Machine Learning-empowered Network Measurement: A Critical Path to Traffic Anomaly Detection in 6G-enabled Massive IIoT

Junxiao Wang, Technique report

***Abstract*—With the emergence of sixth-generation (6G) networks, the degree of interconnection and the amount of processed data is exploding. Massive Industrial-Internet-of-Things (IIoT) is inevitably increasing the stability requirement in network transmission and the necessity of adequate network measurement to detect traffic anomalies. There are two challenges for measurement in 6G-enabled networks. The first challenge is the high processing speed and limited memory space. The second challenge is the varying network traffic. This paper studies how to use sketch techniques to fulfill the required capabilities of 6G network measurement. Sketches have been considered as the most promising solution for network measurement in recent years, because they greatly optimize the speed and memory usage at the cost of small error. However, most sketches do not work well for varying network traffic. These sketches require to adjust their internal memory usage while the optimal point is sensitive to specified flow size distribution and memory size. To address this problem, we propose a machine learning empowered sketch framework. The proposed sketch framework trains a neural network model that online adjusts the memory usage based on a small sample of flow. We conduct data-driven simulations to evaluate the proposed framework. By comparing measurement results on public dataset, we can see that the proposed framework significantly improves the accuracy of measurement and anomaly detection without sacrificing the line-rate processing capability.**

***Index Terms*—6G; IIoT; Traffic Anomaly Detection; Network Measurement; Sketch; Machine Learning;**

I. Introduction

The sixth generation (6G) communication networks are envisioned to revolutionize customer services and applications via the Internet of Things (IoT) towards a future of fully intelligent and autonomous systems [1]. Higher automation in the industrial manufacturing process can greatly improve the efficiency and customization of the production line [2]. Furthermore, vast array of sensors are being embedded in cities, homes, factories and even humans. Data fusion and analysis are realized through ad hoc cloud and edge computing centers [3]. All above has resulted in an explosive growth in the degree of interconnectivity and the amount of data being processed. Massive Industrial-Internet-of-Things (IIoT) is inevitably increasing the stability requirements in network transmission and the necessity of adequate network measurement to detect traffic anomalies [4].

There are two challenges for measurement in 6G-enabled networks. The first challenge is the high processing speed and limited memory space. For high-speed 6G traffic, any practical measurement solution should process each incoming packet at line-rate while ensuring high measurement accuracy. It is desirable to implement such functionalities in on-chip memory like caches on network processors or BRAM in FPGA [5]. However, on-chip memory has a very limited size, imposing a great challenge for the design of compact data structures that can accurately record information for all flows. In this regard, sketches have been considered as the most promising solution for network measurement in recent years, because they greatly optimize the speed and memory usage at the cost of small error. In other words, most sketches address the challenge of high processing speed and limited memory space.

The second challenge is the varying 6G network traffic. It is well known that the flow size distribution tends to be highly skewed, where most flows are small and a few flows are extremely large [6]. State-of-the-art sketches typically have a separated structure with at least two parts, which record large flows and small flows, respectively. The first part is a key-value table used to precisely record the ID and size of large flows. The second part is commonly a hash table used to record the size of small flows in approximate. However, most of these sketches do not work well for varying network traffic. These sketches require to adjust their memory usage among multiple parts while the optimal point is sensitive to specified flow size distribution and memory size.

To address this problem, we propose a machine learning empowered sketch framework. The proposed sketch framework trains a neural network model that online adjusts the memory usage based on a small sample of flow. We conduct data-driven simulations to evaluate the proposed framework. By comparing measurement results on public dataset, we can see that the proposed framework significantly improves the accuracy of measurement and anomaly detection without sacrificing the line-rate processing capability. We envision this framework as a critical path to traffic anomaly detection in 6G-enabled massive IIoT.

The remainder of this paper is structured as follows. We first reviewed the background of the 6G-enabled IIoT, network measurement and their combination. Then, we presented the superiority of using machine learning methods in network measurement including the motivation, the proposed framework, corresponding implementation and deployment. Finally, we verified the proposed method by experiments.

