Journal of Cloud Computing: Advances, Systems and Applications

RESEARCH Open Access

An autonomic prediction suite for cloud resource provisioning

Ali Yadavar Nikravesh, Samuel A. Ajila* and Chung-Horng Lung

Abstract

One of the challenges of cloud computing is effective resource management due to its auto-scaling feature. Prediction techniques have been proposed for cloud computing to improve cloud resource management. This paper proposes an autonomic prediction suite to improve the prediction accuracy of the auto-scaling system in the cloud computing environment. Towards this end, this paper proposes that the *prediction accuracy of the predictive auto-scaling systems will increase if an appropriate time-series prediction algorithm based on the incoming workload pattern is selected*. To test the proposition, a comprehensive theoretical investigation is provided on different risk minimization principles and their effects on the accuracy of the time-series prediction techniques in the cloud environment. In addition, experiments are conducted to empirically validate the theoretical assessment of the hypothesis. Based on the theoretical and the experimental results, this paper designs a self-adaptive prediction suite. The proposed suite can automatically choose the most suitable prediction algorithm based on the incoming workload pattern.

Keywords: Cloud resource provisioning, Auto-scaling, Decision fusion technique, Structural risk minimization, Empirical risk minimization, Multi-layer perceptron, Multi-layer perceptron with weight decay, Workload pattern, Cloud computing

Introduction

The elasticity characteristic of cloud computing and the cloud's pay-as-you-go pricing model can reduce the cloud clients' cost. However, maintaining Service Level Agreements (SLAs) with the end users obliges the cloud clients to deal with a cost/performance trade-off [1]. This trade-off can be balanced by finding the minimum amount of resources the cloud clients need to fulfill their SLAs obligations. In addition, the cloud clients' workload varies with time; hence, the cost/performance trade-off needs to be justified in accordance with the incoming workload. Auto-scaling systems are developed to automatically balance the cost/performance trade-off.

There are two main classes of auto-scaling systems in the Infrastructure-as-a-Service (IaaS) layer of the cloud computing: *reactive* and *predictive*. Reactive auto-scaling systems are the most widely used auto-scaling systems in the commercial clouds. The reactive systems scale out or in a cloud service according to its current performance condition [2]. Although the reactive auto-scaling systems are easy to understand and use, they suffer from neglecting the virtual machine (VM) boot-up time which is reported to be between 5 and 15 min [3]. Neglecting the VM boot-up time results in the under-provisioning condition which causes SLAs violation. Predictive auto-scaling systems try to solve this problem by forecasting the cloud service's future workload and adjusting the compute and the storage capacity in advance to meet the future needs.

The predictive auto-scaling systems generate a scaling decision based on the future forecast of a performance indicator's value. Therefore, to improve the accuracy of the predictive auto-scaling systems, researchers have strived to improve the accuracy of the prediction techniques that are being used in the auto-scaling systems (see [4] for a comprehensive overview of the auto-scaling prediction techniques). According to [4], the most dominant prediction technique in the IaaS layer of the cloud auto-scaling domain is time-series prediction. Time-series prediction techniques use the historical values of a performance indicator to forecast its future value. Although in recent years many innovative time-series prediction techniques have

* Correspondence: ajila@sce.carleton.ca
Department of Systems and Computer Engineering, Carleton University, 1125 Colonel By Drive, Ottawa K1S 5B6, ON, Canada

been proposed for the auto-scaling systems, the existing approaches neglect the influence of the performance indicator pattern (i.e., how the performance indicator values change over time) on the accuracy of the time-series prediction techniques. This paper proposes an autonomic prediction suite using the decision fusion technique for the resource provisioning of the IaaS layer of the cloud computing environment. The proposed suite identifies the pattern of the performance indicator and accordingly selects the most accurate technique to predict the near future value of the performance indicator for better resource management. The central hypothesis in this paper that serves as the fusion rule of the prediction suite is:

The prediction accuracy of the predictive auto-scaling systems is impacted positively by using different prediction algorithms for the different cloud workload patterns

In order to lay out the theoretical groundwork of the prediction suite, this paper first examines the influence of the cloud service's incoming workload patterns on the mathematical core of the learning process. Previous studies on the predictive auto-scaling techniques in the IaaS layer of cloud computing [2, 5, 6] are limited to the experimental evaluation. To the best of our knowledge, none of the research efforts in the predictive auto-scaling domain has investigated the theoretical foundations of the predictive auto-scaling techniques. Establishing a formal foundation is essential to obtain a solid and more generic understanding of various auto-scaling prediction algorithms. Thus, to support the proposed prediction suite, this paper performs a formal study of the theories that have been used in the predictive auto-scaling systems. Further, this paper investigates the components that theoretically affect the accuracy of the models. The theoretical investigation provides a formal analysis and explanation for the behaviors of the time-series prediction algorithms in the cloud environment with different workload patterns. In addition, this paper proposes four sub-hypotheses in section Theoretical investigation of the hypothesis.

According to the theoretical discussion, the risk minimization principle that is used by the time-series prediction algorithms affects the algorithms' accuracy in the environments with the different workload patterns (see Section Theoretical investigation of the hypothesis). Furthermore, to experimentally validate the formal discussion, this paper examines the influence of the workload patterns on the accuracy of three time-series prediction models: the Support Vector Machine (SVM) algorithm and two variations of the Artificial Neural Network (ANN) algorithm (i.e., Multi-Layer Perceptron (MLP) and Multi-Layer Perceptron with Weight Decay (MPLWD)). The SVM and the MLPWD algorithms use Structural Risk Minimization (SRM) principle, but the MLP algorithm uses Empirical Risk Minimization (ERM) principle to create the prediction model. Comparing the MLP with the MLPWD algorithm isolates the influence of the risk minimization principle on the prediction accuracy of the ANN algorithms. Therefore, comparing the MLP with the MLPWD shows the impact of the risk minimization principle on the prediction accuracy of the ANN algorithms. In addition, since the SVM and the MLPWD algorithms use the same risk minimization approach, comparing the SVM algorithm with the MLPWD algorithm isolates the influence of the regression model on the prediction accuracy.

This paper enhances the preciseness of our previous experimental results in [2] by isolating and studying the impact of the risk minimization principle on the prediction accuracy of the regression models in regards to the changing workload patterns. The main contributions of this paper are:

- Proposing an autonomic prediction suite which chooses the most suitable prediction algorithm based on the incoming workload pattern,
- Providing the theoretical foundation for estimating the accuracy of the time-series prediction algorithms in regards to the different workload patterns,
- Investigating the impact of the risk minimization principle on the accuracy of the regression models for different workload patterns, and
- Evaluating the impact of the input *window size* on the performance of the risk minimization principle.

TPC-W web application and Amazon Elastic Compute Cloud (Amazon EC2) are respectively used as the benchmark and the cloud infrastructure in our experiments. It should be noted that this paper is scoped to the influence of the workload patterns on the prediction results at the IaaS layer of the cloud computing. Other IaaS management aspects (such as the VM migration and the physical allocation of the VMs) are out of the scope of this paper.

The remainder of this paper is organized as follows: Background and related work section discusses the background and the related work. In Self-adaptive workload prediction suite section a high level design for the self-adaptive prediction suite is proposed. Theoretical investigation of the hypothesis section, describes the principles of the learning theory and mathematically investigates the hypothesis. Section Experimental investigation of the hypotheses presents the experimental results to support the theoretical discussion. The conclusion and the possible directions for the future research are discussed in Conclusions and future work section.

Background and related work

In this section, the background concepts that are used in the paper and the related work are introduced. Sub-section Workload is an overview of the workload concept and its patterns. Sub-sections Decision making and Prediction techniques provide an overview of the most dominant auto-scaling approaches in two broad categories: decision making and prediction techniques.

Workload

The term workload refers to the number of the end user requests, together with their arrival timestamp [4]. Workload is the consequence of the end users accessing the cloud service [7]. According to [4, 7, 8], there are five workload patterns in the cloud computing environments:

- *Static workload* is characterized by a constant number of requests per minute. This means that there is normally no explicit necessity to add or remove the processing power, the memory or the bandwidth for the workload changes (Fig. 1).
- *Growing workload* represents a load that rapidly increases (Fig. 2).
- *Periodic workload* represents regular periods (i.e., seasonal changes) or regular bursts of the load in a punctual date (Fig. 3).
- *On-and-off workload* represents the work to be processed periodically or occasionally, such as the batch processing (Fig. 4).
- *Unpredictable workloads* are generalization of the periodic workloads as they require elasticity but are not predictable. This class of workload represents the constantly fluctuating loads without regular seasonal changes (Fig. 5).

Resource allocation for the batch applications (i.e., on-and-off workload pattern) is usually referred to as scheduling which involves meeting a certain job execution deadline [4]. Scheduling is extensively studied in the grid environments [4] and also explored in the cloud environments, but it is outside of the scope of this paper. Similarly, the cloud services with a stable (or static) workload pattern do not require an auto-scaling system for resource allocation per se. Therefore, this paper considers cloud services with the *periodic*, *growing*, and *unpredictable* workload patterns.

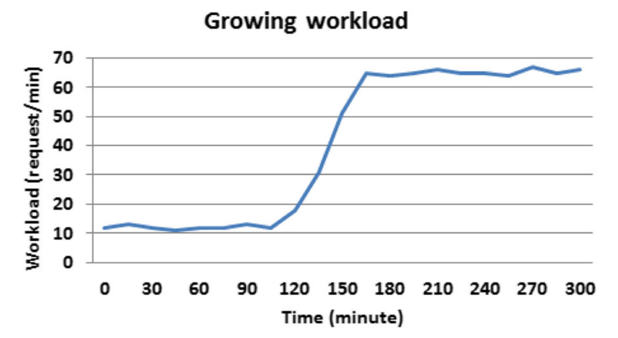

Fig. 2 Growing workload pattern

Decision making

The authors in [4] group the existing auto-scaling approaches into five categories: rule based technique, reinforcement learning, queuing theory, control theory, and time-series analysis. Among these categories, the time-series analysis focuses on the prediction side of the resource provisioning task and is not a "decision making" technique per se. In contrast, the rule-based technique is a pure decision making mechanism while the rest of the auto-scaling categories play the *predicator* and the *decision maker* roles at the same time.

