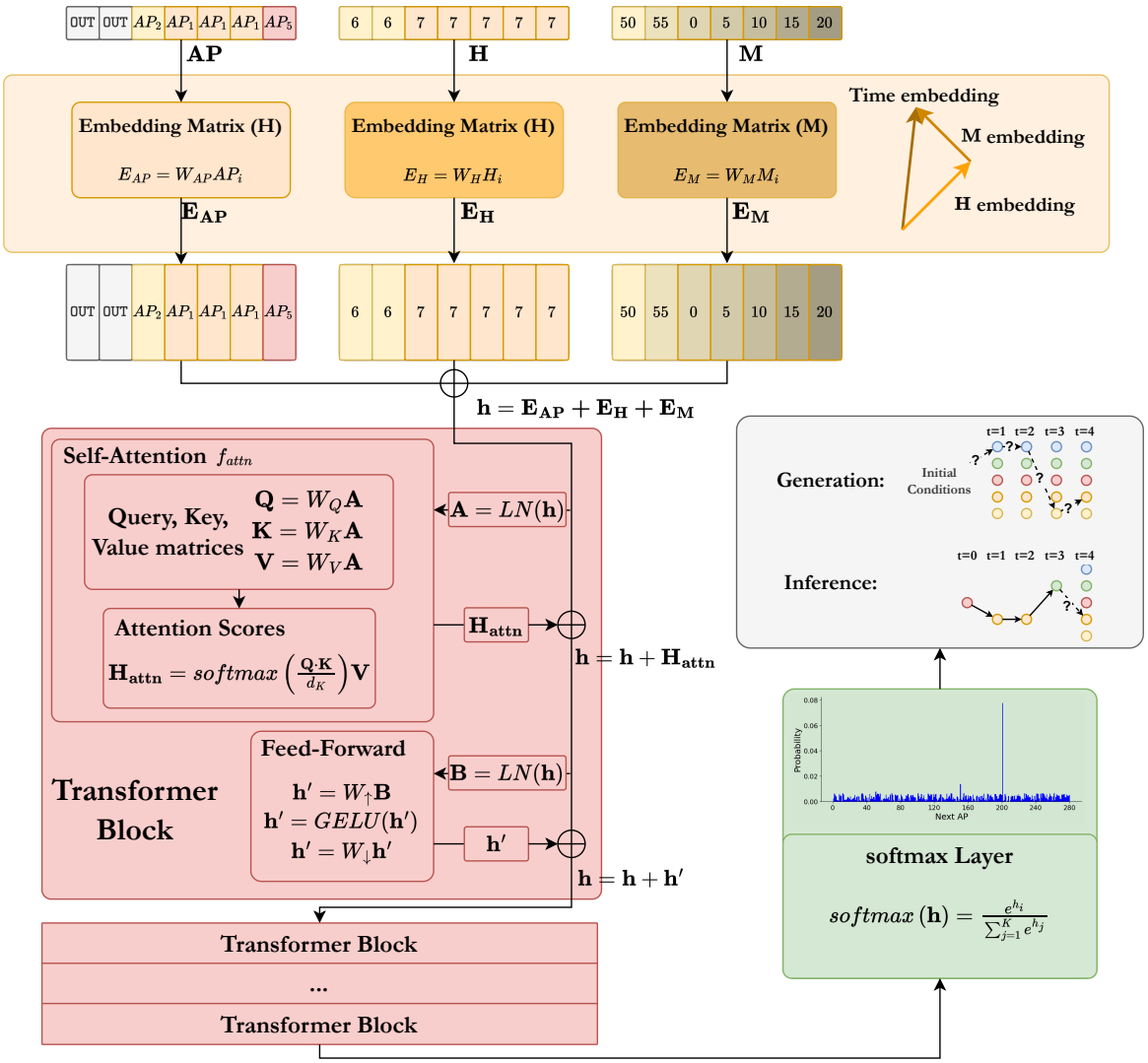
Graphical Abstract

DiWi: A Transformer-Based Digital Twin for Wireless Mobility

Juan Manuel Montes-Lopez, Pablo Serrano, Marco Gramaglia, Albert Banchs



Highlights

DiWi: A Transformer-Based Digital Twin for Wireless Mobility

Juan Manuel Montes-Lopez, Pablo Serrano, Marco Gramaglia, Albert Banchs

- Research highlight 1

- Research highlight 2

DiWi: A Transformer-Based Digital Twin for Wireless Mobility

Juan Manuel Montes-Lopez[a], Pablo Serrano[a], Marco Gramaglia[a], Albert Banchs[a,b]

[a] *University Carlos III of Madrid, Avenida Universidad 30, Leganés, 28911, Madrid, Spain*
[b] *IMDEA Networks, Avenida Mar Mediterráneo 22, Leganés, 28911, Madrid, Spain*

Abstract

Understanding device mobility in wireless networks is essential for multiple purposes: optimizing space usage, managing intelligent buildings, or improving network efficiency. However, the use of real mobility data raises significant privacy concerns. In this work, we propose DiWi (a Digital twin for Wireless mobility), a Transformer-based model to generate spatiotemporal mobility traces that mimic real-life behavior while preserving user privacy. We validate the utility of DiWi by comparing real and synthetic datasets, and by demonstrating its usefulness in a series of use cases related to mobility management, resource consumption, and privacy enhancements. We also confirm that DiWi is secure by evaluating empirical privacy metrics, such as, direct leakage, similarity searches, or membership inference. Our results illustrate that DiWi serves to generate realistic and useful mobility patterns without exposing identifiable user traces, making it a valuable tool for privacy-preserving mobility analysis. Furthermore, we investigate how enforcing differential privacy affects the generative performance of the model.

Keywords: Generative AI, Mobility Network Data, Private Data Publishing
PACS: 0000, 1111
2000 MSC: 0000, 1111

1. Introduction

Wireless networks provide seamless connectivity by eliminating the need for physical connections, enabling broad access to digital services. As devices move through the network, they generate connectivity patterns that represent traces of their activity [1]. These traces provide insights into aspects such as high-traffic areas, peak activity times, and typical usage patterns [2]. These insights into device behavior help operators allocate resources more efficiently, reduce energy use with adaptive management strategies [3], and improve systems like heating optimization [4].

However, the pervasive use of wireless networks comes with significant privacy concerns. Effective network management, optimization, and security often require extensive monitoring, which involves collecting sensitive data about users' mobility, device activity, and personal habits. If unauthorized entities access this data, it could be exploited in a privacy-invasive manner, highlighting the need for strong privacy-preserving techniques in data analysis and synthetic trace generation [5].

To address these concerns, it is essential to implement robust techniques that secure and protect sensitive mobility data even when encryption is breached or data is leaked. Methods such as pseudonymization are known to only mildly solve this issue as demonstrated in [6], where 93 % of the users were re-identified even when this technique is applied.

In contrast to pseudoanonymization, synthetic data offers a practical way to protect user privacy while preserving the statistical properties needed for analysis [7]. By generating datasets that mimic real data without directly linking to individual users, the risk of re-identification is reduced. This allows researchers and organizations to study patterns, train machine learning models, and develop solutions without exposing sensitive information. One potential clear advantage of synthetic data is that it does not have a one-to-one correspondence with real devices (unlike some pseudinimization techniques), but its utility is at question since it may not capture real-life dynamics.

In this work, we propose DiWi, a transformer-based Digital twin for Wireless mobility. DiWi generates synthetic data that captures the features of real-life device mobility in wireless networks, supporting the development of relevant use cases without leaking personal data. More specifically, the key contributions of this paper are:

- **Transformer-based mobility modeling**: We design and implement a Transformer-based model to predict device connectivity sequences in a campus WLAN. By decomposing temporal information during encoding, the model reduces the number of parameters while improving generalization, enabling it to generate realistic spatiotemporal traces. The model is trained on real traces collected from a campus environment with thousands of devices, users, and access points, ensuring accurate and meaningful results.
- **Applications in network and space management**: We explore the model's applicability in key use cases such as optimizing space usage, managing intelligent buildings, predicting user occupancy for energy-efficient Heating, Ventilation, and Air Conditioning (HVAC) systems, and forecasting mobility for network management. Additionally, we demonstrate its utility in developing and evaluating privacy metrics for wireless networks.

- **Privacy-preserving data generation**: To ensure the generated data does not compromise user privacy, we assess risks such as direct leakage, similarity searches, and membership inference attacks [5]. Our analysis confirms that the model effectively captures mobility patterns while preventing the exposure of identifiable user traces. Building on this, we further explore the trade-off between utility and privacy by integrating a Differentially Private mechanism into the model training process.

