A Knowledge Distillation Approach with Time Series Prediction and Predictive Elastic Telescopic Algorithm in Computing Force Network

Shizhan Lan[1,2], Zhenyu Wang[2], Siyuan Song[2] and Yuxuan Long[2,*]
[1]China Mobile Guxangxi Branch Co., Ltd, Nanning, China
[2]South China University of Technology, Guangzhou, China

Keywords: Computing Force Network, Elastic Scaling, Knowledge Distillation.

Abstract: Computing Force Network (CFN) is a new type of information infrastructure that allocates computing, storage and network resources on demand and flexibly schedules computing, storage and network resources among cloud, edge and terminal according to service requirements. Accurate matching between user resource demand and computing resource is a key problem in CFN. Elastic resource scaling enables elastic scheduling of computing resources based on service requirements to meet real-time service requirements and improve computing utilization. Resource elastic scaling based on prediction is a common scaling strategy. We first proposed a new knowledge distillation method, and then used the relatively advanced time series prediction algorithm Informer as the basic model to propose a time series prediction model KD-Informer based on knowledge distillation. Finally, we proposed a predictive elastic telescopic algorithm (P-HPA). We verified through experiments that our proposed knowledge distillation method improved by about 10% compared with the existing knowledge distillation algorithm, KD-Informer prediction model improved the prediction accuracy and reduced the model memory occupation compared with the existing model, and P-HPA improved the QoS compared with the default HPA.

1 INTRODUCTION

In recent years, the traditional network architecture has evolved from large-scale cloud data centers to distributed multi-access edge computing (MEC) servers scattered across different locations and mobile intelligent devices, forming a three-layers network architecture consisting of cloud, edge, and end (Kong, 2022)-(Garg, 2021). Therefore, how to integrate ubiquitous and heterogeneous computing resources with the network to achieve accurate matching of user needs and computing resources has become an important research direction for the future development of networks. In the opportunity of the integration of computing power and network development, European Telecommunication Standards Institute (ETSI) have proposed the concept of Computing Force Network (CFN).

CFN is a new type of information infrastructure that flexibly allocates and schedules computing, storage and network resources according to business needs among cloud, edge, and end (Zhou, 2023). there are many business scenarios with high elasticity demands for resources. Although advanced resource scheduling strategies and mature microservice architectures can be used to achieve the matching of resource demand and supply, existing static resource scheduling mechanisms and microservice scheduling mechanisms cannot meet the dynamic and precise matching of business needs and resource supply for performance burst scenarios.

Resource elasticity scaling is a technology that flexibly and dynamically adjusts resource supply based on real-time resource demands of users(Toka L, 2020), which can dynamically adapt to user demand. When user resource demand decreases, resource allocation is reduced to lower enterprise costs and increase cluster resource utilization. According to the type of elasticity scaling, elasticity scaling can be mainly divided into vertical scaling and horizontal scaling. Vertical scaling is mostly used for small-scale applications, which do not require any modifications to user applications and directly change the resource quota of the server where the user application is located. Horizontal scaling is a more common scaling method, which mainly involves increasing the number of servers rather than changing the resource quota of a single server when the business peak increases. The management and scheduling of resources in the

Lan, S., Wang, Z., Song, S. and Long, Y.
A Knowledge Distillation Approach with Time Series Prediction and Predictive Elastic Telescopic Algorithm in Computing Force Network.
DOI: 10.5220/0012273200003807
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 2nd International Seminar on Artificial Intelligence, Networking and Information Technology (ANIT 2023)*, pages 26-33
ISBN: 978-989-758-677-4
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

computing network mainly rely on the Kubernetes resource orchestration platform. The open-source version of Kubernetes provides HPA and VPA policies. Among them, HPA adopts a polling mechanism to obtain the CPU utilization rate of each Pod from the metrics server, then calculates the average resource utilization rate of the cluster, compares it with the user-defined CPU resource utilization threshold during the decision-making phase, and finally calculates the expected number of Pod replicas. To some extent, HPA solves some elasticity requirements in the computing network. However, the open-source version of HPA is a reactive scaling strategy, which passively adjusts resource allocation based on changes in application load, resulting in a certain degree of lag and increasing the SLAs violation rate (Shin S H, 2023)-(Paulo Pereira, 2019).

