Evaluating Load Adjusted Learning Strategies for Client Service Levels Prediction from Cloud-hosted Video Servers

Obinna Izima, Ruairí de Fréin, Mark Davis

Dublin Institute of Technology, Ireland
Obinna.Izima@mydit.ie, ruairi.defrein@dit.ie, mark.davis@dit.ie

Abstract. Network managers that succeed in improving the accuracy of client video service level predictions, where the video is deployed in a cloud infrastructure, will have the ability to deliver responsive, SLA-compliant service to their customers. Meeting up-time guarantees, achieving rapid first-call resolution, and minimizing time-to-recovery after video service outages will maintain customer loyalty.
To date, regression-based models have been applied to generate these predictions for client machines using the kernel metrics of a server cluster. The effect of time-varying loads on cloud-hosted video servers, which arise due to dynamic user requests have not been leveraged to improve prediction using regularized learning algorithms such as the LASSO and Elastic Net and also Random Forest. We evaluate the performance of load-adjusted learning strategies using a number of learning algorithms and demonstrate that improved predictions are achieved irrespective of the learning approach. A secondary benefit of the load-adjusted learning approach is that it reduces the computational cost as long as the load is not constant. Finally, we demonstrate that Random Forest significantly improve the prediction performance produced by the best performing linear regression variant, the Elastic Net.

1 Introduction

Streaming video content over wired and wireless communication networks will be a major contributor to future internet traffic as can be inferred from [1] which predicts that the global IP video traffic is estimated to be about 82 percent of all consumer traffic by 2021. To safeguard revenues, network providers must be able to proactively monitor customer experience. The authors of [2] adopt a Machine Learning (ML) approach in their work on video service-level prediction using the kernel metrics of the server delivering the video which is a first step to meet this goal.

Fig. 1 depicts the set-up of the system under study in this paper which is representative of real world systems. It is composed of a cloud-based server infrastructure servicing dynamically and time-varying client requests received over a network. Server resources are shared between multiple clients with the video service (RHS) delivering video to target client machines (LHS). The number of

users accessing the server resources changes rapidly and this poses a challenge in predicting the target client's video quality. A video server of this form must be able to handle time-varying loads as users can start and stop videos at arbitrary times simultaneously.

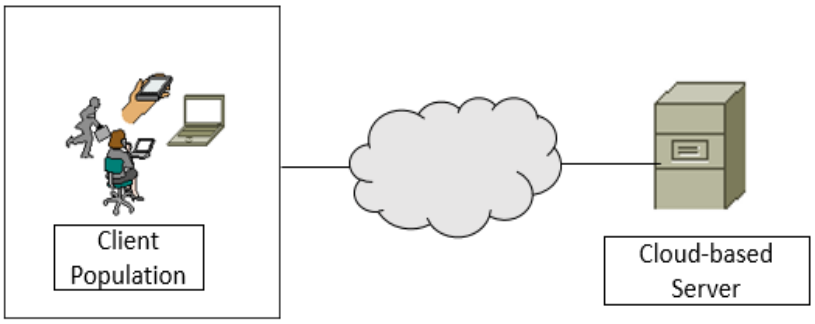

Fig. 1. Client population accessing video servers over the internet under time-varying loads.

We seek a model that characterizes the effect of the time-varying loads on the client's video quality provided we have knowledge of the kernel metrics of the server delivering the video. Both the client machine and the server clocks are synchronized to match up observations. Samples are drawn from the client and server machines every second. A VLC media player services the Video-on-Demand (VoD) requests in [2] and extracts the RTP packet rate, Video Frame Rate (VFR) and Audio Buffer Rate (ABR), y_i at time i. RTP, VFR and ABR are the client's service-level metrics we seek to predict. The System Activity Report (SAR) function on the server extracts the feature set, x. Here, features imply the metrics on the operating system for example, the TCP active connections on the server. The authors of [3] characterize the system we seek to investigate properly; this investigation examines the effect of time-varying requests on the system resources.