II. Background

A. 6G-enabled IIoT

Recent advances in sixth generation (6G) communication networks and smart device technologies have promoted the proliferation of Internet of Things (IoT) with ubiquitous sensing and computing capabilities to interconnect millions of physical objects to the Internet. 6G and IoT enable seamless communications and automatic management between heterogeneous devices, which has the potential to revolutionize industries and provide significant benefits to society through fully intelligent and automated remote management systems. The network and connectivity techniques used in the lower layers include some 5G advances like NB-IoT, LoRaWAN, Time-sensitive communication (TSC), SDN/NFV, and so on.

NB-IoT leads the IoT connectivity solution from cellular networks for low-cost low-power IoT devices. Communications in NB-IoT works in a very narrow bandwidth, i.e. 200 KHz. This brings more than 20 dB extra link budget compared to the standard LTE systems. Thanks to this extra link budget, NB-IoT enables smart devices deployed in remote areas to directly communicate with the base stations. LoRaWAN, as a dominant IoT solution over the unlicensed spectrum. LoRaWAN leverages LoRa communications in its lower layers of protocol stack. LoRa (Long Range) is a proprietary low-power wide-area network modulation technique. In the solution, data is collected by local access points, which are the points of access to the Internet infrastructure for IoT devices. The IoT end-user accesses the gathered data by devices through the IoT server, and potentially sends commands back to them. Time-sensitive communications (TSC) integrates the 5G network and time-sensitive network (TSN), with the 5G network acting as a TSN bridge. This solution will evolve to 6G to provide native TSN, including over wider areas with mobility. Moreover, software-defined networking (SDN) and network function virtualization (NFV) are also widely recognized as key enabling technologies for 5G and 6G communication networks. SDN seeks to separate the control plane from the data plane and introduce novel network control functionalities based on a global view of the network, while NFV decouples specific network functionality from specialized hardware and replaces with software implementations on whitebox hardware.

Enabled by the fundamental technologies as described in above, 6G is envisioned to realize new applications for IoT. In this paper, we explore and discuss the emerging applications of 6G in an important IoT domain, that is IIoT. In the era of Industry 4.0, Industrial-Internet-of-Things (IIoT) draws more and more attentions from the academia and industry, which offers promising solutions for smart factory, data-driven manufacturing system, intelligent transportation system, and so on. Although 6G brings unprecedented benefits to IIoT, security and privacy represent significant challenges to be addressed [7]. With the rapid development of massive IIoT, more and more sensors and smart devices are connected while a large number of industrial APPs are arising. These devices and APPs produce a lot of real-time flows which inevitably increases the stability requirement in network transmission and the necessity of adequate network measurement to detect traffic anomalies.

B. Network Measurement

With the rapid development of the 5G networks and emergence of new networking paradigms like SDN, NFV and TSC (time sensitive communications), traffic measurement becomes increasingly important to support novel network functions and detect anomalies. One of the most fundamental measurements is flow size measurement. The most promising solution to flow size measurement is sketches. State-of-the-art sketches can be divided into two categories based on whether the skewed flow size distribution is considered or not. The first category is sketches not considering the skewed flow size distribution, including sketches of CM, CU, Count. These sketches have simply designed structures with low accuracy. The CM sketch is the most commonly used sketch in network measurement. The CM sketch never has underestimated error, i.e., the reported value is always no less than the actual size of the queried flow. The CU sketch makes a slight change to the CM sketch, called the conservative update. While the CU sketch optimizes the CM sketch for higher accuracy, it has a disadvantage in not supporting deletion. The CU sketch still never has underestimated error. The Count sketch is updated with an equal probability of +1 or -1, thereby it can achieve theoretically unbiased estimation of flow sizes.

It is noted that the flow size distribution tends to be highly skewed, where most flows are small and a few flows are extremely large. The sketches of the first category ignore the skewness of the flow size distribution and simplify structure at the cost of a considerable amount of space waste. The another category is sketches considering skewed flow size distribution, including sketches of Augmented, Elastic, HBL.

