Towards Application-Aware Networking: ML-based End-to-End Application KPI/QoE Metrics Characterization in SDN

Hamed Z. Jahromi
School of Computer Science
University College Dublin, Ireland
Email: hamed.jahromi@ucdconnect.ie

Andrew Hines
School of Computer Science
University College Dublin, Ireland
Email: andrew.hines@ucd.ie

Declan T. Delaney
School of Computer Science
University College Dublin, Ireland
Email: declan.delaney@ucd.ie

***Abstract*—Software Defined Networking (SDN) presents a unique networking paradigm that facilitates the development of network innovations. This paper aims to improve application awareness by incorporating Machine Learning (ML) techniques within an open source SDN architecture. The paper explores how end-to-end application Key Performance Indicator (KPI) metrics can be designed and utilized for the purpose of machine learning in networks. The main goal of this research is to characterize application KPI metrics using a suitable ML approach based on available network data. Resource allocation and network orchestration tasks can be automated based on the findings.**

A key facet of this research is introducing a novel feedback interface to the SDN's Northbound Interface that receives realtime performance feedback from applications. This paper shows how we might exploit the applications feedback to determine useful characteristics of an application's traffic. A mapping application with a defined KPI is used for experimentation. Linear multiple regression is used to derive a characteristic relationship between the application KPI and the network metrics.

***Index Terms*—Application Awareness, SDN, KPI, QoE**

I. Introduction

An increasing number of new network based applications are entering the market. Each application pursues different performance objectives. Applications often evaluate the success of their objectives using KPI and Quality of Experience (QoE) metrics [1]. KPI and QoE metrics help applications to determine how effectively the application is achieving its performance objectives [1]. The network as a transport infrastructure of such applications plays a vital role in the end-to-end performance of applications. Therefore, the performance of applications highly depend on how well the network understands and serves the application requirements [2].

The link between network parameters and QoE is an active area of research, particularly for multimedia applications using video streaming or VOIP. This has led to a better understanding of the relationship between QoE factors: network service factors including delay and packet loss, content factors like room echo effects on speech and context factors such as signal compression. In order to improve perceptual QoE, application based methods to mitigate these effects have been developed. For example jitter buffers provide delay compensation and echo cancellation algorithms suppress unwanted noise.

More recently network based QoE management has sought to apply the same principles to apply QoE. Many applications that would not fit within the traditional categorization of multimedia can still benefit from QoE based management. This paper presents a case study of a web-based mapping application to illustrate how a QoE management strategy can be used to adapt network parameters based on application KPI/QoE metrics.

QoE is used to measure and express perception of the users within an application. QoE is more than just a performance indicator service as it discovers the level of users satisfaction and their response to the service. While QoE is a user metric, this can be mapped to measurable application KPIs.

As shown in Fig.1, the proposed approach shows how KPIs can be defined for applications based on application QoE experiments. Using KPIs as proxies for QoE, minimum performance thresholds can be mapped to network metrics through a QoE model. This allows SDN to become application aware and dynamically adapt network parameters to the applications QoE needs.

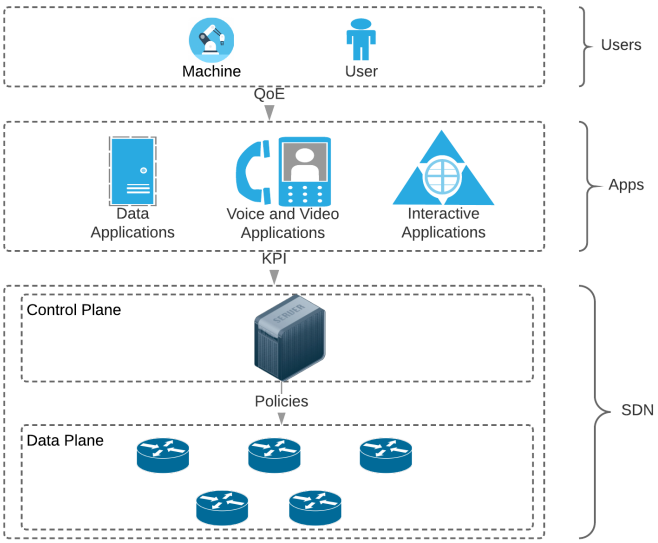

Fig. 1. Applications QoE/KPI and Network Policies

Application aware networks aim to have a better understanding of applications and allocate network resources in accordance with the application requirements. Application awareness in a typical distributed network relies on offline experimental analysis of network expectation of applications. This analysis helps the network orchestrator to find the minimum network resources required by the application

