Data-Driven Emulation of Mobile Access Networks

Ali Safari Khatouni†, Martino Trevisan‡, Danilo Giordano †
†Dalhousie University, ali.safari@dal.ca
‡Politecnico di Torino, first.last@polito.it

Abstract—Network monitoring is fundamental to understand network evolution and behavior. However, monitoring studies have the main limitation of running new experiments when the phenomenon under analysis is over e.g., congestion. To overcome this limitation, network emulation is of vital importance for network testing and research experiments either in wired and mobile networks. When it comes to mobile networks, the variety of technical characteristics, coupled with the opaque network configurations, make realistic network emulation a challenging task.

In this paper, we address this issue leveraging a large scale dataset composed of 500M network latency measurements in Mobile BroadBand networks. By using this dataset, we create 51 different network latency profiles based on the Mobile BroadBand operator, the radio access technology and signal strength. These profiles are then processed to make them compatible with the tc-netem emulation tool. Finally, we show that, despite the limitation of current tc-netem emulation tool, Generative Adversarial Networks are a promising solution used to create realistic temporal emulation.

We believe that this work could be the first step toward a comprehensive data-driven network emulation. For this, we make our profiles and codes available to foster further studies in these directions.

I. INTRODUCTION

Mobile devices dramatically revolutionized the way we interact with the Internet, offering access to the web, video, and messaging applications in mobility with a capacity comparable with the wired network [1]. As a result, nowadays a huge amount of the internet traffic is generated in mobility.

In this scenario, Mobile BroadBand (MBB) network operators have to face severe challenges to catch up with the rapid evolution of the network condition, location, and technologies.

For these reasons monitoring the network became a fundamental task needed to understand the network behavior. By monitoring and measuring, researchers can quantify the evolution of technologies, protocols, setups and website design. Not surprisingly over the years, the research community has put a lot of effort into measuring the benefits of new technologies [2], [3], [4]. Previous studies focused on different aspects, and often addressing specific angles of this complex ecosystem [5], [6], [7], [8], [9], [10].

However, despite monitoring is a perfect solution to profile and understand specific measurements, e.g., network latency, it does not allow per se to run repeatable experiments. Once a particular phenomenon, e.g., congestion, is over, running new experiments on the same condition is impossible. To overcome this limitation, a specific phenomenon can be modeled, and then the model can be used in an emulated environment to run new experiments with the same condition. As a result, new experiments can be performed without the needs of a particular phenomenon happening.

In this work, we address this problem by using an open dataset collected by the MONROE platform[1] The MONROE platform consists of 96 nodes in four European countries. Each node is equipped with MBB subscription that continuously (i.e., every 1 second) measures network latency towards dedicated servers by using ICMP pings. In addition to network latency, the MONROE nodes collect different physical-layer information such as frequency, radio access technology (RAT), signal strength, etc.

By using this dataset, we first create MBB network latency profiles separately for each operator, RAT (i.e., 3G and 4G), signal strength quality, and roaming condition. By leveraging more than 500 M latency measurements from 11 different service providers, in 96 different locations, from 4 different countries, we created 51 latency profiles. Secondly, we feed each profile into the tc-netem Linux tool2 creating an emulated environment where we can run further experiments under specific network latency conditions, e.g., evaluating the users’ experience accessing the network with the poor 4G connection.

Finally, we explore more sophisticated techniques to achieve a realistic temporal emulation. We extract time series from latency measurement, and use them to train Generative Adversarial Neural Network (GAN) [11] models. GANs have been proved to be powerful generative models, able to understand complex distributions behind e.g., images and text. We show that GANs can model the time dimension of latency measurement, generating e.g., peaks, following the same latency distribution of the training data i.e., offering realistic profiles. Despite the promising capabilities of GANs, the de facto standard tool (tc-netem) lacks the capability of using user-defined time series.

Our work poses itself as preliminary work to emphasize the challenges and the importance of latency profiling and emulation in operational mobile networks. We make our tc-netem distribution tables available to the community [12] to allow other researchers to replicate our results and practitioners to benefit from this data.

