LOSEC: Local Semantic Capture Empowered Large Time Series Model for IoT-Enabled Data Centers

Yu Sun, *Member, IEEE*, Haibo Zhou, *Senior Member, IEEE*, Bo Cheng, *Member, IEEE*, Jinan Li, *Member, IEEE*, Jianzhe Xue, *Member, IEEE*, Tianqi Zhang, *Member, IEEE*, and Yunting Xu, *Member, IEEE*

***Abstract*—Deep learning methods for accurately predicting data center status, which are essential for addressing the exponential growth of energy consumption, have gained significant attention, driven by the vast amounts of data collected through the advancement of Internet of Things (IoT) technologies. However, conventional small models often face data scarcity issues in practical deployment. While large models show promise in addressing this challenge, they encounter obstacles such as multivariate tasks, computational intensity, and ineffective information capture. Moreover, their applications in data centers remain largely unexplored. In this paper, we investigate local semantic capture empowered large model for multivariate time series forecasting in IoT-enabled data centers. We first introduce time series tasks within data centers and propose the Point Lag (Plag)-Llama framework with the Lag-Llama backbone to support zero-shot forecasting and fine-tuning for multivariate point time series forecasting. To address computational intensity and enhance the capabilities of multivariate forecasting, we propose the Local Semantic Capture (LOSEC) for adapter fine-tuning, which captures local semantic information across time and channel dimensions alternately with low-complexity. Specifically, time series are patched into tokens, and channels are clustered together, forming local semantic information that can be captured more effectively. Extensive experiments demonstrate that Plag-Llama exhibits superior zero-shot capability and that the LOSEC empowered adapter fine-tuning achieves state-of-the-art performance on real-world datasets collected from data centers, with ablation studies further validating the effectiveness of each module within the proposed models.**

***Index Terms*—Large model, multivariate time series, parameter-efficient fine-tuning, Internet of Things, data center.**

I. Introduction

DATA centers hold a crucial position in global energy consumption and carbon emissions, with their importance projected to escalate exponentially, especially with the widespread adoption of large-scale models like ChatGPT and Llama [2], [3]. Concurrently, governments and data center operators vigorously enact policies and initiatives targeted at curbing energy consumption [4]. The optimization of cooling systems has garnered significant attention, given their substantial share of energy consumption in data centers. Precise modeling of data centers plays a pivotal role in this context, serving as the key foundation for subsequent optimization algorithms [5]. Recently, the evolution of Internet of Things (IoT) techniques has facilitated data-driven modeling [6]–[8], showcasing superiority in providing accurate descriptions of future data center status compared to traditional methods [4], [5], [9]. However, a critical aspect remains unaddressed in these works: multivariate time series forecasting. Despite their successes, these conventional small models encounter challenges related to data scarcity during practical deployment, requiring sufficient data accumulation after all IoT devices are operational, thereby incurring significant time costs.

With the success of large models, also referred to as foundation models, across various domains [10], [11], recent studies have also explored their significant capabilities in time series forecasting, including Large Language Models (LLMs) based methods [12], [13], foundation models tailored for time series [14], [15]. These models demonstrate superior zero-shot and few-shot abilities, which are anticipated to mitigate the data scarcity challenge, thereby are promising solution to facilitate swift deployment in data centers. However, these methods encounter various challenges to different extents. For example, prompt-based LLMs [12] may struggle with multivariate time series, and the high cost of LLMs presents a barrier to widespread adoption. Furthermore, TimeGPT [14], exclusively available via API, is not suitable for deployment in production environments. Additionally, there has been no prior investigation into the realm of data center applications. There are two approaches to building large models for multivariate time series forecasting in data centers: training from scratch and building upon well-established large models. The massive number of parameters in large models makes it challenging to collect sufficient data for training from scratch in data centers and renders the process computationally expensive [16]. In contrast, building upon existing large models—i.e., fine-tuning—offers a more computationally efficient, less data-intensive approach that fully leverages the capabilities of pre-trained models and yields comparable performance. As a result, the latter approach is preferred. Specifically, we choose to build upon Lag-Llama [15], a state-of-the-art (SOTA) time series foundation model designed for univariate probabilistic forecasting. However, it faces several challenges when applied to multivariate time series in data centers. These challenges include: 1) transferring from probabilistic to point forecasting, 2) empowering the multivariate forecasting capabilities of Lag-Llama, and 3) addressing computational intensity and overfitting issues encountered during training.

Part of this work was accepted at the IEEE Global Communications Conference (GLOBECOM) 2024 [1], to be published (*Corresponding author: Haibo Zhou.*).

Y. Sun, H. Zhou, B. Cheng, J. Xue, T. Zhang and Y. Xu are with the School of Electronic Science and Engineering, Nanjing University, Nanjing, China, 210023 (e-mail: yusun@smail.nju.edu.cn, haibozhou@nju.edu.cn, bocheng@smail.nju.edu.cn, jianzhexue@smail.nju.edu.cn, tianqizhang@smail.nju.edu.cn, yuntingxu@smail.nju.edu.cn).

