Machine Learning in Cloud Environments Considering External Information

Matthias Lermer, Stefan Frey, Christoph Reich
Cloud Research Lab
Furtwangen University
Furtwangen, Germany
Email: {matthias.lermer, stefan.frey, christoph.reich}@hs-furtwangen.de

***Abstract*—Machine learning applied to cloud environments can lead to many advantages. One example is the possibility of improved Quality of Service (QoS) by predicting future workloads and reacting dynamically with automated scaling. In reality however, there are cases where the use of machine learning algorithms is not as efficient as imagined. One current problem is the disregard of external information, whose inclusion could help to create better models of the reality. The approach of this paper shows that different machine learning algorithms like Neural Networks (NN), Support Vector Machines (SVM) and Linear Regression can be successfully used to predict the response time of Virtual Machines (VM) within cloud environments. The performed application of those algorithms to different cloud usage scenarios and following evaluation enables to gain insight into the strengths and weaknesses of each algorithm. Furthermore, a work in progress architecture is proposed to deal with the two big challenges, inclusion of external information and handling live data streams.**

Keywords–Machine Learning; Support Vector Machines; Neural Network; Linear Regression; Cloud Computing; SLA.

I. INTRODUCTION

The use of machine learning, a relatively mature and established discipline of computer science, has become more important than ever. Various challenges in the area of Cloud Computing, like efficiently handling big data or the realization of green IT, can be tackled by applying machine learning algorithms. This paper looks at the specific application of response time prediction of Virtual Machines (VM), in order to improve scaling functionality and prevent Service Level Agreement (SLA) violations. The associated implication on the utilization of resources when using different machine learning algorithms is not covered in this evaluation. The remainder of the paper is organized as follows: In Section II a short explanation about the used CloudSim framework and created scenarios is given. Section III describes the application and evaluation of various machine learning algorithms. In Section IV an architectural approach to include external informations during the learning process is presented. Section V completes the paper by drawing a conclusion and suggesting future work.

II. RELATED WORK

Similar research with different focus has been conducted in the past for the use of machine learning in cloud environments. Prevost et al. used a Neural Network (NN), as well as a Linear Predictor [1] to anticipate future workloads by learning from historical URL requests. Although both models were able to give efficient predictions, the Linear Predictor was able to predict more accurately. Li and Wang proposed their modified Neural Network algorithm nn-dwrr in [2]. The application of this algorithm led to a lowered average response time compared to application of traditional capacity based algorithms for scheduling incoming requests to VMs. In similar research Hu et al. [3] have shown that their modification of a standard Support Vector Regression (SVR) algorithm can lead to an accurate forecasting of CPU Load what can be used to achieve a better resources utilization. Another algorithm, which is renowned for providing good results in similar scenarios, is Linear Regression. Although the results are often weaker compared to Neural Networks or Support Vector Machines (SVM) in cases of workload prediction [4] [5], the fast training and deployment time of models built with Linear Regression should not be underestimated.
Those examples show that there are a variety of optimization challenges in cloud environments which can be tackled by applying machine learning algorithms. What separates the current work from previous research is a detailed examination of specific characteristics of three different machine learning algorithms and presenting the results in a visual way. The choice to evaluate Neural Networks, Support Vector Machines and Linear Regression was made because those algorithms earned promising results in previously conducted research.

III. CLOUDSIM

The CloudSim framework [6] [7], developed by the University of Melbourne provides the means to realistically simulate Cloud Computing environments. An extended implementation of this framework was used to simulate specific scenarios in order to obtain relevant log data. This data is used to train and test models with different machine learning algorithms. Furthermore the additions made by the CloudSimEx extension [8] were used.

With the help of additional modifications of the CloudSim source code it was possible to simulate and log the 3 following Cloud usage scenarios:

1) Short bursts of peak requests in the average usage area
2) Slow ascending and descending requests with one larger peak
3) Quick changes in requests with small peaks, followed by medium and large sized peaks

IV. APPLICATION AND EVALUATION OF MACHINE LEARNING ALGORITHMS

In order to apply and evaluate different Machine Learning algorithms, the open source software RapidMiner [9] was used. The log files created by the CloudSim application were utilized as training and test sets. Furthermore, the available Series extension provided by RapidMiner was used. This extension enables an efficient way to quickly replace different

 ISBN: 978-1-61208-477-0

Machine Learning algorithms during the process of creating and evaluating a model in regard to ordered time series. With the help of a horizon of h=20 (2 seconds), it was defined that the learning algorithms gets to learn the next h time steps in order to be able to predict the value of the average response time of h+1. After the prediction, the time window is incremented by 1 and the next value gets predicted. A further advantage is that the data is transformed by the implemented Series operator in a way that enables the use of classification algorithms like Neural Networks and Support Vector Machines in the case of a numerical regression problem.