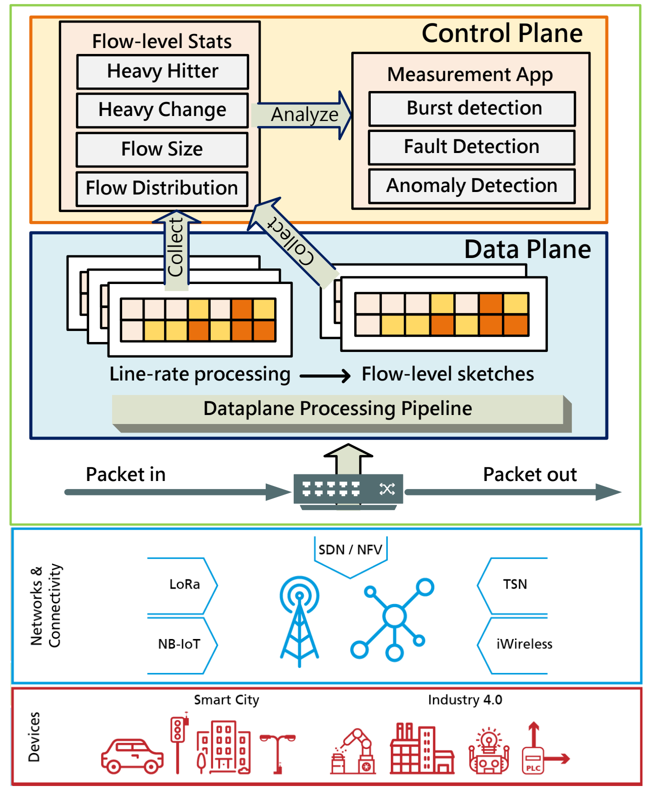

Fig. 1. The 6G network measurement framework is built on the infrastructure of 6G-enabled massive Industrial-Internet-of-Things (IIoT). The framework provides with many statistical measurement results to support advanced tasks like traffic anomaly detection.

These sketches leverage more complex structures with higher accuracy than the sketches of the first category. The Augmented sketch adds an additional filter to the sketch, aiming to record the large flows separately. Elastic sketch also uses a separated structure and it uses the major voting strategy to separate large flows from small flows. HBL sketch adds two additional filter to the sketch, in order to better separate large flows and small flows.

A combination of sketches and SDN/NFV vastly simplifies deployment and operation of network measurement, and enables independent evolution of both hardware and software, allowing for flexible adoption of emerging technologies in both domains. SDN and NFV are necessary for establishing networks that are easy to deploy, manage, and operate. In this paper, we discuss the architectural foundations for SDN and NFV based network measurement systems, which are expected to be among the key enabling technologies for 6G.

C. Challenges

The 6G-enabled IIoT (6G-IIoT) capacitates the robust, near-instant, and extreme-reliable interconnectivity between industrial devices to meet the strict communication requirements conscientiously. The near-instant 6G facilitates the real-time data exchange between the IIoT system while it poses great challenges to network measurement in 6G-enabled IIoT.

There are two challenges for measurement in 6G-enabled networks. The first challenge is the high processing speed and limited memory space. A fundamental problem for network measurement is to count the flow size, that is, the number of data packets in each flow. In high-speed 6G networks, it is typically impractical to record the size of each flow accurately [8]. The data structure of recording flow size should be small enough and placed in the on-chip memory, such as the block RAM in the FPGA or ASIC chip (usually less than 8.25MB). It is almost impossible to keep a counter for each flow to record its size. This is why most solutions focus on estimating the flow size with fixed memory and bounded error rates. Sketch based network measurements have been considered as the most promising solutions in recent years, because they greatly optimize the speed and memory usage at the cost of small error. With sketches, a sequential traversal of data packets can record the estimated flow size in limited memory. Figure 1 shows a system overview of sketch-based 6G network measurements. The framework is build on the 6G-enabled IIoT infrastructure and fundamental communication technologies, can comprise of: 1) sketches in the data-plane to aggregate counters and support simple statistics like flow size estimation and heavy hitter detection, and so on; (2) Algorithms in the control-plane to aggregate statistics from sketches to support complex measurements for applications like traffic burst detection, network fault detection and traffic anomaly detection, and so on.