The rule based technique is the only approach which is widely used in the commercial auto-scaling systems [9–11]. The popularity of this approach is due to its simplicity and intuitive nature. The rule based approaches typically have six parameters: an upper threshold (*thrU*), a lower threshold (*thrL*), *durU* and *durL* that define how long the condition must be met to trigger a scaling action, and *inL* and

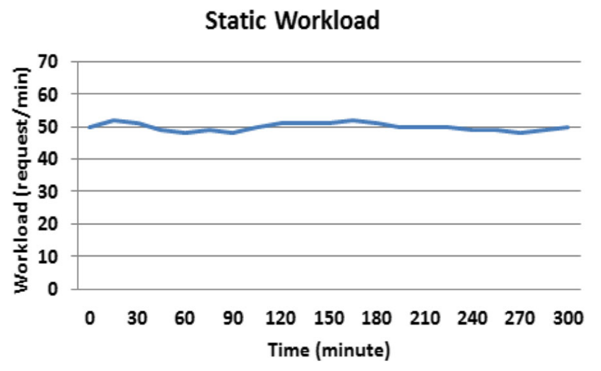

Fig. 1 Static workload pattern

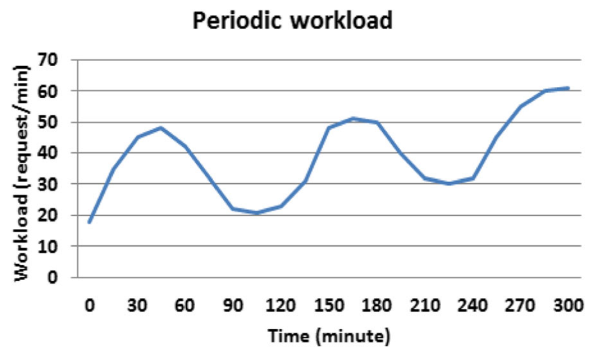

Fig. 3 Periodic workload pattern

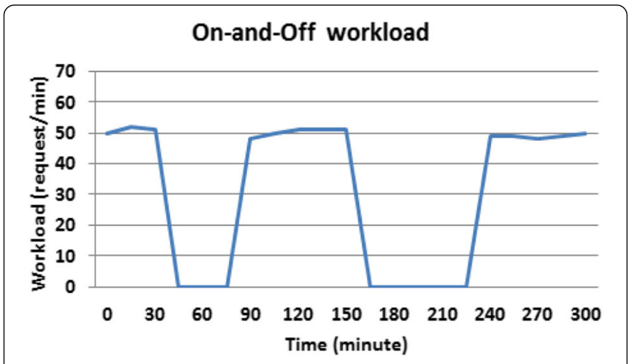

Fig. 4 On-and-off workload pattern

inU which indicate the cool down periods after the scale out and scale in actions [4]. The performance of the rule based technique highly dependents on these parameters. Therefore, finding the appropriate values for these parameters is a tricky task. A common problem in the rule based auto-scaling, which occurs due to an inappropriate threshold value, is the oscillations in the number of the leased VMs. In fact, the *durU* and the *durL* parameters are introduced to decrease the number of the scaling actions and reduce the VM oscillations. Some researchers have proposed alternative techniques to address the VM oscillation problem. For instance, the work in [12] uses a set of four thresholds and two durations. Moreover, some research works (such as [13]) have adopted a combination of the rules and a voting system to generate the scaling actions.

Prediction techniques

The most dominant prediction technique in the cloud auto-scaling domain is the time-series analysis [4]. In order to use the time-series analysis for the cloud auto-scaling purposes, a performance indicator is periodically sampled at fixed intervals. The result is a time-series containing a sequence of the last observations of the performance indicator. The time-series prediction algorithms extrapolate this sequence to predict the future value. Some of the time-series prediction algorithms that are used in the existing cloud resource provisioning systems are Moving Average, Auto-regression, ARMA, exponential smoothing, and machine learning approaches [4].

Moving average generally generates poor results for the time-series analysis [4]. Therefore, it is usually applied only to remove the noise from the time-series. In contrast, auto-regression is largely used in the cloud auto-scaling field. The results in [13] show that the performance of the auto-regression algorithm depends on the monitoring interval length, the size of the history window, and the size of the adaptation window. ARMA is a combination of the moving average and the auto-regression algorithms. The authors in [14] use ARMA to predict the future workload. Machine learning algorithms are used in [3] and [6] to carry out the prediction task in the cloud resource provisioning problem. The authors in [6] verify the Artificial Neural Networks (ANN) and the Linear Regression (LR) algorithms to predict the future value of the CPU load. The results in [6] conclude the ANN prediction model surpasses the LR algorithm in terms of prediction accuracy in the auto-scaling domain. In addition, the authors in [3] compare the SVM, the ANN and the LR algorithms and show the SVM algorithm outperforms the ANN and the LR algorithms to predict the future CPU utilization, response time, and throughput of a cloud service. Furthermore, the authors in [15] propose a self-adaptive method that uses a decision tree to assign the incoming workload to one of the forecasting methods based on the workload characteristics. According to the results of [15] the overall prediction accuracy increases by using different prediction algorithms for different workloads. However, to the best of our knowledge, none of the research works in the predictive auto-scaling domain investigates the theoretical foundations of the correlation between the different workload patterns and the accuracy of the prediction algorithms. Therefore, this paper performs a formal study of the theories that are closely related to the regression models used in the predictive auto-scaling systems and investigates the workload characteristics that affect the accuracy of the regression models.

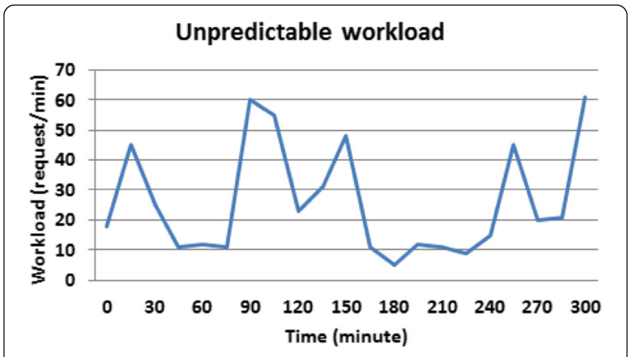

Fig. 5 Unpredictable workload pattern

Self-adaptive workload prediction suite

This section proposes a high level architectural design of the self-adaptive workload prediction suite. The self-adaptive suite uses the *decision fusion* technique to increase the prediction accuracy of the cloud auto-scaling systems. Decision fusion is defined as the process of fusing information from individual data sources after each data source has undergone a preliminary classification [16]. The self-adaptive prediction suite aggregates the prediction results of multiple time-series prediction algorithms to improve the final prediction accuracy. The different time-series prediction techniques use different *risk* *minimization* principles to create the prediction model. The theoretical analysis shows that the accuracy of a risk minimization principle depends on the complexity of the time-series. In addition, since the complexity of a time-series is defined by its corresponding workload pattern, the theoretical analysis concludes that the accuracy of a regression model is a function of the workload pattern (see Theoretical investigation of the hypothesis section).

Furthermore, Experimental investigation of the hypotheses section experimentally confirms the theoretical conclusion of Theoretical investigation of the hypothesis section. In the experiment two versions of an ANN algorithm (i.e., multi-layer perceptron (MLP) and multi-layer perceptron with weight decay (MLPWD)) and the Support Vector Machine (SVM) algorithm are used to predict three groups of time-series. Each time-series group represents a different workload pattern. The objective of the experiment is to investigate the correlation between the accuracy of the risk minimization principle and the workload pattern.

The ANN algorithms are identical except that MLPWD uses the *structural risk minimization* principle and MLP uses the *empirical risk minimization* principle to create the prediction model. Moreover, the SVM algorithm uses the *structural risk minimization* principle to create the prediction model. The experimental results show (see Experimental investigation of the hypotheses section):

- To predict the future workload in an environment with the unpredictable workload pattern it is better to use MLP algorithm with a large sliding window size.
- To predict the future workload in an environment with the periodic workload pattern it is better to use MLPWD algorithm with a small sliding window size.
- To predict the future workload in an environment with the growing workload pattern it is better to use SVM algorithm with a small sliding window size.

The self-adaptive prediction suite uses the experimental results as the fusion rule to aggregate the SVM, the MLP, and the MLPWD prediction algorithms in order to improve the prediction accuracy of the cloud auto-scaling systems. The prediction suite senses the pattern of the incoming workload and automatically chooses the most accurate regression model to carry out the workload prediction. Each workload is represented by a time-series. To identify the workload pattern, the proposed self-adaptive suite decomposes the incoming workload to its components by using Loess package of the R software suite [17]. The Loess component decomposes a workload to its *seasonal*, *trend*, and *remainder* components. If the workload has strong seasonal and trend components which repeat at fixed intervals, then the workload has periodic pattern. If the trend of the component is constantly increasing or decreasing, then the workload has growing pattern. Otherwise the workload has unpredictable pattern.

The self-adaptive suite constantly monitors the characteristics of the incoming workload (i.e., seasonal and trend components) and replaces the prediction algorithm according to a change in the incoming workload pattern. To this end, the autonomic system principles are used to design the self-adaptive workload prediction suite.

The goal of an autonomic system (Fig. 6 is to make a computing system self-managed. The field is motivated by the increasing complexity in the software systems due to objects change, environmental influence, and ownership cost of software [18, 19]. The idea is that a self-managed system (i.e., an autonomic system) must be attentive to its internal operation and adapt to the *behavior* change in order to produce *future* actions.

A typical autonomic system consists of a context, an autonomic element, and a computing environment [20–22]. In addition, the autonomic system receives the goals and gives the feedback to an external environment. An autonomic element regularly senses the sources of change by using the *sensors*. In the prediction suite, the sensor is the change in the workload pattern (Fig. 7).

In this paper, the autonomic system architecture is adopted for the cloud auto-scaling system architecture. The mapping between the two is presented in Fig. 8.

The presented cloud auto-scaling architecture consists of the cloud workload context, the cloud auto scaling autonomic system, and the cloud computing scaling decisions. The cloud workload context consists of two meta-autonomic elements: workload pattern and cloud auto scaling. In addition, a component for autonomic manager, knowledge, and goals is added to the architecture.

The cloud workload usage represents the "real world usage context" while the scaling decisions represents the "computing environment" context. It is important to note that an autonomic system always operates and executes within a context. The context is defined by the environment and the runtime behavior of the system. The purpose of the autonomic manager is to apply the domain specific knowledge to the cloud workload patterns and the appropriate predictor algorithm (Fig. 9) in order to facilitate the prediction. The autonomic manager is constructed around the analyze/decide/act control loop. Figure illustrates a detailed presentation of the cloud auto-scaling autonomic element.

