DEMYSTIFYING DEEP LEARNING IN PREDICTIVE MONITORING FOR CLOUD-NATIVE SLOS

2023 IEEE 16TH INTERNATIONAL CONFERENCE ON CLOUD COMPUTING (CLOUD)

Published in , July 2023

NAME:HABIBUN NABI HEMEL
ID: 22241042

Authors

Andrea Morichetta and Víctor Casamayor Pujol are researchers in **Distributed Systems Group** from TU Wien, Austria, while Thomas Pusztai and Philipp Raith Schahram Dustdar are also affiliated with TU Wien. Deepak Vij and Ying Xiong Zhaobo Zhang are associated with **Futurewei Technologies** in Santa Clara, CA, USA.

ANDREA MORICHETTA

DISTRIBUTED SYSTEMS GROUP, TU WIEN
VIENNA, AUSTRIA

THOMAS PUSZTAI

PHILIPP RAITH

SCHAHRAM DUSTDAR

DISTRIBUTED SYSTEMS GROUP, TU WIEN
VIENNA, AUSTRIA

DEEPAK VIJ

YING XIONG

ZHAOBO ZHANG

FUTUREWEI TECHNOLOGIES, INC.
SANTA CLARA, CA, USA

VÍCTOR CASAMAYOR PUJOL

STEFAN NASTIC

Overview

STORY LINE

ANGRY BOSS

WHY?

WHAT?

HELPLESS ME

HOW?

INTRODUCTION

- this paper talks about making cloud computing systems work better by focusing on high-level goals called Service-Level Objectives (SLOs). Instead of just fixing problems after they happen, they want to predict and prevent issues before they occur.
- Neural networks understand the relationship between low-level system details (e.g., CPU usage) and high-level goals(SLOs)(e.g., efficiency, cost-effectiveness).
- They even talked about putting models into a framework called Polaris, making it simple for developers to use them to manage cloud systems better.
- Overall, they want to help make cloud computing smoother and more efficient by using smart computer models to predict and prevent problems before they happen.

SYSTEM RELIABITY

SLA SLO

SLI

SLA

"I WILL deliver food on table in 20 minutes"

We have to prepare EACH dish in 10 minutes!!

SERVICE LEVEL OBJECTIVE

SLA - SLO = Error Budget

20 mins - 10 mins = 10 mins

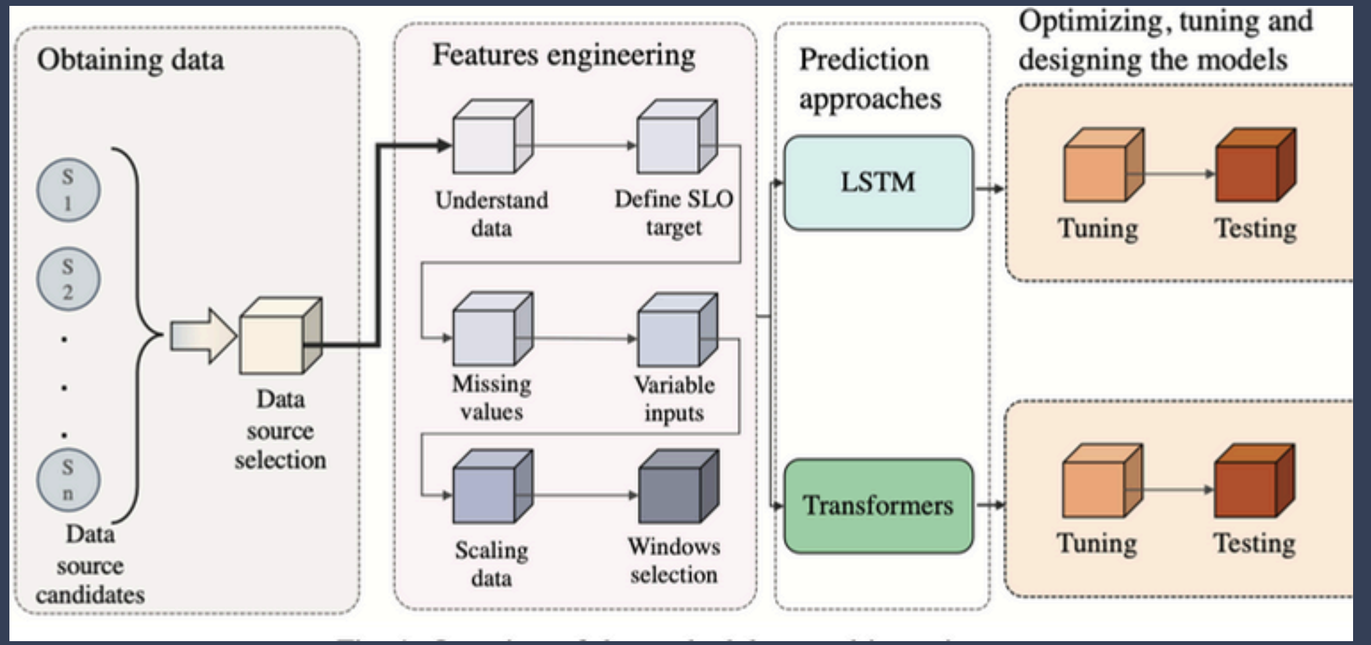
SERVICE LEVEL INDICATOR

How many dishes were delivered on target?

SLA - Realistic
SLO - Achievable
SLI - Accurate

METHODOLOGY

- GETTING THE DATA
- PREPARING THE DATA
- DESIGNING PREDICTION MODELS
- OPTIMIZING THE MODELS



the methodology involves collecting real data, preparing it for analysis, designing and testing neural network models, optimizing these models to achieve the best possible predictions.

