Reducing Consumed Data Volume in Bandwidth Measurements via a Machine Learning Approach

Christian Maier*, Peter Dorfinger*, Jia Lei Du*, Sven Gschweitl†, Johannes Lusak†
*Salzburg Research Forschungsgesellschaft mbH, †alladin-IT GmbH
{christian.maier, peter.dorfinger, jia.du}@salzburgresearch.at, {sg, jl}@alladin.at

***Abstract*—Measurements that determine the available download and upload bandwidth of an end-user Internet connection (so-called speed tests) are typically performed by maximizing the utilization of the connection for a fixed time interval. Especially in broadband connections, such tests consume a huge amount of data volume during their execution. As a result, only a few tests can be performed per month on mobile connections with limited data volumes, since otherwise a significant portion of the volume is used for tests or additional costs are incurred. To reduce the required average data volume of these tests, we present a novel approach with a dynamic test duration based on a machine learning model. We train this model via a supervised learning process, using the recorded data of real speed tests executed by end-users in cellular 4G networks. The evaluation of the resulting method suggests that the amount of saved data volume is significant, while the deviation of the determined bandwidth (compared to a usual test with fixed duration) is negligible.**

I. Introduction

The concept of bandwidth, defined as the amount of data that a link or path can deliver per unit of time, is central to data communication. It often directly relates to the performance of applications and hence has a strong impact on the end-user Quality of Experience (QoE). This bandwidth is influenced by various factors, such as the used technology, the present infrastructure and environment or the number of concurrent users. In the case of Internet connections, it is also limited by a bound specified in the contract concluded between the customer and the Internet service provider. In particular mobile Internet connections provide an actual available bandwidth which varies greatly and often does not reach this bound. Especially if the performance of applications suffers noticeable on this gap between the promised and actual bandwidth, users are interested in determining the latter. The methods to achieve this can roughly be divided into two different categories, depending on whether they actually measure or estimate this quantity. Obviously, the effort for estimations is lower, whereas measurements offer (in general) higher accuracy.

There are various tools for available bandwidth estimations, such as *Pathload* [7], *Yaz* [16] and *Delphi* [14]. All of them are (more or less) carried out by producing some small amount of data transfer between the client and the server and by estimating the unused bandwidth from the occurring behavior.

In this paper, we will deal with measurements of the available bandwidth of an end-user Internet connection. These are usually done via a specified speed test, which determines the available download and upload data rate, as well as some other parameters (e.g. latency) of the used connection. Examples are *NetPerf* [8], *IPerf* [6] and the *RTR Multithreaded Broadband Test* (RMBT) [15]. The *Body of European Regulators for Electronic Communications* (BEREC) specifies in [2] a reference measurement tool for the monitoring of Quality of Service (QoS) parameters of Internet Access Services. This specification, which we regard as a requirement to be fulfilled by the methods considered in this paper, states that the available bandwidth must be measured over TCP by fully utilizing the connection for a fixed time interval. This results in a considerable amount of consumed data, as well as a noticeable influence on the performance of other applications. The former is in particular a huge problem for customers which have contracts with limited data amount per month. One of the determining factors for the amount of consumed data is the duration of the speed test, which is usually a predefined constant. Since this constant is in principle arbitrarily selected, the consumed data volume can be reduced by simply choosing a shorter duration. This may change the result of the test significantly, in particular if there is a huge dynamic in the available data rate of the connection to be measured. The latter behavior occurs especially in cellular networks.

In practice, there are commonly accepted values for the duration of a speed test which lead to meaningful results. We will not deal with the (difficult) question how to determine such a duration. We just want to mention that the result of a single performed speed test should only be used to obtain information about the bandwidth in the respective measurement interval. The point of view in this paper is the following: If one accepts that a speed test with a fixed predefined duration gives a meaningful value, is it possible to specify a new test (using the same method as the original one) with a shorter duration, which gives a result that is not significantly different than the result from the original test and consumes a substantially lower amount of data volume. In the present paper we will investigate a principally shorter duration of the test and a test duration which is determined dynamically for each execution. These two approaches will be compared in terms of their accuracy of the result and the amount of saved data volume.

