School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Internet: office@ecs.vuw.ac.nz

Preliminary Report for Building a Machine Learning Model to Predict Time Series Cloud Workload

Joel Chu

Supervisor: Hui Ma, Aaron Chen

02/06/2023

Submitted in partial fulfilment of the requirements for
Software Engineering with Honors.

Abstract

This project aims to accurately predict cloud workload using a transformer model. Doing this will allow cloud service providers to accurately determine the amount of resources to allocate at any given time. Thus preventing under and over-provisioning of resources. The data sets Google Cluster Trace will be used as training, testing, and validating data sets. The transformer model will be compared to other time series workload prediction models such as LSTM. This project is successful if the transformer model has a greater prediction accuracy than traditional models.

1 Introduction

1.1 Context

In cloud computing, the user will pay for a service offered by a Cloud Service Provider (CSP). The CSP is responsible for maintaining the quality and reliability of the service they are providing to the user. The user must always be satisfied with the response time, cost, and quality that the CSP is providing. This is known as quality of service (QoS). This will mean making sure that the user is always receiving sufficient resources for all times of the day. This is a challenge for the user and CSPs because, throughout the day when the user requires more resources during peak hours, demand may increase where they will require more resources which have to be allocated [1]. As well as providing QoS, the CSP must also take into account the cost factor. There will be financial consequences to the end user if they allocate too many resources so it is key to find the optimal resources required as well as minimizing costs[2]. CSPs tend to offer a service elasticity service that allows the user to allocate and de-allocate resources based on the fluctuation demand during peak times. This will provide end users with QoS and can improve response times, availability, and throughput to the user [3].

1.2 Motivation

We want to build a model that can make accurate predictions for cloud workflow. The motivations for this project will be listed below.

- In a survey by RightScale, they found that 94 per cent of businesses used some form of cloud [4]. This number is substantial because an improvement in workload prediction could potentially have a positive impact on almost every business. Even a small improvement in prediction accuracy will be a large improvement due to the scale of the cloud.
- The incentive to do this reaches an economic and environmental level. Since cloud computing uses a large number of physical resources to run, if we can reduce wasted resources, it would mean that large data centres can save energy. By accurately predicting cloud workload, it will reduce the impact cloud computing has on the environment by saving resources and energy thus making the use of the cloud more sustainable.
- If cloud workflow accuracy reaches a high enough accuracy, we will be able to avoid under-provisioning resources to users. This would improve the overall user experience as CSPs can constantly maintain QoS for all users. Thus maintaining good performance and availability for all users.
- Since over-provisioning can be reduced, this will reduce operating costs in general. End users will also feel the effects of this by not having to compromise the QoS provided by CSPs while being able to save money on services.

1.3 Solution

1.3.1 Aim

This project aims to build a model that can accurately predict future cloud workloads. The model will be able to analyse and train on historical time series data to make accurate predictions. We aim to achieve greater accuracy than existing workload prediction models. This will be explained later in this report.

1.3.2 Deliverables

By the end of the project, we are expected to have a model that can accurately predict cloud workload. This will be in the form of a machine-learning model that can predict cloud workload traffic. There are no intermediate deliverables as we are only aiming to deliver one model at the end of the project.

1.3.3 Measurable Performance Requirements

The main goal of the project is to be able to build a machine learning model that can predict cloud workflow more accurately than existing models. One of the main downfalls of current cloud workload prediction models is the large training time that comes with building these models. We want to use a model that is more efficient in training but can also achieve similar if not better results than existing models. Prominent methods include ARIMA and LSTM as will be discussed further below [5]. The downfall of these models is the training time due to them being sequence-based neural networks. Using attention will allow parallelization when training decreasing computational time [6]. We will measure the training time of existing models in milliseconds and we should see that the new model has a decreased computation time. We will also compare the accuracy of the existing model to the one that we will implement. We expect our model to be more accurate hence having a smaller prediction error. This will be explained later in the report. The problem will be solved if we can improve the computation time and prediction accuracy compared to existing models.

