An Innovative Machine Learning Approach to Improve MPTCP Performance

Fabio Silva
Student Member, IEEE
School of Electronic Engineering
Dublin City University, Ireland
fabio.silva3@mail.dcu.ie

Mohammed Amine Togou
Member, IEEE
School of Electronic Engineering
Dublin City University, Ireland
mohammedamine.togou@dcu.ie

Gabriel-Miro Muntean
Senior Member, IEEE
School of Electronic Engineering
Dublin City University, Ireland
gabriel.muntean@dcu.ie

***Abstract*—This paper presents, describes and evaluates the Machine Learning Performance Monitor (MLPM), an innovative Machine Learning (ML) approach to forecast and extrapolate the performance of several network features (e.g., latency, throughput) in a Multipath TCP (MPTCP) subflow pool. MLPM uses linear regression to predict the performance of network features along with Artificial Neural Network linear classifier to choose the best subflow (i.e., network path) capable of delivering the best performance to a given set of the network features. Results show that MLPM delivers better performance in terms of throughput and latency compared to existing schemes as it improves the MPTCP scheduler performance.**

***Keywords*—Linear regression, Machine Learning, Multipath TCP, supervised learning, neural network**

I. Introduction

When Marshall McLuhan [1], in 1964, anticipated the idea of an interconnected society in a "*Global Village*" or, even earlier, when Nikola Tesla [2], in 1926, made his amazingly accurate predictions about the use of *wireless technology* and the Earth converted to a "*huge brain*", most certainly they were visionaries but even their wildest dreams were overcome by the Internet growth and its widespread use.

According to [3], the number of Internet users has grown from 16 million users 0.4% of the world population (in December 1995) to approximately 4.54 billion users or 58.8% of the world population (in June 2019), and the demand for Internet services increased in size and diversity. For instance, the video share of global traffic, as stated by [4], [5] went from roughly 16% in 2016 to an estimated 80% in 2020, meaning 320 exabytes per month. Additionally, technologies such as Virtual Reality (VR), now supporting a wide spectrum of applications in the military [6], education [7] and healthcare [8], anticipates an escalation in such demands.

Several technologies have been proposed to address such stringent demands. 5G technology can offer 300 Mbps in dense areas and 1Gbps/500Mbps (download/upload) in indoor ultra-high broadband networks [9]. Moreover, emerging technologies, such as Software-Defined Networking (SDN) [10], Network Functions Virtualisation (NFV) and networking slicing [11] can help dynamize how the infrastructure is deployed, scaled and distributed across multiple locations.

This paper explores one of these technologies, MPTCP, which extends the Transmission Control Protocol (TCP) typical functionalities and allows communication to be established over multiple paths concurrently and transparently [12]. This implies that when the application layer requests connections from the transport layer, MPTCP manages these demands and establishes one or more connections through the network layer with no impact on the Open Systems Interconnection (OSI) model.

These concurrent TCP sessions working in *parallel* are known as MPTCP *subflows* [12], [13], [14]. The scheduler responsible for managing the subflows operation implements several algorithms (e.g., round-robin [12], RTT aware [15], congestion-based) commonly found in typical TCP implementations.

To enhance MPTCP's performance, we propose MLPM, an innovative machine learning-based approach that forecasts the performance of the subflows' features (e.g., latency, throughput) and classifies the subflow pool accordingly. To do so, MLPM deploys a linear regression technique to analyse and extrapolate the features' performance of each subflow and uses an Artificial Neural Network (ANN) linear classifier to arrange the different subflows in order to achieve improved performance. MLPM can be of great help to applications requiring, for example, latency management, network congestion avoidance and special traffic prioritisation.

The rest of this paper is organised as follows. Section II surveys some related work. Section III describes MLPM's architecture and machine learning approach. Section IV depicts the testbed implementation and assessment results. Finally, Section V concludes the paper and suggests future work.

II. Related works

In the next subsections, we first give an overview of MPTCP and then present some of the Machine Learning (ML) approaches that were designed for networking.