The determination of a dynamic duration which does not change the result of the test considerably will be done via

a machine learning (ML) model. The use of such techniques in network measurements has increased significantly in recent years (see below). This is on the one hand due to the high dimensionality of network data and on the other hand due to the success of ML (and especially deep learning) in fields like image and signal processing. From the various different ML models we have chosen a simple feed forward artificial neural network for our approach.

The rest of the paper is organized as follows: Sec. II gives an overview on related work. In Sec. III we describe some details of a single execution of the speed test we are investigating. Sec. IV deals with a principally shorter test duration. In Sec. V we specify a test with a dynamic duration determined by a neural network. The evaluation and comparison of both approaches is done in Sec. VI. Finally, Sec. VII concludes the paper and gives an outlook on future work.

II. Related work

One should take care that the notion of available bandwidth is used interchangeably in the literature. In contrast to our point of view, which defines the available bandwidth as the actual achievable data rate that a connection offers to the user, the term is frequently also used to refer to the unused bandwidth of a connection. Note that the latter determines the former, if the current traffic load is known. Additionally, it is important to differentiate between mobile and non-mobile estimations and measurements, since the bandwidth in mobile networks is subject to much greater dynamics.

The topic of available bandwidth estimation (i.e. the estimation of the maximum unused bandwidth) and measurement is largely extended in the literature. To list only a few of them, the survey [13] gives an overview on the underlying techniques and methodologies of some estimation tools. The difficulty of accurate bandwidth measurements is depicted in [10]. In [17] authors investigate test shortening of non-mobile *IPerf* bandwidth measurements. A bandwidth estimation method for mobile networks is presented in [11].

There is a rapidly growing amount of literature which investigates the application of ML to networks and network measurement problems. In [3] the authors jointly present the application of such techniques in various key areas of networking across different network technologies and also provide a primer on ML in general. The focus of [5] is on the comparison of different ML models via their performance in the analysis of network measurements (like detection of network attacks, anomaly detection and QoE prediction). The survey [12] investigates the application of ML techniques to IP traffic classification. In [1] authors use an approach based on ML for QoE modeling. Finally, [4] deals with the application of ML in cyber-security-analytics.

III. Methodology of Bandwidth Measurements

This section provides a brief description of a test method that determines the available download or upload bandwidth of an end-user Internet connection (it is the one of RMBT). In order to keep the exposition simple, we will restrict to the download case. The upload case works almost in the same way. Besides, we only describe those parts of the test which are relevant for the further discussion. For more details we refer to [15].

Such a test has a nominal duration t and a fixed number n of used TCP connections. The nominal duration could either be a predefined constant (which is the case in state-of-the-art implementations), or could be determined dynamically for each execution of the test. We do not specify this for now. Before the actual test begins, a pretest is performed. During this pretest, the client (executing the test) opens n parallel TCP connections to a single test server, which are then used for the transmission of data chunks of growing size. This ensures that the connections are in an active state and also determines some further necessary test parameters (see below). Afterwards, the actual test starts with the server continuously sending data streams, one via each TCP connection, to the client. These data streams consist of chunks of randomly generated data with high entropy. The size of the data chunks varies with a specific distribution, whose parameters are among the test parameters determined during the pretest. For each $k \in \{1, \dots, n\}$, the client records the arrival time $t_k^{(j)}$ of the last bit of the j-th data chunk which was sent on TCP connection k, as well as the total amount $b_k^{(j)}$ of data received on this connection up to this moment. The chunks and the arisen data amount of the pretest are not considered in this recording process.

After time t, the test server stops sending further chunks on all connections. The client waits until the last sent data chunks are completely transmitted. Let m_k denote the total number of data chunks which were sent on connection k during the actual test execution. The sequence $s = (t_k^{(1)}, b_k^{(1)}), (t_k^{(2)}, b_k^{(2)}), \dots, (t_k^{(m_k)}, b_k^{(m_k)})$ describes the course of the test on TCP connection k. To determine the test result from the n recorded courses, let $t^* = min\{t_1^{(m_1)}, \dots, t_n^{(m_n)}\}$ denote the time when the first transmission was finished. Then an approximation b_k for the amount of data received over TCP connection k from the beginning of the test until time t^* is given as

$$b_k = b_k^{(m_k-1)} + \frac{b_k^{(m_k)} - b_k^{(m_k-1)}}{t_k^{(m_k)} - t_k^{(m_k-1)}}(t^* - t_k^{(m_k-1)})$$

and $R = (b_1 + \dots + b_n)/t^*$ is defined to be the bandwidth determined by the measurement method. The total data amount consumed during the test execution is given as $V = \sum_{k=1}^{n} b_k^{(m_k)}$. This value depends on the nominal duration and on the characteristics of the connection to be measured.

