Early Bird: Ensuring Reliability of Cloud Systems Through Early Failure Prediction

Yudong Liu[†], Minghua Ma[‡*], Pu Zhao[†], Tianci Li[†], Bo Qiao[†], Shuo Li[§], Ze Li[‡], Murali Chintalapati[‡], Yingnong Dang[‡], Chetan Bansal[‡], Saravan Rajmohan[‡], Qingwei Lin[†], Dongmei Zhang[†]

[†]Microsoft, Beijing, China

[‡]Microsoft, Redmond, USA

***Abstract*—As cloud service continues to dominate various sectors, the reliability of cloud infrastructures becomes crucial. Traditional methods of failure prediction often fall short in providing sufficient time for preventative measures. This paper presents a failure prediction framework, EARLY BIRD, designed to address these challenges by integrating novel data handling and prediction strategies. Our approach utilizes enhanced sample generation techniques and a unique adaptive loss function within a unified prediction model, aiming for early and precise failure detection. We present a comprehensive analysis conducted at Microsoft, demonstrating the ability to predict potential failures up to 20 minutes earlier than conventional methods while maintaining accuracy across various prediction models, including LSTM and Transformer.**

***Index Terms*—Failure Prediction, Cloud Systems, Reliability**

I. INTRODUCTION

Large-scale cloud providers, including Microsoft Azure, AWS, and Google Cloud, have been serving millions of customers with the rapid development of cloud technology [1]–[3]. It is crucial to maintain the reliability of these systems from both user experience and economic perspectives [4]–[6]. However, cloud failures, such as node [7]–[9] and disk failures [10], [11], have been a major threat to the cloud reliability.

To improve reliability, many approaches [12]–[14] have been proposed to predict cloud failures. These approaches frame the prediction of such failures as a time series classification task and employ temporal sequential models such as LSTM [15], RNN [16], Transformer [17], and TCNN [18], to predict potential failures in the next time interval. However, previous approaches mainly focus on enhancing prediction accuracy and neglect the importance of making predictions much earlier. Engineers may require significant time to take proactive actions for potential failures, and predicting only one timestamp ahead could be too late. Therefore, making early predictions is of great importance, in addition to making accurate predictions. The topic of early failure prediction [14], [19], [20] has been investigated by many researchers.

Our paper introduces a novel solution to cloud failure prediction that not only focusing on enhancing accuracy, but also our objective is to achieve both prompt and precise predictions. To attain early prediction, a data sample generation method is employed to generate samples with varying *ahead interval*, akin to a sliding window technique to collect data from several windows. The samples with larger *ahead intervals* are called hard examples, since they contain less indicative information about an impending failure event. Inspired by multi-task learning [21], We propose a unified prediction model that captures normal patterns among these samples, resulting in improved early prediction performance. Additionally, we address the issue of hard samples by introducing an adaptive loss function that balances the weights among different data samples based on their ahead interval. This allows for more attention to be paid on hard samples, ultimately enhancing prediction accuracy even further.

To demonstrate the effectiveness and robustness of our EARLY BIRD solution, we conduct experiments in the cloud failure prediction task, where making early prediction is obviously important for taking proactive actions in advance. The experimental results indicate that, based on various state-of-the-art time series models, such as LSTM, Transformer and more, our EARLY BIRD solution is capable of making early prediction while keeping comparable prediction accuracy compared to previous methods. Encouragingly, our EARLY BIRD allows us to maintain a constant F1-score while making predictions in approximately 20 minutes in the context of cloud failure prediction.

The main contributions of this paper are as follows:

- We focus on early prediction in cloud failure prediction, which is crucial for enabling engineers to make better preparations and take proactive actions in a timely manner.
- We propose a unified EARLY BIRD solution that integrates data samples from different ahead intervals to observably enhance the prediction accuracy of the early prediction. Besides, we propose a novel adaptive loss function to balance the weights of samples from different ahead interval.
- Extensive experiments in the cloud failure prediction demonstrate our EARLY BIRD solution can achieve early failure prediction, while maintaining comparable accuracy.

II. METHODOLOGY

A. Problem Definition

Concept of ahead interval. In cloud systems, system-level signals are periodically recorded by various monitors, resulting in time series data with M dimensions, representing different attributes of signals. The data is denoted as X_t, which covers a time sequence of length h from timestamp $t-h+1$ to timestamp t. To predict the state of an object at T-th timestamp, a prediction model takes X_t as input and outputs