A. MPTCP

As described in [12], [14], MPTCP is used to enable data transport over multiple paths (subflows) concurrently and transparently, and ensures that the basic operation of TCP sessions are used to establish the subflows. This policy is important to guarantee that subflows behave as regular TCP sessions and, consequently, preserve network compatibility. Also, by keeping the interaction between OSI layers unaltered, MPTCP can be seamlessly integrated into the OSI model and suffer from no interference of middleboxes (e.g., firewalls, routers).

Fig. 1 illustrates how the expansion of the TCP stack [16] helps MPTCP to keep the compatibility with the upper layers (application layers) and the lower layers (network layers) of the OSI model.

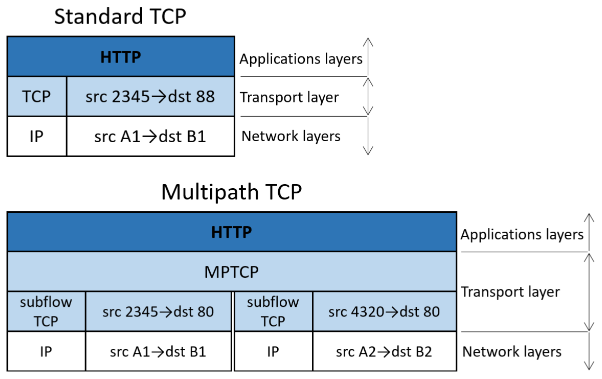

Fig. 1: Standard TCP and Multipath TCP (MPTCP)

This technology is extensively explored by the the Internet Engineering Task Force (IETF) in [12] covering several aspects such as architectural guidelines for MPTCP development and use cases to support the community.

Differently of a single-path transport protocol (which can be undoubtedly impacted by the conditions found on that pathway), in the MPTCP technology the flow of information can be distributed or shifted between the available subflows with better delivery conditions (e.g., lower packet loss and/or latency, higher throughput).

By doing so, MPTCP can mitigate some of the problems found in single-path technologies. However, a multipath transport protocol poses quite a challenge concerning optimisation of the intrinsic resources of multipath transmissions and the proportional complexity increment coming from the number of paths available.

To mention just multipath-related congestion control alternatives (one of the most important components for MPTCP design), [17] presents an extensive study of congestion control mechanism approaches and exemplifies the number of distinct algorithms and different levels of complexity for this topic. Those algorithms' metrics include, but are not limited to, Round-Trip Time (RTT), throughput, TCP window size, loss rate, bandwidth and other features.

Similarly, [18] evaluates some MPTCP scheduling algorithms - also one of the most important components for MPTCP design. Working in a complementary manner to congestion control, the scheduling algorithms distribute packets on multiple subflows based on their congestion window size. In [18] the author also cover the impact of different metrics on the performance of widely deployed scheduling algorithms. However, no approach uses any type of forecasting scheme as proposed in this paper.

B. ML *in networking*

ML techniques are used in a broad range of applications and areas and networking technologies are not an exception. ML can support several types of networking applications, e.g., route measurement, traffic prediction and job scheduling [19]. Several distinct approaches can be applied in all distinct levels of network infrastructure or architecture: from how infrastructure should be deployed (e.g., regional allocation based on traffic prediction) to low-level applications (e.g., throughput prediction).

Another technique presented in [20] offers a superior throughput prediction scheme to help select the best initial bitrate and the midstream adaptation settings for video streaming. For that, the authors examined several prediction models (such as harmonic mean of past measurements and a time series technique known as auto-regressive technique), and proposed an offline clustering approach that considers measurements of previous sessions to identify current sessions that are likely to experience similar throughput patterns.

In [21], the authors propose a new congestion control architecture based on live experimental evidence by sending packets for a period of time and verifying the results of these transmissions (e.g., loss or latency). The proposed algorithm runs several *micro-experiments* continuously, using different rates/configurations and examines the empirical results to identify the one with the highest transmission rate using an online learning algorithm.

In [22], the authors evaluate the historical data samples from specific network features (in this case, RTT and interarrival Acknowledgement (ACK)) as a way to define the most appropriate TCP congestion size. This type of approach, commonly referred to as feature engineering process, was used to develop a congestion control mechanism that can adjust itself to the network environment under scrutiny - instead of using a fixed policy.