METHODOLOGY

GETTING THE DATA

- Real data needed for understanding cloud system usage.
- Data should span at least a week and include CPU, memory, and disk usage.
- It's important that the data is well-documented and comes from actual systems, not made-up scenarios.
- Chosen dataset: Google Cluster Workload Datasets from 2011 because it meets their criteria and provides a good representation of real-world cloud usage.

TABLE I: Datasets comparison.

Dataset	Size	Resources usage infos	Machine infos	Year	Documentation
Google 2011	41GB (zip)	CPU, memory, and disk	Yes	2011	Github, community and papers
Google 2019	2.4TB	CPU, memory, and disk	Yes	2019	Github, community and papers
Azure	235GB	CPU and memory	No	2019	Github, papers
Bitbrains	280MB (in total)	CPU, memory, network and disk	No	2013	Papers
Alibaba	280 GB	CPU, memory, network and disk	Yes	2018	Github, papers

We target three resource metrics for each job to extract the *efficiency*, namely the CPU rate (CPU), the canonical memory usage (mem), and the disk usage ($disk$). We can define the total *efficiency* for a job j with m active tasks k at time t with the Equation 2; notice that we divide by three as we

PREPARING THE DATA

- This involves selecting the most relevant metrics, such as CPU rate, memory usage, and disk usage.
- They define a high-level Service Level Objective (SLO) metric called "efficiency" & cost-effectiveness)based on resource usage, which they aim to predict using the low-level metrics(.like CPU usage)
- To prepare the data, the researchers followed several steps:

Selection of Relevant Metrics, Aggregation of Data, Definition of High-Level SLO Metric

Takeaways *Considering multiple requirements for dataset selection is fundamental to developing an in-depth analysis of high-level SLO metric prediction. Open-source, well-documented data guarantees reliability and research repeatability. Therefore, we select the 2011 Google Cluster Data as it satisfies the major data requirements.*

$$Eff_t^j(CPU, mem, disk) = \frac{\sum_{k=1}^{m} eff_t^k(CPU) + eff_t^k(mem) + eff_t^k(disk)}{3m}$$

METHODOLOGY

PREDICTION MODELS

- LSTM(Long short-term memory) & Transformer
- They experiment with different configurations of these models, such as **adding convolutional layers** to LSTM and **adjusting** the number of layers in the Transformer.
- Both models are trained to predict the **efficiency** metric based on the low-level system metrics.

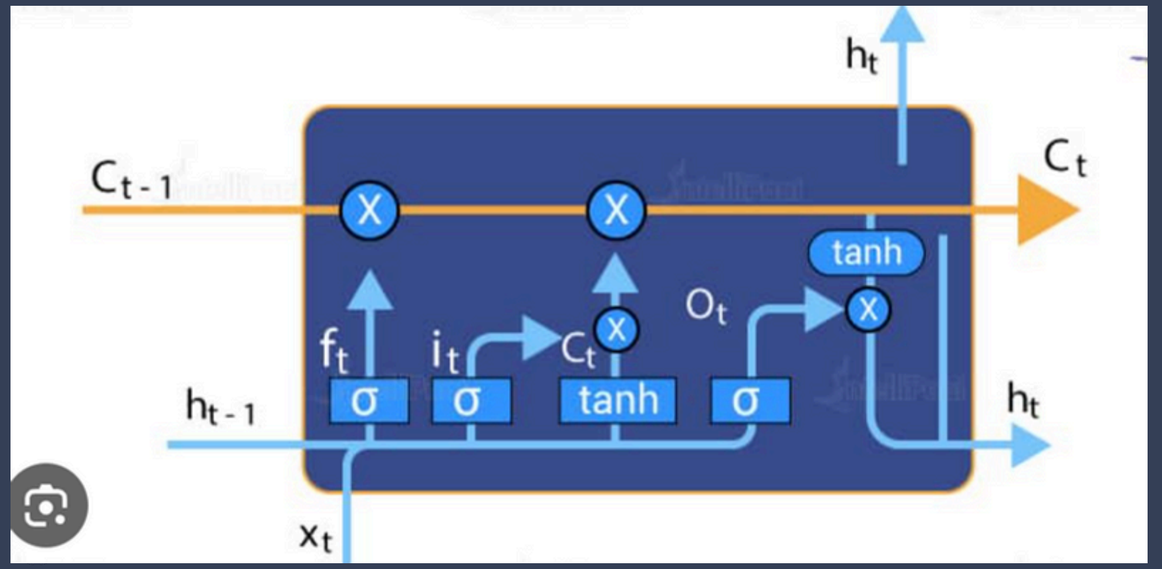


TABLE II: Hyper-parameters of the LSTM model.

Hyper-parameters		Value
Model	Neurons	50
	Dropout	0.0
	Recurrent dropout	0.0
	Activation	tanh
	Activation output	sigmoid
	CNN filters	**None**
	Bidirectional	**No**
	# LSTM layers	**1**
Learning	Epochs	100
Data related	Batch size	72
	Input window length	24

TABLE III: Hyper-parameters of the transformer model.