The challenges of CFN is presented as follows. 1) Scaling decisions are typically based solely on the observations from the current scaling period, primarily focusing on average CPU usage. However, considering a broader historical perspective could provide a more accurate understanding of what lies ahead.2) The default capacity expansion mechanism is lagging behind. When user resource requirements change rapidly, the default capacity expansion mechanism reduces user QoS. 3) The existing prediction algorithms have some problems, such as insufficient prediction accuracy and large memory consumption. Moreover, we propose a predictive HPA mechanism, on the one hand to monitor the traffic, on the other hand to achieve early expansion before load burst. The contributions of this paper are summarized as follows.

- We proposed a new knowledge distillation method and it is experimentally verified that the proposed knowledge distillation method improves the model accuracy compared to existing knowledge distillation methods.
- Based on the knowledge distillation method and the existing time series prediction model Informer, a new time series prediction model KD-Informer is proposed in this article. It is experimentally verified that the proposed model improves the prediction accuracy compared to Informer, Reformer, and other time series models.
- We propose P-HPA based on the above time series prediction model. Experiment results show that the P-HPA improves the scaling accuracy and reduces the SLAs violation rate compared to the default HPA.

2 RELATED WORK

Elastic resource scaling is not a unique concept to the computing power network, which has always been a key factor to consider in cloud computing. We have conducted research and summarized the current state of research on resource elasticity scaling, and Classified it as follows.

Threshold-Based: (Yahya Al-Dhuraibi, 2017) and (Gourav Rattihalli, 2019) improved vertical elasticity in lightweight virtualization technology cloud systems using threshold-based scaling rules. (Hamzeh Khazaei, 2017) proposed Elastic Docker, an autonomous solution based on IBM's MAPE-K principle, which enables autonomous vertical elasticity for Docker containers. In (Zhicheng Cai, 2022), the authors introduced an automatic scaling system based on resource utilization, enhancing Kubernetes' VPA and dynamically adjusting container allocation in Kubernetes clusters. Both papers explore container migration and investigate vertical scaling possibilities, while our proposed solution focuses on enhancing horizontal autoscalers. (Toka, 2020) demonstrated the architecture and initial implementation of Elascale, which provides automatic scalability and monitoring as a service for any cloud software system. Elascale makes scaling decisions based on a tunable linear combination of CPU, memory, and network utilization.

Forecast-Based: Short-term demand forecasting has been extensively studied in various domains (Kader, 2022)-(Punia, 2020). In the energy sector, accurate prediction of electricity generation from wind turbines is crucial. Li et al. (H. Arabnejad, 2017) proposed a four-input neural network that outperformed traditional single-parameter methods. For predicting electricity market prices, Catalao et al. introduced a three-layer feedforward neural network approach, which proved to be superior to the previously proposed autoregressive integrated moving average (ARIMA) methods in terms of time efficiency and ease of implementation. Inspired by Tan's work (Horovitz, 2018), who designed a predictive HPA using the AdaBoost-LSTM algorithm with improved QoS, we adopt a prediction-based approach in our system. We aim to select the most suitable scaling method by leveraging prediction techniques.

Reinforcement Learning-Based: Arabnejad et al. (Sherstinsky, 2020) proposed a resource allocation approach for virtual machines (VMs) by combining Q-learning and the SARSA algorithm with an adaptive fuzzy logic controller. They focused on dynamic resource allocation. Similarly, Horovitz et al. (Vaswani A,2017) introduced a threshold-based horizontal container autoscaling solution that utilized Q-learning to adjust scaling thresholds.

3 SOLUTIONS

In this section, we propose new load prediction algorithm and prediction HPA algorithm (P-HPA). We will select a basic model from several time series prediction models, and a new knowledge distillation method is used to improve the above model.