* Corresponding author. Email: minghuama@microsoft.com

§ Work was performed when he was at Carnegie Mellon University

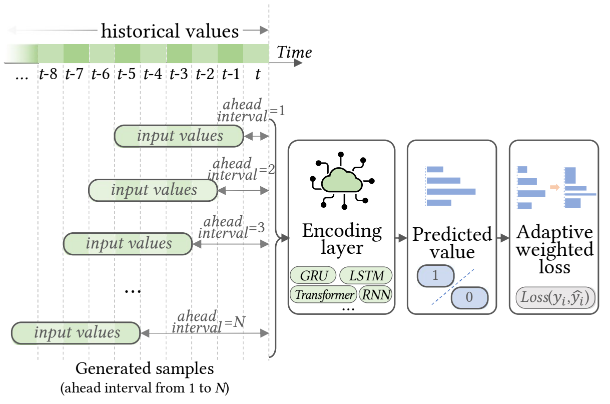

Fig. 1. Architecture of early prediction model

a corresponding classification of either "health" or "failure". The number of timestamps ahead for prediction is denoted by $aheadinterval$, where previous approaches have set it to 1, meaning $t = T - 1$, for simplicity.

Objective. In practical situations, taking mitigation actions for potential failures can be a time-consuming process. Therefore, predicting failures only one timestamp ahead ($aheadinterval = 1$) may not be sufficient. To address this issue, larger values of $aheadinterval$ are desirable, as they enable more accurate predictions further into the future. Specifically, EARLY BIRD aims to maximize the accuracy of the prediction model at each value of $aheadinterval$, denoted as A_1, A_2, ..., A_N, rather than solely focusing on A_1, where N represents the maximum value of $aheadinterval$ based on the given scenarios.

B. Data Sample Generation

The proposed solution requires the positive and negative examples with a range of ahead intervals, allowing the model to generalize better across different predictive scenarios. Generating positive samples for different values of $aheadinterval$ is a straightforward process. If an object experiences failure at timestamp T, then the label $y_i = 1$ is assigned to data X_{T-i} with $aheadinterval$ value of i, indicating that failure will occur after i timestamps. For objects that do not experience failure, we randomly select a timestamp T and generate negative samples. In this case, the label $y_i = 0$ is assigned to data X_{T-i} with $aheadinterval$ value of i, indicating that there will be no failure after i timestamps.

The selection of maximum value of $aheadinterval$, denoted as N, is a critical factor in the prediction process. If N is large, it may result in impractical predictions since excessively early time series data may not indicate failure and can be considered as negative samples. If N is small, time may be limited to take mitigation actions. Therefore, it is essential to select an appropriate value for the maximum $aheadinterval$.

C. EARLY BIRD

One intuitive approach to maximize the values of $A_1, \ldots, A_N$ would be to design different models for each $aheadinterval$. However, there are two significant disadvantages to utilizing multiple models. Firstly, maintaining multiple models can be costly and inefficient. Secondly, there is a non-negligible correlation among samples of different $aheadinterval$. Although different models handle different durations before failure, the data samples from different $aheadinterval$ contain similar feature patterns that indicate potential failure. In fact, predicting cloud failures for different $aheadinterval$ can be viewed as different tasks. Based on the insight of multi-task learning [22], leveraging these tasks in the same model could enhance the performance of different tasks. Therefore, we propose a unified model to make predictions for different $aheadinterval$.

The model architecture is depicted in Figure 1, where the input is time series data with different $aheadinterval$. After extracting the feature vector of the input data, the model outputs the probability of failure after i timestamps, where i denotes the corresponding $aheadinterval$. The loss function in Figure 1 will be explained in the following section.

D. Adaptive Weighted Loss Function

The tradition *cross entropy* loss function can be expressed in the following:

$$Loss(y_i, \hat{y_i}) = \begin{cases} -log(\hat{y_i}), & y_i = 1 \\ -log(1 - \hat{y_i}), & y_i = 0 \end{cases}$$

Here, $\hat{y_i}$ represents the prediction result, while y_i denotes the ground truth. For positive samples, the loss is calculated as $-log(\hat{y_i})$, while $-log(1 - \hat{y_i})$ is used for negative samples. Note that the weight for each sample is equally considered in *cross entropy* loss function.

For the same object that experiences failure, it is much easier for the model to make predictions for samples with smaller $aheadinterval$ compared to those with larger $aheadinterval$. This is logically reasonable since data closer to the failure could provide more indications. Simply utilizing the traditional *cross-entropy* loss function could neglect the learning for hard samples, for example, samples with larger $aheadinterval$. Therefore, different weights are assigned to positive samples with different $aheadinterval$. As for negative samples, there is no need to balance the weights, since the data does not contain any failure indications.