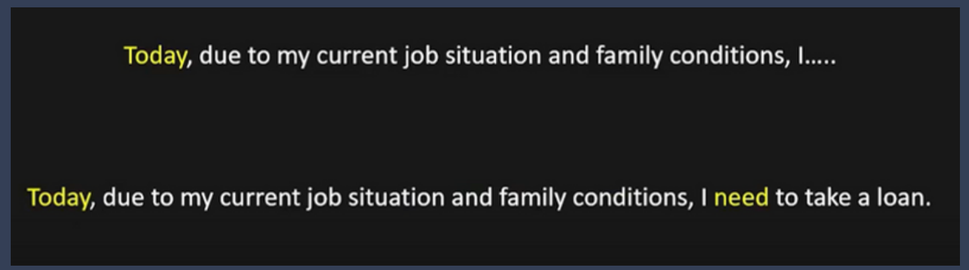
Hyper-parameters		Value
Model	**Encoder layers**	**8**
	Decoder layers	**1**
	Attention heads	**10**
	QKV internal dimension	**3**
	Model inner dimension	**8**
Learning	Learning rate	0.01
	Learning decay factor	0.99
	Epochs	30
Data related	Batch size	4
	Input window length	24

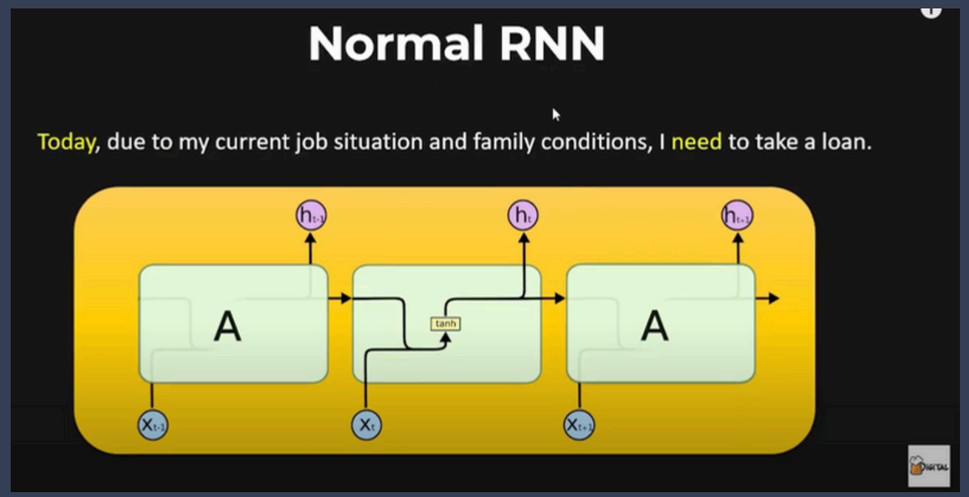
OPTIMIZING THE MODELS

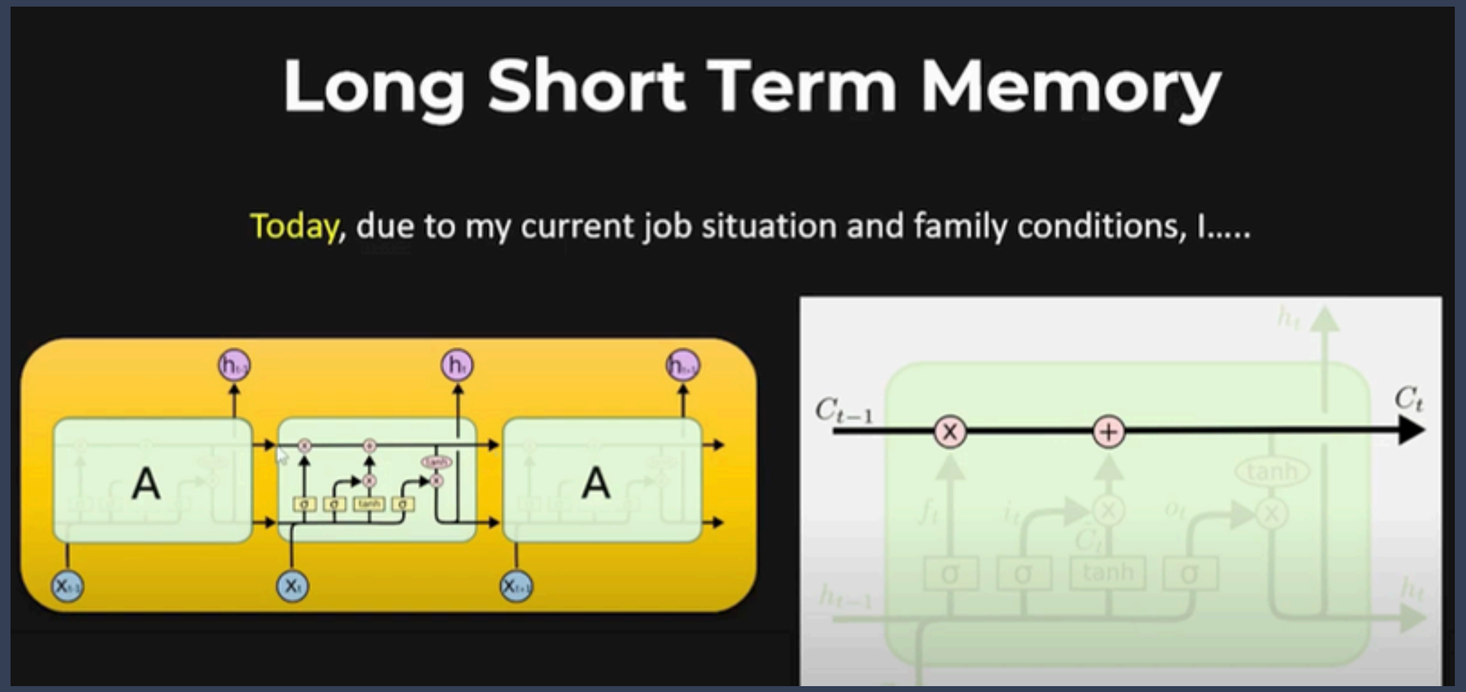
- They **fine-tune** the models to improve their performance using a method called **hyperparameter tuning.**
- Hyperparameters are settings that control how the **models learn from the data,** and tuning them can significantly impact the models' accuracy.
- They use a technique called **ASHA** (Asynchronous Successive Halving Algorithm) scheduler provided by RayTune for **efficient hyperparameter tuning.**

