Knowledge Enhanced GAN for IoT Traffic Generation

Shuodi Hui[1], Huandong Wang[1§], Zhenhua Wang[1], Xinghao Yang[1],
Zhongjin Liu[2], Depeng Jin[1], Yong Li[1]
[1]Beijing National Research Center for Information Science and Technology (BNRist),
Department of Electronic Engineering, Tsinghua University
[2]National Computer Network Emergency Response Technical Team/Coordination Center of China

ABSTRACT

Network traffic data facilitates understanding the Internet of Things (IoT) behaviors and improving IoT service quality in the real world. However, large-scale IoT traffic data is rarely accessible, and privacy issues also impede realistic data sharing even with anonymous personal identifiable information. Researchers propose to generate synthetic IoT traffic but fail to cover the multiple services provided by widespread real-world IoT devices. In this work, we take the first step to generate large-scale IoT traffic via a knowledge-enhanced generative adversarial network (GAN) framework, which introduces both the semantic knowledge (e.g., location and environment information) and the network structure knowledge for various IoT devices via a knowledge graph. We use a condition mechanism to incorporate the knowledge and device category for IoT traffic generation. Then, we adopt LSTM and a self-attention mechanism to capture the temporal correlation in the traffic series. Extensive experiment results show that the synthetic IoT traffic datasets generated by our proposed model outperform state-of-art baselines in terms of data fidelity and applications. Moreover, our proposed model is able to generate realistic data by only training on small real datasets with knowledge enhanced.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; • **Networks** → *Network simulations*; • **Information systems** → *Traffic analysis.*

KEYWORDS

IoT, traffic generation, GAN, knowledge graph

ACM Reference Format:
Shuodi Hui[1], Huandong Wang[1§], Zhenhua Wang[1], Xinghao Yang[1],, Zhongjin Liu[2], Depeng Jin[1], Yong Li[1]. 2022. Knowledge Enhanced GAN for IoT Traffic Generation. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3485447.3511976

1 INTRODUCTION

Internet of Things (IoT) expands the way humans perceive and interact with the world via connecting various sensors, actuators, and computing devices to the Internet. With the increasing popularity of IoT applications, various devices are connected to the Internet and serve in smart energy projects, home automation, manufacturing, commerce, etc. [9]. Under this circumstance, understanding the behavior of IoT devices and improving IoT service quality based on IoT traffic data attract increasing attention. Specifically, IoT traffic data contains all the command and feedback between users and IoT devices, reflecting their activities, and thus contributes to numerous applications, including behavior analysis of specific IoT devices [3, 18, 21], privacy leakage identification [2, 10, 26, 30, 34], and IoT device management [7, 27, 32, 37], etc.

At the same time, the growing number of software standards and frameworks of IoT devices designed by countless companies for different applications have increased the fragmentation of IoT devices and platforms. To counter the fragmentation, the Web of Things (WoT) pursues to integrate the IoT devices seamlessly with the Web technologies [15], where IoT traffic data also plays an important role. Specifically, the IoT traffic data across various platforms and application domains facilitates the WoT to comprehend the IoT devices and provide appropriate web-based communication mechanisms. For example, devices with heavy communication load (e.g., smart camera) and devices requiring high reliability (e.g., point-of-sale) demand different communication protocols [25].

However, most of the IoT traffic datasets in the existing studies are collected in laboratories or simplex application scenes because large-scale IoT traffic data is accessible for only a few organizations, e.g., Internet service providers, IoT service providers. Unfortunately, these organizations are reluctant to share realistic data in consideration of privacy. Although some organizations anonymize the datasets via removing personal identifiable information, this naive method is demonstrated to be vulnerable to a number of De-Anonymization (DA) attacks [20, 41, 44]. Under this circumstance, generating synthetic IoT traffic becomes an appealing solution. The generated IoT traffic can retain the features of IoT behaviors without real personal identifiable information, supporting IoT and WoT applications while avoiding privacy leakage.

Recently, Nguyen-An et al. [29] propose IoTTGen to generate synthetic traffic for smart home and bio-medical IoT environments. This model requests configuration for each IoT device before traffic generation, in which the packet size, port number, payload, and arrival time interval are given as fixed parameters, whereas dynamic in practice. To generate IoT traffic dynamically, Shahid et al. [36] combine an auto-encoder with generative adversarial networks (GAN) to generate sequences of packet sizes that correspond to real traffic flows produced by a Google Home Mini (a smart speaker), which is actively used for a week. Nevertheless, the previous two works conduct experiments in laboratories for simplex scenes, which require the specific configurations of devices and parameters in traffic data. For the widespread IoT devices providing multiple services in the real world, collecting the specific configurations and parameters is impracticable, which limits the application of the previous two works. Therefore, we take the first step to propose a traffic generation model to simulate various IoT devices serving in multiple scenes based on large-scale realistic data.

[§]Corresponding Author.

IoT traffic generation can be regarded as a particular case for time series generation, which is influenced by complex background information of IoT devices, i.e., device category, manufacturer design, user habit, and application service. The background information facilitates the data fidelity once introduced in the generative model. In various background information, the device category is an inherent attribute for IoT devices and is able to provide important instructions for the function of devices and potent guidance for the traffic series generation without privacy issues. Therefore, generating both the IoT device category and traffic series for the synthetic dataset comes naturally. As GAN is capable of generating multiple forms of data cooperated with different manners of generators, Lin et al. [23] propose DoppelGANger and generate attributes for objects together with feature series, which reaches state-of-art results on several network traffic datasets. However, in the experiments for IoT traffic, the dataset generated by DoppelGANger omits rare device categories, and the model fails to simulate the heavy imbalance and sparsity of IoT traffic in the absence of background information.

Nevertheless, generating IoT traffic is challenging for the following reasons:

- Real-world IoT traffic is influenced by the complicated factors from users, environments, and applications, which brings challenges to acquire the core background information and feed it into the generative model while preserving privacy.
- The variable lengths of IoT traffic series, which are caused by the different communication time intervals of IoT devices to perform multiple functions, bring challenges to learning the temporal patterns. Our observations show that except for short traffic series, which are sparse in the time domain, there also exist long traffic series containing thousands of elements. As the generative model is requested to learn both long-term and short-term temporal patterns for the series, generating long series is especially challenging.
- The distributions of device categories and features in the traffic series are heavily unbalanced, which brings challenges to generating a realistic and diverse synthetic dataset.

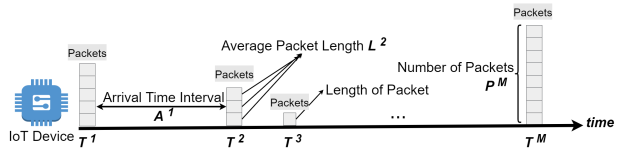
For IoT devices, the background information contains both semantic knowledge and interaction relations, inspiring us to adopt knowledge graphs naturally. Hence, we propose a knowledge-enhanced GAN for IoT traffic generation to address the above challenges. First, we construct a knowledge graph via IoT traffic data and other background information collected from the manufacturers, suppliers, and users. Then, we construct a GAN framework to generate IoT device category and traffic series simultaneously, consisting of a comprehensive generator and a simple discriminator. To introduce background knowledge into the framework and capture the influence of the device category on the traffic series, we adopt a condition mechanism. Finally, we evaluate our knowledge-enhanced GAN on a real-world IoT traffic dataset, and extensive experiments illustrate that our model outperforms five baselines and performs well on small datasets via introducing background knowledge into generation.

In conclusion, our contributions are summarized as follows:

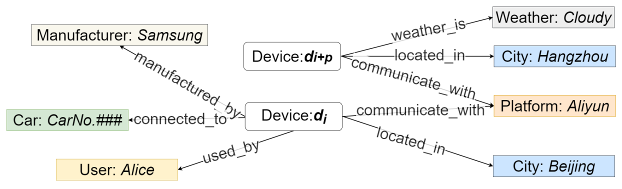
- We build a knowledge graph to describe the background information for IoT devices, learning both semantic knowledge and interaction features.
- We propose a knowledge-enhanced GAN for IoT traffic generation, which uses a condition mechanism to incorporate the knowledge and device category for IoT traffic generation, and adopts LSTM and self-attention mechanism to capture the long-term and short-term temporal correlation in the traffic series.