To address the aforementioned issue, we propose a modification to the *cross-entropy* loss function and introduce a novel adaptive loss function to balance the weights among positive samples with different $aheadinterval$. The loss between the prediction result and ground truth can be expressed as follows:

$$Loss(y_i, \hat{y_i}) = \begin{cases} -\alpha \cdot (1 - \frac{e^{\hat{y_i}}}{\sum e^{\hat{y_j}}})^{\gamma} \log(\hat{y_i}), & y_i = 1 \\ -\log(1 - \hat{y_i}), & y_i = 0 \end{cases}$$

where y_i denotes the ground truth for some object with ahead interval i, $\hat{y_i}$ denotes the corresponding prediction score, and $\hat{y_j}$ denotes prediction score for the same object with different ahead interval. α and γ are hyper-parameters. We set the value of α as the ratio between positive and negative samples to deal with the imbalance issue, and γ as 2 experimentally. We will conduct ablation study on the hyper-parameters in Section III-D. The loss for negative samples is the same with one in *cross entropy* loss. While for positive samples, we balance the weights for various ahead interval by corresponding prediction scores. Specifically, we set the weight for $\hat{y_i}$ as the sum of all other y_j divided by total sum, and we utilize the power of natural logarithms *e* to do normalization. In this way, samples with low prediction scores are distributed larger weights, which helps the learning for hard samples. Our adaptive loss function is similar to the focal loss [23]. A comparison between the two loss functions in Section III-D.

III. Experiment

In this section, we evaluate the effectiveness of our Early Bird method on a cloud failure prediction scenario.

A. Compared Approaches

We designed several prediction approaches to compare with, i.e., *baseline*, *multi-model*, *early-pred*, and *adploss*.

- *baseline* trains the model with $aheadinterval = 1$, while tests on dataset with various ahead intervals. It is obvious to see that the *baseline* is consistent with previous time series prediction methods.
- *multi-model* trains multiple time series models, where each model corresponds to a specific ahead interval. We test the result for each model on dataset with corresponding ahead interval. It is noted that the performance of $aheadinterval = 1$ for the two methods is consistent, since they share the same training data.
- *early-pred* method trains a unified model for samples with different ahead interval.
- *adploss* denotes utilizing adaptive weighted loss function based on our method.

B. Time Series Models and Implementations

To demonstrate the robustness of Early Bird, we conduct experiments based on the following state-of-the-art models:

- **RNN** [24]: Recurrent neural network is a popular deep learning model for time series data. RNN uses its recurrent unit to highlight the distinction between positive and negative samples from the input time series data during prediction.
- **LSTM** [25]: Compared to traditional RNNs, Long Short-Term Memory is an enhanced recurrent neural network architecture that was created to better accurately describe time series and their long-range dependencies. Long term characteristics enable LSTM to consistently outperform RNN and can often classify challenging samples in RNN.
- **GRU** [26]: GRU can be regraded as a simplicity version of LSTM, which does not process any internal memory. Compared to LSTM, GRU is much simpler and requires less computational power.
- **Transformer** [17]: Transformer is a brand-new, attention-based deep learning technique for time series prediction to extract the temporal information from time series data.

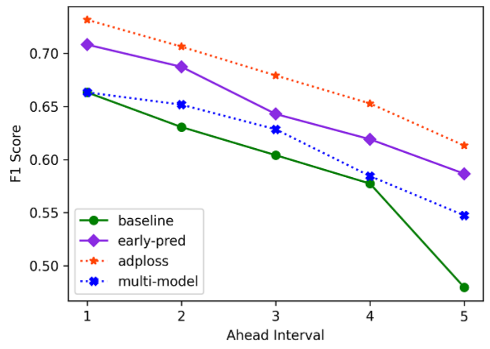

Fig. 2. F1-score for cloud failure prediction under different ahead intervals.

All experiments are conducted on a workstation equipped with NVIDIA Tesla P100 GPU and CUDA 10.2. The code is implemented based on Python 3.8 and PyTorch 1.9. During the training process, we utilize Adam optimizer and set the initial learning rate as $2e^{-3}$. In addition, the number of the training epoch is set to 100 and the batch size is 64. Considering the characteristics of the three scnearios, we set the value of max ahead interval as 5.

C. Experimental Results

In this section, we conduct experiments based on different time series models (such as Transformer and LSTM). We compare the performance of four approaches: *baseline*, *multi-model*, *early-pred*, and *adploss*. The descriptions of these approaches can be referred to Section III-A.