The rest of the paper is organized as follows: In Section 2, we describe DiWi, providing an overview of the system, the dataset used, the sequence encoding, and the modeling and training stages. We evaluate DiWi in Section 3, both by analyzing device and Access Points (AP) statistics, and by exploring its applicability in key use cases related to sustainability, mobility management, and understanding anonymity schemes. We assess DiWi's privacy features in Section 4 by examining exact matches, dataset closeness, and membership inference risks. Finally, we review the related work in Section 5 and conclude the paper in Section 6.

2. DiWi: a Digital Twin for Wireless Mobility

Here we describe DiWi, a Transformer-based model designed for predicting the next location of a user and generating synthetic mobility traces. We detail its key components and their roles in processing device connectivity sequences. Specifically, we explain how the model encodes spatial and temporal information, integrates these representations, and leverages a Transformer framework to predict the next connectivity state.

2.1. Dataset

Unless otherwise noted, we rely on a dataset of users connected to the WiFi network at in one Campus of Universidad Carlos III de Madrid (UC3M)[1]. The network is provided by 279 APs distributed across seven buildings. These buildings, most with at least three floors, serve various purposes such as classrooms, cafeterias, and study areas. Each AP provides coverage for approximately 100 m^2, while the entire campus spans 1.6 million m^2 and supports a community of around 10,000 individuals. The dataset spans four weeks of connectivity logs, with no holidays or special events in between.

We are provided the data by system administrators, which rely on a system that logs device connections and discretizes them every 5-minute intervals[2], according to the following rules:

- If a device remained connected to a single AP during the interval, that AP was recorded.
- If the device connected to multiple APs, the AP where it spent the majority of time was assigned.
- If no connection was detected, the device was considered disconnected, and we introduce an artificial `OUT` status to represent periods without connectivity.

Before processing the data, we consulted the Data Protection Officer (DPO) at UC3M to ensure full compliance with ethical and legal requirements. The dataset was pseudoanonymized using MD5 hashing of user identifiers and MAC addresses. Access to the data was restricted to secure servers with user-based authentication and activity logs. In addition, the university provided public notice of data collection practices, including the right to access, modify, or delete personal data. Our team also formally committed not to attempt any form of re-identification.

To ensure that each device's behavior is accurately captured in the dataset, we examined the potential impact of MAC address randomization on data quality. In our WLAN deployment, all access points broadcast the same SSID, which limits the effect of MAC randomization, as most modern devices use SSID-specific randomization, meaning they keep the same address when connecting to the same SSID [8]. We specifically searched for more aggressive forms of randomization that would produce different identifiers each day, did not observe them: the number of addresses per user remained stable during the whole period, which is consistent with the lack of aggressive randomization.

2.2. Model overview

The architecture of DiWi is illustrated in Figure 1, which summarizes the entire workflow (represented from top to bottom). Starting from the dataset, the system sequentially encodes spatial and temporal components of device connectivity traces (top part of the figure). These components are then merged into a unified spatiotemporal representation, capturing the full behavior of each device. Finally, the model predicts the next connectivity state: whether the device will remain connected to the same AP, transition to a different AP, or disconnect entirely (`OUT`). The training process is designed to optimize these predictions, ensuring accurate modeling of mobility behavior across the network.

2.3. Sequence Encoder

We restrict ourselves to the interval between 6:00 AM and 10:00 PM. Given the 5-minute discretization, a trace consists of 192 tokens (i.e., 12 per hour). The model's vocabulary ($\mathcal{V}$) includes the 279 APs plus the `OUT` token, and therefore it captures all possible device locations, including disconnections.

To model device connectivity, we encode both spatial and temporal information into a unified vector representation. This combined encoding allows the model to capture the full context of a device's mobility, i.e., its location within the network and the timestamp.

For **spatial encoding**, we use an AP embedding matrix W_{AP}, which maps one-hot vectors representing possible connectivity states—either a specific AP or the `OUT` state for disconnections—into high-dimensional vectors of size d. This process

[1] `https://www.uc3m.es/life-on-campus/campuses-plans/leganes`

[2] While this temporal resolution could be a limitation in high-mobility scenarios, our results show that it is sufficient to accurately capture typical trends in a campus WLAN, as demonstrated later.

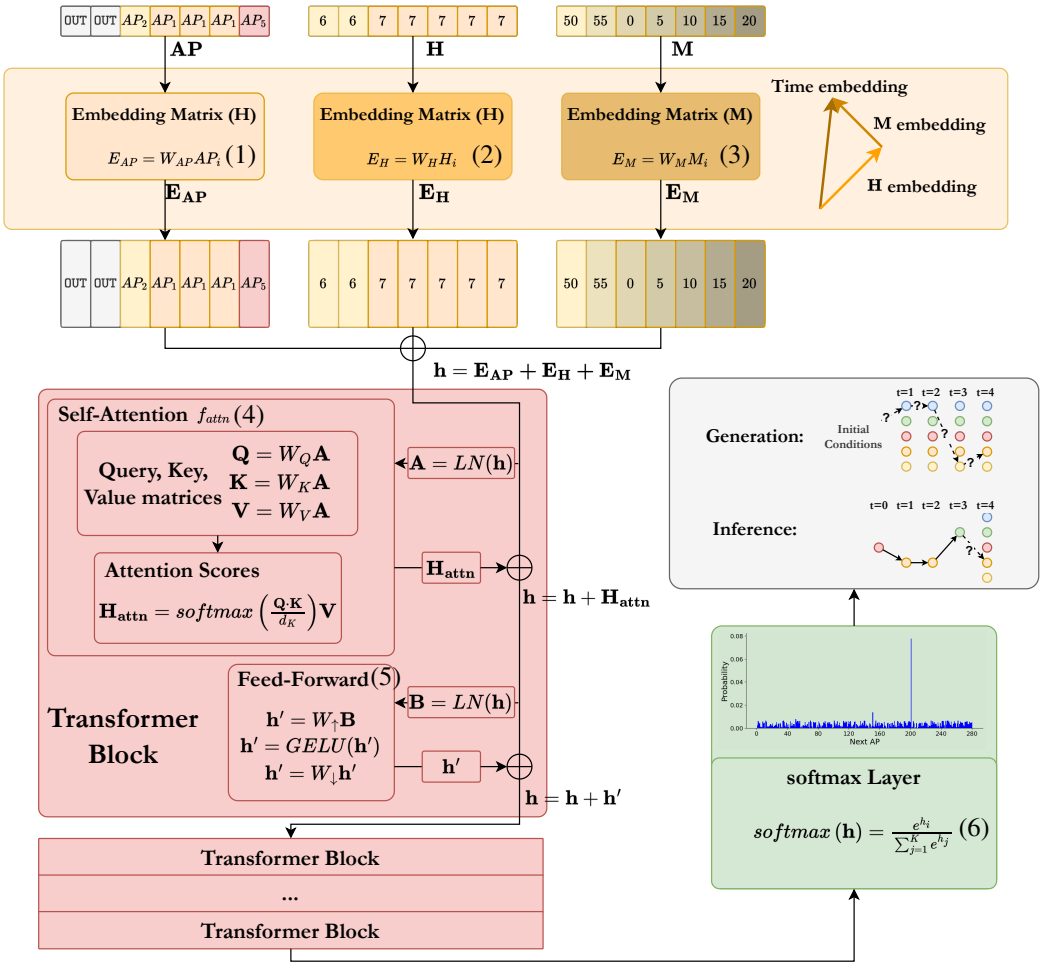


Figure 1: Architecture of DiWi. The orange box shows the decomposed encoding and creation of the joint spatiotemporal representation, the red box represents the transformer blocks and their computations, and the green box indicates the prediction of the next AP and the inference process.

generates spatial embeddings $\mathbf{E_{AP}}$, as shown in Eq. (1), which captures the spatial properties of each connection:

$$\mathbf{E_{AP}} = W_{AP} \cdot \mathbf{AP}_i \quad \text{(Embedding for input AP)} \tag{1}$$

For the **temporal encoding**, we use absolute positional embeddings to represent the exact timestamp of each connection. Unlike relative positional embeddings commonly used in natural language processing (e.g., to preserve word order in text), absolute embeddings are better suited for mobility data, since the absolute position of a token within the timeline provides critical contextual information. For example, being disconnected at 7:00 AM (typically before a user arrives) conveys different behavior than being disconnected at 2:00 PM (a common lunch hour in Spain). To encode absolute time, we decompose it into two components: hours (H) and minutes (M). Each component is mapped into high-dimensional space using its own embedding matrix as follows:

- The hour embedding $\mathbf{E_H}$ is computed using W_H, an embedding matrix of shape $(N_H \times d)$, where N_H represents the number of possible hour values:

$$\mathbf{E_H} = W_H \cdot \mathbf{H}_i \quad \text{(Embedding for input H)} \tag{2}$$

- The minute embedding $\mathbf{E_M}$ is computed using W_M, an embedding matrix of shape $(N_M \times d)$, where N_M represents the number of possible minute values:

$$\mathbf{E_M} = W_M \cdot \mathbf{M}_i \quad \text{(Embedding for input M)} \tag{3}$$

The final temporal embedding at each time step is obtained by adding both embeddings ($\mathbf{E_H} + \mathbf{E_M}$). This segmented approach to temporal encoding is more efficient than using a single absolute positional encoding matrix of size $L \times d$, where $L = 192$ represents the total number of time steps in a trace. As we will further analyze in Section 3, by breaking time into smaller components (hours and minutes), we significantly reduce the number of parameters required, which improves memory efficiency while preserving the semantic meaning of absolute time, making the model both scalable and practical for processing large mobility traces.

