Journal of Grid Computing (2024) 22:46
https://doi.org/10.1007/s10723-024-09764-4

RESEARCH

Signature-based Adaptive Cloud Resource Usage Prediction Using Machine Learning and Anomaly Detection

Wiktor Sus · Piotr Nawrocki

Received: 13 February 2024 / Accepted: 10 April 2024

Abstract One of the challenges in managing cloud computing clusters is assigning resources based on the customers' needs. For this mechanism to work efficiently, it is imperative that there are sufficient resources reserved to maintain continuous operation, but not too much to avoid overhead costs. Additionally, to avoid the overhead of acquisition time, it is important to reserve resources sufficiently in advance. This paper presents a novel reliable general-purpose mechanism for prediction-based resource usage reservation. The proposed solution should be capable of operating for long periods of time without drift-related problems, and dynamically adapt to changes in system usage. To achieve this, a novel signature-based ensemble prediction method is presented, which utilizes multiple distinct prediction algorithms suited for various use-cases, as well as an anomaly detection mechanism used to improve prediction accuracy. This ensures that the mechanism can operate efficiently in different real-life scenarios. Thanks to a novel signature-based selection algorithm, it is possible to use the best available prediction algorithm for each use-case, even over long periods of time, which would typically lead to drifts. The proposed approach has been evaluated using real-life historical data from various production servers, which include traces from more than 1,500 machines collected over more than a year. Experimental results have demonstrated an increase in prediction accuracy of up to 21.4 percent over the neural network approach. The evaluation of the proposed approach highlights the importance of choosing the appropriate prediction method, especially in diverse scenarios where the load changes frequently.

P. Nawrocki (✉)· W. Sus
Faculty of Computer Science, AGH University of Krakow, al. Adama Mickiewicza 30, Krakow 30-059, Poland
e-mail: piotr.nawrocki@agh.edu.pl

W. Sus
e-mail: wiktor.sus@agh.edu.pl

Keywords Cloud computing · Resource usage prediction · Signature analysis · Machine learning · Long-term prediction · Anomaly detection

1 Introduction

An important aspect of cloud computing is managing the resources available, both from the cloud owner's and the customer's perspectives [17]. These resources are typically reserved to meet the maximum predicted load that could be experienced, with the assumption that this threshold will never be exceeded, in order to ensure that no service outage will occur as a result of insufficient resources. This results in increased costs for the customer and for the provider, as it means unnecessarily reserving resources that could be assigned to other customers.

With advancements in automated prediction systems and the growing popularity of infrastructure as a service

(IaaS), such manual reservation is often relinquished in favor of automatic resource management solutions. These can be further divided into two groups – reactive [21] and proactive [15]. Both of these automatically increase and decrease the amount of resources reserved based on the demands, the difference being the time at which it is done. Reactive solutions adjust the reservations based on current system load, while proactive systems adjust them based on predicted load. The disadvantage of a reactive system is that there is a noticeable time overhead after every change, which requires an appropriate buffer so that the system is not overloaded during the initial increase in load. Proactive solutions can reduce this buffer, which leads to a decrease in costs. However, these solutions must accurately predict the amount of resources required by each individual customer so that no under-provision is experienced while keeping the costs of those resources as low as possible at the same time.

The objective of this research is to develop a novel algorithmic approach based on machine learning to accurately predict the amount of resources needed by a customer's virtual machines to fulfill the aforementioned goal of providing sufficient resources while keeping costs low. The concept of ensemble learning (which uses multiple machine learning algorithms) is not new and has been studied in the time-series forecasting area; however, the selection variant (such as Dynamic Regressor Selection (DRS) [5]) is fairly new and research on optimal selection approaches is ongoing. The ensemble proposed here utilizes a novel signature-based selection method, which is able to extract various features from input data to increase its dimensionality and allow the system to correlate certain trends represented by these signatures with the most competent prediction model available.

The major contributions of this paper can be summarized as follows:

- a novel signature-based selection method that allows additional features to be extracted from low-dimensional data in order to identify usage scenarios;
- a self-adapting algorithm has been proposed that can utilize historical data to select the most appropriate available prediction models in a continuous fashion;
- performance has been evaluated using data obtained from more than 1,500 virtual machines from 2 providers;
- the experiments conducted showed up to 21.4 percent increase in prediction accuracy over the neural networks method.

The rest of this paper is structured as follows: Section 2 contains a description of related work, Section 3 focuses on the description of an adaptive resource usage prediction system and its components, Section 4 describes the experiments performed and their results, and Section 5 contains the conclusion and further work.

2 Related Work

The main component of the proposed framework is the signature generation method, which is responsible for matching up the best suited prediction models to the current state of the cloud computing cluster. The signatures are obtained using techniques from the pattern analysis domain. In this work, anomaly detection techniques are used to prepare suitable data for prediction. The literature describes various studies devoted to pattern analysis, anomaly detection and resource usage prediction.

2.1 Pattern Analysis

Pattern analysis is a data analysis method that allows for discovering patterns and regularities in data. The subject of pattern analysis is most often studied in the context of signal analysis and fingerprint generation. Fingerprint is a more specific type of pattern that should ideally characterize a single entity, in this case a sample or a sequence of samples.

Through pattern analysis, it is possible to recognize familiar patterns, enabling a range of useful applications. Typical approaches include classification, clustering, ensemble learning, regression, and other methods. Pattern analysis is especially useful in fields that commonly deal with regular types of data. Cloud computing is a good example of such a field as trends can be observed, especially in relation to user-generated traffic.

In this paper, the system leverages methods commonly employed in the signal analysis domain due to its general application to time-series data, and specifically methods used to discover patterns within signals. An example of such work is an algorithm often employed in content-based copy detection systems, such as audio copyright detectors, as presented in [25]. The method presented enables audio fingerprints to be generated that consist of multiple samples with varying degree of detail, which allows accurate detection even where noise is introduced into the audio sample. The approach presented is not the most recent, although for the purposes of the proposed signature generation mechanism, the techniques it involves are still valid and useful.

Using the assumption that specific trends in resource usage can be represented as a signal, it is possible to employ a similar technique to generate such fingerprints for them. The proposed solution's goal is to generate a signature that can correspond to multiple similar patterns for easier identification of a certain class of behaviors, as opposed to unique audio fingerprints.

Another such approach was analyzed in the context of cloud computing in [1], where the authors developed a solution that was capable of generating fingerprints characterizing a user's behavior. Based on this information, it was possible to classify the behaviors as normal or anomalous. The approach presented in this paper differs, as it is based on analyzing data from a cluster as opposed to the behavior of an individual user. The goal of pattern analysis in the context of signature generation is to extract as many features describing a window of samples as possible. While the research presented is not the most recent, the subject of pattern analysis as defined above has not been thoroughly explored to date.

The proposed approach utilizes techniques commonly used in signal processing, taking inspiration from the audio fingerprinting approaches presented above, to generate signatures based on the current cloud workload. The signature generation algorithm is different, however, since its goal is not to uniquely identify each workload, but rather certain types. Additionally, as opposed to the audio domain, in cloud computing multiple time series measurements (like hardware performance metrics) are often available, and these are also used in this approach. Together, this enables our solution to select the most appropriate prediction algorithm.