The results are presented the results in the Table I and Figure 2, which shows the F1-score for different methods based on Transformer. From the table and figure, we can draw several conclusions. Firstly, as the value of ahead interval increases, the model performance for various methods drops observably. On average, the performance for $aheadinterval = 1$ exceeds $aheadinterval = 2$ by 2.78% F1-score, 5.58% for $aheadinterval = 3$, 8.68% for $aheadinterval = 4$, and 17.03% for $aheadinterval = 5$. This is consistent with the intuition of the trade-off between early prediction and higher accuracy. In addition, the recall values observed for an ahead interval of 5, with the average of 46.55%, reflecting the inherent challenge of predicting failures well in advance. Secondly, the performance of Transformer exceeds RNN by 4.50%, LSTM by 2.29%, and GRU by 2.74% respectively, which accords with the recognized principle that Transformer is very effective against other prediction models. Thirdly, based on *multi-model*, the performance of three models all exceeds *baseline* when $aheadinterval > 1$, especially for $aheadinterval = 5$. This is reasonable since the former trains model for each ahead interval. While based on *early-pred*, the performance of three models is enhanced to a large margin, which demonstrates the effectiveness of our method. In addition, we further enhance the performance of three models based on our proposed adaptive loss function, named Early Bird. In conclusion, our method outperforms *baseline* by an average of 7.76% F1-score for $aheadinterval = 1$, 8.49% for $aheadinterval = 2$, 7.88% for $aheadinterval = 3$, 8.54%

TABLE I
COMPARATIVE RESULTS OF VARIOUS TIME SERIES MODELS WITH AND WITHOUT EARLY BIRD ON CLOUD FAILURE PREDICTION DATA. P, R, AND F1 ARE REFERRING TO PRECISION, RECALL, AND F1-SCORE, RESPECTIVELY.

Model	Approach	ahead interval=1			ahead interval=2			ahead interval=3			ahead interval=4			ahead interval=5		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
LSTM	*baseline*	92.34	49.01	64.04	89.66	46.91	61.59	86.00	44.33	58.50	69.84	45.36	55.00	52.01	43.30	47.26
	multi-model	92.34	49.01	64.04	90.23	48.53	63.11	87.52	45.31	59.71	71.88	49.36	58.53	64.05	45.86	53.45
	early-pred	94.12	54.08	68.69	93.38	51.75	66.59	86.28	49.73	63.09	84.70	48.77	61.90	84.47	45.30	58.97
	EARLY BIRD	95.63	57.39	**71.73**	93.52	55.82	**69.91**	91.53	51.35	**65.79**	89.27	49.77	**63.91**	86.33	46.37	**60.33**
Transformer	*baseline*	94.28	51.16	66.33	89.21	48.77	63.06	90.28	45.39	60.41	73.55	47.51	57.73	52.17	44.38	47.94
	multi-model	94.28	51.16	66.33	90.36	49.75	64.17	89.11	48.54	62.85	74.27	48.17	58.44	72.87	43.82	54.73
	early-pred	94.46	56.66	70.83	92.51	54.69	68.74	93.76	48.92	64.29	88.69	47.55	61.91	86.35	44.42	58.66
	EARLY BIRD	95.16	59.44	**73.17**	92.58	57.11	**70.64**	92.24	53.75	**67.92**	89.35	51.43	**65.28**	90.06	46.47	**61.31**
GRU	*baseline*	93.22	48.25	63.59	87.44	46.65	60.84	75.63	46.93	57.51	76.22	43.10	55.06	78.92	33.19	46.73
	multi-model	93.22	48.25	63.59	89.96	46.34	61.17	89.73	43.31	58.42	87.93	44.87	59.42	71.51	41.99	52.91
	early-pred	95.24	54.84	69.60	93.78	52.68	67.46	90.32	47.51	62.27	84.47	47.68	60.96	75.63	46.39	57.51
	EARLY BIRD	92.17	58.48	**71.56**	93.22	55.68	**69.71**	89.63	52.46	**66.18**	90.19	58.87	**63.39**	84.55	48.17	**61.37**
RNN	*baseline*	92.46	46.44	61.83	88.62	44.41	59.17	79.55	44.49	57.06	82.03	39.45	53.28	74.91	32.92	45.74
	multi-model	92.46	46.44	61.83	90.17	45.48	60.46	88.35	43.92	58.67	86.99	42.64	57.23	73.58	41.61	53.16
	early-pred	94.77	52.19	67.31	91.22	50.79	65.25	89.79	46.32	61.11	88.63	45.15	59.82	79.41	43.83	56.48
	adploss	93.59	56.38	**70.37**	91.37	54.51	**68.28**	87.33	51.88	**65.09**	90.86	47.82	**62.66**	86.71	47.26	**61.18**
Average	*baseline*	93.08	48.71	63.95	88.73	46.51	61.17	82.86	45.29	58.37	75.41	43.86	55.27	64.50	38.45	46.92
	multi-model	93.08	48.71	63.95	90.18	47.55	62.23	88.68	45.27	59.91	80.27	46.26	58.40	70.50	43.11	53.56
	early-pred	94.65	54.44	69.11	92.72	52.48	67.01	90.04	48.12	62.69	86.62	47.29	61.15	81.47	54.38	57.90
	EARLY BIRD	94.14	57.92	**71.71**	92.67	55.78	**69.66**	90.18	52.86	**66.25**	89.03	51.97	**63.81**	86.91	46.55	**61.05**

