Challenges and Approaches: Forecasting for Autonomic Computing

André Bauer

University of Würzburg, Würzburg, Germany,
`andre.bauer@uni-wuerzburg.de`,
Home page: `https://descartes.tools`

Abstract. In a fast-paced world, software systems require autonomic management. In order to enable accurate and proactive autonomic systems, reliable time series forecasting methods are required. However, most forecasting methods have either a high variance of accuracy and/or time-to-result. To this end, forecasting methods with robust performance are demanded. In order to select the best-suited approach, a fair benchmark of forecasting methods is needed. In this work, the challenges of forecasting in autonomic computing and the associated approaches are outlined.

Keywords: Forecasting, Machine Learning, Benchmark

1 Introduction

Nowadays, software systems are pushed to their limits, on the one hand, by the fast living and changing requirements of their users and, on the other hand, by a huge amount of data that they create or have to process. Although cloud computing is a paradigm that allows facing the increasing scale and complexity of modern software, clouds respectively data centers require skilled IT professionals for configuration and maintaining these systems.

Already in 2001, IBM released the vision of autonomic computing as they observe that programs or environments tend to grow more and more complex. Further, they assume that this complexity exceeds the human capacity [1]. Almost 20 years later, we have billions of interconnected devices (smartphones, IoT, ...), an innumerable amount of services/applications, and approaches such as organic computing [2], self-aware computing [3], and more that build upon this vision.

All these approaches have in common that they are equipped with sensors and interact with their environment. Based on the observations, the systems plan and execute actions either to adjust to the environment or adapt the environment. However, when triggering actions based on observations, the system only reacts. These actions have an inherent delay that may lead to problems.

In nature, animals or humans have the ability, which is commonly called intuition, to ”predict” upcoming events. In order to upgrade the systems with this ability, i.e., allowing the system to be proactive, the systems require accurate and reliable forecasting methods.

In many research areas, forecasting is an established but important discipline that allows predicting the future by examining the past observations. Based on the "No-Free-Lunch Theorem" [4] of 1997, which states that there is no optimization algorithm that is best suited for all scenarios, an analogy can be drawn to the forecasting of time series since the methods try to model the historical data best. So there is no forecasting method that performs best for all time series. In other words, forecasting methods have their benefits and drawbacks depending on the specific use cases respectively time series. Indeed, expert knowledge is required for choosing the best forecasting method. However, expert knowledge is expensive, may take a long time to deliver results, and it cannot be completely automated. In order to overcome this, this work identifies problems and the corresponding 7 challenges of forecasting in the context of autonomic computing. These challenges are grouped in online, offline, and benchmarking challenges. Further, this work proposes approaches to face these challenges, suggests auto-scaling as a use case of a proactive automatic system, and shows some preliminary results.

The remainder of this work is structured as follows: The next section presents briefly foundations regarding time series and forecasting methods. In Section 3, the challenges of forecasting in the context of autonomic computing are stated. Section 4 presents the ideas to face the challenges and suggest an use case for an proactive autonomic system. Preliminary results of tackling one of the challenges are discussed in the Section 5. In Section 6, related work is summarized before the paper is concluded.

2 Time Series Forecasting

In order to better understand the following sections respectively recapitulate common terms, this section introduces foundations from the field of time series and forecasting methods.

2.1 Time Series Foundations

This section gives a short introduction to common terms in the context of forecasting. A univariate time series is a sequence of data points ordered by equidistant time steps. Mathematically, let $y_t \in \mathbb{R}$ be an observation at date t, the univariate time series is defined by

$$Y : \{y_t\}_{t=1}^{T} .$$

Especially in the context of autonomic computing, systems have several sensors. Either the observations of each sensor can be saved as a univariate time series or the observations of correlated sensors can be stored as a multivariate time series. In other words, a time series can also be multivariate and has for each date at least two observations. In the following, the term time series is used for a univariate time series.

Decomposition A further mathematical definition of a time series can be derived by decomposing the time series into its components. Indeed, there are different decomposition methods that lead to diverse components. In this work, STL (Seasonal and Trend decomposition using Loess) [5] is considered for time series decomposition. STL is a commonly used method, where the time series is decomposed in the trend component T, the seasonal component S, and the irregular component I or noise:

$$Y(t) = T(t) \diamond S(t) \diamond I(t).$$

The operation $\diamond$ depends on whether the time series has an additive or multiplicative decomposition. The trend is the long-term movement in time series, i.e., upwards, downwards, or stagnate. Usually, the trend is a monotone function unless extrinsic events trigger a break and cause a change in the direction. The presence of recurring patterns within a regular period in time series is called seasonality. These patterns are caused by climate, customs, or traditional habits such as night and day phases. Figure 1 depicts an exemplary decomposition of a time series.