Notation	Definition
O_i	IoT traffic data for device d_i.
C_i	Category of device d_i.
T_i	Network traffic series for device d_i.
M_i	Length of the network traffic series T_i for device d_i.
A_i^j	Inter-arrival time between packets in $j-1_{th}$ and j_{th} sampling for device d_i.
P_i^j	Total number of packets in j_{th} sampling for device d_i.
L_i^j	Average packet length of the P_i^j packets in j_{th} sampling for device d_i.
K_i	Knowledge graph embedding (KGE) for device d_i.

Table 1: Notations and definitions for IoT traffic generation.



(a) Traffic Data.



(b) IoT Knowledge Graph.

Figure 1: Illustration of IoT traffic data and IoT knowledge graph : (a) Traffic data of IoT device, (b) IoT knowledge graph.

- We conduct experiments on a real-world IoT traffic dataset, our proposed model outperforms other state-of-art baselines on data fidelity and application. The model is also demonstrated to generate realistic data trained on small real datasets by introducing background knowledge into generation.

2 BACKGROUND AND PROBLEM

The IoT traffic dataset can be formally denoted as a set of objects $S = \{O_i\}_{i=1}^{N}$, where O_i represents the data of the i_{th} IoT device d_i. For each device, the data $O_i = (C_i, T_i)$, C_i represents the device category, and T_i represents the 3-dimension network traffic series. As presented in Figure 1(a), the traffic series $T_i = \{A_i^j, P_i^j, L_i^j\}_{j=1}^{M_i}$, where M_i is the lengths of traffic series for IoT device d_i, and Table 1 presents details of the three features: arrival time interval A_i^j, total number of packets P_i^j, and average packet length L_i^j. Given a real-world IoT traffic dataset S, our goal is to generate a realistic traffic dataset $\hat{S}$ with a generative model G.

Generative adversarial networks [14] is a state-of-art generative model based on adversarial learning, which achieves remarkable results in generation tasks of several fields [42, 46, 47]. A typical GAN has two components, a generator G_θ and a discriminator D_ϕ,

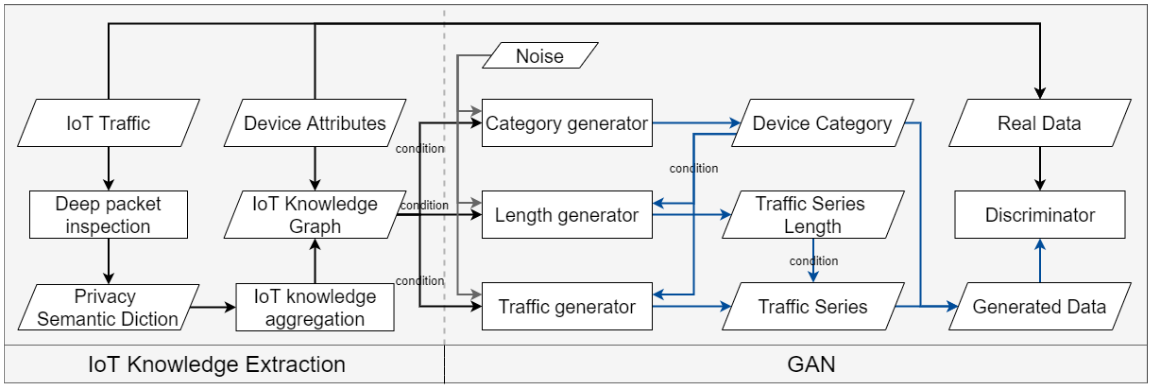


Figure 2: Framework for knowledge enhanced IoT traffic generation.

which are alternately trained to generate fake data in comparison with real data. The generator G_θ learns to fool the discriminator D_ϕ via generating fake data with similar distribution to real data, and the discriminator D_ϕ learns to distinguish between the fake and real samples, which performs a min-max competition as follows,

$$\min_{\theta} \max_{\phi} \mathbb{E}_{x\sim p_d} \left[\log \mathrm{D}_\phi \left(\mathrm{x}\right)\right] + \mathbb{E}_{x\sim G_\theta} \left[\log\left(1 - \mathrm{D}_\phi \left(\mathrm{x}\right)\right)\right], \quad (1)$$

where x is the samples, p_d is the distribution of real data, G_θ represents the generator parameterized by θ, and D_ϕ represents the discriminator parameterized by ϕ. Various structures for generator, discriminator and loss function are constructed for different generation tasks, and we design a particular framework for IoT traffic data generation.

3 METHODS

Generally, IoT traffic data is heavily unbalanced and sparse, which leads to the failure of existing generative models. Especially, GAN-based models easily suffer from mode collapse, which means that the generator provides limited sample variety despite being trained on diverse data. To generate realistic IoT traffic while avoiding mode collapse, as each IoT device's knowledge graph embedding (KGE) is unique, we introduce the background information of IoT devices via KGE to provide diverse conditions for the GAN model. Then, we adopt a condition mechanism to acquire the influence of knowledge and device category to the traffic series, and use LSTM and self-attention mechanism to capture the temporal correlation in series. The framework of our proposed model is presented in Figure 2 , lines in black, blue, and gray represent the transmissions of real data, generated data, and noise, respectively. First, we construct a knowledge graph from the basic information and network traffic of IoT devices, and extract the KGE information for each device. Then, we train a generator G and a discriminator D on the condition of KGE information. The generator G consists of three sub-generators: category generator G^C, series length generator G^M, and traffic series generator G^T, which are associated with each other via the condition mechanism.

3.1 Knowledge Graph Construction

The background information of IoT devices contains both semantic knowledge and network structure, which inspires us to adopt knowledge graphs naturally. To introduce the information to the generative model, we construct a knowledge graph via IoT traffic data and other background information. First, we collect basic information for each device from the descriptions of manufacturers, suppliers, and users, e.g., the type of hardware model. Then, we use an IoT privacy leakage quantification framework [19] to extract the user, platform, location, and environment information from the network traffic. The communications between IoT devices, IoT users, and cloud platforms can be detected via the source and destination IP addresses in network traffic packets, which contain the network structure information. As presented in Figure 1(b), IoT devices are considered as head entities in the knowledge graph, and the user, platform, location, and environment information are regarded as tail entities. The following triples give several examples,

- <device d_i, manufactured_by, manufacturer: *Samsung*>,
- <device d_i, located_in, city: *Beijing*>,
- <device d_i, communicate_with, platform: *Aliyun*>.

Finally, we acquire 39,598 entities (including 10187 devices) and 133,075 relations for the knowledge graph, and the relations fall into twenty categories. Then, we use the TransE model [5] to learn the embedding K_i of each IoT device d_i. To ensure the semantic information is preserved in the embeddings, we train serval classifiers to predict the device category C_i by KGE K_i, and the accuracy of most classifiers is around 90%, of which the details are shown in Appendix A.

3.2 Generator

In terms of IoT traffic generation, the generator is designed to satisfy three major requirements. 1) Introducing the IoT network structure and semantic knowledge to the generated data, 2) capturing the correlation between device category and traffic series, and 3) capturing both the long-term and short-term temporal patterns for traffic series. To satisfy these requirements and avoid mode collapse, we design a comprehensive generator that consists of three sub-generators, as presented in Figure 3. For the first requirement, these three sub-generators are associated with each other via a condition mechanism. For the second requirement, we combine the condition mechanism with device generator G^C and series length generator G^M. For the third requirement, G^T uses an LSTM with a self-attention mechanism to generate traffic series T_i based on C_i and M_i.