A. Configuration of the algorithms

RapidMiner provides a large number of configuration parameters which can be tuned. After the execution of test runs with different parameters the following configuration provided the best results:

1) Neural Network: |Feed forward NN, training back propagation |Hidden layers: 8 |Training cycles: 1000 |Learning rate: 0.3 |Momentum: 0.2 |Decay: true |Normalize: true |Error epsilon: 0.00001

2) Support Vector Machines: |Kernel Type: radial |Kernel Gamma: 1.0 |Kernel cache: 200 |C: 0 |Convergence epsilon: 001 |Max iterations: 10000 |Scale: true |L pos: 1.0 |L neg: 1.0

3) Linear Regression: |Feature selection: Iterative T-Test |Max iterations: 1.0 |Forward alpha: 0.05 |backward alpha: 0.05 |eliminate colinear features: true |min tolerance: 0.05 |use bias: true

B. Graphs

The following graphs show the aforementioned scenarios (see Section III). The x-axis represents the time in seconds. The y-axis to the left indicates the response time of the cloud environment with the differentiation in predicted values (red line) and actual values (black line). Whereas the y-axis to the right indicates the current number of active VMs. Furthermore, the active VMs are highlighted in a light blue in the background.

C. Scenario 1

1) Neural Network: Figure 1 shows that the NN overestimates the peak of the burst load in every case. It can be seen that the difference between predicted peak and real peak is the biggest during the first burst and that there is an improvement when predicting the later peaks of the bursts. The briefly following decline and rise after each peak, e.g., during sec 8-15 is respectively underestimated and overestimated but it can clearly be seen that there is an improvement in the last iteration. Figure 2 shows the delay characteristics and that the algorithm in general can adapt well to the problem.

2) Support Vector Machines: Figure 3 shows a contrast to the NN algorithm. In the case of SVMs, the first peak of each burst is underestimated. The following cooldown phase before the second peak of each burst is overestimated but an improvement over time can be seen, especially on the last burst. Figure 4 looks specifically at the first burst and a comparison to the NN 2 makes it clear that SVMs predict a more smooth curve. It should be kept in mind that it is realistic to assume that in real life there are scenarios with different requirements regarding the reaction to those predictions where this specific differences, smooth or rough, could be seen as either an advantage or disadvantage.

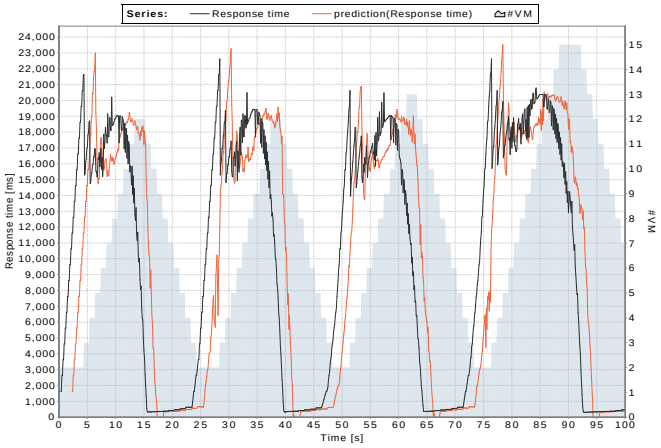

Figure 1. NN Scenario 1: 0s-100s

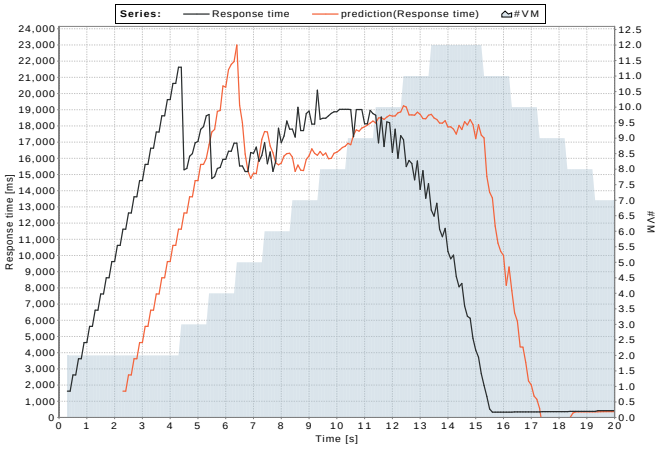

Figure 2. NN Scenario 1: 0s-20s

3) Linear Regression: The predictions made with the help of a Linear Regression model, shown in Figure 5, seem to be very similar to those predictions made by the SVM in Figure 3. This insight is further substantiated by taking a closer look at the bursts in Figure 6 and Figure 4 where a similar prediction pattern can be seen. Worth mentioning is that the predictions made by Linear Regression lead to an even smoother curve compared to the curve predicted by the other 2 algorithms.