J. Li is with the Guangdong Branch, China Unicom Group Co., Ltd., Guangzhou, China, 510627 (e-mail: eljn@foxmail.com).

These challenges are expected to be addressed through specific fine-tuning methods. However, with the advent and rapid scaling of large models, the computational demands of full fine-tuning have also become prohibitive for most devices and scenarios. Parameter-Efficient Fine-Tuning (PEFT) [17], which modifies only a subset of existing parameters or adds new ones, aims to reduce computational requirements while preserving learned capabilities in large models, enabling them to generalize to new tasks. Nonetheless, existing PEFT methods fail to capture new correlations beyond those encoded in the original models [18], leaving core challenges in Lag-Llama unresolved. Furthermore, many PEFT methods are constrained by simplistic feed-forward [19] or attention-based designs [20] that lack additional information-capturing capabilities, rendering them inadequate for empowering Lag-Llama in multivariate forecasting. Similarly, in multivariate time series tasks, the effectiveness of current channel-dependent and channel-independent attention mechanisms [21], [22], as well as transformer-based architectures [23], remains a subject of debate, warranting further investigation to address their limitations. Therefore, developing efficient architectures that can effectively capture essential information during adapter fine-tuning and address the challenges posed by Lag-Llama, thereby empowering large models for multivariate time series forecasting in data centers, remains a significant open problem.

In this paper, we investigate large models empowered by local semantic capture for multivariate time series forecasting in IoT-enabled data centers, effectively capturing local semantic information across time and channel dimensions, achieving SOTA performance, and addressing challenges such as data scarcity in conventional small models. We first introduce the multivariate time series forecasting problem specific to data centers, alongside the large model, Lag-Llama. Subsequently, we propose the point Lag (Plag)-Llama framework, adapted from Lag-Llama, to support zero-shot forecasting and fine-tuning for multivariate point time series forecasting. Furthermore, we propose the **Lo**cal **Se**mantic **C**apture (LOSEC)-empowered adapter fine-tuning technique, which constructs and captures semantic information for both time steps and channels, information that is absent in previous studies but essential for the attention mechanism. This approach enhances the performance of large models for multivariate forecasting while reducing computational intensity. LOSEC alternates between capturing local semantic information across time and channel dimensions with low-complexity: time capture patches the series into tokens for time attention, while channel capture clusters channels for channel attention. Finally, we validate our proposed methods through extensive experiments on real-world datasets, demonstrating their superiority over SOTA approaches. The main contributions are highlighted as follows:

- We construct a multivariate time series forecasting framework for IoT-enabled data centers, employing large models to address data scarcity, facilitate rapid deployment, and achieve SOTA performance. To our knowledge, this is the first effort to investigate large models within the data center domain.
- We introduce the Plag-Llama framework, extending Lag-Llama's capabilities from probabilistic univariate to multivariate point time series forecasting, enabling zero-shot ability and full fine-tuning for new tasks. We further propose a novel **Lo**cal **Se**mantic **C**apture (LOSEC) empowered adapter fine-tuning method that reduces computational intensity while achieving SOTA performance.
- We propose LOSEC, a model that alternately constructs and captures local semantic information across time and channel dimensions with low complexity. In this model, time series are patched to form local semantic temporal information, while channels are clustered to generate local semantic channel information. This information is effectively captured through the utilization of multi-head attention mechanisms.

The remainder of this paper is organized as follows. Section III introduces the preliminaries of multivariate time series forecasting in IoT-enabled data centers and describes our adopted backbone large model. The proposed Plag-Llama architecture and the LOSEC empowered adapter fine-tuning method are detailed in Section IV. Experimental results are presented in Section V, followed by the conclusion in Section VII.

II. Related Works

In this section, we review SOTA works relevant to our research, focusing on time series forecasting in data centers, large models for time series forecasting, fine-tuning methods, and mechanisms for capturing correlations in multivariate time series, highlighting their strengths and limitations.

Recently, time series forecasting has become a key foundation for data center optimization [24], [25], with numerous studies exploring this domain. Vu et al. [5] proposed using more appropriate data-driven models by incorporating domain knowledge into their analyses. Specifically, they applied distinct deep learning algorithms tailored to specific module types within a module-wise modeling approach, predicting power, water flow, and water temperature. In contrast, Zhao et al. [4] adopted purely data-driven methods, considering temporal properties within data centers and employing a gated recurrent unit-based approach to leverage historical data for predicting power usage effectiveness. Given the importance of safety-guaranteed optimization in data center operations, Sun et al. [26] leveraged supervised autoencoders to predict cooling capacity and ensure safe operations, achieving accurate predictions with low complexity. To address multiple prediction tasks in data centers, Jiang et al. [9] proposed a model-wise, multi-task transformer-based network to ensure both accuracy and interpretability. However, these works often overlook multivariate and multi-step time series forecasting. Furthermore, conventional small models encounter data scarcity challenges and require substantial time to deploy in real data center scenarios.