Together, the spatial embedding $\mathbf{E_{AP}}$ and temporal embeddings ($\mathbf{E_H}$ + $\mathbf{E_M}$) form the joint spatiotemporal representation of device traces $\mathbf{h^{(l)}}$, where the super index represents the layer l of the model.

2.4. Sequence Modeling

After the joint spatiotemporal vector $\mathbf{h}^{(l-1)}$ is composed, it is processed through a series of Transformer blocks designed to capture dependencies and relationships between connectivity states. Each block includes two residual connections: one around the self-attention sub-layer and another around the position-wise feed-forward sub-layer, which together facilitate gradient flow through deep networks.

The Transformer block begins by normalizing the embeddings from the previous layer, denoted by $\mathbf{h}^{(l-1)}$ to improve training stability and gradient flow [9]. The normalized input is then used to compute self-attention by projecting the normalized input embeddings $\mathrm{LN}(\mathbf{h}^{(l-1)})$ into query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$ vectors. A scaled dot-product operation between $\mathbf{Q}$ and $\mathbf{K}$ captures relationships between elements in the sequence. These scores are then normalized using a *softmax* function and applied to $\mathbf{V}$, producing the attention output $\mathbf{H}_{\text{attn}}$. To retain the original contextual information, the attention output is added back to the input $\mathbf{h}^{(l-1)}$, and the result is passed through a normalization layer, forming the final output of the attention sub-layer. This process is summarized by Eq. (4)

$$\mathbf{H}_{\text{attn}} = f_{\text{attn}}(\mathrm{LN}(h^{(l)})), \quad \mathbf{h}^{(l)} = \mathbf{h}^{(l-1)} + \mathbf{H}_{\text{attn}} \tag{4}$$

where f_{attn} represents the self-attention mechanism applied to the combined embeddings of access points, hours, and minutes, and LN denotes the Layer Normalization operation applied after each sub-layer to stabilize training and improve convergence [10].

The output from the attention mechanism, $\mathbf{h^{(l)}}$, is normalized and passed through a two-layer feed-forward network with a GELU non-linearity. This network consists of two linear transformations: the first, parameterized by $\mathbf{W}_{\uparrow}$, expands the dimensionality of the representation, while the second, $\mathbf{W}_{\downarrow}$, reduces it back to its original size. The resulting output is added back to the input of the sub-layer and re-normalized, completing the Transformer block.

$$\mathbf{h}' = \mathbf{W}_{\downarrow}\,\mathrm{GELU}(\mathbf{W}_{\uparrow}\,\mathrm{LN}(\mathbf{h}^{(l)})), \quad \mathbf{h}^{(l)} = \mathbf{h}^{(l)} + \mathbf{h}' \tag{5}$$

Finally, the output of the last Transformer block, $\mathbf{h}^{(L)}$, is projected to token logits and normalized using a softmax function to produce a probability distribution over the model's vocabulary $\mathcal{V}$. This distribution represents the likelihood of each possible next connection state (e.g., an access point or disconnection), and the prediction process is formalized in Equation (6):

$$\mathbf{P}_{\text{next}} = \mathrm{softmax}(\mathbf{h}^{(N_{layers})}) \tag{6}$$

Using the probability distribution defined in Equation (6), we generate synthetic datasets sequentially. At each step, the model predicts the next AP by sampling from the probabilities calculated based on the sequence of tokens observed so far. This approach assumes conditional independence, meaning that each prediction depends solely on the context provided by the preceding sequence.

2.5. Training

The model is trained in a supervised manner using real device connectivity traces. As we previously mentioned, each trace contains 192 tokens per day the device was connected to the network, corresponding to 5-minute intervals over a full 16-hour day. To enable the model to capture longer temporal dependencies and better predict multi-day behavior, we use a fixed context length of 250 tokens during training. If a device appeared on the network for only one day, we extend its trace by padding with the special `OUT` token to reach this length. Each training sample consists of 250 input tokens, and the expected output is the sequence shifted by one step, allowing the model to learn temporal dependencies and predict the next connection state.

The hyperparameters were chosen following the principles outlined in [11], which analyze the relationship between model performance and hyperparameter choices. These parameters are: the embedding dimension ($d = 384$), number of transformer layers (6), attention heads (6), and dropout rate (0,15), to balance performance and generalization. We use a cross-entropy loss function to compare predictions with actual states, optimizing the model with the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) and a learning rate of $2 \cdot 10^{-4}$. Training runs for up to 50 epochs, with early stopping if validation loss stagnates for 5 consecutive epochs to prevent overfitting. We also found that a batch size of 64 ensures stable and efficient convergence.

Table 1 presents the hyperparameters of the models used to compare against DiWi. The first part summarizes architectural details of Long Short-Term Memory (LSTM), GPT2, and DiWi, including embedding dimension d, number of layers, number of attention heads (for transformers), and approximate total parameter count. The second part breaks down the parameter composition of GPT2 and DiWi, showing how both models share the same Transformer architecture. Despite having a similar total number of parameters, DiWi achieves greater efficiency through a more compact and structured input representation—using fewer tokens and a smaller embedding table—resulting in lower memory usage and faster computation.

3. Performance Evaluation

In this section, we evaluate DiWi's ability to generate realistic data by comparing the statistical relationships in the synthetic dataset to those in the real dataset. This analysis ensures that the model not only generates plausible individual samples but also captures the broader patterns and dependencies found in the original data. We also highlight potential applications of DiWi, particularly in sustainability and user behavior analysis within wireless networks.

Model	d	N_{layers}	N_{heads}
LSTM	384	6	256
GPT2	384	6	6
DiWi	384	6	6

Model	GPT2	DiWi
Embedding Size	$(\lvert\mathcal{V}\rvert + N_{\text{context}}) \times d$	$(\lvert\mathcal{V}\rvert + N_{\text{H}} + N_{\text{M}}) \times d$
Transformer Blocks	$N_{\text{layers}} = 6$	
Attention Block Params	$4 \times d^2$	
Feedforward Block Params	$2 \times d \times 3d$	
LayerNorm Params	$4 \times d$	
Total Tokens	$\sim$ 10.9M	$\sim$ 10.8M

Table 1: Best hyperparameters found for DiWi, and the baseline models GPT2 and LSTM. Token embedding sizes are computed based on vocabulary size and embedding dimension. Total tokens embedded refers to the sum of token categories passed through the input layer. For LSTM, the "N_{heads}" column refers to the hidden dimension.

3.1. Prediction of next AP

We first evaluate DiWi's next-token prediction accuracyfor a fixed input length. Because every token represents a five-minute interval, an input of one token corresponds to five minutes of past activity, whereas 168 tokens span fourteen hours. For the prediction test we fix the context window to the last 12 hours of activity, which corresponds to 144 tokens at five-minute resolution.

We evaluate accuracy on a set of real connectivity traces and compare our model against two baselines—(i) an LSTM and (ii) a standard GPT-2 without our temporal encoding—both of which support next-token prediction and full trace generation, allowing for a fair comparison across the two core tasks.

- **LSTM:** A standard LSTM network serves as a strong recurrent baseline due to its ability to capture temporal dependencies in sequential data.
- **GPT-2:** We include a GPT-2 model with standard sinusoidal positional embeddings [10] to isolate the contribution of our proposed temporal encoding (2.3). It shares the same architecture (aside from the encoding) and training setup as our model.

We provide in Table 2 the performance of each model in the considered campus (first column). To illustrate the ability to generalize of DiWi, we also consider a second campus of the University, consisting in 434 APs (second column) and around 25,000 individuals. According to the results, DiWi consistently outperforms all baselines in both campuses. Although our evaluation was limited to these both environments, we see that DiWi outperforms both baselines.

Model	Campus 1 Acc. [%]	Campus 2 Acc. [%]
LSTM Network	91.2	89.8
GPT-2	91.8	91.9
DiWi	**92.3**	**92.4**

Table 2: Model accuracy when predicting the next connectivity state.