These solutions are similar to the proposed solution in this paper. However, they are distinct in terms of the methodology used. As described in more details in Section III, MLPM uses a reduced dataset captured in real-time (performance tracking of specific networking features during operation) and uses a simplified linear regression algorithm (slope only evaluation) to analyse these specific features.

It is important to highlight that this analysis runs over the reduced data periods of a time series and avoids long time series that would result in values operating by average and low variation. The linear regression was chosen due its low complexity (when compared to more sophisticated algorithms such as ARIMA) and can be used in this near real-time operation with lower overhead.

The results of the linear regression analysis are then fed to a linear classifier that applies an ANN classification to select the subflow that meets the application's requirements. Also here the linear classifier computation is reduced due to the typical lower number of neurons (an input layer representing the subflows features, a hidden layer for subflow classification and output layer the evaluate the classification) and lower number of features typically analysed.

III. MLPM

MLPM main contribution is its ML approach to forecast the subflows' performance by using linear regression and classify the results through the use of an ANN classification to select the best subflow that satisfies a *"feature x weight constraint"* (detailed explained in subsection III-B). Through this ML approach, MLPM seeks to outperform the other algorithms used as a comparison (Default MPTCP NS-3 implementation and RTT-aware Packet Delivery Prioritisation Algorithm (RDPA) [15]).

Figure 2 shows how MPTCP is extended with two basic MLPM blocks. The MLPM algorithm description follows a modified version of the machine learning for networking basic workflow described in [19] and summarised here:

•**Problem formulation**: the performance of a subflow's feature must be predicted in real-time, and the time series representing its behaviour must be small to guarantee that the memory and processing capabilities needed have a small footprint and process overhead.

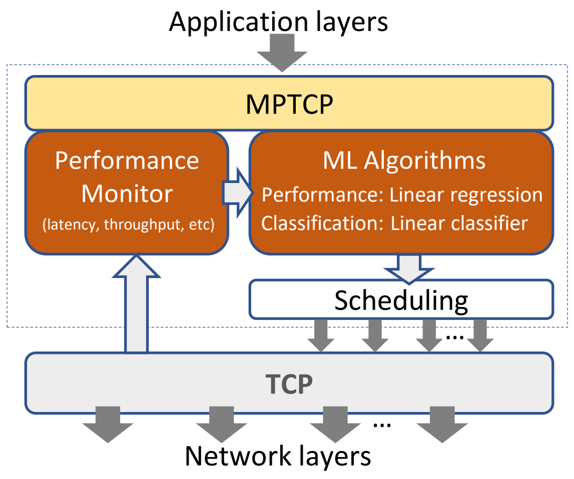
Application layers
MPTCP
Performance Monitor
(latency, throughput, etc)
ML Algorithms
Performance: Linear regression
Classification: Linear classifier
Scheduling
...
TCP
...
Network layers

Fig. 2: MLPM block architecture

•**Data collection**: MLPM algorithm probes the MPTCP implementation and captures the subflows' features behaviour in real-time, which is used as the performance history data represented as a time series.

•**Data analysis**: MLPM algorithm will extract the latency and throughput features for analysis and performance prediction.

•**Model construction**: first, a simplified linear regression (slope evaluation) is used for predicting a feature's behaviour trend (see Subsection III-A for more details). Afterwards, a linear classifier based on these predictions is applied to choose the most suitable subflow (see Subsection III-B).

•**Model validation**: MLPM algorithm will be compared to two algorithms: an algorithm used by Linux for the MPTCP implementation and a basic round-robin algorithm.

•**Deployment and inference**: MLPM algorithm is implemented as one scheduler in the MPTCP implementation [13] used for tests and assessment.

As MLPM algorithm operates at the transport layer and analyses the data in real-time, it cannot support the use of heavy training data or large datasets like algorithms which work "off-line" [19]. The dataset used by the linear regression - with length varying between the last 10 or 20 occurrences - is a time series composed by a key-value pair that represents the feature performance for a given packet transport (e.g., at *00:00:02s* the packet had a delay of 100ms). That is also why MLPM applies a simplified linear regression slope (and not the full linear regression calculation) and a quick linear classifier.