With the remarkable success of large models in natural language processing, researchers are increasingly interested in seeing similar capabilities applied to time series forecasting. Recent studies in this area fall into two main categories: transferred large models from other domains [12], [13], [27] and

foundation models developed specifically for time series [14], [15], [28]. Initially, studies observed the remarkable capabilities of LLMs, leading to exploration of their application in time series forecasting. Xue et al. [12] directly employed LLMs for forecasting by transforming numerical inputs and outputs into language prompts, demonstrating superior generalization compared to traditional numerical forecasting. To apply LLMs to time series, Jin et al. [13] aligned time series and language modalities by training text prototypes with frozen LLMs, leveraging the internal capabilities of LLMs. Chang et al. [27] pursued a similar approach but proposed a deeper and more flexible fine-tuning method to transfer LLM abilities from natural language to time series, using a two-stage fine-tuning process: supervised fine-tuning for LLMs and task-specific downstream fine-tuning. Moreover, foundation models tailored for time series forecasting have emerged. The first foundation model, TimeGPT [14], based on transformer architecture and trained on 100 billion data points, exhibits high accuracy and strong zero-shot capabilities. In the same vein, Liu et al. [28] developed a GPT-style architecture trained on 1 billion time points, showing promising capabilities across diverse time series applications. Lag-Llama [15], targeting probabilistic univariate time series forecasting, has demonstrated superior performance on various time series datasets. Additional large models for time series forecasting can be found in recent surveys [29]–[31]. However, foundation models face several challenges in multivariate time series forecasting within data centers, including an inability to meet privacy requirements in production environments [14], difficulties in handling multivariate time series [12], [13], and limitations in capturing complex multivariate relationships [15], [27]. Moreover, to the best of our knowledge, no large models have been specifically designed for time series forecasting in data centers.

PEFT methods are promising for adapting large models to downstream tasks and can be categorized into addition-based, selection-based, and reparameterization-based approaches [17]. Specifically, addition-based methods augment original models with extra layers or parameters, fine-tuning only these new parameters. Selection-based methods choose a subset of parameters for fine-tuning, while reparameterization-based methods use low-rank representations to minimize the number of parameters [18]. However, except for adapter-based methods, these approaches struggle to capture new correlations beyond the original models, leaving core challenges in Lag-Llama unsolved. Existing works have explored various adapter architectures as a solution. Houlsby et al. [19] proposed inserting layers serially into original models with bottleneck architectures, yielding compact and extensible models with superior performance. Lei et al. [32] introduced conditional adapters focused on both accuracy and efficiency by skipping heavy computation, achieving faster speeds and improved accuracy. Zhao et al. [20] argue that contexts are more important than the number of parameters. They apply attention layers to adapter fine-tuning to capture conditional position correlations that were missed by previous feed-forward networks, demonstrating superior performance. However, these architectures are limited by simple feed-forward or attention-based designs that lack additional information-capturing capabilities, rendering them insufficient to enhance models like Lag-Llama for multivariate forecasting.

In multivariate time series forecasting, channel-dependent models [33]–[35] intuitively capture correlations between channels to enhance performance. However, recent advances in channel-independent models have demonstrated more superior performance due to separate attention per channel, requiring less data and reducing overfitting [22]. The latest studies [21] suggest that well-designed channel-dependent strategies can effectively capture inter-channel correlations and outperform channel-independent models. Nonetheless, the debate regarding the best method for capturing channel-dependent correlations remains unresolved. These challenges are not limited to channel attention; similar concerns extend to the transformer models themselves. Zeng et al. [23] showed that simple one-layer linear models can outperform some transformer-based SOTA models, raising questions about the true efficacy of transformers for time series forecasting. The ongoing discussion about the effectiveness of various model architectures highlights the need for designing efficient frameworks that can capture channel correlations, temporal dependencies, and other critical information in time series forecasting. This remains an open research question.

III. Preliminaries

A. Multivariate Time Series Forecasting in IoT-Enabled Data Centers

The framework for multivariate time series forecasting in IoT-enabled data centers is illustrated in Fig. 1. Leveraging advanced IoT technologies, we can monitor the real-time status of data centers by deploying diverse sensors, including those for temperature, humidity, and flow measurement. After deploying all IoT devices, continuous data streams are sensed, collected, and processed by gateway nodes before storage in the database. Over time, as sufficient data accumulates, we retrieve this information-including parameters related to chillers, cooling towers, water systems, ambient conditions[1], and more-as training data for constructing the multivariate time series forecasting module. In this work, our primary motivation is to harness the few-shot and zero-shot capabilities of foundation models to shorten the data accumulation period and accelerate the deployment of AI methods in real-world production settings. Given that accurately predicting the status of data centers forms the basis for optimization algorithms[2], we formulate this task as a multivariate time series forecasting problem, leveraging the time-dependent intrinsic characteristics and interdependencies among variables.

Let $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t, \ldots\}$ denote the multivariate time series, where $\boldsymbol{x}_t \in \mathbb{R}^C$ represents the multivariate vector with dimension C (i.e., number of variates or channels) at time t, expressed as $\boldsymbol{x}_t = [x_1, x_2, \ldots, x_C]^T$. The multivariate time series forecasting problem in data centers is defined as follows: Given an L-length context window $\mathbf{X}_{t-L+1:t} \in \mathbb{R}^{C \times L}$, the

[1]We use a water-based cooling system as an example; however, our methods are adaptable, and the data collection process is consistent across all types of heating, ventilation, and air conditioning (HVAC) systems.