TABLE II
COMPARISON UNDER DIFFERENT γ IN CLOUD FAILURE PREDICTION TASK. "AH" DENOTES AHEAD INTERVAL.

	ah=1	ah=2	ah=3	ah=4	ah=5
$\gamma = 2$	72.85	70.38	67.82	65.29	61.47
$\gamma = 2.4$	72.59	71.06	68.17	64.71	60.99
$\gamma = 2.8$	73.17	70.64	67.92	65.28	61.31
$\gamma = 3.2$	73.33	70.59	67.54	65.46	60.82

TABLE III
COMPARISON OF ADAPTIVE LOSS AND FOCAL LOSS IN CLOUD FAILURE PREDICTION TASK.

ahead interval	Method	P	R	F
1	*focal loss*	89.06	59.32	71.21
	adploss	95.16	59.44	**73.17**
2	*focal loss*	91.42	53.28	67.32
	adploss	92.58	57.11	**70.64**
3	*focal loss*	86.39	52.84	65.57
	adploss	92.24	53.75	**67.92**
4	*focal loss*	88.58	46.68	62.83
	adploss	89.35	51.43	**65.28**
5	*focal loss*	78.33	48.24	59.71
	adploss	90.06	46.47	**61.31**

$ahead interval = 4$ and 14.13% for $ahead interval = 5$. An alternative interpretation of our results is that by utilizing our EARLY BIRD, we can make predictions within approximately 4 ahead interval (equivalent to 20 minutes) while maintaining the same F1-score.

D. Ablation Study

To demonstrate the robustness of our method, we firstly compare the prediction accuracy underlying different values of γ. Note that we set the value of α as the ratio between positive and negative samples to deal with imbalance issue. Then we compare our method with focal loss, which is similar with our adaptive loss function in the aspect of format.

1) Hyper-parameter of adaptive loss function: We firstly explore the robustness of prediction accuracy with changes of hyper-parameter. We conduct experiments in cloud failure prediction task based on Transformer model. From Table II, we can observe that the prediction accuracy keeps stable as the change of γ, with a bias less than 1% F1-score. The result demonstrates the robustness of our adaptive loss function.

2) Comparison with focal loss: Our adaptive loss function seems similar with focal loss in the aspect of format. However, as mentioned in Section II-D, the two loss functions are actually different. We compare our loss function with focal loss based on Transformer model in cloud failure prediction task. According to the result in Table III, our proposed adaptive loss function outperforms focal loss by 1.96 % for $ahead interval = 1$, 3.32 % for $ahead interval = 2$, 2.35 %

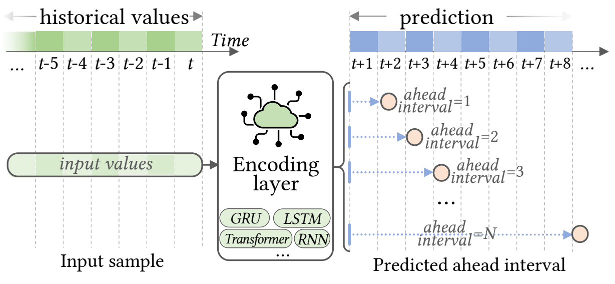

Fig. 3. Directly predicting ahead interval.

for $ahead interval = 3$, 2.45 % for $ahead interval = 4$, and 1.60 % for $ahead interval = 5$, respectively.

IV. DISCUSSION

Can we predict the ahead interval? We collect ahead interval values for various samples in the training phase, and predict a failure probability score for the near future. As depicted in Figure 3, suppose that a failure occurs at timestamp T. The model predicts x when given data X_{T-x}, indicating that failure will occur after x timestamps. As for negative samples,

TABLE IV
COMPARISON WITH DIRECTLY PREDICTING AHEAD INTERVAL IN CLOUD FAILURE PREDICTION TASK. HERE *multi-class* DENOTES REGARDING PREDICTION OF AHEAD INTERVAL AS A MULTI CLASSIFICATION PROBLEM.

ahead interval	Method	P	R	F
1	*multi-class*	94.28	51.16	66.33
	EARLY BIRD	95.16	59.44	**73.17**
2	*multi-class*	89.21	48.77	63.06
	EARLY BIRD	92.58	57.11	**70.64**
3	*multi-class*	90.28	45.39	60.41
	EARLY BIRD	92.24	53.75	**67.92**
4	*multi-class*	73.55	47.51	57.73
	EARLY BIRD	89.35	51.43	**65.28**
5	*multi-class*	52.17	44.38	47.94
	EARLY BIRD	90.06	50.96	**61.31**

the model should output zero. This approach transforms the problem from binary classification into multi-classification.

