon, have another class in the afternoon, and sleep at dormitory during whole night. Consequently, we divide one day into N(N=9 when interval is 2 hours) intervals like 07:00-09:00, 09:00-11:00, …, 21:00-23:00, and 23:00-07:00. To represent each student by a vector, we firstly get everyone's trajectory from original connection data. Second, we use vote method to find his/her a main access point with most connection records in each interval. Then we obtain 9 *ap name* with 27 (9*3) labels. Third, we combine the labels' vectors to a 27-dimension vector V_s—trajectory of student s in one day. Now we can cluster students by distance between their trajectory vectors computed by equation 3.

$$d(V_{s1}, V_{s2}) = \sum_{j=1}^{N} \sum_{n=1}^{3} w_n \cdot I(a_{j1,n} = a_{j2,n}) \tag{3}$$

5.2 Markov Chain

A Markov chain is a discrete-time, finite state stochastic process. For a sequence of random variables $X_1, X_2, \ldots$ in a Markov chain, the probability of moving from X_n to the next state X_{n+1} depends only on the present state and not on the previous states as shown in equation 4.

$$\begin{aligned} &\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) \\ &= \Pr(X_{n+1} = x | X_n = x_n) \end{aligned} \tag{4}$$

The Markov chain can be interpreted as a state machine with transition probability hopping from state x_k to an adjacent state x_l. We define this transition probability as $P_{kl} = P(X_{n+1} = x_l | X_n = x_k)$. We define state space as $\mathbf{T}$, which contains the integral address of each access point, e.g. *UTown->UTown-YNC-RC2->RC2*. Transition probability matrix P is composed by all possible $P_{kl}, \forall x_k, x_l \in \mathbf{T}$.

5.3 Student Transition Matrix

We build our Student Transition Matrix based on historical data. We keep the division of one day, and for each moving pattern r, we construct one transition matrix $P_{r,n}$ in each interval n. In our experiment, if we build only one transition matrix for all patterns in each interval, it will occupy lots of memory space, slow down the running speed of program, and will not increase any prediction accuracy. We define the number of connections to *ap name* P_l ($P_l \in \mathbf{P}$—set of access points) in interval n as $c_{n,kl}$, when the previous *ap name* connected by student in each tuple is P_k. Then the transition

Table 2: Feature of WiFi connection data

session start time	end time	ap name
23:33:09	23:33:13	RC2-L21-AP07
23:33:28	23:34:44	RC2-L21-AP06
23:34:59	23:39:20	RC2-L21-AP06
23:37:52	23:37:58	RC2-L21-AP06
23:39:35	23:45:19	RC2-L21-AP06
23:45:34	23:53:18	RC2-L21-AP06
23:53:33	23:53:49	RC2-L21-AP06

probability can be computed as equation 5.

$$P_{kl,n} = \frac{c_{n,kl}}{\sum_{l}^{P_l \in \mathbf{P}} c_{n,kl}} \tag{5}$$

Since we have clustered students before so the students in the same pattern will have similar trajectory, i.e., similar current state P_k and similar next state P_l. Therefore, every $P_{r,n}$ has a not too large size.

In historical data, we classify the students into appropriate clusters and count the number of students in each cluster. When creating synthetic data, we maintain the proportion of people in different clusters and computer the number of tuples we need by the same method used to compute *ap name* frequency in Section 4. If the number of tuples in cluster U is m_u, we generate m_u series of random number as probabilities and apply the transition matrix to each series so that we can get a series of *ap name*(the number of *ap name* is equal to the number of intervals) and a time range.

In above process, when we create one trajectory, each transition matrix will be used only once in each interval. So, there are definitely not enough connection records because most of students have more than one connection during two hours in real data. A useful observation of some students shown in Table 2 is that people may have frequent connections even though during a very short period of time. Because the network may be not so stable and it will disconnect their sessions for a while. The instability should be relevant to access point. Therefore, for each *ap name*, we extract its instability represented by reconnection times during a short period from historical data and assume the reconnection times as a uniform distribution during a interval. Then in each interval, to make the synthetic data more similar to the real data, we not only use transition probability more than once to get more *ap name* but also create some connection and reconnection records during a short period for each *ap name*.

5.4 Smoothing Matrix

Previously, we construct only one transition matrix in each interval and pattern. Inspired by [18], in each interval, we build more transition matrices and smooth them by assigning different weights like the concept of the Exponential Smoothing. We subdivide the 2-hour-interval into about 6 parts and build 6 transition matrices by previous method. For each part t, we smooth the later 5 matrices by equation 6, where $\sum_{i=0}^{m} a_i = 1 \& a_i > a_j$, if $i < j$. Although we increase the number of matrices, the size of each matrix decrease so that it will not put much pressure on time and space.

$$P^{\mathrm{t}} = a_0 P^{\mathrm{t}} + a_1 P^{\mathrm{t}-1} + \cdots + a_m P^{\mathrm{t}-\mathrm{m}}, \text{where } t \in 2, \ldots, 6, m = t-1 \tag{6}$$

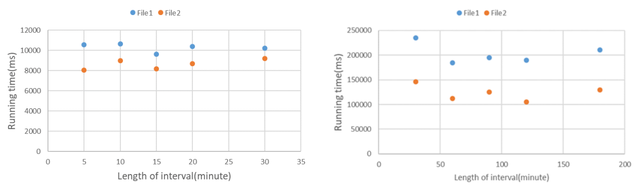
We obtain *stu token*, *mac token*, *ap name*, *session start time*, *session end time* by applying Student Tracing Model. After that, we generate other attributes by using the same algorithm in Section 4.