3.1 Time Series Prediction Model

LSTM: LSTM(Forecasting, 2021) is a common time series prediction model. There are three kinds of LSTM gates: forget gate, input gate, and output gate. A gate is a structure that enables information to pass through selectively. LSTM selects to update or delete information using the gate structure, and transmits useful information to the latest decision unit through cell state for reference. Although this design method of LSTM solves the problem that RNN is easy to forget long-term historical information, LSTM can only solve sequences of 100 orders of magnitude, and the performance of LSTM is not good for sequences of longer orders of magnitude. Moreover, since each unit of LSTM has four fully connected layers, LSTM can be very time consuming if the network is very deep and the sequence length is very long.

Transformer: Transformer (Zhao, 2022) is a deep learning model based on self-attention. Compared with LSTM model, Transformer can pay more attention to historical information and improve the prediction accuracy. However, for long time series, the calculation time and memory consumption are huge. When the original Transformer model deals with the sequence length of N time series, Both time and space complexity are $O(N^2)$.

Informer: Informer(Kang, 2022) is an improved model based on Transformer model, mainly to solve the problem of long time sequence prediction. Informer uses ProbSparse self-attention mechanism and distillation mechanism to shorten the training time of Transformer model and improve the predictive performance. In this paper, the Informer model will be used as the basic model, and we used knowledge distillation method to improve the prediction performance of the model. And we call the final model KD-Informer. Next we will introduce the process of knowledge distillation.

3.2 Knowledge Distillation

Knowledge distillation refers to the process of "distilling" the knowledge learned by a complex and powerful large model and transferring it to a smaller model with a relatively simpler structure. In knowledge distillation, the large model with high complexity and strong learning ability is called the teacher model, while the smaller model generated by the large model with a simple structure and fewer parameters is called the student model. The most common application of knowledge distillation is to enable the student model with fewer parameters and a smaller structure to learn the knowledge of the teacher model with a more complex structure and higher accuracy, and deploy the student model on devices with limited computing power instead of the teacher model.

Since the proposal of knowledge distillation, there have been many research achievements in the field of classification. Zhao et al.(Takamoto, 2020) reconstructed the classic KD loss into two parts, namely Target Class Knowledge Distillation (TCKD) and Non-Target Class Knowledge Distillation (NCKD), and then proposed Disentangled Knowledge Distillation (DKD), which enables TCKD and NCKD to play their roles more effectively and flexibly, and ultimately demonstrated the effectiveness of the method in image classification and object detection tasks. Kang et al.(Xu, 2022) proposed a new deep neural network-based class incremental learning method, which effectively solves the problem of catastrophic forgetting. These algorithms aim to solve classification problems. Regression problems are different from classification problems. Regression problems are unbounded, which means that the predicted values of the teacher model may deviate significantly from the true labels, causing negative effects on the student model. For regression problems, Makoto Takamoto et al.(Chen, 2017) proposed a new method for knowledge distillation for regression problems, which first defines a new loss function, and uses the predicted values of the teacher model to detect abnormal labels in the original data, thus increasing the robustness of the student model. Xu et al.(Kitaev, 2020) proposed a Contrastive Adversarial Knowledge Distillation (CAKD) for time series regression tasks where the student and teacher use different architectures. The core idea is to use a novel contrastive loss to achieve instance alignment between the student and teacher.

It can be seen that in regression problems, the current research mainly designs new loss functions to make the student model imitate the output of the teacher model, thus achieving the goal of knowledge distillation. In this article, a new method is proposed, which does not adjust the loss value of the student model, but uses the weighted sum of the predicted values of the teacher model and the true labels as the real labels for training the student model based on the predicted accuracy of the teacher model. The purpose of this approach is to enable the student model to imitate the output of the teacher model and learn the generalization ability of the teacher model on the one hand, and on the other hand, to reduce the impact of abnormal values in the true labels on the student model and improve the prediction accuracy of the model.