The test records can also be used to calculate the result R_τ and the consumed data amount V_τ that a test execution with a shorter nominal duration τ would have yielded. This is done by removing all tuples $(t_k^{(j)}, b_k^{(j)})$ in the sequence s that contain information about data chunks sent after time τ and a subsequent repetition of the above calculation. The absolute percentage deviation of R_τ from R is then given as $d_\tau = 100 \cdot |1 - R_\tau/R|$. By replacing R with V and R_τ with V_τ in the right-hand-side of this formula, one obtains the percentage

s_τ of saved date volume that this reduction of the nominal duration from t to τ would cause. These two quantities (d_τ and s_τ) will serve as metrics for evaluating the effect of test shortenings. In general, the percentage deviation d_τ will be small if the courses on all threads are approximately linear. Otherwise, there may be a huge deviation.

IV. Fixed Test Duration

The simplest approach to reduce the consumed data volume of a state-of-the-art implementation of a speed test with fixed predefined nominal duration T (based on the method specified in the last section) is an implementation with a shorter fixed duration τ. This shortening may lead to a large deviation d_τ between the result R_τ of the new test and the result R of the original one. One suspects that this is particularly the case when there is a big difference between τ and T, which, on the other hand, would also result in a huge amount of reduced consumed date volume. This leads to the question if there is a $\tau \in (0, T)$ such that, on the one hand, the average consumed date volume of the test is reduced considerably and on the other hand, the test result is not changed significantly. To make the latter precise, we say that the result is changed significantly, if $d_\tau > 10\%$. Here, we have chosen the bound of 10% because it seems to be a value which end-users may accept as inaccuracy in the result of a speed test.

To get an answer to this question, we will consider a huge database which contains the recorded courses of real executions of this speed test with fixed duration T. The average amount of reduced data volume $s(\tau)$, which is the mean of the values s_τ for all tests in the database, and the proportion $p(\tau)$ of tests in the database with significant deviation d_τ can then be considered as functions depending on τ. Both functions tend to zero if τ tends to T. The existence of a $\tau \in (0, T)$ such that $p(\tau)$ is sufficiently small and $s(\tau)$ is substantial would yield a positive answer to the question stated above.

V. Dynamic test duration

Lets consider a speed test (measuring either download or upload throughput) as presented in detail in Sec. III with a fixed nominal duration T. The data rate determined by this test (on a single execution) is denoted with R. Our goal is to specify a new test with a dynamic nominal duration t (satisfying $t \leq T$) which, on the one hand, gives a result R' that only slightly deviates from the data rate R and, on the other hand, reduces the average consumed data amount extensively. To keep the exposition simple, the set of possible values for t is chosen to consist just of two elements. One of these elements is T. We denote the second element with τ (which satisfies $\tau < T$). This means that the new test either stops when the last data chunk, which was sent before time τ, arrives (we will henceforth call this a short test) or the test proceeds as the original one. The decision, which value t takes (that is, which of the just mentioned cases occurs), is made on the basis of a calculation performed by a trained artificial neural network. To be more specific, this neural network (which is a simple multi-layer perceptron) has two output values, which are real numbers, denoted y_1 and y_2. It is trained in a way such that a high value of y_1 is an indicator that a test execution with nominal duration τ would yield a result that does not deviate from R considerably. On the other hand, a high value of y_2 indicates the opposite, i.e. that there is a huge difference between R and the data rate that a short test would determine. Hence the natural decision rule (determining t) using these two output values is the following:

> If $y_1 \geq y_2 + \lambda$, then $t = \tau$ (with $\lambda \in \mathbb{R}$).
> Otherwise, $t = T$.

Here we introduced a parameter λ, which controls how circumspectly the rule decides for a short test: The larger λ is, the higher the indicator y_1 must be (compared to y_2) in order to make such a decision.

