RESEARCH-ARTICLE

Datacenter Network Deserves Better Traffic Models

SIJIANG HUANG, Tsinghua University, Beijing, China

LINGFENG PENG, Tsinghua University, Beijing, China

MOWEI WANG, Huawei Technologies Co., Ltd., Shenzhen, Guangdong, China

YASHE LIU, Huawei Technologies Co., Ltd., Shenzhen, Guangdong, China

ZHENHUA LIU, Huawei Technologies Co., Ltd., Shenzhen, Guangdong, China

XIN WANG, Stony Brook University, Stony Brook, NY, United States

View all

Open Access Support provided by:

Huawei Technologies Co., Ltd.

Tsinghua University

Stony Brook University

Published: 28 November 2023

Citation in BibTeX format

HotNets '23: The 22nd ACM Workshop on Hot Topics in Networks
November 28 - 29, 2023
MA, Cambridge, USA

Conference Sponsors:
SIGCOMM

Datacenter Network Deserves Better Traffic Models

Sijiang Huang
Tsinghua University

Lingfeng Peng
Tsinghua University

Mowei Wang
Huawei Technologies

Yashe Liu
Huawei Technologies

Zhenhua Liu
Huawei Technologies

Xin Wang
SUNY Stony Brook

Yong Cui*
Tsinghua University

ABSTRACT

Traffic modeling of Datacenter Network (DCN) today is oversimplified, deviating from the ground truth. Adopted by numerous researchers, the common practice relies on the assumptions of traffic homogeneity and independence for ease of use. Based on our investigation of a real-world traffic dataset, we disprove these assumptions and point out the severe fidelity issue of the common practice that could invalidate many motivations and conclusions from influential research works. In this paper, we present *Encore*, a DCN traffic modeling framework for fine-grained traffic modeling and high-fidelity synthetic traffic generation. Leveraging machine learning techniques, Encore effectively extracts and preserves essential distribution and sequential features from raw traffic. Preliminary experiments demonstrate that the traffic generated by Encore not only restores the key features of real traffic but also achieves high consistency when used to evaluate network performance. We envision further expanding Encore to full-process traffic modeling and generation, and expect these critical improvements in traffic models can facilitate the DCN performance evaluation and optimization.

CCS CONCEPTS

• **Networks** → **Network simulations**; **Data center networks**;

KEYWORDS

Datacenter network, network traffic modeling

ACM Reference Format:
Sijiang Huang, Lingfeng Peng, Mowei Wang, Yashe Liu, Zhenhua Liu, Xin Wang, and Yong Cui. 2023. Datacenter Network Deserves Better Traffic Models. In *The 22nd ACM Workshop on Hot Topics in Networks (HotNets '23), November 28–29, 2023, Cambridge, MA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3626111.3628209

*Corresponding author (cuiyong@tsinghua.edu.cn)

HotNets '23, November 28–29, 2023, Cambridge, MA, USA

ACM ISBN 979-8-4007-0415-4/23/11…$15.00
https://doi.org/10.1145/3626111.3628209

1 INTRODUCTION

Traffic modeling and generation play an important role in the Datacenter Network (DCN). Traffic models extract and preserve key features from raw traffic and use these features to guide the generation of synthetic data, which can be widely used in various scenarios, including but not limited to evaluating network performance, testing new network designs, and fueling data-driven approaches.

The most commonly used method to model traffic in DCN is quite simple and has gained favors from many researchers over the years. It models the flow-level traffic based on the assumptions of homogeneity and independence, and generates new traffic by repetitive independent sampling from a global average distribution of flow sizes. By studying the public traffic data of a real DCN [5] we found these assumptions untenable in practice. Real traffic exhibits evident heterogeneity of flow size distribution and temporal correlation of flow size sequences. Failing to include such characteristics in traffic modeling could induce severe fidelity problems in the generated traffic, and potentially invalidate important results of network performance evaluation and optimization.