Condition Mechanism. We use a condition mechanism to introduce the IoT network structure and semantic knowledge to the generated data and capture the influence of device category on the traffic series. The conditions are presented in the following

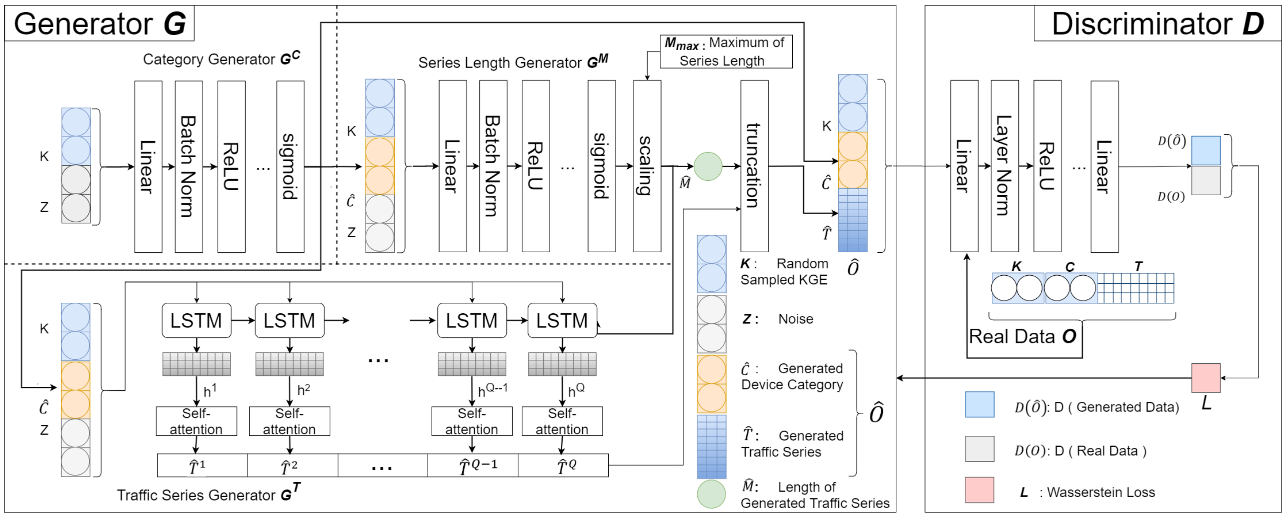

Generator G
Category Generator G^C
Series Length Generator G^M
M_{max} : Maximum of Series Length
Linear
Batch Norm
ReLU
sigmoid
scaling
truncation
LSTM
Self-attention
Traffic Series Generator G^T
K : Random Sampled KGE
Z : Noise
$\hat{C}$: Generated Device Category
$\hat{T}$: Generated Traffic Series
$\hat{M}$: Length of Generated Traffic Series
Discriminator D
Layer Norm
Real Data O
$D(\hat{O})$: D (Generated Data)
$D(O)$: D (Real Data)
L : Wasserstein Loss

Figure 3: Illustration of our proposed knowledge enhanced GAN for IoT traffic generation.

formulation,

$$\begin{aligned} P(\hat{O}_i, K_i) &= P(\hat{O}_i|K_i) \cdot P(K_i) \\ &= P(\hat{C}_i, \hat{T}_i, \hat{M}_i|K_i) \cdot P(K_i) \\ &= P(\hat{T}_i|\hat{C}_i, \hat{M}_i, K_i) \cdot P(\hat{M}_i|\hat{C}_i, K_i) \cdot P(\hat{C}_i|K_i) \cdot P(K_i). \end{aligned} \tag{2}$$

First, we randomly sample a device d_i, feed the corresponding KGE information K_i into each sub-generator as conditions, and $P(K_i)$ represents the distribution of K_i. Then, the device category $\hat{C}_i$ generated by G^C obeys the distribution of $P(\hat{C}_i|K_i)$, which is fed into G^M and G^T together with K_i and Gaussian random noise vector Z_i. Similarly, G^M generates $\hat{M}_i$ with the distribution of $P(\hat{M}_i|\hat{C}_i, K_i)$, and $\hat{M}_i$ is fed into G^T to control the length of generated traffic series. Finally, G^T generates $\hat{T}_i$ on the condition of $\hat{C}_i$, $\hat{M}_i$, and K_i. Hence, the generated samples $\hat{O}_i$ follows the distribution of $P(\hat{O}_i|K_i)$.

MLP Generators. We use multilayer perceptrons (MLP) with sigmoid activation function to generate the device category $\hat{C}_i$ and the length of traffic series $\hat{M}_i$. For category generator G^C, we use KGE information K_i and noise vector Z_i as input, then the generated device category $\hat{C}_i$ is a zero-one normalized vector with the same dimension as the total category number. For gradient calculation, the category generator G^C outputs continuous results, and the dimension with the maximum value indicates the final category. For series length generator G^M, the generated device category $\hat{C}_i$ and the KGE information K_i are both fed into it as condition vectors. The noise and condition vectors are mapped to a zero-one value after the linear layers and sigmoid activation function. Then, we scale the value to the length of traffic series $\hat{M}_i$ via the maximum threshold M_{max}, which can be calculated from the real dataset or manually configured.

LSTM Generator with Self-attention. To capture both the long-term and short-term temporal patterns of the traffic series, we adopt LSTM networks [17] to generate the traffic series $\hat{T}_i$. LSTM is an RNN architecture particularly appropriate to process series data. In a typical LSTM unit, each record in the series is mapped to the hidden internal state in the corresponding step and merged with the patterns of all the past records. Then, the j_{th} record is generated correlated with the previous $j-1$ records, and it takes M_i steps to generate a series with M_i length generally.

Although LSTM is known for memorizing history values over arbitrary intervals, modeling series with thousands of dimensions is challenging in efficiency and effectiveness. A common solution is to segment long series into several short series as independent samples. However, as IoT devices have traffic series with variable lengths, segmenting the traffic for some of the devices while maintaining others' integrity is unreasonable in generation tasks. Therefore, we generate B samples $\hat{T}_i^k$ in each step to improve the efficiency, where $\hat{T}_i^k = \{\hat{A}_i^j, \hat{P}_i^j, \hat{L}_i^j\}_{j=B(k-1)+1}^{Bk}$, $k = 1, 2, \dots, Q$, and $Q = \lceil \hat{M}_i/B \rceil$. It takes Q steps to generate a series with M_i length. Specifically, as presented in Figure 3, in the k_{th} step, the KGE information K_i, generated device category $\hat{C}_i$, and noise vector Z_i^k are fed into the LSTM unit, which outputs an embedding h_i^k for the samples $\hat{T}_i^k$.

Moreover, to capture the correlation inner each B samples of one step, we adopt a scaled dot-product self-attention mechanism [38].

$$\begin{aligned} &R_K = ReLU(h_i^k W_K), R_Q = ReLU(h_i^k W_Q), R_V = ReLU(h_i^k W_V), \\ &\hat{T}_i^k = softmax(\frac{R_Q \cdot R_K^T}{\sqrt{d}}) \cdot R_V \cdot W_s. \end{aligned} \tag{3}$$

In the self-attention layer, as presented in Equation 3, h_i^k is mapped to three representations (i.e., key R_K, query R_Q, and value R_V) via linear projections. Then, we compute the dot products of the query with all keys and compute the weights on the values via a softmax activate function after normalizing the products by the dimension of keys. Finally, the values are mapped to B samples $\hat{T}_i^k$ by the above weights. After Q steps, the generated samples $\{\hat{T}_i^k\}_{k=1}^{Q}$ are reshaped into $\hat{T}_i = \{\hat{A}_i^j, \hat{P}_i^j, \hat{L}_i^j\}_{j=1}^{\hat{M}_i}$. Note that the traffic series generator G^T has a single LSTM unit and self-attention layer. The expanded structure presented in Figure 3 stands for the Q steps in the generation.