1.3.4 Environmental and Sustainability

To address the sustainability of the proposed solution, one of the metrics that we use to measure the success of the model will be the computation time. We want to achieve a smaller computation time than that of existing models. Doing this will mean a smaller cost when training the model since it is more efficient. The decreased training cost will mean that we will be using fewer computing resources compared to previous models. Another way that we are going to address the issue of sustainability is to write code as efficiently as possible. While we are developing the solution we will try to use energy that is sustainably sourced. Since we are in New Zealand it would be appropriate to assess the sustainability of the resources and energy that we use in the development phase. According to the Work Energy Council, New Zealand is ranked 8th in the world for energy sustainability [7]. This ranking is derived from energy security, energy equity and environmental sustainability. Since New Zealand ranks so highly in energy sustainability, it would be appropriate to use electricity from New Zealand to develop the product. Developing the product using New Zealand energy will ensure that we are causing minimal harm to the environment. As for future use cases of the product, if this was in another country, we must also consider where the model is being used to ensure the resources being used to run the model are being generated sustainably.

2 Background Research

In the past, methods to dynamically allocate resources have been based on a reactive approach which will allocate according to the current demand [8]. The problem with allocating reactively is that there will be moments where demand is greater than the current resources allocated, thus not providing QoS to users. A better method would be to predict the future workload and then allocate according to the future demand. For a CSP to be able to predict

the resources required for the required workload, they will need an appropriate model to make this prediction. They can then allocate and de-allocate resources based on the prediction. In the past, models have used time series analysis to make this prediction[1][9]. These models work by analysing past workloads and identifying patterns to make future workload predictions. Long short-term memory (LSTM) and autoregressive integrated moving average (ARIMA) are models that have previously been used to make time-series predictions. Qihang Ma used the two models for stock price prediction [9] as well as Calheiros et al who used an ARIMA model for dynamic resource provisioning predictions[1]. The results of these studies and others will be discussed in the following sections.

2.1 Long Short Term Memory (LSTM)

LSTM is a time series prediction model that is a kind of recurrent neural network (RNN). LSTM works by remembering and forgetting long and short-term memory. The input will determine how much of each long and short-term input is predicted[9]. An advantage of LSTM is that it can accurately make term predictions by analyzing short-term and long-term data whereas other RNN-based models will only be able to make accurate predictions based on short-term data[3].

What are the disadvantages?

- LSTM models require a lot of input data to make accurate predictions. When there are not many training points, the LSTM will not make accurate predictions as seen by Jiechao [10]

- As noted in the study "Attention is all you need", LSTM as well as other RNNs, rely on long sequences of computation which can not be done in parallel[6]. This will mean that computation time will be very large due to these large sequences.

2.2 Autoregressive Integrated Moving Average (ARIMA)

ARIMA is a model that is used to generate short-term forecasts on time series data. ARIMA uses previous data points to make forecasting predictions [11]. A trait of ARIMA is that it does not need to assume any underlying principles or model equations for the context of the data. This means it will not need to assume any basic economic terms for stock market prediction because the model should be able to learn the principles through training on the data [9]. This means the model can be used for different time-series-based data sets but can still be used effectively. The downside to this is that these basic principles may mean that the training process does not allow ARIMA to pick up these underlying principles which may make for inaccurate predictions.

What are the disadvantages?

- ARIMA works by making a linear prediction from the input variables and error terms. This linear model will work well for short-term predictions but may not perform as well for more complex non-linear problems [9]. Since we do not know the relationship of the data set we are using, we will not know if the linear prediction of the ARIMA model can accurately forecast jobs in cloud workloads.

- In the study by Calheiros, they found that ARIMA has a very large computation time when the input data set is large [1]. The data set we will use for training will be large which could be a downfall for ARIMA.

The data set that we choose to use will be large which means that LSTM will have an advantage over ARIMA due to the LSTM being better at handling larger data sets. Looking at the comparison from the study done by Gupta, we can see that LSTM consistently outperforms ARIMA when predicting cloud workload. A Bidirectional LSTM will almost have half the error that ARIMA has, with ARIMA having an RMSE of 0.0159 at 60 steps and bi-directional LSTM having an RSME of 0.0095 at 60 steps. This means that given the particular context of cloud prediction, LSTM will have greater accuracy than ARIMA.