In this paper, we envision a better DCN traffic modeling that satisfies the following requirements: 1. *fidelity*. It should preserve fine-grained key traffic characteristics such as statistical, temporal, and spatial features. 2. *ease of use*. It should be easily used to generate previously unseen synthetic traffic. 3. *confidentiality*. The generation process should not rely on the presence of original traffic data. 4. *explainability*. The model parameters should hold clear physical meanings and the generation process should be explainable. The common practice favors the latter three requirements to the extreme at the expense of sacrificing fidelity. However, we believe that the first criterion (fidelity) should always come first, without which none of the other characteristics would matter.

Modeling traffic in DCN involves a number of issues: locality modeling, statistical modeling, and sequential modeling. In this paper, we first tackle the problem of pair-level flow size modeling and present our vision of the whole system design including all factors listed above. Pair-level flow size

modeling takes the flow trace between a pair of hosts (a source and a destination) and extracts statistical (flow size distributions) and temporal (sequential dependency between flow sizes) features for traffic generation.

We propose Encore, a framework for fine-grained DCN traffic modeling and high-fidelity synthetic traffic generation. Facing intertwined features of traffic traces, Encore adopts a divide-and-conquer philosophy and designs different models for several well-defined tasks. For pair-level distributional and sequential modeling, Encore adopts the Variational Autoencoder (VAE) [10] to model the distribution of flow size distributions. Taking the distribution as input, Encore uses the Gated Recurrent Unit (GRU) [6] to model the sequential features of traffic that can be used to generate unseen traffic traces that satisfy high-order dependency features. With the flow size distribution explicitly linking the two models, readable statistical information is preserved for better explainability. Preliminary results provide evidence that compared with the common practice, Encore effectively preserves crucial traffic characteristics including diversity in distributions and sequential structures. We utilize Encore to generate traffic traces for DCN performance evaluation and the results show high consistency with the ground truth.

With Encore, we make the following contributions:

- Based on our investigation of real-world DCN traffic, we point out the fundamental deficiencies of the common practice of traffic modeling: unreliable assumptions of traffic heterogeneity and independence.
- We design Encore, a DCN traffic modeling framework for fine-grained traffic modeling and high-fidelity synthetic traffic generation.
- We present our vision of full-process traffic modeling and generation based on Encore and discuss possible technical roadmaps to achieve it.

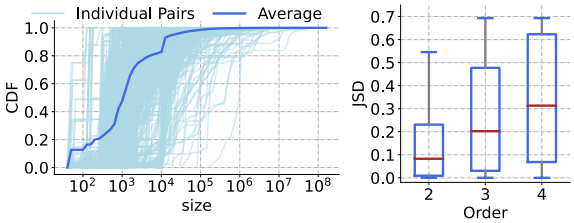
The rest of this paper is organized as follows: §2 provides the background of DCN traffic modeling and our motivation. The design of Encore is presented in §3 and preliminarily evaluated in §4. Our vision of full-process traffic modeling and generation is shown in §5, and §6 concludes the paper.

2 DCN TRAFFIC MODELING

In this section, we introduce the basic concepts of flow-level traffic modeling in DCN, followed by the deficiencies of the current common practice to motivate our research.

2.1 Flow-level Traffic Modeling

Traffic modeling in DCN can be conducted at the packet [2] or flow level [12]. In this paper, we choose the latter since the concept of flow is essential in DCN, and Flow Completion Time (FCT) is one of the most concerned performance metrics in DCN [7]. A flow is a set of packets sharing the same five-tuple (source/destination IP address, source/destination port, transport protocol) with two important features: its *start time* (the sending time of the first packet) and its *size* (the sum of all packet sizes). A traffic trace is a series of flows sharing the same source and destination ordered by their start time. Traffic modeling refers to the procedure of abstracting key features from raw traffic traces, such as spatial, temporal, and statistical features.