D. Scenario 2

1) Neural Network: When looking at the overview in Figure 7 it is demonstrated that moderate changes in response times are learned rather well. The interesting part, shown in more detail in Figure 8, showcases the nature of overestimation. Again, the peak is overestimated, but the predicted curve recovers very fast and yet this issue occurs again after the second plateau. It should be noted that with a different configuration of the NN algorithm a very different curve can be predicted. For this paper we looked at a specific configuration of the algorithm because this characteristic can be utilized and will be explained during the comparison of the algorithms.

2) Support Vector Machines: The overview shown in Figure 9 displays the capability of the algorithm to be able to adapt to a singular, steadily climbing response time. The prediction

 ISBN: 978-1-61208-477-0

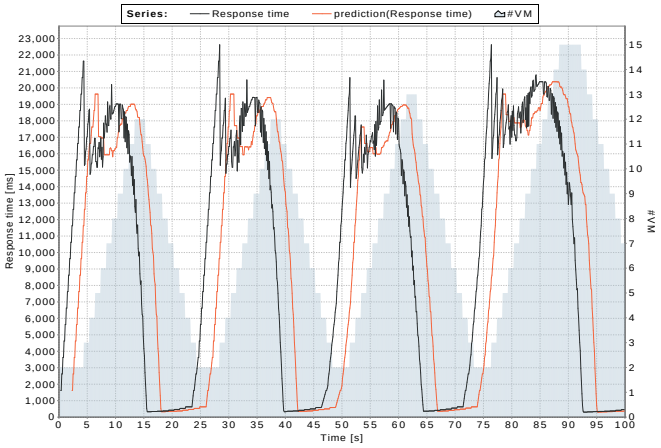

Figure 3. SVM Scenario 1: 0s-100s

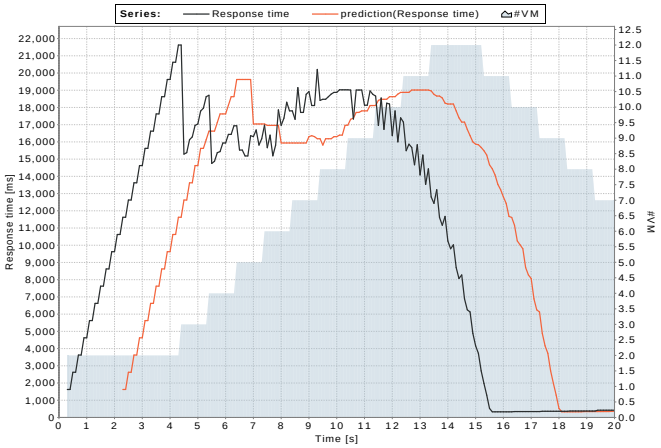

Figure 4. SVM Scenario 1: 0s-20s

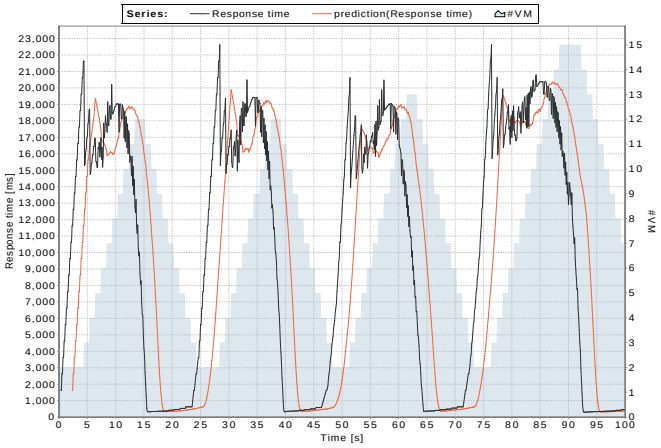

Figure 5. Linear Regression Scenario 1: 0s-100s

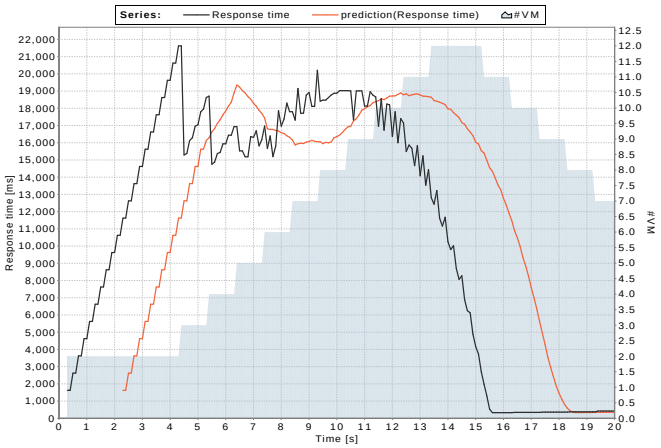

Figure 6. Linear Regression Scenario 1: 0s-20s

during the first phase (0-60s) is handled well by the algorithm. Figure 10 illustrates that the spontaneous and large decline in response time is learned very well. This is an important characteristic as predictions based on those quick changes could be the focus during the application in real-time scenarios. None of the other algorithms is able to predict scenario 2 this precisely.

3) Linear Regression: Figure 11 shows again great similarity between predictions made with the help of Linear Regression and SVMs. Again the difference is that the ascent of the curve is predicted in a smoother way. Additionally, the spontaneous and large decline seen in Figure 12 is not predicted very well. The same characteristic applies on the following smaller decline. As a conclusion it can be said that in this specific scenario the model trained by Linear Regression is the weakest.