3 Proposed Solution

In this section, we present the progress of the project so far. Mainly, we have performed data preparation, implemented LSTM, and performed an initial evaluation on a benchmark data set.

3.1 Data Preprocessing

When we first looked at the data, we knew that we would have to do some data preprocessing to address the issues of raw data. These are the preprocessing steps that we used to clean the data before we could use it.

- **Remove missing data:** We noticed that there were some missing data points. These tended to be cycles per instruction (CPI) and mean memory access per instruction (MAI) values that were not recorded. Since we have so much data, we removed the missing data as it would not make a negligible difference in the outcome of the model.
- **Remove outliers:** As stated in the documentation of the data set, some very small CPI and MAI measurements were inaccurate. The documentation suggested filtering out excessively large and small CPI and MAI measurements.
- **Remove unrelated columns:** There are columns such as machine id, job id, task index, sample portion and aggregation type which were not related to the measured data. As stated in Gupta's article which is the baseline model, they only used a subset of the available columns [5]. We removed the unrelated column as they are redundant in hopes to increase the performance of the model.
- **Data Sampling (Binning):** When looking at the data, we could see that measurements were taken every second. Since there are 7 days of data, this would make for a lot of data. There are thousands of measurements taken every second. This means that for each minute that goes by, we would observe tens of thousands of data points. We used data sampling to first group the measurements by the second. This largely reduced the amount of data that we worked with. Gupta and Shen's article, both mentioned sampling the data further and aggregating the data [12] [5]. Gupta has grouped the data into 10-second intervals while Shen has grouped the data into 30-minute intervals. After looking at the data in 10-second intervals, the data still had an excessive amount of noise. Therefore we used intervals every 30 seconds. This removed the bulk of the noise that we saw before.

3.2 LISTM for Workload Prediction

In this project we first adopt the model proposed by Gupta in [5], to predict cloud workload as the baseline model. Due to Gupta's implementation not specifying the parameters that

they used, we used parameters from another article. We have developed the LSTM using the settings found in the study by Shen [12], as they presented some parameters that we can use for our model. In Gupta's article [5], their goal was to predict cloud workloads using a bidirectional LSTM network. They use the Google Cluster Trace data set for training and testing to predict the CPU sample usage [13]. Since this is a multivariate problem, an array of features were used to predict CPU sample usage. These include features such as mean CPU, maximum memory, mean disk, maximum CPU, maximum disk, CPI, MAI and others. The article from Gupta also stated that they used 7 days of samples which equates to 60480 for training data and a variety of test data instances. This ranged from 60 (10 minutes) to 120 (20 minutes) and 180 (30 minutes). This also suggests that Gupta used 10-second intervals of data. [5]. Using MSE(mean squared error) as the loss function, Gupta was able to achieve an accuracy of 0.0095, 0.0184, 0.0255 for 60 steps, 120 steps and 180 steps respectively[5]. The goal is to mimic the results and model developed by Gupta as closely as possible to use as a baseline model to compare against the Transformer model. Since the report never specified any parameters, we had to use some parameters from Hengheng's study where they specified some parameters that were used[12]. Some important parameters were the batch size of 128, 90 epochs and dropout rate of 0.01 [12]. We will use these parameters when developing the baseline LSTM model as we are not told the exact parameters used by Gupta.

RMSE results from Gupta [5]		
60 steps	120 steps	180 steps
0.0095	0.0184	0.0255

These errors are the results of the RSME of the Bidirectional LSTM implemented by Gupta. Due to some implementation details being missing from the report, we will not be able to obtain the exact results observed by Gupta although we should see similar results. This could be a result of differences in data cleaning and parameters used.