Directly predicting ahead interval seems more intuitive and applicable in real practice, but we do not take this solution based on two considerations. Firstly, it is easy to obtain the labels for positive samples, but hard to label negative samples properly since failure will not occur. Secondly, predicting the specific failure timestamp is difficult in real practice. Historical time series data only indicates potential failure in the near future, and the occurrence of failure depends on complicated factors beyond data attributes. Instead, predicting failure tendency in the near future is much more reasonable. To demonstrate our injection, we conduct an experiment based on cloud failure prediction task, .and compare our method with the multi-classification baseline mentioned above. Transformer is used for both methods. Considering that the metric F1-score fits for binary classification, for multi-classification baseline, we calculate F1-score for each ahead interval separately to ensure fair compare. As shown in Table IV, the result of various ahead interval underlying the multi-classification baseline performs worse than our method.

Deployment in practice. We deploy the EARLY BIRD pipeline on Azure Databricks. The raw time series data is processed with data cleaning and feature engineering before fed into the model. During the model training phase, each positive sample is augmented with $aheadinterval = 4$. The α in adaptive function is set as the ratio between positive and negative samples in training dataset to deal with the imbalance issue, and β is set as 2 experimentally. During the model inference phase, time series data is fed into the model and the failure probability scores are obtained. Data samples with top rank scores are provided to engineers. Engineers can take proactive actions to alleviate the potential cloud failures according to their domain knowledge. In addition, A/B testing is conducted to compare the number of cloud failures saved by prediction model to evaluate the effectiveness of our method. Compared to baseline approach, EARLY BIRD observably reduces the number of cloud failures, which demonstrates that it can enhance the performance of cloud failure prediction and improve the cloud service reliability.

V. RELATED WORK

Cloud system management. Cloud system management involves the continuous monitoring, configuration, and optimization of cloud resources to ensure performance, security, and cost-efficiency. Anomaly detection plays an important role in identifying unusual patterns in system performance that often precede failures [27]–[32]. Similarly, root cause analysis helps in diagnosing the underlying issues that trigger these anomalies, enabling more accurate and actionable predictions [33]–[35]. Additionally, cloud failure prediction leverages historical data and machine learning algorithms to foresee potential system failures, enabling proactive management and minimizing downtime [36], [37].

Early failure prediction. Early failure prediction has gained popularity across various fields, including healthcare [38], manufacturing [39], and finance [40], due to its potential to preemptively address issues and minimize downtime. In cloud systems, this area of research is particularly critical as it helps maintain system reliability and ensure continuous service availability. Cloud systems are prone to failures due to their complex and distributed nature, making early prediction essential. Research in this area often focuses on developing models that can predict failures before they occur, allowing for proactive measures to prevent them.

Multi-step-ahead prediction. Our proposed EARLY BIRD may seem similar to multi-step-ahead prediction [41], [42]. The EARLY BIRD method is designed for single-step prediction, which focuses on a single value in the near future. Multi-step-ahead prediction, on the other hand, aims to predict a series of values for future timestamps. For EARLY BIRD, the ahead interval denotes the number of timestamps ahead we make predictions. In contrast, multi-step-ahead prediction has more than one blue part, indicating that multiple values are predicted. In our approach, we have incorporated a specialized design that facilitates online prediction through multi-task learning and adaptive weighted loss function.

VI. CONCLUSION

Cloud failure prediction is a crucial aspect of ensuring cloud reliability, and many approaches have been proposed to enhance prediction performance. In this paper, we aim to predict failures early and accurately. To address the accuracy issue for early prediction, we propose a unified method called EARLY BIRD, which leverages samples from different timestamps to capture common patterns among samples with various ahead intervals, using a data sample generation method. Additionally, we propose an adaptive loss function that pays more attention to data samples with larger ahead intervals. Our method can be easily integrated into various state-of-the-art time series models, such as LSTM, Transformer, and more. We conducted experiments on an industrial cloud failure prediction dataset in Microsoft, and the results show that we can make predictions several timestamps ahead while maintaining comparable accuracy to previous methods.

References

[1] D. Ardagna, B. Panicucci, and M. Passacantando, "A game theoretic formulation of the service provisioning problem in cloud systems," in *Proceedings of WWW*, 2011, pp. 177–186.

[2] S. He, X. Zhang, P. He, Y. Xu, L. Li, Y. Kang, M. Ma, Y. Wei, Y. Dang, S. Rajmohan *et al.*, "An empirical study of log analysis at microsoft," in *Proceedings of the 30th ESEC/FSE*, 2022, pp. 1465–1476.

[3] S. Zhang, P. Jin, Z. Lin, Y. Sun, B. Zhang, S. Xia, Z. Li, Z. Zhong, M. Ma, W. Jin *et al.*, "Robust failure diagnosis of microservice system through multimodal data," *IEEE Transactions on Services Computing*, vol. 16, no. 6, pp. 3851–3864, 2023.