The cloud auto-scaling autonomic elements (workload patterns and predictor) are designed such that the architecture can be implemented using the strategy design pattern [23] (Fig. 10). The strategy design pattern consists of a *strategy* and a *context*. In the self-adaptive prediction suite the prediction model is the strategy and the workload pattern is the context. A context passes all data (i.e.,

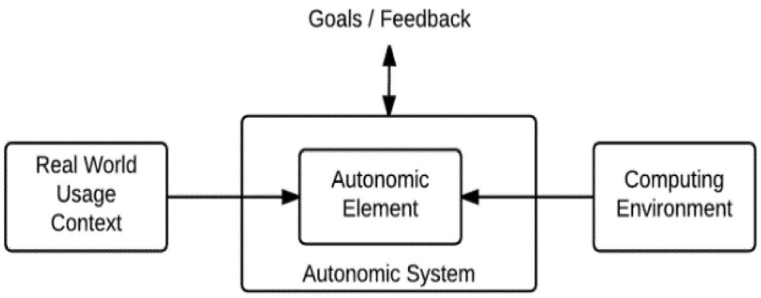

Fig. 6 Autonomic system

the workload pattern) to the strategy. In the prediction suite, the context passes itself as an argument to the strategy and lets the strategy call the context as required. The way this works is that the context determines the workload pattern and passes its pattern interface to the strategy's interface. The strategy then uses the interface to invoke the appropriate algorithm based on the workload pattern interface. All of these functions are realized at runtime automatically.

A careful examination of the strategy design pattern (Fig. 10) shows that the context is in turn designed by using the template design pattern. The intent of the template design pattern is to define the skeleton of an algorithm (or a function) in an operation that defers some steps to subclasses [23].

In a generic strategy design pattern, the context is simply an abstract class with no concrete subclasses. We have modified this by using the template pattern to introduce the concrete subclasses to represent the different workload patterns and to implement the workload pattern context as an autonomic element. This way, the cloud workload pattern is determined automatically and the pattern interface is passed on to the predictor autonomic element which then invokes the appropriate prediction algorithm for the workload pattern. After which the training is carried out and the testing (i.e., the prediction) using the appropriate algorithm is done.

Theoretical investigation of the hypothesis

Machine learning can be classified into the supervised learning, semi-supervised learning, and unsupervised learning. The supervised learning deduces a functional relationship from the training data that generalizes well to the whole dataset. In contrast, the unsupervised learning has no training dataset and the goal is to discover the relationships between the samples or reveal the latent variables behind the observations [5]. The semi-supervised learning falls between the supervised and the unsupervised learning by utilizing both of the labeled and the unlabeled data during the training phase [24]. Among the three categories of the machine learning, the supervised learning is the best fit to solve the prediction problem in the auto-scaling area [5]. Therefore this paper investigates the theoretical foundation of the supervised learning.

To accept or reject the hypothesis, we start with the formal definition of the machine learning and then explore the risk minimization principle as the core function of the learning theory. The definitions in the following sub-sections are taken from [25].

Formal definition of the machine learning process

Vapnik describes the machine learning process through three components [25]:

1. A *generator* of random vectors x. The generator uses a fixed but unknown distribution $P(x)$ to independently produce the random vectors.
2. A *supervisor* which is a function that returns an output vector y for every input vector x, according to a conditional distribution function $P(y|x)$. The conditional distribution function is fixed but unknown.

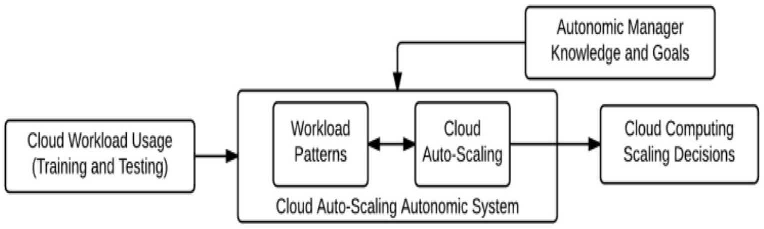
Autonomic Manager Knowledge and Goals
Cloud Workload Usage (Training and Testing)
Workload Patterns
Cloud Auto-Scaling
Cloud Auto-Scaling Autonomic System
Cloud Computing Scaling Decisions

Fig. 7 Cloud auto-scaling autonomic system

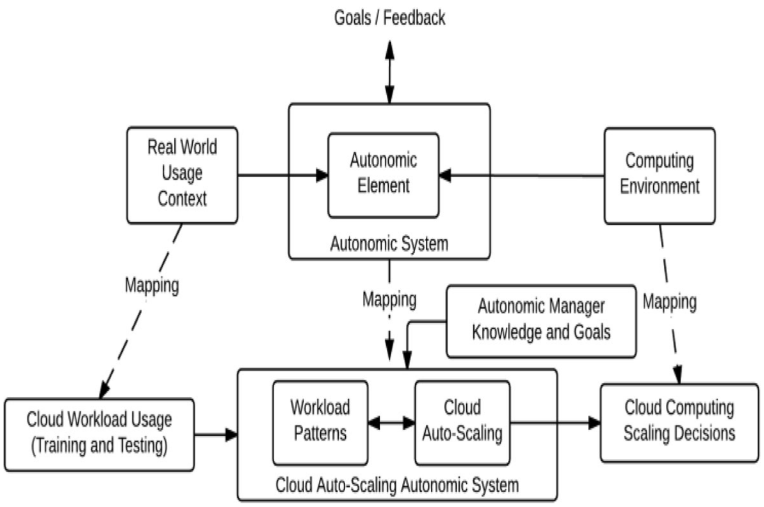

Fig. 8 Mapping classical autonomic system to cloud auto-scaling autonomic system

3. A learning machine that is capable of implementing a set of functions $f(x, w), w \in W$, where x is a random input vector, w is a parameter of the function, and W is a set of abstract parameters that are used to index the set of functions $f(x, w)$ [25].

The problem of learning is choosing from a given set of the functions, the one which best approximates the supervisor's response. The selection is based on a training set of l independent observations:

$$(x_1, y_1),\ \ldots,\ (x_l,\ y_l) \tag{1}$$

The machine learning technique objective is to find the best available approximation to the supervisor's response. To this end the loss $L(y, f(x, w))$ between the supervisor response y with respect to a given input x and the response $f(x, w)$ provided by the learning machine should be measured. The expected value of the loss, given by the *functional risk* is [25]:

$$R(w) = \int L(y, f(x, w))dP(x, y) \tag{2}$$

To improve the accuracy, the functional risk $R(w)$ should be minimized over a class of functions $f(x, w), w \in W$. The problem in minimizing the functional risk is that the joint probability distribution $P(x, y) = P(y|x)P(x)$ is unknown and the only available information is contained in the training set.

In the predictive auto-scaling problem domain, the *Predictor* component corresponds to the learning machine of the learning process. The goal is to find the most accurate predictor, which is the learning machine with the minimum functional risk. Components of the formal learning process can be mapped

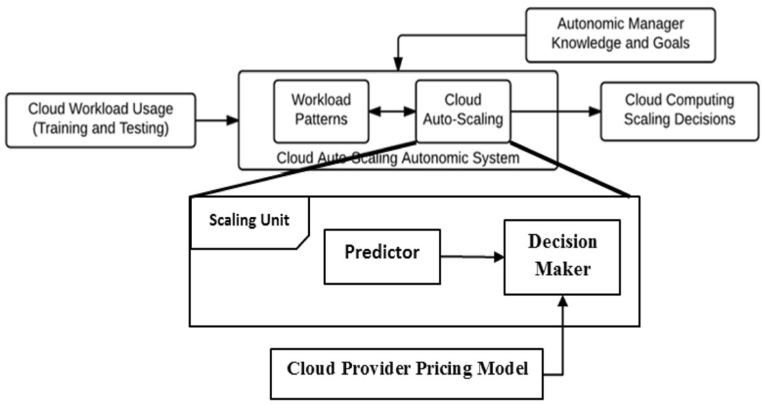
Autonomic Manager Knowledge and Goals
Cloud Workload Usage (Training and Testing)
Workload Patterns
Cloud Auto-Scaling
Cloud Computing Scaling Decisions
Cloud Auto-Scaling Autonomic System
Scaling Unit
Predictor
Decision Maker
Cloud Provider Pricing Model

Fig. 9 Projection of the cloud auto-scaling autonomic element

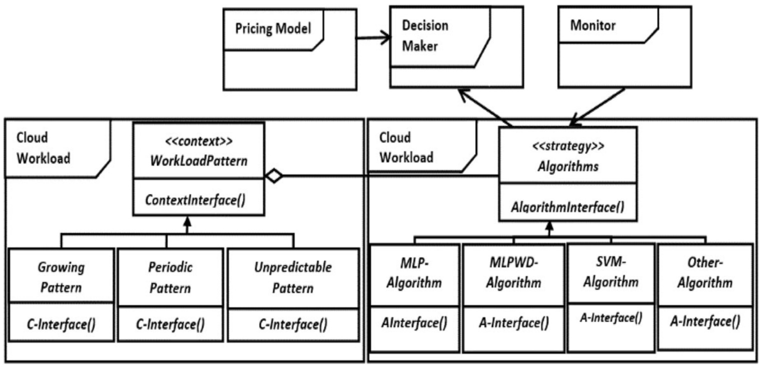

Fig. 10 Design of autonomic elements (Cloud workload pattern and predictor) using strategy design patter

to those of the predictive auto-scaling problem as follows:

- Supervisor's response is analogous to the time-series of workload values which is determined by $P(x, y)$.
- Independent observations are equivalent of the training dataset and indicate the historical values of the workload.
- Learning machine maps to the *Predictor* component.

In the auto-scaling problem domain, $P(x, y)$ refers to the workload distribution. Suppose that we have a set of candidate predictor functions $f(x, w),\ w \in W$ and we want to find the most accurate function among them. Given that only the workload values for the training duration are known, the functional risk $R(w)$ cannot be calculated for the candidate predictor functions $f(x, w),\ w \in W$; hence, the most accurate prediction function cannot be found.