3.3 Discriminator

We consider two major factors in the design of discriminator D. First, the training process of generator and discriminator are alternative and adversarial in GANs, which indicates that a significantly stronger generator or discriminator leads to failure. Particularly, the generating task is more complicated than the discriminating task for IoT traffic, which requests a more powerful generator than the discriminator for model design, and an MLP is adequate for a discriminator in IoT traffic generation task. Second, the device category C_i is one-hot encoding in the real dataset, which is discrete, but the generated device category $\hat{C}_i$ is continuous. Therefore, we use Wasserstein distance [4] with gradient penalty [16] to deal with the continuous and discrete data simultaneously, which requires calculating the second derivative for the loss function. As the calculation of a second derivative for the loss function in deep learning models is difficult in practice, MLP becomes a better choice. For the above two reasons, we adopt an MLP for discriminator D .

3.4 Loss Function and Training

As discussed in Section 3.3, the device category C_i is discrete in the real dataset, and the values in traffic series T_i are continuous. To deal with the continuous and discrete data simultaneously, we use Wasserstein distance [4] with gradient penalty [16] in our model, which is also demonstrated to be effective in performance improvement for GANs. The loss function is as follows,

$$L = \underset{\hat{O}\sim\mathbb{P}_g}{\mathbb{E}}[D(\hat{O})] - \underset{O\sim\mathbb{P}_r}{\mathbb{E}}[D(O)] + \lambda \underset{\tilde{O}\sim\mathbb{P}_{\tilde{O}}}{\mathbb{E}}\left[(\left\|\nabla_{\tilde{O}}D(\tilde{O})\right\|_2 - 1)^2\right], \tag{4}$$

where $D(O)$ is the discrimination results of real samples O, and $\mathbb{P}_r$ represents the real data distribution, $D(\hat{O})$ is the discrimination results of generated samples $\hat{O}$, and $\mathbb{P}_g$ represents the generator distribution, $\tilde{O}$ are samplings uniformly along straight lines between pairs of objects sampled from the real and generated data, $D(\tilde{O})$ is its discrimination results and $\mathbb{P}_{\tilde{O}}$ represents its distribution.

The generator is trained to minimize the loss, while the discriminator is trained to maximize it. We use mini-batch to improve the training efficiency, and the discriminator is trained several times before the generator is trained in each iteration. Details are shown in Appendix B.

4 EXPERIMENTS

To verify the proposed generative model, we acquire a real-world IoT traffic dataset, compare our model's performance with five baseline models in terms of data fidelity and application value, and discuss the knowledge enhancement effect via training models on small datasets.

4.1 Dataset

We obtain three-day IoT traffic from one of the largest mobile network operators in China and identify 10,187 devices connected to 37 different IoT platforms, such as logistics and vehicle management platforms. These devices work for multiple IoT applications and services, and we search various data to collect their basic information and extract the dataset. In addition to the network traffic, we refer to their Type Allocation Code allocated by the Global System for Mobile Communications Alliance (GSMA), their product descriptions on relevant websites, the official documents or user guides provided by the manufacturers, and the product descriptions or instructions given by distributors. Finally, the devices are classified into ten categories presented in Figure 4, which cover the commonly used functions of IoT devices. Similar to the device categories, the lengths of traffic series and the other three features are on heavily unbalanced distributions, which requires the generative models to learn the characteristics of "long tail" with limited train samples, details are shown in Appendix C.

4.2 Performance Comparison

We compare the performance of our proposed model with baseline models from two perspectives. First, we evaluate the fidelity of generated data via Jensen–Shannon divergence (JSD) between the distribution of generated data and real data. Then, we conduct a case study on the generated datasets to verify their effect in the application.

4.2.1 Baselines. We compare our proposed model with the following five baselines:

Auto-regressive (AR) [12]. Typical AR models generate the j_{th} record in series according to the previous $j-1$ records, which can only generate series with a fixed length. Hence, we denote $\bar{T}_i^j = (C_i, T_i^j, M_i) = (C_i, A_i^j, P_i^j, L_i^j, F_i^j)$, where $F_i^j = 1$ if $j < M_i$ else 0. Then, we train an AR model such that $\bar{T}_i^j = AR(\bar{T}_i^{j-1}, \bar{T}_i^{j-2}, \ldots, \bar{T}_i^{j-t})$. In terms of generation, for each generated object $\hat{O}_i$, we randomly sample an initial $\bar{T}_i^0 = (C_i, A_i^0, P_i^0, L_i^0, 1)$ from the real dataset for the trained AR model. Finally, we reshape the generated series $\{\bar{T}_i^j\}_{j=1}^{M_{max}}$ into $\hat{O}_i = \{\hat{T}_i^j\}_{j=1}^{M_i}$.

Hidden Markov models (HMM) [13]. Similar to AR, we use $\bar{T}_i^j$ conditioning on device categories with variable length to help HMM generate traffic series. Particularly, as the device category C_i is directly sampled from the real data, we leave out the category generation evaluation for AR and HMM.

Long Short Term Memory (LSTM) [17]. We train the LSTM generative model according to the series prediction manner. The true values in the IoT traffic series are fed into the LSTM unit at each step to predict the next values. In addition, we use $\bar{T}_i^j$ to introduce the device category and series length into the model similar to AR and HMM, and add several linear layers after all steps to generate the device category $\hat{C}_i$. Then, we use random noise as the initial input for the trained model, and traffic series are generated step by step.

Naive GAN [14]. In this model, we use MLPs as the generator and discriminator, in which the generation of traffic series $\hat{T}_i$ is on the condition of the device category $\hat{C}_i$ and length of traffic series $\hat{M}_i$, similar to our proposed model presented in Figure 3.

DoppelGANger [23]. DoppelGANger is a state-of-art GAN-based generative model that can generate variable-length feature series together with attributes of the series. For our generation task, the device category C_i is considered as attributes in this model, and the traffic series $\hat{T}_i = \{\hat{A}_i, \hat{P}_i, \hat{L}_i\}$ is the feature series.

Specifically, AR, HMM, and LSTM are representative series data processing models, Naive GAN is the typical GAN model, and DoppelGANger is a state-of-art GAN-based model for series data generation. As for other state-of-art GAN models for IoT traffic generation[29, 36], they mainly focus on a limited number of IoT devices in simplex applications, while we aim to generate large-scale IoT traffic in various applications. Thus, their performance is not appropriate to be compared in our experiments, of which the details are discussed in the related work. Overall, the performance of our model can be credibly evaluated via being compared with these selected models.

4.2.2 Fidelity. To evaluate the fidelity of generated data, we calculate Jensen–Shannon divergence (JSD) between the distribution

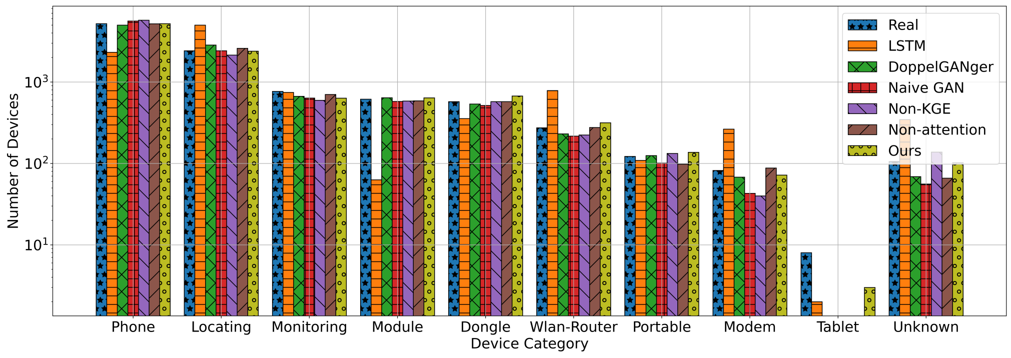


Figure 4: The number of devices of each category in datasets generated by different models.