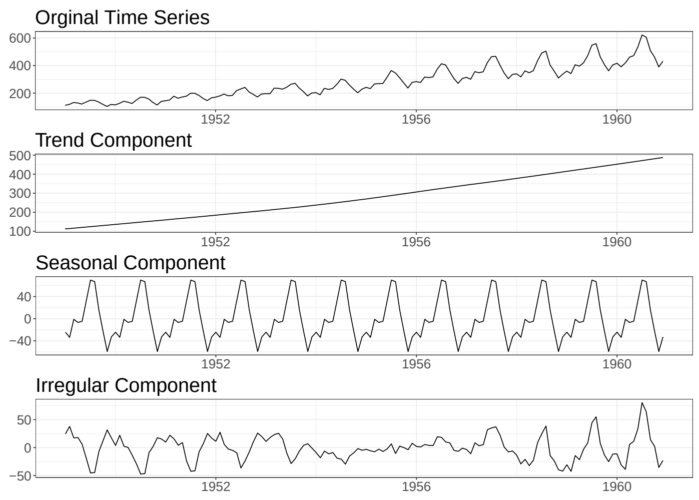

Fig. 1: Examplary time series decomposition.

Box-Cox Transformation As observed data may contain complex patterns, an adjustment respectively simplification of this data can improve the forecasting model. To this end, there exist different methods. A useful approach is the Box-Cox transformation [6]. This transformation depends on the transformation parameter λ. The parameter can be estimated with the method introduced by

Guerrero [7]. The Box-Cox transformation is defined as follows:

$$w_t = \begin{cases} \ln(y_t) & \text{if } \lambda = 0; \\ (y_t^\lambda - 1)/\lambda & \text{otherwise.} \end{cases}$$

Based on this model, the forecast is conducted. Then, the forecast is re-transformed with the same λ:

$$y_t = \begin{cases} \exp(w_t) & \lambda = 0; \\ (\lambda w_t + 1)^{1/\lambda} & \text{otherwise.} \end{cases}$$

Error Measurements In order to evaluate the accuracy of forecasting results, several error measurements can be used. Each method has its use cases, benefits, and drawbacks. Thus, the selection of the measurements has to be done carefully. A common method for error measurement is the mean absolute percentage error:

$$\text{MAPE} = \frac{1}{T}\sum_{t=1}^{T} |\frac{y_t - f_t}{y_t}|.$$

In this equation, f_t is the forecast value, y_t the observation at date t, and T the observed period of time. The MAPE is chosen for reflecting the forecast accuracy due to its measure characteristics: (i) It represents the percentage of average absolute error compared to the original observations. (ii) Also, the MAPE is independent of the scale of the measurement. (iii) Further, due to the absolute error, contrary signed errors do not offset each other. [8]

2.2 Forecasting Methods

As forecasting is an important aspect in many research fields, there exist different methods from different disciplines. In this work, three methods are outlined as they are part of the approach: (i) an artificial neuronal network method from the field of deep learning, (ii) seasonal ARIMA from the field of statistical methods, and (iii) extreme gradient boosting from the field of machine learning.

ANN The Artificial Neuronal Network is a feed-forward neuronal network that is trained with lagged values of a time series. In contrast to other neuronal networks, the feed-forward neural network moves the information only in one direction. That is, there are no loops or cycles in the network. In order to forecast time series, this network consists of one hidden layer. In addition, the number of lags and nodes in the hidden layer are automatically selected. [9]

sARIMA In 1938, H. Wold lays the groundwork for using ARMA models for time series. An ARMA model is a combination of an autoregressive $AR(p)$ model and a moving-average $MA(q)$ model. Here, the order p determines the number of past values and the order q determines the number of past errors. As ARMA models require a stationary time series, ARIMA models overcome this restriction with differencing of the time series. In contrast to ARIMA, sARIMA is capable of modeling seasonal data. To this end, each non-seasonal component of the ARIMA model is extended with its seasonal counterpart. [10]