The calculation of the neural network and the subsequent decision for $t = \tau$ or $t = T$ is done at the aforementioned time of arrival of the last chunk sent before time τ. As input, the neural network obtains the course of the data rate from the beginning of the test up to this moment. Since the calculation has to be done on mobile end devices, one needs to keep in mind the required computing effort, which depends on the actual structure of the used neural network. To train this neural network, the obvious approach is via a supervised learning process using the recorded data of real executions of the original speed test (with fixed nominal duration T) on mobile devices in cellular 4G networks. These records are also used to determine the parameter λ, which completes the specification of the proposed method. Next, we will describe this in more detail. The subsequent evaluation will be done by calculating the test proportion with significant absolute deviation between the result of the test with dynamic duration and the result of the test with fixed duration (where the deviation is again called significant if it is greater than 10%.), as well as the amount of saved data volume.

A. Inputs of the Neural Network

The neural network gets as input the course of the data rate in the time interval $[0, \tau]$ of each thread $k = 1, \ldots, n$. This course can be calculated form the sequence $s = (t_k^{(1)}, b_k^{(1)}), (t_k^{(2)}, b_k^{(2)}), \ldots, (t_k^{(m_k)}, b_k^{(m_k)})$ by forming the fractions $b_k^{(j)}/t_k^{(j)}$. Usually, the length of this sequence varies from one test execution to another (and also from one TCP connection to another). Since the number of actual input values of the neural network has to be constant, a preprocess is necessary. Therefore, one chooses a resolution $\delta > 0$ in a way such that τ is an integer multiple of δ. The inputs delivered to the neural network are then the approximations $x_k^{(i)}$ of the data rate on connection k in the time intervals $[(i-1)\delta, i\delta]$ (with $i = 1, \ldots, \tau/\delta$), which are calculated from the fractions $b_k^{(j)}/t_k^{(j)}$ by a linear approximation. This results in $n\tau/\delta$ input values.

B. Structure of the Neural Network

The considered neural network is a feed-forward neural network with one input layer, several hidden layers and one

output layer (i.e. a multi-layer perceptron). The discussion of the preceding subsection leads to $n\tau/\delta$ input nodes. As activation functions we selected exponential linear units (given as $x \mapsto e^x - 1$ for $x \leq 0$ and $x \mapsto x$ for $x > 0$) for all neurons in the hidden layers and linear functions for the nodes at the output layer. This is a common choice for classification tasks.

C. Training Process

To train a neural network such that its output values are indicators for or against a short test duration, we will use a database containing the recorded courses of real speed test executions with fixed nominal duration T. For a supervised learning process, it is necessary to label this data. This is done in the following way: Using the procedure described in Sec. III we calculate for each record in the database the absolute percentage deviation d_τ in the result that a test execution with nominal duration τ would yield. The data then splits into two classes: Class A if d_τ is smaller than 5%, and class B otherwise. Here, we have chosen the class border so that a (supposed) poor accuracy of the neural network in the classification of tests close this border has no strong influence in the evaluation, where we will consider deviations as significant if they are greater than 10%. After the data are labeled in this way, the neural network is trained to determine from the inputs $x_k^{(i)}$ the class which the test belongs to. That is, the neural network should learn to determine from the course of the data rate in the first τ seconds of the test execution, if the deviation of R_τ from R is above or below 5%. Note that R_τ can be approximately calculated from the values $x_k^{(i)}$. The accuracy which the neural network reaches in its classification task will show how much information the values $x_k^{(i)}$ contain in order to determine R. Not all tests in the available database should be used for the training. With some of them, one should check if no overfitting occurs. This data is called test data. Usually the ratio of training to test data is about $9:1$.

D. Determination of the Control Parameter

The test data can also be used to determine the control parameter λ for the proposed speed test. This can be done in the following way: One selects a small percentage ρ such that a significant deviation of the result in not more than ρ percent of the tests is acceptable (for both the provider of the test and the customer). This can be done, since this portion tends to 0 with ascending λ. However, it is not self-evident that this bound on the proportion of tests with significant deviation then also holds for a set of tests different from the test data. This has to be checked in a subsequent evaluation of the method with a database containing exclusively tests not used for the training of the neural network or the determination of λ.