JSD	Cate-gory	Series Length	Arrival Time Interval	Packet Number	Average Packet Length
AR	–	0.4218	0.6085	0.4667	0.5385
HMM	–	0.2217	0.2892	0.1694	0.5506
LSTM	0.3445	0.3196	0.5228	0.4950	0.6220
Naive GAN	0.0558	0.3130	0.5077	0.6588	0.5114
Doppel-GANger	0.0531	0.2562	0.6406	0.5153	0.6390
Non-KGE	0.0626	<u>0.2136</u>	0.1757	<u>0.1453</u>	0.2828
Non-attention	<u>0.0381</u>	0.2192	<u>0.1594</u>	0.2323	**0.0799**
Ours	**0.0352**	**0.1165**	**0.0898**	**0.1379**	<u>0.1404</u>

Table 2: JSD between real data and generated data from different generators, where lower results are better. Bold denotes the best(lowest) results and <u>underline</u> denotes the second-best results.

of generated data and real data, which is defined as,

$$\mathrm{JSD}(\mathbf{P}_g, \mathbf{P}_r)) = \sqrt{\frac{\mathbf{KL}(\mathbf{P}_g \| \mathbf{P}_{\tilde{O}}) + \mathbf{KL}(\mathbf{P}_{\tilde{O}} \| \mathbf{P}_r)}{2}}, \tag{5}$$

where $\mathbf{P}_r$ represents the real data distribution, $\mathbf{P}_g$ is the generator distribution, $\mathbf{P}_{\tilde{O}}$ represents the point-wise mean of $\mathbf{P}_r$ and $\mathbf{P}_g$, and **KL** is the Kullback-Leibler divergence. A lower JSD means a closer distribution to the real data, which indicates a better generative model. We include all the IoT traffic dataset features in fidelity evaluation, i.e., device category C, length of traffic series M, arrival time interval A, packet number P, and average packet length L. In addition to the baseline models, we conduct ablation experiments by removing the KGE information or self-attention mechanism in our proposed model, denoted by 'Non-KGE' and 'Non-attention'.

Table 2 presents the JSD between real data and generated data from different generative models, where our model outperforms the other models on four features and achieves a second-best result on the last feature. In comparison with baselines, JSD between real data and our generated data is conspicuously lower. For the device category, our JSD result is 34% lower than the best baseline, i.e., DoppelGANger. Specifically, Figure 4 presents the device number of each category in datasets generated by different models. As the device number in category *Tablet* is too small, only our model and LSTM-based model successfully generate the corresponding samples. However, the LSTM-based model has poor performance on other categories, with JSD far higher than all other models, which is a decuple of our JSD result. For the length of series, arrival time interval, number of packets, and average packet length, our JSD result is 47%, 69%, 19%, and 15% lower than the corresponding best baselines, i.e., HMM and DoppelGANger. Meanwhile, even with the KGE information or self-attention mechanism removed, our model outperforms the baseline models on all five features. Generally, the model without self-attention performs better than the model without KGE, which implies the importance of the knowledge and condition mechanism proposed in Section 3.2.

JSD	Series Length	Arrival Time Interval	Packet Number	Average Packet Length
AR	0.5376	0.4838	0.3382	0.5118
HMM	0.4142	0.3391	<u>0.3003</u>	0.5709
LSTM	0.4540	0.3179	0.3111	0.4473
Naive GAN	0.3406	0.5001	0.6609	0.5712
Doppel-GANger	0.3519	0.3786	0.4821	0.5309
Non-KGE	0.8278	0.3987	0.6020	0.5417
Non-attention	<u>0.3340</u>	<u>0.2216</u>	0.3722	<u>0.2217</u>
Ours	**0.2566**	**0.1563**	**0.1844**	**0.1945**

Table 3: Categorized JSD between real data and generated data from different generators, where lower results are better. Bold denotes the best(lowest) results and <u>underline</u> denotes the second-best results.

Furthermore, to evaluate the correlation between device category and other features, we calculate the JSD between real data and generated data for each category and define the mean value for all categories as categorized JSD, as presented in Table 3. Obviously, categorized JSD is larger than JSD for the difficulty of capturing the correlation between device category and other features. Our model outperforms the other models on all the features, which demonstrates the effectiveness of the condition mechanism. Meanwhile, the model without the attention mechanism is the second-best one, and the model without KGE has a poor performance, which implies that KGE information plays a crucial part in capturing the

Accuracy	SVM	kNN	LR	RF	MLP	NB
Real	**0.6621**	**0.6828**	**0.6774**	**0.8011**	**0.7549**	**0.5034**
AR	0.1805	0.4939	0.1687	0.5061	0.4399	0.1348
HMM	0.3942	0.4413	0.4201	0.3941	0.4179	0.4056
LSTM	0.2617	0.2455	0.2919	0.2352	0.1242	0.0447
Naive GAN	<u>0.5267</u>	0.4843	0.5304	0.3668	0.3670	0.2577
Doppel-GANger	0.4628	0.2627	0.4952	0.1011	0.2884	<u>0.4958</u>
Ours	0.4777	<u>0.5890</u>	<u>0.5349</u>	<u>0.5648</u>	<u>0.5275</u>	0.4134

Table 4: Results of device category classification via classifiers trained on different datasets, where higher results are better. Bold denotes the best(highest) results and <u>underline</u> denotes the second-best results.

correlation between device category and other features. In addition, the LSTM model seems to perform better on categorized JSD, , which indicates that an LSTM is more suitable for generating series on condition of given category distribution, like the traffic series generator G^T in our proposed model.

The above results demonstrate that our proposed model improves the fidelity for IoT traffic generation compared to baselines, and the knowledge and condition mechanism play crucial parts.

4.2.3 *Application.* As discussed in the introduction, IoT traffic classification is an important task in IoT network traffic study. Therefore, we conduct a traffic series classification task to evaluate the application effectiveness of our generative model, and compare the results with a number of baseline models. We train the classifier on the generated dataset, and apply the trained classifier to the real dataset. To control the impact of classification methods, the experiment includes six common-used classification algorithms: linear support vector machine (SVM), k-nearest neighbors (kNN), logistic regression (LR), random forest (RF), MLP, and multinomial naive Bayes (NB). In contrast, we split the real dataset into training and testing set accounting for 70% and 30%, train these classifiers on the training set and apply the trained model to the testing set. Table 4 presents the classification results, and classifiers trained on the real dataset outperform others naturally. The classifiers trained on the dataset generated by our proposed model perform second-best for most of the classification algorithms. Naive GAN performs better than our model for the SVM classifier, and DoppelGANger performs better for the NB classifier. However, SVM and NB have poor performances in comparison with other algorithms even trained on the real dataset. For experiment conveniences, these classifiers are based on oversimplified classification algorithms, and a well-designed classifier trained on the generated dataset could achieve better results.

These classification results illustrate that our model is capable of capturing the correlation between device category and traffic series, and the generated dataset is applicable in practice.

4.3 Knowledge Enhancement Effect

Performance comparison illustrates the effectiveness of our proposed model and implies the importance of KGE information. Therefore, we conduct a contrast experiment to study the effect of knowledge used in our model particularly. First, we randomly sample the real dataset using sampling rates that vary from 20% to 80%. Second, we train the models with and without KGE simultaneously on different sizes of data. Then, we calculate JSD between the generated datasets and the complete real dataset. The sampling, training, and generation are repeated several times to avoid bias introduced by sampling.

Figure 5 shows the JSD to real data of these datasets generated by models trained on different sizes of data. The results for models with and without KGE are presented as solid blue lines and yellow dashed lines varying according to the sampling rates of training sets, where the knowledge-enhanced models achieve better results. We find that the knowledge enhanced models perform well on small training sets with 20% or 40% sampling rates, which outperform or have comparative results to models on large training sets with 80% or 60% sampling rates. The training set with a 20% sampling rate only contains two thousand devices, which indicates that introducing background knowledge helps to generate high-fidelity data with limited samples.