[2]The research on optimization algorithms is beyond the scope of this paper.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

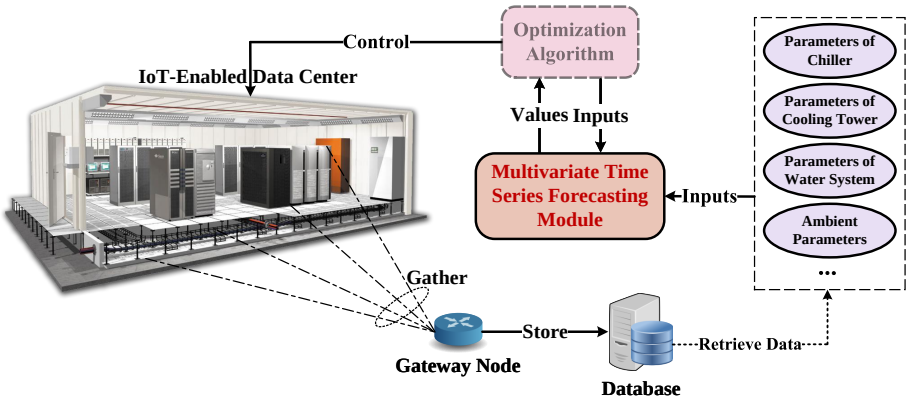

Fig. 1: A framework for multivariate time series forecasting in IoT-enabled data centers.

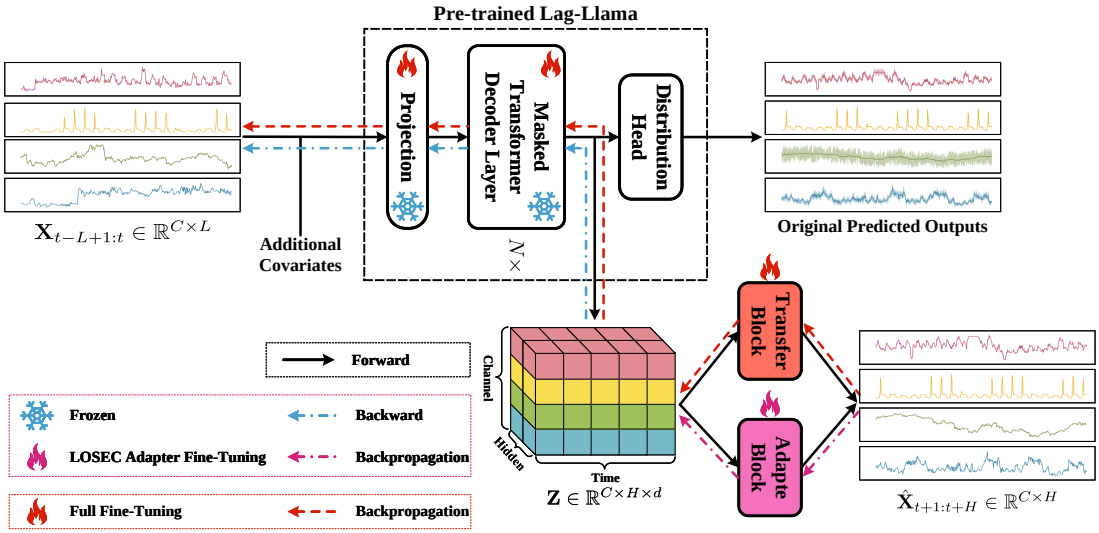

Fig. 2: Plag-Llama: Overview of the framework.

task aims to accurately predict the future H time steps, i.e., $\mathbf{X}_{t+1:t+H} \in \mathbb{R}^{C\times H}$, by minimizing the following objective:

$$\mathcal{L}\left(\mathbf{X}_{t+1:t+H}, \hat{\mathbf{X}}_{t+1:t+H}\right), \tag{1}$$

where $\mathcal{L}$ represents the loss function, and $\hat{\mathbf{X}}_{t+1:t+H}$ is derived from the multivariate time series forecasting model $\mathcal{M}$:

$$\hat{\mathbf{X}}_{t+1:t+H} = \mathcal{M}\left(\mathbf{X}_{t-L+1:t}\right). \tag{2}$$

B. Lag-Llama

Lag-Llama [15], a foundational model targeting probabilistic univariate time series forecasting, is built upon the decoder-only transformer-based LLaMA architecture, which integrates lagged features. It has demonstrated superior zero-shot ability and SOTA performance across diverse time series datasets. However, Lag-Llama is specifically designed for univariate forecasting. As highlighted in [15], its superior performance arises from processing and forecasting multivariate time series as multiple independent univariate sequences across all benchmark models. Thus, its applicability to multivariate time series forecasting remains uncertain. In our data center scenarios, the primary requirement is precise prediction of future states rather than the full probabilistic distribution, necessitating modifications to Lag-Llama. Furthermore, our experiments on real-world datasets reveal that Lag-Llama suffers from overfitting. Surprisingly, fine-tuning not only fails to improve performance but also exacerbates it. Additionally, the full fine-tuning process for Lag-Llama is computationally expensive and resource-intensive. These findings highlight the necessity of adapting foundational models like Lag-Llama for multivariate time series forecasting in data centers, with attention to both predictive accuracy and computational efficiency.