Due to the unbounded nature of time series prediction, the predicted values from the teacher model can be arbitrary. In the process of knowledge distillation, we hope that the student model can take into account both the true label and the predicted value provided by the teacher model. When the predicted value from the teacher model is close to the true label, the student model should avoid completely fitting the true label and learn the generalization ability of the teacher model. When the predicted value from the teacher model deviates significantly from the true label, the student model should try to discard the erroneous knowledge provided by the teacher model. Most existing knowledge distillation methods for regression problems add distillation losses to the student model based on the difference between the predicted value from the teacher model and the true label, so as to make the student model's predicted value closer to the teacher model's predicted value. In this paper, based on the above idea, we propose to dynamically adjust the target of the student model based on the prediction error of the teacher model, instead of adding additional losses to the student model. Formula(1) reflects the relationship between the predicted value from the teacher model, the true label, and the label used for training the student model.

$$y^{\wedge} = \lambda y^{t} + (1-\lambda) y \qquad (1)$$

y^{t}represents the predicted output of the teacher model, y represents the true label, and y^ represents the actual label used during the training of the student model. λ is a weight coefficient between 0 and 1, where a lower value of λ indicates that the student model is less influenced by the teacher model, and vice versa. The formula for calculating the weight coefficient λ is as follows:

$$\lambda = 1 - \sqrt{\frac{|y^{t} - y|}{2}} \qquad (2)$$

Formula(2) reflects the relationship between the predicted output of the teacher model and the true label, where both y^{t} and y are values normalized to the range [-1,1], and the absolute difference between $y^{t}$and y can take any value within the range [0,2], reflecting the error between the predicted output and the true label. When the error between these two values is larger, the value of λ becomes smaller. The process of knowledge distillation algorithm is as follows.

Algorithm 1: Knowledge distillation algorithm.

```
1.  Training teacher model with big dataset.
2.  Initialize the student model S
3.  **While** there are samples that haven't been
    trained **do**
4.        sample← $(x, y)$
5.        $y^{t} \leftarrow Predict(T, x)$
6.        $\lambda \leftarrow F(y^{t}, y)$
7.        $y^{\wedge} \leftarrow P(\lambda, y^{t}, y)$
8.        Train(S,x, $y^{\wedge}$)
9.  **end While**
10. **return S**
```

3.3 P-HPA

In order to meet the elastic demands of the computing network, We proposed P-HPA to solve this problem, which architecture diagram of P-HPA is presented as follows.

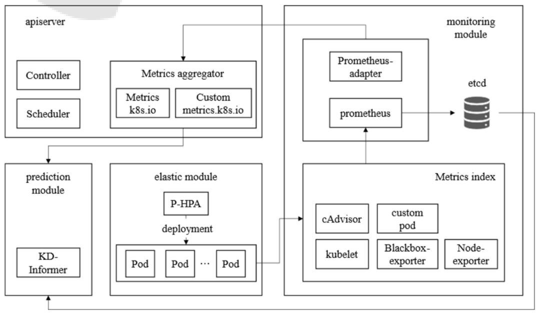


Figure 1: P-HPA architecture diagram.

The monitoring module is the foundation of the prediction and scaling modules in the elastic scaling system. The monitoring module collects various real-time computing network data through Prometheus and stores it in a persistent database.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

The prediction module obtains historical data collected by the monitoring module and stored in the database for training the prediction model. After the prediction model is trained based on this historical data, it is not directly deployed and predicted. Instead, the knowledge distillation method is used to compress the pre-trained large model into a small model, which is then deployed and provides prediction results for the scaling module. After the scaling module obtains the prediction results from the prediction module, the P-HPA calculates the expected number of replicas, and the deployment performs the scaling operation.

To ensure the normal operation of the predictive HPA, two parts of work need to be done: deploying monitoring components to obtain target metric values, and using load prediction algorithms for prediction. In this system, the target metric is QPS (Query Per Second), specifically the number of http requests per second. QPS is a custom metric that cannot be obtained by existing monitoring components in the cluster, and the metrics server can only provide core metrics such as CPU and memory information. To solve the problem of obtaining custom metrics, a set of Prometheus Custom Metrics monitoring system needs to be deployed.

Firstly, node-exporter and Prometheus are deployed, where node-exporter acts as a client to collect monitoring data from various nodes in the cluster, and Prometheus acts as a server to store the collected data in a time-series format. However, the data collected by Prometheus is incompatible with the data actually used in Kubernetes. Therefore, the kube-state-metrics component needs to be deployed, which converts the monitoring data in Prometheus into a format that can be used in Kubernetes. At this point, the monitoring system can monitor the core metrics in Kubernetes, but to obtain custom metrics such as QPS, the Prometheus-adapter component needs to be deployed, which provides an API for obtaining custom metrics. In order to observe the cluster status information clearly, the Grafana visualization component is also deployed.