6 EVALUATION

In this section, we report the evaluation results on our two models.

6.1 Data Analysis

We base on real WiFi connection data provided by ALSET Education Data Lake which contains 13 days of students' connections. There are 16 columns and 200,000 500,000 rows in data file of each day. In our first experiment we test the running time of generating 500,000 synthetic tuples by our Decomposition and Reorganization Model(defined as Model1) and Student Tracing Model(defined as Model2) as shown in Figure 8. File1 contains 509226 rows and File2 contains 210925 rows. The running time of Model1 in File1 ranges from 8000ms to 9000ms and in File2 from 9000ms to 10000ms, while the Model2's ranges from 184,000ms to 235,000ms in File1 and from 105,000ms to 145,000ms in File2. Model2 costs much more time than Model1 because we spend more time on constructing transition matrix and prediction than just shuffling queue in Model1.



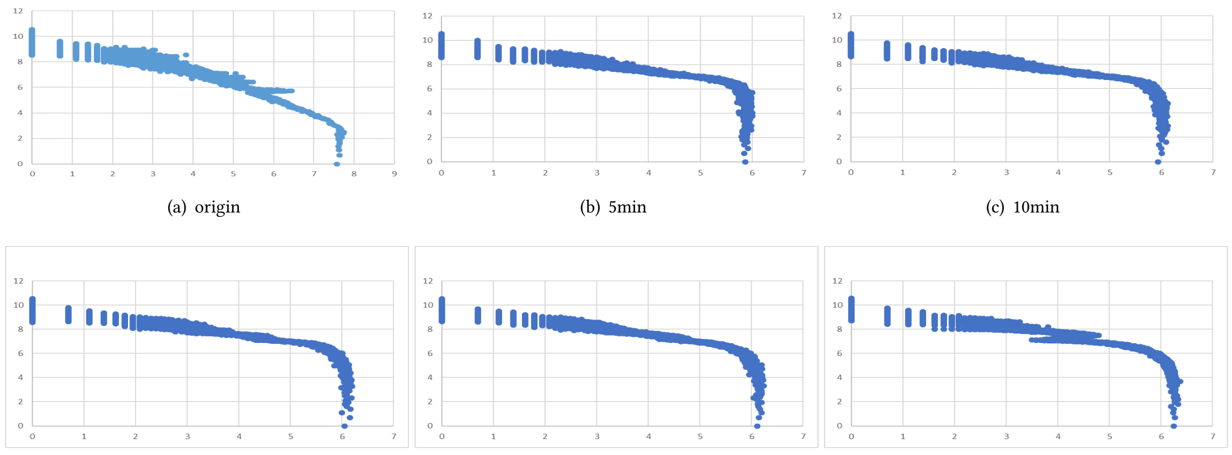
(a) Model1 (b) Model2

Figure 8: Time of generating 500,000 tuples

In our second experiment, we plot the duration distribution in our synthetic data with different lengths of interval as shown in Figure 9. The x-axis is the log value of number of people and the y-axis is the log value of connection duration. When the length of interval is short, we can maintain the distribution well. However, when it comes to 30min, the duration distribution becomes chaotic because we add too much noise in *session start time* and *session end time*.

In our third experiment, we test the prediction error of access point $P_j \in \mathbf{P}$ in synthetic data generating by Model2 and Model1. We define the error ϵ by equation 7, where v_j is the proportion of P_j in real data and $\hat{v}_j$ in synthetic data. As shown in Figure 10, where x axis in length of interval and y is the error rate, Model2 has more error in generating *ap name* than Model1. Because we decrease the frequency of *ap name* that seldom appear in original data and increase the frequency of original frequent one when using transition matrices to prediction. Additionally, when the length of interval grows down, the transition predicted by matrices is more accurate since it is fine-grained to construct matrix and predict in such short period.

$$\epsilon = \sum_{j}^{P_j \in \mathbf{P}} \left| v_j - \hat{v}_j \right| \tag{7}$$


(a) origin (b) 5min (c) 10min

(d) 15min (e) 20min (f) 30min

Figure 9: Duration distribution

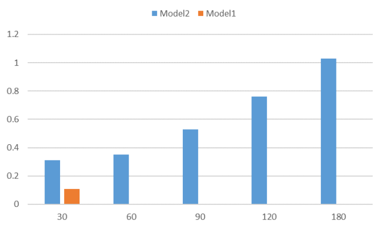


Figure 10: Prediction error of *ap name*

6.2 Improvement

The performance of synthetic data generated by Student Tracing Model is mainly limited by our pattern clustering algorithm. The way we encoding location and computing distance is not so accurate because we do not have the latitude and longitude data of access point and cannot take its actual location into account. A further step to do is using Google Map to collect the geographical information of *ap name* and compute the real world distance. There are many excellent algorithm on trajectory prediction based on real world location data. In [12], S Qiao et al. proposed a three-in-one TP model in road-constrained transportation networks and achieved more than 80% prediction accuracy. In [8], Tong Liu et al. developed a hierarchical user mobility model based on wireless ATM networks that can closely represent the movement behavior of users.