We contribute adaptive learning techniques which reduce the computational complexity, and provide more accurate predictions over the baseline approach in [2].

1. We demonstrate this by considering the performance of linear regression methods and non-linear methods using the baseline approach. The linear methods we evaluate are Linear Regression (LR) and also members of the family of shrinkage methods: Ridge Regression (RR), LASSO and Elastic Net. We compare the performance of the best performing linear method with a non-linear method, Random Forest;
2. We evaluate the efficacy of the load-adjusted (LA) learning (i.e. using the TCP socket count in our learning algorithms to improve performance) on two different traces which vary periodically and according to a flashcrowd behaviour;

3. We determine whether the load-adjusted technique works better in linear or non-linear learning algorithms.

This paper is organized as follows. In Section 2, we introduce the load-adjusted learning technique and the Machine Learning (ML) techniques. Section 3 introduces the model fitting procedures and the evaluation framework. In Section 4 we evaluate the efficacy of each of the approaches. Section 5 places our contribution in the context of related literature and we conclude our work in Section 6.

2 Learning Strategies

The server in Fig. 1, collects device statistics, x using SAR. The number of active clients, the load signal at time i can be measured with the TCPSCK feature of x. A load generator dynamically allocates client requests for video to the server under two load patterns, a periodic-load pattern and flashcrowd-load pattern. In the periodic-load pattern, clients are started following a Poisson process with an average arrival rate of 30 clients per minute. This arrival rate is modulated using a sinusoidal function with a period of an hour and an amplitude of 20 clients. The flashcrowd-load pattern starts clients with a Poisson process with 5 clients per minute average arrival rate and peaks at randomly created events at a rate of 10 events per hour. During flash events, the average arrival rate swaps to 50 clients per minute for a minute and then gradually reduces to 5 clients per minute within the next 4 minutes.

Using the device statistics computed from SAR, service-level metrics can be computed at the clients. In this paper, we are interested in predicting the (i) Video Frame Rate (VFR), the number of displayed video frames for any time i; (ii) Audio Buffer Rate (ABR), audio buffers played for time i; and (iii) RTP packet count, the number of received RTP packets in time i. Fig. 2 illustrates a plot of the RTP packet count and the TCPSCK count, the load proxy recorded over a period of 15000 seconds for both load patterns. As can be inferred from Fig. 2, the TCPSCK count rises with an increase in load and may result to a reduction in the RTP packet received at the client because the system has limited resources.

2.1 Load-adjusted Learning (LA)

The effect of concurrent requests on the server kernel are examined. In Fig. 2, we plot the RTP packet count recorded over 15000 seconds and the TCPSCK kernel parameter over the same period of time. From visual inspection of the plot, the RTP packet count has the same periodic pattern as the TCPSCK. It is also evident that as the load increases, the TCPSCK count increases and may lead to a decrease in the number of RTP packet counts at the client because the system does not have unlimited resources.

Load-adjusted model: Let θ_n represent one video resource currently being used by a client. A server response with respect to its kernel metric, the n-th