A. Linear regression

MLPM algorithm uses a *simplified* linear regression [23] that only calculates the slope of the linear regression, using Equation 1, to define the behaviour trend of a subflow's feature. The input data of the linear regression is a time series representing the node behaviour over time and the output from this regression is the value of the slope of the curve representing the trend value expected.

$$S = \frac{n\left(\sum_{i=1}^{n} x_i y_i\right) - \left(\sum_{i=1}^{n} x_i\right)\left(\sum_{i=1}^{n} y_i\right)}{n\left(\sum_{i=1}^{n} x_i^2\right) - \left(\sum_{i=1}^{n} x_i\right)^2} \tag{1}$$

Where S represents the slope of the linear regression meaning the trend value for a subflow feature, n is the number of samples used in the calculations and x_i and y_i are the i_{th} values of x (time value) and y (subflow feature value).

To illustrate how the slope of linear regression can help to choose between two or more subflows, Figure 3 illustrates a hypothetical scenario where a specific feature (e.g., latency or throughput) is monitored in a group of subflows - during a *concurrent* period of time - to identify the one with the highest chances to offer low latency. This is done as follows:

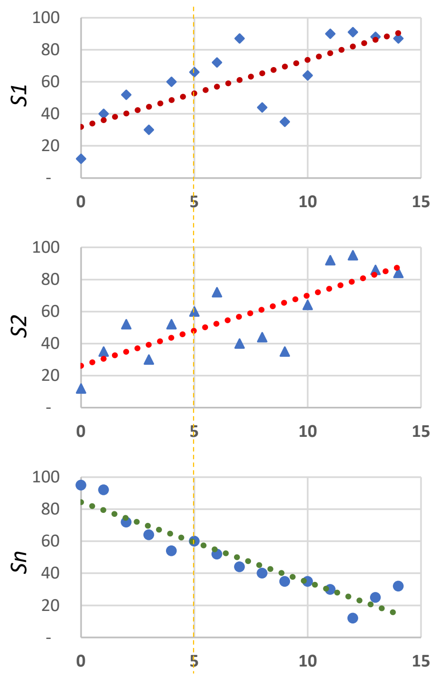

Fig. 3: Regression-based decision example

Note that the vertical axis has not unit as it represents hypothetically any features (latency would be represented in milliseconds and throughput in bytes per second). The horizontal axis represents time but, again, in this hypothetical example, it could be time in different scales (e.g., seconds, milliseconds and so on).

- **from t$_0$ to t$_5$**: regular operation of the subflows generates a time series used for the linear regression calculation.
- **at t$_5$**: the subflows' latency are: *S1* = 66ms, *S2* = 60ms, and *Sn* = 60ms. *S1* has the highest latency, but *S2* and *Sn* are a few milliseconds shorter. Yet, the linear regression (dotted line in each plot) shows that *Sn* has a higher likelihood to offer a lower latency for the next operation: *Sn*'s slope is negative.

Algorithm 1 details the pseudocode for the linear regression slope calculation used generically for any given subflow feature under scrutiny.

Algorithm 1: Linear regression slope calculation

```
Result: Linear regression slope.
Input: $V_h p$ ← subflow's history performance.
1  n, numerator, denominator = 0;
2  $sum_{xy}$, $sum_x$, $sum_y$, $sum_{x^2}$ = 0;
3  foreach ($f$ in $V_h p$) do
4      foreach $obj$ in $V_s$ do
5          n++;
6          $sum_{xy}$ += (obj.feat.time * obj.feat.value);
7          $sum_x$ += obj.feat.time;
8          $sum_y$ += obj.feat.value;
9          $sum_{x^2}$ += pow(obj.feat.time, 2);
10     end
11 end
12 numerator = (n * $sum_{xy}$) - ($sum_x$ * $sum_y$);
13 denominator = (n * $sum_{x^2}$) - pow($sum_x$, 2);
14 return (num / den);
```

B. Linear classifier

After the subflows have their specific features analysed, a classification scheme must address the subflow pool and identify the most suitable subflows for a specific usage. In order to achieve this, MLPM scrutinises the subflows through a linear classifier approach [24]. Figure 4 presents the subflow and its features using an ANN notation, i.e., as a neuron in a neural network.