using available monitored network features or metrics (e.g. bandwidth, delay, jitter, packet loss ,error rate, etc.). Once the application requirement is analyzed, the network orchestrator defines generic QoS or routing policies in order to classify and allocate network resources for an application or group of applications [3, 4]. Application requirements, however, are subject to change over time. Depending on the active use case, the number of concurrent users or the running environment an application might target different objectives [5]. The dynamic nature of applications makes it difficult to perform a diverse off-line analysis that complies with various use cases. Even with such analysis, it can lead to over-provisioning of network resources or the defined QoS and routing policies may not fulfill the applications needs [5].

This paper presents a novel approach to improve application awareness using customized application QoE/KPI management. It shows how application KPI metrics can be analyzed using network metrics in SDN environment. A trained machine learning based KPI model and SDN architecture has been utilized to demonstrate realtime characterization of application specific KPI metrics. This paper focuses on QoS-KPI network management and the granularity of the model is per-case per-application. Future work will investigate flow installation based on KPI-QoE mapping to allow KPIs to be further explored and utilized effectively in network management.

The main contributions of this paper are utilizing application feedback for the purpose of learning application behavior and demonstrating the ability of ML algorithms to characterize application performance metrics in a network.

In this research we assume that: 1.The SDN can classify application type for each flow [6], 2.The SDN can gather and monitor a set of metrics from the network [7]. 3. Applications evaluate KPI performance metrics [8].

II. MOTIVATION

In distributed networks, automating applications KPI/QoE metric analysis and updating network transport polices suffer from a number of issues which include:

1. Collecting and characterizing end-to-end network metrics (Network Telemetry): in distributed networks each node operates independently. Therefore, collecting network information and metrics across the entire network infrastructure requires an extensive interaction and effort between all nodes across the network [9].

2. Incorporating feedback from application layer: the network operates based on defined policies for each node and does not provide any concrete or abstracted mechanism to interact with the applications [10]. It can be said that the network does not consider the applications feedback in realtime during any flow installation or decision making processes.

3. Changing network policies on-demand: Application traffic in the network is handled using control plane protocols which exist on the each node of the network (e.g OSPF, ISIS, EIGRP and etc.). Therefore, dynamically updating network transport policies are limited to the options available within the control plane protocols [11].

The analysis conducted in this paper fits into a model that offers solutions to these issues. The proposed approach incorporates supervised machine learning and software defined networking in order to characterize an arbitrary application KPI metric. Based on the supervised machine learning structure, network metrics collected by the SDN controller will be considered as input and the application KPI metric as an output. Correlation analysis has been performed to examine the hypotheses and determine whether KPI metric is highly and positively related to network features. The end goal is to characterize application KPI metric using network metrics and approximate a mapping function which can predict the KPI metric based on the path metric values.

III. RELATED WORKS

The emergence of Software defined networking (SDN) and programmable networks with centralized architecture allow network research to explore new level of application awareness. Othman and Okamura and Nam et al. [12, 13] utilize SDN platform and present dynamic traffic flow control for a Content Distribution Network (CDN) [12, 13]. Despite the novelty of the idea, customization of standardized transport protocol stack may not scale across different networks and causes compatibility issues. Furthermore, the presented platform [14] is targeting specific application and is not adaptable to different applications. Lee et al., Soulé et al. and Gorlatch et al. [15, 16, 17] employ SDN framework and provide a programmable northbound interface. This acts as an abstraction layer that let applications declare their networking requirements with minimal networking knowledge. Gorlatch et al. [17] proposed a mechanism for application-related QoS metrics, rather than the typical network-related metrics. Gorlatch et al. developed a Northbound API for the application developers in order to declare the expected QoS metrics. Although [15, 16, 17] define an abstraction layer that receive application requirement, one can argue that impacts of the decisions made by the network on the applications are yet to be explored.

To the best of our knowledge a few key aspects of application awareness in SDN are yet to be investigated. This include: 1. Evaluation of historical network actions against applications performance objectives, 2. How well a network fulfills the performance goals of a specific application, 3. A model that helps the network to consider applications performance objectives when making a new decision.