3.3 Transformer Model

The transformer model was first proposed in 2017 to address the shortcomings of recurrent neural networks, long short-term memory and gated recurrent neural networks [6]. The transformer model works around the idea of improving existing recurrent models that have a high computation cost. Recurrent models generate sequences to hidden states which makes parallelization impossible due to the sequential nature of the model[14]. When parallelization is not available, training recurrent models can get very expensive and time-consuming. Although certain factorizations and improvements have been made to previous models, recurrent networks would ultimately still be sequence-based which these improvements could never address [15]. Although transformers are mainly known for their use in NLP, speech processing and computer vision. The transformer has seen large success in these areas but not much work has been done in time series data prediction. Transformers use an attention mechanism which allows them to capture long and short-term patterns regardless of the distance. This is something that LSTM struggles to capture especially with historical long-term dependencies**li2019enhancing**. The ability for transformers to capture patterns regardless of the distance makes it suitable for predicting cloud workloads. At the time of writing, transformers have not been implemented much for time series forecasting and have never been used for cloud workload forecasting**li2019enhancing**[16].

What are the advantages?

- Transformers allow the use of parallelization due to the use of attention and breaking up the need for a sequence-based neural network

- Attention allows the model to capture long and short-term patterns regardless of which LSTM struggles with.

The transformer model works by encoding the input vector to an output vector. The decoder will then take this vector and generate an output vector for the model, each time, using the output from before to influence each value in the vector[6]. Transformers use attention which is different to regular recurrent networks.

3.3.1 Attention

The attention function is what separates transformer models from other models. This gives it the weighted sums of each output vector. The output is made up of key-value pairs for each output. The output is the weighted sum of all values which is how each output has able to relate to each output. This attention builds the relationship for all output values. Using attention will speed up computation time by splitting up sequential operations. According to the article "Attention is all you need" [6], recurrent neural networks have a sequential operation of O(n) with a maximum path length of O(n). Self-attention on the other hand has a sequential operation of

O(n) and a maximum path length of O(1). This difference will mean that computation can be done paralleled when using the transformer model. Scaled dot-product attention and multi-head attention are also components that are used by a transformer model.

- Scaled dot-product attention is used on all the keys, It will then go through a softmax function to output the weights of the values. The dot-product will be scaled by dividing by the square root of the query. This will prevent large values from having too great of an impact on the output.

- Multi-head attention is when single attention is projected linearly in the value, key and query dimension. Doing this will allow the information to take into account different dimensions which can not be done with single-attention.

We will develop a transformer model for workload prediction as it addresses the speed and accuracy weaknesses of the baseline model LSTM. We aim to have better accuracy and faster computation speeds than the baseline model.

4 Evaluation

In this section we will evaluate the metrics that we will use to measure the performance, the data set we are going to use on the model as well as an evaluation of the development strategy that we are going to use when building this model.

4.1 Metrics

We will use a variety of metrics and measures to measure the performance of both of these models. The transformer model aims to obtain a higher prediction accuracy than the LSTM model. We will be using the metrics mean absolute error (MAE), root mean squared error (RMSE) and mean squared error (MSE) to measure the performance of the two models [14]. These metrics were used to evaluate the performance of the LSTM trained by Mahendra so we will use the same metrics to measure the accuracy of the transformer model [14]. We will also be measuring how long the computation takes as the prediction will not be effective if it takes too long to output regardless of the accuracy. In the study done by Gupta[5], we

can see that they only used RMSE as a measure of accuracy, the problem with RMSE is that is sensitive to outliers. Although we will do some data cleaning in the preprocessing stage, we cannot guarantee that the data set will be free of outliers. To work around this, we will also use the measures MAE, RMSE and MSE to give us a better picture of the accuracy of the models involved. These three metrics were used by Calheiros in their study[1].

4.2 Data set

After looking at some articles, we noticed that the majority of articles tend to use the Google Cluster Trace data set[1][5] 0[12]. There was the option to use either 2019 or 2011 data [17] [13]. Most articles used the 2011 data set as the 2019 data set was considerably new so not many relevant studies used the 2019 data set. Compared to the Alibaba data set, the method to retrieve the data was much more complicated. As the Google data set was stored on their cloud container, we had to manually download the data using the Google command line. This took around a full day to download the whole data set.

4.3 Development Strategy

For this project, we will use the Agile software development methodology. Agile allows us to work in iterations where we will spend small increments to work on a smaller specific goal [18]. This makes it easier to visualise progress compared to traditional approaches. This works well with how the ENGR489 course is set out where we are expected to meet with the client at least once a week. We will have a set day where we will meet with the client and discuss what we have done in the previous sprint. During this time we also do a sprint plan to plan what to expect the next time we need with the client. This in itself is Agile and the clients(supervisors) will be able to see the progress we have made in the last sprint while also having the opportunity to give us incremental feedback.