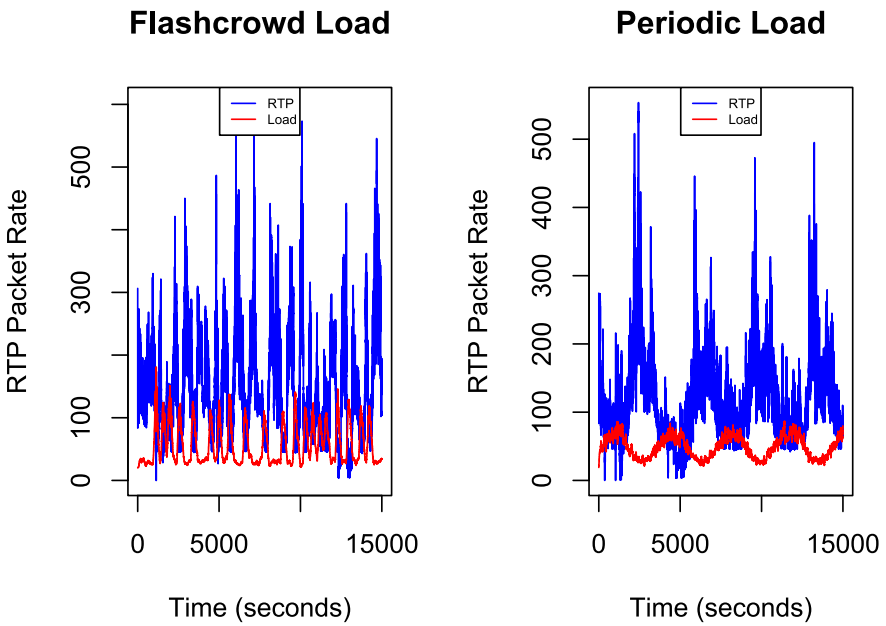

Fig. 2. Service-level metric, y_i, RTP is illustrated for 15000s with the system load, TCPSCK for both load traces.

feature, to one request for a video at time i is the sum of the resource a user has and some deviation signal specific to a feature [4].

$$x_i[n] = \theta_n + \epsilon_i[n], \quad \text{where} \quad i \in \mathbb{Z}, x_i[n], \theta_n \in \mathbb{R}. \tag{1}$$

An additional request for resources by the current user or a new client would invoke a feature response of the form:

$$x_i[n] = 2\theta_n + \epsilon_i[n,1] + \epsilon_i[n,2]. \tag{2}$$

The deviation from the ideal performance arising from the second user is denoted by $\epsilon_i[n,2]$. Assume that at any time, i the number of users requesting the service is $K[i]$. When there are five client requests for server resources, $K[i] = 5$. The response of the n-th feature to the time-varying load is

$$x_i[n] = \theta_n K[i] + \sum_{k=1}^{K(i)} \epsilon_i[n,k]. \tag{3}$$

The load signal $\theta_n K[i]$ denotes the number of active users at time i times the resources one user uses, θ_n. The TCPSCK is $K[i]$.

Un-adjusted Learning (UA): Previous attempts at predicting service-level metrics from device statistics do not model the effect of the time-varying load.

They assume that $K[i]$ is constant C.

$$x_i[n] = \theta_n C + \sum_{k=1}^{K[i]} \epsilon_i[n,k]. \tag{4}$$

Problem Statement: Our objective is to learn a model that predicts service-level QoS metrics, y: RTP packet rates, video frame rate and audio buffer rate using the features $x_i[n]$ given a time varying load $K[i]$. Using both the flashcrowd-load and periodic-load traces, we test the hypothesis that a learning algorithm which takes the load value into consideration produces better predictions than models obtained using algorithms which ignore the load.

Table 1. We investigate the hypothesis that models learned which consider effects of time-varying loads offer better prediction performance than models which do not consider the effects of the load. We evaluate these two learning techniques using a range of machine learning algorithms on both the flashcrowd and periodic-load datasets. LA technique out-performs the UA approach.

Learning Technique	Learning Algorithm	Datasets	Outcome
Load-adjusted learning (LA)	Linear Regression Ridge Regression LASSO **Elastic Net** **Random Forest**	Periodic-load trace Flashcrowd-load trace	**LA technique offers better predictions**
Un-adjusted learning (UA)			

2.2 Machine Learning Techniques

In this section, we briefly introduce the different ML techniques we adopted for our experiments.

Linear Regression: We start with Linear Regression (LR), a baseline for many ML techniques. The LR models a linear relationship between the metrics we want to predict y, the dependent variables and the independent variables of predictors x as a linear function of the form:

$$\hat{y_i} = \sum_{n=1}^{N} x_i[n]\beta_n \tag{5}$$

where $x_n[1]$ represents the intercept and the remaining features represent the feature space of the predictors. The variable $\hat{y_i}$ represent the estimates for y_i. The model coefficients, which we use for prediction, are β_n where $n = 1, \ldots N$. LR computes the coefficients that minimize the residual sum of squares (RSS).