(a) Distribution heterogeneity **(b) Sequential dependency**

Figure 1: Structures in real DCN traffic

2.2 Why DCN Needs Traffic Models?

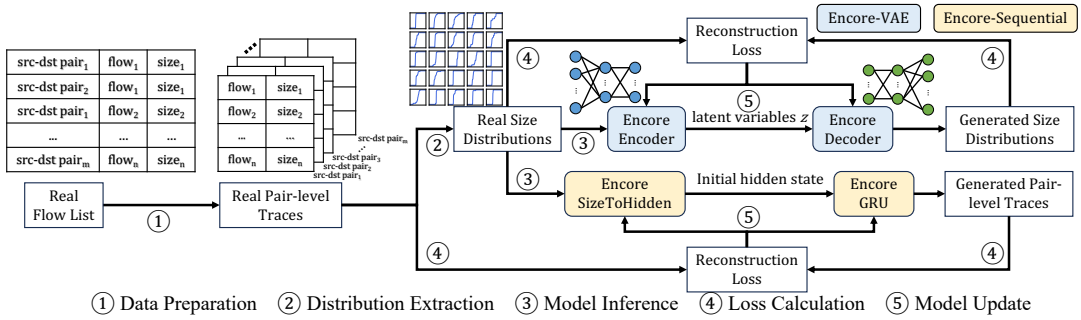
Traffic is the input and one of the key variables in DCN. A great deal of conclusions cannot be correctly drawn without traffic as the condition. One cannot prove something works (e.g., a new congestion control algorithm can reduce the tail latency) or something is better (an algorithm outperforms another) without the traffic for evaluation.

Although traffic is being generated and transmitted all the time and everywhere, many practical concerns (e.g., policy, privacy, and legal restrictions) impede such data from being shared freely [20]. Even if one has access to real traffic, there are still many restrictions on its usage. For example, when applying the traffic to a different topology or requiring high-load traffic that does not appear in the dataset, raw data have limited variety and flexibility. As an alternative, the traffic model can generate synthetic data to meet the needs of a wide range of tasks, including evaluating network performance, testing new designs and training data-driven models.

2.3 Why DCN Needs Better Traffic Models?

The common practice of traffic modeling and generation is incredibly simple: A global flow size distribution is used to model traffic and to generate new traffic between a source and designation pair, flow sizes are *independently* sampled following this *single* distribution to form a size sequence. Notwithstanding its over-simplicity, this method is popularly used by literature studies with profound influences on network research [1, 3, 4, 8, 11, 13, 21].

This common practice relies on two key assumptions: 1. flow size distributions are *homogeneous*, i.e., flow sizes of different host pairs follow the same distribution. 2. flow sizes are



(a) Training Phase

(b) Generation Phase

Figure 2: Encore framework

independent, i.e., they are generated independently following this distribution. However, real-world network traffic is a lot less ideal. We derive the flow-level traffic demand from a publicly available real-world packet trace [5] to demonstrate that real traffic conforms to neither of the above assumptions. In particular, we acquire 1,000 traces and investigate their flow size distributions and flow size sequences.

Figure 1a shows the global average flow size distribution and that of each trace from individual pairs. We observe that the average distribution is not representative of the entire dataset. Mixing traces containing mostly elephant flows with those containing mostly mice flows to form an average distribution casts away the information on size diversity therefore significantly compromising the model accuracy. Figure 1b shows the sequential dependency within each trace. We measure this dependency using the deviation of high-order distributions of flow sizes since temporal dependency would render certain tuples/triplets/quadruples of flow sizes to appear more frequently compared with independent sampling. Specifically, we randomly permute the flows within each trace to eliminate their temporal dependency and calculate Jensen-Shannon Divergence (JSD) between the original traces and the permuted ones of different orders. The results demonstrate the evident existence of temporal structures.