We will be using Visual Studio Code as the development IDE alongside Gitlab for version control. The version control will allow us to track the work we have been doing in increments. This allows us to see have done in the last sprint. GitLab has tools such as burn-down charts that allow us to visualize the progress we make as well as the number of commits for each weekly sprint.

We have decided to use Python as the chosen development language because Python has libraries that can be imported which can assist with building the models and data cleaning.

5 Development Progress

In the following section, we will outline the developments that have been made so far in the project. This will include the effects of data preprocessing as well as the results of the implementation of the LSTM model.

5.1 Data Preprocessing

From Figure 1, we can see the difference that the data preparation made on the data. The graphs are a subset of 1/200 of the data set that was used for the actual prediction below. The subset was used to show the difference in the data. On the left, we have the subset of the raw data. The data appears to be very noisy and there is no noticeable trend in the data apart from occasional dips in the CPU usage. The data points seem to have no recurring pattern with some times recording extremely small CPU usage while other times recorded much larger CPU usage. This could be due to how the raw data was collected. There are

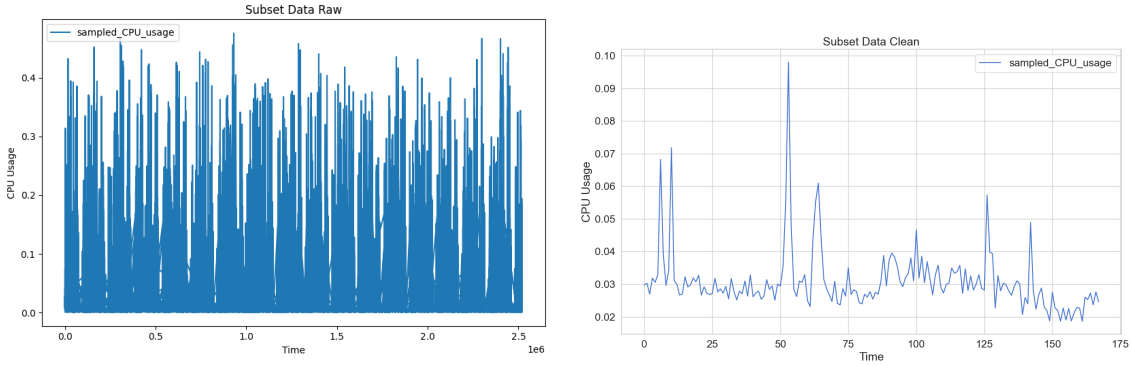


Figure 1: Graph of Subset of Data Raw and Clean

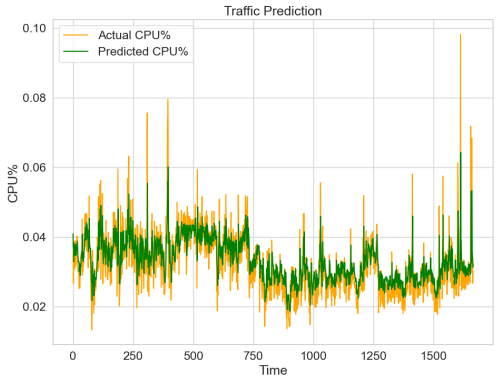
multiple data points for each given time. The intervals of the raw data were every second. This means that for the small subset shown below, there would be approximately 2500000 instances for a time frame of around 6 minutes. Missing and incorrectly measured data could also be responsible for the small CPU usage that we can see in the raw data.

After cleaning the data using the methods remove missing data, remove outliers, remove unrelated columns and data sampling, we were able to remove most of the noise that we saw before. We can see that in the cleaned data subset on the right, there is a noticeable trend in the data where there aren't any random spikes that we saw in the raw data. The use of binning also helped as we look at data from specific intervals, so instead of looking at the 2500000 instances in the raw data, we are now looking at 170 instances. This means that the amount of data is much more manageable and less noisy. The binning of data also allowed us to smooth out the effects of outliers and unusual data so we can take the aggregate of the data for each 30-second time interval.