Ridge Regression (RR): In a second approach, we use Ridge Regression (RR), a variant of the LR which includes an ℓ_2-norm loss function on the coefficients, and maintains a small amount of energy in each coefficient [5]. RR seeks model coefficients that best suit the data by minimizing the RSS in the LR equation with the addition of a regularization parameter $\lambda \sum_n \beta_n^2$, where $\lambda \geq 0$.

Least Absolute Shrinkage and Selection Operator (LASSO): We then apply the LASSO, another LR variant that imposes an ℓ_1-norm on the regression coefficients. Similar to the RR, the LASSO solves the LR model with the addition of a regularization parameter $\lambda \sum_n |\beta_n|$, where $\lambda \geq 0$. In contrast to RR, the ℓ_1-norm in LASSO performs a form of automatic variable selection and continuous shrinkage by forcing some model coefficients to zero and effectively *turning-off* some features. The feature space we examine is a high dimensional one and LASSO is known to obtain sparse linear models in such cases [6].

Elastic Net (EN): We apply the Elastic Net (EN) model owing to its ability to perform shrinkage and variable selection just like the LASSO method. The LASSO is known to suffer from poor prediction accuracy due to high correlations between the features especially in cases where the number of observations are greater than the feature space, as is the case in our data set [7]. The EN combines both the LASSO and RR, that is, it applies a mixture of ℓ_1-norm and ℓ_2-norm penalties on the coefficients. The EN automates the choice of the regularization parameter and produces sparse solutions just like the LASSO. However, the EN tends to select more features than the LASSO does as the EN overcomes the grouping effect situation in LASSO. The grouping effect is a situation where the LASSO tends to select only one feature from a group of features with high pairwise correlations.

Random Forest (RF): We apply a non-linear method, the Random Forest algorithm, which is an ensemble method which builds multiple decision trees and consolidates their results to obtain stable and more accurate predictions. In simple terms, the RF estimates $\hat{y}$ for the metric y using the average predictions from a large number of regression trees [8].

3 Experiments, Model Fitting and Evaluation Procedure

We perform model computations using the traces made publicly available in [2]. All four evaluation frameworks were implemented in RStudio [version1.1.423]. The traces we utilize for our experiments are the periodic-load pattern and flashcrowd-load pattern. The periodic-load contains 51043 observations for 297 features while the flashcrowd-load pattern has 275 features with 15150 observations. We start by pre-processing the data sets to remove all non-numeric and constant value features. Using the ML techniques above we learn models to predict the service-level metrics ABR, VFR and RTP using the device statistics with the un-adjusted method. We perform two sets of experiments. One set of experiments with the periodic-load trace and the second set of experiments with the flashcrowd-load trace.

Un-Adjusted Method (UA): For the UA learning, we adopt the technique used in [2], we generate the train and test data using any sample from the data regardless of the load value. We also adopt the validation set approach [5] for all model building and evaluation with 60% of the trace in the training set and 40% in the testing set. The 60/40 split was done for both traces with 60% set aside for training the models and 40% for test data and prediction. Using the

UA approach, we learn LR, RR, LASSO and EN models for the service-level metrics.

The regularization techniques, RR, LASSO and EN required a method for selecting the regularization parameter, λ, for the penalty function. RR, LASSO and EN use an ℓ_2-norm, ℓ_1-norm and combination of both norms as a weighted (by λ) penalty term. The entire path for the results for these models was calculated using path-wise cyclical coordinate descent algorithms. Computationally efficient and effective approaches for evaluating these convex optimization problems were implemented using the glmnet package in R.

To obtain the value of λ, we employed 10-fold cross-validation (CV) approach for both learning approaches. This value was used in subsequent learning and prediction experiments. The 10-fold CV was implemented for training the models and during testing using both traces. Different values for λ were determined for both the UA and LA algorithms. A sequence of values between 0.0001 and 1 was selected and cross-validation applied to select optimal values for the regularized models. EN outperformed the other three linear methods evaluated and was adopted for our model performance comparison between the LA and UA models.