Even though the common practice has many problems, there are reasons that make it favorable among researchers. On the one hand, it's incredibly easy to use. With the model, one can generate unlimited traffic satisfying certain statistical features by repeatedly sampling from a given distribution. On the other hand, it's explainable. The parameters in this model are probabilities with clear physical meanings. What's more, it has great confidentiality. Keeping only the statistical information, this method does not require the original traffic to be available for generation, preventing information leakage. With that being said, none of the above compensate for its poor fidelity. It only has coarse-grained distribution modeling while completely ignoring the sequential modeling of temporal dependency, losing information essential to traffic reconstruction. The main goal of this paper is to devise a traffic modeling method that addresses the fidelity problem while preserving other characteristics, which can extract and preserve sufficient key features of the original traffic and be easily used to generate realistic synthetic traffic.

3 ENCORE

We present Encore, a DCN traffic modeling framework that aims to address the vital fidelity problems of common practice: modeling flow size distributions and sequences.

3.1 A Traffic Model that DCN Needs

Figure 2 overviews the Encore framework, which consists of a training and a generation phase. The former is responsible for extracting essential features from real traffic data and delivering several trained models to the latter for traffic generation. We choose the VAE to model the flow size distributions for its excellent capability of extracting latent features from data and generating from "thin air". The GRU is selected to model the sequential structure of traffic traces to strike a balance between decent accuracy and efficient training. We will get to the details in §3.2 and §3.3.

At a high level, the training phase consists of the following steps: Break the original flow list down by grouping the flows by their host pairs to acquire a series of traces (①). For each trace, we place the flow sizes into $N = 30$ buckets with hand-picked boundaries (we will discuss the implications

of this choice in §5) and count the frequency of each flow size to get the real size distribution (②). With Encore-VAE and Encore-Sequential modules, we perform model inference in an attempt to reconstruct the input distribution or the original sequence (③). Using the loss calculated by the difference between the reconstructed data and the input (④), we iteratively update the models until the loss is sufficiently low or the training steps reach a predetermined value (⑤).

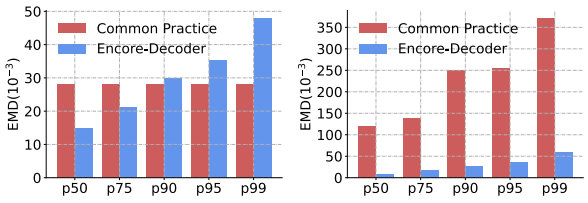
The generation phase of Encore is much simpler. After sampling a vector of latent variables from a standard normal distribution, the VAE decoder transforms it into a flow size distribution and passes it to the sequential models to generate a sequence of flow sizes.

3.2 Distribution Modeling

Encore uses VAE to model fine-grained flow size distributions, that is, the distribution of flow size distributions. Given a set of distributions, we aim to extract the essential latent features among them for later creation. The Encore-VAE model is composed of an encoder and a decoder, each implemented with a Neural Network (NN) [19]. The encoder takes a given probability distribution of flow sizes and encodes it into a vector in the latent space by which the decoder tries to reconstruct the distribution. The models are trained to minimize the reconstruction error while restraining the latent variables to the standard normal distribution with an additional loss item (Kullback-Leibler Divergence) that measures the difference between the latent distribution and the standard normal distribution. After training, the decoder can generate flow size distributions similar to the inputs taking only random variables sampled from a given distribution.

3.3 Sequence Modeling

The Encore-Sequential model is responsible for generating sequences of flow sizes given flow size distributions, which can be achieved using Recurrent Neural Networks (RNN) [6, 9, 18] in their conditional forms. In this work, we use GRU for this task. Modeling long sequences is inherently difficult for sequential models. In the special case of traffic generation, we contend that modeling arbitrary long sequences is neither feasible nor necessary. We choose to model size sequences with a length of $L = 16$ to make a trade-off between preserving sufficient sequential information and efficient model training, that is, we use the model to generate sequences of 16 consecutive flows and concatenate them to form longer traces. Another problem is how to incorporate the flow size distribution as a condition. We find the initial hidden state is a decent choice to place this additional information. We use an auxiliary NN, SizeToHidden, to embed the flow size distribution into the initial hidden state of GRU. When generating