E. Scenario 3

1) Neural Network: Figure 13 shows that during the first phase (0-100s) the model trained by a NN has minor problems in predicting the response time. Although the peaks are generally predicted well, sometimes they are underestimated and sometimes overestimated. But in contrast to the other algorithms the difference in error margin is very small in most cases. During the recovery times after each slope, the local minima are overestimated almost in every case. While the first big peak of a response time over 42000ms is overestimated as well, the second one is predicted almost perfectly. The relative smooth slopes before, during and after the larger peaks are predicted very well with no prominent deficit.

2) Support Vector Machines: Figure 14 shows that a model trained by SVMs can predict the response time for a varying scenario rather well. The occurring peaks during the first phase (0-100s) are underestimated in every case, but not to a large degree. This leads likewise to the underestimation of the recovery times after each peak, which are the consequence of adding and deleting Virtual Machines. The two larger peaks with a response time of over 42000ms are underestimated again by a small margin while the relative smooth slopes before, in between and after are learned well with no prominent deficit in their prediction.

3) Linear Regression: Figure 15 shows that a model trained by Linear Regression can cope well in a varying scenario. Similar to the SVM model it slightly underestimates the response time in the first phase (0-100s). In general, it can be said that those models are very similar and have only minor, negligible differences. The main difference is that the use of