2.2 Anomaly Detection

The process of anomaly detection consists of identifying data elements that are outliers compared to the majority.

The analysis of various approaches and classes of anomalies was conducted in [9]. There, the authors find that classic and deep machine learning have proven to be among the most popular ones due to their high accuracy and robustness. Due to these findings, the presented approach utilizes deep learning techniques such as LSTM-based anomaly detection.

Most work in the anomaly detection area in the context of cloud computing resource usage prediction was carried out by the authors of this article in [22]. The solution presented in this paper utilizes the presented mechanisms to increase the accuracy of the model.

State-of-the-art mechanisms utilize the concept known as lifelong anomaly detection, which was analyzed by the authors in [6]. The idea is to utilize lifelong learning to constantly update the normal state, which describes the standard operating parameters of the model, while ”forgetting” the oldest data (by removing it). This ensures that the normal state is always up to date with the state of the system. The solution presented in this paper utilizes this approach to allow the anomaly detection mechanisms to adapt to the changes in the system. Another such solution is [27], in which the authors employ a novel approach which mimics the workings of the cerebral cortex in unsupervised learning. The solution can both learn and detect anomalies online. It does not require large amounts of data and outperforms other evaluated models. However, both these solutions were not evaluated in the context of resource usage prediction.

Research is also being conducted on another type of anomaly detection, which utilizes prediction mechanisms for anomaly detection, thus allowing the system to prepare for upcoming anomalies. One such work is [32], in which three models were combined in ensemble (similarly to the approach presented in this article) to predict when the anomalies would occur. Another predictive solution was presented in [7]. The authors used highly accurate LSTM networks in stacked configuration and their bidirectional variants, which were tested on a load simulated within the OpenStack environment. Another such approach is [11], which puts more focus on analyzing not just anomalies but also the cloud's state in order to detect potential attacks, such as Spectre or Meltdown. The goal of the presented mechanism is to work with as many types of workloads as possible. The PCA method was used to obtain anomaly markers from hardware performance metrics. The above solutions show promise in utilizing predictive anomaly detection; however, they do not detect anomalies for the purpose of resource usage prediction and allocation, as opposed to the presented approach.

In [10], the authors present an unsupervised approach, which allows them to learn from unlabeled data, since it is difficult to ensure that the training datasets did not contain anomalies. The solution is unique in utilizing the topological data provided that describe the architecture of the underlying system, which are then used to enhance detection accuracy. However, this method does not leverage lifelong anomaly detection, which is needed for long-term unsupervised operation and is used in the presented approach.

All of the presented approaches are valid in the context of anomaly detection; however, none of them have been evaluated in the context of resource usage prediction. This area still requires more in-depth analysis, since predicting the resources required by a service is strongly linked to the quality of data that the prediction model can use, and this directly impacts its accuracy.

2.3 Resource Usage Prediction

Resource usage prediction is a general term for predicting the amount of resources required, for example by a cloud computing cluster, at a given point in future. The resources in this context are physical (or virtualized) parameters of the virtual machines, such as the amount of CPU cores, RAM, disk space and speed, network speed.

Machine learning algorithms are often used in research related to predicting cloud resource consumption. In [29], the authors propose a method of automatic preparation and training of prediction models using machine learning in time-series scenarios. In [24], the authors used FLNNs (*Functional Link Neural Networks*) trained with the PSO genetic algorithm to predict resource consumption. Selecting such a method with lower computational complexity shortened training time and yielded results faster. The LSTM model is a type of a recurrent neural network, which is known to work well with various types of data, often outperforming simpler models [13]. Also in [8], neural networks including BLSTM (*bidirectional LSTM*) were used for prediction purposes, which produced significantly better results than the reference ARIMA (*Autoregressive Integrated Moving Average*) model. A study on the prediction of CPU consumption using an RNN (*Recurrent Neural Network*) alongside evolutionary optimization algorithms was presented in [18]. The method used yielded better results compared to linear regression and a classical neural network. In [12], the authors present a method of training a neural network with the use of a genetic algorithm to dynamically adjust its parameters. The method was used to generate predictions for up to 60 minute intervals.

In recent studies, LSTM-based solutions have gained popularity. One such study is [14], in which the BiLSTM network is stacked with the GRU network to improve accuracy. The authors used the open-source Google dataset and achieved an MSE 5% lower than the reference individual models. A similar approach was also employed in [32], where the BiLSTM model is combined together with the GridLSTM prediction model, with the hybrid model using layers from both. In [7], the authors have used a different approach which combines convolutional neural networks with various models that are used in succession to preprocess data, before feeding them into the GRU network in multivariate prediction. The evaluation showed an improvement of 2-28% compared to baseline models. In [31], various neural network architectures were evaluated, where the BiLSTM showed the greatest improvement again. A different approach was presented in [9], where the LSTM network was used predicting container load per virtual machine, which was then used to balance the containers between the machines. All of these studies deal with prediction and many of the recent ones are leaning towards combined models or LSTM-based approaches; however, combinations of multiple heterogeneous models are still being explored. Additionally, none of the solutions presented incorporate anomaly detection mechanisms, and none deal with long-term prediction, with the longest time range analyzed being a month, due to the lack of sufficient public datasets. The approach presented in this article utilizes anomaly detection together with multiple diverse predictors, and evaluates its performance on a dataset that spans over a year. This gives a better understanding of long-term performance which in real-life scenarios may be reduced over time due to model drift.

In the context of long-term prediction, most of the studies were carried out by the authors of this paper. In [22], the authors describe a long-term prediction solution for a cloud computing system with additional anomaly detection mechanisms. The solution utilizes multiple algorithms to generate predictions, with the inclusion of a weighted average function that aggregates their outputs using statically defined weights. In this research, the impact of multiple prediction models and dynamic selection function is explored, which alleviates the problem of manually defining weights. Additionally, more accurate prediction algorithms are used.

In [30], the authors explore the possibility of long-term prediction via virtual machine plan reservation. The experimental solution utilizes a single deep learning-based prediction model. The solution proposed in this paper is capable of operating on lower-level data, managing cluster resources instead of reservation plans, and utilizes multiple prediction models at once for increased accuracy. The solution proposed in this paper utilizes multiple prediction algorithms, as well as additional mechanisms like anomaly detection and signature generation.

In [23], the authors describe preliminary research and propose a machine learning-based solution which employs multiple prediction models along with a selection module that decides which one should be used at which moment. The selection module proposed extracts certain predefined features, and subsequently the system correlates the samples with the best matching prediction models. This paper continues the preliminary research presented to date. The new selection algorithm is based on signatures, which work similarly as described above, although they contain more features to improve the chance of discovering relations between them and the competence of prediction models. Additionally, the solution proposed is not limited to operating in discrete steps, but is instead able to output predictions continuously, given samples spanning a defined period of time in order to warm up. The above solutions proposed by the authors enable not just long-term predictions, but also short-term ones.