(a) Accuracy (b) Coverage

Figure 3: Performance of distribution modeling

new traffic, at each step, GRU generates a probability distribution of the next sizes. Instead of taking the one with the highest probability, we sample the next size from this distribution to manually introduce randomness to improve trace diversity. Otherwise, we would end up with almost exactly the same traces as the original ones, which is against the primary purpose of generating previously **unseen** traffic.

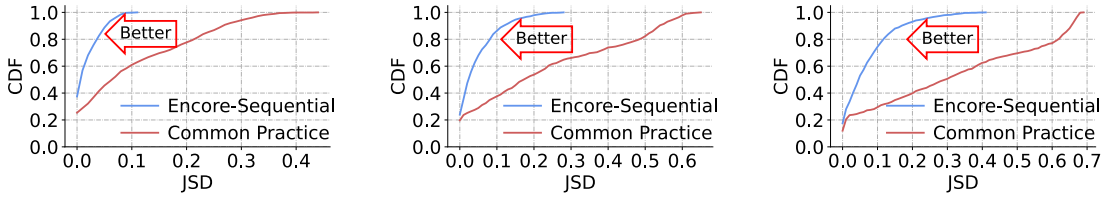
4 PRELIMINARY EVALUATION

In this section, we first put Encore to the test to examine whether they can preserve key features of the real traffic. Then, we further dig into its usability by testing its ability to maintain performance consistency in an evaluation task.

4.1 Effectiveness of Distribution Modeling

Metrics: Measuring the quality of a synthetic dataset is much more complex than one data point. For a single data point, the reconstruction error is a decent measurement. However, for datasets where there are no clear one-to-one mappings, different metrics are needed. In this paper, we consider two metrics: accuracy and coverage. **Accuracy** measures how close the generated distributions resemble the real ones, i.e., whether a traffic model is correct. **Coverage** measures how the generated set covers the real distributions, i.e., whether a traffic model is comprehensive. To be more specific, given a real set T and a generated set T', both of M distributions, we define accuracy as the minimum reconstruction error between a generated distribution and the distributions in T and coverage as the minimum reconstruction error between a real distribution and the synthetic distributions in T'.

It's relatively easier to achieve high accuracy. An extremely conservative model (like the common practice) can only keep one distribution that resembles any of the M real distributions and repeat it as many times as needed for generating new traffic and never "makes a mistake" at the cost of poor coverage. On the other hand, high coverage implies a traffic model truly reserves comprehensive information on the real traffic. In the pursuit of coverage, a model is encouraged towards diversity, increasing the chances of inaccuracy. One should carefully trade-off between accuracy and coverage.



(a) Second-order dependency **(b) Third-order dependency** **(c) Fourth-order dependency**

Figure 4: Accuracy of sequential dependency modeling

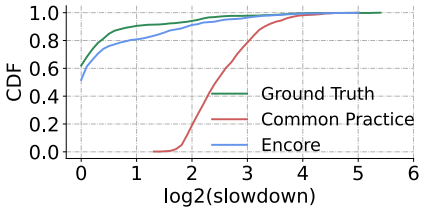


Figure 5: Performance consistency of Encore

We use the same 1,000 traces as before and calculate their flow size distributions as the ground truth. We use the Earth Mover's Distance (EMD) [17] to measure the difference between distributions. The smaller the distance, the higher the accuracy and the coverage. Figure 3 shows the percentiles of accuracy and coverage of Encore-Decoder and the common practice (using the average size distribution). The results show that while achieving excellent accuracy (outperforms the common practice in more than 75% of cases, comparable to the common practice at the tail), Encore-Decoder has a remarkable performance improvement on coverage over the common practice (over 5× reduction on EMD at the tail), providing strong evidence that Encore-Decoder can effectively preserve fine-grained distribution features of real traffic.