Load-Adjusted Method (LA): We obtain subsets of the entire data set for which the load value, the TCPSCK, is fixed and has more than 500 samples. To ensure that we have enough data to split between train and test samples, we use the top subsets with the most samples in them for the LA learning. Using the validation set approach, we divide the traces into two; 60% of the trace in the training set and 40% in the test set. We apply the same cross-validation procedures used for the UA models to find the best λ values for the EN algorithm using the LA method. We learn EN models for each subset of the data based on the TCPSCK value. We then apply EN to the traces to learn UA prediction models for VFR, ABR and RTP packet rate. We refer to these models as Un-adjusted Elastic Net (UA-EN) models. We learn Load-adjusted Elastic Net (LA-EN) models for our service level metrics with the same number of samples as was done for the UA-EN models.

Using the same samples used for the UA-EN and LA-EN models, we then apply Random Forest to learn Load-Adjusted Random Forest (LA-RF) and Un-adjusted Random Forest (UA-RF) models for the service-level metrics.

All models are evaluated in terms of two accuracy measures. The first is the Root Mean Squared Error (RMSE), computed as $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$. The best model is the model with the lowest RMSE. The second measure is the R-squared which is essentially a statistical measure that explains the goodness of fit of our regression models. R-squared achieves this by comparing our regression models with a baseline model, one which is simply the average of of observed responses of the dependent feature. The R-squared is computed as $R^2 = 1 - \frac{SSE}{SST}$ where the Sum of Squared Errors (SSE) in our model is computed as $\sum_{i=1}^{n}(y_i-\hat{y}_i)^2$ and the Sum of Squared Total (SST) of our baseline model is computed as $\sum_{i=1}^{n}(y_i-\bar{y}_i)^2$. The model with the highest R-squared value is the best, and a model with an R-squared value of 1 is a perfect model. We report only the test RMSE and R-squared values.

4 Results

We present three results: (1) We compare the performance of the four linear models we examined using the UA approach. (2) We examine the performance of the best performing linear model with the non-linear model, the Random Forest algorithm. (3) Finally, we compare the performance of our LA models with the UA approach.

UA Linear Models: Table 2 lists the performance of the linear models LR, RR, LASSO and the EN on the test data for both the Periodic-load and Flashcrowd-load traces using the UA technique.

Table 2. RMSE, R Squared for Video Frame Rate, Audio Buffer Rate and RTP Packet rate for UA Models. For both traces, Elastic Net produces the best RMSE and R Squared results for the linear models (bold font); Random Forest produces the best results (bold font) out of all models evaluated.

		ABR		VFR		RTP	
Un-adjusted Method	Model	RMSE	R^2	RMSE	R^2	RMSE	R^2
Periodic-load Trace	Linear Regression	14.75	42.01	3.27	63.34	28.09	82.63
	Ridge Regression	16.70	28.99	3.31	58.58	28.76	80.36
	LASSO	14.19	53.30	3.18	63.80	28.09	82.98
	Elastic Net	**14.03**	**55.28**	**2.99**	**64.20**	**25.82**	**83.63**
Non-linear	**Random Forest**	**4.42**	**95.44**	**1.95**	**84.58**	**22.74**	**87.59**
Flashcrowd-load Trace	Linear Regression	11.26	63.20	2.67	73.82	29.73	85.03
	Ridge Regression	11.96	58.48	2.73	72.67	31.49	84.13
	LASSO	11.36	63.16	2.66	74.27	29.74	85.56
	Elastic Net	**11.17**	**64.37**	**2.62**	**74.76**	**29.64**	**85.93**
Non-linear	**Random Forest**	**4.76**	**94.37**	**2.09**	**84.70**	**25.53**	**89.63**

1. The EN gives the best result out of all four linear models for both traces. The RMSE for all three metrics predicted is lowest for the EN and the R-squared is highest for the EN models in both traces (boldfont in table).
2. The EN performance is closely matched by the LASSO and the LR for both traces across all three metrics. The EN offers the best prediction accuracy due to its ability to overcome the limitations of LASSO by automatically tuning the loss function based on the data. The results indicate that the EN does well in both traces but seems to offer lower RMSE values for the VFR and ABR using the Flashcrowd trace.
3. The RR algorithm offers the worst predictions across all three metrics for both traces.