IV. Model Architecture

A. Plag-Llama: A Multivariate Time Series Point Forecasting Framework

We introduce Plag-Llama, a framework crafted for multivariate time series point forecasting that extends the capabilities of pre-trained foundation models such as Lag-Llama, with its overall framework illustrated in Fig. 2. In Lag-Llama, historical inputs $\mathbf{X}_{t-L+1:t} \in \mathbb{R}^{C\times L}$ undergo sequential processing involving a projection layer, N masked transformer

decoder layers, and a distribution head, yielding a probabilistic distribution for each time series. To adapt the probabilistic forecasting capability to point forecasting, the pre-trained Lag-Llama, excluding the distribution head, serves as the backbone of Plag-Llama. Furthermore, Plag-Llama integrates supplementary blocks, such as the transfer block and adapter block, incorporates a revised loss function, and employs fine-tuning techniques. The transfer block corresponds to the full fine-tuning method, while the LOSEC adapter block implements the adapter fine-tuning method. As shown in Fig. 2, both methods share a forward pass process, but differ in the backward pass. Specifically, full fine-tuning involves backpropagation through both the pre-trained Lag-Llama model and the transfer block, whereas LOSEC adapter fine-tuning involves backpropagation only through the adapter block and a backward pass through the pre-trained Lag-Llama. This distinction arises from their differing operational mechanisms, which are explained in the following two sections.

B. Full Fine-Tuning

Fine-tuning, an effective transfer mechanism that enables foundation models to adapt to new downstream tasks, has garnered significant attention in the research community. One of the basic methods, the full fine-tuning approach, is also employed in Lag-Llama, wherein all the parameters of the pre-trained model are tuned for new downstream tasks. However, it is important to note that the full fine-tuning strategy used here for Plag-Llama differs from that in Lag-Llama due to the distinct forecasting objectives. Specifically, in Plag-Llama, the distribution head is replaced by a transfer block. Furthermore, the negative log-likelihood loss, typically employed for probabilistic forecasting, is appropriately substituted with the Mean Squared Error (MSE) loss for point forecasting. During full fine-tuning, both the parameters of the pre-trained Lag-Llama backbone and the transfer block are updated. The entire process, including forward passes, backpropagation, and module update status, is illustrated in Fig. 2.

Transfer blocks can be implemented in various ways. In this work, we propose an intuitive and zero-shot block that leverages the average operation across feature dimensions:

$$\hat{\mathbf{X}}_{t+1:t+H} = \text{Average}\left(\mathbf{Z}_{t+1:t+H}\right), \tag{3}$$

where $\mathbf{Z}_{t+1:t+H} \in \mathbb{R}^{C\times H\times d}$ denotes the output of the backbone model, with d representing the feature (hidden) dimension. The averaging operation is performed along the feature dimension, resulting in $\hat{\mathbf{X}}_{t+1:t+H}$, which directly corresponds to the prediction values. The introduction of the feature-averaging trick is primarily grounded in the consideration of two key factors. Firstly, the foundation model excels in its zero-shot capability, which can be reacquired through this straightforward method. Secondly, it is essential to acknowledge that a more intricate structure might negatively impact performance during full fine-tuning, given that such a simple trick is also susceptible to overfitting. In the context of Plag-Llama, the update rule for full fine-tuning can be succinctly described as follows:

$$(\boldsymbol{\omega}_{i+1}, \boldsymbol{\theta}_{i+1}) = (\boldsymbol{\omega}_i, \boldsymbol{\theta}_i) - \alpha\nabla\mathcal{L}\left(\boldsymbol{\omega}_i, \boldsymbol{\theta}_i\right), \tag{4}$$

where i denotes the training step, α represents the learning rate, $\boldsymbol{\omega}$ and $\boldsymbol{\theta}$ correspond to the parameters of the backbone and transfer block, respectively, and $\mathcal{L}\left(\boldsymbol{\omega}, \boldsymbol{\theta}\right)$ denotes the loss function.

C. Local Semantic Capture Empowered Adapter Fine-Tuning

In the full fine-tuning approach, all parameters, $\boldsymbol{\omega}$ and $\boldsymbol{\theta}$, are adjusted for each task, leading to significant computational and resource demands. Furthermore, although techniques such as early stopping and random sampling are employed, full fine-tuning remains vulnerable to overfitting, as demonstrated in prior work [36] and corroborated by our experimental results in Section V. Adapter fine-tuning offers a promising alternative to address these challenges. However, existing studies [19], [20], [32], [37] fail to capture new correlations beyond those established by the original models and inadequately address channel correlations, temporal dependencies, and other critical factors. These limitations lead to significant performance degradation. To address these issues, we propose a novel model, LOSEC, designed to capture local semantic information, overcome the limitations of foundation models, and achieve SOTA performance.

One of the primary considerations in adapter fine-tuning is the placement of the inserted module, which aims to address the challenges posed by Lag-Llama. These challenges include the transition from univariate to multivariate forecasting, as well as from probabilistic to point forecasting, along with issues of computational intensity and overfitting. Fortunately, the adapter inherently mitigates concerns related to point forecasting and overfitting. However, effective multivariate forecasting requires a well-designed strategy for capturing inter-channel dependencies and other essential information, which necessitates significant information flow and may lead to increased computational overhead. Thus, achieving well-balanced trade-offs is crucial for optimal performance. Consequently, we strategically position the adapter module after the masked transformer decoder layer, as illustrated in Fig. 2. This placement ensures that informative channel, time, and feature dimensions are preserved in the outputs.