LSTM

- LSTM (Long Short-Term Memory): Imagine you have a special kind of calculator that **learns patterns over time**. When you give it a sequence of numbers, it can figure out what might come next. LSTM is like that calculator. It's good at remembering important things from the past and using them to predict the future. It's like having a clever friend who knows how things usually go and can guess what happens next based on that.
- They describe LSTM as a model that can learn patterns over time. It has a **special kind of memory** that helps it **remember** important things from the past. They mention that LSTM is designed to predict future trends based on the patterns it learns from past data.







TRANSFORMER

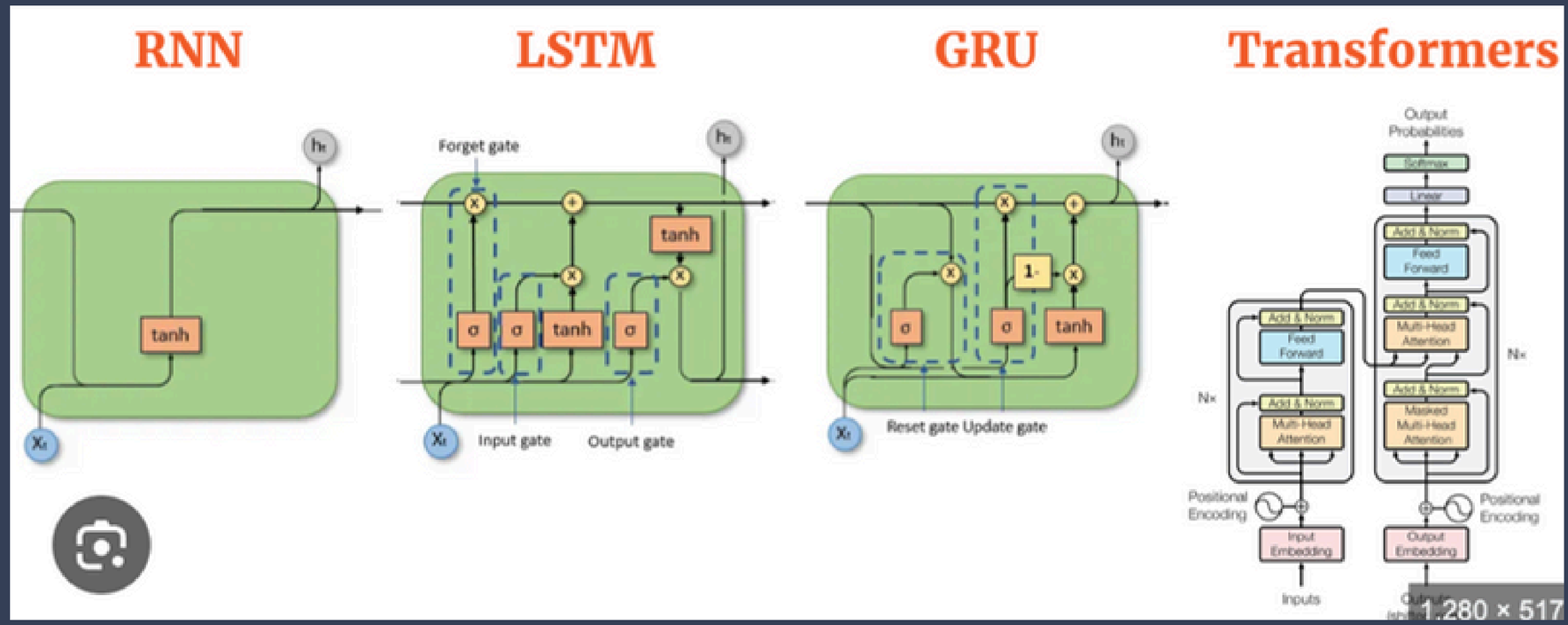
- Now, think of a big puzzle where each piece represents a different part of a story. The Transformer model is like a **super-fast puzzle solver**. It looks at all the pieces **together** and figures out how they fit to make the whole picture. It's great at understanding **complex relationships** between **different parts** of data. It's like having a superhero who can put together pieces of information really quickly to see the big picture.
- Similarly, they discuss the Transformer model as a deep learning architecture that uses attention mechanisms to understand relationships between different parts of data. They explain that it's especially good at handling complex sequences of data and can quickly piece together information to make predictions about the future.

NOTE:

the paper does not explicitly describe the workflow of the Transformer model. It primarily focuses on explaining the methodology, data preprocessing, model design (both LSTM and Transformer), and model optimization. While the paper mentions the Transformer architecture and its components, it does not provide a detailed step-by-step workflow specific to the Transformer model.

SUMMARY

- Overall, the paper provides details on how both LSTM and Transformer models function, highlighting their capabilities in learning from data and making predictions.
- So, in simple terms, LSTM is good at remembering and predicting based on past patterns, while Transformer is great at understanding and piecing together complex information to make predictions.