IV. MAP BASED APPLICATION

In order to validate the proposed architecture, we study a preliminary use case. In this case, a web mapping application similar to Google Maps or Open Street Maps web applications. These applications use raster image tiles to present a map view that can be zoomed or panned. Pre-buffering and caching of tiles allow panning and zooming out with a smooth quality of experience. However there is an asymmetry to the amount of data required for zoom in compared to zoom out. Zooming in several zoom levels exhausts the pre-buffered data and with

black tile squares visible leading to a poor QoE. In the web map example an "load time" KPI is impacting QoE which is important for both human users and a machine e.g. self driving car systems. Both human and machines require direction data from the system in real time to navigate. However, using zoom-in click rate as a KPI could allow networks to deal with flow installation to the application for QoE as this is a human perceptual KPI rather than a machine to machine application KPI. This proves an interesting test case as little work has previously been conducted on mapping traffic despite online maps becoming a ubiquitous and well used application.

V. Testbed

The main objective of the testbed is to demonstrate the effectiveness of centralized SDN architecture towards collecting and preparing network and application information for ML-Based application awareness analysis. During this test, we stress test the network giving it various metrics and examine a mapping applications client "load time" KPI on each condition. Then we demonstrate how the KPI metric can be analyzed and predicted using network metrics.

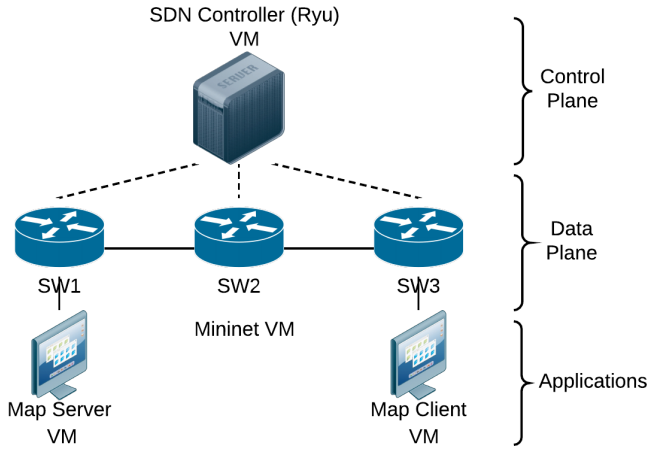

Fig. 2. Application Aware Network Testbed

The testbed (Fig. 2) is built on top of a virtual box hypervisor installed on Dell OPTIFLEX 5040 With Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz and 16.0 GB of RAM. The testbed is composed of four different components in which each component is running on a separate VM. The testbed components include: SDN controller, Open-Flow based switches, map application server and map application client.

A. SDN Controller

The SDN controller acts as a central intelligence unit of a software defined networking architecture. SDN controllers operate with minimum of two interfaces which include Southbound Interface (SBI) and Northbound Interface (NBI). Northbound interface provides network APIs and allows higher layers applications to interact with the SDN controller. Southbound interface facilitates flow installation and data plane programming for the network devices. Ryu [7] has been chosen as the SDN controller for the testbed. Ryu is deployed on a VM running Ubuntu 16.04.3 X86 64 bits with two CPUs, 2.0 GB of RAM and one network adapter. Ryu is a component-based SDN controller framework which provides well defined API and facilitates development of new network management and control applications.

B. OpenFlow Based Switches

An OpenFlow switch is a software based or hardware based switch that forwards packets in a SDN environment. OpenFlow switches use the OpenFlow protocol (Southbound) to interact with the SDN controller. The SDN controller pushes flow records to the switch flow-table using OpenFlow. Mininet software [18] is used to emulate the data plane and deploy multiple OpenFlow based switches. Mininet is using lightweight virtualization mechanism which employs the default Linux bridge or Open vSwitch [18] and switches packets across interfaces. In the testbed, Mininet is running on an Ubuntu 14.04.5 X86 64 bits VM with one CPU, 1GB of RAM and three network interfaces. We have emulated a linear topology of 3 OVS switches using Mininet (Fig. 2). While the topology used is basic it serves to test the KPI-QoS relationship as a basis for further investigation.