4. The performance of the Random Forest algorithm using the UA approach is listed in Table 2 for both load traces. The RF algorithm offers a big improvement in RMSE and R-squared over the EN. The RF performance gain over the indicates that non-linear methods perform significantly better than linear models as can be inferred from the results.

Comparison of LA and UA: We have listed results for EN and RF models learned using the LA and UA approaches in Table 3.

Table 3. Load-adjusted Technique Versus Un-adjusted Technique; LA techniques offer better estimates than the UA approach for both traces. The RF produces more accurate predictions than the EN using the LA technique.

		ABR		VFR		RTP	
Trace	Method	RMSE	R^2	RMSE	R^2	RMSE	R^2
Periodic-load	**Load-adjusted-EN**	**12.03**	**65.84**	**3.08**	**66.54**	**30.51**	**83.64**
	Un-adjusted EN	12.10	57.53	3.28	63.36	36.47	73.16
	Load-adjusted RF	**6.46**	**92.52**	**1.79**	**86.66**	**28.29**	**84.29**
	Un-adjusted RF	9.49	72.89	2.11	85.68	33.62	80.62
Flashcrowd-load	**Load-adjusted EN**	**8.38**	**84.94**	**2.20**	**85.82**	**23.94**	**80.32**
	Un-adjusted EN	9.79	65.04	2.71	78.84	25.66	74.33
	Load-adjusted RF	**4.12**	**97.11**	**1.88**	**89.09**	**20.55**	**87.57**
	Un-adjusted RF	4.40	94.07	1.89	78.89	29.55	82.11

1. The LA models for EN and RF outperform the UA models for both the Periodic-load and Flashcrowd-load traces across all three service-level metrics. The LA-EN estimates are over 5 audio buffers/second better than for the UA-EN in both load traces; the LA-EN estimates for VFR and RTP indicate similar improvements in both load traces. The LA-EN offers similar performance gains for both traces.
2. The RF algorithm offers better RMSE and R-squared values for the LA technique than the LA-EN. The LA-RF shows a big improvement in estimates. We compare the average RTP prediction to illustrate what the RMSE values imply. For instance, the LA-RF estimates for the RTP using the Flashcrowd trace indicate that the values lie between 90 to 449 RTP packets. The UA-RF estimates for the same trace offer estimates between 8 to 347 RTP packets/second. True RTP values lie between 83 to 545 RTP packets/second. Expressed in percentages, the average improvement in prediction performance is $\approx 50\%$ to $\approx 60\%$ better for the LA RF learning.
3. The LA models in the Flashcrowd-load trace offer better prediction metrics than the Periodic-load. Fig. 3 illustrates the accuracy of the LA predictions for the RTP packet/second for both traces for comaprison with the UA predictions.