COMPARATIVE ANALYSIS

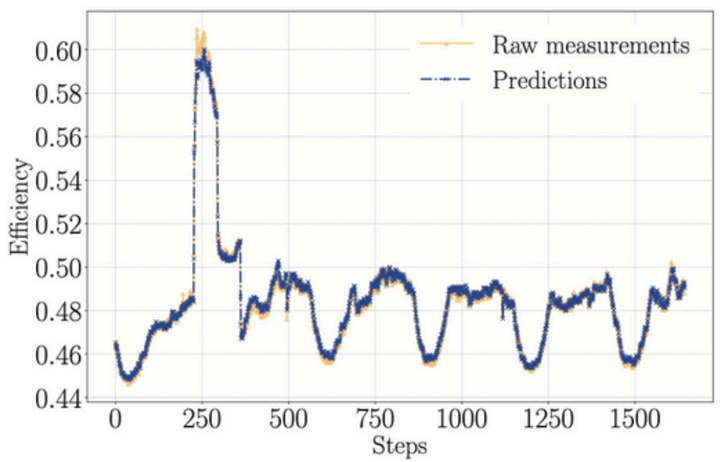
LSTM

TRANSFORMER

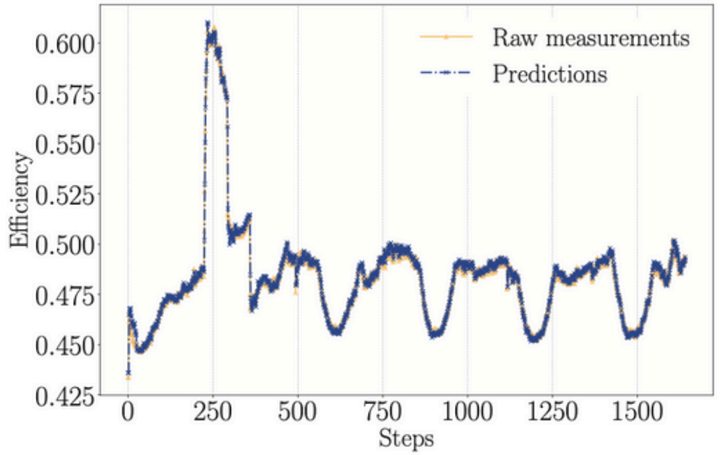
1. Evaluation on the Test Set:
   - Both the LSTM and Transformer models did a good job predicting how efficiently computer systems would work.
   - When we looked closely at the results, we saw that both models followed the efficiency patterns well.
   - However, as they tried to predict further into the future, they started to make slightly bigger mistakes.
2. Evaluation on Out-of-Distribution Data:
   - We tested the models on more than 400 different computer system scenarios.
   - The Transformer model performed much better, with errors that were ten times smaller than the LSTM model.
   - This shows that the Transformer model is better at handling new and different situations.
3. Discussion:
   - Main point: the LSTM model is more like a **quick learner** that can give decent predictions but struggles with new challenges while the Transformer model is like a super-smart computer that can **handle new situations really well**

the test set evaluation checks if the models can remember and apply what they've learned, while the evaluation on OOD data checks if the models can adapt and perform well in unfamiliar situations.

COMPARATIVE ANALYSIS



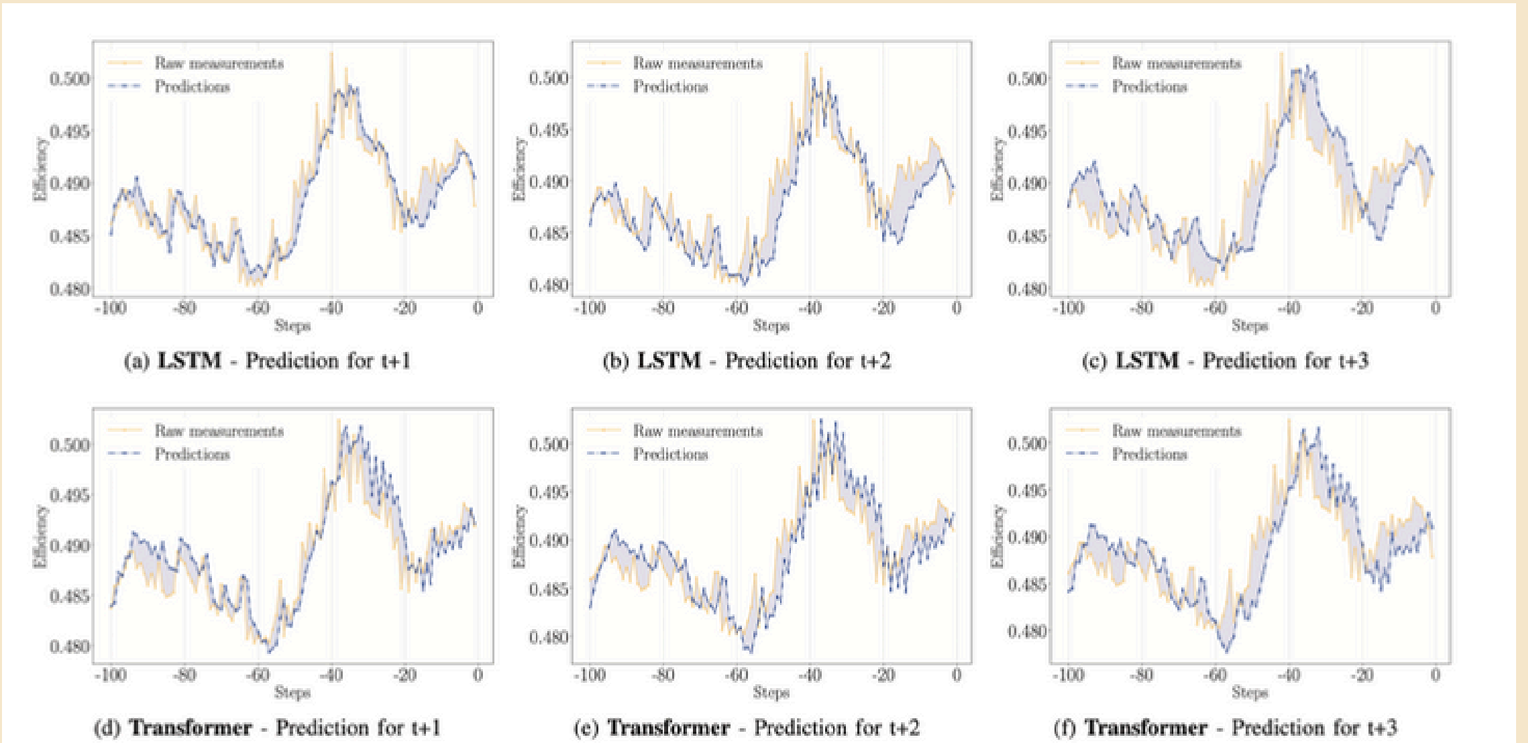
(a) **LSTM** - model performance .