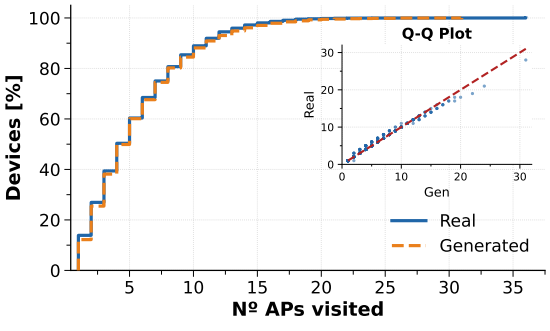
The results in Table 2 show that DiWi achieves the highest accuracy in both environments, outperforming all other models. We attribute this improvement to the richer semantic information captured by the decomposed temporal embeddings. This is particularly evident when comparing GPT2 and DiWi: the two share the same architecture, but only DiWi includes explicit temporal decomposition into hours and minutes. This design allows the model to learn that certain time combinations carry similar meanings — for example, 9:55 AM and 10:00 AM may correspond to transitions between classes or breaks, and thus share behavioral patterns. By embedding hours and minutes separately, DiWi can better generalize across such patterns, recognizing temporal similarities that GPT2, with a flat encoding, cannot easily capture.

In the following sections, we compare key statistics between a sample of real device traces from the test dataset in Campus 1 and synthetic traces generated by our model. Each synthetic trace starts with a token drawn from the distribution of first states seen in the real data—so about 82 % of traces begin in `OUT`—rather than forcing every trace to start the same way. From that seed the model produces a probability vector for the next state; we sample a token from it, slide the context window to keep only the most recent tokens, and repeat the process. Generation stops when the trace reaches a length of 192 tokens (one day from 6:00 AM to 10:00 PM).

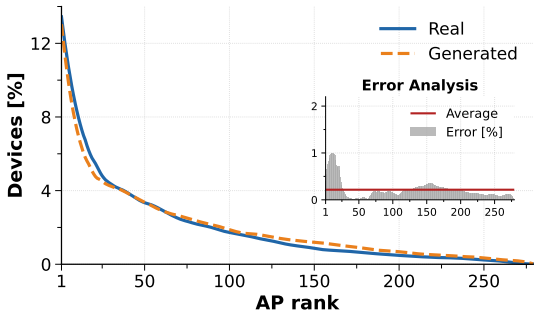
3.2. Device and AP statistics

Here we compare different statistics from the data generated by DiWi and the real dataset. We start by analyzing the number of unique APs a device connects to in a single day, which is illustrated in Fig. 2a. The figure presents the Empirical Cumulative Distribution Function (ECDF) of this metric, providing insights into mobility patterns. The blue line represents the data from the real dataset, while the dashed orange line corresponds to DiWi. The results confirm the similarity of the statistics from both datasets, with 80% of devices connecting to fewer than 10 unique APs per day (some form of localized mobility, where users connect to one or two buildings at most). We also compare both distributions using a Q-Q plot (subplot within Fig. 2a) that illustrates a strong alignment along the reference line (red dashed line), indicating that the synthetic dataset closely replicates the mobility patterns of the real data $R^2 = 0.98$.

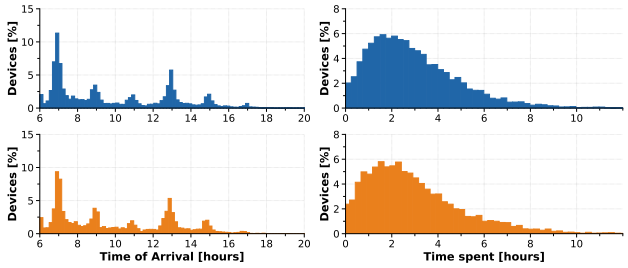
Next, we analyze how many devices connect to a given AP. More specifically, we compute the *rank* of the APs for both datasets, and compare them in Fig. 2b. The figure displays the proportion of devices connected to each AP, ranked from most visited to least. As in the previous case, we depict in blue the real dataset and in orange DiWi. The results reveal that for both



(a) Distribution of visited APs.



(b) Device Distribution Across APs.



(c) Distribution of arrival times. (d) Distribution of time spent.

Figure 2: Trace mobility statistics comparing the synthetic and real traces. (a) Number of APs visited during a day by each device, (b) Ranking of most visited APs, (c) Distribution of time of arrival, and (d) Time spent connected to the network.

datasets a small number of APs handle the majority of connections, indicating that activity is clustered in key locations (e.g., building halls, the library, and cafeterias). Both the real and synthetic datasets exhibit the same trend, the largest difference is less than 1%, demonstrating that the synthetic dataset effectively captures key mobility behaviors.

Next, we analyse temporal patterns. To this aim, we compare for each dataset the time of arrival and the time spent in the network in Fig. 2c and Fig. 2d, respectively. The figures confirm close alignment between the real dataset and DiWi, while revealing both short-term interactions, such as brief check-ins, and extended periods of connectivity, which could correspond to activities like attending classes or studying in the library.

To quantify the similarity between the statistics from DiWi and the real dataset, we use the Kullback–Leibler (KL) divergence [12], where lower values indicate a closer match between distributions. The results from these comparisons are summarized in Table 3. The KL divergence values demonstrate that the statistics from DiWi closely align with the real dataset across spatial and temporal dimensions. As a reference, we also provide in the table the KL divergence values corresponding to the baseline models.

Metric	LSTM	GPT2	DiWi
Nº visited APs	0.26	0.13	**0.015**
AP rank	0.083	0.076	**0.036**
Time spent	0.62	0.65	**0.017**
Time of arrival	0.078	0.059	**0.032**

Table 3: Kullback–Leibler (KL) Divergence between the real dataset and the generated with the LSTM, GPT2, and DiWi.

Interestingly, Table 3 shows that although the accuracy values in Table 2 are similar between DiWi and the baseline models, the KLD between the real and generated data reveals much larger differences. Specifically, DiWi consistently outperforms both LSTM and GPT-2 across all statistics. This highlights DiWi's ability to capture the underlying dynamics of user mobility on campus, not just the next connection prediction.

These results validate the model's ability to generate realistic mobility traces that preserve key structural and behavioral patterns observed in real-world data. We next present a series of scenarios to illustrate how DiWi can support the design of novel network management solutions, without incurring in privacy risks (as demonstrated in Section 4).

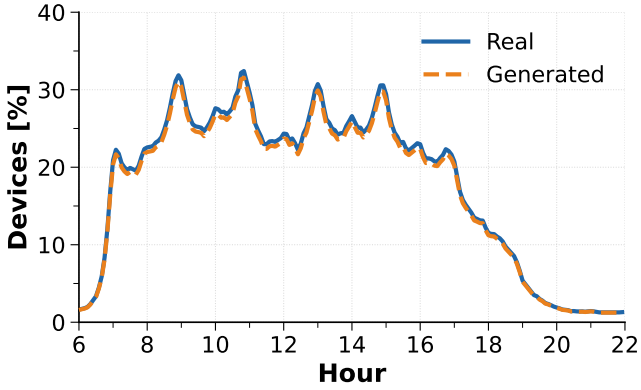
3.3. Scenario 1: Sustainability

In this section, we show how synthetic traces generated by DiWi can be used to analyze space occupancy patterns, like student arrival times and duration of stay, to help optimize HVAC schedules [13], reduce energy use, and improve building management while protecting privacy. Research [14, 15] has shown that HVAC systems can save energy by adjusting settings based on occupancy. While traditional methods, such as camera-based systems [14], pose privacy risks, synthetic traces simulate device mobility without linking real and generated data (see Section 4), providing a safer, privacy-friendly alternative.

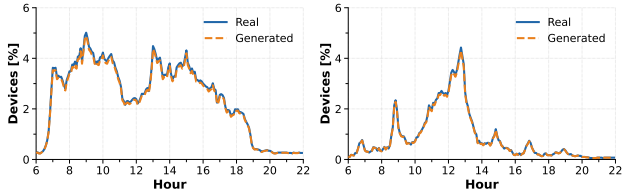
To validate the utility of DiWi, we compare its campus space occupancy with that of the real dataset. More specifically, we depict in Fig. 3a the total number of connected devices in the campus every 5 minutes, with the blue line representing real data and the orange dashed line corresponding to DiWi. The figure confirms that DiWi generated dataset accurately captures overall trends, such as minimal network activity at 6:00 AM, peak activity around 9:00 AM, and a gradual decrease in occupation from 5:00 PM.

In addition to per-Campus occupation, we also analyze the ocuppation in a couple of representative buildings: a classroom building in Fig. 3b) and a cafeteria building in Fig. 3c. According to the results, both datasets expose similar activity patterns, with sharp peaks at the start times of lectures in the case of the classroom building, and at times of coffee breaks and lunch times in the case of the cafeteria.

These results confirm that DiWi can be used to design intelligent HVAC systems without compromising user privacy.



(a) Occupancy of the Campus.



(b) Occupancy of Classrooms. (c) Occupancy of Cafeteria.