7 CONCLUSION

In this paper, we have applied various synthetic data generation methods in WiFi connection data. We initially built an anonymization tool to globally anonymize the private information. We have exploited the relations among 16 attributes, and constructed Decomposition and Reorganization Model to generate preliminary synthetic data and maintain the joint distribution and density distribution of original data. We have designed a Student Tracing model adjusting the relation among student, session time, and access point location improved the authenticity of synthetic data and its similarity with real data. Through simulation on real data, we have shown the excellent performance of our algorithms on maintaining the distributions among attributes and trajectories of students.

REFERENCES

[1] M. N. Borenovic and A. M. Neskovic. 2009. Comparative analysis of RSSI, SNR and Noise level parameters applicability for WLAN positioning purposes. In *IEEE EUROCON 2009*. 1895–1900. https://doi.org/10.1109/EURCON.2009.5167905
[2] Cristian Chilipirea, Mitra Baratchi, Ciprian Dobre, and Maarten van Steen. 2018. Identifying Stops and Moves in WiFi Tracking Data. *Sensors* 18, 11 (2018), 4039.
[3] Tejaswini Ganapathi, Satish Raghunath, Xu Che, Shauli Gal, and Andrey Karapetov. 2019. ON DEMAND SYNTHETIC DATA MATRIX GENERATION. US Patent App. 15/803,501.
[4] Ling Gu, Minqi Zhou, Zhenjie Zhang, Ming-Chien Shan, Aoying Zhou, and Marianne Winslett. 2015. Chronos: An elastic parallel framework for stream benchmark generation and simulation. In *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 101–112.
[5] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. 2016. Synthetic Data for Text Localisation in Natural Images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
[6] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. 2016. Understanding real world indoor scenes with synthetic data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4077–4085.
[7] Adam Houenou, Philippe Bonnifait, Véronique Cherfaoui, and Wen Yao. 2013. Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 4363–4369.
[8] Tong Liu, Paramvir Bahl, and Imrich Chlamtac. 1998. Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks. *IEEE Journal on selected areas in communications* 16, 6 (1998), 922–936.
[9] Astasia Myers. 2019. Deepfakes: What's real with synthetic data? https://medium.com/memory-leak/deepfakes-whats-real-with-synthetic-data-5c8348b041d2
[10] NUS. 2019. The ALSET Educational Data Lake. http://nus.edu.sg/alset/data-lake/
[11] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. 2016. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 399–410.
[12] Shaojie Qiao, Nan Han, William Zhu, and Louis Alberto Gutierrez. 2014. TraPlan: an effective three-in-one trajectory-prediction model in transportation networks. *IEEE Transactions on Intelligent Transportation Systems* 16, 3 (2014), 1188–1198.
[13] S. Ronanki, O. Watts, S. King, and G. E. Henter. 2016. Median-based generation of synthetic speech durations using a non-parametric approach. In *2016 IEEE Spoken Language Technology Workshop (SLT)*. 686–692. https://doi.org/10.1109/SLT.2016.7846337

[14] Y. Saito, S. Takamichi, and H. Saruwatari. 2018. Statistical Parametric Speech Synthesis Incorporating Generative Adversarial Networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26, 1 (Jan 2018), 84–96. https://doi.org/10.1109/TASLP.2017.2761547

[15] Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2019. Code-switched language models using neural based synthetic data from parallel sentences. *arXiv preprint arXiv:1909.08582* (2019).

[16] Sebastien C Wong, Adam Gatt, Victor Stamatescu, and Mark D McDonnell. 2016. Understanding data augmentation for classification: when to warp?. In *2016 international conference on digital image computing: techniques and applications (DICTA)*. IEEE, 1–6.

[17] H. Yiğitler, R. Jäntti, and N. Patwari. 2017. On Log-Normality of RSSI in Narrowband Receivers Under Static Conditions. *IEEE Signal Processing Letters* 24, 4 (April 2017), 367–371. https://doi.org/10.1109/LSP.2017.2657332

[18] C. Yuan, D. Li, and Y. Xi. 2015. Campus trajectory forecast based on human activity cycle and Markov method. In *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. 941–946. https://doi.org/10.1109/CYBER.2015.7288071

[19] JW Zhang and YC Tay. 2018. A tool framework for tweaking features in synthetic datasets. *arXiv preprint arXiv:1801.03645* (2018).