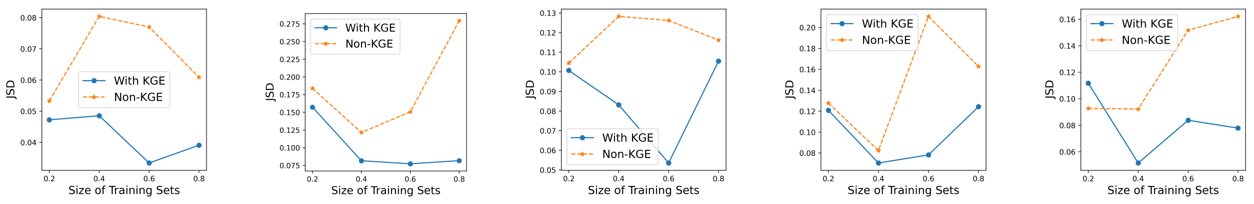
In conclusion, the above experiments illustrate that our proposed model is capable of capturing both the distribution characteristics and the correlation between device category and traffic series, which helps to generate a realistic IoT traffic dataset. Our proposed model not only outperforms a number of generative models but also performs well on small datasets via introducing background knowledge into generation, which helps to provide abundant synthetic data based on limited real data.

5 RELATED WORK

5.1 IoT Traffic Analysis and Modeling

Since IoT services have become ubiquitous in every walk of life gradually, IoT devices bring colossal influence to our society through the Internet, and their network traffic behaviors are gaining increasing popularity and importance among researchers. Interested in the new requirements and challenges of the machine to machine traffic, Shafiq et al. [35] firstly study the characteristics of large scale M2M traffic, such as temporal patterns, applications, and mobility. Then, IoT device behaviors focused on certain application scenes like vehicle [3] and wearable [21] are also analyzed. Later, motivated by better network design, application scenario validation, or mitigating security threats, multiple frameworks are proposed for IoT traffic modeling, predicting, or load estimation [1, 22, 31, 40]. In addition to behavior analysis and modeling, IoT traffic are applied to identifying the security issues [2, 10, 26, 30, 34], web services or applications [24], and device types in IoT [27, 28, 32, 37].

These analyzing, modeling, predicting, and identifying tasks require large-scale realistic datasets on various kinds of IoT devices, but most of their experiments are conducted based on single application scenes or even testbeds in laboratories with several devices, which spirits us to fill in the gaps of IoT data generation. Recently, Nguyen-An et al. [29] propose IoTTGen to generate synthetic traffic for smart home and bio-medical IoT environments, which succeed in capturing the entropy of traffic characteristics for several smart home devices (i.e., smart hub, light, camera, plug) and bio-medical sensors (i.e., body temperature, blood pressure, heart rate, respiratory rate, electromyography, cardiography). Unfortunately, their model requests configuration for each IoT device before traffic generation, in which the packet size, port number, payload, and arrival time interval are given as fixed parameters. To learn the temporal patterns in IoT traffic, Shahid et al. [36] propose to combine an auto-encoder with a GAN to generate sequences of packet sizes that correspond to bidirectional flows. The generated sequences of packet sizes behave closely to real bidirectional flows produced by



(a) Device Category. (b) Series Length. (c) Arrival Time Interval. (d) Packet Number. (e) Average Packet Length.

Figure 5: JSD between real data and dataset generated by models trained on different size of data, where blue dots represent the results for knowledge enhanced model, yellow stars represent the results for model without KGE, and lower results are better: (a) JSD of the device category, (b) JSD of the length of traffic series, (c) JSD of the arrival time interval. (d) JSD of the total number of packets, (e) JSD of the average packet length.

a Google Home Mini, which was actively used for a week. However, the IoT traffic data generated by the above two models is limited to simplex application scenes. Their experiments are conducted on a limited number of IoT devices in laboratories with detailed parameters, where the devices can be easily configured based on the prior knowledge. In our experiment, which is closer to the real situations, collecting the detailed parameters of the 10,187 real-world IoT devices in various applications and configuring all the devices is not feasible. Therefore, we do not compare the above models with our proposed model in the experiments.

5.2 Network Traffic Generation

Network traffic generation is a long-standing problem, which is traditionally applied to test network equipment, network services, and security products [48]. In the early stage, network traffic can be generated based on network traffic models (e.g., Poisson model), traffic characteristics (e.g., packet size and packet arrival distributions [39]), and network protocols (e.g., TCP [43]), which are appropriate for network performance tests but incapable of generating a high fidelity synthetic IoT traffic dataset for their limited representational ability.

Recently, various machine learning methods are applied to generate network traffic data, for instance, auto-regressive based models [6, 45]. As GAN [14] achieve remarkable effects in a number of generation tasks, including videos [42], audio [47], images [46], GAN-based models become popular for network traffic generation [8]. Ring et al. [33] design three different pre-processing approaches based on GANs to transform flow-based data into continuous values and generate flow-based network traffic trained on the CIDDS-001 dataset. The dataset is created in a virtual environment using OpenStack, providing abundant parameters, including IP addresses, port numbers, TCP flags, etc. However, the required parameters are hardly accessible in realistic environments, which restricts the application of their method. Dowoo et al. [11] propose PcapGAN to generate pcap files trained on cyberattack data and normal data for intrusion detection. Except for generating traffic series or attributes respectively, Lin et al. [23] propose DoppelGANger to generate data attributes and feature series simultaneously and reach state-of-art results on several traffic forms datasets, including web traffic time series, geographically-distributed broadband measurements, and compute cluster usage measurements.

These generative models for network traffic data provide us with experience to generate IoT traffic. Based on these models, our GAN framework generates realistic and diverse IoT traffic data enhanced by network structure and semantic knowledge.

6 DISCUSSION AND CONCLUSIONS

In this paper, we propose a knowledge-enhanced GAN for IoT traffic generation. First, we build a knowledge graph to describe the background information for IoT devices, which contains both semantic knowledge and network structure feature. Then, our GAN framework incorporates the knowledge and device category for IoT traffic generation via condition mechanism; it captures the long-term and short-term temporal correlation in the traffic series by LSTM and self-attention mechanism. Based on a large-scale real-world IoT traffic dataset, we generate a synthetic IoT traffic dataset via our proposed model, which outperforms other state-of-art baselines on data fidelity and application. The model is also demonstrated to generate high fidelity data trained on small real datasets, which implies that introducing background knowledge to generation helps GANs provide abundant synthetic datasets based on limited real data. Our proposed model can easily extend to similar datasets with both graph structure background and series behaviors, for instance, sensor signals and social network behaviors. Furthermore, we intend to enrich the knowledge graph and explore more methods to introduce knowledge into generation models.

Moreover, our model can be reused to generate all kinds of traffic data with variable lengths and attributes, e.g., mobile users' traffic or website visiting series. Other embeddings can replace the KGEs of IoT devices to enhance the generation, and our model still outperforms other models without KGE. Therefore, our model is reusable in different traffic generation tasks regardless of the existence of the corresponding knowledge graphs. We have released our code of the IoT traffic generation model on Github[1]. In addition, we also provide the trained models and generated IoT traffic. We believe this work promotes further studies of the traffic generation problem. One limitation of our proposed model is that we do not directly consider the interaction between IoT devices because we find few flows between devices in the real IoT dataset in our experiments. Specifically, the indirect relations between devices are learned via the IoT knowledge graph. As interactions between devices may be more frequent in the next generation of IoT, our model is able to easily adapt itself by supplementing the traffic between devices to the training dataset and adding the relations between devices to the IoT knowledge graph.

ACKNOWLEDGMENTS

This work was supported in part by The National Key Research and Development Program of China under grant 2020AAA0106000, the National Natural Science Foundation of China under U1936217, 61971267, 61972223, 61941117, 62171260, 61861136003, and the China Postdoctoral Science Foundation under 2021M691830

[1]https://github.com/shirdy/TrafficGeneration/tree/master/IoT

REFERENCES

[1] Ali R Abdellah and Andrey Koucheryavy. 2020. Deep Learning with Long Short-Term Memory for IoT Traffic Prediction. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*. Springer, 267–280.