Another work that employs a similar approach is [2], in which the authors propose the use of ARIMA and LSTM models together with a selection algorithm that decides whether to use the former or latter of the prediction models based on data analysis. If the data is highly random, then LSTM is a better choice, while if the data has a high degree of seasonality, ARIMA is better suited. In this research, a more advanced selection algorithm was employed to enable the automatic discovery of relationships between samples and prediction model competence. Additionally, it allows the user to seamlessly integrate new models without the need to manually add new selection criteria.

Research is also conducted that does not use resource consumption predictions but still allows optimization. In [19], the authors propose a genetic algorithm approach to optimizing the usage of cloud resources. The authors noted that their solution outperformed other analyzed approaches.

At the same time, studies are conducted on dynamic selection multiple classifier systems (DSMCSs). Such systems provide a taxonomy in which multiple classifiers or regressors (further referred to as models) are employed in an ensemble together with a selection function. The idea is that since traditional ensembles increase the accuracy of the entire ensemble, thus additionally selecting the most competent models for a given sample improves the accuracy further. The models employed can either be homogeneous, meaning a variation of a single prediction model – for instance a set of deep neural networks with varying architectures – or heterogeneous. A heterogeneous ensemble consists of a diverse set of prediction models, which is expected in the context of time-series forecasting, since models tend to specialize in different areas.

In [4], the authors present a comprehensive analysis of different methods used in DSCMSs, describing various ensemble types and selection methods. Most of the research, however, is focused on the classification task, which differs from regression. As shown in the comparison presented, the prevalent selection method utilized k-nearest neighbors search, and this requires multidimensional data as opposed to our solution, which can utilize a single metric and generate additional features from it through pattern analysis. Additionally, most often just a single metric (accuracy) is utilized for measuring competence for a given sample, whereas in this paper the solution includes additional metrics which measure over- and under-provision, since if these were to occur, they would have a negative business impact.

In [20], the authors presented a framework for the selection of regressors in the context of prediction. The method described makes it possible to utilize more than one competence metric and facilitates the preparation of such models. The selection method itself is based on the behavior of prediction models, whereas in this paper the signature generation system provides the data for prediction models.

In [26], a similar multi-criterion approach was presented where models are selected based on a combination of available competence metrics. The metrics are various statistical measurements, such as error, noise, variation, and deviation. In this paper, the multi-criterion selection is also explored; however, the metrics have been picked specifically for the cloud computing scenario.

In [28], the authors describe a method for dynamically selecting the most promising combination of prediction models from an ensemble. The combined competence is calculated using the nearest neighbors method. The solution proposed in this paper selects a single best prediction model instead of a set, using a signature generation method.

2.4 Summary

Significant research has been underway in the pattern analysis, anomaly detection and prediction areas, with many new advances in the dynamic selection area; however, the measurement of competence of individual prediction models in the context of resource usage prediction in cloud computing still remains a fairly new subject, especially in the context of long-term prediction. The most noticeable shortcoming of many recent approaches is the lack of long-term datasets, with the longest available amounting to around a month. This is insufficient to analyze the drift and long-term capabilities of the model. Additionally, at the time of writing there are few ongoing research projects concerning multiple prediction regressors, and there are none that also utilize behavioral selection functions with anomaly detection mechanisms. The selection function used in the multiple regressor system is also being studied, since while there are known good approaches, there is no single one that would be universally optimal.

This paper proposes the use of the signature generation algorithm to improve prediction accuracy in the context of cloud computing where the system may experience drifts over longer operation periods. This algorithm allows the system to utilize multiple predictors and select them based on current system usage patterns, making it possible for the system to work autonomously and dynamically respond to changes in usage without the need for long-term changes in configuration, while reducing drift. Additionally, the prediction solution is evaluated on real-life data from production clusters, which contain more than one year of samples, as opposed to most research relying on public datasets, which do not exceed one month.

3 Signature-based Adaptive Cloud Resource Usage Prediction System

The signature-based adaptive resource usage prediction system (SARUPS) is a type of DSMCS with a custom selection function consisting of four modules - anomaly detection, signature generation, selection and prediction ensemble. This paper presents a method for extracting usage signatures based on their individual characterization in terms of their patterns, and describes its impact in the prediction context.

Additional refinement is achieved through the employment of the anomaly detection method, which removes outliers in the sample window as these may introduce unwanted noise to predictions. Through the use of these signatures, which are matched with the best prediction model by their accuracy, under-provision and over-provision characteristics, it is possible to achieve greater prediction accuracy than when using raw metrics.

Additionally, by employing diverse prediction models, it is possible to conduct prediction in the cloud computing cluster over long periods of time, during which the machine learning models may experience drift. Drift occurs when the data that the machine learning model was trained upon becomes outdated, for example when the usage of the system changes – the scenario that the presented framework aims to handle properly by dynamically adapting to these changes.

The concept for the SARUPS is presented in Fig. 1 and the prediction process utilized by the SARUPS is presented in Algorithm 1.

The method for selecting the prediction model uses four modules: the *Anomaly Detection Module*, the *Signature Generation Module*, the *Selection Module* and the *Prediction Ensemble Module*. This paper proposes a novel approach, which is capable of enhancing overall prediction by utilizing the signatures generated from the filtered measurement of resource usage with additional anomaly detection mechanisms. This signature generation method serves as one part of the dynamic

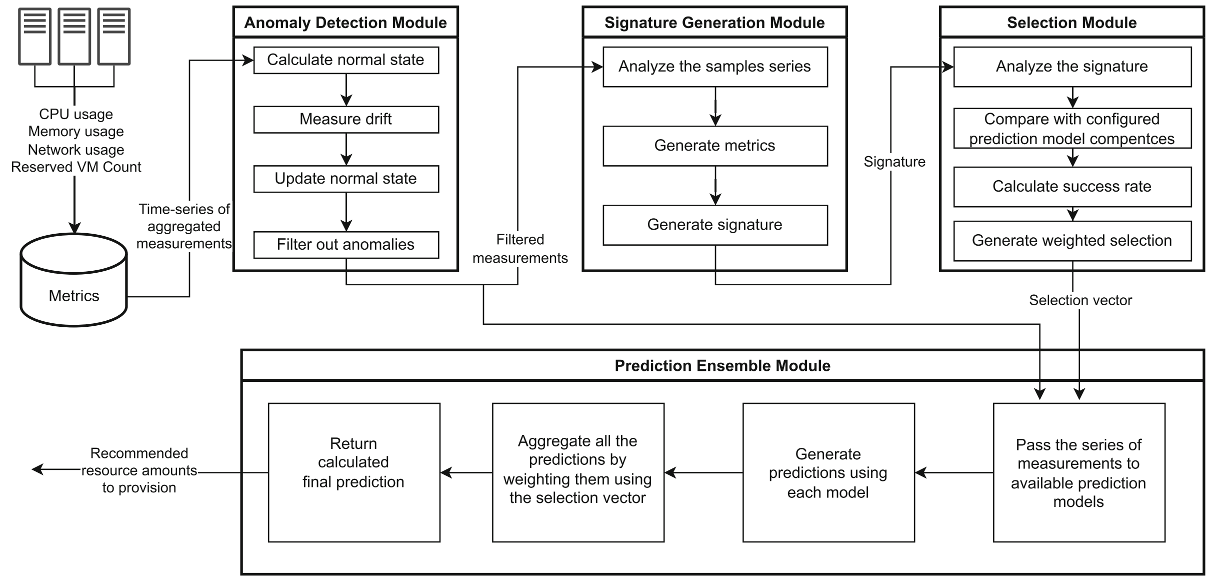
CPU usage
Memory usage
Network usage
Reserved VM Count
Metrics
Time-series of aggregated measurements
Anomaly Detection Module
Calculate normal state
Measure drift
Update normal state
Filter out anomalies
Filtered measurements
Signature Generation Module
Analyze the samples series
Generate metrics
Generate signature
Signature
Selection Module
Analyze the signature
Compare with configured prediction model compentces
Calculate success rate
Generate weighted selection
Selection vector
Prediction Ensemble Module
Pass the series of measurements to available prediction models
Generate predictions using each model
Aggregate all the predictions by weighting them using the selection vector
Return calculated final prediction
Recommended resource amounts to provision