4.2 Effectiveness of Sequential Modeling

Metrics: Measuring how well traffic models keep the sequential information is straightforward. Smaller JSDs of high-order tuples between the generated and the original traces indicate better preservation of the sequential structure.

Using Encore-Sequential and the common practice (independent sampling from the distribution), we generate traces respectively and compare their performance. Figure 4 demonstrates that Encore is able to recreate traces with authentic sequential dependency of different orders, significantly reducing the JSD compared with the common practice.

4.3 What's More: Performance Consistency

Our ultimate goal is not to obtain accurate models, but to generate, through the models, traffic that can assist downstream tasks, e.g., network performance evaluation. In this paper, we conduct simulation-based experiments with the ns-3 simulator [15] with a simple setting and leave more complex scenarios for future work.

We consider a scenario where each trace traverses a bottleneck whose bandwidth shrinks from 100Gbps to 10Gbps. The flows are controlled by DCQCN [21] with the default parameters provided by [11]. The sending intervals between flows are sampled from a Poisson distribution to generate a 30% load from the view of the sender. Flows are delayed according to the congestion states of the bottleneck link, which is the result of the flow size distribution and the sequential dependency between flow sizes. We choose the average slowdown of flow completion time as the performance metric and show the evaluation results of different traffic generation methods. Figure 5 plots the cumulative probability distribution (CDF) of experimental results for the 1,000 traces, and take the logarithm of the results for better demonstration.

In over half of the traces, the network experiences little to no congestion, while on the other hand, a small number of traces lead to extreme congestion. Compared to the common practice, the traces generated by Encore have a better consistency with the ground truth obtained using the original traffic in terms of network performance. Benefiting from fine-grained distribution modeling and accurate sequential modeling, Encore can cover more diverse network congestion situations. The common practice, on the contrary, fails to capture this diversity and inevitably results in inaccurate performance. Numerically, the error (measured by the EMD to the ground truth) of Encore is **an order of magnitude smaller** than that of the common practice.

5 WHAT'S NEXT: ENCORE IN ACTION

Pair-level trace modeling is a significant step forward in DCN traffic modeling, yet the endeavor is not accomplished. To put Encore into actual use, besides modeling flow size distributions and sequences, several other problems require further exploration. Figure 6 presents our vision of the next step of Encore, full-process traffic modeling and generation.

In our vision, traffic modeling is solely done by data holders (e.g., service providers) who provide models to the data users (e.g., researchers in universities) for traffic generation. By only sharing models, one can preserve the authenticity of

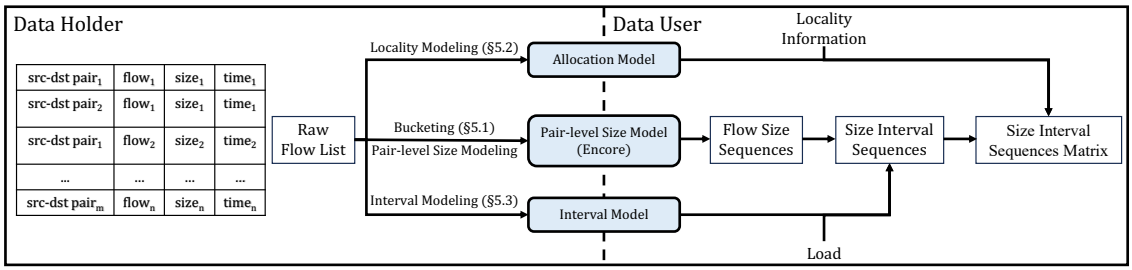


Figure 6: Full-process traffic modeling and generation

traffic features while ensuring the privacy of other sensitive information. Presented in this paper, the pair-level size model is responsible for generating flow sequences of different size distributions. Then, the interval model "inserts" the intervals between flows according to a user-defined load. The allocation model further incorporates the locality information into traffic generation by assigning each size-interval sequence to a host pair to form a matrix of traffic. In the following, we will discuss the challenges of full-process traffic modeling and generation, and possible solutions.