 ISBN: 978-1-61208-477-0

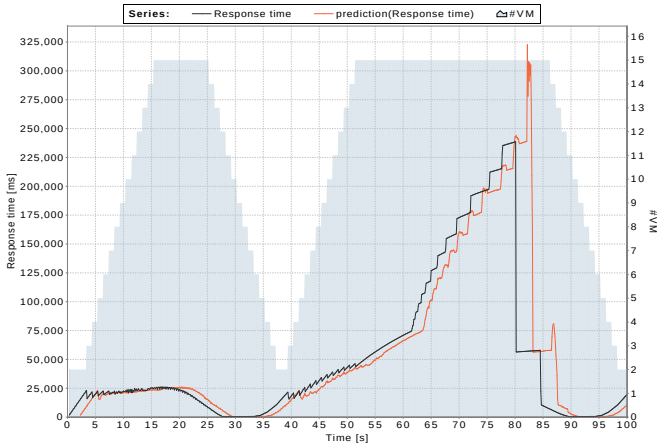

Figure 7. NN Scenario 2: 0s-100s

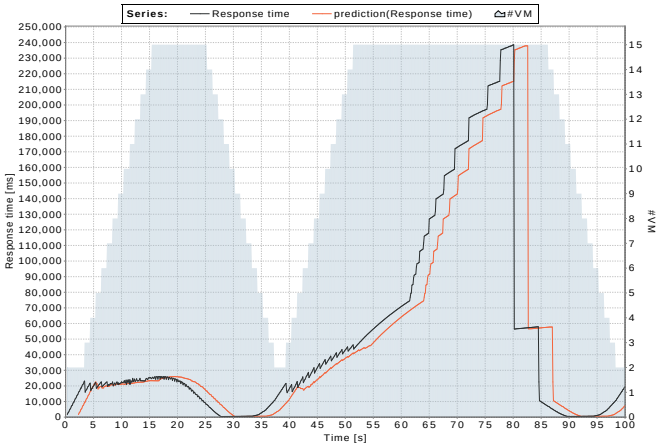

Figure 9. SVM Scenario 2: 0s-100s

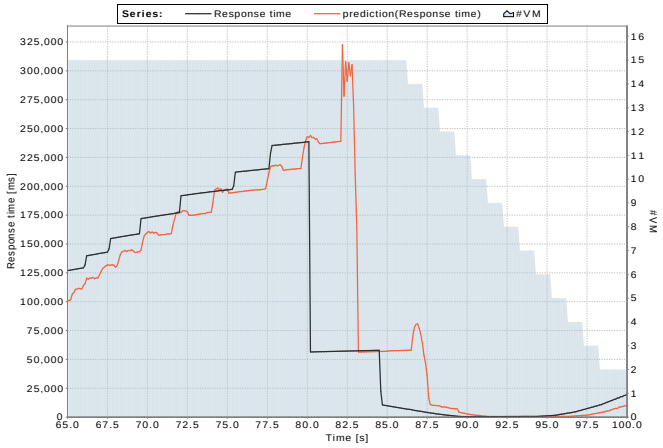

Figure 8. NN Scenario 2: 65s-100s

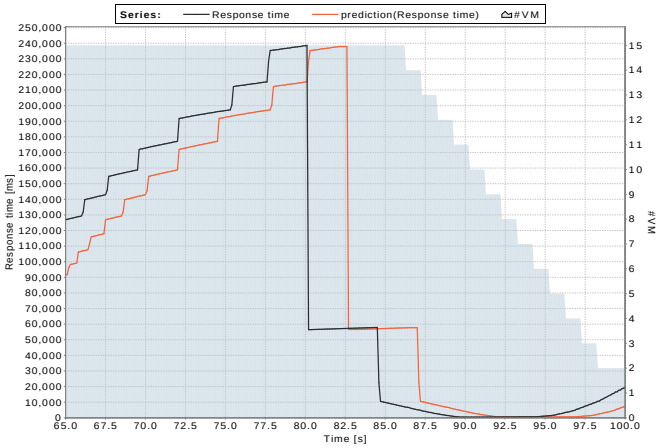

Figure 10. SVM Scenario 2: 65s-100s

Linear Regression leads to smoother slopes.

F. Comparison

While it was shown that all 3 algorithms can be effectively used for predicting the response time in different scenarios it can be said that the NN has a minor advantage over the other algorithms. The main reason is that the NN, in general, slightly overestimates and almost never underestimates the response time. The practical application of this knowledge, e.g., using those predictions in combination with a scaler who manages the quantity of VMs leads to a more assuring state that requirements like defined SLAs can be covered more carefully than with other algorithms. In less critical business-cases, where the defined SLAs and response times are not that sensitive, the other 2 algorithms, SVMs and Linear Regression, can be used despite their tendency to slightly underestimate response times. Especially the Linear Regression with its fast training and deployment times could be considered in near real-time scenarios.

G. Related Work: Fuzzy

A similar research has been conducted by Frey et al. in [10]. In their scenario the driving factor was to use predictions based on fuzzy logic to automatically scale the quantity of Virtual Machines in a Cloud Computing environment in order to be able to guarantee that a certain threshold regarding response times is not exceeded. While the paper presented here takes a more general approach, Frey et al. have successfully proven that a model trained by fuzzy logic can predict response times well and that the thereby gained knowledge can be successfully applied in a real-time scenario.