[4] Y. Chen, X. Yang, Q. Lin, H. Zhang, F. Gao, Z. Xu, Y. Dang, D. Zhang, H. Dong, Y. Xu *et al.*, "Outage prediction and diagnosis for cloud service systems," in *Proceedings of WWW*, 2019, pp. 2659–2665.

[5] Z. Yu, M. Ma, C. Zhang, S. Qin, Y. Kang, C. Bansal, S. Rajmohan, Y. Dang, C. Pei, D. Pei *et al.*, "Monitorassistant: Simplifying cloud service monitoring via large language models," in *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, 2024, pp. 38–49.

[6] H. Li, M. Ma, Y. Liu, S. Qin, B. Qiao, R. Yao, H. Chaturvedi, T. Tran, M. Chintalapati, S. Rajmohan *et al.*, "Codec: Cost-effective duration prediction system for deadline scheduling in the cloud," in *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2023, pp. 298–308.

[7] M. Ma, Y. Liu, Y. Tong, H. Li, P. Zhao, Y. Xu, H. Zhang, S. He, L. Wang, Y. Dang *et al.*, "An empirical investigation of missing data handling in cloud node failure prediction," in *Proceedings of the 30th ESEC/FSE*, 2022, pp. 1453–1464.

[8] M. Ma, Z. Yin, S. Zhang, S. Wang, C. Zheng *et al.*, "Diagnosing root causes of intermittent slow queries in cloud databases," *Proceedings of the VLDB Endowment*, vol. 13, no. 8, pp. 1176–1189, 2020.

[9] L. Li, X. Zhang, S. He, Y. Kang, H. Zhang, M. Ma, Y. Dang, Z. Xu, S. Rajmohan, Q. Lin *et al.*, "Conan: Diagnosing batch failures for cloud systems," in *In 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice*, 2023.

[10] M. M. Botezatu, I. Giurgiu, J. Bogojeska, and D. Wiesmann, "Predicting disk replacement towards reliable data centers," in *Proceedings of KDD*, 2016, pp. 39–48.

[11] J. Meza, Q. Wu *et al.*, "A large-scale study of flash memory failures in the field," in *Proceedings of SIGMETRICS*, 2015, pp. 177–190.

[12] Y. Xu, K. Sui, R. Yao, H. Zhang, Q. Lin, Y. Dang, P. Li, K. Jiang, W. Zhang, J. Lou, M. Chintalapati, and D. Zhang, "Improving service availability of cloud systems by predicting disk error," in *Proceedings of ATC*, 2018, pp. 481–494.

[13] M. M. Botezatu, I. Giurgiu, J. Bogojeska, and D. Wiesmann, "Predicting disk replacement towards reliable data centers," in *Proceedings of the 22nd ACM International Conference on KDD*, 2016, pp. 39–48.

[14] C. Zhao, M. Ma, Z. Zhong, S. Zhang, Z. Tan, X. Xiong, L. Yu, J. Feng, Y. Sun, Y. Zhang *et al.*, "Robust multimodal failure detection for microservice systems," in *Proceedings of the 29th ACM Conference on KDD*, 2023, pp. 5639–5649.

[15] J. Zhang, J. Wang, L. He, Z. Li, and P. S. Yu, "Layerwise perturbation-based adversarial training for hard drive health degree prediction," in *Proceedings of ICDM*, 2018, pp. 1428–1433.

[16] C. Xu, G. Wang, X. Liu, D. Guo, and T. Liu, "Health status assessment and failure prediction for hard drives with recurrent neural networks," in *IEEE Transactions on Computers*, vol. 65, no. 11, 2016, pp. 3502–3508.

[17] C. Luo, P. Zhao, B. Qiao, Y. Wu, H. Zhang, W. Wu, W. Lu, Y. Dang, S. Rajmohan, Q. Lin, and D. Zhang, "NTAM: neighborhood-temporal attention model for disk failure prediction in cloud platforms," in *Proceedings of WWW*, 2021, pp. 1181–1191.

[18] X. Sun, K. Chakrabarty, R. Huang, Y. Chen, B. Zhao, H. Cao, Y. Han, X. Liang, and L. Jiang, "System-level hardware failure prediction using deep learning," in *Proceedings of DAC*, 2019, p. 20.

[19] M. D. Dangut, Z. Skaf, and I. K. Jennions, "An integrated machine learning model for aircraft components rare failure prognostics with log-based dataset," *ISA transactions*, vol. 113, pp. 127–139, 2021.

[20] V. Ganatra, A. Parayil, S. Ghosh, Y. Kang, M. Ma, C. Bansal, S. Nath, and J. Mace, "Detection is better than cure: A cloud incidents perspective," in *Proceedings of the 31st ESEC/FSE*, 2023, pp. 1891–1902.