Empirical risk minimization

To solve the functional risk problem, the functional risk $R(w)$ can be replaced by the empirical risk [25]:

$$E(w) = \frac{1}{l}\sum_{i=1}^{l} L(y_i, f(x_i,\ w)) \tag{3}$$

The empirical risk minimization (ERM) assumes that the function $f(x_i,\ w_l^*)$, which minimizes $E(w)$ over the set $w \in W$, results in a functional risk $R(w_l^*)$ which is close to minimum.

According to the theory of the uniform convergence of empirical risk to actual risk [26], the convergence rate bounds are based on the *capacity* of the set of functions that are implemented by the learning machine. The capacity of the learning machine is referred to as VC-dimension (for Vapnik-Chervonenkis dimension) [27] that represents the complexity of the learning machine.

Applying the theory of uniform convergence to the auto-scaling problem domain concludes that the convergence rate bounds in the auto-scaling domain are based on the complexity (i.e., VC-dimension) of the regression model that is used in the *Predictor* component.

According to the theory of the uniform convergence, for a set of indicator functions with VC-dimension h, the following inequality holds [25]:

$$R(w) < E(w) + C_0\left(\frac{l}{h}, \eta\right) \tag{4}$$

With confidence interval [25]:

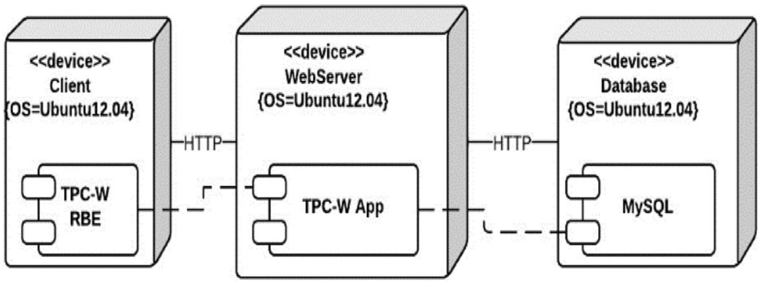
<<device>> Client {OS=Ubuntu12.04}
TPC-W RBE
HTTP
<<device>> WebServer {OS=Ubuntu12.04}
TPC-W App
HTTP
<<device>> Database {OS=Ubuntu12.04}
MySQL

Fig. 11 Experimental setup

$$C_0\left(\frac{l}{h},\eta\right)=\sqrt{\frac{h\left(ln\frac{2l}{h}+1\right)-\ln\eta}{l}} \tag{5}$$

where l is the size of the training dataset, h is the VC-dimension of the regression model, e is the Euler's number, and $(1-\eta)$ is the probability of the validity of Eq. (4) for all $w\in W$.

Equation (4) determines the bound of the regression model's error. Based on this equation, the probability of error of the regression model is less than the frequency of error in the training set plus the confidential interval. According to Eq. (4) the ERM principle is good to be used when the confidence interval is small (i.e., the functional risk is bounded by the empirical risk).

Structural risk minimization

Equations (4) and (5) show the bound of the regression model's error and the confidence interval. In Eqs. (4) and (5), l is the size of training dataset and h is the VC-dimension or the complexity of the regression model. According to Eq. (5) when $\frac{l}{h}$ is large, the confidence interval becomes small and can be neglected. In this case, the functional risk is bounded by the empirical risk, which means the probability of error on the testing dataset is bounded by the probability of error on the training dataset.

On the other hand, when $\frac{l}{h}$ is small, the confidence interval cannot be neglected and even $E(w)=0$ does not guarantee a small probability of error. In this case to minimize the functional risk $R(w)$, both $E(w)$ and $C_0\left(\frac{l}{h},\eta\right)$ (i.e., the empirical risk and the confidence interval) should be minimized simultaneously. To this end, it is necessary to control the VC-dimension (i.e., complexity) of the regression model. In other words, when the training dataset is complex, the learning machine increases the VC-dimension to shatter[1] the training dataset. By increasing the VC-dimension, the regression model becomes strongly tailored to the particularities of the training dataset and does not perform well to new data (the overfitting situation).

To control the VC-dimension, structural risk minimization principle (SRM) is used. SRM uses a nested structure of subsets $S_p=\{f(x,w),\ w\in W_p\}$ such that:

$$S_1\subset S_2\subset\ldots\subset S_n \tag{6}$$

The corresponding VC-dimensions of the subsets satisfy:

$$h_1<h_2<\ldots<h_n \tag{7}$$

Therefor the structural risk minimization (SRM) principle describes a general model of the capacity (or complexity) control and provides a trade-off between the hypothesis space complexity (i.e., the VC-dimension) and the quality of fitting the training data.

Workload pattern effects on prediction accuracy of empirical and structural risk minimizations

According to Workload section, there are three workload patterns in the cloud computing environment: periodic, growing, and unpredictable. The periodic and the growing workload patterns follow a repeatable pattern and their trend and seasonality is predictable. Contrariwise, the unpredictable workload pattern does not follow a repeatable trend. Thus, the unpredictable workload pattern is more complex than the growing and the periodic patterns, which suggests using a regression model with a higher VC-dimension to forecast the unpredictable pattern. From the discussions in sections Formal definition of the machine learning process to Summary and Workload, we propose the following sub-hypotheses in addition to our main hypothesis in the introduction:

- Hypothesis 1a: The structural risk minimization principle performs better in the environments with the periodic and growing (i.e., predictable) workload patterns.
- Hypothesis 1b: The empirical risk minimization principle performs better in the environments with the unpredictable workload pattern.
- Hypothesis 1c: Increasing the window sizes does not have a positive effect on the performance of the structural risk minimization principle in the cloud computing environments.
- Hypothesis 1d: Increasing the window size improves the performance of the empirical risk minimization principle in the unpredictable environments and has no positive effect on the performance of the empirical risk minimization principle in the periodic and the growing environments.

Making these sub-hypotheses provides a basis for proving the main hypothesis of this research. To systematically prove the sub-hypotheses, this section provides a theoretical reasoning to explain the empirical and the structural risk minimization principles behaviors in regards to the different workload patterns in the cloud computing environment.

As shown in Empirical risk minimization section, $\frac{l}{h}$ determines whether to use the empirical or the structural risk minimizations. In this paper we assume the training dataset size (i.e., , l) is static, therefore for the small values of h, $\frac{l}{h}$ fraction is large. In this case, the confidence interval is small and the functional risk is bounded by the empirical risk.

In environments with the predictable workload patterns (i.e., periodic or growing) the training and the testing datasets are not complex. Thus, in such environments h is small and the empirical and the structural risk minimizations perform well. However, it is possible that the empirical risk minimization becomes over fitted against the training dataset. The reason is that, although the periodic and the growing workloads follow a repeatable pattern, it is highly probable that some of the data points in the training dataset do not follow the main pattern of the time-series (i.e., noise data). The noise in the data increases the complexity of the regression model. Increasing the complexity (i.e., VC-dimension) increases the confidence interval as well as the probability of error (see Eq. (5)), which reduces the ERM accuracy. On the other hand, the SRM principle controls the complexity by neglecting the noise in the data, which reduces the confidence interval. Therefore, in the environments with the periodic and the growing workload patterns the SRM approach is expected to outperform the ERM approach (hypothesis 1a).

The same reasoning applies to the environments with the unpredictable workload pattern. In the unpredictable environments there is no distinctive workload trend and none of the data points should be treated as the noise. In the unpredictable environments, the ERM approach increases the VC-dimension to shatter all of the training data points. However, since the training and the testing datasets follow the same unpredictable pattern, increasing the VC-dimension helps the prediction model to predict the fluctuations of the testing dataset, as well. On the contrary, the SRM approach controls the VC-dimension to decrease the confidence interval. Therefore, the SRM approach cannot capture the fluctuating nature of the unpredictable workload pattern and trains a less accurate regression model compared to the ERM approach (hypothesis 1b).

In the machine learning domain, window size refers to the input size of the prediction algorithm. Increasing the window size provides more information for the prediction algorithm and is expected to increase the accuracy of the prediction model. However, increasing the input size makes the prediction model more complex. To manage the complexity, the SRM approach compromises between the accuracy and the VC-dimension. Therefore, increasing the window size does not necessarily affect the accuracy of the SRM prediction model. (Hypothesis 1c).

Furthermore, because the ERM approach cannot control the complexity of the regression model, increasing the window size increases the VC-dimension of the prediction model. In the predictable environments (i.e., the periodic and the growing patterns) the training and the testing datasets are not complex and the ERM principle is able to capture the time-series behaviors by using smaller window sizes. However, increasing the window size in the predictable environments increases the noise in the training dataset which causes a bigger confidence interval, and reduces the accuracy of the prediction model. On the other hand, due to the fluctuations in the unpredictable datasets, none of the data points in the training dataset should be considered as a noise. Therefore, in the unpredictable environments increasing the window size helps the ERM principle to shatter more training data. However, since the training and the testing datasets follow the same unpredictable pattern, increasing the window size improves the ERM precision to predict the fluctuations of the testing dataset, as well (hypothesis 1d).

Experimental investigation of the hypotheses section experimentally investigates the theoretical discussion of this section and evaluates the four sub-hypotheses.

Summary

The research in the learning theory provides a rich set of knowledge in learning the complex relationships and patterns in the datasets. Vapnik et al. show that the proportion of the training dataset size to the complexity of the regression model determines whether to use the empirical or the structural risk minimizations [25]. In the auto-scaling domain, the *Predictor* component corresponds to the learning machine of the leaning process. Therefore, to improve the accuracy of the *Predictor* component, the risk minimization principle should be determined based on the complexity of the prediction techniques (i.e., the VC-dimension) and the training dataset size. The workload pattern complexity is the main driving factor of the *Predictor* component's VC-dimension. Four sub-hypotheses are introduced in order to experiment the risk minimization principles vis-à-vis the different workload patterns.

Experimental investigation of the hypotheses

The main goal of the experiment presented in this section is to verify the empirical and the structural risk minimization principles behaviors in the environments with the periodic, growing, and unpredictable workload patterns. There are various learning algorithms that have been used as the predictor for the auto-scaling purposes (see Prediction techniques section) which use either the empirical or the structural risk minimizations. In our previous work (see [2]) the SVM algorithm which is based on the structural minimization and the ANN algorithm which uses the empirical minimization principle were used. Our experimental results in [2] showed that in the environments with the periodic and the growing workload patterns the SVM algorithm outperforms the ANN algorithm, but ANN has a better accuracy in forecasting the unpredictable workloads. These results support the theoretical discussion in Evaluation metrics section. However, in this paper the goal is to zero-in on two different implementations of the ANN algorithm in order to compare the effect of the structural and the empirical risk minimizations on the ANN prediction accuracy. Therefore, in this experiment two implementations of the ANN algorithm (i.e., MLP and MLPWD) are used to isolate the influence of the risk minimization principle on the prediction accuracy. MLP uses the ERM principle and MLPWD uses the SRM principle. In addition, since both of the MLPWD and the SVM algorithms use the SRM principle, the accuracy of the MLPWD is compared with the SVM accuracy to isolate the impact of the regression model structure on the accuracy of the machine learning algorithms.