1) ***Local Semantic Capture****:* To empower Lag-Llama with the ability to perform multivariate forecasting, we seek to understand the correlations between different channels. However, simple attention-based methods struggle to capture this information due to irrelevant and even interfering data among channels [21]. Original attention mechanisms are designed to capture correlations between words in a sentence; however, individual channels in a multivariate time series are more analogous to letters in a sentence, and therefore do not inherently carry semantic meanings. Moreover, this issue extends beyond capturing correlations between channels, affecting the temporal relationships between time steps as well [22]. To address this, we propose the LOSEC model, which constructs and extracts semantic information for both the channel and time dimensions. In this context, certain individuals (channels or time steps) are grouped together to form meaningful semantic units, which we refer to as "local semantics" due to their localized nature. To efficiently capture local semantic

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

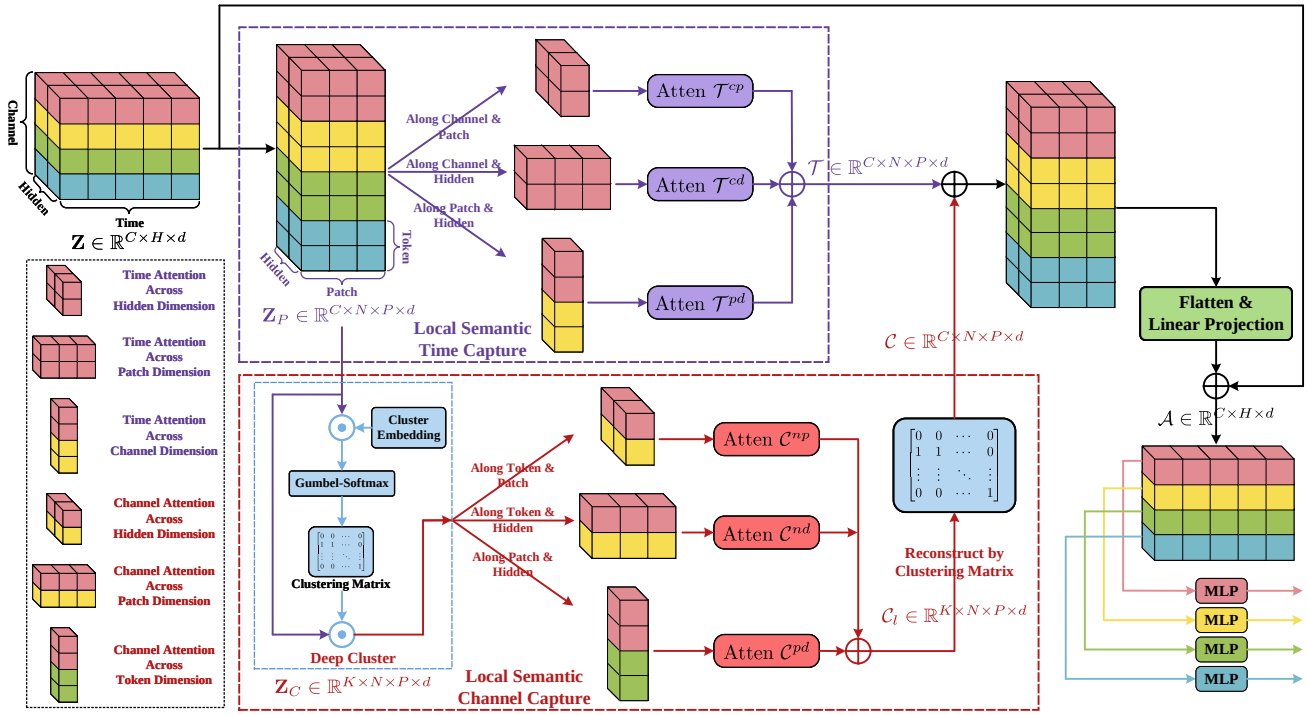

Fig. 3: Architecture of LOSEC: Alternately constructing and capturing local semantic information across time and channel dimensions.

information while maintaining complexity, we first extract local semantics along the time dimension, followed by the channel dimension. This results in two distinct processes: local semantic time capture and local semantic channel capture. This ensures that the attention mechanism can effectively perform its role in capturing these relationships. These captures across the two dimensions are key components of LOSEC. The design philosophy behind these modules is elaborated as follows:

- **Local Semantic Channel Capture Module:** Analogous to semantics in natural language, we treat channels as letters in a sentence to construct local semantic information by organizing these "letters" (channels) into words. In multivariate time series, some channels exhibit similar variations while others are irrelevant; therefore, an intuitive approach is to assemble similar channels together. Additionally, since channels are unordered, they can be organized arbitrarily. As a result, we cluster channels to construct local semantic channel information within this module. By adopting this local semantic approach, the attention mechanism can operate at the "word" level (i.e., clustered channels), thereby more effectively distinguishing relevant channels from irrelevant ones, reducing noise and interference, and facilitating the identification of key features. This approach also results in a low-complexity design.
- **Local Semantic Time Capture Module:** Similar to the channel dimension design, we treat time steps as letters and organize them into words. Time steps exhibit varying degrees of relevance; unlike channels, time steps are continuous and ordered. Furthermore, neighboring time steps have higher relevance due to temporal dependencies. Therefore, in this module, we cluster time steps while preserving their temporal continuity, specifically through a patching operation on neighboring time steps. This approach enables LOSEC to effectively capture overall trends over time, enhance feature extraction and robustness against noise and interference, and maintain low complexity.