C. Map Application Server

The map application sever VM hosts OpenStreetMap [19] application. OpenStreetMap is an open-content license map of the world built by contribution of volunteers. it is used in teaching geography, mathematics, ecology, community planning and technology [19]. The map application server is an Ubuntu 12.04 X86 64 bits VM with one CPU, 1.0 GB of RAM and one network adapter. The network adapter is connected to the SW1 of the emulated network topology (Fig. 2). The map application server hosts a Node.js [20] script that receives KPI metrics values along with the timestamps from clients, and pass it to the SDN controller. In this experiment, the script is limited to serve a single client.

D. Map Application Client

The map application client is an HTML file which loads world map from the server using HTTP, Leaflet, Ajax and JavaScript [20]. The default latitude, longitude and zoom level of map is set to 53.30, -6.22 and 10 respectively. The online map is segmented into multiple tiles based on the zoom level. Depends on where you click on the map, N number of tiles will be fetched from the server. In order to examine the performance of the map, We have added a JavaScript KPI function into the HTML file. The function calculates the amount of time that it takes to load all the tiles (Load Time KPI) based on the current latitude, longitude and zoom level. The output of the KPI function will be passed to the map application server using an HTML Ajax form and used in ML analysis. Map application client is running on a Microsoft Windows Seven VM with one CPU, 2.00 GB of RAM and one network interface. As shown in Fig. 2, the map application client network adapter is attached to the SW3 of the emulated network topology.

VI. Metric Collection and Preparation

Metric collection and preparation follows the structure shown in Fig. 3 for collecting and storing metrics from the network and the application. This includes Application Metric Collection (AMC) and Network Metric Collection (NMC) processes. Both processes are implemented as an independent application of the Ryu SDN controller.

A. Application Metric Collection

Application Metric Collection (AMC) is providing a North-Bound Feedback Interface (NBFI) to the application. The NBFI facilitates a realtime connection between applications and the SDN controller. This allows applications to declare the KPI metrics and pass the respective timestamped KPI values. AMC stores KPI/QoE metrics values in a json file structure [21] and creates a historical dataset.

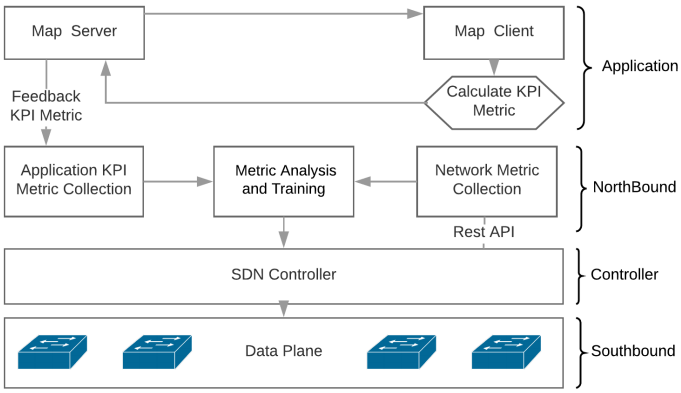

Fig. 3. Application Aware Network Functional Block

B. Network Metrics Collection

Network Metrics Collection (NMC) process takes network snapshots and creates a structured data for each snapshot. The structured data represents the network metrics at the time that the snapshot is taken. NMC establishes a connection with Ryu SDN controller using REST APIs provided by the controller and collect network information [22]. In this experiment, NMC collects and calculates network metrics which include Delay, Bandwidth Capacity, Rx Load, Tx Load and Packet Loss. A significant challenge still remains in determining which metric in the network should be collected and how these metric should be grouped and used for analysis. Multicollinearity in predictive analysis is a fundamental approach and is used in this paper to determine the key network metrics that are moderately or highly correlated.

VII. Metrics Analysis and Training

Metric Analysis and Training (MAT) is implemented as an application of the Ryu SDN controller. MAT translates the applications KPI metrics by analyzing the data collected by the application and network metric collection processes. MAT translates application KPI metrics using network metrics by applying machine learning algorithms and extracting the relationship between each KPI metrics and the set of network metrics. It performs this in two steps:

1. Preparing a Dataset for analysis: MAT Fetches the historical application KPI and network metric values from json files prepared by AMC and NMC. It then joins both network metrics and application KPI metric values base on the recorded timestamps. The end result is a dataset that includes network metrics as input features and the KPI metric as an output feature.