The highly varying network traffic is another great challenge, since traffic in 6G-enabled IIoT fluctuates frequently. Spatially, heterogeneous devices in different regions experience significantly different flow rates and traffic variability. Temporally, the aggregate traffic usually peaks in daytime and falls at night. Moreover, traffic variability also exists in shorter time scales even when the total traffic remains the same. All these factors cause the highly varying network traffic, posing significant challenges to the sketch based measurement framework. State-of-the-art sketches typically have a separated structure with at least two parts, which record large flows and small flows, respectively. The first part is a key-value table used to precisely record the ID and size of large flows. The second part is commonly a hash table used to record the size of small flows in approximate. However, most of these sketches do not work well for varying network traffic. These sketches require to adjust their memory usage among multiple parts while the optimal point is sensitive to specified flow size distribution and memory size.

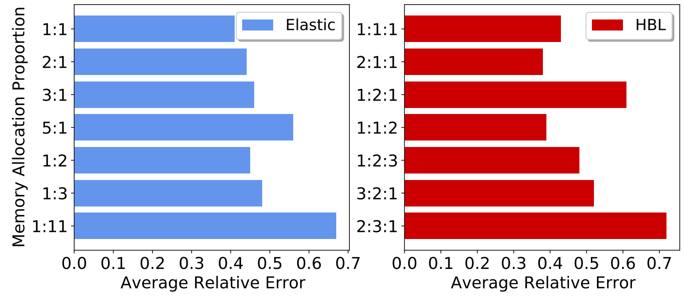

Fig. 2. Average relative error under different proportion of memory allocation. For the Elastic sketch, 1:1 means that the memory allocation proportion between the first part and the second part is equal. For the HBL sketch, 1:1:1 means that the memory allocation proportion between the first part, the second part and the third part is equal.

III. ML-empowered Measurement

A. Practice Motivation

As a probabilistic data compression technology, the classic implementation of sketch is to hash arrived packets into multiple buckets, accumulate the number of packets hashed into each bucket, and treat it as the estimated flow size. Due to hash collisions, sketches will cause estimation errors: mapping packets of different flows to the same bucket, estimates of different flows will affect each other. The level of estimation error largely depends on the mode of collision. Collisions with large flows will typically cause a significant estimation error, so it is best to avoid them. To avoid collisions with large flows, researchers adopted multi-part sketches using separate part to record large flows in precise and record the remaining flow approximately. Since the precise part and approximate part share limited memory, inter-part memory allocation becomes a critical issue that affects the estimation error. Existing studies use fixed or straightforward rules to allocate memory; unfortunately, we find these methods do not achieve ideal results due to unoptimized inter-part memory allocation with respect to varying flow distributions.

In fact, the adjustment to memory allocation is not a trivial problem in sketch design. If we reduce the proportion of memory occupied by the precise part in a sketch, some large flows can only be squeezed into the approximate part, resulting in increased estimation errors. Conversely, if we enlarge the proportion of memory occupied by the precise part in a sketch, more hash collisions will occur at the approximate part, which may also lead to an increase in estimation errors. The proportion of memory allocation between different parts of sketches is a determining factor in measurement accuracy. Figure 2 depicts the average relative errors (ARE) under different memory allocation proportions for two representative multi-part sketches: Elastic sketch and HBL sketch; The structure of Elastic comprises two separated parts, while the structure of HBL comprises a additional part. It is found when the memory allocation of each part is different, the estimation error of the sketch also changes. The proportion of memory of each part is an essential parameter that affects the accuracy of sketch-based network measurements.

In summary, it is necessary to design an adaptive mechanism to reasonably allocate the memory between different parts of the sketch, so that the network measurement can achieve higher accuracy. In this paper, we provide an essential building block for filling this void by adjusting the memory allocation with reinforcement learning (RL). RL as an important machine learning (ML) method, is broadly applied to solve decision-making problems. We use RL to model the memory adjustment problem. Using the powerful representation capabilities of deep neural networks, we train a neural network model that can dynamically decide on memory allocation based on a small sample of flow.