[21] Y. Liu, H. Yang, P. Zhao, M. Ma, C. Wen, H. Zhang, C. Luo, Q. Lin *et al.*, "Multi-task hierarchical classification for disk failure prediction in online service systems," in *Proceedings of the 28th ACM Conference on KDD*, 2022, pp. 3438–3446.

[22] R. Caruana, "Multitask learning," in *Machine learning*, vol. 28, no. 1, 1997, pp. 41–75.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[24] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[27] M. Ma, S. Zhang, D. Pei, X. Huang, and H. Dai, "Robust and rapid adaption for concept drift in software system anomaly detection," in *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2018, pp. 13–24.

[28] Z. Wang, C. Pei, M. Ma, X. Wang, Z. Li, D. Pei, S. Rajmohan, D. Zhang, Q. Lin, H. Zhang *et al.*, "Revisiting vae for unsupervised time series anomaly detection: A frequency perspective," in *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 3096–3105.

[29] Y. Chen, C. Zhang, M. Ma, Y. Liu, R. Ding, B. Li, S. He *et al.*, "Imdiffusion: Imputed diffusion models for multivariate time series anomaly detection," *VLDB Endowment*, vol. 17, no. 3, p. 359–372, 2023.

[30] Z. Yu, C. Pei, X. Wang, M. Ma, C. Bansal *et al.*, "Pre-trained kpi anomaly detection model through disentangled transformer," in *Proceedings of the 30th ACM SIGKDD Conference on KDD*, 2024.

[31] M. Ma, S. Zhang, J. Chen, J. Xu, H. Li, Y. Lin, X. Nie, B. Zhou, Y. Wang, and D. Pei, "Jump-starting multivariate time series anomaly detection for online service systems," in *USENIX*, 2021, pp. 413–426.

[32] Z. Zeng, Y. Zhang, Y. Xu, M. Ma, B. Qiao, W. Zou, Q. Chen, M. Zhang *et al.*, "Traceark: Towards actionable performance anomaly alerting for online service systems," in *the 45th ICSE-SEIP*, 2023, pp. 258–269.

[33] Z. Xie, S. Zhang, Y. Geng, Y. Zhang, M. Ma, X. Nie, Z. Yao *et al.*, "Microservice root cause analysis with limited observability through intervention recognition in the latent space," in *Proceedings of the 30th ACM SIGKDD Conference on KDD*, 2024.

[34] X. Zhang, S. Ghosh, C. Bansal, R. Wang, M. Ma, Y. Kang, and S. Rajmohan, "Automated root causing of cloud incidents using in-context learning with gpt-4," in *Companion Proceedings of the 32nd ACM International Conference on FSE*, 2024, pp. 266–277.

[35] Y. Chen, H. Xie, M. Ma, Y. Kang, X. Gao, L. Shi, Y. Cao, X. Gao, H. Fan, M. Wen *et al.*, "Automatic root cause analysis via large language models for cloud incidents," in *Proceedings of the Nineteenth European Conference on Computer Systems*, 2024, pp. 674–688.

[36] Q. Lin, T. Li, P. Zhao, Y. Liu, M. Ma, L. Zheng, M. Chintalapati, B. Liu, P. Wang, H. Zhang *et al.*, "Edits: An easy-to-difficult training strategy for cloud failure prediction," in *Companion Proceedings of the ACM Web Conference 2023*, 2023, pp. 371–375.

[37] J. Gao, H. Wang, and H. Shen, "Task failure prediction in cloud data centers using deep learning," *IEEE transactions on services computing*, vol. 15, no. 3, pp. 1411–1422, 2020.

[38] S. L. Hyland, M. Faltys, M. Hüser, X. Lyu, T. Gumbsch, C. Esteban, C. Bock, M. Horn, M. Moor, B. Rieck *et al.*, "Early prediction of circulatory failure in the intensive care unit using machine learning," *Nature medicine*, vol. 26, no. 3, pp. 364–373, 2020.

[39] J. Leukel, J. González, and M. Riekert, "Adoption of machine learning technology for failure prediction in industrial maintenance: A systematic review," *Journal of Manufacturing Systems*, vol. 61, pp. 87–96, 2021.

[40] S. B. Jabeur, C. Gharib *et al.*, "Catboost model and artificial intelligence techniques for corporate failure prediction," *Technological Forecasting and Social Change*, vol. 166, p. 120658, 2021.

[41] N. H. An and D. T. Anh, "Comparison of strategies for multi-step-ahead prediction of time series using neural network," in *2015 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE, 2015, pp. 142–149.

[42] Y. Lu, Z. Tian, R. Zhou, and W. Liu, "Multi-step-ahead prediction of thermal load in regional energy system using deep learning method," *Energy and Buildings*, vol. 233, p. 110658, 2021.