2. Applying machine learning technique: MAT perform machine learning analysis and extract a model which describes the relationship between the network metrics (input features) and the Application KPI metric (output feature). The outcome of MAT process is a model which approximates the application KPI metric value. The model can then be used by the SDN controller to make decisions during flow installation process and maximizes the applications performance.

Machine learning algorithms used to find the best fit model between the KPI network metrics and network features. Success of such algorithms highly depend on the type of KPI metric. KPI metrics values may fall under different categories which include numerical, binary, categorical and time series. There are several machine learning mechanisms which may be effectively employed such as Recurrent Neural Network (RNN), Support Vector Machines (SVM), decision tree and Bayesian network [23]. The use of these technologies needs to be studied further. The challenge remains to determine and select an appropriate algorithm based on the KPI metrics value types. The current scope of this experiment is limited to the demonstration of the use of supervised machine learning with multiple linear regression on characterization of application KPI metric. Multiple linear regression facilitates identifying the strength of the effect of individual network metrics on an application KPI metric and a model that predicts application KPI metrics using network metrics.

VIII. Experiment

In this experiment, network and application KPI metrics are collected while the SDN network is dynamically stressed using a python script. The script uses Mininet features to manipulate network delay (0 to 500 ms), bandwidth capacity (1Mb to 100Mb), link load (0% to 100%) and packet loss (0% to 5%) . IPERF [24] is used to inject dynamic load on the path between the map server and client. The map client fetches an online map from the map application server and repeats every 10 second for 2300 iterations. On each iteration, “load time” KPI will be calculated and submitted to the AMC process using the Northbound Feedback Interface (NBFI). At the same time, NMC collects and creates historical network metrics. MAT combines application metric and the network metric dataset and prepares it for regression analysis. The outcome of this analysis include: 1.KPI metric characterization: impact level of each network metric on the KPI metric (Fig. 6, Fig. 4). 2.Prediction Model: A model that predicts KPI metric value based on the network metric(Fig. 5).

The summary of the results of the conducted multiple linear regression analysis for the purpose of testing the hypothesis are shown in Fig. 6 and Fig. 4. KPI metric characterization is realized using regression coefficients (Fig. 6). The regression coefficient (coef) represents the change in the KPI metric value resulting from a one unit change in the network metric, with all other network metrics being held constant. For instance, as shown in shown in Fig. 6 and Fig. 4, we find that a one unit (Millisecond) increase in delay increase map load time by 6.129 (Millisecond). Fig. 4 reflects Adj. R-squared that explains how efficient the prediction model is. The model shows a high R-squared value, 0.864, indicating a high confidence in directly explaining the variance in the "load time" KPI.

Based on the results visualized in Fig. 5 and Fig. 6, we make the following observations. First, the end-to-end delay has the most significant impact on the "load time" KPI. Second, The importance of path capacity (Bandwidth) is minimal. Third, Rx load, Tx load and Loss have almost same level of impact on the KPI metric. Forth, Fig. 5 proves that the model is relatively correlated. As we expected, each metric has a different influence on the application. However, the level of influence differs based the application type, application users and the use case. This level of granularity allows the network to evolve over time and become more aware of applications.

A second major finding from the analysis proves that while multiple regression analysis is appropriate to model many of the metrics (Bandwidth, delay and packet loss) it is not appropriate to model others (additional load). This finding instigates a need for further research to present methods which can more fully model the application response in the network.

Overall, this experiment gives evidence that it is possible, to characterize application defined performance metrics using network features. It is intended that a model such as this be used within an SDN controller to drive the installation of application flows based on the path with

lowest predicted value. This is a first step in obtaining higher application awareness in SDN with the use of ML tools.