[2] Yaser Al Mtawa, Harsimranjit Singh, Anwar Haque, and Ahmed Refaey. [n.d.]. Smart Home Networks: Security Perspective and ML-based DDoS Detection. In *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE, 1–8.

[3] Carlos E Andrade, Simon D Byers, Vijay Gopalakrishnan, Emir Halepovic, David J Poole, Lien K Tran, and Christopher T Volinsky. 2017. Connected cars in cellular network: a measurement study. In *Proceedings of the 2017 Internet Measurement Conference*. ACM, 235–241.

[4] Martin Arjovsky and Léon Bottou. 2017. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862* (2017).

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*. 1–9.

[6] Alisson Assis Cardoso and Flávio Henrique Teles Vieira. 2019. Generation of Synthetic Network Traffic Series Using a Transformed Autoregressive Model Based Adaptive Algorithm. *IEEE Latin America Transactions* 17, 08 (2019), 1268–1275.

[7] Batyr Charyyev and Mehmet Hadi Gunes. 2020. IoT Traffic Flow Identification using Locality Sensitive Hashes. In *2020 IEEE International Conference on Communications (ICC)*.

[8] Adriel Cheng. 2019. Pac-gan: Packet generation of network traffic using generative adversarial networks. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 0728–0734.

[9] Louis Columbus. 2017. Roundup of internet of things forecasts. *Forbes, Dec* 10 (2017), 2017.

[10] Shuaike Dong, Zhou Li, Di Tang, Jiongyi Chen, Menghan Sun, and Kehuan Zhang. 2020. Your Smart Home Can't Keep a Secret: Towards Automated Fingerprinting of IoT Traffic. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*. 47–59.

[11] Baik Dowoo, Yujin Jung, and Changhee Choi. 2019. PcapGAN: Packet Capture File Generator by Style-Based Generative Adversarial Networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, 1149–1154.

[12] Behrouz Farhang-Boroujeny. 2013. *Adaptive filters: theory and applications*. John Wiley & Sons.

[13] Shai Fine, Yoram Singer, and Naftali Tishby. 1998. The hierarchical hidden Markov model: Analysis and applications. *Machine learning* 32, 1 (1998), 41–62.

[14] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661* (2014).

[15] Dominique Guinard, Vlad Trifa, Friedemann Mattern, and Erik Wilde. 2011. From the internet of things to the web of things: Resource-oriented architecture and best practices. In *Architecting the Internet of things*. 97–129.

[16] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028* (2017).

[17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[18] Shuodi Hui, Huandong Wang, Dianlei Xu, Jing Wu, Yong Li, and Depeng Jin. 2021. Distinguishing Between Smartphones and IoT Devices via Network Traffic. *IEEE Internet of Things Journal* (2021).

[19] Shuodi Hui, Zhenhua Wang, Xueshi Hou, Xiao Wang, Huandong Wang, Yong Li, and Depeng Jin. 2020. Systematically Quantifying IoT Privacy Leakage in Mobile Networks. *IEEE Internet of Things Journal* (2020).

[20] Shouling Ji, Weiqing Li, Mudhakar Srivatsa, and Raheem Beyah. 2014. Structural data de-anonymization: Quantification, practice, and implications. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1040–1053.

[21] Harini Kolamunna, Ilias Leontiadis, Diego Perino, Suranga Seneviratne, Kanchana Thilakarathna, and Aruna Seneviratne. 2018. A First Look at SIM-Enabled Wearables in the Wild. In *Proceedings of the Internet Measurement Conference 2018*. ACM, 77–83.

[22] M Laner, N Nikaein, P Svoboda, M Popovic, D Drajic, and S Krco. 2015. Traffic models for machine-to-machine (M2M) communications: types and applications. In *Machine-to-machine (M2M) Communications*. Elsevier, 133–154.

[23] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. 2020. Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions. *Proceedings of the ACM Internet Measurement Conference*, 464–483.

[24] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. 2017. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access* 5 (2017), 18042–18050.

[25] Zheng Ma, Ming Xiao, Yue Xiao, Zhibo Pang, H Vincent Poor, and Branka Vucetic. 2019. High-reliability and low-latency wireless communication for internet of things: challenges, fundamentals, and enabling technologies. *IEEE Internet of Things Journal* 6, 5 (2019), 7946–7970.

[26] Eman Maali, David Boyle, and Hamed Haddadi. 2020. Towards identifying IoT traffic anomalies on the home gateway. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 735–736.

[27] Samuel Marchal, Markus Miettinen, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and N Asokan. 2019. Audi: Toward autonomous iot device-type identification using periodic communication. *IEEE Journal on Selected Areas in Communications* 37, 6 (2019), 1402–1412.

[28] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. 2017. IoT Sentinel: Automated device-type identification for security enforcement in IoT. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2177–2184.

[29] Hung Nguyen-An, Thomas Silverston, Taku Yamazaki, and Takumi Miyoshi. 2020. Generating iot traffic: A case study on anomaly detection. In *2020 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN*. IEEE, 1–6.

[30] Laisen Nie, Zhaolong Ning, Xiaojie Wang, Xiping Hu, Yongkang Li, and Jun Cheng. 2020. Data-Driven Intrusion Detection for Intelligent Internet of Vehicles: A Deep Convolutional Neural Network-based Method. *IEEE Transactions on Network Science and Engineering* (2020).

[31] Navid Nikaein, Markus Laner, Kaijie Zhou, Philipp Svoboda, Dejan Drajic, Milica Popovic, and Srdjan Krco. 2013. Simple traffic modeling framework for machine type communication. In *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*. VDE, 1–5.

[32] Antônio J Pinheiro, Jeandro de M Bezerra, Caio AP Burgardt, and Divanilson R Campelo. 2019. Identifying IoT devices and events based on packet length from encrypted traffic. *Computer Communications* 144 (2019), 8–17.

[33] Markus Ring, Daniel Schlör, Dieter Landes, and Andreas Hotho. 2019. Flow-based network traffic generation using generative adversarial networks. *Computers & Security* 82 (2019), 156–172.

[34] Muhammad Shafiq, Zhihong Tian, Ali Kashif Bashir, Xiaojiang Du, and Mohsen Guizani. 2020. IoT Malicious Traffic Identification Using Wrapper-Based Feature Selection Mechanisms. *Computers & Security* (2020), 101863.

[35] Muhammad Zubair Shafiq, Lusheng Ji, Alex X Liu, Jeffrey Pang, and Jia Wang. 2012. A first look at cellular machine-to-machine traffic: large scale measurement and characterization. *ACM SIGMETRICS performance evaluation review* 40, 1 (2012), 65–76.

[36] Mustafizur R Shahid, Gregory Blanc, Houda Jmila, Zonghua Zhang, and Hervé Debar. 2020. Generative Deep Learning for Internet of Things Network Traffic Generation. In *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 70–79.

[37] Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. 2018. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. *IEEE Transactions on Mobile Computing* (2018).

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[39] Kashi Venkatesh Vishwanath and Amin Vahdat. 2009. Swing: Realistic and responsive network traffic generation. *IEEE/ACM Transactions on Networking* 17, 3 (2009), 712–725.

[40] Yinxin Wan, Kuai Xu, Feng Wang, and Guoliang Xue. 2020. Characterizing and Mining Traffic Patterns of IoT Devices in Edge Networks. *IEEE Transactions on Network Science and Engineering* (2020).

[41] Huandong Wang, Chen Gao, Yong Li, Gang Wang, Depeng Jin, and Jingbo Sun. 2018. De-anonymization of mobility trajectories: Dissecting the gaps between theory and practice. In *The 25th Annual Network & Distributed System Security Symposium (NDSS'18)*.

[42] Yaohui Wang, Piotr Bilinski, Francois Bremond, and Antitza Dantcheva. 2020. ImaGINator: Conditional spatio-temporal GAN for video generation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 1160–1169.