V. Addition of External Sources

In order to be able to enrich data by external information, the process and work flow has to be defined and created. The following Figure 16 provides an overview concerning that matter. In the following scenario it is presumed that all steps of the Cross Industry Standard Process for Data Mining (CRISP-DM) [11] Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation and Deployment have been traversed at least once. As a result, a working system was established but after one or more evaluations it becomes clear that there are possibilities to create a better model by considering the use of appropriate external data sources. Those can enrich the existing historical training data and provide the ability to dynamically adapt the specific or general needs of a good model. This can be realized by the use of specific kinds of agents. Polling agents are responsible for the following tasks:

 ISBN: 978-1-61208-477-0

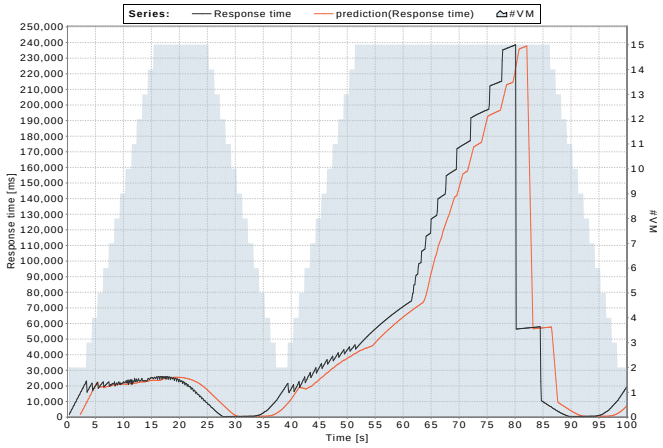

Figure 11. Linear Regression Scenario 2: 0s-100s

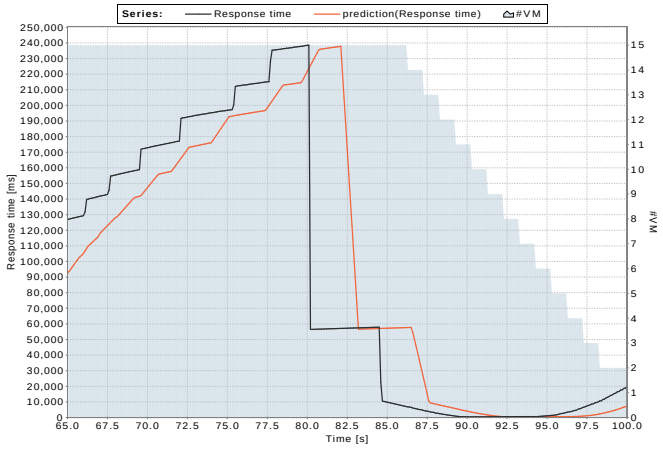

Figure 12. Linear Regression Scenario 2: 65s-100s

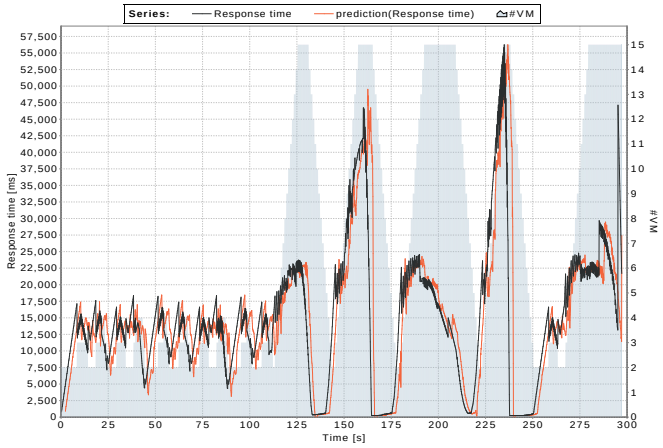

Figure 13. NN Scenario 3: 0s-300s

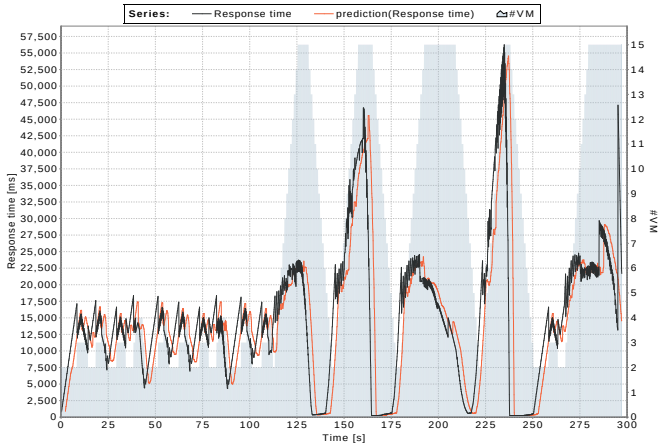

Figure 14. SVM Scenario 3: 0s-300s

- Retrieve specified information from described sources.
- Cope with missing values in a defined way, as it is not always the case that information can be gathered consistently.
- Recognize and filter wrong and erroneous data, which is especially important in cases where the external sources are filled with data collected by humans and not computers.
- Transform and correlate the gathered information and attributes to the training set, e.g., timestamp synchronization.
- Store the prepared information in the training database and thus enrich existing historical data.