5.1 Bucketing

As alluded to earlier (§3.1), we put the flow sizes into N buckets before counting their frequency. When generating new flow sizes, after choosing one bucket, we uniformly sample within its range. The error introduced by bucketing should also be taken into consideration. Since mice flow usually takes up the majority, one should place more buckets to distinguish small flow sizes and reduce the overall loss in precision. However, elephant flow can impose a greater influence in the context of network congestion than mice flow. Therefore, coarse-gained bucketing of elephant flow potentially induces severe problems.

Given the number of buckets N, we can form the problem of finding the bucket boundaries as an optimization problem that can be solved using heuristic search algorithms such as the Bayesian Optimization algorithm [14]. The challenge here is finding the right optimization target. We contend that one should set the target to strike a balance between minimizing the overall and per-trace reconstruction error while providing an upper bound for errors of elephant flow.

5.2 Traffic Locality

In this work, we model the statistical and temporal features of traffic. A critical dimension we haven't fiddled with is spatial, in other words, traffic locality. In DCN, the traffic between two hosts depends on their locality. For example, traffic within a region (Rack, Point-of-Delivery) could differ from traffic across a region [5, 16]. Traffic from "hot" hosts tends to have special statistical and temporal features.

Considering that traffic models are often used in different topologies, such location information is less desired to be hardwired. Instead, we want to establish a mapping relation from the location features to other features. We can include such features by adding amendments to Encore such as training an allocation model that takes the generated traces and finds a suitable place in the traffic matrix for it.

5.3 Modeling Intervals

So far, we have focused on traffic features regarding flow size. Another factor that needs attention is the intervals between flows. We briefly mentioned in §4.3 that we use Poisson intervals between flows, as usually assumed by existing works. Preliminary experiments (not shown in this paper) provide evidence that this assumption does not always hold either.

Modeling intervals introduces new challenges since it should also consider the role of traffic loads, in addition to the heterogeneity and dependency. Interval modeling aims to address the "what if" problem of traffic modeling, i.e., given a traffic load not seen in the original traffic, can we generate plausible synthetic intervals? Since real traffic is often light-loaded, data users have to devise a reasonable method to increase the load of synthetic traces when heavily loaded traffic is needed (e.g., for stress testing). One might consider resorting to the generalizability of machine-learning approaches when linear scaling no longer applies.

6 CONCLUSION

In this paper, we present *Encore*, a DCN traffic modeling framework for pair-level flow trace modeling. *Encore* leverages VAE and GRU to model fine-grained distributional and sequential traffic features and generate synthetic flow size traces with high fidelity. We hope *Encore* can shed light on more practical traffic modeling and generation in the future.

ACKNOWLEDGMENTS

We thank our anonymous reviewers for their insightful feedback. This work was supported by the NSFC Project under Grant 62132009 and Grant 62221003.

REFERENCES

[1] Mohammad Alizadeh, Shuang Yang, Milad Sharif, Sachin Katti, Nick McKeown, Balaji Prabhakar, and Scott Shenker. 2013. PFabric: Minimal near-Optimal Datacenter Transport. In *Proceedings of the ACM SIGCOMM 2013 Conference (SIGCOMM '13)*. Association for Computing Machinery, New York, NY, USA, 435–446. https://doi.org/10.1145/2486001.2486031

[2] Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid. 2020. On the Complexity of Traffic Traces and Implications. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '20)*. Association for Computing Machinery, New York, NY, USA, 47–48. https://doi.org/10.1145/3393691.3394205

[3] Wei Bai, Li Chen, Kai Chen, Dongsu Han, Chen Tian, and Hao Wang. 2015. Information-Agnostic Flow Scheduling for Commodity Data Centers. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation (NSDI'15)*. USENIX Association, USA, 455–468.