OLS Regression Results

Dep. Variable:	map_load_time	R-squared:	0.864
Model:	OLS	Adj. R-squared:	0.864

	coef	std err	t	P>|t|	[0.025	0.975]
capacity	2.1720	0.505	4.299	0.000	1.181	3.163
delay	6.1291	0.106	57.985	0.000	5.922	6.336
loss	139.9405	8.985	15.575	0.000	122.321	157.560
rx_load	13.4711	1.857	7.255	0.000	9.830	17.112
tx_load	8.5434	1.844	4.633	0.000	4.928	12.159

Fig. 4. Multiple Linear Regression Result

IX. Conclusion and Future work

The use of machine learning within the SDN paradigm has not yet been fully explored. This paper has presented a novel approach to expand application awareness in computer networks and presents an opportunity for a networked application to have a network recognized performance metric. This work differs from existing approaches in that it dynamically analyzes and translates an application performance metric or KPI using objective network metrics in an SDN. This allows the network to improve the level of application awareness in the network and adapt to the performance goal of applications.

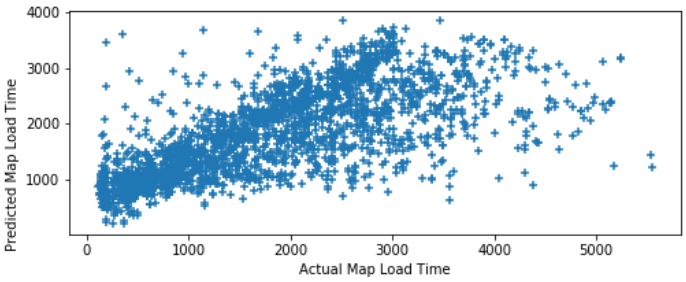

Fig. 5. Actual Load Time vs Predicted Load Time

In ongoing work, we investigate a systematic approach that selects a best fit machine learning algorithm in accordance with the performance metric types. We are also working on a mechanism to capture a more expansive set of network metrics that may characterize the network fully and covers diverse range of application performance objectives. The end goal is to purpose a framework that automatically characterizes KPI and QoE metrics declared by applications.

References

[1] S. M. Al-Shehri, P. Loskot, T. Numanoglu, and M. Mert, "Common metrics for analyzing, developing and managing telecommunication networks," *arXiv preprint arXiv:1707.03290*, 2017.

[2] T. J. Krautkremer, "Methods, apparatuses and systems enabling a network services provider to deliver application performance management services," Aug. 23 2005, uS Patent 6,934,745.

[3] J. W. Jorgensen, "Application-aware, quality of service (qos) sensitive, media access control (mac) layer," Oct. 28 2003, uS Patent 6,640,248.

[4] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2003, pp. 3–10.

[5] T. Ahmed, R. Boutaba, and A. Mehaoua, "A measurement-based approach for dynamic qos adaptation in diffserv networks," *Computer Communications*, vol. 28, no. 18, pp. 2020–2033, 2005.

[6] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in sdn," *ACM SIGCOMM computer communication review*, vol. 43, no. 4, pp. 487–488, 2013.

[7] S. Ryu, "Framework community: Ryu sdn controller," 2016.

[8] R. Kazhamiakin, B. Wetzstein, D. Karastoyanova, M. Pistore, and F. Leymann, "Adaptation of service-based applications based on process quality factor analysis," in