XGBoost In 2014, a scalable end-to-end tree boosting system called eXtreme Gradient Boosting was released. This method uses tree boosting. That is, a tree ensemble model with additive functions are created. Each function corresponds to an independent tree structure. In terms of time series forecasting, the leaves of regression trees are summed up to predict the output. The method tries to select a model with simple and predictive functions. This is done in an additive, greedily manner that is also known as gradient tree boosting. [11]

3 Challenges

In this section, the challenges, which arise in the field of autonomic computing, are outlined. The following challenges are grouped into three groups: (i) general challenges, (ii) online challenges, and (iii) benchmark challenges. The general challenges occur before applying time series forecasting. The online challenges arise in addition to the general challenges in an online scenario. The benchmark challenges concern themselves with how to compare different forecasting methods.

3.1 General Challenges

Based on its former work [12], D. Wolpert postulated the "No-Free-Lunch Theorem" [4] in 1997. It claims that there is no optimization algorithm that performs best for all scenarios. That is, improving the performance of one aspect normally leads to a degradation in performance for some other aspect. As forecasting methods try to best fit to historical data, parallels can be drawn to optimization problems. In other words, if a forecasting method is tuned for a specific time series, it gets worse for another time series. Thus, there is no forecasting method that outperforms the others for all time series. In order to underline this statement, a measurement over 50 time series is conducted. The forecast error and time-to-result are listed in Table 1. For instance, The forecasting method ANN performs worst according to the accuracy and has a standard deviation of more than 300%. In contrast, sARIMA has the lowest average forecast error, but has the highest average time-to-result and also the highest standard deviation. Based on this example and the theorem, the following challenge can be formulated:

Challenge 1

How to build/find a generic forecasting approach that delivers robust forecasting accuracy?

Here, robust means that the variance in forecasting results should be reduced, not necessarily improving the forecasting accuracy itself.

Table 1: Accuracy and time-to-result comparison.

Method	**MAPE**		**Time-to-Result**	
	Avg [%]	SD [%]	Avg. [s]	SD [s]
ANN	80.98	306.56	31.75	91.88
sARIMA	24.92	41.14	82.32	248.52
XGBoost	49.43	162.39	0.011	0.01

As autonomic systems have multiple sensors, software systems tend to generate a high amount of different and heterogeneous data. This results in multivariate time series. Although there is an extension for ARIMA for handling multivariate time series, the most statistical methods do not support multivariate time series. Thus, machine learning algorithms are used as forecasting tools. Due to the nature of these data, there is a need for data selection. This is also known as feature selection in the context of machine learning. As the selection is a crucial part for the quality, the following challenge can be formulated:

Challenge 2

How to automatically select the important features for forecasting?

Besides the feature selection, also the transformation of the data influence the forecasting accuracy. Hence, transforming the historical data may lead to simpler patterns that usually allows more accurate forecasts [10]. There are different kinds of transformations as calendar adjustments or mathematical transformations. As calendar adjustments require human intervention, only mathematical transformations are suitable in autonomic systems. Thus, the subsequent challenge can be posed:

Challenge 3

How to automatically transform the features for increasing the forecast accuracy?

This challenge is tackled in a preliminary evaluation in Section 5.

3.2 Online Challenges

While forecasting, for example, the visitors of a zoo has no time constraints, the time-to-result in the context of autonomic computing has strict requirements. However, time to result varies heavily for traditional approaches as shown in Figure 2 or in Table 1. The diagram shows the time-to-results distribution of the introduced methods in Section 2.2. Each method forecasts 50 time series and is depicted on the horizontal axis. The vertical axis is in log scale and shows the time-to-result in milliseconds. For instance, the time-to-result of ARIMA ranges from approximately 200 milliseconds to 2,000,000 milliseconds. Similarly, ANN and XGBoost show both a high degree of variance. The table lists forecast error and time-to-result. For instance, sARIMA has a standard deviation of 248.52% for the time-to-result. Therefore, the next challenge can be stated:

Challenge 4

How to build/find a generic forecasting approach that has a reliable time-to-result?

Here, reliable means that the variance in time-to-result should be reduced without degrading the time-to-result itself.