Fig. 1 Concept of the SARUPS system

Algorithm 1 Prediction process in the SARUPS system, configured to work with a sample window of 24 hours.

```
1: wl ← 24                                        ▷ Window Length
2: ms ← MetricSource()
3: sig_gen ← SignatureGenerator()
4: ad ← AnomalyDetector()
5: selector ← SelectionModule()
6: ensemble ← PredictionEnsemble()
7: while True do
8:     sw ← ms.get_samples(wl)                    ▷ Obtain Sample Window
9:     fsw ← ad.filter_sample(sw);                ▷ Filter the anomalies from SW
10:    sig ← sig_gen.generate_signature(fsw)      ▷ Generate signature from FSW
11:    selected_predictor ← selector.select_predictor(sig)
12:    prediction ← ensemble.predict(selected_predictor, sw)
       ▷ Use selected predictor within ensemble to obtain predictions based on SW
13: end while
```

Algorithm 2 Anomaly Detection Module.

```
1: procedure FILTERSAMPLE(sample_window)
2:     da ← DDM(CurrentState)        ▷ Obtain drift analyzer from current state
3:     af ← AnomalyFilter            ▷ Prepare anomaly filter
4:     da.analyse(sample_window)     ▷ Use DA to decide whether the system has undergone a change
5:     if state of da equals ”Warning” then
6:         return af.filter(sample_window);
7:     end if
8:     if state of da equals ”Change” then
9:         da.update(sample_window)
10:    end if
11:    return sample_window;
12: end procedure
```

selection approach, the other part being the algorithm correlating the signatures with the most competent prediction model. Signature generation makes it possible to dynamically adjust the behavior of the ensemble to the current needs of the system, while anomaly detection mechanisms are used to filter out noise, therefore increasing accuracy.

The proposed framework can easily be adapted to work in different domains; however, the presented configuration is specifically tailored for the cloud computing context. The predictors were selected on the basis of their abilities, such as seasonality and trend detection, high accuracy, ability to work with multiple input metrics and diversity and high configurability, which makes them suitable for this scenario.

3.1 Metrics Database

The *Metrics* database is the source of aggregated data in the SARUPS, which is provided by the cluster, and is a general term for each measurable characteristic of the system, usually used for diagnostic purposes. These characteristics include, but are not limited to: CPU usage, memory usage, network activity and reserved VM count. Additionally, depending on cluster configuration it may be possible to utilize custom metrics, such as incoming request count per time unit, error logs, etc. The source should be able to provide those metrics, either in batches (sliding windows of *n* samples) or as individual samples. Additionally, it should be possible to obtain samples from a specified period if such a need arises.

In this paper, information about the current number of virtual machines reserved is used in univariate prediction (multiple metrics are used to determine a single value – how many virtual machines are required at the moment), since it is the most versatile and most regular metric available. Other metrics can be used for multivariate prediction; however, due to the use of a load balancer and the aggregation of metrics within the cluster, it is impossible to analyze individual virtual machines, and additionally their load should ideally stay constant in time. Using the virtual machine count yields an additional benefit: for comparison purposes, the cost of operating a single virtual machine in the context of biggest cloud service providers such as AWS or Azure, where the user pays for a specific virtual machine configuration, is constant.

3.2 Anomaly Detection Module

The *Anomaly Detection Module* is responsible for removing anomalous readings from the samples. It receives aggregated time-series data consisting of timestamps and the relevant metric values measured, and outputs data in the same format with the anomalies removed. This allows the system to generate predictions from denoised samples, and therefore increases prediction accuracy. The module keeps track of its internal state, allowing it to detect drift in the system's nominal state, which is used in comparison to the current state. By analyzing the nominal state and the current one it is possible to decide whether an anomaly was encountered and thus the data should be removed. If the system's nominal state changes, for example due to drift, it is dynamically updated. Algorithm 2 outlines the basics of the module's operation.

3.3 Signature Generation Module

The goal of the *Signature Generation Module* is to obtain a tensor of features that represent a certain type of behavior exhibited in the sample. Behavior types are a way of describing specific types of trends that appear in the given sample. From the technical point of view, there may be no changes at all in the metric, certain seasonalities may be present or the sample may be completely chaotic.

An example of typical behavior is a web server where resources are used more heavily during daytime, since more users are active during this time, while some systems run computational tasks constantly, resulting in nearly constant or chaotic resource usage. A cloud computing system may exhibit many such types of behaviors (for example in a multi-user computing cluster), and therefore by selecting the most accurate prediction model greater accuracy can be achieved than with a single model.

The *Signature Generation Module* is calibrated to return features extracted from the pattern (as opposed to a unique fingerprint) in order to allow the system to recognize similar patterns. The module obtains time-series data from the *Anomaly Detection Module* and generates signatures on this basis. Typical approaches leverage multi-dimensional data and clustering methods to calculate competence, while the proposed approach can also work with low-dimensional data. A sample consists of a series of measurements of configurable length, which makes it possible to analyze the trends exhibited by the system during a given period. The features contain various metrics, which are local to each sample. These include amplitude, highest value, lowest value, spectral entropy, cycle length (if found, otherwise 0).