B. RL based Memory Allocation of Sketch

It can be seen from Figure 2 that the optimization of memory allocation is a challenge. Therefore, it is important to design a mechanism to reasonably tune the memory between different parts of the sketch such that network measurement can result a high accuracy. RL does not require any data in advance, and can update model parameters and obtain learning information based on the received environmental feedback on actions. Because of its many advantages, RL technology is widely used in the field of decision making, including information theory, game theory, automatic control and other fields. It is noted that RL is particularly suited for the memory tuning problem. First of all, memory allocation is usually highly repeatable, which generates a large amount of training data for RL algorithms. Second, RL algorithms can model complex systems and decision making strategies with deep neural networks, so that nonlinear strategies can be used in random environments. Third, RL algorithms can be trained for targets that are difficult to optimize due to the lack of accurate models directly. Last but not least, through periodic continuous learning, RL algorithms can dynamically adapt to changes in the system environment.

RL focuses on learning and tries to maintain a balance between exploration and exploitation. RL comprises of five components: agent, states, rewards, actions, and environment. Overall, RL describes a process of continuous interaction between the agent and environment, through learning strategies to maximize the goal revenue of a specific problem. Specifically, the agent can perform actions in the environment, observe the states and corresponding rewards of the actions, and adjust the actions accordingly. For each round of interaction, the agent observes the state and takes action. After performing this action, the state will change, and the agent will receive the corresponding reward. The probability of a state transition to reward depends only on the state of the environment and the agent's actions. In other words, continuous interactions are assumed to have Markov properties. The agent can only control its actions. It does not know how the state of the environment changes, nor does it know how the rewards are calculated. We name the framework *RL-Sketch* for short.

1) Agent: The agent of the reinforcement learning can be defined as a thing with behavioral capabilities, of course, it can also be a simple algorithm, such as an inverted pendulum, mobile robot, robotic arm, etc. can all be called an agent. The input and output of the agent are states and actions respectively, and actions are made according to the observed states. The selection from states to actions is called the strategy learning process. In *RL-Sketch*, the agent of reinforcement learning is defined as a deep neutral network. Through continuous interaction with the environment, the neutral network observes new states and new rewards and learns from the experience gained on how to make better decisions.

2) Environment: The environment of reinforcement learning can be defined as a relatively abstract probability model, or it can be the actual environment in which the agent is located. The environment receives the actions performed by the agent, and after the actions are executed, the change in state is fed back to the agent in the form of rewards. Through interactions between the environment and the agent, the best strategy can be learned to achieve a specific goal. In *RL-Sketch*, an environment is defined as a mapping between the memory allocation and the corresponding estimation error. The estimation errors are calculated from a sequence of packets uniformly sampled from the original flow.

3) States: The states are used as an input to the agent, which will change after the agent executes the action. The change of states can be used as a basis to judge whether the choice of action is good or bad. Reinforcement learning algorithms need to design variables that are more convenient to obtain and more convenient to observe. In *RL-Sketch*, states are defined as the proportion of memory space allocated to each part of the sketch.

4) Actions: These actions are generally discrete or continuous actions occurring in a space. In response to various target tasks, reinforcement learning needs to define different actions accordingly. After the actions are defined, the agent selects the actions to be executed according to the action selection policy. In *RL-Sketch*, actions are defined as the adjustment on the memory allocation proportion of different parts.

5) Rewards: The rewards help the agent to optimize in a direction that is more conducive to completing the goal. The design of the reward function has a significant impact on the learning effect. The rewards are the feedback obtained after performing an action. Generally, if the current state is closer to the target, a positive number is used to represent the reward, and if the target state is further away, a negative number is used to represent the penalty. In *RL-Sketch*, the reward function is defined as the ratio of the average relative error