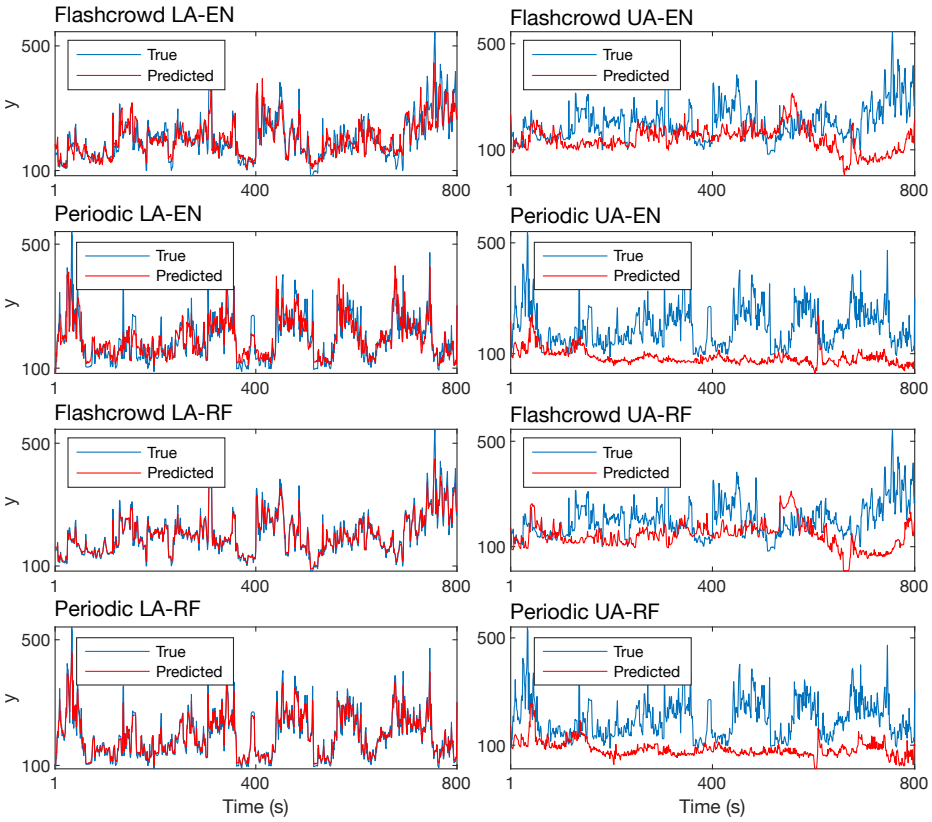

Fig. 3. Accuracy of RTP Packet Count Predictions ($\hat{y}_i$): The true RTP packet count received by a client is illustrated for 800 seconds when the system is under a flashcrowd and then a periodic load. These counts are then predicted using Load-Adjusted (LA), in column 1 (c1), and UnAdjusted (UA), in c2, learning algorithms, e.g. the Elastic Net (EN) and Random Forest (RF). Rows 1 (r1) and r2 compare Load-Adjusted Elastic Net (LA-EN) learning with UnAdjusted Elastic Net (UA-EN) learning on the flashcrowd trace (r1) and the periodic trace (r2). R3 and r4 compare Load-Adjusted Random Forest (LA-RF) learning with UnAdjusted Random Forest (UA-RF) learning on the flashcrowd trace (r3) and the periodic trace (r4). We draw the conclusion that LA learning (c1) is always more accurate than UA learning (c2). Secondly, RFs (r3 and r4, c1) are more accurate than ENs (r1 and r2, c1).

5 Related Work

Yanggratoke et al.in [2] applied Machine Learning using a UA approach for service-level prediction from cloud-hosted device statistics. We have demonstrated that it is possible to improve the accuracy of the predictions achieved in [2]. Similarly, the authors of [9] investigated the problem of service-level estimation using ML for another cloud hosted service, Voldemort, a Key-Value store. We posit that our LA approach may also work in this scenario.

The authors of [10] applied a signal processing approach for prediction of service-level metrics from cloud-based device statistics. In their work, the authors developed an initial system load model to aid subsequent service-level prediction. This technique is called load-adjusted learning. It provides the foundation for the approach undertaken in this paper. The load-adjusted technique trains prediction weights conditional on the load value. The work in [10] was limited to regression models. We have also demonstrated that Random Forest models give better predictions when load adjusted.