Based on the signature (line 8 in Algorithm 1), the SARUPS selects the most suitable algorithm. Spectral entropy is a measure of spectral power density in the signal. The idea is that a sample (a set of measurements made over time) with higher randomness contains more information and therefore has a higher entropy. In terms of trends, low entropy is correlated with a sample with relatively minor changes, and thus low entropy would mean a more stable usage pattern (easy to predict) and high entropy would mean a chaotic usage pattern (hard to predict). Spectral entropy is based on Shannon entropy [16] and calculated as follows. First, the spectrum $X(\omega_i)$ is calculated, where ω_i is a single measurement in the sample. Subsequently, the Power Spectral Density (PSD) [3] is calculated as:

$$P(\omega_i) = \frac{1}{N}|X(\omega_i)^2 \tag{1}$$

where N is the number of bins. The calculated PSD needs to be normalized so it can be viewed as a Probability Density Function:

$$p_i = \frac{P(\omega_i)}{\sum_i P(\omega_i)} \tag{2}$$

Finally, Power Spectral Entropy (PSE) can be calculated as:

$$PSE = -\sum_{i=1}^{n} p_i \ln p_i \tag{3}$$

To calculate the length of the cycle, the Fast Fourier Transform is used and the frequency of highest peak is used (if different from 0).

3.4 Selection Module

The *Selection Module* (further referred to as selector) serves as the core of the system along with the *Signature Generation Module*, deciding which predictor should be used based on the signature provided. The selector algorithm receives the signature and on the basis of its training with model results, it can decide which of the models available is the most competent at the given time. The signature is a vector that contains various features provided by the generation algorithm. It outputs a vector containing confidence values for all available models which can be used to calculate the final prediction. In this paper, the selector is a machine learning algorithm, more specifically a Random Forest. It was trained in a supervised manner on the scores achieved by individual predictors and the signature accompanying the given timeframe, as well as on over- and under-provisioning metrics. The objective is to select the prediction model with highest accuracy and the lowest over-/under-provision, as this metric was also included in the dataset based on the predictors' individual results. This allows the selector to discover the relation between the signature, which is based on the behavior of the actual system, and prediction accuracy. The *Selection Module* returns the selected predictor, which is its index in the ensemble array.

3.5 Prediction Ensemble Module

The *Prediction Ensemble* consists of heterogeneous machine-learning prediction models due to their high accuracy with time-series data and the ability to discover non-obvious patterns in loosely coupled data. The goal of utilizing heterogeneous models is to provide a diverse set of prediction models that are well-suited to work in various scenarios, which are differentiated with the help of the aforementioned pattern analysis algorithms. The models selected are Convolutional Neural Network, Dense Neural Network, Long-Short Term Memory (LSTM) and the Autoregressive Recurrent Neural Network based on LSTM. The frequently used ARIMA model was skipped, as it requires calibrating for each scenario based on the problem and is not well-suited to continuous operation, only allowing predictions based on the initial fit, additionally being very sensitive to noisy data.

The algorithms selected are in general highly accurate in the context of prediction based on historical data, capable of working with seasonalities and discovering trends. If the models are to be able to predict seasonal data, it is important to provide them with a sufficient amount of historical lag to make the predictions. This can either be configured manually, provided that the cycle is known, or calculated using techniques like Fast Fourier Transformation. In this paper, a one-day cycle was selected, with data being sampled at 1 hour intervals, and therefore the prediction models are provided with 24-sample windows and output 24 samples of predictions. This choice was dictated by two factors: the first is the dataset sampling frequency, which is roughly 5 minutes. The second reason is that the cluster provides real-time service to users who interact with it with varying intensity throughout the day. This leads to noticeable cycles during the day, which can be modeled. It is possible to select a greater frequency to allow the system to respond to changes more quickly.

The module receives the same data that is being fed to *Signature Generation Module* together with the selection vector from *Selection Module* and uses this data to generate predictions and combine them after applying confidence values to each of them. The final result is the recommended amount of resources needed in future.

4 Evaluation

The proposed SARUPS is implemented in the Python3 language due to its flexibility, community support and a wide choice of analytic libraries. The framework used for training was Keras with Tensorflow. All training was conducted using a consumer-grade PC running the Linux operating system, using the AMD Ryzen 7 5700X CPU @3.40GHz with 32GB of RAM.

The datasets are real-life historical data obtained from production clusters during a period spanning more than a year. The servers provide user-facing services (including web servers), and therefore their load is reliant on user interactions. The datasets contain individual data from more than 1,500 virtual machines with various simple metrics, such as CPU usage, memory usage, disk access and network usage, and more complex ones such as the number of virtual machines reserved, count of diagnostic errors per minute and count of requests per minute per single virtual machine – all of those provided by the orchestration backend, which conducts autoscaling. The autoscaler takes into account the average amount of resources such as CPU and RAM and when the thresholds are crossed, it adds or removes machines to maintain optimal load. The resulting virtual machine (VM) count is recorded as a metric. The datasets contain roughly one year of real-life data from two different production clusters, used in different scenarios. No synthetic datasets were used at any point.

4.1 Dataset and Model Configuration

The datasets were split into two sets of training and validation datasets by date (90% of training data and 10% of validation data), since both the prediction algorithms and selection algorithm require training. The most recent part of the dataset (about two months) was left out for evaluation purposes. This ensures that every machine, potentially with varying usage behaviors, is included in the training data; however, the disadvantage is that important parts of data, which can contain seasonal trends, may potentially be missed.

All of the available predictors were trained on the first training dataset and validated on the corresponding validation set. For the sake of simplicity, only the count of virtual machines reserved at any single time was used for experiments, since the main goal is to demonstrate the increased accuracy of the ensemble over its individual components. The calibration of hyper-parameters was carried out via cross-validation. The loss metric used was mean squared error with Adam optimizer. Subsequently, the prediction models were used to generate predictions and their competence was measured on the second training dataset, never seen before by the prediction models, to obtain training data for the selection system. Finally, the entire ensemble was evaluated on the remaining data.

4.2 Goals of the Experiments

The main evaluation goal is to ascertain whether a signature-based ensemble algorithm performs better than individual predictors of which it is composed and whether the signature-based selection method outperforms typical approaches. The secondary goal is to determine the usefulness of the approach in different scenarios, in this case in a stable environment (experiencing fairly cyclical loads) and in a chaotic one (where there are no trends that would be easy to spot). The visualization of all prediction models compared to the actual usage of virtual machines is presented in Fig. 2. Compared to the other models available, the ensemble model follows the actual line most accurately. Additionally, its accuracy over longer time periods has been evaluated and is presented in Fig. 3.

This is easy to verify, as it is possible to compare the baseline performance of each individual predictor and compare them against the performance of the SARUPS algorithm. Performance is measured as the Mean Square Error (MSE) of the prediction – the actual value for each time step, with the inclusion of weighted under- and over-provision metrics. A lower MSE value means a smaller difference between the prediction and actual data, and therefore higher accuracy. The *under-provision* and *over-provision* metrics were measured as an average of absolute positive and negative difference, and are defined in Algorithm 3. The unit for all metrics is the number of virtual machines, for example under-provision equal to 6 means that the system would need 6 more virtual machines to operate properly. On the

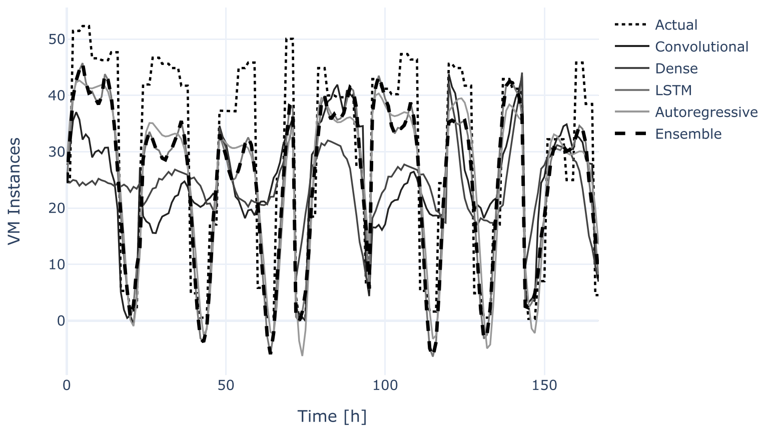

Fig. 2 Visualization of prediction accuracy of different models

other hand, significant over-provision means that the additional machines are not needed, but generate costs.