Figure 3: Number of devices connected during a day. (a) Overall occupancy, (b) Occupancy in Building 1, (c) Occupancy in Building 5.

For instance, it is clear that the system could be switched off at 7:00 AM, as fewer than 5% of devices remain connected across the university at that time. Furthermore, more sophisticated approaches could be designed, with a proper isolation of zones, e.g., the HVAC system in the the cafeteria between 3:00 PM and 5:00 PM could be switched to a low power mode (the actual design of these policies is outside the scope of this work).

3.4. Scenario 2: Mobility Management

Here we evaluate DiWi's ability to support the design of advanced mobility management schemes, i.e. mechanisms forecasting the movement of devices, to support the design of better mobility schemes or the design of infrastructure on demand approaches by activating APs as needed. More specifically, here we focus on the ability of DiWi to analyze the connection history of a device, which we refer to as context, and then forecast its next connection point. To assess the model's predictive performance, we evaluate how its accuracy varies with the amount of context provided for inference. To compute this, we first select the `context_length` used by the model for inference. Then, we sample a sequence of the chosen length for each device and use it to predict the next connection. This process is repeated across different context lengths, allowing us to assess how the model performs under varying amounts of historical data.

We represent in Fig. 4 the accuracy as a function of `context_length`, measured in hours, where each hour corresponds to 12 tokens (with each token representing a 5-minute interval). The shaded regions around the curve represent twice the standard deviation (2σ), indicating the variability in accuracy across samples. The results illustrate that accuracy improves as context length increases, but gains level off beyond 2 hours. This suggests that while additional context initially enhances predictions, there is a point of diminishing returns. We note that there is a slight increase in accuracy after 10 hours of context, which we conjecture that may be caused by the ability of DiWi to leverage context from the previous day to predict the behavior on a given next day.

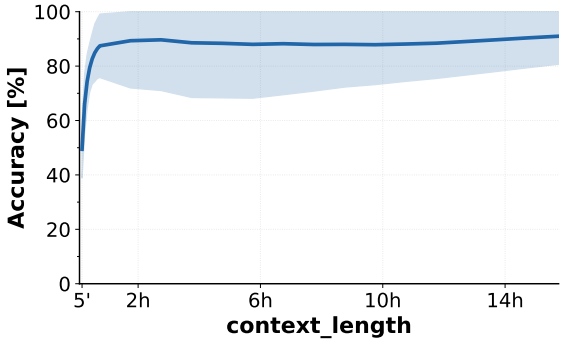


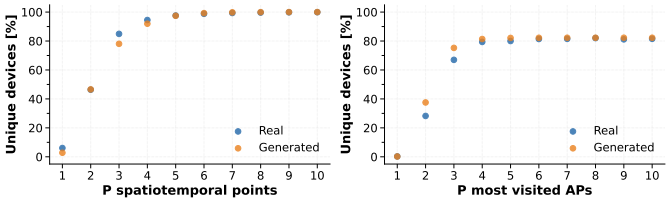
Figure 4: Accuracy as function of the `context_length`.

3.5. Scenario 3: Anonymity schemes

In this section we describe how DiWi can support the design of privacy-preserving schemes. This is motivated by previous studies, such as [6] and [16], which have identified some variables related to network activity that can be used to unequivocally identify a device (without using any device ID). In this section, we briefly introduce two different metrics regarding the identifiability of a device, and illustrate how the synthetic trace provided by DiWi shares the same results as the real-life dataset.

First, we define spatiotemporal unicity as the number of randomly chosen spatiotemporal points in a device's trace that make it uniquely identifiable. Following this definition, the uniqueness of a dataset is the average unicity across all devices in that dataset, which summarizes how identifiable the devices are. We represent in Fig. 5a the unicity for an increasing number of randomly chosen spatiotemporal points. According to the figures, both the real dataset and DiWi produce vary similar results. With four spatiotemporal points, 93% of devices in the real dataset are unique, compared to 91% in the synthetic dataset. These results confirm that DiWi preserves the privacy dynamics of the real data, making it a useful tool for evaluating privacy risks in mobility networks and designing new Randomized and Changing MAC (RCM) schemes.

Second, we can also defined a device's uniqueness based on its most frequently visited APs. To do this, we create a fingerprint for each device using the list of APs where it spends most time. Following this, a device is considered unique if no other device has the same list of top p APs in that order. We represent in Fig. 5b the relative number of unique devices in a trace as p increases. As in the previous case, both the real dataset and DiWi follow a similar pattern, e.g., with fingerprints of 4



(a) (b)

Figure 5: Metrics for evaluating the uniqueness of device mobility patterns.

APs, 77% of devices are unique in the real dataset, compared to 81% in the synthetic dataset. Beyond $p = 5$, uniqueness levels off, suggesting that adding more APs to the fingerprint does not significantly increase a device's distinctiveness.

4. Privacy Assessment

In this section, we assess the potential risks of real-life data from the use of DiWi. This is essential to confirm that the generated data has the ability to mimic mobility patterns (as illustrated above) while preventing the exposure of sensitive information, making it suitable for the scenarios discussed in Section 3. Our assessment is two-fold: first, we perform an empirical evaluation of the privacy risks, which serves to provide some context about the small risks from using DiWi (but does not provide formal guarantees), and second, we illustrate how DiWi can be extended to support formal privacy guarantees.

4.1. Exact matches

First, we assess the risk of the model leaking complete real traces, i.e., the possibility of generating a trace that is identical to one from the training dataset. This scenario represents a significant concern as it would indicate two problems: (1) the model is overfitting and failing to generalize, and (2) the behavior of a real device is being exposed in the synthetic dataset. Such outcomes would compromise the purpose of using synthetic data by diminishing its privacy-preserving benefits.

To assess this risk, we analyze the trace generation process described in Section 2.4, where traces are generated sequentially based on the model's learned probabilities. Assuming independence, the likelihood of reproducing a training trace is computed as the product of probabilities at each step (or equivalently, the sum of log probabilities). During synthetic dataset generation, we typically select the most likely token at each step. However, to evaluate the probability of reproducing a training trace, we take a different approach: instead of selecting the most likely AP, we use the actual AP observed in the training trace. By summing the log probabilities of these events across the sequence, we estimate the likelihood of generating an exact replica of a training trace.

Figure 6 presents the distribution of log probabilities for generating traces from the training set (blue) and the synthetic dataset (orange). The x-axis represents the log probability, while the y-axis shows the percentage of training traces with a given probability. The results indicate that the average probability of exactly reproducing a training trace is approximately 10^{-120}. This suggests that, on average, the model would need to generate 10^{120} traces to replicate a single training trace, reinforcing that the model generalizes patterns rather than memorizing them.

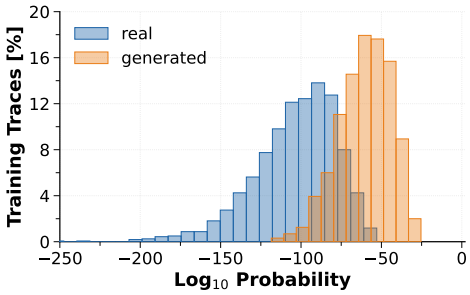


Figure 6: Distribution of the likelihood of generating a trace.

The orange distribution represents the probability of generating traces from the synthetic dataset used in this study. While some synthetic traces exhibit probability values close to those in the training set, the overlap remains limited. The overall probability of generating new, distinct traces remains dominant, indicating that the model primarily creates new patterns rather than reproducing previously seen data.

Interestingly, some traces are more likely to be replicated than others. This variation, discussed further in the next section, may result from certain AP sequences appearing more frequently in the training dataset. While this imbalance reflects the real-world distribution of user activity and preferences on campus, it is also necessary for the model to accurately capture device behavior. By representing these natural patterns, the synthetic dataset maintains the fidelity of user behavior without directly exposing sensitive information.

4.2. Closeness between datasets

Above we have analyzed the probability of the DiWi generating an exact replica of a real device's trace, which is practically negligible. However, the model may still produce synthetic traces that closely resemble real ones. For instance, consider two devices with similar mobility patterns: both connect to a sequence of the same APs but differ at one or two points. While these traces are not identical, their strong similarities may result in the synthetic data inadvertently revealing aspects of the real device's mobility pattern.

To assess this risk, we adapt inter- and intra-similarity analysis [12], originally used for benchmarking dataset quality, to evaluate privacy leakage. Specifically, we measure the "closeness" between real-to-real traces and real-to-synthetic traces. By comparing the inter- and intra-similarity scores between pairs of real devices and pairs of real and synthetic devices, we evaluate whether the synthetic data retains identifiable patterns from the real data. If the synthetic traces are too similar to the real ones, privacy risks may emerge. Conversely, traces that are too dissimilar could make it obvious which traces are synthetic. Striking a balance ensures that the synthetic data preserves the utility of the real data without compromising privacy.