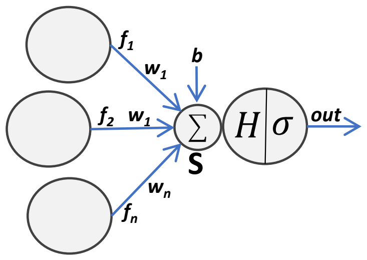

Fig. 4: Regression-based decision

In this method, sublow *features* and their *weights* are represented by vectors, $\vec{f}$ (the historical performance behaviour of the nodes for all the features analysed) and $\vec{w}$ (the desired weight given to each feature by design) respectively. The *dot product* (weighted sum) is applied to both vectors and then added up to the bias b, as detailed Equation 2:

$$\begin{aligned} H &= (\vec{w} \times \vec{f}) + b \\ &= (\sum_{i=1}^{n} w_i \cdot f_i) + b \\ &= ((w_1 \cdot f_1) + (w_2 \cdot f_2) + ... + (w_n \cdot f_n)) + b \end{aligned} \tag{2}$$

The term *weight* concerns how some feature can be *adjusted or balanced* in the search of the best implementation performance. For example, for a given scenario the latency can be more important whilst throughput can be secondary and then they can have specific weight values to balance the scheduler accordingly. The weighted sum it the operation performed over the matrix of values of *features X weights*. To compute the weighted sum into a single result, an activation function $H(v)$ (Equation 3) is used to define whether the neuron would be fired or not.

$$\begin{aligned} H(v) &= H(\vec{w}.\vec{f} + b) \\ &= \begin{cases} 1, & \text{if } v <= 0 \\ 0, & \text{if } v > 0 \end{cases} \end{aligned} \tag{3}$$

Where $\vec{f}$ is the subflow's features analysed in the previous process and $\vec{w}$ are parameters that can be used to adjust the *relevance* of each feature. The bias b is an inclusion of an intercept added to a linear equation, usually used to adjust the output – along with the weighted sum – and offers means to adapt to a specific domain or applicability. Once the model is defined, b remains constant and it adjusts the model to the given data. Generally, $H(v)$ is a Heaviside step function achieved by a *Sigmoid* activation function defined in Equation 4.

$$\sigma(v) = \frac{1}{1 + e^{-v}} \tag{4}$$

Known as a linear classifier, this model defines the trigger boundary based on a linear combination of inputs. Algorithm 2 describes how to implement a weighted sum calculation using a Sigmoid function.

IV. Testbed and Assessment

The MLPM algorithm is implemented, tested and assessed in a simulation environment based on the Network Simulator 3 (NS-3) open-source MPTCP [13] implementation of the IETF Request for Comments (RFC) 8684 [12].

Algorithm 2: Linear classifier

```
Result: Weighted sum of $\vec{f} \times \vec{w}$ added to $b$.
Input: features ← $\vec{f}$;
weights ← $\vec{w}$;
b ← $bias$;
result = 0;
if $inputs.size > 0$ && $weights > 0$ then
    $ws$ = 0;
    $limit = inputs.size$;
    while $index > limit$ do
        $ws$ =+ inputs[i] . weights[i];
    end
    $ws$ =+ $ws$ + b;
    result = 1 / (1 + (1/pow(e, $ws$)));
end
return result;
```

The simulation scenario consists of a Point-to-Point (P2P) model where nodes are configured to have 1Mbps data rate and 2ms delay. It is composed of a source node (as a device that would generate the traffic data), a sink node (to where the data will be sent) and two intermediary nodes (as router devices or middleboxes).

A MpTcpBulkSender application is set on the source node and a MpTcpPacketSink application is set on a sink node. They are extensions of the original NS-3 applications prepared to send and receive simulated data, as fast as possible, over the MPTCP implementation. Other simulation parameters are depicted in Table I.