[43] Michele C Weigle, Prashanth Adurthi, Félix Hernández-Campos, Kevin Jeffay, and F Donelson Smith. 2006. Tmix: a tool for generating realistic TCP application workloads in ns-2. *ACM SIGCOMM Computer Communication Review* 36, 3 (2006), 65–76.

[44] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. 2017. Trajectory recovery from ash: User privacy is not preserved in aggregated mobility data. In *Proceedings of the 26th international conference on world wide web*. 1241–1250.

[45] Shengzhe Xu, Manish Marwah, and Naren Ramakrishnan. 2020. STAN: Synthetic Network Traffic Generation using Autoregressive Neural Models. *arXiv preprint arXiv:2009.12740* (2020).

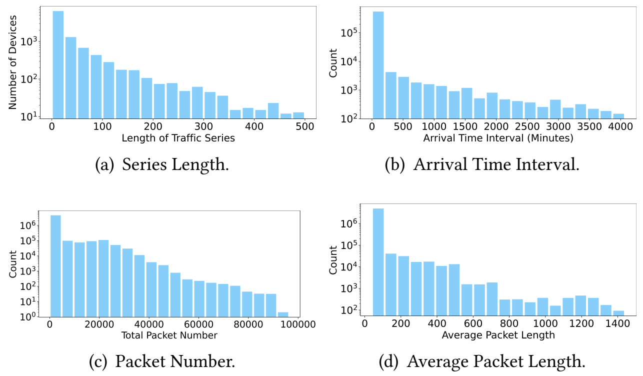
[46] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1316–1324.

[47] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. 2017. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847* (2017).

[48] Junhui Zhang, Jiqiang Tang, Xu Zhang, Wen Ouyang, and Dongbin Wang. 2015. A survey of network traffic generation. (2015).

Algorithm 1 Knowledge-enhanced GAN. Default values : λ = 10, n_D=5, q = 64, α=0.0001, β_1=0.5, β_2=0.9.

```
Require: The gradient penalty coefficient $\lambda$, the number of discriminator iterations per generator iteration $n_D$, the batch size $q$, Adam hyperparameters $\alpha$, $\beta_1$, $\beta_2$.
Initialize: initial discriminator parameters $\phi_0$, initial parameters $\theta_0$.
1:  while $\theta$ has not converged do
2:      for $t$=1, ... ,$n_D$ do
3:          for $i$=1, ... ,$q$ do
4:              $D_\phi$ Training:
                Sample real data $O_i \sim P_r$,
                      KGE $K_i \sim P(K)$,
                      noise $Z_i \sim P(Z)$,
                      random number $\epsilon \sim U[0,1]$
5:              $\widehat{C}_i \leftarrow G^C_\theta\ (Z^C_i|K_i)$
6:              $\widehat{M}_i \leftarrow G^M_\theta (Z^M_i|\widehat{C}_i,\mathrm{K}_i)$
7:              $\widehat{T}_i \leftarrow G^T_\theta (Z^T_i|\widehat{M}_i,\widehat{C}_i,\mathrm{K}_i)$
8:              $\widehat{O}_i \leftarrow (\widehat{C}_i,\widehat{T}_i)$
9:              $\widetilde{O}_i \leftarrow \epsilon O_i$+(1-$\epsilon$)$\widehat{O}_i$
10:             $L_i \leftarrow D_\phi\ (\widetilde{\widehat{O}})$-$D_\phi(O_i)$+$\lambda$ ($\|\ \nabla_{\widetilde{O}_i}\ D_\phi\ (\widetilde{O}_i)\ \|_2$-1)$^2$
11:         end for
12:         $\phi \leftarrow$ Adam($\nabla_\phi \frac{1}{q}\ \sum_{i=1}^{q} L_i$, $\epsilon$,$\alpha$,$\beta_1$,$\beta_2$)
13:     end for
14:     $G_\theta$ Training:
        Sample a batch of noise$\{Z_s\}_{s=1}^{q} \sim P(Z)$,
                        KGE $\{K_s\}_{s=1}^{q} \sim P(K)$
15:     $\widehat{C}_s \leftarrow G^C_\theta\ (Z^C|K_s)$
16:     $\widehat{M}_s \leftarrow G^M_\theta(Z^M|\widehat{C}_s,\mathrm{K}_s)$
17:     $\widehat{T}_s \leftarrow G^T_\theta(z^T|\widehat{M}_s,\widehat{C}_s,\mathrm{k}_s)$
18:     $\widehat{O}_s \leftarrow (\widehat{C}_s,\widehat{T}_s)$
19:     $\theta \leftarrow$ Adam($\nabla_\phi \frac{1}{q}\sum_{s=1}^{q}$-$D_\phi(\widehat{O}_s)$,$\theta$,$\alpha$,$\beta_1$,$\beta_2$)
20: end while
```



(a) Series Length. (b) Arrival Time Interval. (c) Packet Number. (d) Average Packet Length.

Figure 6: The distributions of features in IoT traffic dataset: (a) The length of traffic series, (b) arrival time interval, (c) total number of packets. (d) average length of packets.

Model	SVM	kNN	LR	RF	MLP	NB
Accuracy	0.8884	0.9167	0.8311	0.9114	0.9286	0.6856

Table 5: Results of device category prediction via knowledge graph embedding.

A DEVICE CATEGORY PREDICTION BASED ON KGE

We split the real dataset into training and testing set accounting for 50% and 50%, train the classifiers on the training set, and apply the trained model to the testing set. To control the impact of classification methods, the experiment includes six common-used classification algorithms: linear support vector machine (SVM), k-nearest neighbors (kNN), logistic regression (LR), random forest (RF), MLP, and multinomial naive Bayes (NB). Table 5 presents the classification results, which indicates the semantic information is preserved in the knowledge graph embeddings.

B MODEL TRAINING

The training process of our knowledge enhanced GAN is presented in Algorithm 1, G^C_θ, G^M_θ, G^T_θ denote the θ-parameterized category generator, series length generator, and traffic series generator, D_ϕ denote the ϕ-parameterized discriminator. We adopt a Wasserstein loss with gradient penalty [16], the generators G^C_θ, G^M_θ, G^T_θ are trained to minimize the loss, while the discriminator D_ϕ is trained to maximize it. We use mini-batches of size q, and adopt an Adam optimizer. In each iteration, the discriminator D_ϕ is trained n_D times before the generator G_θ.

To train the discriminator D_ϕ, we sample real data O_i, knowledge graph embedding K_i, noise Z_i, and a random number ϵ_i at first. Then, device category $\hat{C}_i$ is generated by G^C_θ on the condition of K_i, series length $\hat{M}_i$ is generated by G^M_θ on the condition of K_i and $\hat{C}_i$, and traffic series $\hat{T}_i$ is generated by G^T_θ on the condition of K_i, $\hat{C}_i$, and $\hat{M}_i$. The generated sample $\hat{O}_i$ consists of $\hat{C}_i$ and $\hat{T}_i$. We use ϵ_i to sample $\tilde{O}_i$ uniformly along straight lines between the real sample O_i and the generated sample $\hat{O}_i$. The samples O_i, $\hat{O}_i$, and $\tilde{O}_i$ are fed into the discriminator D_ϕ to compute the loss, and update the parameters ϕ for discriminator D_ϕ to maximize the loss. After n_D iterations for discriminator D_ϕ, we generate samples $\hat{O}_s$ from G^C_θ, G^M_θ, and G^T_θ, feed $\hat{O}_s$ into D_ϕ to compute the loss and update parameters θ for generators by minimize it. The training ends after the convergence of generator parameters θ or configured iterations.

C FEATURE DISTRIBUTION IN REAL DATASET

Figure 6(a) presents the lengths of traffic series for these devices, which vary between 10 and 500, and the distributions of arrival time interval, number of packets, and average packet length are also presented in Figure 6. The heavily unbalanced distributions require the generative models to learn the characteristics of "long tail" with limited training data.