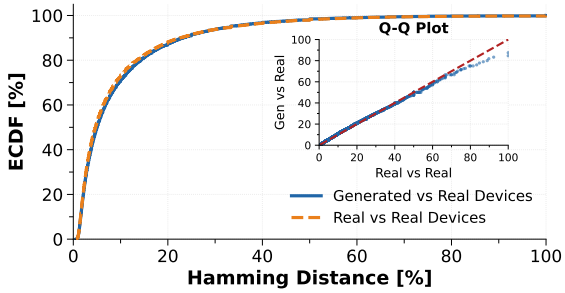


Figure 7: Hamming Distance between traces.

To quantify similarity, we use the Hamming distance[3] between two traces. This metric counts the number of matching APs in the same order, excluding instances of the OUT state. This exclusion is justified because the OUT state does not provide meaningful information about a device's mobility. Using this metric, we calculate the percentage of a real trace that is "replicated" in a synthetic trace, providing a concrete measure of how much real mobility data is reflected in the synthetic dataset.

Figure 7 presents the ECDF of the proportion of training traces that overlap with other training traces and with generated traces. In both cases, 90% of training traces intersect with at most 20% of others, indicating that generated traces are as similar to real traces as real traces are to each other. The QQ plot further confirms a strong alignment between the distributions, suggesting that the generated traces maintain a balance between similarity and diversity, reducing the risk of privacy breaches.

As noted earlier, some mobility patterns are more unique than others. These common movement sequences are not a major concern since they are shared by many traces, making reidentification difficult. The real issue is the unique sequences in the generated traces that match specific real devices, posing a privacy risk.

To assess this risk, we analyze the overlap between generated traces and the training set. For each overlapping subsequence (Figure 7), we determine how many device traces from the training set contain that same subsequence. This allows us to assess whether the overlap is distributed across multiple training devices, or if it is limited to a single user, which could indicate potential leakage of identifying patterns.

Our analysis shows that only 2% of exposed training patterns uniquely identify a single device. This suggests that while some subsequences are exposed, they are usually shared among multiple devices, keeping the risk of identifiability low.

4.3. Membership inference

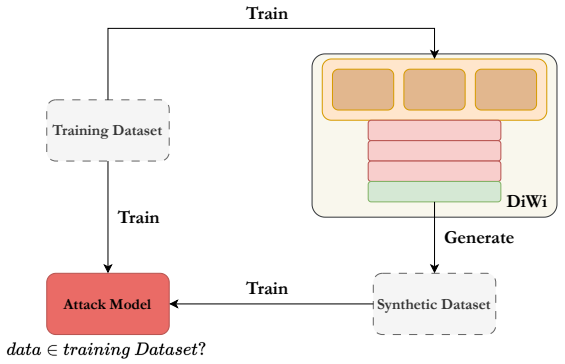
We now examine a scenario in which an attacker has partial access to real device traces and the ability to query the model to generate synthetic traces.

[3]Technically, this is more of a similarity metric than a true distance measure. $|L| - H(L)$

In this attack, the goal is to train a classifier that determines whether a given trace comes from the real training dataset (D_{tr}) or the synthetic dataset (D_{syn}). The attacker has access to both synthetic traces and a portion of the training data but relies solely on the information within the traces to distinguish between them.

To make this distinction, the attacker trains a classifier to assign a label of 1 to traces from the training set and 0 to synthetic traces. The classification is based on device behavior profiles, which include features such as time of arrival, duration of connection, most frequently used AP, and the number of APs visited.

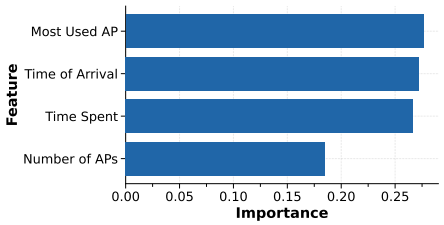
As depicted in Figure 8a, the attacker first accesses real traces from DiWi's training dataset, constructs profiles for these traces, and assigns them the label 1. The attacker then queries the model to generate synthetic traces, computes profiles for these traces, and assigns them the label 0. Using this labeled dataset, the attacker trains a classifier to conduct the membership inference attack. By employing an explainable model, the attacker can identify which features the model uses to make its decisions, uncovering correlations and patterns that may reveal potential vulnerabilities in the synthetic dataset.



(a) Membership Inference Attack (MIA) scheme.

Metric	%
Accuracy	56
Precision	54
Recall	54
F1-Score	54

(b) Succes of MIA.



(c) Feature importance.

Figure 8: Results of the Membership Inference Attack (MIA). (a) MIA schema, (b) Accuracy of the model, (c) Feature importance.

In Figure 8b, we present the results for decision trees and random forests. The observed accuracies are 52% and 56%, respectively, which are only slightly better than random guessing. In a binary classification task like distinguishing between real and synthetic traces, a random classifier would achieve approximately 50% accuracy. The minimal improvement over this baseline suggests that the models struggle to identify meaningful patterns or signals to reliably distinguish between the two

classes. This indicates that the synthetic dataset closely replicates the distribution of connectivity patterns observed in the real dataset.

Figure 8c presents the feature importance of the Random Forest model used for the membership inference attack, which determines whether a given trace was part of the training dataset. The results show that Most Used AP is the most influential feature, indicating that access point usage patterns contribute to distinguishing between real (training) and synthetic traces. However, its importance is not significantly higher than other features, and the overall distribution of feature importance is relatively uniform.

These results suggest that the membership inference attack struggles to distinguish between real and synthetic traces, with accuracy barely exceeding random guessing. The lack of a strong signal in feature importance further indicates that the generated dataset closely mirrors real connectivity patterns without exposing identifiable information. This reinforces the idea that DiWi effectively preserves privacy while maintaining realistic mobility patterns.

4.4. Formal Privacy guarantees

So far, we have assessed the privacy implications of our generator through empirical evaluations, measuring how likely it is to produce traces that resemble those seen during training. While these analyses provide useful indicators of re-identification risk, they do not offer formal guarantees. In this section, we address this limitation by exploring the integration of differential privacy (DP) into the training process. We analyze how applying DP affects the behavioral fidelity of the generated traces, and quantify the trade-off between privacy strength and data utility.

Differential privacy, formalized in 2006 [17], offers a rigorous framework for protecting individual data while enabling useful data analysis. In the context of deep learning, DP is commonly applied by modifying stochastic gradient descent to use a differentially private variant (DP-SGD) [18]. At each training step, per-example gradients are clipped to a fixed norm, and Gaussian noise is added to their sum. This ensures that no single trace has a disproportionate influence on the model, embedding privacy guarantees directly into training.

Naturally, clipping and noise introduce bias and variance into gradient estimates, potentially degrading model performance. To evaluate this privacy–utility trade-off, we measure trace fidelity, this is, how realistic the generated trajectories remain as privacy constraints increase.

We train our model using DP-SGD, which enforces differential privacy by clipping each per-example gradient to a fixed norm C and adding Gaussian noise with variance $\sigma^2 C^2$. The cumulative privacy loss after T updates is tracked using the Rényi moments accountant [19], yielding a formal (ε, δ)-differential privacy guarantee: for any two training datasets that differ by a single trace, the probability that the algorithm produces a given output changes by at most a factor e^{ε}, plus an additive term δ [17]. This means the presence or absence of any one trace has only a negligible impact on the model's output, except with very low probability, its influence is limited to a small multiplicative factor. Following common practice, we fix $\delta = 1/N$, where N is the number of training traces [20].

To interpret the privacy budget ε, we follow widely adopted thresholds [21]: values $\varepsilon \leq 1$ indicate very high privacy, $1 < \varepsilon \leq 4$ reflect high to moderate privacy, and $\varepsilon > 4$ correspond to low privacy. These categories guide our evaluation in Figure 9, where we assess models trained with $\varepsilon = \{0.9, 2, 5\}$. The figure reports the similarity—measured via Hamming distance—between validation trajectories and synthetic traces generated under different privacy budgets. As expected, the non-private baseline (blue) achieves the highest fidelity, while fidelity gradually declines for models trained with stronger privacy guarantees: orange ($\varepsilon = 5$), green ($\varepsilon = 2$), and red ($\varepsilon = 0.9$).

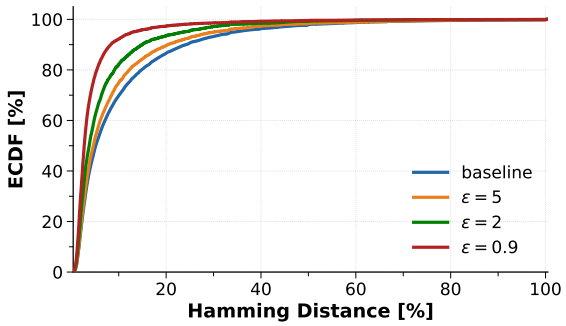


Figure 9: Hamming Distance between real and synthetic traces under different privacy budgets.