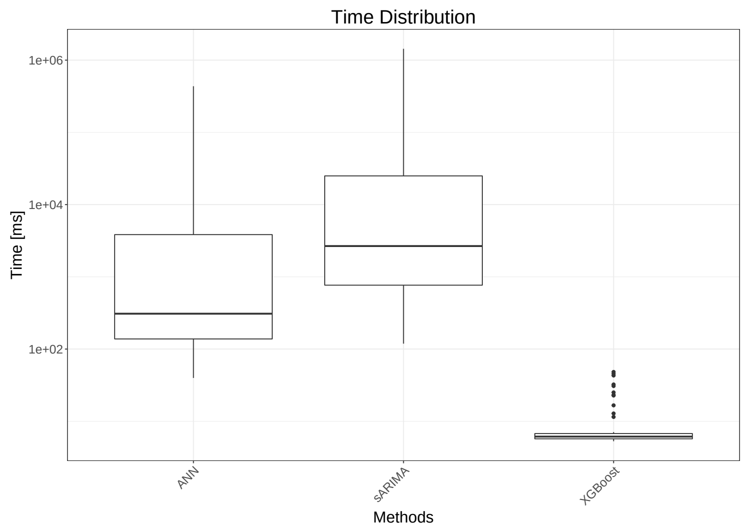

Fig. 2: Examples for Challenge 4.

As autonomic systems may plan actions in advance, the future values have to be forecast. The longer the forecast, the less accurate is the forecast towards the end. Thus, the planned actions may be updated as new data may influence the decision process. In other words, the regular incoming information allows quantifying the accuracy of the forecast. In order to update the forecasts, there are different approaches. One solution is to build a new forecast model with each new observation. Another option is to refine the model with the new data if this is possible for the forecasting method. In contrast to building a new model or updating the model with the incoming data, a further approach is to use the old model until a significant deviation between observation and forecast take place. Indeed, each of this approach has its strengths and weaknesses. That is, there is a trade-off between time-to-result and the accuracy of the forecast. In order to investigate this trade-off, the following challenge can be formulated:

Challenge 5

How to refine/refit the forecast model with incoming data?

3.3 Benchmarking Challenges

As forecasting is an important part of the decision-making process and is used in many fields, there is a broad range of academic work concerning forecasting. However, the degree of quality of the evaluations suffers, on the one hand, due to the lack of commonly used respectively good data sets or, on the other hand, on the methodology. For instance, the proposed forecasting methods (see Section 6) only predict the next value. From a statistical perspective, only the next value is important, but planning in advance requires multiple values. However, there is one work out of the surveyed ones that forecasts up to 15 points. Indeed, this is also a short range in the context of automatic computing, where a fine granularity leads to several data points in a short time span. Further, the surveyed papers compare their approaches on a small data set ranges from 1 to 10 time series. Also, a comparison with other methods either is missing or contains only a few methods. Upon this, the authors only consider the forecast accuracy. By taking these limitations into account, the next challenge can be stated:

Challenge 6

How to compare different forecasting methods in a fair manner?

As stated in Section 2.1, there exists several measures for evaluating the accuracy of a forecast. Although the measure MAPE is used for this work based on its advantages, the selected measure has some characteristics that introduce issues in the context of automatic computing. For example, suppose there are two different time series, each with a forecast. The two forecast values $f_1 = 3$, $f_2 = 3$ are predicted for the observations $y_1 = 2$, $y_2 = 4$. In other words, the first forecast overestimates and the second forecast underestimates the observation. Although both forecast values differ by one, the error measure has two different values: $\text{MAPE}_1 = 0.5$ and $\text{MAPE}_2 = 0.25$. That is, MAPE prefers forecasting methods that underestimate the observations. In practice, underestimation may lead to faults. Thus, additional metrics or new metrics have to be considered while comparing forecasting methods. The resulting challenge is:

Challenge 7

What are suitable/reliable metrics for quantifying the forecast?

4 Approaches and Ideas

This section presents the ideas to face the identified challenges from Section 3. On the one hand, a hybrid forecasting method is envisioned and, on the other hand, a benchmark for forecasting method is planned. Further, a use case showing the importance of forecasting in the context of autonomic computing is outlined.

4.1 Hybrid Forecasting

In order to tackle Challenge 1 and 4, the idea is to design a hybrid machine learning mechanism that combines different forecasting methods. The preliminary version is able to improve the forecast accuracy and to reduce the variance of the results while maintaining a short time-to-result on two time series [13]. The proposed hybrid forecasting approach is called *Telescope* according to the analogy with the vision of far-distanced objects. The mechanism is implemented in R and can be found at GitHub[1].