The rest of this paper is organized as follows. In Section II, we briefly discuss the related work. In Section III, we describe the measurement setup in which the data is collected, while in Section IV we discuss our methodology to generate latency

---

1https://www.monroe-project.eu/access-monroe-platform/

2http://man7.org/linux/man-pages/man8/tc-netem.8.html
profiles. In Section V, we present time series generation using GANs. Finally, Section VI concludes the paper and discusses future work.

II. RELATED WORK

Monitoring performance is fundamental for regulators to ensure transparency and the general quality level of the basic Internet access service [13]. Some of them responded to this issue with ongoing nationwide efforts [14]. Continuous measurements are crucial to understanding the latency and performance of mobile networks. Network latency plays a key role when it comes to the users’ Quality of Experience (QoE) [15], [16]. Balachandran et al. [17] highlight the impact of low-level measurable radio network characteristics on the user QoE during web browsing. A survey of end-to-end delay prediction methods is offered by authors of [18]. They show several methodologies to estimate latency, e.g., *Queueing Network Modelling, System Identification, Time Series Approach*, and *Neural Networks*. Nunes et al. [19] focus on TCP performance in a Mobile Ad Hoc network and propose a machine learning technique called *Experts Framework*. Mandalari et al. [20] show that a roaming user in Europe suffers an additional latency of ~60 ms or more, depending on geographical distance. Safari et al. [21] illustrate accurate mobile network latency prediction is not effective by using the common machine learning algorithms. However, while performing monitoring studies the problem of online privacy should not be overlooked. Many parties can collect and access these kind of data other than researchers. In [22], the authors highlight what are the privacy and ethical issues that arise for users, companies, scientists and governments and presenting the current legislation.

Network emulation is of crucial importance for network testing, and it is a well-studied topic in the literature. The Linux operating system includes *tc-netem*, a tool for this purpose since more than 20 years. Considering emulation of mobile networks, many works aimed at achieving realism with fine-grained models of network devices. For instance, NIST Net [23] is a flexible and powerful emulator for WANs, while authors of [24] exploit advanced modeling methods to build a mobile network emulation environment. More recently, mobile network emulation followed the general trends of research in networking, and has been used to mimic real deployments of e.g., Software Defined Networks [25] or flying vehicles networks [26]. There are few research studies building on a data-driven approach to obtain a realistic emulation. Seminal work [27] proposes a pure trace-driven network emulation that re-creates the observed end-to-end characteristics of a real wireless network. More recently, KauNet [28] provides pattern-based emulation with a higher level of detail, controlling the behavior of each individual packet. In contrast to previous works, we build on a large measurement dataset of operational mobile networks and make our latency profiles available to the community and compatible with the *tc-netem*.

Machine learning has not yet been used in the context of mobile network emulation, but the recent advances in neural networks make it a promising technique for modeling complex phenomena. Recently, GANs have been proposed as generative models. Introduced in 2014 [11], GANs are used for image generation [29] and recognition [30], text-to-photo synthesis [31] and URL generation [32]. In this paper, we show that GANs are able to model MBB network latency, even though the real applicability of this approach is limited by inflexibility of the current *de facto* standard tool, namely *tc-netem*.

III. MEASUREMENTS SETUP

In this section, we briefly describe the measurement setup and the employed dataset.

A. Measurement Infrastructure

Systematic repeatable measurements are crucial for evaluating network performance and assessing the quality experienced by end-users. As such, researchers have built platforms dedicated to broadband networks, e.g., RIPE Atlas 3, CAIDA Ark 4, or PlanetLab 5. In contrast to them, MONROE [33] is a unique platform that enables controlled experimentation with different commercial mobile carriers. It enables users to run custom experiments and to schedule experimental campaigns to collect data from operational MBB and Wi-Fi networks, together with full context information (metadata). It covers 4 countries in Europe (Italy, Norway, Spain, and Sweden) with more than 100 nodes equipped with Ethernet, WIFI and 3G/4G interfaces with commercial subscriptions.

The MONROE platform allows us to access the information about network, time and location of experiments, as well as metadata from the mobile modems, including, e.g., signal strength, RAT, cell identifier for each network provider6. Each node performs a set of predefined experiments (e.g., ping, ping).

Fig. 1: The experimental setup for ping measurement in the MONROE platform.