Sections Multi-layer perceptron with empirical risk minimization, Multi-layer perceptron with structural risk minimization, and Support vector machines briefly explain MLP, MLPWD, and SVM algorithms, respectively. Sections Training and testing of MLP and MLPWD, Evaluation metrics, and Experimental results describe the experiment and the results.

Multi-layer perceptron with empirical risk minimization

There are different variations of the Artificial Neural Network (ANN), such as back-propagation, feed-forward, time delay, and error correction [5]. MLP is a feed-forward ANN that maps the input data to the appropriate output.

A MLP is a network of simple neurons that are called perceptron. Perceptron computes a single output from the multiple real valued inputs by forming a linear combination to its input weights and putting the output through a nonlinear activation function. The mathematical representation of the MLP output is [25]:

$$y = \varphi\left(\sum\nolimits_{i=1}^{n} w_i x_i + b\right) = \varphi\left(W^T X + b\right) \tag{8}$$

where W denotes the vector of weights, X is the vector of inputs, b is the bias, and φ is the activation function.

The MLP networks are typically used in the supervised learning problems. Therefore, there is a training set that contains an input–output set similar to Eq. (1). The training of the MLP refers to adapting all the weights and biases to their optimal values to minimize the following equation [25]:

$$E = \frac{1}{l}\sum\nolimits_{i=1}^{l} \left(T_i - Y_i\right)^2 \tag{9}$$

where T_i denotes the predicted value, Y_i is the actual value, and l is the training set size. Equation (9) is a simplified version of Eq. (3) and represents the empirical risk minimization.

Multi-layer perceptron with structural risk minimization

The general principle of the structural risk minimization can be implemented in many different ways. According to [28] there are four steps to implement the structural risk minimization (see section Structural risk minimization), of which the first step is to choose a class of functions with hierarchy of nested subsets in ordered of the complexity. Authors of [25] suggest three examples of the structures that can be used to build the hierarchy of the neural networks.

- Structure given by the architecture of the neural network.
- Structure given by the learning procedure
- Structure given by the preprocessing.

The second proposed structure (i.e., given by the learning procedure) uses “weight decay” to create a hierarchy of the nested functions. This structure considers a set of the functions $S = \{f(x, w),\ w \in W\}$ that are implemented by a neural network with a fixed architecture. The parameters {w} are the weights of the neural network. Nested structure is introduced through $S_p = \{f(x, w),\ ||w|| \le C_p\}$ and $C_1 < C_2 < \ldots < C_n$, where C_i is a constant value that defines the ceiling of the norm of the neural network weights. For a convex loss function, the minimization of the empirical risk within the element S_p of the structure is achieved through the minimization of [29]:

$$E\left(w, \gamma_p\right) = \frac{1}{l}\sum\nolimits_{1}^{l} L(y_i, f(x_i,\ w)) + \gamma_p ||w||^2 \tag{10}$$

The nested structure can be created by appropriately choosing Lagrange multipliers $\gamma_1 > \gamma_2 > \ldots > \gamma_n$. According to Eq. (10), the well-known weight-decay procedure refers to the structural minimization [25].

Training the neural networks with the weight decay means that during the training phase, each updated weight is multiplied by a factor slightly less than 1 to prevent the weight from growing too large. The risk minimization equation for the Multi-Layer Perceptron with Weight Decay (MLPWD) algorithm is [29]:

$$E = \frac{1}{l}\sum\nolimits_{i=1}^{l} \left(T_i - Y_i\right)^2 + \frac{\lambda}{2}\sum\nolimits_{i=1}^{l} {w_i}^2 \tag{11}$$

Authors of [29] have shown that the conventional weight decay technique can be considered as the simplified version of the structural risk minimization in the neural networks. Therefore, in this paper we use MLPWD algorithm to study the accuracy of the structural risk

Table 1 Hardware specification of servers for experiment

	Memory	Processor	Storage
Client	1 GB	4 core	8 GB
Web server	1 GB	4 core	8 GB
Database	2 GB	8 core	20 GB

Table 3 SVM configuration

Parameter Name	Value
C (complexity parameter)	1.0
kernel	RBF Kernel
regOptimizer	RegSMOImproved

minimization for predicting the different classes of workload.

Support vector machines

Support Vector Machine (SVM) is used for many machine learning tasks such as pattern recognition, object classification, and regression analysis in the case of the time series prediction. Support Vector Regression (SVR), is the methodology by which a function is estimated by using the observed data. In this paper the SVR and the SVM terms are used interchangeably.

SVM uses Eqs. (12) and (13) to define the prediction functions for the linear and the non-linear regression models, respectively [6]:

$$f(x) = (w.x) + b \tag{12}$$

$$f(x) = (w.\varphi(x)) + b \tag{13}$$

where, w is a set of weights, b is a threshold, and φ is a kernel function.

If the time-series is not linear, the regression model maps the time-series x to a higher dimension feature space by using kernel function $\varphi(x)$. Then the prediction model performs the linear regression in the higher dimensional feature space. The goal of the SVM training is to find the optimal weights w and the optimal threshold b. There are two criteria to find the optimal weights and the optimal threshold. The first criterion is the *flatness* of the weights, which can be measured by the Euclidean norm (i.e., minimize $||w||^2$). The second criterion is the error generated by the estimation process of the value, also known as the empirical risk, which is to be minimized. The overall goal is to find a regression function $f(x, w)$ which minimizes the structural risk R_s [6]:

$$R_s = E + \frac{\lambda}{2}||w||^2 \tag{14}$$

Table 2 MLP and MLPWD configurations

Parameter Name	MLP Value	MLPWD
Learning Rate (ρ)	0.3	0.3
Momentum	0.2	0.2
Validation Threshold	20	20
Hidden Layers	1	1
Hidden Neurons	(attributes + classes)/2	(attributes + classes)/2
Decay	False	True

where, E is the empirical risk, and $||w||^2$ represents the flatness of the weights of the regression function. The scale factor λ is the regularization constant and is often referred to as the capacity control factor. The scale factor λ is useful for reducing the complexity of the regression model to prevent the overfitting problem.

Experimental setup

In this experiment workload represents the web service requests arrival rate. Workload is a key performance indicator of a given web service that can be used to calculate other performance indicators (such as utilization, and throughput) of that web service. Furthermore, monitoring workload of a web service is straightforward and can be carried out by using instrumentation technique. Therefore, in this experiment workload of the web service is the target class of the prediction techniques.

The goal of this experiment is to compare the accuracy of the MLP, the MLPWD, and the SVM algorithms for predicting the periodic, the growing, and the unpredictable workload patterns. The required components to conduct this experiment are: a benchmark to generate the workload patterns, an infrastructure to deploy the benchmark, and an implementation of the prediction algorithms. Java implementation of TPC-W [30] and Amazon EC2 are used as the benchmark and the infrastructure, respectively. In addition, the implementation of Multi-Layer Perceptron and Support Vector Machine algorithms in WEKA tool is used to carry out the prediction task.

The MLP algorithm in WEKA tool [31] has various configuration parameters including a parameter to switch on/off the weight decay feature (i.e., *decay* parameter). Therefore, to use the empirical risk minimization the default value of the *decay* parameter (i.e., off) is used. Also, to use the structural risk minimization, the *decay* parameter is switched on.

The TPC-W benchmark emulates an online book shop and is implemented on 3-tier architecture. As shown in Fig. 11, the experimental setup consists of three virtual machines running on Ubuntu Linux. Table 1 shows the details of the virtual machines. Note that to decrease the experiment complexity, the experiment is limited to monitoring the performance of the web server tier in and it is assumed that the database is not a bottleneck. For this reason, a relatively powerful virtual machine is dedicated to the database tier.

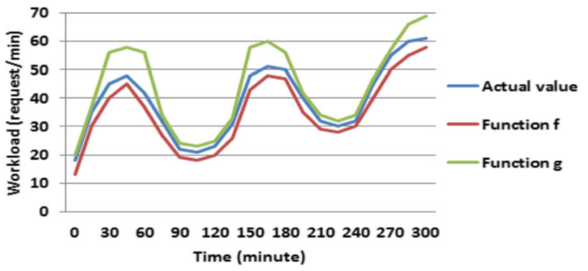

Fig. 12 RMSE vs. MAE

On the client side, a customized script is used along with the TPC-W workload generator to produce the growing, the periodic, and the unpredictable workload patterns. In this experiment workload represents the webpage requests arrival rate. Each of the workload patterns is generated for 500 min. To improve accuracy of the results, the experiment is repeated 10 times for each workload pattern. On the web-server machine, the total number of the user requests is stored in the log files every minute. This results in 10 workload trace files, for each of the workload patterns. Each of the workload trace files has 500 data points. We refer to the workload trace files as the *actual workloads* in the rest of this paper.

Training and testing of MLP and MLPWD

In our previous work [1] we proved that in the auto-scaling domain the optimum training duration for the ANN and the SVM algorithms is 60% of the experiment duration. Therefore, in this experiment the first 300 data points (i.e., 60%) of the actual workload trace files are considered as the training datasets and the rest 200 data points are dedicated to the test.

Another important factor in the training and the testing of the time-series prediction algorithms is the *dimensionality* of the datasets (i.e., the number of the features that exist in the dataset). In this experiment, the actual datasets have only one feature, which is the number of the requests that arrive at the cloud service per minute. Therefore, in order to use the machine learning prediction algorithms *sliding window* technique is used. The sliding window technique uses the last k samples of a given feature to predict the future value of that feature. For example, to predict value of b_{k+1} the sliding window technique uses $[b_1, b_2, \ldots, b_k]$ values. Similarly, to predict b_{k+2}, the sliding window technique updates the historical window by adding the actual value of b_{k+1} and removing the oldest value from the window (i.e., the sliding window becomes $[b_2, b_3, \ldots, b_{k+1}]$). Setting the sliding window size is not a trivial task. Usually the smaller window sizes do not reflect the correlation between the data samples thoroughly, while using the bigger window size increases the chance of the overfitting. Thus, in this experiment the effect of the sliding window size on the prediction accuracy of MLP and MLPWD is studied, as well.