5.2 Results so far

After implementing a bidirectional LSTM with the appropriate parameters, we were able to achieve the results that we can see in Figure 2. With the orange line being the actual CPU and the predicted CPU being the green line. We can see the predicted CPU closely follows the trend of the actual CPU. Visually, this is a promising result as it tells us that we can predict the pattern and trends in the cloud workflow. Although it is not perfect, we have achieved a high accuracy.

Figure 2: Graph of Workload Prediction using LSTM adapted from Gupta [5]



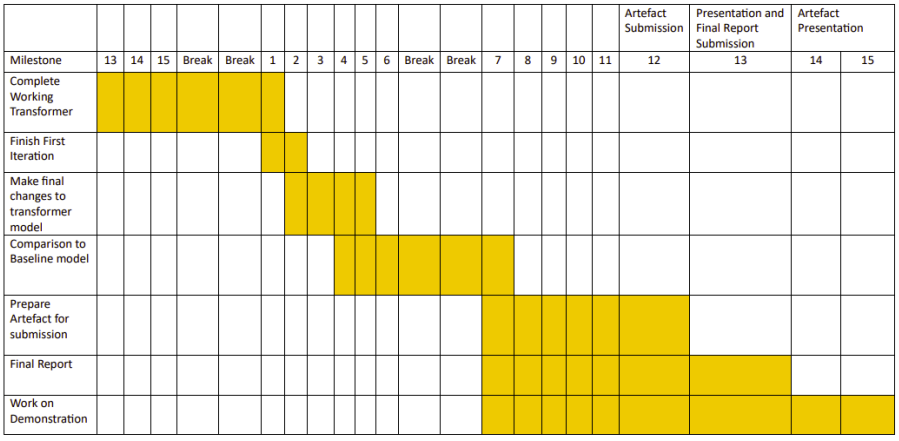
Results from LSTM adapted from Gupta [5]		
MSE	MAE	RMSE
0.0009	0.0212	0.0009

Looking at the results from the LSTM using the metrics, we can see that the MSE and RMSE errors are almost the same while the MAE is much larger with an error of 0.212. The larger MAE can be explained by the large differences between the actual and predicted CPU values at around time units 250 and 1750. These differences all add up to give a large MAE since it does not account for directional differences and finds the total of all the errors without squaring them.

6 Project Progression

In developing the initial GANTT, we were expected to work on the Transformer model before the baseline model. After doing some more analysis, it made more sense to work on the baseline model before starting on the transformer model. This is because there was already existing implementation for the baseline model which would make it easier to familiarize us with the data set. As we are working on the baseline model first, we will work on the transformer model later. The baseline model is almost complete but the transformer model that is the main deliverable is still yet to be implemented.

Figure 3: Table of GANTT Chart of Remainder of Course



As shown in Figure 3, we can see that we have allocated the next eight weeks for developing the Transformer model. We have allocated a lot of time to this because we know that there will be unknown complexities when working on a model that is still reasonably new. Since the ENGR489 course spans across the whole year, we are also expected to work on it during the breaks which leaves us with the next 10 weeks to develop the transformer model as well as 5 weeks after that to make the comparison between the baseline model and the transformer model. We may also need to make more changes to the baseline and transformer model at this stage. After the last break, we will be working on the deliverables which include the artefact submission, presentation and final report submission and the artefact presentation which has an unknown date.

7 Future Plan

We have the following plan for the remainder of the project. We first need to develop the transformer model to make a comparison to the LSTM model. This model is the most important as it will address the problem that we are trying to solve. If time permits, we will also develop an ARIMA model to add to the comparison as well as test the models on a different data set such as the Alibaba Cluster data set.

7.1 Further Implementation

We are still yet to implement the transformer model. In the next few weeks, we will be focusing on developing a transformer model to predict cloud workload. If we have more time, we will develop another existing model such as ARIMA to compare the results of the transformer model to other models.

7.2 Test on More data sets

In the future, we can look at using the Alibaba Cluster data set to see if we can achieve the same results using another data set. Doing this will ensure that there is no biasedness towards the Google Cluster data set. Using another data set will also assure us that the model solves the problem as a whole and can be adapted to different data sets which addresses the problem of predicting cloud workload in general.