The figure illustrates the privacy–utility trade-off: With $\varepsilon = 0.9$ (red, very high privacy), over 90% of traces intersect less than 10% with real ones, showing low fidelity. With $\varepsilon = 2$ (green, moderate privacy), there is a better balance between utility and privacy. With $\varepsilon = 5$ (orange, low privacy), the generated traces closely resemble real ones but offer weaker privacy guarantees.

This analysis confirms that as the privacy budget increases, the model better captures patterns in the training data—yet also becomes more susceptible to potential privacy risks. Thus, selecting ε involves balancing privacy protection against the desired realism and utility of the generated traces.

5. Related Work

Randomized and Changing MAC addresses. By chaning the device's MAC address over time, this technique prevents consistent identification of devices, thereby aming the ability of third parties to track user activity. However, the effectiveness of this approach depends on the frequency of MAC address rotations. Studies, such as [22], emphasize that while frequent rotations improve privacy, they come with significant performance drawbacks. High rotation frequencies can lead to inefficient device authentication, prolonged handover processes, and increased network overhead, ultimately degrading the quality of service for users.

Furthermore, the adoption of randomized MAC addresses undermines the ability to track users for beneficial applications. Mobility tracking plays a crucial role in optimizing resource allocation, managing network traffic, and enabling sustainability-focused initiatives like energy-efficient network planning. As noted in [8], the inability to persistently identify devices due to randomized MAC addresses limits the development of such applications. This trade-off poses a challenge for network operators, who must balance the need for user privacy with the operational benefits derived from mobility insights.

Privacy of Network Mobility Data is challenging due to the intrinsic identifiability of human mobility traces. Studies such as [6] and [16] have demonstrated that mobility patterns are inherently unique, making re-identification possible even with pseudonymization. For instance, [6] showed that as few as four spatiotemporal points can uniquely identify 95% of individuals in a dataset, while [16] revealed that Wi-Fi probe requests can infer social relationships between individuals based on shared or rare SSIDs. These findings emphasize the significant risks of privacy breaches when handling mobility data, even with anonymization.

Building on this, research such as [23] has highlighted how combining anonymized mobility traces with external datasets further increases re-identification risks. The predictability of human mobility, quantified at up to 88% in [24], was analyzed in a mobile network deployed across an entire country. Together, these studies reveal a multi-faceted challenge: while mobility data offers valuable insights, its use exposes sensitive patterns that are difficult to protect. These risks underscore the pressing need for robust privacy-preserving techniques, such as differential privacy and synthetic data generation, to enable safe utilization of mobility datasets without compromising individual privacy [25].

Private Data Publishing (PDP) seeks to address privacy concerns by enabling the release of network datasets in a manner that protects user privacy while preserving data utility for analysis. Techniques such as differential privacy [17], data anonymization [26], and synthetic data generation have been widely explored to achieve this balance [27].

Among these approaches, synthetic data generation using generative models has emerged as a promising solution for PDP [28]. Notably, [29] presents a generative pre-trained Transformer specifically designed for network traffic data, including packets and flows. This model achieves remarkable results in both understanding tasks, such as traffic classification, and generative tasks, such as simulation. Their results show that the synthetic datasets generated by the model closely replicate the behavior of the training data, offering a reliable way to preserve privacy while retaining utility.

Foundational Transformer models have become a dominant architecture for modeling sequential data, particularly in Natural Language Processing (NLP), since their introduction in [10]. They process sequences as discrete units called tokens, representing elements such as words or punctuation marks, depending on the tokenization strategy. Transformers are foundational for Large Language Models (LLMs) like BERT [30], GPT [31], and LLama [32].

The core of the Transformer architecture is the self-attention mechanism, which computes dependencies between all tokens in a sequence. While this enables the model to effectively capture long-term relationships, its $O(L^2)$ computational complexity poses challenges for longer sequences [33]. Tokens are first represented as one-hot vectors and transformed into high-dimensional embeddings, which encode semantic relationships [34]. Positional encodings are then added to embeddings to provide sequence order information [10].

Transformers stack layers combining two components: the Multi-Head Self-Attention Mechanism, which identifies relationships between tokens through Query, Key, and Value vectors [10], and the Feed-Forward Network (FFN), which refines token representations through linear transformations and non-linear activations. These capabilities make Transformers versatile and applicable beyond NLP, finding use in multivariate time series analysis [35], music generation [36], and network traffic modeling like [29].

Synthetic mobility data generation is shifting from passive observation to active simulation. Instead of only recording how devices, vehicles, or people move, generative AI now allows us to create synthetic traces to test infrastructure, or evaluate congestion-mitigation strategies.

Several recent systems tackle synthetic mobility data generation. [37] guides a diffusion model with detailed street maps to create realistic GPS-level paths for cities it has never seen. [38] couples a multimodal predictor with a reinforcement-learning controller so that simulated vehicles respect road geometry and basic kinematics. [39] embeds a city-scale trajectory-flow graph into a Transformer to suggest a user's next point of interest within the next 30 minutes. Although their techniques differ, all three methods depend on explicit geometric priors such as road networks, movement constraints, or flow maps, and they either output continuous coordinates or stop at single-step prediction.

Transformer-based mobility generators continue to improve fidelity, yet they still rely on rich structural cues. [13] augments a hierarchical Transformer with building type, floor labels, and room-level topology to forecast multi-hour indoor trajectories.[40] treats path completion as masked-token infilling and conditions its language-style Transformer on geohashes, road-network adjacency, and elapsed-time hints to preserve spatio-temporal realism. [41] shapes its attention weights with explicit distance and inter-visit-time kernels to recommend the next point of interest. In contrast, DiWiconsumes only raw Wi-Fi association sequences, avoiding maps, building metadata, simulators, and cross-user graphs. Its autoregressive Transformer can roll out an entire day of activity, enabling traffic replay and what-if analysis well beyond single-step forecasting. Crucially, we make privacy a first-class objective: alongside standard quality metrics we evaluate empirical leakage tests and formal differential-privacy accounting, a dimension previous work has overlooked.

Our contributions include leveraging Transformer-based models to generate privacy-preserving synthetic mobility data, enabling space utilization analysis and network planning without persistent device identification. Unlike MAC address ran-

domization, our approach maintains mobility insights while reducing re-identification risks. We assess privacy through leakage analysis and membership inference attacks, confirming that the generated data preserves user anonymity. Additionally, we extend Transformer-based models beyond packet-level analysis, optimizing temporal representations to better capture complex movement patterns in mobility traces.

The work most closely related to ours is [13], which also employs a Transformer model for modeling user mobility patterns. However, while their approach uses a base model with multimodal embeddings, we introduce a specialized encoding designed specifically for temporal data. This enhances the model's ability to capture time-dependent mobility behaviors, improving the accuracy.

Most prior works on modeling network data [13, 24, 29] focus on improving predictive performance, with little attention to privacy risks. In contrast, our study explicitly evaluates the privacy implications of synthetic data generation. Given the sensitivity of user traces, assessing whether models inadvertently expose identifiable patterns is crucial in these environments.

6. Conclusions

In this work, we have analyzed a dataset capturing the mobility patterns of users connected to the campus WLAN. Our objective is to model device activity using Transformer models, leveraging their ability to process sequential data effectively.

Our model uses a Transformer architecture to analyze and predict device connectivity sequences while efficiently encoding temporal information to reduce parameters and improve generalization. This enables the generation of realistic spatiotemporal sequences that mimic real user behavior. We validate its performance through a comparative analysis with real traces and explore applications in space optimization, intelligent building management, user occupancy prediction, network mobility forecasting, and privacy evaluation in wireless networks.

To ensure the generated data does not compromise the privacy of real devices, we conducted a thorough analysis of potential risks. These included evaluation of empirical metrics such as, direct leakage (found to be negligible), similarity searches between real and generated traces, and membership inference attacks. In all cases, our results show that the model effectively generalizes the behavior of real traces without exposing identifiable patterns that could link back to individual users. We further anlyse the effect of differential privacy in the generation capabilities of the model.

Acknowledgements

This article is part of the project 6GINSPIRE PID2022-137329OB-C42, funded by MCIN/AEI/10.13039/501100011033 and the TUCAN6-CM project (TEC-2024/COM-460), funded by CM (ORDEN 5696/2024). This work is also partly funded by the SNS JU under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101139198, iTrust6G project.

References

[1] A. Trivedi, J. Gummeson, P. Shenoy, Empirical characterization of mobility of multi-device internet users (2020).
URL https://arxiv.org/abs/2003.08512

[2] Y. Zhang, User mobility from the view of cellular data networks, in: IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, 2014, pp. 1348–1356. doi:10.1109/INFOCOM.2014.6848068.