TABLE I
Simulation setup

Parameter	Value
Environment	NS-3 open source MPTCP [13]
Simulation length	1200 seconds
Number of nodes	4 Nodes
Data Rate	1Mbps
Delay	2ms
Number of subflows	8
Prioritised/non-prioritised ratio	1/500

The simulation scenario consists of a prioritisation scheme which identifying specific packets and applies MLPM to pinpoint which subflow would be able to transmit that specific packet with the best performance. For that, the ratio of 1/500 (one in every 500) packets is prioritised. Although its application is not limited to this technology, this ratio is based on an example of prioritisation mentioned by [25] which uses a VR application average between "simpler" components (e.g., Inertial Measuring Unit (IMU)) and video data packets.

To assess MLPM performance, we compare it to two algorithms: 1) the default MPTCP used in NS-3; and

2) a "smaller RTT policy" named here as "LoRTT" which uses the subflow presenting the lowest RTT value for the transmission of information. This approach is suggested by [26] in a Linux kernel implementation.

It is important to highlight that the *time parameter* presented in Table II is "different" among the assessed algorithms. This happens because each algorithm has its own initialisation time (due to each implementation). In other words, the *time parameter* is defined by the initial processing time of each implementation and not as a fixed parameter. Also, the simulations' analyses does not consider the initial 300 seconds (on average) to avoid transient states.

In this simulation scenario, RTT and throughput are the features under scrutiny and Table II shows the measured results for the prioritised packets using the three algorithms. We observe that MLPM outperforms both LoRTT and the default MPTCP algorithm. For instance, MLPM incurs an average RTT that is 7.6% and 3.0% less than that of the default MPTCP and LoRTT, respectively.

However, in some instances, MLPM incurs RTT 28% lower than the other two algorithms. This happens especially in cases when there is a higher variation of RTT between the subflows during operation due to network conditions and congestion. This behaviour and results are expected given that MLPM is capable of predicting the future trend of subflows' features. Additionally, MLPM generates a throughput that is 5.8% and 2.8% higher than the default MPTCP and LoRTT, respectively.

In addition, to check for unintentional and/or undesired side effects that might be caused by the implementations on the testbed, we monitored other aspects/features of the TCP protocol to assure that its regular operation was not impacted. Results are depicted in Table III.

We observe MLPM performs as good as the default MPTCP in all of the monitored features (i.e., retransmissions, duplicate ACK, lost segments or fast retransmissions. Even though the number of retransmissions and duplicate ACK have increased slightly (.08% and 0.10% respectively) for MLPM, these values are surely considered within operational range for most applications [27].

The algorithm low complexity is achieved basically due to the way the main algorithms are applied: the simplified linear regression approach focuses on the calculation of the slope trend, and the linear classifier does not have a complex regression-based decision given the relatively small number of "*neurons*" defined by a typically small number of subflows.

Also, the relatively small size of the time series – designed to monitor the short term performance and to represent the behaviour in a short set of samples and not by long term average – presents a small footprint and process overhead. These characteristics are the main aspects related to a viable implementation.

TABLE III
PACKET LOSSES - SINGLE-HOMED

Type	MPTCP (%)	MLPM (%)
retransmission	1,11	1,19
duplicate ACK	2,40	2,50
lost segment	0,49	0,48
fast retransmission	0,03	0,01

These results indicate that MLPM improves both RTT and throughput. It also indicates that it can be a resource for prioritisation of specific VR content as it can identify subflows with the best performance and send specific content/components. Consequently, this network communication improvement promoted by MLPM can enhance also the overall VR immersive experience.

V. CONCLUSIONS

This paper proposes MLPM to improve MPTCP management and its intrinsic multipath characteristics. MLPM introduces the use of ML techniques applied to the transport layer of the OSI model. Given the restrictive conditions of its operation (small footprint and process overhead due to real-time operation), MLPM employs a simplified approach of linear regression to forecast the performance of subflows' features, and an ANN linear classifier to help select the best subflow that can meet the application requirements. Simulation results show that MLPM outperforms existing schemes in terms of latency and throughput.