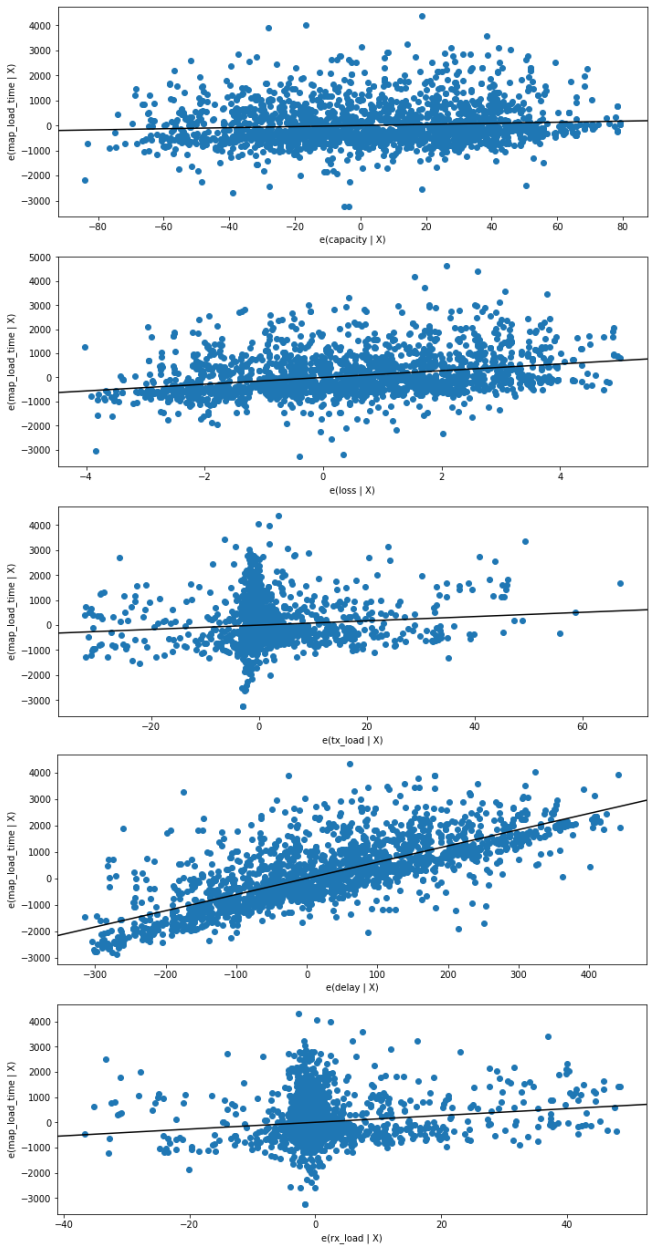

Fig. 6. Partial Regression Plot

Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops. Springer, 2010, pp. 395–404.

[9] S. Olejniczak, "Medical telemetry system with wireless and physical communication channels," Nov. 21 2000, uS Patent 6,150,951.

[10] T. Zinner, M. Jarschel, A. Blenk, F. Wamser, and W. Kellerer, "Dynamic application-aware resource management using software-defined networking: Implementation prospects and challenges," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–6.

[11] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Advances in neural information processing systems*, 1994, pp. 671–678.

[12] O. M. Othman and K. Okamura, "Design and implementation of application based routing using openflow," in *Proceedings of the 5th International Conference on Future Internet Technologies*. ACM, 2010, pp. 60–67.

[13] H. Nam, D. Calin, and H. Schulzrinne, "Intelligent content delivery over wireless via sdn," in *Wireless Communications and Networking Conference (WCNC), 2015 IEEE*. IEEE, 2015, pp. 2185–2190.

[14] G. Valetto, L. Goix, and G. Delaire, "Towards service awareness and autonomic features in a sip-enabled network," *Autonomic Communication*, pp. 202–213, 2006.

[15] J. Lee, Y. Turner, M. Lee, L. Popa, S. Banerjee, J.-M. Kang, and P. Sharma, "Application-driven bandwidth guarantees in datacenters," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4. ACM, 2014, pp. 467–478.

[16] R. Soulé, S. Basu, P. J. Marandi, F. Pedone, R. Kleinberg, E. G. Sirer, and N. Foster, "Merlin: A language for provisioning network resources," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 2014, pp. 213–226.

[17] S. Gorlatch, T. Humernbrum, and F. Glinka, "Improving qos in real-time internet applications: from best-effort to software-defined networks," in *Computing, Networking and Communications (ICNC), 2014 International Conference on*. IEEE, 2014, pp. 189–193.

[18] M. Team, "Mininet," 2014.

[19] A. Ather, "A quality analysis of openstreetmap data," *ME Thesis, University College London, London, UK*, vol. 22, 2009.

[20] P. Crickard III, *Leaflet. js Essentials*. Packt Publishing Ltd, 2014.

[21] D. Crockford, "The application/json media type for javascript object notation (json)," 2006.

[22] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou, "Feature-based comparison and selection of software defined networking (sdn) controllers," in *Computer Applications and Information Systems (WCCAIS), 2014 World Congress on*. IEEE, 2014, pp. 1–7.

[23] M. A. Razi and K. Athappilly, "A comparative predictive analysis of neural networks (nns), nonlinear regression and classification and regression tree (cart) models," *Expert Systems with Applications*, vol. 29, no. 1, pp. 65–74, 2005.

[24] A. Tirumala, T. Dunigan, and L. Cottrell, "Measuring end-to-end bandwidth with iperf using web100," in *Presented at*, no. SLAC-PUB-9733, 2003.