*2) **Overview of LOSEC**:* In contrast to the full backpropagation process in full fine-tuning, where all parameters are updated, the fine-tuning of the LOSEC adapter for new tasks selectively adjusts the adapter block while keeping the backbone frozen. Specifically, only the newly introduced parameters $\boldsymbol{\nu}$ within the LOSEC block are updated, whereas the original parameters $\boldsymbol{\omega}$ remain unchanged. Consequently, backpropagation halts at the adapter block, and no gradients are propagated to the shallower layers. The fine-tuning procedure for LOSEC is illustrated in Fig. 2, and the update rule for LOSEC adapter fine-tuning is expressed as:

$$\boldsymbol{\nu}_{i+1} = \boldsymbol{\nu}_i - \alpha \nabla \mathcal{L}'\left(\boldsymbol{\nu}_i\right), \tag{5}$$

where $\mathcal{L}'\left(\boldsymbol{\nu}\right)$ represents the Huber loss, selected for its robustness compared to MSE, and is defined as:

$$\text{HuberLoss}\left(y, \hat{y}\right) = \begin{cases} \frac{1}{2}\left(y-\hat{y}\right)^2, & \text{if } |y-\hat{y}| \le \delta, \\ \delta\left(|y-\hat{y}| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases} \tag{6}$$

The overall architecture of LOSEC is illustrated in Fig. 3. The input to LOSEC, denoted as $\mathbf{Z} \in \mathbb{R}^{C\times H\times d}$ (with subscripts $t+1 : t+H$ omitted for simplicity), represents the output from the backbone and encodes information about time, channels, and hidden dimensions. Initially, the tensor $\mathbf{Z}$ is patched into tokens $\mathbf{Z}_P \in \mathbb{R}^{C\times N\times P\times d}$ and processed by the

local semantic time capture module, where N represents the token dimension and P denotes the patch (hidden) dimension. This module computes attention along the channel & patch, channel & hidden, and patch & hidden dimensions, resulting in the time attention $\mathcal{T} \in \mathbb{R}^{C\times N\times P\times d}$. Next, the patched time series $\mathbf{Z}_P$ is clustered into $\mathbf{Z}_C \in \mathbb{R}^{K\times N\times P\times d}$ and passed through the local semantic channel capture module, where K represents the clustered channel dimension. This module computes attention along the token & patch, token & hidden, and patch & hidden dimensions, yielding cluster attention $\mathcal{C}_l \in \mathbb{R}^{K\times N\times P\times d}$. By using the clustering matrix, which records channel-cluster mappings, the channel attention is reconstructed, producing $\mathcal{C} \in \mathbb{R}^{C\times N\times P\times d}$. By combining time and channel attention, we obtain $\mathcal{T}+\mathcal{C} \in \mathbb{R}^{C\times N\times P\times d}$. This result is flattened and linearly projected to yield the overall attention $\mathcal{A} \in \mathbb{R}^{C\times H\times d}$. The final predicted output, $\hat{\mathbf{X}}_{t+1:t+H}$, is obtained by adding a residual connection and passing through the corresponding MLP layer. Detailed descriptions of the local semantic time capture and local semantic channel capture modules are provided in the following sections.

*3) **Local Semantic Time Capture**:* To capture local semantic information over the time dimension, we first patch the matrix $\mathbf{Z}$ into semantic tokens $\mathbf{Z}_P \in \mathbb{R}^{C\times N\times P\times d}$, where $N = \lfloor \frac{H-P}{S} + 1 \rfloor$, and S denotes the stride used for patching. We apply multi-head self-attention over time to identify correlations among these semantic tokens. However, since $\mathbf{Z}_P$ is high-dimensional, fully capturing attention by flattening $N \times P \times d$ and treating it as the hidden dimension would result in prohibitively high computational complexity, requiring $\mathcal{O}(C^2NPd)$ operations. Instead, we propose capturing local semantic information alternately across specific dimensions by applying time attention along the channel & patch, channel & hidden, and patch & hidden dimensions, respectively, as shown in Fig. 3. Specifically, to compute time attention across the hidden dimension, we slice $\mathbf{Z}_P$ along channel & patch dimensions, forming sets $\left\{\mathbf{Z}_P^{cp} \in \mathbb{R}^{N\times d}, c=1,2,\ldots,C, p=1,2,\ldots,P\right\}$. We then derive query, key, and value matrices based on $\mathbf{Z}_P^{cp}$:

$$\mathbf{Q}^{cp} = F_q^{cp}\left(\mathbf{Z}_P^{cp}\right), \tag{7a}$$

$$\mathbf{K}^{cp} = F_k^{cp}\left(\mathbf{Z}_P^{cp}\right), \tag{7b}$$

$$\mathbf{V}^{cp} = F_v^{cp}\left(\mathbf{Z}_P^{cp}\right), \tag{7c}$$

where $\mathbf{Q}^{cp}, \mathbf{K}^{cp}, \mathbf{V}^{cp} \in \mathbb{R}^{N\times d}$, and $F_q^{cp}, F_k^{cp}, F_v^{cp}$ are linear transformation layers. Multi-head attention is also adopted to better capture their correlations:

$$\mathcal{T}_i^{cp} = \text{softmax}\left(\frac{1}{\sqrt{d}}\mathbf{Q}_i^{cp}(\mathbf{K}_i^{cp})^T\right)\mathbf{V}_i^{cp},\ i=1,2,\ldots,h^{cp}, \tag{8}$$

where $\mathcal{T}_i^{cp} \in \mathbb{R}^{N\times d_h}$ represents attention for head i, and $\mathbf{Q}_i^{cp}, \mathbf{K}_i^{cp}, \mathbf{V}_i^{cp} \in \mathbb{R}^{N\times d_h}$ are query, key, and value matrices for head i derived from $\mathbf{Q}^{cp}, \mathbf{K}^{cp}, \mathbf{V}^{cp}$. The number of heads and head dimension are denoted by h^{cp} and d_h, respectively.