The results demonstrate that MLPM outperforms MPTCP in 7.6% and LoRTT in 3.0% when considering RTT. It also shows a peak performance of around 28% as it achieves better performance in situations where the variation of RTT is higher. It indicates that the ML approach used by MLPM improves the performance by forecasting future performance of specific features. Additionally, MLPM slightly outperforms MPTCP in 5.8% and LoRTT in 2.8% on average. Finally, MLPM can be a significant contribution to prioritisation schemes, especially in scenarios with higher noisy and/or higher feature variation.

As future work, we plan to design an adaptive algorithm combining ML algorithms and content prioritisation schemes as well as developing a study correlating Quality of Service (QoS) improvement (or deterioration) and its impact on Quality of Experience (QoE).

TABLE II
MLPM PERFORMANCE - SAMPLE DATA

MPTCP			LoRTT			MLPM		
time (s)	RTT (ms)	thru (B)	time (s)	RTT (ms)	thru (B)	time (s)	RTT (ms)	thru (B)
304.5	912	1535	298.5	1087	1287	298.4	1111	1260
310.3	923	1516	304.3	1087	1287	304.4	935	1497
316.1	935	1497	310.2	1111	1260	310.4	935	1497
321.9	959	1459	316.0	1122	1247	316.1	1040	1346
327.8	982	1425	321.9	1146	1221	321.9	1064	1315
333.6	994	1408	327.7	1146	1221	327.8	1076	1301
339.4	1040	1346	333.7	1181	1185	333.6	1087	1287
...	...	...	...	...	...	...	...	...
1001.0	1157	1210	995.6	994	1408	995.7	947	1478
1006.8	1169	1197	1001.4	1005	1393	1001.5	959	1459
1012.7	1169	1197	1007.3	1005	1393	1007.3	970	1443
1018.5	1181	1185	1013.1	1005	1393	1013.2	982	1425
1024.4	1181	1185	1019.0	1017	1376	1019.0	994	1408
1030.3	1169	1197	1024.8	1017	1376	1024.9	1005	1393
1036.1	1157	1210	1030.7	1029	1360	1030.7	1017	1376
Average RTT								
1110ms ± 68ms			1063ms ± 101ms			1032ms ± 111ms		
Average Throughput								
1266B ± 82B			1317B ± 146B			1356B ± 153B		

ACKNOWLEDGEMENTS

This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under grants 688503 and 870610 and by Science Foundation Ireland grants 13/RC/2094 (Lero), 16/SP/3804 (ENABLE) and 12/RC/2289_P2 (Insight).

REFERENCES

[1] M. McLuhan, *Understanding media: the extensions of man*, 1st ed. London, UK: Routledge and Kegan Paul, 1964.

[2] N. Tesla, "When woman is boss," *Collier's magazine*, 1 1926.

[3] M. G. Miniwatts, "Internet usage statistics," 2019. [Online]. Available: www.internetworldstats.com

[4] Nielsen, "The topsy-turvy video-viewing landscape," *Video on Demand*, no. March, pp. 1–20, 2016.

[5] Cisco, "Cisco Annual Internet Report (2018–2023)," *Cisco*, pp. 1–41, 2020. [Online]. Available: https://bit.ly/3bVPJyc

[6] R. T. Azuma, "A survey of augmented reality," *Presence by MIT*, vol. 6, no. 4, pp. 355–385, 1997. [Online]. Available: http://bit.ly/2NBMcKJ

[7] D. Bogusevschi, M. Bratu, I. Ghergulescu, C. H. Muntean, and G.-M. Muntean, "Primary School STEM Education: Using 3D Computer-based Virtual Reality and Experimental Laboratory Simulation in a Physics Case Study," *Innovative Pedagogies for Effective Technology-Enhanced Learning (IPETeL) Workshop*, p. 5, 2018.