8 Request for Feedback

We do not have any specific feedback requests but would appreciate any feedback that the examiners can give.

9 Bibliography

References

[1] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using arima model and its impact on cloud applications' qos," *IEEE transactions on cloud computing*, vol. 3, no. 4, pp. 449–458, 2014. DOI: 10.1109/TCC.2014.2350475.

[2] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang, "Quality-of-service in cloud computing: Modeling techniques and their applications," *Journal of Internet Services and Applications*, vol. 5, no. 1, pp. 1–17, 2014.

[3] D. K. Yadav, M. Pratap, P. Nisha, and D. Kumar, "Workload prediction over cloud server using time series data," in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, IEEE, vol. 11, 2021, pp. 267–272. DOI: 10.1109/Confluence51648.2021.9377032.

[4] ZIPPA, "Rightscale 2019 state of the cloud report from flexera," 2019. [Online]. Available: https://resources.flexera.com/web/media/documents/rightscale-2019-state-of-the-cloud-report-from-flexera.pdf (visited on 03/30/2023).

[5] G. Shaifu and D. D. Aroor, "Resource usage prediction of cloud workloads using deep bidirectional long short term memory networks," in *2017 IEEE international conference on advanced networks and telecommunications systems (ANTS)*, IEEE, 2017, pp. 1–6. DOI: 10.1109/ANTS.2017.8384098.

[6] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017. DOI: 1706.03762.

[7] W. E. Council, "In partnership with oliver wyman trilemma index 2022," 2022. [Online]. Available: https://www.worldenergy.org/assets/downloads/World_Energy_Trilemma_Index_2022.pdf?v=1669839605 (visited on 05/29/2023).

[8] Z. Qian and A. Gagan, "Resource provisioning with budget constraints for adaptive applications in cloud environments," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, vol. 5, 2010, pp. 304–307. DOI: 10.1109/TSC.2011.61.

[9] M. Qihang, "Comparison of arima, ann and lstm for stock price prediction," in *E3S Web of Conferences*, EDP Sciences, vol. 218, 2020, pp. 1–26. DOI: 10.1051/e3sconf/202021801026.

[10] G. Jiechao, W. Haoyu, and S. Haiying, "Machine learning based workload prediction in cloud computing," in *2020 29th international conference on computer communications and networks (ICCCN)*, IEEE, vol. 9, 2020, pp. 1–9. DOI: 10.1109/ICCCN49398.2020.9209730.

[11] Meyler, Aidan, Kenny, Geoff, Quinn, and Terry, "Forecasting irish inflation using arima models," vol. 1998, no. 3, pp. 1–48, 1998.

[12] H. Shen and X. Hong, *Host load prediction with bi-directional long short-term memory in cloud computing*, 2020. DOI: 10.48550/arXiv.2007.15582. arXiv: 2007.15582 [eess.SP].

[13] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: Format + schema," 2011. [Online]. Available: https://drive.google.com/file/d/0B5g07T_gRDg9Z0lsSTEtTWtpOW8/view?resourcekey=0-cozD56gA4fUDdrkHnLJSrQ (visited on 05/26/2023).

[14] M. P. Yadav, N. Pal, and D. K. Yadav, "Workload prediction over cloud server using time series data," in *2021 11th International Conference on Cloud Computing, Data Science Engineering*, vol. 11, 2021, pp. 267–272. DOI: 10.1109/Confluence51648.2021.9377032.

[15] K. Oleksii and G. Boris, *Factorization tricks for lstm networks*, 2017. DOI: 10.48550/arXiv.1703.10722.

[16] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=vSVLM2j9eie.

[17] C. Reiss, J. Wilkes, N. Deng, M. Tirmazi, and M. E. Haque, "Google cluster-usage traces v3," 2019. [Online]. Available: https://drive.google.com/file/d/10r6cnJ5cJ89fPWCgj7j4LtLview (visited on 05/30/2023).

[18] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, *Agile software development methods: Review and analysis*, 2017. DOI: 10.48550/arXiv.1709.08439. arXiv: 1709.08439 [cs.SE].