(b) **Transformer** - model performance.

The graph compares the actual **efficiency** of a system with what the models predict. **Both models seem to do a good job at predicting efficiency accurately.** The reason for this accuracy could be because the models have been trained well on the data and have learned to make accurate predictions.

This graph zooms in on the predictions made by the models for a **specific period**. It shows that **both models are good** at following the efficiency trend over time, but **one model may react more sharply to sudden changes,** while the other might be smoother in its predictions. This difference in behavior could be due to the nature of the algorithms used in each model. For example, one model might be more sensitive to short-term fluctuations, while the other might be better at capturing longer-term trends.

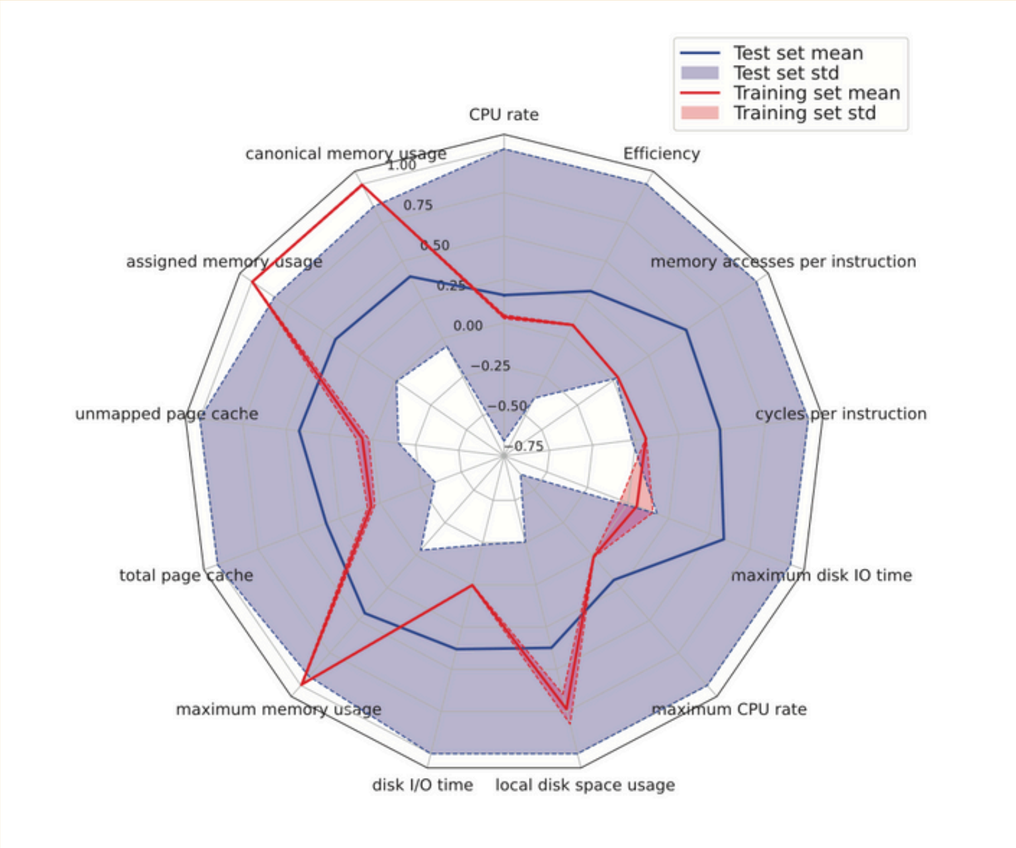


(a) **LSTM** - Prediction for t+1 (b) **LSTM** - Prediction for t+2 (c) **LSTM** - Prediction for t+3

(d) **Transformer** - Prediction for t+1 (e) **Transformer** - Prediction for t+2 (f) **Transformer** - Prediction for t+3

Fig. 6: Last 100 steps of the test series for LSTM and Transformer. The orange dashed line shows the target *efficiency*, while the blue dash-dotted line the predicted values. The light blue area highlights the gap between the target and predicted values.

COMPARATIVE ANALYSIS



In this graph, we compare the behavior of the test data (the blue line) with the training data (the red line) used to train the models. The test data behaves differently from the training data, which is expected since the models were trained on a specific dataset. This difference in behavior could be due to variations in the real-world environment that the models weren't exposed to during training.