Our results are of relevance to networking professionals. Our load adjusted approach is computationally cheap than UA learning. We consider subset of the data based on the load value which improves prediction accuracy and reduces computation. From the perspective of the network service provider, the authors of [11] evaluated monitoring of the quality of a compressed video transmitted in a lossy packet network using bitstream measurements only. In their work, they adopted the Mean Squared Error (MSE) as an estimation of the video quality. They examined three different techniques for MSE estimation NoParse (NP), QuickParse (QP) and FullParse (FP). The FP method extracts detailed information regarding effects of packet loss on the video; the QP method is only concerned with extracting high-level details about the video bitstream quality and as a result requires less computational time than the FP. The NP method estimates the MSE based using network-level measurements only. They concluded that the FP was the most accurate of the methods examined. In a practical network system spanning multiple Internet Service Providers over a broad geographical area, there may be instances when there are no available measurements for in-network video quality estimation except for the packet loss rate and bitrate; in such cases, the NP could be a handy tool. However, our LA approach using readily available device statistics of the server(s) delivering the video resources can learn the client video without any detailed knowledge of the system or the video.

There is a significant momentum behind in the concept of Software Defined Networks (SDN) which will lay the foundation for our future work, particularly, how we will deploy our learning engine. The authors of [12] proposed an approach to measure different video Quality of Experience (QoE) metrics running on client's devices in order to improve QoE. They also explored the possibility of dynamic routing of requests or designation of best available delivery node based on predetermined network conditions. Using a light-weight plugin they created for HTML5 video player, they were able to monitor various QoE factors (e.g. buffering state and video resolution at target). With these they were able to analyze user-perceived experience while using the video is streaming. This setup points us towards how we might extend our testbed.

The LA improvements in performance makes no additional assumptions about the data. We will investigate how weakened forms of the independence assumption made by these models can improve prediction.

6 Conclusion

We introduced a method for improving predictions of service-level metrics using a load adjusted learning technique. We provided evidence that the EN algorithm provides the best prediction performance among the linear regression variants using the baseline UA approach. We also presented evidence which shows that the LA learning algorithm improves the UA prediction performance for all three metrics under study. We further demonstrated that the Random Forest predictions outperform the EN estimates using the load adjusted approach. The LA method offers significant improvements in the prediction accuracy and reduces the computational requirements of the system delivering the resources.

Acknowledgement. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the Grant Number 15/SIRG/3459.

References

1. Cisco Systems 2017. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021 White Paper. *Cisco*, 2017.
2. R. Yanggratoke, J. Ahmed, J. Ardelius, C. Flinta, A. Johnsson, D. Gillblad, and R. Stadler. Predicting real-time service-level metrics from device statistics. In *IFIP/IEEE Int. Sym. on Int. Net. Man. (IM)*, 2015.
3. R. de Fréin. Effect of system load on video service metrics. *IEEE Irish Signals & Systems Conference*, pages 1–6, 2015.
4. R. de Fréin. Take off a load: Load-adjusted video quality prediction and measurement. In *IEEE Inter. Conf. on Comp. and IT*, pages 1886–1894, Oct 2015.
5. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer New York Inc., New York, USA, 2001.
6. R. Tibshirani. Regression Shrinkage and Selection Via the LASSO. *J. Roy. Stat. Soc. Series B (Methodological)*, 58(1):267–288, 1996.
7. H. Zou and T. Hastie. Regularization and variable selection via the Elastic-Net. *J. Roy. Stat. Soc.*, 67(2):301–320, 2005.
8. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.
9. R. Stadler, R. Pasquini, and V. Fodor. Learning from network device statistics. *J. Netw. Syst. Manage.*, 25(4):672–698, October 2017.
10. R. de Fréin. Source separation approach to video quality prediction in computer networks. *IEEE Comm. Let.*, 20(7):1333–6, Jul. 2016.
11. A. R. Reibman, V. A. Vaishampayan, and Y. Sermadevi. Quality monitoring of video over a packet network. *IEEE Trans. on Multim.*, 6(2):327–334, April 2004.
12. H. Nam, K. Kim, J. Y. Kim, and H. Schulzrinne. Towards QoE-aware video streaming using SDN. In *2014 IEEE Globecom*, pages 1317–1322, Dec 2014.