[3] J. Plachy, Z. Becvar, E. C. Strinati, Dynamic resource allocation exploiting mobility prediction in mobile edge computing, in: 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016, pp. 1–6. doi:10.1109/PIMRC.2016.7794955.

[4] J. Pan, R. Jain, S. Paul, A survey of energy efficiency in buildings and microgrids using networking technologies, IEEE Communications Surveys & Tutorials 16 (2014) 1709–1731.
URL https://api.semanticscholar.org/CorpusID:16586568

[5] YData, How is Diversity Preserved while Ensuring Privacy in Synthetic Data?, https://ydata.ai/resources/syntheticdata-quality-metrics, accessed: 2025-05-07 (2023).

[6] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, V. D. Blondel, Unique in the Crowd: The privacy bounds of human mobility, Scientific Reports 3 (1) (2013) 1376. doi:10.1038/srep01376.
URL https://doi.org/10.1038/srep01376

[7] H. Surendra, S. MohanH, A review of synthetic data generation methods for privacy preserving data publishing, International Journal of Scientific & Technology Research 6 (2017) 95–101.
URL https://api.semanticscholar.org/CorpusID:67051890

[8] J. Henry, Y. Lee, Randomized and changing mac address: Context, network impacts, and use cases, Internet Engineering Task Force (IETF) Draft, available online at https://www.ietf.org/archive/id/draft-ietf-madinas-use-cases-18.html (January 2024).
URL https://www.ietf.org/archive/id/draft-ietf-madinas-use-cases-18.html

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (8) (2019) 9.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need (2023). arXiv:1706.03762.
URL https://arxiv.org/abs/1706.03762

[11] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, D. Amodei, Scaling laws for neural language models, ArXiv abs/2001.08361 (2020).
URL https://api.semanticscholar.org/CorpusID:210861095

[12] M. Wolf, J. Tritscher, D. Landes, A. Hotho, D. Schlör, Benchmarking of synthetic network data: Reviewing challenges and approaches, Computers & Security 145 (2024) 103993. doi:https://doi.org/10.1016/j.cose.2024.103993.
URL https://www.sciencedirect.com/science/article/pii/S0167404824002980

[13] A. Trivedi, K. Silverstein, E. Strubell, P. Shenoy, M. Iyyer, Wifimod: Transformer-based indoor human mobility modeling using passive sensing, in: Proceedings of the 4th ACM SIGCAS Conference on Computing and Sustainable Societies, COMPASS '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 126–137. doi:10.1145/3460112.3471951.
URL https://doi.org/10.1145/3460112.3471951

[14] M. Aftab, C. Chen, C.-K. Chau, T. Rahwan, Automatic hvac control with real-time occupancy recognition and simulation-guided model predictive control in low-cost embedded system, Energy and Buildings 154 (2017) 141–156. doi:https://doi.org/10.1016/j.enbuild.2017.07.077.
URL https://www.sciencedirect.com/science/article/pii/S0378778817305091

[15] D. Ardiyanto, M. Pipattanasomporn, S. Rahman, N. Hariyanto, Suwarno, Occupant-based hvac set point interventions for energy savings in buildings, in: 2018 International Conference and Utility Exhibition on Green

Energy for Sustainable Development (ICUE), 2018, pp. 1–6. doi:10.23919/ICUE-GESD.2018.8635595.

[16] M. Cunche, M. A. Kaafar, R. Boreli, I know who you will meet this evening! Linking wireless devices using Wi-Fi probe requests, in: 2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012, pp. 1–9. doi:10.1109/WoWMoM.2012.6263700.

[17] C. Dwork, Differential privacy, in: M. Bugliesi, B. Preneel, V. Sassone, I. Wegener (Eds.), Automata, Languages and Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 1–12.

[18] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 308–318. doi:10.1145/2976749.2978318.
URL https://doi.org/10.1145/2976749.2978318

[19] I. Mironov, Rényi differential privacy, in: 2017 IEEE 30th Computer Security Foundations Symposium (CSF), IEEE, 2017, p. 263–275. doi:10.1109/csf.2017.11.
URL http://dx.doi.org/10.1109/CSF.2017.11

[20] M. Van Dijk, N. V. Nguyen, T. N. Nguyen, L. M. Nguyen, P. H. Nguyen, Proactive DP: A multiple target optimization framework for DP-SGD, in: R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, F. Berkenkamp (Eds.), Proceedings of the 41st International Conference on Machine Learning, Vol. 235 of Proceedings of Machine Learning Research, PMLR, 2024, pp. 49029–49077.
URL https://proceedings.mlr.press/v235/van-dijk24a.html

[21] N. I. of Standards, Technology, Differential privacy: A primer for practitioners, Tech. Rep. NIST IR 8288, U.S. Department of Commerce (2020). doi:10.6028/NIST.IR.8288.

[22] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, F. Piessens, Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms, in: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, ASIA CCS '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 413–424. doi:10.1145/2897845.2897883.
URL https://doi.org/10.1145/2897845.2897883

[23] P. Golle, K. Partridge, On the anonymity of home/work location pairs, in: H. Tokuda, M. Beigl, A. Friday, A. J. B. Brush, Y. Tobe (Eds.), Pervasive Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 390–397.

[24] X. Lu, E. Wetter, N. Bharti, A. J. Tatem, L. Bengtsson, Approaching the Limit of Predictability in Human Mobility, Scientific Reports 3 (1) (2013) 2923. doi:10.1038/srep02923.
URL https://doi.org/10.1038/srep02923

[25] R. Mendes, M. Cunha, J. P. Vilela, Impact of frequency of location reports on the privacy level of geo-indistinguishability, Proceedings on Privacy Enhancing Technologies 2020 (2) (2020) 379–396. doi:10.2478/popets-2020-0032.

[26] A. Aleroud, F. Yang, S. C. Pallaprolu, Z. Chen, G. Karabatis, Anonymization of network traces data through condensation-based differential privacy, Digital Threats 2 (4) (Oct. 2021). doi:10.1145/3425401.
URL https://doi.org/10.1145/3425401

[27] A. Majeed, S. Lee, Anonymization techniques for privacy preserving data publishing: A comprehensive survey, IEEE Access 9 (2021) 8512–8545. doi:10.1109/ACCESS.2020.3045700.

[28] B. Van Breugel, Z. Qian, M. Van Der Schaar, Synthetic data, real errors: How (Not) to publish and use synthetic data, in: A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, J. Scarlett (Eds.), Proceedings of the 40th International Conference on Machine Learning, Vol. 202 of Proceedings of Machine Learning Research, PMLR, 2023, pp. 34793–34808.
URL https://proceedings.mlr.press/v202/van-breugel23a.html

[29] X. Meng, C. Lin, Y. Wang, Y. Zhang, Netgpt: Generative pretrained transformer for network traffic, arXiv preprint arXiv:2304.09513 (2023).

[30] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding (2019). arXiv:1810.04805.
URL https://arxiv.org/abs/1810.04805

[31] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (8) (2019) 9.

[32] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).

[33] R. Child, S. Gray, A. Radford, I. Sutskever, Generating long sequences with sparse transformers, arXiv preprint arXiv:1904.10509 (2019).

[34] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation (2014). arXiv:1406.1078.
URL https://arxiv.org/abs/1406.1078

[35] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, C. Eickhoff, A transformer-based framework for multivariate time series representation learning, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, KDD '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 2114–2124. doi:10.1145/3447548.3467401.
URL https://doi.org/10.1145/3447548.3467401

[36] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, D. Eck, Music transformer (2018). arXiv:1809.04281.
URL https://arxiv.org/abs/1809.04281

[37] Z. Tao, W. Xu, X. You, Map2traj: Street map piloted zero-shot trajectory generation with diffusion model, arXiv preprint arXiv:2407.19765 (2024).

[38] Q. Zhang, Y. Gao, Y. Zhang, Y. Guo, D. Ding, Y. Wang, P. Sun, D. Zhao, Trajgen: Generating realistic and diverse trajectories with reactive and feasible agent behaviors for autonomous driving, IEEE Transactions on Intelligent Transportation Systems 23 (12) (2022) 24474–24487. doi:10.1109/TITS.2022.3202185.

[39] S. Yang, J. Liu, K. Zhao, Getnext: Trajectory flow map enhanced transformer for next poi recommendation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 1144–1153. doi:10.1145/3477495.3531983.
URL https://doi.org/10.1145/3477495.3531983

[40] S.-L. Hsu, E. Tung, J. Krumm, C. Shahabi, K. Shafique, Trajgpt: Controlled synthetic trajectory generation using a multitask transformer-based spatiotemporal model, in: Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems, 2024, pp. 362–371.

[41] Y. Luo, Q. Liu, Z. Liu, Stan: Spatio-temporal attention network for next location recommendation, in: Proceedings of the web conference 2021, 2021, pp. 2177–2185.