The current version supports only univariate time series with a seasonal pattern. The core idea of the approach is to leverage additional information from the time series itself and consists of three fundamental steps: (i) Preprocessing of the time series, (ii) building the forecast model, and (iii) forecasting the future behavior of the time series.

In the preprocessing step, the frequency, i.e., the length of a period, of the time series is estimated and anomalies are removed. In this approach, the assumption is that the frequencies match the human behavior (e.g., hourly, daily, weekly, ...). For the model learning, the machine learning algorithm eXtreme Gradient Boosting is chosen due to its success at Kaggle challenges[2]. In order to create features for XGBoost, the time series is decomposed via STL in its components: trend, season, and noise. Further, a second season is determined from the time series. This second frequency allows capturing a better seasonal model as, for example, a time series has a weekly pattern (first season retrieved by STL) and a daily pattern. With the usage of clustering and ANN, this second seasonal pattern is detected. Afterward, the model is trained with both seasonsonal patterns and the trend. The irregular component is ignored since it is hard to predict and so, correlated with a high error rate. For the forecasting, the seasonsal patterns and the trend have to be forecast. As the seasonsonal patterns are per definition recurring, both patterns can be pursued. In contrast, the trend is predicted with ARIMA. Finally, XGBoost combines the ”predicted” components to forecast the time series.

To face Challenge 2 and to enhance Telescope for multivariante time series, an autonomic feature selection process is envisioned. As the current version uses XGBoost as machine learning approach, a further improvement is to dynamically choose the most suitable machine learning approach based on the time series characteristics. Also, the extension to support multivariate time series is

[1] `https://github.com/DescartesResearch/telescope`
[2] `https://github.com/dmlc/xgboost/tree/master/demo`

intended. In order to tackle Challenge 5 and improve Telescope for online usage, different strategies of model refinement will be investigated.

4.2 Forecast Benchmark

Due to the lack of good data sets and the sometimes arbitrary evaluations (see Challenge 6), the idea is to establish a forecast benchmark. On the one hand, the concept is that the benchmark takes control over the evaluation. This leads to a uniform evaluation of forecasting methods. On the other hand, the benchmark provides a broad data set with a high degree of diversity. Further, the data set is split into different use cases, for instance, time series that are related to the finance sector. This allows benchmarking forecasting methods in different fields.

Indeed, there are data sets such as the M3 Competition[3] that contains 3003 time series from different domains. However, most time series have a high degree of similarity and a length below 100 data points. In other words, this data set is, for instance, not suitable for validating forecasting methods in the context of autonomic computing as time series in this domain are generally larger. For instance, when sampling data each second, actions that are planed hourly have to take 3600 data points into account.

Besides the data set, fair competition among the forecasting methods is required. That is, the forecast benchmark has to fulfil the requirements of benchmarking principles. As highlighted in Challenge 7, the current metrics may not be capable of comparing different forecasting methods in a fair manner. Thus, the benchmark has to introduce also a new set of metrics. Besides, the benchmark requires a good reporting system and usability.

4.3 Use Case

Autonomic systems are the solution for problems that are to complex for human beings. In the course of digitization, the Internet usage is fast-paced and has changing requirements. Also, the increasing amount of connected devices has an influence on the usage. In the last years, cloud computing emerged as a computing model that allows fast access to resources and has a high level of scalability.

Due to these benefits and the requirement of the Internet usage, the deployment of autonomic resource management systems arose in cloud computing. In practice such as in AWS (Amazon Web Services), reactive mechanisms are used that trigger releasing respectively provisioning of resources. As the name indicates, these mechanisms react to workload changes and try to adapt the resources to the current workload. However, these approaches are pragmatic and the actions are delayed. The lag can be seen in Figure 3. This diagram shows the scaling behavior of the reactive auto-scaler introduced by Chieu et al. [14]. On the horizontal axis the time in minutes and on the vertical axis the amount of resources are depicted. The blue dashed curve depicts the demanded amount

[3] `https://forecasters.org/resources/time-series-data/m3-competition/`

of resources. The black curve shows the supplied resources by the auto-scaler. Especially when the demand changes, the adaptions of the auto-scaler lags behind.