Configuration agents must be able to realize the following:

- Query the polling agents about meta information
- Use this meta information to change the configuration of the training process.
- Initiate new training sessions after defined periods, as well as after enrichment of the training set.
- Initiate the application and validation of models while storing the results in a database.

If a scenario has the need for evaluating live data streams there must be a coordinating agent who has the task to react in a defined way. The application shown in Section III gives an example how a live data stream could be integrated. The knowledge gained by machine learning algorithms could be used to automatically scale an appropriate amount of VMs in order to never exceed a certain response time. Furthermore, the additional knowledge gained by training different models with various machine learning algorithms can be seen as external information. This information about the strength and weakness of each algorithm can be exploited. For example it can be declared that in critical business cases, where the transgression of response times is inevitably paired with costly SLA violations, the use of the NN algorithm, which predicts in a more cautious way by overestimating response times, could be prioritized. One possibility to fully realize this potential would be to offer classifications of individual SLAs in gold, bronze and silver. In this example the knowledge and application of the different algorithms could be used by a scaler regarding the Service Level Objective of the SLA ”Maximum Response Time of service X shall not surpass Y ms”. The use of the Neural Network could be set up by an coordination agent for gold customers whereas the use of the weaker but less expensive Linear Regression could be considered for bronze customers.

 ISBN: 978-1-61208-477-0

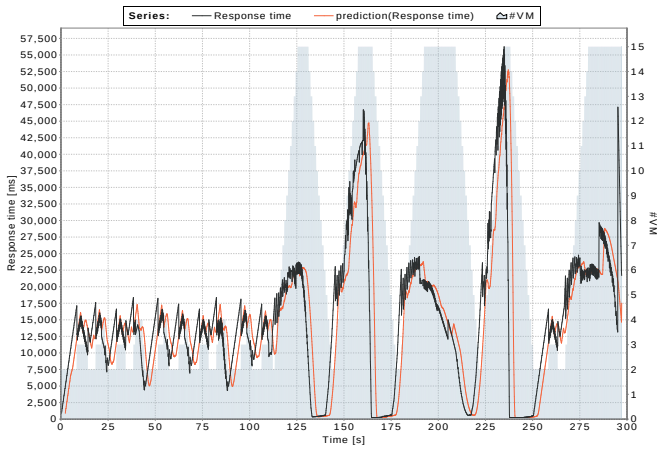

Figure 15. Linear Regression Scenario 3: 0s-300s

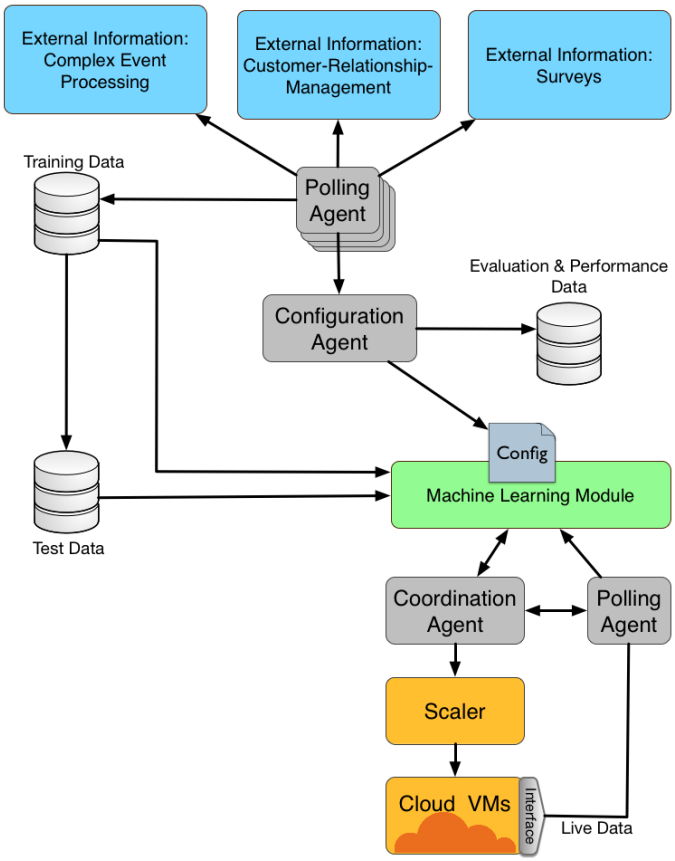

Figure 16. Architecture

This is just one example to show the synergy of the proposed architecture with the evaluated application of machine learning learning techniques.