3https://atlas.ripe.net/
4http://www.caida.org/projects/ark/
5https://www.planet-lab.org/
6The description and detail information about metadata is available on https://github.com/MONROE-PROJECT/UserManual
TABLE I: Statistics on the dataset collected during 2018 by the MONROE platform.

<table>
<thead>
<tr>
<th>Country</th>
<th>Nodes</th>
<th>Operators</th>
<th>Latency samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Italy</td>
<td>27</td>
<td>3</td>
<td>112.2 M</td>
</tr>
<tr>
<td>Norway</td>
<td>18</td>
<td>3</td>
<td>122.4 M</td>
</tr>
<tr>
<td>Spain</td>
<td>19</td>
<td>2</td>
<td>80.4 M</td>
</tr>
<tr>
<td>Sweden</td>
<td>32</td>
<td>3</td>
<td>191.4 M</td>
</tr>
<tr>
<td>Total</td>
<td>96</td>
<td>11</td>
<td>506.4 M</td>
</tr>
</tbody>
</table>

TABLE II: Binning boundaries for 3G and 4G. Values are expressed in $dB$.

<table>
<thead>
<tr>
<th>Quality</th>
<th>3G</th>
<th>4G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poor</td>
<td>$\text{rssi} \leq -100$</td>
<td>$\text{rssi} \leq -85$</td>
</tr>
<tr>
<td>Ordinary</td>
<td>$-100 &lt; \text{rssi} \leq -85$</td>
<td>$-85 &lt; \text{rssi} \leq -75$</td>
</tr>
<tr>
<td>Good</td>
<td>$\text{rssi} &gt; -85$</td>
<td>$\text{rssi} &gt; -75$</td>
</tr>
</tbody>
</table>

passive monitoring, etc.) continuously. Experiment results are collected in the central database, and stored in a Hadoop based cluster. In the following, we provide details about the experimental setup and describe the considered metrics.

Figure 1 shows the experiment setup we consider in this paper. The leftmost element is the MONROE node. It is equipped with three different mobile modems, each equipped with a SIM card of a different operator. Traffic (e.g., ping) through the specified operator and the Internet, toward the selected server – on the rightmost part of the figure. This allows us to measure up to three operators at the same location and time, and using identical device and software, thus, limiting potential sources of bias. This allows one to understand the coverage, and all the time-varying parameters that may impact the latency.

ICMP ping (84 Byte payload) experiments run regularly every second to measure the latency (round-trip time) from nodes toward a dedicated server. The system logs collected data to the central repository after the experiments. Beside latency, we analyze a set of meta-data that characterize the specific context in which each experiments run. Especially, we consider access network parameters: They include parameters from the physical layer such as radio status during the experiment (RSSI, RSRQ, and RSRP).

B. Dataset Description

In this work, we present the data collected on the MONROE platform during the whole year of 2018. It contains measurements from 96 MONROE nodes, measuring 11 operators in 4 European countries, i.e., Italy, Norway, Spain, and Sweden. Nodes operate in a different scenario such as university campuses, urban areas, countryside, aboard buses, and trains. This dataset focuses on latency measurements. In total, we collected latency for about 500 M samples, as detailed in Table I.

IV. EMULATING MOBILE ACCESS NETWORK LATENCY

In this section, we describe the proposed approach to emulate mobile access network latency.

A. Data processing and cleaning

Provided the dataset described in the previous section, we process it to extract clean latency profiles for scenarios that differ for the SIM card operator, RAT, etc. Given the large size of data, all processing is done using Apache Spark for scalability reasons.

We first join the ping samples with physical layer information. As such, separately for each node, we join the two datasets on the time dimension. We then get rid of those time intervals where few or no ping sample is observed due to e.g., network failures or power outages. At this point, we sort ping samples by time and extract sequences of fixed length $l = 60$.\footnote{The node sends a ping every second. As such sequences are 60 seconds long.} We then exclude those sequences where physical layer conditions are not stable due to e.g., intra-RAT or inter-RAT handovers, strong variations in signal strength, etc. This step is necessary to obtain clean data and allows us...
We provide each profile in the form of a custom made public and available to researchers and practitioners [12]. The obtained latency profiles are values of average and standard deviation computed before. We then use the delay tables to enforce an latency requires delay tables to be normalized with respect to such metrics. We first create a tc-netem Linux tool to impose the selected latency profile on the desired network interfaces. We use the obtained latency samples together with the tc-netem tool to impose the selected latency profile on the desired network interfaces. We first create a netem delay distribution table for each profile. Besides, we compute average and standard deviation for further use, as netem requires delay tables to be normalized with respect to such metrics. We then use the delay tables to enforce an latency profile on the selected network interface: the delay table is first copied on the proper system directory, and we then add a netem queuing using the corresponding table, and with the values of average and standard deviation computed before. Initial verification confirms that netem correctly handles our latency table and the resulting latency is perfectly consistent with the expected outcome. The obtained latency profiles are made public and available to researchers and practitioners [12]. We provide each profile in the form of a custom netem delay distribution. We also implemented simple convenience scripts that allow the user to easily install such profiles in the proper system directory and start delay emulation on a selected interface.