Thus, the idea is to extend such reactive methods to be proactive respectively improve the forecast accuracy of already proactive auto-scalers in order to achieve a better auto-scaling performance. Here, the performance is measured by metrics proposed by SPEC[4] [15]. In context of the use case, one work was published concerning about the input of auto-scalers [16] and another work introduces a new proactive mechanism with a reactive fallback [17].

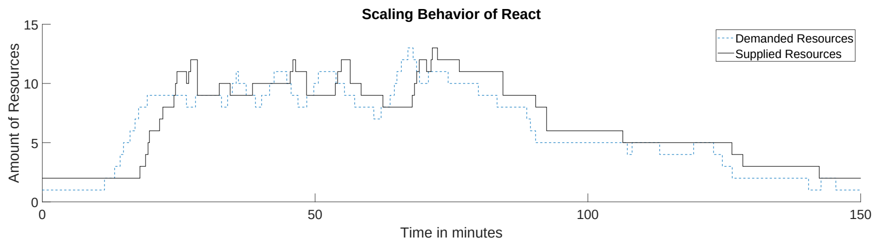

Fig. 3: Example.

5 Preliminary Results: Feature Engineering

In order to investigate the influence of feature transformation and tackle Challenge 3, preliminary experiments are conducted. To this end, the forecast accuracy of ANN and sARIMA are observed while transforming the historical data. The used data set contains the fma package from R. For the experiments four transformations are used: (i) identity function *Id:* $x \mapsto x$, (ii) natural logarithm *Log:* $x \mapsto ln(x)$, (iii) square root *Sqrt:* $x \mapsto \sqrt{x}$, and (iv) box-cox transformation *Box* (see Section 2.1).

Table 2: Feature transformations and the resulting forecast performance.

Rank	**ANN**				**sARIMA**			
	Id	Log	Sqrt	Box	Id	Log	Sqrt	Box
1	34%	32%	7%	31%	31%	36%	10%	29%
2	8%	27%	43%	24%	11%	22%	45%	20%
3	25%	23%	35%	19%	20%	19%	42%	23%
4	33%	18%	14%	25%	37%	23%	04%	28%

[4] Standard Performance Evaluation Corporation

Table 2 shows for both ANN and sARIMA the proportion of the achieved rank while using a certain transformation. For instance, sARIMA with *Log* achieves on 36% of the time series the best forecast compared to the other transformations. In other words, when choosing *Id* for ANN, there is the probability of 34% that this transformation is the best one compared to the other ones. However, in 33% *Id* leads to the worst forecast. For both forecasting methods, *Sqrt* achieves in less than 11% the best forecast. Further, it can be stated that adjusting the data in these experiments (in ANN 64% and sARIMA 69% cases of the time series) increase the forecast accuracy. To sum up, there is no transformation that outperforms the other ones and thus, the selection is crucial.

Based on the information from Table 2, different transformation strategies are investigated. First, each strategy adjusts with a specific transformation the data. Then, the adjusted time series is forecast by both ANN and sARIMA. Afterward, the mean rank of the forecast of ANN and sARIMA is reported. The observed strategies are: (i-iv) using the associated transformation on each time series, (v) choosing a random transformation (*Random*), and (vi) the proposed automatic approach for the transformation selection (*Aut*). The latter approach adjusts each time series with each of the four transformations. Then, for each transformation the forecast model is built. Based on the model accuracy, the best transformation is selected. Afterward, the chosen adjusted time series is forecast. Finally, for each strategy the forecast is re-transformed and the accuracy is calculated.

Table 3: Average performance of different strategies.

	Id	Log	Sqrt	Box	Random	Aut
Avg. Rank	2.60	2.14	2.48	2.44	2.50	2.42

The average ranks over all time series and both forecasting methods of the different approaches are depicted in Table 3. Due to the 4 transformations, the ranks range from 1 to 4. *Random* achieves an average rank of 2.5. That is, when choosing a transformation randomly, the forecast is either the second or third best forecast. The best rank is achieved while transforming the data with *Log*. The automatic selection of the transformation has a rank of 2.6. That is, it is better to choose random selection than this approach.