At this point, the cluster already has the ability to obtain custom metrics. However, the predictive HPA still cannot function properly because it does not know what custom metrics it needs to obtain. Therefore, the Config Map file of Prometheus-adapter needs to be modified to include the rules for obtaining the QPS metric.

4 EXPERIMENT

In this section, we first introduce the dataset used in this article and the data processing process. Then we provide the experimental results of KD-Informer and P-HPA on the dataset.

4.1 Dataset

The dataset is the access volume dataset of the UK academic backbone network, which can well represent web applications with seasonal features. The dataset contains two dimensions, timestamp and access volume, with a time span of more than two months and a sampling time of 5 minutes. The original dataset was divided into three parts: training set, validation set, and test set, with a ratio of 8:1:1. There are only 20,000 pieces of data in this data set, and it is difficult for the model to get better performance when the amount of data is small. Therefore, we propose to expand the data set by data enhancement. The specific methods are as follows.

Algorithm 2: Data enhancement algorithm.

```
1. Divide the original data set into m child data sets
2. Randomly selected from all child data set data set $Sample_t$
3. $Sample_t$ weight for $\alpha$, others weight for $\beta$, $\beta = (1 - \alpha)/(m - 1)$
4. **for** s in S except $Sample_t$ **do**
5.     $S^{*} += \beta * s$
6. **end for**
7. $Sample_{m+1} = S*/m$
8. Adding gaussian noise
9. **return** $Sample_{m+1}$
```

We used the above data enhancement algorithm to extend the original data from 20,000 to 64,172. Before the start of the experiments, we took the consecutive 20,000 data as the incremental dataset. The complete extended dataset was used to train the Informer ensemble model as the teacher model, the baseline model trained on the incremental dataset was used for comparison, as well as the KD-Informer trained on the incremental dataset with the teacher model.

The evaluation metrics used in this chapter are mean squared error (MSE), mean absolute error (MAE), and model memory usage. We first compares the proposed knowledge distillation method with the knowledge distillation methods proposed in references (Chen,2017) and (Lai, 2018).

The teacher models of the three algorithms are all ensemble Informer models, and the training sets all use the incremental dataset. The student models have the same parameters and structure. Table I shows the experiment results. From the experimental results, we can see that the knowledge distillation methods proposed in this paper and in (Chen, 2017) have significantly improved the prediction accuracy of the student model. The former reduced the MSE loss by 10.8% and the MAE loss by 11.8% compared to the latter. This result demonstrates the effectiveness of the proposed knowledge distillation method.

Table 1: Knowledge distillation experiment.

Metrics	KD-Informer	(Lai, 2018).	(Chen, 2017)	none
MSE	**0.0222(±0.002)**	0.0260(±0.002)	0.0249(±0.002)	0.0266(±0.003)
MAE	**0.1069(±0.006)**	0.1238(±0.06)	0.1212(±0.006)	0.1245(±0.01)

We compare the performance of KD-Informer with Informer, Reformer, LSTNet, MTGNN under different prediction horizons of 36, 72, and 144. We set the encoder and decoder of KD-Informer to 2 and 1, and the encoder and decoder of informer and Reformer to 3 and 2, The results are obtained from the experiments conducted in this study. Table II shows the experiment results. The experimental results show that KD-Informer outperforms the baseline Informer model in terms of both prediction accuracy and model memory consumption under different prediction horizons. Specifically, for prediction horizons of 36, 72, and 144, KD-Informer achieves MSE reductions of 16.5%, 15.0%, and 13.6%, and MAE reductions of 12.0%, 13.3%, and 30.2%. Based on these experimental results, we can conclude that KD-Informer can learn the generalization ability of the teacher model and improve the prediction accuracy, and achieve similar results to the teacher model using a small amount of data and smaller memory.