When accuracy is high, these metrics are close to zero, but in a scenario where it is difficult to select a competent algorithm (for example due to high data randomness), they support making the decision which algorithm should be used to reduce the risk of under-provision, which can be problematic from a cloud user's perspective, as under-provision may lead to service outages, because there are not enough resources – virtual machines – to handle the incoming traffic and this may lead to the service being unresponsive or to its complete outage.

4.3 Cloud Resource Usage Prediction

Experiments were carried out for highly regular data (with stable daily and weekly cycles), and also for

Algorithm 3 Calculation of over- and under-provision supporting metrics.

```
1: actual_values: Array;
2: predicted_values: Array;
3: up_sum ← 0;
4: op_sum ← 0;
5: up_count ← 0;
6: op_count ← 0;
7: for $i$ from 0 to length(actual_values) do
8:     if $predicted\_values[i] < actual\_values[i]$ then
9:         up_sum ← up_sum + (actual_values[i]
10:          - predicted_values[i]);
11:        up_count ← up_count + 1;
12:    end if
13:    if $predicted\_values[i] > actual\_values[i]$ then
14:        op_sum ← op_sum + (predicted_values[i]
15:          - actual_values[i]);
16:        op_count ← op_count + 1;
17:    end if
18: end for
19: average_up ← abs(up_sum / up_count);
20: average_op ← abs(op_sum / op_count);
```

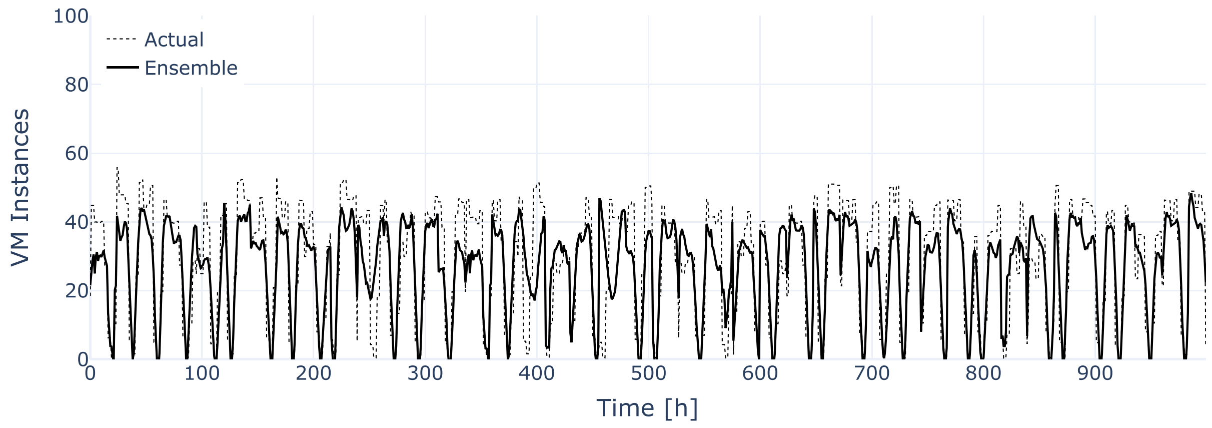

Fig. 3 Visualization of long-term prediction of Ensemble model

highly random data in order to compare the performance of the system for data that are, respectively, relatively easy and difficult to predict. The evaluation consists of analyzing the performance of the ensemble with different selection methods and the signature-based ensemble compared to its individual components. They should be analyzed from the point of MSE, under-provisioning and then over-provisioning since the system was calibrated to prioritize these metrics in this order. These measurements use virtual machines as their unit, which is useful for interpretation purposes – under-provision and over-provision show directly how many virtual machines were needed, but not requested, or reserved unnecessarily. The selection method comparison includes the evaluation of three different selection methods – neural network-based, which uses a simple dense network to correlate samples with the most competent algorithm, completely random selection and finally the signature-based selection method.

First, a stable dataset was analyzed. In this scenario, it is clear that utilizing the selection method yields noticeable improvements, as both neural network-based and signature-based methods outperform the random selection method, with marginally better results (by about 1 percentage point) for the neural network-based method. Both of them achieved MSEs of roughly 3 percentage points, while for the random method the value was 6 percentage points. The results are similar for both under- and over-provision for both of these methods.

Subsequently, the chaotic dataset was analyzed. First, selection method performance was examined. While the overall results are noticeably worse (on the order of 20 virtual machines), the signature method outperforms both random (by about 3 machines or 12%) and the neural network method (by 6 virtual machines or 21%), although with relatively high over-provision. The data obtained via these experiments is presented in Table 1. While the relative differences of MSE in the stable dataset are small, the best performing algorithm is Neural Network, closely followed by Signatures. Random is higher in MSE but also in Under-Provision, with Neural Network and Signatures being close. Over-Provision is almost identical. This establishes that employing a tailored approach instead of randomly selecting prediction algorithms does indeed increase accuracy, although if ensemble algorithms are already accurate to start with, the differences are small. The Signatures algorithm might be performing worse due to insufficient calibration. In the chaotic dataset, the MSE is noticeably higher for all algorithms, with Signatures as the best one, although its Under-Provision is marginally higher than Neural Network's which is the second best. Over-Provision is almost equally high. These results might suggest that the Signatures algorithm performs better when there is a use-case for multiple prediction algorithms.

Next, the performance of the ensemble with the signature-based selection method was compared with its individual components to verify whether it was possible for this method to outperform their individual results. The experiment included comparison of the MSE of the individual prediction models – Convolutional Neural Network (*Convolutional*), Dense Neural Network (*Dense*), Long-Short Term Memory Recurrent Neural Network (*LSTM*), Autoregressive Recurrent Neural Network (*Autoregressive*) and Ensemble with signature-based selection (*Ensemble*). In this experiment, the two best results are achieved by the LSTM and Ensemble models, with a difference of about 2 virtual machines between them. Subsequently, the ensemble with the signature selection method is compared to individual prediction models. Again, the MSE of the ensemble model with the signature selection method is the best, with a difference of 1 virtual machine compared to its best component. The results of those experiments are presented in Table 2.