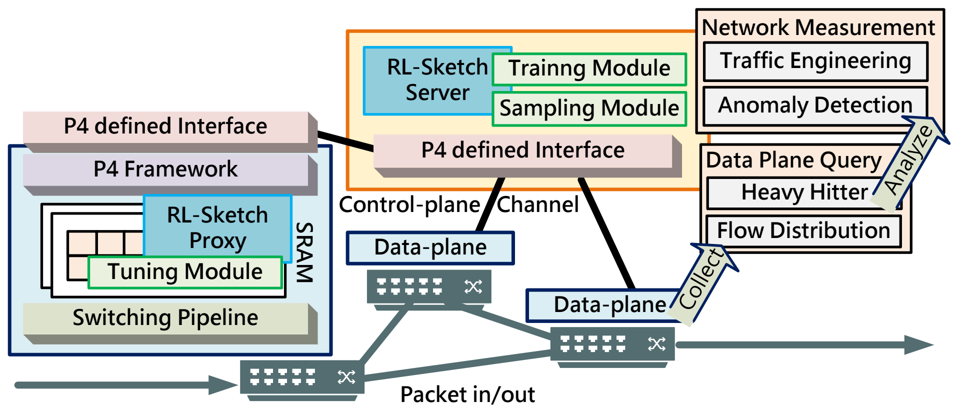

Fig. 3. Overview of integrated deployment on data-plane and control-plane.

across consecutive rounds. The training goal of reinforcement learning is to maximize the expectation of long-term rewards.

C. Implementation

The decision strategy used for memory adjustment is obtained through neural network training. The agent responsible for memory adjustment will observe certain indicators, such as the current memory proportion between different parts, input these values into the neural network, and then output current actions, that is, the memory adjustment decision. The agent will then observe the estimation errors obtained in the current round and use this information to train and improve the neural network model. *RL-Sketch* uses a deep RL algorithm based on the Actor-Critic framework. The Actor-Critic framework uses the Actor network and Critic network to learn and improve their strategies. The Actor network is responsible for the selection and execution of actions, which is understood as a policy function network. The Critic network is responsible for judging whether the Actor network action selection is good or bad, that is, a value function network that can avoid local optimal value problems.

The policy function is the basis for selection of actions, as required by the agent responsible for memory space adjustment. The policy function can be defined as the probability distribution of actions, that is, the probability of selecting actions in different states. In other words, the Critic network is responsible for learning the estimation of the value function from the observed rewards. *RL-Sketch* uses a time difference algorithm to train the parameters of the Critic network. The key idea in the policy gradient method is to estimate the gradient by observing the actions that are taken following the policy. The agent first samples the memory adjustments, and then updates the policy parameters according to the gradient descent method. Specifically, to calculate the step size of each parameter update, we need to obtain an estimation of the value function, that is, the total expected rewards following the policy in the state. With the Critic network, the Actor network can complete an update of the policy parameters. To help the policy converge, we use the entropy regularization term in Actor network. The entropy regularization term updates the policy parameters in the direction of increasing rewards, and finally converges to an optimal policy.

The input of the Actor network is the different part memory allocation over the past multiple rounds. They are fed into a 1×3 one-dimensional convolutional neural network (1D-CNN) with 128 filters. When all the outputs go through a fully connected layer, a 1×128 vector is obtained. Finally, a decision is made using the softmax function, and a vector describing the selection probability of the different actions is obtained. Each action describes the memory allocation changes in different parts. The structure of the Critic network is the same as that of the Actor network. The difference is that the final output of Critic network is a single value, not a vector. In the Actor-Critic framework, the Critic network's role is limited to training the Actor network, and only the Actor network is needed to perform the actual memory allocation. To speed up the training process of neural networks, we use the A3C technology to deploy multiple agents at the same time to carry out parallel learning. In each agent, the input from the neural network is independent of the overall input. The states, actions, and rewards corresponding to each round of the agent will be sent to the parameter server in the form of triples, which will then be used to generate a training model.

Code and Data sharing. We make the model snapshot and code used for *RL-Sketch* available to the research community, in the hope that this stimulates and facilitates further research. https://github.com/FlowAnalysis/RL_MemoryAllocation

D. Deployment

As the 6G networks move into an era of massive amounts of data, traffic is becoming heavier in some critical scenarios. Modern high-speed data-plane devices forward packets at a speed of hundreds of GB or even hundreds of TB per second. The amount of flows that traverse a core data-plane device can be tens of millions. Simultaneously tracking of such a large number of flows poses a great challenge for both the hardware and software used to deploy sketch-based network measurements. In order to maintain a high throughput, data-plane devices forward packets from incoming ports to outgoing ports via a fast switching pipeline, bypassing primary memory and CPU resources. If the sketch's logic is implemented as a module to process packets in real-time, one way to implement it on network processors is to use on-chip cache memory to store the IO ports. The cache on processor chips is SRAM, typically with a much faster IO rate and a much higher price than DRAM. SRAM generally has to be shared across multiple functions for forwarding, routing, and other security purposes. In that context, the amount of memory that can be allocated for different parts can be very limited. Figure 3 shows an overview of integrated deployment on data-plane and control-plane.