Table 4 investigates why *Aut* has the worst rank. Therefore, for each transformation, the loss of rank is listed. As the automatically selection takes the transformation for each time series with the best model accuracy rank, the loss of rank reflects the difference between the expectation of having the best forecasting rank and the actual forecast rank. For example, *Log* achieved the best rank for ANN and sARIMA over 44 time series based on model accuracy. When using the logarithmic transformation, however, ANN only achieves the best forecast for 10% of these time series. For all transformations and both methods, more than the half of the selected transformations achieves a rank worse than 1. Hence, the selection of the transformation based on the model accuracy achieves a rank worse than 2.5. These results underline Challenge 3. To this end, further and more intelligent approaches are required for achieving a better forecasting method recommendation.

Table 4: Rank loss after taking the best model transformation.

	ANN				**sARIMA**			
Loss	Id (2)	Log (44)	Sqrt (10)	Box (27)	Normal (0)	Log (44)	Sqrt (11)	Box (28)
0	0%	39%	10%	48%	0%	48%	9%	39%
-1	0%	11%	20%	7%	0%	9%	27%	7%
-2	0%	18%	50%	19%	0%	14%	64%	14%
-3	100%	32%	20%	26%	0%	30%	0%	39%

6 Related Work

This section introduces related work of the fields (i) time series forecasting, (ii) feature engineering, and (iii) and auto-scaling. To the best of my knowledge, there is no forecast benchmark or a generic comparison method between forecasting methods in the literature.

6.1 Time Series Forecasting

In order to minimize the variance of single forecasting mechanisms and to face the ”No-Free-Lunch Theorem”, many hybrid mechanisms have been developed. These approaches can be grouped into three categories. The first and oldest category is the ensemble forecasting. This technique was introduced by Bates and Granger in 1969 [18] and combines the results of at least two forecasting methods. Further works in this field are form R. Clemen [19] and D. Menezes [20]. The second category uses forecast recommendation, where the best forecast method is guessed based on certain time series characteristics. The approaches either use an expert system [21] or machine learning techniques [22]. The last category uses the decomposition of time series to leverage additional information of a given time series. These approaches split the time series into linear and non-linear parts [23,24,25] or decompose the time series in its components (trend, seasonality, and noise) and/or further components [26].

6.2 Feature Engineering

In the scope of this work, two works are of interest. Khurana et al. [27] introduce a framework for feature selection and engineering. To this end, a transformation graph and reinforcement learning are used to explore suitable features and transformations. Further, J. Heaton [28] observed that the machine learning approach influences the features to consider as different methods can learn different kind of relationships.

6.3 Auto-Scaling

Lorido-Botran et al. [29] survey existing auto-scalers and propose a classification of auto-scalers into five groups: (i) threshold-based rules, (ii) queueing theory, (iii) control theory, (iv) reinforcement learning, and (v) time series analysis. Further, Nikravesh et al. propose a kind of expert system to improve the prediction accuracy of auto-scaling [30].

7 Conclusion

Due to the fast-paced and changing requirements, software systems have a high degree of complexity. Thus, managing of some of these systems may exceed the human capacity. One solution is the usage of autonomic systems. In order to enable autonomic computing with proactive actions, reliable and accurate forecasting methods are required. Although forecasting is an important and established part of decision making, in the context of autonomic computing, there are challenges, such as a high variance in accuracy and time-to-result, that have to be faced. Thus, this work identifies 7 challenges (see Section 3) that have to be faced and proposes approaches to tackle them. The goals are, on the one hand, to develop a hybrid forecaster, which has robust performance, and, on the other hand, to establish a forecast benchmark for comparing different forecasting methods in a fair manner. Further, auto-scaling as use case for proactive autonomic computing is introduced. Based on the challenges, preliminary experiments are conducted.

Besides the challenges, proposed solutions, possible extensions, and the preliminary experiments, a lot of work have to be done. Firstly, the evaluation of the proposed hybrid forecasting method have to extended as the preliminary experiments covers two time series. Further, the method has to be enhanced to handle time series without seasonality. As autonomous systems have multiple sensors, an extension for multivariate time series is required. Although the results of the automatic feature transformation is currently not promising, further approaches have to be investigated.