[4] Ran Ben Basat, Sivaramakrishnan Ramanathan, Yuliang Li, Gianni Antichi, Minian Yu, and Michael Mitzenmacher. 2020. PINT: Probabilistic In-Band Network Telemetry. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 662–680. https://doi.org/10.1145/3387514.3405894

[5] Theophilus Benson, Aditya Akella, and David A. Maltz. 2010. Network Traffic Characteristics of Data Centers in the Wild. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*. Association for Computing Machinery, New York, NY, USA, 267–280. https://doi.org/10.1145/1879141.1879175

[6] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1724–1734. https://doi.org/10.3115/v1/d14-1179

[7] Nandita Dukkipati and Nick McKeown. 2006. Why Flow-Completion Time is the Right Metric for Congestion Control. *SIGCOMM Comput. Commun. Rev.* 36, 1 (jan 2006), 59–62. https://doi.org/10.1145/1111322.1111336

[8] Peter X. Gao, Akshay Narayan, Gautam Kumar, Rachit Agarwal, Sylvia Ratnasamy, and Scott Shenker. 2015. PHost: Distributed near-Optimal Datacenter Transport over Commodity Network Fabric. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '15)*. Association for Computing Machinery, New York, NY, USA, Article 1, 12 pages. https://doi.org/10.1145/2716281.2836086

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (nov 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[10] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR '14)*. http://arxiv.org/abs/1312.6114

[11] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang, Zheng Cao, Ming Zhang, Frank Kelly, Mohammad Alizadeh, and Minlan Yu. 2019. HPCC: High Precision Congestion Control. In *Proceedings of the ACM SIGCOMM 2019 Conference (SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 44–58. https://doi.org/10.1145/3341302.3342085

[12] Zhiwen Liu, Mowei Wang, and Yong Cui. 2022. Locality Matters! Traffic Demand Modeling in Datacenter Networks.. In *Proceedings of the 6th Asia-Pacific Workshop on Networking (APNet'22)*. 525–532.

[13] Behnam Montazeri, Yilong Li, Mohammad Alizadeh, and John Ousterhout. 2018. Homa: A Receiver-Driven Low-Latency Transport Protocol Using Network Priorities. In *Proceedings of the ACM SIGCOMM 2018 Conference*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3230543.3230564

[14] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. 1999. BOA: The Bayesian Optimization Algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1 (GECCO'99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 525–532.

[15] George F Riley and Thomas R Henderson. 2010. The ns-3 network simulator. In *Modeling and tools for network simulation*. Springer, 15–34.

[16] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren. 2015. Inside the Social Network's (Datacenter) Network. In *Proceedings of the ACM SIGCOMM 2015 Conference (SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 123–137. https://doi.org/10.1145/2785956.2787472

[17] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *Int. J. Comput. Vision* 40, 2 (nov 2000), 99–121. https://doi.org/10.1023/A:1026543900054

[18] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. *Learning Internal Representations by Error Propagation*. MIT Press, Cambridge, MA, USA, 318–362.

[19] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks* 61 (2015), 85–117.

[20] Yucheng Yin, Zinan Lin, Minhao Jin, Giulia Fanti, and Vyas Sekar. 2022. Practical GAN-Based Synthetic IP Header Trace Generation Using NetShare. In *Proceedings of the ACM SIGCOMM 2022 Conference (SIGCOMM '22)*. Association for Computing Machinery, New York, NY, USA, 458–472. https://doi.org/10.1145/3544216.3544251

[21] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron, Jitendra Padhye, Shachar Raindel, Mohamad Haj Yahia, and Ming Zhang. 2015. Congestion Control for Large-Scale RDMA Deployments. In *Proceedings of the ACM SIGCOMM 2015 Conference (SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 523–536. https://doi.org/10.1145/2785956.2787484