P4 is a language that specifies how the switch processes packets. It provides a register abstraction that offers a form of stateful memory which can store user-defined data structures and can be arranged into one-dimensional arrays of user-defined length. The register cells can be read or updated via a P4 action and are also accessible through the control plane API. The P4 framework allows *RL-Sketch* to define an API for control-data plane communication. Alternatively, we can extend the interface and implement a proxy in *RL-Sketch*, which comprises a tuning module for tuning memory proportions from the control-plane. Software Defined Networking (SDN) makes network management easier by separating the control plane and data plane. Network monitoring with *RL-Sketch* is a natural fit with SDN as operators only need to install a training module or a decision module into the controller. From this perspective, we can assume that the *RL-Sketch* server is a part of SDN controller. In the future, with more flexible and novel platform design, sketch-based measurement technology needs to embrace these changes and evolve within the network itself.

IV. Evaluation

We prototype proposed mechanism of memory allocation in this paper based on deep reinforcement learning, and the implementation is based on Tensorflow v1.1.0 and TFLearn v0.3.1. As experimental hardware environments, we conducted all the experiments on a server equipped with 6 Intel® Core™ i7-8700K 3.70GHz CPU cores. The server's total memory is 62.9 GB, and runs the Ubuntu 16.04 operating system.

We experiment using public Internet traffic datasets collected from CAIDA's Equinix-Chicago monitors to evaluate *RL-Sketch*'s performance. The number of data packets in the experimental dataset is about 2.5M, and the number of flows is about 110K. The packet element consists of five tuples for identification: source IP address, destination IP address, source port number, destination port number, and protocol type. There are a total of 11 trace files in the dataset. We select one of the trace files to create the training set required by the deep reinforcement learning algorithm. During the measurement process, we recorded the different parts' memory allocations and the corresponding relative error rates of the different parts. The remaining 10 trace files are used to test the effectiveness of the memory allocation mechanism proposed in this paper, that is, after adjusting the memory allocation proportion, test whether the results of network measurements can achieve higher accuracy.

We apply *RL-Sketch*'s power to two representative multi-part sketches: Elastic sketch and HBL sketch, respectively. To match situations where available memory for different parts may be very limited, we set the total memory available for