References

1. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. Computer **36**(1) (2003) 41–50
2. Müller-Schloer, C., Schmeck, H., Ungerer, T.: Organic computing - A paradigm shift for complex systems. Springer Science & Business Media (2011)

3. Kounev, S., Kephart, J.O., Milenkoski, A., Zhu, X., eds.: Self-Aware Computing Systems. Springer Verlag, Berlin Heidelberg, Germany (2017)
4. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation **1**(1) (Apr 1997) 67–82
5. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: Stl: A seasonal-trend decomposition procedure based on loess. Journal of Official Statistics **6**(1) (1990) 3–73
6. Box, G.E., Cox, D.R.: An analysis of transformations. Journal of the Royal Statistical Society. Series B (Methodological) (1964) 211–252
7. Guerrero, V.M.: Time-series analysis supported by power transformations. Journal of Forecasting **12**(1) (1993) 37–48
8. Adhikari, R., Agrawal, R.K.: An introductory study on time series modeling and forecasting. CoRR **abs/1302.6613** (2013)
9. Hyndman, R., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E., Yasmeen, F.: forecast: Forecasting functions for time series and linear models. (2018) R package version 8.4.
10. Hyndman, R.J., Athanasopoulos, G.: Forecasting: principles and practice. OTexts, Melbourne, Australia (2014)
11. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: ACM SIGKDD 2016, ACM (2016) 785–794
12. Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural computation **8**(7) (1996) 1341–1390
13. Züfle, M., Bauer, A., Herbst, N., Curtef, V., Kounev, S.: Telescope: A Hybrid Forecast Method for Univariate Time Series. In: Proceedings of the International work-conference on Time Series. (September 2017)
14. Chieu, T., more: Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment. In: IEEE ICEBE 2009, IEEE (2009) 281–286
15. Herbst, N., Bauer, A., Kounev, S., Oikonomou, G., van Eyk, E., Kousiouris, G., Evangelinou, A., Krebs, R., Brecht, T., Abad, C.L., Iosup, A.: Quantifying Cloud Performance and Dependability: Taxonomy, Metric Design, and Emerging Challenges. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (ToMPECS) **3**(4) (August 2018) 19:1–19:36
16. Bauer, A., Grohmann, J., Herbst, N., Kounev, S.: On the Value of Service Demand Estimation for Auto-Scaling. In: 19th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems (MMB 2018), Springer (February 2018)
17. Bauer, A., Herbst, N., Spinner, S., Ali-Eldin, A., Kounev, S.: Chameleon: A Hybrid, Proactive Auto-Scaling Mechanism on a Level-Playing Field. IEEE Transactions on Parallel and Distributed Systems (September 2018) Preprint.
18. Bates, J.M., Granger, C.W.: The combination of forecasts. Journal of the Operational Research Society **20**(4) (1969) 451–468
19. Clemen, R.T.: Combining forecasts: A review and annotated bibliography. International Journal of Forecasting **5**(4) (1989) 559–583
20. De Menezes, L.M., Bunn, D.W., Taylor, J.W.: Review of guidelines for the use of combined forecasts. European Journal of Operational Research **120**(1) (2000) 190–204
21. Collopy, F., Armstrong, J.S.: Rule-based forecasting: Development and validation of an expert systems approach to combining time series extrapolations. Management Science **38**(10) (1992) 1394–1414

22. Wang, X., Smith-Miles, K., Hyndman, R.: Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. Neurocomputing **72**(10 - 12) (2009) 2581 – 2594
23. Zhang, G.P.: Time series forecasting using a hybrid arima and neural network model. Neurocomputing **50** (2003) 159–175
24. Pai, P.F., Lin, C.S.: A hybrid arima and support vector machines model in stock price forecasting. Omega **33**(6) (2005) 497–505
25. Liu, N., Tang, Q., Zhang, J., more: A hybrid forecasting model with parameter optimization for short-term load forecasting of micro-grids. Applied Energy **129** (2014) 336–345
26. Taylor, S.J., Letham, B.: Forecasting at scale. The American Statistician **72**(1) (2018) 37–45
27. Khurana, U., Samulowitz, H., Turaga, D.: Feature engineering for predictive modeling using reinforcement learning. arXiv preprint arXiv:1709.07150 (2017)
28. Heaton, J.: An empirical analysis of feature engineering for predictive modeling. In: SoutheastCon, 2016, IEEE (2016) 1–6
29. Lorido-Botran, T., Miguel-Alonso, J., Lozano, J.A.: A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments. Journal of Grid Computing **12**(4) (2014) 559–592
30. Nikravesh, A.Y., Ajila, S.A., Lung, C.H.: An autonomic prediction suite for cloud resource provisioning. Journal of Cloud Computing **6**(1) (2017) 3