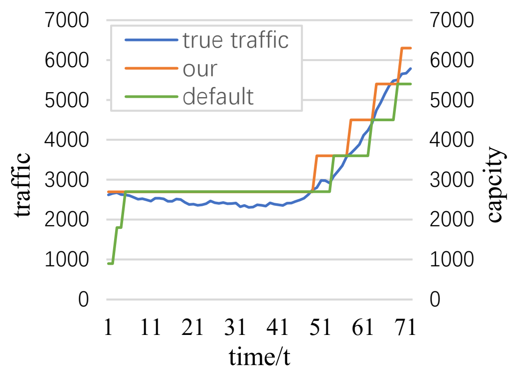
Table 2: Prediction Model Experiment.

Model	Metrics	36	72	144
Informer	MAE	0.1245(±0.01)	0.1601(±0.02)	0.1768(±0.03)
	MSE	0.0266(±0.004)	0.0412(±0.008)	0.0580(±0.01)
	MEMORY	93.7		
Reformer	MAE	0.1583(±0.02)	0.2080(±0.03)	0.2162(±0.04)
	MSE	0.0493(±0.003)	0.0683(±0.004)	0.0713(±0.01)
LSTNet	MAE	0.1175(±0.02)	0.1726(±0.04)	0.2308(±0.06)
	MSE	0.0437(±0.008)	0.0999(±0.010)	0.1524(±0.015)
MTGNN	MAE	0.1103(±0.04)	0.1652(±0.04)	0.2219(±0.08)
	MSE	0.0283(±0.004)	0.0660(±0.005)	0.1172(±0.01)
KD-	MAE	0.1096(±0.006)	0.1388(±0.008)	0.1528(±0.01)

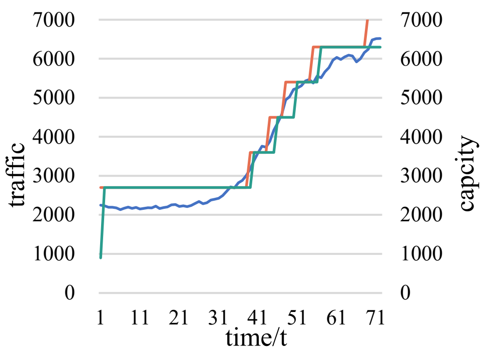
Model	Metrics	36	72	144
Informer	MAE	0.1245(±0.01)	0.1601(±0.02)	0.1768(±0.03)
	MSE	0.0266(±0.004)	0.0412(±0.008)	0.0580(±0.01)
	MEMORY	93.7		
Informer	MSE	0.0222(±0.002)	0.0350(±0.004)	0.0405(±0.005)
	MEMORY	**62.7**		

We compare the experimental results of the Informer model without knowledge distillation and the model with knowledge distillation to verify the effectiveness of knowledge distillation. Table III shows the experiment results. The experimental results are shown as follows. From the experimental results, we can see that the addition of knowledge distillation greatly improves the predictive performance of the model compared with the default Informer.

According to the experimental results in Fig. 2, we can see that compared with default HPA, P-HPA can realize the capacity expansion in advance when the traffic surges. Therefore, in the computing network, P-HPA proposed in this paper can effectively avoid the situation of QoS reduction caused by insufficient resource supply. As shown in Fig.3, we can also see that P-HPA based on KD-Informer and the P-HPA based on Informer can basically achieve capacity expansion before load burst, but the expansion ratio of the former is more accurate than that of the latter. At the same time, it also proves that KD-Informer has improved the prediction accuracy compared with Informer algorithm. From the above experimental results and analysis, we can draw the following conclusions: Compared with the default HPA mechanism, the P-HPA can realize capacity expansion before load changes, effectively ensure the QoS of user applications, and provide a solution to the resource elastic scaling problem under the computing network.