To reduce the probability of the overfitting problem, the cross-validation technique is used in the training phase. Readers are encouraged to see [32] for more details about the cross-validation technique. Table 2 shows the configuration of the MLP and the MLPWD algorithms in this experiment. Configuration of the SVM algorithm is shown in Table 3.

Evaluation metrics

Accuracy of the experimental results can be evaluated based on the different metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), PRED (25) and R2 Prediction Accuracy [33]. Among these metrics, PRED(25) only considers the percentage of the observations whose prediction accuracy falls within 25% of the actual value. In addition, R2 Prediction Accuracy

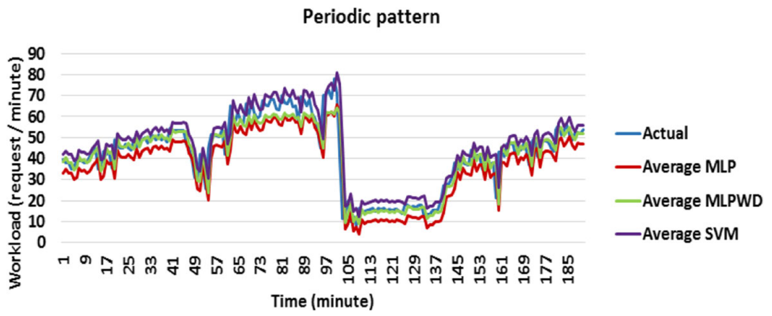

Fig. 13 Sample prediction results (periodic pattern)

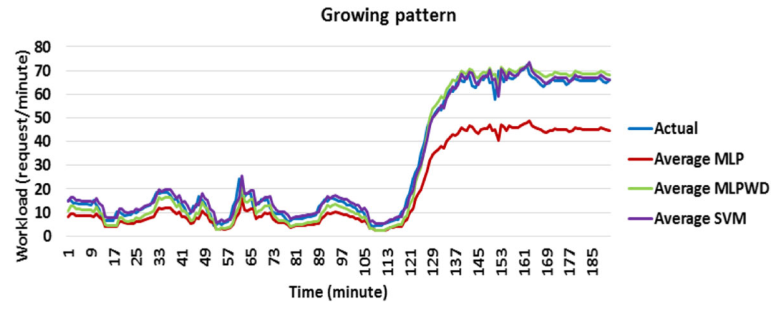

Fig. 14 Sample prediction results (growing pattern)

is a measure of the goodness-of-fit, which its value falls within the range [0, 1] and is commonly applied to the linear regression models [6]. Due to the limitations of PRED (25) and R2 Prediction Accuracy, the MAE and the RMSE metrics are used in this paper. The formal definitions of these metrics are [33]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |YP_i - Y_i| \qquad (15)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (YP_i - Y_i)2}{n}} \qquad (16)$$

where YP_i is the predicted output and Y_i is the actual output for ith observation, and n is the number of the observations for which the prediction is made. The MAE metric is a popular metric in statistics, especially in the prediction accuracy evaluation. The RMSE represents the sample standard deviation of the differences between the predicted values and the observed values. A smaller MAE and RMSE value indicates a more effective prediction scheme.

The MAE metric is a linear score which assumes all of the individual errors are weighted equally. Moreover, the RMSE is most useful when the large errors are particularly undesirable [34].

In the auto-scaling domain, a regression model that generates a greater number of small errors (function *f* in Fig. 12) is more desirable than a regression model that generates a fewer number of the large errors (function *g*

Table 4 MAE and RMSE values (periodic pattern)

Phase	Window size	Average MAE			Average RMSE		
		MLP	MLPWD	SVM	MLP	MLPWD	SVM
Training	2	6.88	4.16	4.65	8.55	6.65	7.31
	3	6.7	4.12	4.62	8.32	6.32	7
	4	6.5	4.11	4.62	8.12	6.12	6.99
	5	5.95	4.05	4.52	8	6.44	6.8
	6	5.78	4.02	4.52	7.56	6.12	6.7
	7	5.68	3.88	4.32	7.5	6.2	6.7
	8	5.68	3.95	4.3	7.12	6.21	6.6
	9	5.51	4.02	4.3	6.9	6.18	6.8
	10	4.98	4	4.31	6.52	6.18	6.7
Testing	2	6.2	6	6	8.31	8	8.1
	3	6.3	5.9	6	8.31	7.9	7.98
	4	6.3	5.8	6.1	8.34	7.9	8.05
	5	6.99	5.9	6.2	8.62	7.8	8.15
	6	7.15	5.7	6.1	8.77	7.4	8
	7	7.25	5.72	6	9.12	7	7.71
	8	7.98	5.75	6	9.15	7	7.65
	9	8.56	5.66	5.8	10.36	7.1	7.5
	10	9.2	5.58	5.7	11.89	6.9	7.6

Table 5 MAE and RMSE values (growing pattern)

Phase	Window size	MAE			RMSE		
		MLP	MLPWD	SVM	MLP	MLPWD	SVM
Training	2	2.5	2.1	1.7	3.9	4.02	3.8
	3	2.8	2.3	1.75	3.9	3.82	3.7
	4	2.7	2.3	1.8	4.1	3.87	3.6
	5	2.7	2.5	1.8	3.98	3.89	3.7
	6	2.6	2.4	1.8	3.88	3.71	3.6
	7	2.7	2.4	1.81	3.84	3.81	3.6
	8	2.66	2.33	1.78	3.78	3.62	3.5
	9	2.8	2.22	1.78	3.95	3.7	3.3
	10	2.57	2.25	1.78	4	3.7	3.4
Testing	2	3.77	3	2.5	4.4	4	3.7
	3	3.85	3.6	2.5	4.91	4.21	3.7
	4	3.55	3.5	2.6	4.92	4.5	3.65
	5	3.64	3.41	2.4	4.71	4.22	3.6
	6	3.89	3.42	2.3	4.52	4.31	3.7
	7	3.84	3.31	2.2	5.11	4	3.7
	8	3.95	3.02	2.2	5.52	3.99	3.4
	9	4.12	3	2.2	5.98	3.95	3.5
	10	4.1	2.8	2.2	6.02	3.9	3.7

Table 6 MAE and RMSE values (unpredictable pattern)

Phase	Window size	MAE			RMSE		
		MLP	MLPWD	SVM	MLP	MLPWD	SVM
Training	2	1.4	1.74	1.81	2.6	2.9	3.15
	3	1.42	1.73	1.73	2.61	2.88	3.2
	4	1.43	1.72	1.78	2.59	2.87	3.31
	5	1.4	1.73	1.75	2.55	2.89	3.15
	6	1.35	1.69	1.73	2.4	2.91	3.19
	7	1.46	1.66	1.72	2.48	2.98	3.2
	8	1.44	1.65	1.74	2.31	2.74	3.2
	9	1.48	1.66	1.66	2.2	2.65	3.16
	10	1.44	1.65	1.68	2.15	2.74	3.17
Testing	2	2.6	2.82	3.12	3.01	3.41	3.31
	3	2.5	2.8	3.1	3	3.4	3.64
	4	2.34	2.77	2.9	3	3.38	3.7
	5	2.21	2.76	2.88	2.98	3.41	371
	6	1.98	2.44	2.89	2.9	3.42	3.78
	7	1.65	2.4	2.85	2.8	3.21	3.88
	8	1.42	2.1	2.91	2.7	3.2	3.9
	9	0.98	2.11	2.92	2.4	3.11	4.1
	10	0.98	2.1	2.9	2.2	2.8	4.18

in Fig. 12). The reason is because the rule-based decision makers issue the scale actions based on the prediction values and to generate a *correct* scale action, the prediction should be *close enough* to the actual value. In other words, the rule-based decision makers are not sensitive to the small errors in the prediction results. Therefore, the smaller errors in the prediction results are negligible. Our previous work [1] investigates the sensitivity of the rule-based decision makers to the prediction results. As a result, in the cloud auto-scaling domain, the RMSE factor is more important than the MAE factor. However, considering both metrics (i.e., MAE and RMSE) provides a comprehensive analysis of the accuracy of the prediction models. The greater is the difference between RMSE and MAE the greater is the variance in the individual errors in the sample.

Experimental results

The experiment has three iterations and each of the iterations evaluates the accuracy of the SVM, the MLP and the MLPWD algorithms for predicting one of the workload patterns. For each workload pattern, the prediction models are trained and tested based on 10 workload trace files and their accuracy is measured by MAE and RMSE metrics. The overall accuracy of each prediction model is represented by its average MAE and RMSE metric values. Figures 12 ,13 and 14 show the average MLPWD, MLP, and SVM prediction results in the test phase (window size = 3 min) for the periodic, growing, and unpredicted workload patterns, respectively.

Tables 4, 5 and 6 present the training and the testing accuracy of the MLWPD, MLP, and SVM for predicting the periodic, the growing and the unpredictable workloads, respectively. The results are also plotted in Figs. 16, 17, 18, 19, 20, 21. Note that the MAE and RMSE values in Tables 4, 5 and 6 are the average of the MAE and RMSE results over 10 repetitions of the experiment for each of the prediction algorithms. The following subsections analyze the experimental results in regard to the four sub-hypotheses that are introduced in Section Workload pattern effects on prediction accuracy of empirical and structural risk minimizations.