Table 1 Comparison of different selection methods metrics over two different datasets

	Stable dataset			Chaotic dataset		
	MSE	Under-provision	Over-provision	MSE	Under-Provision	Over-provision
Random	6	20.3	12.1	35	3.6	52.4
Neural Network	2.5	14.8	11.2	28.1	2.6	53
Signatures	3.2	15.1	11.6	22.1	4.8	54.2

Table 2 Comparison of prediction MSE over two different datasets

	Stable	Chaotic
Convolutional	6.7	26.1
Dense	11	24.2
LSTM	2.7	28.4
AutoRegressive	4.7	22.3
Ensemble	4.5	20.3

It is clear that the selection method increases accuracy, with fairly similar results for all non-random methods for the stable (easy to predict) dataset, and noticeably better MSE results for the chaotic dataset. However, with the regular dataset, the improvements are negligible in the evaluated scenario, although the employment of drift detection may be utilized to increase the performance of the network over a longer period of time when service usage may undergo changes. If one of the predictors performs well, it will continue to perform in this way as long as the usage pattern does not change. The real benefit of employing the ensemble-based solution lies in utilizing it in evolving systems where resource usage is constantly changing, since multiple prediction models with diverse competences can outperform individual models.

Finally, to assess the efficiency of the solution developed in the context of cost savings, a rough estimate of possible savings was generated. The pricing is based on a popular cloud service provider's billing model which assumes a flat price per instance per hour. Using such billing, at the assumed rate of \$0.04 per hour (based on Amazon AWS pricing) the monthly cost of running the experimental cluster was calculated as the sum of virtual machine counts during individual hours. Then, using the same approach, costs were calculated for this cluster when run using the predictions generated by the Ensemble model with the Neural Network and Signatures selection methods. The final result is \$917.67 for the default autoscaler, compared to \$861.12 for the Ensemble model with the Neural Network and \$851.08 for the Ensemble model with the Signatures selection method. In conclusion, savings under this common billing model amounted to more than 7% using the Ensemble model with the Signatures selection method, which would result in significant savings for large systems using cloud services.

4.4 Anomaly Detection

The evaluation of anomaly detection was carried out both for the stable and chaotic datasets, using a fairly standard and simple anomaly filter. The primary goal was to ascertain whether using anomaly detection yields any noticeable improvements. Experiments on the stable dataset generated almost identical results, with some improvements for anomaly detection, although those were negligible. In the chaotic dataset, however, the differences were much more noticeable, with the Ensemble algorithm achieving an MSE lower by 20 percentage points than in an identical scenario without anomaly detection (a decrease from 44.87 to 34.96 virtual machines). The Anomaly Detection mechanism had a noticeable impact on almost all the models (on the order of 10-15%), with the exception of LSTM, which was quite accurate from the beginning. The results suggest that such a mechanism can indeed impact a model's learning abilities. This approach is similar to employing dropout layers in convolutional networks.

5 Conclusion

In this paper, the usage of a signature-based adaptive prediction algorithm is explored and its accuracy is compared to traditional solutions. The proposed solution performs better than a single prediction algorithm for a broader spectrum of use-cases. The algorithm proposed achieved a 21.4 percent increase in accuracy over directly correlating prediction models to samples via a neural network (Table 1, comparison of Neural Network and Signatures MSE in the Chaotic dataset). Additionally, the authors analyzed various predictors and evaluated their suitability for resource usage prediction in cloud computing as well as their competence for various usage patterns discovered through pattern analysis. The competence of the proposed system was evaluated on two different real-life datasets, without the use of synthetic data, to ascertain its performance in various scenarios. Finally, the comparison included new metrics not seen in the literature analyzed, such as under- and over-provision, which are useful in cloud computing scenarios since they help uphold service level agreements.

The experiments conducted show that the ensemble approach is viable and it performs best in chaotic scenarios where prediction via traditional means proves difficult. The increase in accuracy is not extreme, but noticeable. It is important to keep in mind that the results were achieved for basic implementation without precise model tuning. Moreover, employing the anomaly detection method yields further improvements of prediction accuracy, which are most pronounced in unstable environments where the data provided by the system contains anomalies. The research highlights the importance of selection methods in a multiple classifier system, since they are the crucial component with a significant impact on overall accuracy. Additionally, it presents the advantages of utilizing pattern recognition in prediction. The proposed system can be adapted for use in the cloud as an auto-scaling solution to decrease over-provisioning costs.

The most important limitations of the research conducted were primarily related to the inadequate quantity and diversity of data. This led to possibly insufficient fine-tuning of the models, failure to fully analyze the impact of anomaly detection, the absence of more advanced pattern analysis, the absence of multivariate evaluation and the inability to evaluate drift-resilience capabilities of the system.

The most important directions of further research planned by the authors, after obtaining new appropriately diversified data, are fine-tuning the prediction algorithms, improving anomaly detection algorithms for the multi-variate approach and fine-tuning the signature generation algorithm which could yield significant improvements in overall system accuracy. Additionally, the behavior of the trained system during longer periods require further evaluation to ascertain whether the drift detection method yields noticeable improvements, especially in the case of seasonal events that render prediction models inaccurate. Finally, it may be beneficial to incorporate anomaly detection mechanisms in other modules, such as prediction outputs or the model selection method to eliminate errors.

Acknowledgements The research presented in this paper was supported by funds from the Polish Ministry of Science and Higher Education allocated to the AGH University of Krakow. The authors would like to thank Ringier Axel Springer Poland and Nokia for providing the data used in the tests.

Author contributions W.S.: Methodology, Software, Investigation, Writing - original draft. P.N.: Supervision, Conceptualization, Methodology, Writing - review & editing.

Funding Information The research presented in this paper was supported by funds from the Polish Ministry of Science and Higher Education allocated to the AGH University of Krakow.

Data Availability Statement The data that supports the findings of this study is available from Ringier Axel Springer Poland and Nokia, but restrictions apply to the availability of this data, which was used under license for the current study, and thus is not publicly available. The data is, however, available from the authors upon reasonable request and with permission of Ringier Axel Springer Poland and Nokia.

Declarations

Ethics Approval and Consent to Participate Not applicable.

Consent for Publication Not applicable.

Competing interests The authors declare no competing interests.