E. Computing Effort

The computing effort of the neural network to decide whether a test belongs to class A or B consists essentially of matrix multiplications, additions and the multiple application of the activation functions. Especially with optimized frameworks (like TensorFlow Lite) and by using a small number of hidden layers and neurons, this can be done on mobile end-devices in real time. On the other hand, the training process takes much longer and requires more computational power. However, this can be done in advance on a suitable machine, and hence does not influence the applicability of the test.

F. Summary of the Method

To recapitulate, the proposed test with dynamic duration starts as the usual (fixed duration) test. After the last data chunk which was sent before time τ arrived completely at the client, the courses of the data rate on each TCP connection from the beginning of the test up to this moment is sampled with resolution δ. This sampling process yields the inputs for the neural network, which was trained before to decide on the basis of these values, whether the test with nominal duration τ gives a result which is not significantly different from the result, that a test with nominal duration T would produce. A possibly low confidence in the result of the neural network can be counteracted by selecting a high parameter λ. After the decision for or against a short test is made, the test either stops or the server continuous with sending data chunks until time T.

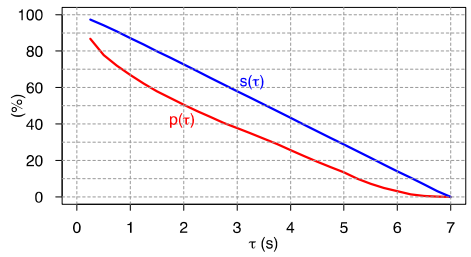
VI. Evaluation

For the evaluation we considered a speed test implementation with a fixed nominal duration of $T = 7s$ and $n = 3$ parallel TCP connections, together with a database which contains 37 935 download records and 37 861 upload records of executions of this test. The distinction between download and upload tests is necessary, since one expects a difference in the behavior of them. All the tests were performed by end-users via RMBT in the period between January 2015 and April 2018 on their personal mobile Android or iOS devices in 4G cellular networks. The total number of different device types was 1564. The average measured bandwidth was 43.8MBit/s for download tests and 17.9MBit/s for upload tests.

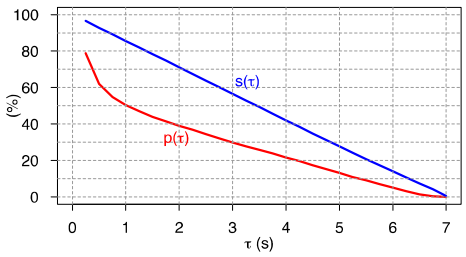
First of all we investigate the percentage deviation in the result and the amount of saved data volume that a general shortening of the duration T would cause. As explained in Sec. IV, this information is contained in the functions $s(\tau)$ and $p(\tau)$, which both depend on the nominal duration τ of the short test. The course of these functions (with respect to the database in question) is shown in Fig. 1. Some special values are shown in Tab. I. It is not surprising that the amount of saved data volume $s(\tau)$ depends linearly on τ and that both functions are monotonously decreasing (which obviously is

TABLE I
Special values of $s(\tau)$ and $p(\tau)$

		Nominal test duration τ		
		4s	5s	6s
Download	$s(\tau)$	43.95%	28.74%	14.13%
	$p(\tau)$	25.93%	13.50%	3.09%
Upload	$s(\tau)$	42.67%	27.74%	14.02%
	$p(\tau)$	22.00%	13.20%	5.06%



(a) Download



(b) Upload

Fig. 1. The amount of saved data volume $s(\tau)$ and the proportion of tests $p(\tau)$, where the deviation of the result of a test with fixed nominal duration τ from the result of a test with fixed nominal duration $T = 7s$ would be significant (i.e. greater than 10%).

TABLE II
EVALUATION OF THE SPEED TEST WITH DYNAMIC DURATION

	Download	Upload
Number of speed test records used for the evaluation	37 935	37 861
Number of tests executed as short ones by the proposed method	11 851	13 162
Number of tests with significant absolute deviation	510	375
Proportion of tests with significant absolute deviation	1.34%	0.99%
Overall consumed data volume of the original speed test with fixed duration	1453 GB	586 GB
Overall consumed data volume of the proposed speed test with dynamic duration	1160 GB	443 GB
Percentage of reduced data volume	20.18%	24.49%