Hypothesis 1.a: the SRM principle performs better in the environments with the predictable workload patterns

In the environments with the predictable workloads, the training and the testing datasets are not complex. Therefore, both of the ERM and SRM principles are accurate. For instance, in the environments with the periodic workload pattern (Fig. 15), the MAE and the RMSE values of the MLP and the MLPWD algorithms for window size = 2 are very close (see Table 4). Because the SRM neglects the noise data its accuracy is slightly better than the ERM. However, by increasing the window size the noise in the training data

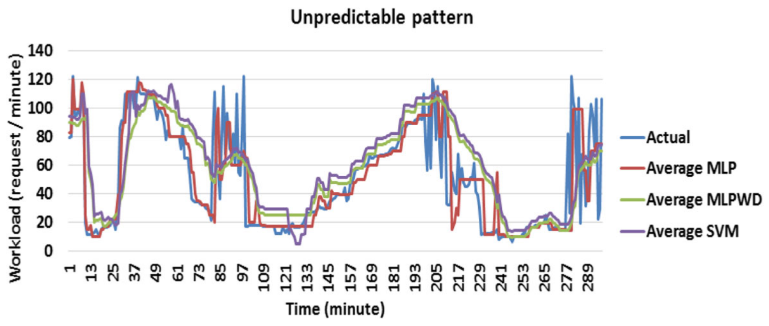

Fig. 15 Sample prediction results (unpredictable pattern)

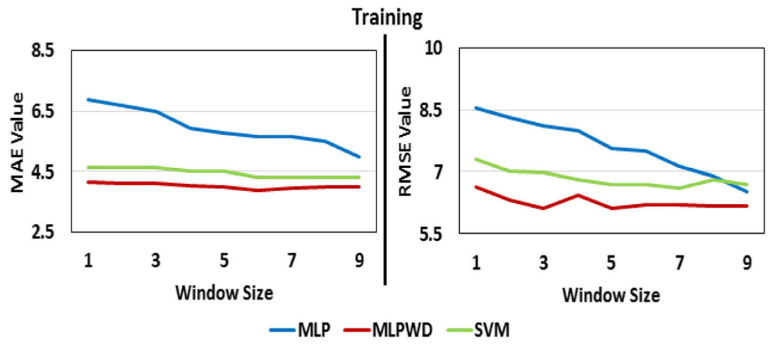

Fig. 16 MAE and RMSE values in the training phase (periodic pattern)

increases which reduces the accuracy of MLP, but because the MLPWD neglects the noise it's accuracy doesn't affect much.

Table 4, Figs. 16 and 17 show the prediction results for the periodic workload pattern. According to the results, the MLPWD algorithm outperforms the SVM and the MLP algorithms in the environments with the periodic workload pattern. The only difference between the MLPWD and the MLP algorithms is the risk minimization approach. Therefore, the results show that for the periodic workload pattern it is better to use the SRM principle.

Furthermore, the prediction results for the growing workload pattern are shown in Table 5, Figs. 18 and 19. The results show the SVM algorithm has better accuracy compared with MLPWD and MLP in the environments with the growing workload pattern. However, similar to the results of the periodic pattern, the MLPWD algorithm outperforms the MLP algorithm for predicting the growing workloads. This indicates that the SRM principle is more suitable compared to the ERM principle for predicting the growing workloads.

Based on the results, the SRM principle is more accurate than the ERM principle for forecasting the predictable workload patterns (i.e., the periodic and the growing workloads).

Hypothesis 1.b: the ERM principle performs better in the environments with the unpredictable workload patterns

According to Table 6, Figs. 20 and 21, the MLP algorithm has a better prediction accuracy compared with the SVM and the MLPWD algorithms in the environments with the unpredictable workload pattern. The MLP algorithm uses the ERM principle and tries to cover all of the training data. On the other hand, the MLPWD and the SVM algorithms use SRM principle and try to reduce the complexity by finding a smooth curve to cover the training data. Since the unpredictable data has a fluctuating nature, the SRM principle assumes some of the training data points are noise and removes them from the training dataset. As the result, in the environments with many fluctuations, the MLPWD and the SVM algorithms assume that the spikes are noise in the data. Therefore, the MLPWD and the SVM algorithms do not capture the spikes in the dataset. The result is that in the environments with the unpredictable workload pattern the MLP algorithm outperforms the MLPWD and the SVM algorithms. This confirms *hypothesis 1.b*.

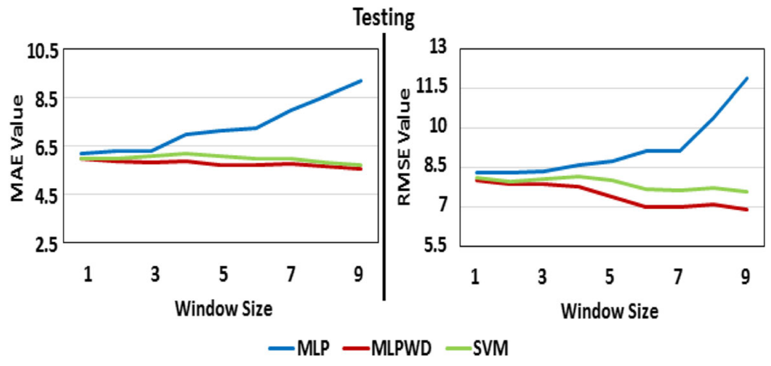

Fig. 17 MAE and RMSE values in the testing phase (periodic pattern)

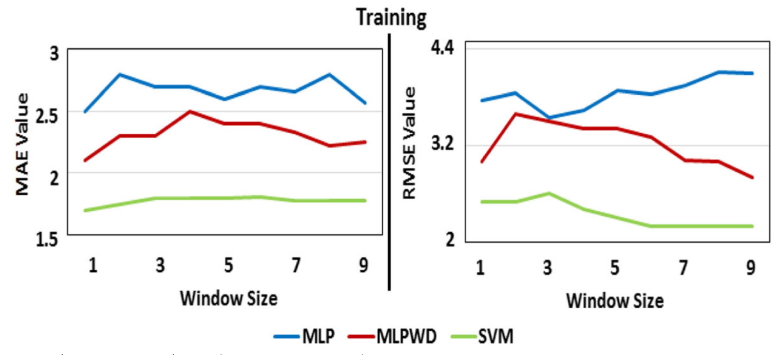

Fig. 18 MAE and RMSE values in the training phase (growing pattern)

Hypothesis 1.c: increasing the window sizes does not have a positive effect on the performance of the SRM principle

According to Tables 4 and 5 in the periodic and the growing environments increasing the window size does not affect the accuracies of the MLPWD and the SVM algorithms. The reason is because the SRM principle controls the prediction model's complexity by neglecting some of the training data points. As a result, increasing the window size neither increase nor decreases the accuracy of the prediction models.

By increasing the window size in the unpredictable environments the MLPWD accuracy slightly improves while the SVM accuracy slightly reduces (Table 6). However, the changes in the accuracies of the MLPWD and SVM in the unpredictable environments are negligible. Therefore, it can be concluded that for all of the workload patterns, increasing the window size has no substantial effect on the prediction accuracy of the SRM principle.

Hypothesis 1.d: Increasing the window size improves the performance of the ERM principle in the unpredictable environments and has no positive effect of the performance the ERM principle in the predictable environments.

Based on Fig. 16, for the smaller window sizes in the periodic environment the MLP accuracy is close to the MLPWD and the SVM accuracies. However, by increasing the window size, the MLP accuracy decreases. Similar to the results of the periodic pattern, in the environments with the growing workload pattern, the MLP prediction accuracy has a decreasing trend but does not change too much by increasing the window size. This is because increasing the window size of the MLP algorithm leads to the overfitting issue which decreases the MLP accuracy. As shown in Fig. 16, during the training phase the MLP accuracy increases by increasing the window size. This shows the MLP algorithm becomes over fitted to the training dataset by increasing the window size. The results confirm that in the environments with the periodic workload pattern, increasing the sliding window size has no positive effect on the prediction accuracies of the ERM principle.

Unlike the growing and the periodic patterns, increasing the window size has a positive effect on the prediction accuracy of the MLP algorithm in the environment with the unpredictable workload pattern. The reason is that in the unpredictable environments there are many fluctuations in the data; therefore, the ERM prediction models cannot extract the relationships between the features thoroughly. Thus, increasing the window size increases the input size of the algorithms, which improves the ERM's prediction accuracies.

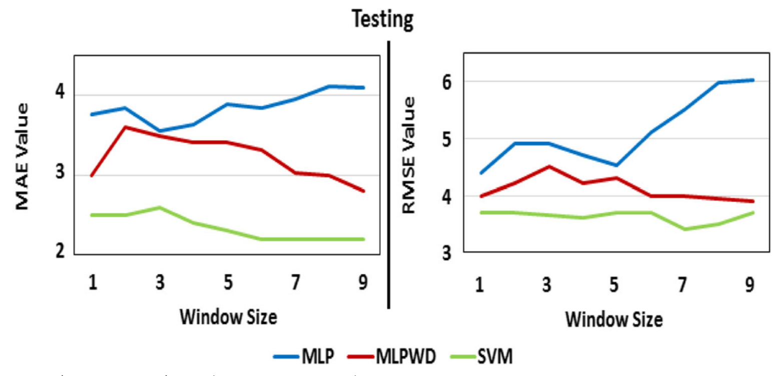

Fig. 19 MAE and RMSE values in the testing phase (growing pattern)

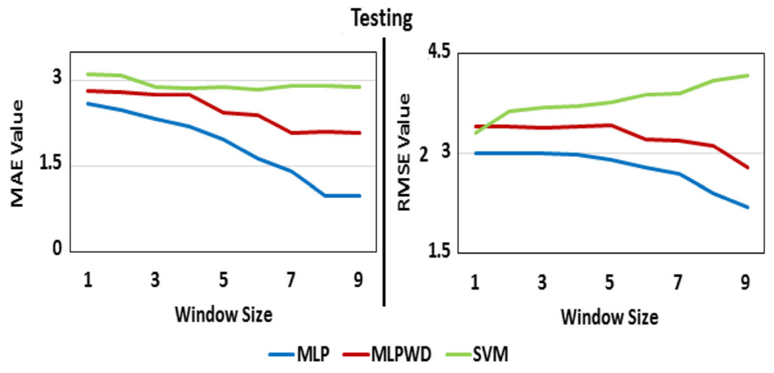

Fig. 20 MAE and RMSE values in the testing phase (unpredictable pattern)

Experimental results conclusion

The results of the experiments support the theoretical conclusion presented in Section Workload pattern effects on prediction accuracy of empirical and structural risk minimizations, which suggests the use of the SRM principle in the environments with the growing and the periodic workload patterns. In addition, the experimental results show that increasing the window size does not improve the SRM accuracy. On the other hand, for the environments with the unpredictable workload pattern, it is better to use the ERM principle with the bigger window sizes. According to the experimental results, Section Self-adaptive workload prediction suite proposes an autonomic prediction suite which chooses the most accurate prediction algorithm based on the incoming workload pattern.