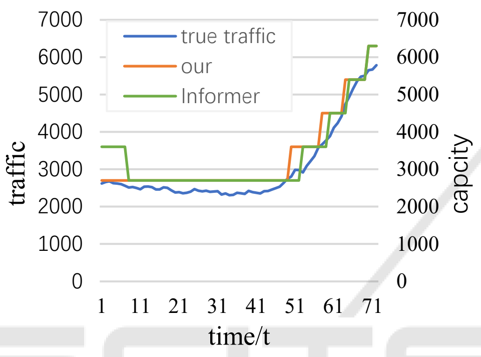


(a) sequence 1

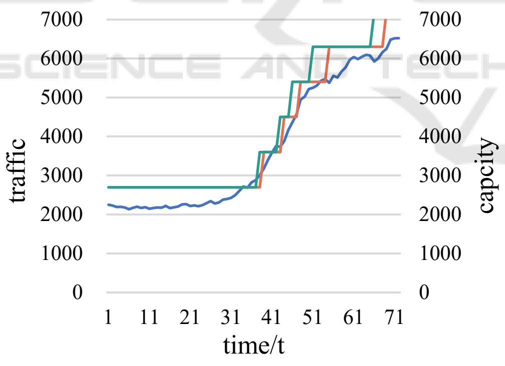


(b) sequence 2

Figure 2: Comparison between P-HPA and default HPA.



(a) sequence 1



(b) sequence 2

Figure 3: Comparison between P-HPA and based on KD-Informer and Informer.

5 CONCLUSION

In this paper, we proposed a new knowledge distillation method, a advanced time series prediction model and a predictive HPA. A new knowledge distillation method is proposed to solve the problem that the prediction model occupies a large memory and requires a large number of training samples. In order to achieve advance expansion, we proposed a time series prediction model and predictive HPA mechanism, and our method achieves advance accurate scaling compared with the existing methods. The experiment results have shown that our methods are better the state of arts.

In future work, we plan to apply our P-HPA in a real cloud container cluster and try to improve the accuracy of P-HPA. In addition, we can also try to design more advanced knowledge distillation algorithm to address the challenges in CFN.

REFERENCES

Kong X, Wu Y, Wang H, et al. Edge Computing for Internet of Everything: A Survey[J]. *IEEE Internet of Things Journal*, 2022, 9(23): 23472-23485. https://doi.org/10.1109/JIOT.2022.3200431

Garg S, Kaur K, Kaddoum G, et al. Security in IoT-driven mobile edge computing: New paradigms, challenges, and opportunities[J]. *IEEE Network*, 2021, 35(5): 298-305. https://doi.org/10.1109/MNET.211.2000526

Zhou Z, Zhang C, Ma L et al. AHPA : Adaptive Horizontal Pod Autoscaling Systems on Alibaba Cloud Container Service for Kubernetes[J]. *arXiv preprint arXiv*, 2023. https://arxiv.org/abs/2303.03640

Toka L, Dobreff G, Fodor B et al. Adaptive AI-based auto-scaling for Kubernetes[J]. *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing*, 2020: 599–608. https://doi.org/10.1109/CCGrid49817.2020.00-33

Shin S H, Kim D, Kim T et al. High Concurrency Response Strategy based on Kubernetes Horizontal Pod Autoscaler[J]. *Journal of Physics: Conference Series*, 2023, 2451(1): 012001. https://doi.org/10.1088/1742-6596/2451/1/012001

MK Mohan Murthy, HA Sanjay J A. Threshold based auto scaling of virtual machines in cloud environment[A]. *IFIP International Conference on Network and Parallel Computing*[C]. 2014: 247–256. https://doi.org/10.1007/978-3-662-44917-2_21

Paulo Pereira, Jean Araujo, Paulo Maciel et al. A Hybrid Mechanism of Horizontal Auto-scaling Based on Thresholds and Time Series[C]. *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. 2019: 2065–2070. https://doi.org/10.1109/SMC.2019.8914522

Yahya Al-Dhuraibi, Fawaz Paraiso F P. Autonomic Vertical Elasticity of Docker Containers with ELASTICDOCKER[C]. *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*. 2017: 25–30. https://doi.org/10.1109/CLOUD.2017.67

Gourav Rattihalli, Madhusudhan Govindaraju, Hui Lu et al. Exploring potential for non-disruptive vertical auto scaling and resource estimation in kubernetes[C]. *2019 IEEE 12th International Conference on Cloud*

Computing (CLOUD). 2019: 08–13. https://doi.org/10.1109/CLOUD.2019.00018

Hamzeh Khazaei et al. Elascale: Autoscaling and Monitoring as a Service[J]. *arXiv preprint arXiv*, 2017. https://arxiv.org/abs/1711.03204