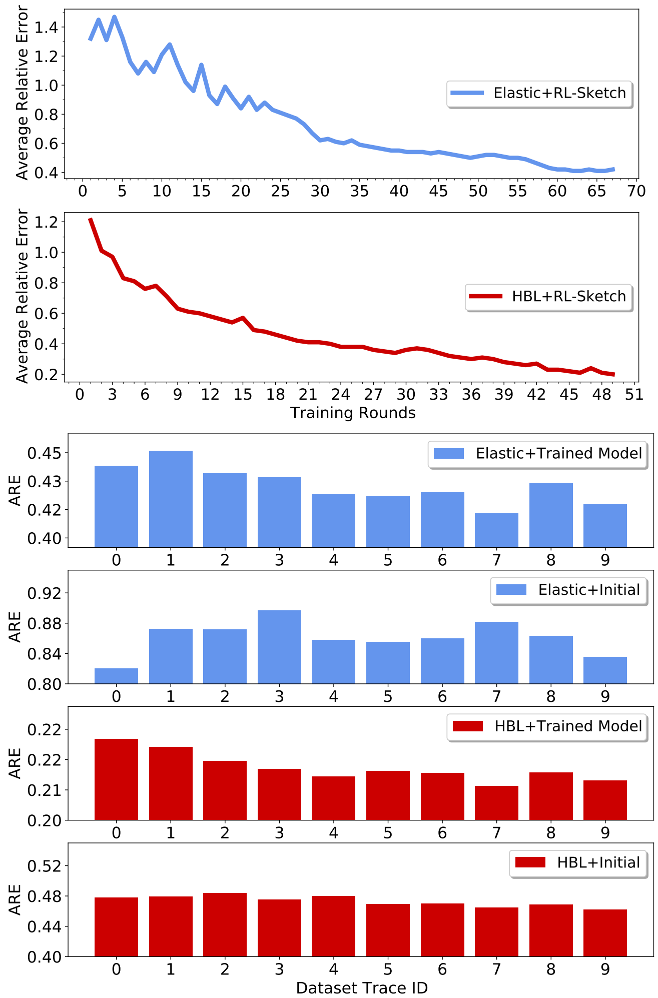

Fig. 4. The trend of measurement accuracy as the training progresses and the effect of applying the trained model to other traffic traces.

each sketch to 600KB. For Elastic sketch, we set its initial memory allocation proportion as (570KB, 30KB), that is, we assign 570KB to its precise part and 30KB to its approximate part. For HBL sketch, we set its initial memory allocation size as (20KB, 400KB, 180KB), that is, we assign 20KB to its first part, 400KB to its second part and 180KB to its third part.

Figure 4 depicts the trend of measurement accuracy as the training progresses, where the x-axis coordinates represent the number of training rounds. For Elastic sketch, the average relative error with an initial memory allocation proportion is 1.3. After 60 rounds of training, its average relative error drops to about 0.4. For an HBL sketch, the average relative error with an initial memory allocation proportion is 1.2. After 45 rounds of training, its average relative error drops to about 0.2. Figure 4 also shows the effect of applying the trained model to other traffic traces, where the x-axis coordinates represent the ids of the trace files. Compared to initial memory allocation with a straightforward method, a memory allocation tuned by the trained model can significantly reduce estimation errors. Moreover, the trained model can produce broad effects on different traffic traces, which shows that reinforcement learning can effectively capture the general characteristics of memory allocation and estimation error.

V. Conclusion

In this paper, we comprehensively review the sketch techniques for space-efficient flow measurements, and their potential power in 6G-enabled IIoT. We observe that memory allocation in state-of-the-art multi-part sketches significantly affects the accuracy of flow estimation. To improve the estimation accuracy of multi-part sketches, we propose *RL-Sketch*, an inter-part memory allocation mechanism based on reinforcement learning, to optimize memory allocation of multi-part sketches. *RL-Sketch* can reasonably adjust the memory proportions between different parts to make the network measurement results more accurate. We use real Internet datasets to experimentally verify the performance of *RL-Sketch*. Experimental results show, compared with state-of-the-art solutions, *RL-Sketch* can significantly improve measurement accuracy.

References

[1] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, D. Niyato, O. Dobre, and H. V. Poor, "6g internet of things: A comprehensive survey," *IEEE Internet of Things Journal*, 2021.

[2] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6g networks: Use cases and technologies," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 55–61, 2020.

[3] G. Han, J. Tu, L. Liu, M. Martínez-García, and Y. Peng, "Anomaly detection based on multidimensional data processing for protecting vital devices in 6g-enabled massive iiot," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5219–5229, 2021.

[4] H. Qi, J. Wang, W. Li, Y. Wang, and T. Qiu, "A blockchain-driven iiot traffic classification service for edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2124–2134, 2020.

[5] Y. Zhou, Y. Zhang, C. Ma, S. Chen, and O. O. Odegbile, "Generalized sketch families for network traffic measurement," *ACM SIGMETRICS Performance Evaluation Review*, vol. 48, no. 1, pp. 63–64, 2020.

[6] T. Yang, J. Jiang, P. Liu, Q. Huang, J. Gong, Y. Zhou, R. Miao, X. Li, and S. Uhlig, "Adaptive measurements using one elastic sketch," *IEEE/ACM Transactions on Networking*, vol. 27, no. 6, pp. 2236–2251, 2019.

[7] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6g be?" *Nature Electronics*, vol. 3, no. 1, pp. 20–29, 2020.

[8] Q. Xiao, Z. Tang, and S. Chen, "Universal online sketch for tracking heavy hitters and estimating moments of data streams," in *Proceedings of the IEEE Conference on Computer Communications*, 2020, pp. 974–983.