V. EXPERIMENTS USING GENERATIVE ADVERSARIAL NETWORKS

Our approach based on the mere sample distribution has drawbacks that limit the realism of the emulation. Whereas the overall distribution of the emulated and real samples is the same, the approach does not model the time dimension. As such, there is no temporal correlation among the generated samples, and we cannot model time-based events, such as a temporary increase in the latency caused by a short-term network failure.

To overcome such limitation, we propose the use of GANs, that have been proven able to model (and generate) complex data such as images [29] and network data [32]. A GAN is composed of a generator and a discriminator model. Provided with random noise and feedback from the discriminator, the generator task is to produce samples that are similar to the real data, trying to make up artificial samples that cannot be distinguished by the discriminator. The discriminator task instead is to distinguish between real samples in the training set and the artificial ones produced by the generator. As the two models compete to win their adversarial tasks, artificial samples become more and more realistic, whereas the discriminator becomes robust to noise.

In our experiments, we train a GAN for each latency profiles. Figure 4 sketches the building blocks of our GAN during training. We feed them with sequences of length \( l = 60 \) of latency samples. In the original data, Pings are performed each second, and we remove all sequences with missing data to obtain a fully-formed time series. We tested different architectures for both generator and discriminator models, and we obtained the best results using two stacked LSTMs [34] in both. Examples of real and generated sequences are depicted in Figure 3. The two generated sequences (right) correctly show temporal correlation across samples, and we can identify short-term peaks in which latency increases. Moreover, the generated samples have absolute values consistent with the original, as shown in Figure 5 for two examples. Real and generated distributions mostly overlap, even if they do not perfectly match. Indeed, Kolmogorov–Smirnov tests provide KS statistics in the order of \([0.1, 0.4]\) for the considered profiles.

Our results show that using GANs to emulate latency profiles is a promising approach, but technical issues limit the applicability to real cases. In particular, widespread network emulation tools (e.g., netem) cannot emulate time-series based latency profiles, and as such require the design and implementation of ad-hoc tools.

VI. CONCLUSION AND FUTURE WORK

Mobile devices have become one of the main tool used for accessing the Internet and web services. However, the diversity
of devices, technologies, and mobility makes it highly dynamic and utterly difficult to predict. As such, network emulation is of fundamental importance for testing new applications, new protocols, or performing research experiments.

In this paper, we exploit a large and one-of-the-kind dataset collected in 4 countries from 11 operational operators. We presented two different approaches for emulating MBB network latency. Firstly, we obtained 51 latency profiles, for different operators, RAT, and signal quality. They can be used in tc-netem as distribution tables and they are available to the community. Secondly, we explored to use GANs to generate realistic latency time series and found that they fit for the purpose.

This work is the first step toward a comprehensive data-driven network emulation. Firstly, we created latency profiles, as our experiments do not yet include bandwidth and packet loss measurements. Secondly, GANs are promising as generative models to achieve high realism, but a big effort is required to fit them in the current network emulation tools.

ACKNOWLEDGEMENTS

The research leading to these results has been funded by the European Union’s Horizon 2020 research and innovation program under grant agreement No. 644399 (MONROE) and the Smart-Data@Polito center for Big Data technologies.

REFERENCES

habits,” ACM Transactions on the Web (TWEB), vol. 12, no. 4, p. 24, 2018.


[4] O. Fagboun, “Comparative studies on 3g, 4g and 5g wireless technology,” vol. 9, pp. 133–139, 01 2014.