References

1. Albayrak, S., Camtepe, S.A., Edman, M., et al.: Host-based anomaly detection via resource usage signatures. Tech. rep., Distributed Artificial Intelligence Laboratory - Technische Universitat Berlin, Berlin, Germany (2009)
2. Anupama, K.C., Shivakumar, B.R., Nagaraja, R.: Resource utilization prediction in cloud computing using hybrid model. Int. J. Adv. Comput. Sci. Appl. **12**(4) (2021). https://doi.org/10.14569/IJACSA.2021.0120447
3. Bisina, K.V., Azeez, M.A.: Optimized estimation of power spectral density. In: 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 871–875 (2017). https://doi.org/10.1109/ICCONS.2017.8250588
4. Cruz, R.M., Sabourin, R., Cavalcanti, G.D.: Dynamic classifier selection: recent advances and perspectives. Inf. Fusion **41**, 195–216 (2018). https://doi.org/10.1016/j.inffus.2017.09.010
5. Cruz, R.M., Souza, M.A., Sabourin, R., et al.: Dynamic ensemble selection and data preprocessing for multi-class imbalance learning. Int. J. Pattern Recognit. Artif. Intell. **33**(11), 1940009 (2019)

6. Faber, K., Corizzo, R., Sniezynski, B., et al.: Lifelong learning for anomaly detection: new challenges, perspectives, and insights. arXiv:2303.07557 (2023)
7. Girish, L., Rao, S.K.: Anomaly detection in cloud environment using artificial intelligence techniques. Computing **105**(3), 675–688 (2023)
8. Gupta, S., Dileep, A.D., Gonsalves, T.A.: Online sparse blstm models for resource usage prediction in cloud datacentres. IEEE Trans. Netw. Serv. Manage. **17**(4), 2335–2349 (2020). https://doi.org/10.1109/TNSM.2020.3013922
9. Hagemann, T., Katsarou, K.: A systematic review on anomaly detection for cloud computing environments. In: Proceedings of the 2020 3rd Artificial Intelligence and Cloud Computing Conference. Association for Computing Machinery, New York, NY, USA, AICCC '20, pp. 83–96 (2021). https://doi.org/10.1145/3442536.3442550
10. He, Z., Chen, P., Li, X., et al.: A spatiotemporal deep learning approach for unsupervised anomaly detection in cloud systems. IEEE Trans. Neural Netw. Learn. Syst. **34**(4), 1705–1719 (2023). https://doi.org/10.1109/TNNLS.2020.3027736
11. He, Z., Hu, G., Lee, R.B.: Cloudshield: real-time anomaly detection in the cloud. In: Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy. Association for Computing Machinery, New York, NY, USA, CODASPY '23, pp. 91–102 (2023). https://doi.org/10.1145/3577923.3583639
12. Kumar, J., Singh, A.K.: Workload prediction in cloud using artificial neural network and adaptive differential evolution. Futur. Gener. Comput. Syst. **81**, 41–52 (2018). https://doi.org/10.1016/j.future.2017.10.047
13. Kumar, J., Goomer, R., Singh, A.K.: Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. Procedia Comput. Sci. **125**, 676–682 (2018). https://doi.org/10.1016/j.procs.2017.12.087
14. Li, X., Wang, H., Xiu, P., et al.: Resource usage prediction based on bilstm-gru combination model. In: 2022 IEEE International Conference on Joint Cloud Computing (JCC), pp. 9–16 (2022). https://doi.org/10.1109/JCC56315.2022.00009
15. Liao, P., Pan, G., Wang, B., et al.: Efficient proactive resource allocation for multi-stage cloud-native microservices. In: Tari, Z., Li, K., Wu, H. (eds.) Algorithms and Architectures for Parallel Processing, pp. 411–432. Springer Nature Singapore, Singapore (2024)
16. Lin, J.: Divergence measures based on the shannon entropy. IEEE Trans. Inf. Theory **37**(1), 145–151 (1991). https://doi.org/10.1109/18.61115
17. Malav, A., Gupta, S.K., Mahariya, S.K., et al.: Optimal resource management in cloud computing. AIP Conf. Proc. **2771**(1), 020040 (2023). https://doi.org/10.1063/5.0152298
18. Mason, K., Duggan, M., Barrett, E., et al.: Predicting host cpu utilization in the cloud using evolutionary neural networks. Futur. Gener. Comput. Syst. **86**, 162–173 (2018). https://doi.org/10.1016/j.future.2018.03.040
19. Mohapatra, S.S., Kumar, R.R., Alenezi, M., et al.: Qos-aware cloud service recommendation using metaheuristic approach. Electronics **11**(21) (2022). https://doi.org/10.3390/electronics11213469
20. Moura, T.J., Cavalcanti, G.D., Oliveira, L.S.: Mine: a framework for dynamic regressor selection. Inf. Sci. **543**, 157–179 (2021). https://doi.org/10.1016/j.ins.2020.07.056
21. Nawrocki, P., Smendowski, M.: Long-term prediction of cloud resource usage in high-performance computing. In: Mikyška, J., de Mulatier, C., Paszynski, M., et al. (eds.) Computational Science – ICCS 2023, pp. 532–546 . Springer Nature Switzerland, Cham (2023)
22. Nawrocki, P., Sus, W.: Anomaly detection in the context of long-term cloud resource usage planning. Knowl. Inf. Syst. **64**(10), 2689–2711 (2022). https://doi.org/10.1007/s10115-022-01721-5
23. Nawrocki, P., Osypanka, P., Posluszny, B.: Data-driven adaptive prediction of cloud resource usage. J. Grid Comput. **21**(1), 6 (2023). https://doi.org/10.1007/s10723-022-09641-y
24. Nguyen, T., Tran, N., Nguyen, B.M., et al.: A resource usage prediction system using functional-link and genetic algorithm neural network for multivariate cloud metrics. In: 2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA), pp. 49–56 (2018). https://doi.org/10.1109/SOCA.2018.00014
25. Ouali, C., Dumouchel, P., Gupta, V.: A robust audio fingerprinting method for content-based copy detection. In: 2014 12th International Workshop on Content-Based Multimedia Indexing (CBMI), pp. 1–6 (2014). https://doi.org/10.1109/CBMI.2014.6849814
26. Park, J., Baik, J.: Improving software reliability prediction through multi-criteria based dynamic model selection and combination. J. Syst. Softw. **101**, 236–244 (2015). https://doi.org/10.1016/j.jss.2014.12.029
27. Riganelli, O., Saltarel, P., Tundo, A., et al.: Cloud failure prediction with hierarchical temporal memory: an empirical assessment. In: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 785–790 (2021). https://doi.org/10.1109/ICMLA52953.2021.00130
28. Sergio, A.T., de Lima, T.P., Ludermir, T.B.: Dynamic selection of forecast combiners. Neurocomputing **218**, 37–50 (2016). https://doi.org/10.1016/j.neucom.2016.08.072
29. Shah, S.Y., Patel, D., Vu, L., et al.: Autoai-ts: autoai for time series forecasting. CoRR abs/2102.12347. arXiv:2102.12347 (2021)
30. Sniezynski, B., Nawrocki, P., Wilk, M., et al.: VM reservation plan adaptation using machine learning in cloud computing. J. Grid Comput. **17**(4), 797–812 (2019). https://doi.org/10.1007/s10723-019-09487-x
31. Ullah, F., Bilal, M., Yoon, S.K.: Intelligent time-series forecasting framework for non-linear dynamic workload and resource prediction in cloud. Comput. Netw. **225**, 109653 (2023). https://doi.org/10.1016/j.comnet.2023.109653
32. Xin, R., Liu, H., Chen, P., et al.: Robust and accurate performance anomaly detection and prediction for cloud applications: a novel ensemble learning-based framework. J. Grid Comput. (2023). https://doi.org/10.1186/s13677-022-00383-6