This graph shows the performance of the models in predicting efficiency across different scenarios. One model performs better than the other, especially when dealing with new or unfamiliar data. This difference in performance could be due to the architecture or complexity of the models. For example, one model might be better at generalizing from the training data to new data, while the other might struggle with this generalization.

What is the RMSE score in machine learning?
It measures the average difference between values predicted by a model and the actual values.

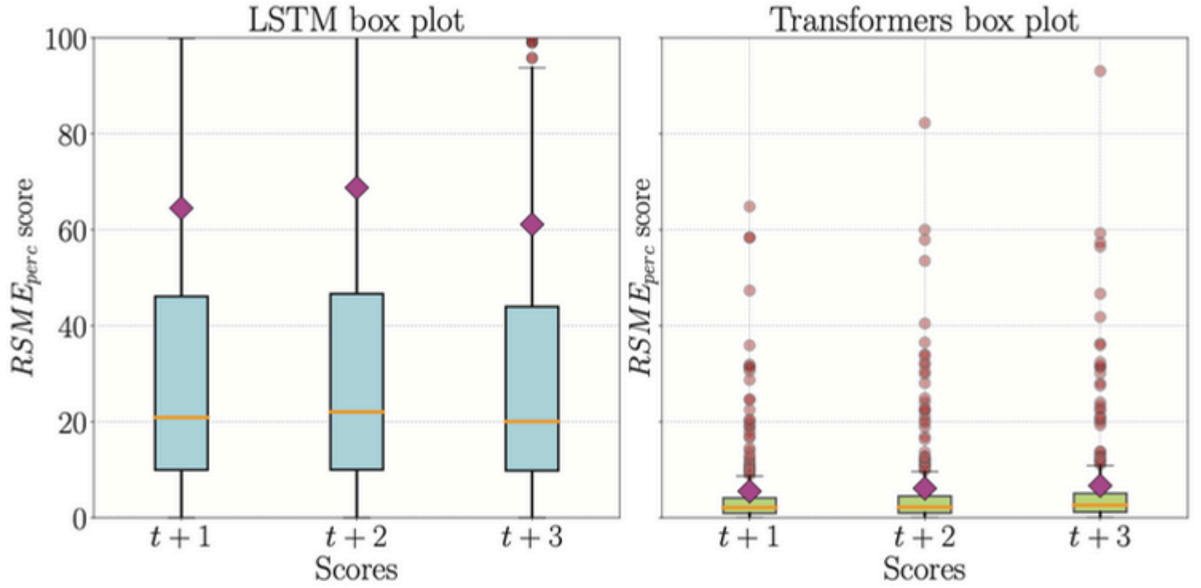


Fig. 9: $RMSE_{perc}$ results for LSTM and Transformer over *out-of-sample (OOS)* workload.

INTEGRATION WITH POLARIS SLO CLOUD

POLARIS

- Framework for simplifying real-time cloud system management, offering tools and pre-built models.
- It simplifies the process of using ML models for managing cloud systems.
- Developers can focus on defining SLOs and optimizing their applications, while Polaris takes care of predicting and managing violations automatically.
- Polaris SLO Cloud is like a smart helper that uses machine learning to predict how well a computer system will perform, making it easier for developers to keep their applications running smoothly without worrying about technical details.

CLOUDSIM

- Simulation framework for modeling cloud infrastructures, enabling scenario analysis and performance study.