One of the biggest challenges is without a doubt the inclusion and evaluation of external information which a model has never seen before. The correlation between historical and current information has to be established. This is no easy task as the problem starts already at the often needed transformation and preprocessing of the data in order to be able to train a model in the first place. The implementation of the architecture proposed in Figure 16 would enable a step to tackle this problem. But the next problem waits just around the corner. The evaluation of models, especially if the use of live data streams is involved. This is a current research problem and first proposals for solutions are presented by de Faria et al. in [12]. Although there is a noticeable progress in this area of expertise in general, it is still a long way from being able to provide a general approach and methodology.

VI. CONCLUSION AND FUTURE WORK

The aim of this paper was to show how selected machine learning algorithms cope with the prediction of response times in a cloud environment. Three different cloud usage scenarios were defined and three different algorithms (NN, SVM, Linear Regression) were applied. Knowledge about specific strengths and weaknesses about each algorithm was gained in the process. The general conclusion is that although each of the algorithms can be used for predicting response times effectively, some show specific characteristics which can be exploited. Additionally, an architecture was proposed in order to be able to deal with external information in an efficient way. Future work is to examine more algorithms with different configurations and scenarios in regard to response time. Furthermore, those results shall be substantiated by application on real cloud environments. Also, the creation of a framework of the architecture, proposed in Section IV is planned.

REFERENCES

[1] J. J. Prevost, K. Nagothu, B. Kelley, and M. Jamshidi, “Prediction of cloud data center networks loads using stochastic and neural models,” in System of Systems Engineering (SoSE), 2011 6th International Conference on, June 2011, pp. 276–281.

[2] C.-C. Li and K. Wang, “An SLA-aware load balancing scheme for cloud datacenters,” in Information Networking (ICOIN), 2014 International Conference on, Feb 2014, pp. 58–63.

[3] R. Hu, J. Jiang, G. Liu, and L. Wang, “KSwSVR: A New Load Forecasting Method for Efficient Resources Provisioning in Cloud,” in Services Computing (SCC), 2013 IEEE International Conference on, June 2013, pp. 120–127.

[4] A. Bankole and S. Ajila, “Predicting cloud resource provisioning using machine learning techniques,” in Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on, May 2013, pp. 1–4.

[5] M. Imam, S. Miskhat, R. Rahman, and M. Amin, “Neural network and regression based processor load prediction for efficient scaling of Grid and Cloud resources,” in Computer and Information Technology (ICCIT), 2011 14th International Conference on, Dec 2011, pp. 333–338.

[6] Cloudbus.org, “The CLOUDS Lab: Flagship Projects - Gridbus and Cloudbus,” Available: http://www.cloudbus.org/cloudsim, [retrieved: 04, 2016].

[7] R. N. Calheiros, R. Ranjan, A. Beloglazov, A. F. De Rose, and R. Buyya, “CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms,” Softw. Pract. Exper., vol. 41, no. 1, Jan. 2011, pp. 23–50, http://dx.doi.org/10.1002/spe.995, [retrieved: 04, 2016].

[8] Nikolay Grozev et al., “Cloudslab CloudSimEx,” Available: https://github.com/Cloudslab/CloudSimEx, [retrieved: 04, 2016].

[9] RapidMiner, “RapidMiner - #1 Open Source Predictive Analytics Platform,” Available: https://rapidminer.com, [retrieved: 04, 2016].

[10] S. Frey, C. Luthje, C. Reich, and N. Clarke, “Cloud QoS Scaling by Fuzzy Logic,” in Cloud Engineering (IC2E), 2014 IEEE International Conference on, March 2014, pp. 343–348.

[11] P. Chapman et al., “CRISP-DM 1.0 Step-by-step data mining guide,” August 2000, Available: https://the-modeling-agency.com/crisp-dm.pdf, [retrieved: 04, 2016].

 ISBN: 978-1-61208-477-0

[12] E. Ribeiro de Faria, I. Ribeiro Goncalves, J. Gama, and A. Carlos Ponce de Leon Ferreira Carvalho, “Evaluation of Multiclass Novelty Detection Algorithms for Data Streams,” IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 11, Nov 2015, pp. 2961–2973.

 ISBN: 978-1-61208-477-0