[8] T. Mazuryk and M. Gervautz, "Virtual Reality History, Applications, Technology and Future," *Digital Outcasts*, vol. 63, no. ISlE, pp. 92–98, 2013. [Online]. Available: https://bit.ly/357J824

[9] NGMN Alliance, "NGMN 5G White Paper," *Next Generation Mobile Networks, White paper*, pp. 1–125, 2015. [Online]. Available: https://bit.ly/2TuBv07

[10] M. A. Togou, D. A. Chekired, L. Khoukhi, and G.-M. Muntean, "A Hierarchical Distributed Control Plane for Path Computation Scalability in Large Scale Software-Defined Networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 1019–1031, 2019.

[11] D. A. Chekired, M. A. Togou, L. Khoukhi, and A. Ksentini, "5G-Slicing-Enabled Scalable SDN Core Network: Toward an Ultra-Low Latency of Autonomous Driving Service," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1769–1782, 2019.

[12] A. Ford, C. Raiciu, M. Handley, U. Bonaventure, and C. Paasch, "RFC 8684: TCP Extensions for Multipath Operation with Multiple Addresses," Internet Engineering Task Force (IETF), Tech. Rep., 2020.

[13] M. Kheirkhah, I. Wakeman, and G. Parisis, "Multipath-TCP in ns-3," in *Cornell University Library*, 2015, pp. 3–4. [Online]. Available: https://bit.ly/2MfiElc

[14] H. Liu, Q. Li, B. Wu, Q. Wu, J. Zhang, and J. Zhou, "Trusted Multipath-TCP (MPTCP) extension," Internet Engineering Task Force (IETF), Tech. Rep., 2019.

[15] F. Silva, D. Bogusevschi, and G.-M. Muntean, "An MPTCP-based RTT-aware Packet Delivery Prioritisation Algorithm in AR/VR Scenarios," in *IEEE IWCMC 2018 - 14th International Wireless Communications and Mobile Computing Conference*. Limassol, Cyprus: IEEE Xplore, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8450524

[16] J. Young and D. Wing, "MPTCP and Product Support Overview," 2013. [Online]. Available: https://bit.ly/2CsIfmn

[17] C. Xu, S. Member, J. Zhao, and G.-m. Muntean, "Congestion Control Design for Multipath Transport Protocols : A Survey," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 4, pp. 2948–2969, 2016.

[18] P. Dong, J. Xie, W. Tang, N. Xiong, H. Zhong, and A. V. Vasilakos, "Performance Evaluation of Multipath TCP Scheduling Algorithms," *IEEE Access*, vol. 7, pp. 29 818–29 825, 2019.

[19] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine Learning for Networking: Workflow, Advances and Opportunities," *IEEE Network*, vol. 32, no. 2, pp. 92–99, 2018.

[20] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction," *SIGCOMM 2016 - Proceedings of the 2016 ACM Conference on Special Interest Group on Data Communication*, pp. 272–285, 2016.

[21] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "PCC: Re-architecting congestion control for consistent high performance," *Proceedings of the 12th USENIX Symposium on*

Networked Systems Design and Implementation, NSDI 2015, pp. 395–408, 2015.

[22] K. Winstein and H. Balakrishnan, "TCP ex Machina: Computer-Generated Congestion Control," *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM - SIGCOMM '13*, 2013.

[23] D. Mladenić, J. Brank, M. Grobelnik, and N. Milic-Frayling, "Feature selection using linear classifier weights: Interaction with classification models," *Proceedings of Sheffield SIGIR - Twenty-Seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 234–241, 2004.

[24] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: a case study in handwritten digit recognition," in *12th IAPR International Conference on Pattern Recognition*, 2002, pp. 77–82.

[25] M. Chouiten, J.-Y. Didier, and M. Mallem, "Distributed Augmented Reality Systems: How Much Performance is Enough?" *2012 IEEE International Conference on Multimedia and Expo Workshops*, pp. 337–342, 2012.

[26] C. Paasch, S. Barré, F. Duchêne, and G. Detal, "Multipath TCP in the Linux Kernel," 2019. [Online]. Available: https://www.multipath-tcp.org

[27] J. S. Mwela and O. E. Adebomi, "Impact of Packet Loss on the Quality of Video Stream Transmission," Ph.D. dissertation, Blekinge Institute of Technology, 2010.