Conclusions and future work

This paper proposed a self-adaptive prediction suite with an aim to improve the accuracy of predictive auto-scaling systems for the IaaS layer of cloud computing. The prediction suite uses the decision fusion technique and facilitates the selection of the most accurate prediction algorithm and the window size with respect to the incoming workload pattern. The proposed architecture used the strategy and the template design patterns which guarantees the automatic runtime selection of the appropriate prediction algorithm as well as detection of a suitable workload pattern and an appropriate window size. To lay out the theoretical foundation of the prediction suite, this paper proposed and evaluated a main hypothesis and four sub-hypotheses on the accuracy of several time-series prediction models in the IaaS layer of cloud computing. According to the main hypothesis, the prediction accuracy of the predictive auto-scaling systems can be increased by choosing an appropriate time-series prediction algorithm based on the incoming workload pattern.

To the best of our knowledge, the theoretical foundation of the predictive auto-scaling systems has not been investigated in the existing research works. Therefore, this paper performs a formal study of the theories that are closely related to the accuracy of predictive auto-scaling systems. To evaluate the main hypothesis, we have proposed four sub-hypotheses concerning the influence of the risk minimization principle on the prediction accuracy of the regression models in the environments with different workload patterns. To test these sub-hypotheses, the theoretical fundamentals of the prediction algorithms were investigated through analyzing the learning theory and the risk minimization principles.

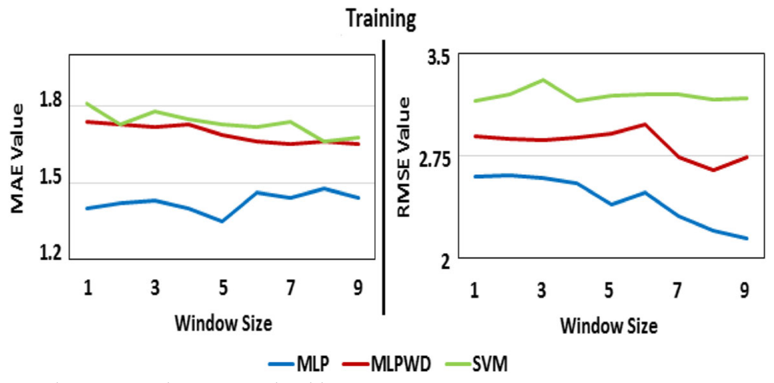

Fig. 21 MAE and RMSE values in the training phase (unpredictable pattern)

Based on the formal analysis, the structural risk minimization outperforms the empirical risk minimization for predicting the periodic and the growing workload patterns, but the empirical risk minimization is a better fit for forecasting the unpredictable workload pattern. Furthermore, experiments were conducted to validate the theoretical discussion. In the experiments, the influence of the risk minimization principle on the accuracy of the MLP and the MLPWD algorithms for predicting different workload patterns was examined. Moreover, the experiments compared the accuracy of the MLPWD and the SVM to isolate the impact of the regression model's structure on the prediction accuracy. The experimental results support the theoretical discussion. Also, the results show that increasing the sliding window size only has positive impact on the accuracy of the MLP algorithm in the environments with the unpredictable workload pattern. However, in other environments (i.e., growing or periodic workload patterns), increasing the window size does not improve the prediction accuracies of the MLP, MLPWD, and the SVM algorithms. The theoretical analysis and the experimental results demonstrated that using an appropriate prediction algorithm based on the workload pattern increases the prediction accuracy of the auto-scaling systems. Thus, based on the theoretical and experimental results in this paper, we can accept the main hypothesis that is, *the prediction accuracy of time-series techniques is positively impacted by using different prediction algorithms for the different cloud workload patterns.*

In the current work we assume that the database tier has no negative impact on the auto-scaling prediction accuracy. Investigating the impact of the database tier on the prediction accuracy warrants further research. In addition, we aim to investigate the relationship between the database tier auto-scaling and the workload patterns and the sliding window sizes. Finally, the autonomic elements in Fig. 10 will be re-designed to include more time series algorithms and possibly more work load patterns.

Endnotes

[1]*Shattering definition*: Model f with some parameter vector θ shatters a set of data points $(x_1, x_2, \dots, x_n)$ if for all assignments of labels to the data points there exists a θ such that the model f makes no error evaluating that set of data points.

Acknowledgements
We will like to express our thanks to departmental technical and administrative staff who provided resources and supports to the AYN during his PhD research work.

Authors' contributions
This research work is primarily based on AYN's PhD research and thesis report which was co-supervised by SAA and CL. All authors contributed to the technical aspects and the writing of the paper. AYN designed and implemented the experiments based on guidance from SAA and CL. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Received: 25 August 2016 Accepted: 19 January 2017
Published online: 03 February 2017

References

1. Nikravesh AY, Ajila SA, Lung C-H (2015) Evaluating sensitivity of auto-scaling decisions in environments with different workload patterns, Proceedings of the 39th IEEE International Computers, Software & Applications Conference Workshops., pp 690–695
2. Nikravesh AY, Ajila SA, Lung C-H (2015) Towards an autonomic auto-scaling system for cloud resource provisioning, Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems., pp 33–45
3. Ajila SA, Bankole AA (2013) Cloud client prediction models using machine learning techniques, Proceedings of the IEEE 37th Computer Software and Application Conference., p 143
4. Lorido-Botran T, Miguel-Alonso J, Lozano JA (2014) A review of auto-scaling techniques for elastic applications in cloud environments. Journal of Grid Computing 12(4):559–592
5. Bankole AA (2013) Cloud client prediction models for cloud resource provisioning in a multitier web application environment, Master of Applied Science Thesis, Electrical and Computer Engineering Department, Carleton University
6. Islam S, Keung J, Lee K, Liu A (2012) Empirical prediction models for adaptive resource provisioning in the cloud. Journal of Future Generation Computer Systems 28(1):155–165
7. Fehling C, Leymann F, Retter R, Schupeck W, Arbitter P (2014) Cloud computing patterns: fundamentals to design, build, and manage cloud applications, 1st edn. Springer-Verlag Wien publisher, ISBN 978-3-7091-1568-8
8. Workload Patterns for Cloud Computing (2010) [Online], Available http://watdenkt.veenhof.nu. Accessed 3 July 2010
9. Amazon Elastic Compute Cloud (Amazon EC2) (2013) [Online], Available http://aws.amazon.com/ec2/. Accessed 10 Feb 2013
10. RackSpace, The Open Cloud Company (2012) [Online], Available: http://rackspace.com. Accessed 12 June 2012
11. RightScale Cloud management (2012) [Online], Available: http://www.rightscale.com/home-v1?utm_expid=41192858-85.eCMJVCEGRMuTt8X6n9PcEw.1. Accessed 20 June 2012
12. Hasan MZ, Magana E, Clemm A, Tucker L, Gudreddi SLD (2012) Integrated and autonomic cloud resource scaling, Proceesings of IEEE Network Operation Management Symposium., pp 1327–1334
13. Kupferman J, Silverman J, Jara P, Browne J (2009) Scaling into the cloud, Technical report, Computer Science Department, University of California, Santa Barbara
14. Roy N, Dubey A, Gokhale A (2011) Efficient autoscaling in the cloud using predictive models for workload forecasting, Proceesings of 4th IEEE International Conference on Cloud Computing., pp 500–507
15. Herbst NR, Huber N, Kounev S, Amrehn E (2013) Self-adaptive workload classification and forecasting for proactive resource provisioning, Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering., pp 187–198
16. Benediktsson JA, Kanellopoulos I (1999) Classification of multisource and hyperspectral data based on decision fusion. Journal of IEEE Transactions on Geoscience and Remote Sensing 37(3):1367–1377
17. Local polynomial regression fitting. [Online], Available: http://stat.ethz.ch/R-manual/R-devel/library/stats/html/loess.html. Accessed 10 Feb 2010
18. Garlan D, Schmerl B (2002) Model-based adaptation for self-healing systems, Proceedings of the 1st Workshop on Selfhealing systems., pp 27–32
19. Sterritt R, Smyth B, Bradley M (2005) PACT: personal autonomic computing tools, Proceedings 12th IEEE International Conference and Workshops on Engineering of Computer-Based Systems., pp 519–527
20. Bigus JP, Schlosnagle DA, Pilgrim JR, Mills WN III, Diao Y (2002) ABLE: a toolkit for building multiagent autonomic systems. IBM Syst J 41(3):350–371
21. Littman ML, Ravi N, Fenson E, Howard R (2004) Reinforcement learning for autonomic network repair, Proceedings of International Conference on Autonomic Computing., pp 284–285
22. Dowling J, Curran E, Cunningham R, Cahill V (2006) Building autonomic systems using collaborative reinforcement learning. Journal of Knowledge Engineering Review 21(03):231–238

23. Gemma E, Helm R, Johnson R, Vlissides J (1994) Design patterns: elements of reusable object-oriented software, 1st edn. Addison-Wesley Professional publisher, ISBN 0201633612 (22nd printing, July 2001)
24. Wang S, Summers RM (2012) Machine learning and radiology. Journal of Medical Image Analalysis 16(5):933–951
25. Vapnik V (1922) Principles of risk minimization for learning theory, Proceedings of Advanced Neural Information Processing Systems Conference., pp 831–838
26. Vapnik V, Chervonenkis A (1978) Necessary and sufficient conditions for the uniform convergence of means to their expectations. Journal of Theory Probability 3(26):7–13
27. Sewell M (2008) VC-Dimension, Technical report, Department of Comuter Science University of Collage London
28. Sewell M (2008) Structural risk minimization, Technical report, Department of Computer Science, University College London
29. Yeh C, Tseng P, Huang K, Kuo Y (2012) Minimum risk neural networks and weight decay technique, Proceedings of 8th International Conference on Emerging Intelligent Computing Technology and Applications., pp 10–16
30. TPC-W benchmark. [Online]. Available: http://www.tpc.org/tpcw/. Accessed 10 Feb 2010
31. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA data mining software. Newsletter of ACM SIGKDD Explorations 11(1):10–18
32. Trevor H, Tibshirani R, Friedman RJ (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer Series in Statistics publisher, ISBN 978-0-387-84858-7
33. Witten I, Frank E (2011) Data mining practical machine learning tools and techniques with Java implementations, 3rd edn. Morgan Kaufmann publisher, ISBN 978-0-12-374856-0 (pbk)
34. Chai T, Draxler R (2014) Root mean square error (RMSE) or mean absolute error (MAE) – arguments against avoiding RMSE in the literature. Journal of Geoscience Model Development 7(1):1247–1250