Zhicheng Cai et al. Inverse Queuing Model-Based Feedback Control for Elastic Container Provisioning of Web Systems in Kubernetes[J]. *IEEE Transactions on Computers*, 2022, 71(2): 337–348. https://doi.org/10.1109/TC. 2021.3049598

Toka L, Dobreff G, Fodor B, et al. Adaptive AI-based auto-scaling for Kubernetes[C]//*2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE, 2020: 599-608. https://doi.org/10.1109/CCGrid49817.2020.00-33

Kader N I A, Yusof U K, Khalid M N A, et al. A review of long short-term memory approach for time series analysis and forecasting[C]//*International Conference on Emerging Technologies and Intelligent Systems. Cham: Springer International Publishing*, 2022: 12-21. https://doi.org/10.1007/978-3-031-20429-6_2

Li S, Wunsch D C, O'Hair E A et al. Using neural networks to estimate wind turbine power generation[J]. *IEEE Transactions on Energy Conversion*, 2001, 16(3): 276–282. https://doi.org/10.1109/60.937208

Punia S, Nikolopoulos K, Singh S P, et al. Deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail[J]. *International journal of production research*, 2020, 58(16): 4964-4979.

H. Arabnejad, C. Pahl, P. Jamshidi and G. Estrada. A Comparison of Reinforcement Learning Techniques for Fuzzy Cloud Auto-Scaling[C]. *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. 2017: 64–73. https://doi.org/10.1109/CCGRID.2017.15

Horovitz S. Horovitz et al. Efficient Cloud Auto-Scaling with SLA Objective Using Q-Learning[C]. *2018 IEEE 6th International Conference on Future Internet of Things and Cloud*. 2018: 06–08. https://doi.org/10.1109/FiCloud.2018.00020

Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network[J]. *Physica D: Nonlinear Phenomena*, 2020, 404: 132306.

Vaswani A, Shazeer N, Parmar N et al. Attention Is All You Need[J]. *Advances in neural information processing systems*, 2017, 30. https://doi.org/10.1016/j.physd.2019.132306

Forecasting T, Zhou H, Zhang S et al. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting[C]. *Proceedings of the AAAI conf. on Artificial Intelligence*. 2021: 11106–11115. https://doi.org/10.1609/aaai.v35i12.17325

Zhao B, Song R, Qiu Y. Decoupled Knowledge Distillation[J]. *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2022: 11953–11962. https://arxiv.org/ abs/2203.08679

Kang M, Park J. Class-Incremental Learning by Knowledge Distillation with Adaptive Feature Consolidation[J]. *Proceedings of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2022: 16071–16080. https://arxiv.org/abs/ 2204.00895

Takamoto M, Morishita Y. An Efficient Method of Training Small Models for Regression Problems with Knowledge Distillation[C]. *2020 IEEE Conf. on Multimedia Information Processing and Retrieval (MIPR)*, 2020: 67–72. https://doi.org/10.1109/MIPR49039. 2020.00021

Xu Q, Chen Z, Ragab M et al. Neurocomputing Contrastive adversarial knowledge distillation for deep model compression in time-series regression tasks[J]. *Neurocomputing*, Elsevier B.V., 2022, 485: 242–251. https://doi.org/10.1016/j.neucom.2021.04.139

Chen G, Choi W, Yu X, et al. Learning efficient object detection models with knowledge distillation[C] *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017: 742-751. https://dl.acm.org/doi/abs/10.5555/3294771.3294842

Kitaev N, Kaiser Ł, Levskaya A. Reformer: The Efficient Transformer[J]. *arXiv preprint arXiv*, 2020: 1–12. https://arxiv.org/abs/2001.04451

Lai G, Chang W, Yang Y et al. Modeling long-and short-term temporal patterns with deep neural networks[C]. *The 41st international ACM SIGIR conference on research & development in information retrieval*. 2018: 95–104. https://doi.org/10.1145/3209978.3210006

Wu Z, Pan S, Long G et al. Connecting the dots: Multivariate time series forecasting with graph neural networks[C]. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020: 753–763. https://doi.org/10.1145/3394486.3403118