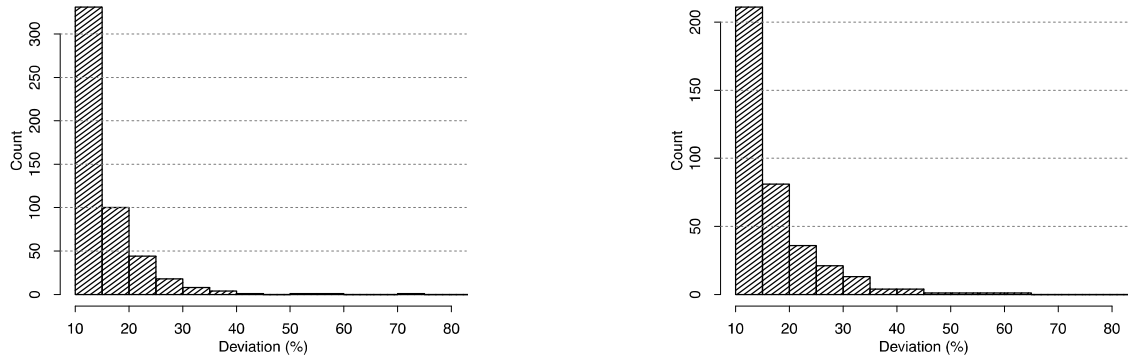
even a general fact for $s(\tau)$). As one can see from these graphs, the amount of tests $p(\tau)$ with significant deviation between R_τ and R is, even for values of τ which are close to $T = 7s$, not negligible. Hence the question stated in Sec. IV, i.e. if there is a $\tau \in (0, T)$ such that, on the one hand, the average consumed date volume of the test is reduced considerably and on the other hand, the test result is not changed significantly, must be answered negatively.

For the test with dynamic duration, we selected a duration of $\tau = 4s$ for a short test and a resolution of $\delta = 0.25s$ for the sampling of the courses. This leads to $n\tau/\delta = 48$ input values for the neural network. The number of hidden layers was set to 2, with 32 neurons in the first and 16 neurons in the second hidden layer. The training process of this ML model was done with another database consisting of test records of the same speed test implementation. This database is disjoint from the database used for the evaluation, i.e. no test in the one database is contained in the other. Due to the differing behaviour already mentioned above (which can now even be observed in Fig. 1), a neural network has to be trained separately for the download and the upload test. The data was further divided into actual training data and test data (to check if no overfitting behavior occurs and to determine λ). To be precise, we used 375 000 speed test records for the training process in each case, 26 601 records as test data in the download case and 25 790 records as test data in the upload case.

The neural networks were implemented in Python using the TensorFlow Framework. Weights were initialized randomly and all bias values were initially set to 0. Optimization was done with the Adam Optimization Algorithm [9], which is one of the state-of-the-art methods. The number of training epochs was set to 1000 with a batch size of 50. The neural networks reached a maximal accuracy of about 76% in the classification of download tests and about 78% in the upload case. This moderate accuracy is not surprising, since the test course of the first four seconds contains information about the necessary test length, but does not fully determine it. After that, we selected the necessary parameter λ in a way such that the proportion of tests with significant deviation between the result of the test with dynamic duration and the result of the test with fixed duration is exactly 1% (i.e. the value of ρ was selected to be 1%). The resulting values are $\lambda = 0.9$ for the download test process and $\lambda = 1.5$ for the upload process.

The subsequent evaluation was again done with the 37 935 download records and the 37 861 upload records which were already used for the evaluation of the general test shortening. Also the principle was similar: We calculated the test proportion with significant deviation between the result of the test with dynamic duration and the result of the test with fixed duration, as well as the amount of saved data volume. Tab. II shows the resulting values. In contrast to the general test shortening, the proportion of tests with significant deviation is small, while the amount of saved data volume is considerable.

Fig. 2 shows a histogram of the percentage deviations of those tests where this value is above 10%. As one can observe, most of them are just slightly above 10%. From the values in Tab. I and Tab. II it can be deduced that a percentage of 31.24% of short download tests results in a saving of 45.92% in relation to the total possible saving (i.e. if all tests would be performed with a nominal duration of four seconds). This suggests that a short measurement is especially possible for tests with high data volume. A more detailed investigation provided the following results: The average consumed data volume of all download tests is 38.31 MB, whereas the average



(a) Download (b) Upload

Fig. 2. Histogram of deviations which are significant (i.e. greater than 10%).

consumed data volume of all download tests where a short measurement is possible is 56.76 MB. For upload tests the corresponding values are 15.49 MB and 26.21 MB.