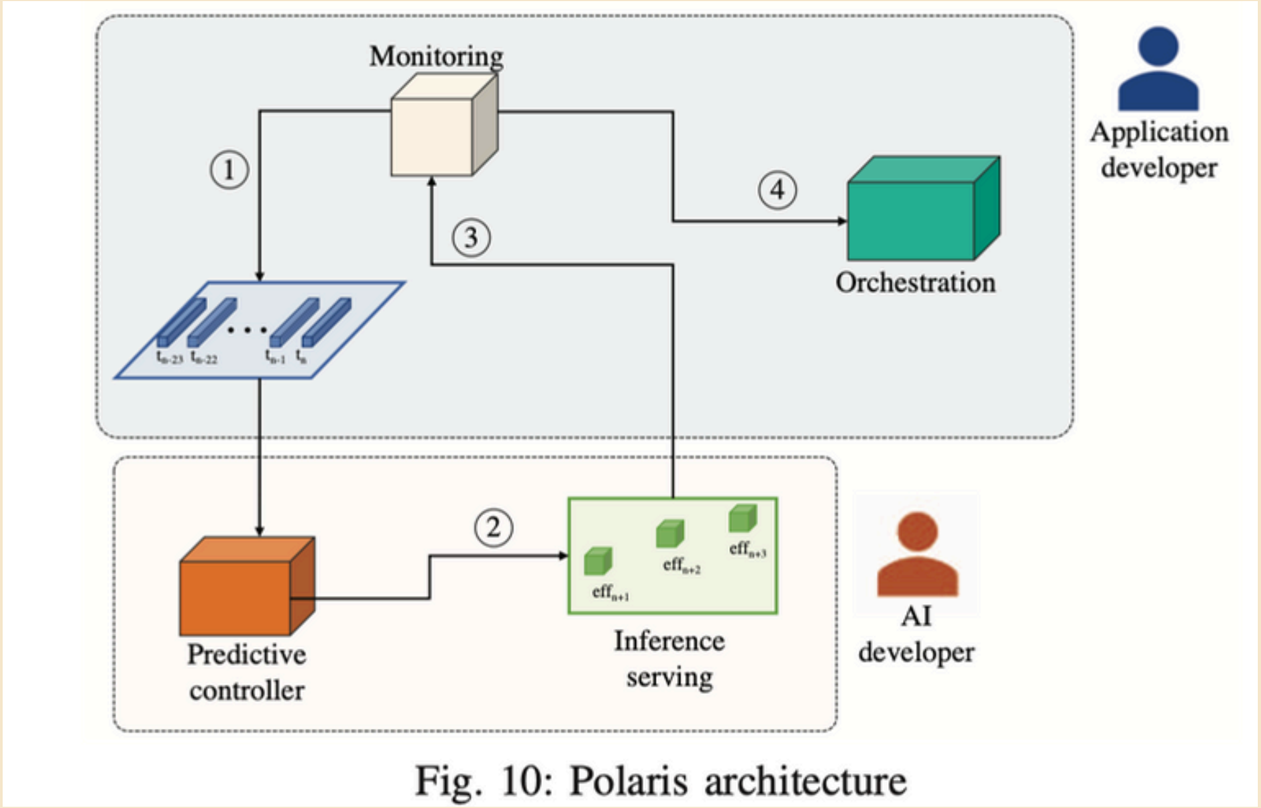


Fig. 10: Polaris architecture

Difference: **Polaris focuses on real-time management, while CloudSim is for simulating cloud environments.**

While they didn't explicitly mention using the Polaris SLO Cloud in the findings presented here, it's possible that they used it as part of their research process or plan to integrate it into their future work. The focus here is more on the comparison of different models (Long Short-Term Memory and Transformer) and their performance in predicting performance goals (SLOs) for cloud servers. The mention of the Polaris SLO Cloud in the future direction suggests that they may explore its use in their ongoing research.

RELATED WORK

Traditional Methods:

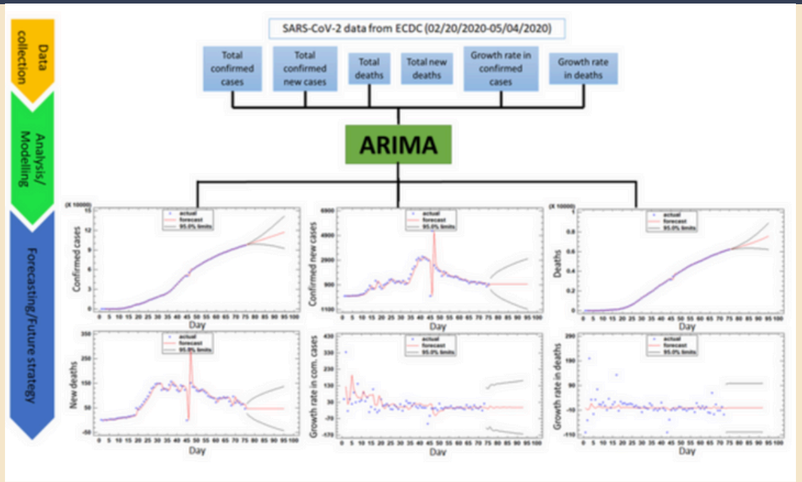
- Old-school methods like ARIMA used for cloud forecasting.

Machine Learning Methods:

- Advanced techniques like LSTM models used for accurate predictions.

Focus and Limitations:

- Previous studies often focused on low-level metrics like CPU usage.
- They sometimes overlooked higher-level objectives like SLOs.
- Also, they didn't always consider how **well models would work with new data or in real cloud systems.**



An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that uses time series data to either better understand the data set or to predict future trends. A statistical model is autoregressive if it predicts future values based on past values.

Overall Importance:

- While helpful, there's still room to improve cloud forecasting.
- Future research should consider a **broader range of metrics** and better integration into cloud management systems.
- In simple terms, researchers are using fancy math and computer techniques to predict how cloud servers will behave. But sometimes they forget to look at the big picture or think about **how these predictions fit into the overall system.** There's still a lot to learn and improve!

CONCLUSION

In simple terms, here's what we've learned from this study:

What We Did:

- Used cloud data to predict server performance goals (SLOs).(like efficiency .....
- Compared two computer models, Long Short-Term Memory and Transformer, for prediction accuracy.

What We Found:

- Tested models with different data types, finding the **Transformer model better** with new data.
- Both models have strengths and weaknesses.

What's Next:

- Improve predictions for **long term future** performance .
- **Test models in real cloud systems, like Polaris SLO Cloud.**
- Aim to optimize cloud server management for better efficiency.

So, in a nutshell, we're trying to make cloud servers smarter by predicting how they'll perform and using that information to manage them better in the future.

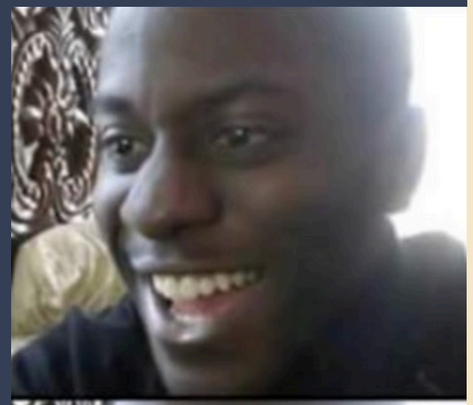
RECAP

- STORY(WHY AND WHAT)
- SLO
- FUTURE PREDICTION(HOW)
- LSTM
- TRANSFORMER
- COMPARE
- FRAMEWORK(REAL-LIFE)
- FUTURE WORKS

THANK YOU

ANY QUESTIONS?

No Question



Hemel, I have a Question?