VII. Conclusion

In this paper we investigated two different approaches to reduce the consumed data volume in tests which determine the available download and upload data rate of an end-user Internet connection. The first approach is simply a general shortening of the test duration. The second (novel) approach is a test with a dynamic test duration determined through a trained artificial neural network. The evaluation showed that, on the one hand, a principally test shortening often leads to huge deviations in the result, while the test result is only changed insignificantly in the method with dynamic duration (compared to the result of a test with a fixed duration). On the other hand, the amount of saved data volume of the test with dynamic duration is still considerable.

There are various possible approaches for improvements or extensions of the proposed method. For example, until now only the test course of the first τ seconds was used as input for the neural network. This could be supplemented by further information, like the measured signal strength or GPS data, which may lead to a higher accuracy in the classification task of the neural network. Secondly, up to now the set of possible values for the dynamic test duration consists just of two elements. Here, an extension to a larger number of possible duration times could be examined. This may further reduce the volume of consumed data.

Finally, the general philosophy of this paper, to reduce the effort for measurements by ML techniques, may has other applications than just bandwidth measurements.

References

[1] Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, Ion Stoica, and Hui Zhang. Developing a Predictive Model of Quality of Experience for Internet Video. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 339–350, 2013.

[2] BEREC. Net neutrality measurement tool specification. *BoR (17) 179*, 2017.

[3] Raouf Boutaba, Mohammad A Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano, and Oscar M Caicedo. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, 9(1):1–99, 2018.

[4] Anna L. Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, 2016.

[5] Pedro Casas. On the analysis of network measurements through machine learning: The power of the crowd. In *Network Traffic Measurement and Analysis Conference (TMA)*, pages 1–8. IEEE, 2018.

[6] Chung-Hsing Hsu and Ulrich Kremer. Iperf: A framework for automatic construction of performance prediction models. In *Workshop on Profile and Feedback-Directed Compilation (PFDC)*. Citeseer, 1998.

[7] Manish Jain and Constantinos Dovrolis. Pathload: A measurement tool for end-to-end available bandwidth. In *Proceedings of Passive and Active Measurements (PAM) Workshop*. Citeseer, 2002.

[8] Rick Jones et al. Netperf: a network performance benchmark. *Information Networks Division, Hewlett-Packard Company*, 1996.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *preprint arXiv:1412.6980*, 2014.

[10] Kevin Lai and Mary Baker. Measuring bandwidth. In *INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*, volume 1, pages 235–245. IEEE, 1999.

[11] Zoltán Móczár and Sándor Molnár. Bandwidth estimation in mobile networks by busy period detection. In *25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, pages 1354–1358. IEEE, 2014.

[12] Thuy TT Nguyen and Grenville J Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys and Tutorials*, 10(1-4):56–76, 2008.

[13] Ravi Prasad, Constantinos Dovrolis, Margaret Murray, and KC Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *IEEE network*, 17(6):27–35, 2003.

[14] Vinay Joseph Ribeiro, Mark J. Coates, Rudolf H. Riedi, Shriram Sarvotham, Brent Hendricks, and Richard G. Baraniuk. Multifractal cross-traffic estimation. In *ITC Conference on IP Traffic, Modeling and Management*, 2000.

[15] Christoph Soelder, Leonhard Wimmer, Dietmar Zlabinger, Ulrich Latzenhofer, Ursula Prinzl, Philipp Sandner, Lukasz Budryk, Ulrich Liener, and Thomas Schreiber. Rtr multithreaded broadband test (rmbt): Specification. *http://netztest.at/doc/*, 2017.

[16] Joel Sommers, Paul Barford, and Walter Willinger. A proposed framework for calibration of available bandwidth estimation tools. In *11th IEEE Symposium on Computers and Communications (ISCC)*, pages 709–718. IEEE, 2006.

[17] Ajay Tirumala, Les Cottrell, and Tom Dunigan. Measuring end-to-end bandwidth with iperf using web100. In *Web100, Proc. of Passive and Active Measurement Workshop*. Citeseer, 2003.