As a result, the time attention across the hidden dimension $\mathcal{T}_i^{cp}$ is computed along the channel & patch dimensions. Similarly, time attention across the patch dimension $\mathcal{T}_i^{cd}$ is derived along the channel & hidden dimensions, while time attention across the channel dimension $\mathcal{T}_i^{pd}$ is computed along the patch & hidden dimensions. Finally, the overall time attention is computed as:

$$\mathcal{T}_{c,n,p,d} = \mathcal{T}_{c,p}^{cp} + \mathcal{T}_{c,d}^{cd} + \mathcal{T}_{p,d}^{pd}, \tag{9}$$

where $\mathcal{T}_{c,n,p,d}$ is the (c,n,p,d) element in the time attention matrix $\mathcal{T} \in \mathbb{R}^{C\times N\times P\times d}$, $\mathcal{T}_{c,p}^{cp}$ is the (c,p) element in $\mathcal{T}^{cp}$, $\mathcal{T}_{c,d}^{cd}$ is the (c,d) element in $\mathcal{T}^{cd}$, and $\mathcal{T}_{p,d}^{pd}$ is the (p,d) element in $\mathcal{T}^{pd}$. The matrices $\mathcal{T}^{cp}$, $\mathcal{T}^{cd}$, and $\mathcal{T}^{pd}$ are formed by concatenating the outputs from their respective multi-attention heads.

*4) **Local Semantic Channel Capture**:* To capture local semantic information over channel dimensions, we cluster channels into clusters, which also reduces attention complexity by focusing on clusters rather than individual channels. We cluster $\mathbf{Z}_P$ along the channel dimension into K clusters, denoted as $\mathbf{Z}_C \in \mathbb{R}^{K\times N\times P\times d}$. There are various methods to cluster C channels into K clusters; here, we consider a deep learning-based approach that is simple and compatible with our overall framework. We aim to assign a probability $p_{c,k}$ for channel c and cluster k, represented as the normalized inner product of two vectors. One of these vectors is the input $\mathbf{Z}_p$, and we also need to define an initialized K cluster embedding $\mathbf{C} \in \mathbb{R}^{K\times d}$. Consequently, the probability matrix $\mathbf{P} \in \mathbb{R}^{C\times N\times P\times K}$ can be computed as follows:

$$\mathbf{P}_{c,n,p,k} = \sum_d \mathbf{Z}_{P(c,n,p,d)} \cdot \mathbf{C}_{k,d}, \tag{10}$$

where $\mathbf{P}_{c,n,p,k}$ is the (c,n,p,k) element of $\mathbf{P}$, $\mathbf{Z}_{P(c,n,p,d)}$ is the (c,n,p,d) element of $\mathbf{Z}_P$, $\mathbf{C}_{k,d}$ is the (k,d) element of $\mathbf{C}$. We expect the clustering assignment matrix to be a one-hot matrix, that is, $\mathbf{A} \in \mathbb{R}^{C\times N\times P\times K}$, $\mathbf{A}_{c,n,p,k} \sim \text{Bernoulli}(p_{c,k})$. Although the softmax operation can ensure that the probability distribution satisfies $\sum_k \mathbf{P}_{c,n,p,k} = 1$, the $\arg\max$ $\mathbf{P}_{c,n,p,k}$ operation is not differentiable and does not preserve the probability distribution, rendering backpropagation infeasible and clustering less useful. Therefore, we utilize a reparameterization trick along with the softmax operation, specifically Gumbel-Softmax [38], to make the clustering assignment matrix differentiable and approximate the Bernoulli distribution, which is defined as follows:

$$\mathbf{A}_{c,n,p,i} = \frac{\exp\left(\left(\log(\mathbf{P}_{c,n,p,i}) + g_i\right)/\tau\right)}{\sum_{j=1}^{k}\exp\left(\left(\log(\mathbf{P}_{c,n,p,j}) + g_j\right)/\tau\right)}, \quad i=1,2,\ldots,k, \tag{11}$$

where g_i are i.i.d. samples drawn from the Gumbel distribution, and τ is the temperature parameter. Subsequently, the clustered $\mathbf{Z}_C$ can be obtained as:

$$\mathbf{Z}_{C(k,n,p,d)} = \sum_c \mathbf{Z}_{P(c,n,p,d)} \cdot \mathbf{A}_{c,n,p,k}. \tag{12}$$

Based on the clustered $\mathbf{Z}_C$, the channel attention can be derived in a manner similar to the computation of time attention. Specifically, we obtain the channel attention across the hidden dimension $\mathcal{C}^{np}$ along the token & patch dimensions; the channel attention across the patch dimension $\mathcal{C}^{nd}$ along the token & hidden dimensions; and the channel attention across

the token dimension $\mathcal{C}^{pd}$ along the patch & hidden dimensions. Consequently, the overall channel attention is computed as:

$$\mathcal{C}_{c,n,p,d} = \mathcal{C}^{np}_{n,p} + \mathcal{C}^{nd}_{n,d} + \mathcal{C}^{pd}_{p,d}. \tag{13}$$

D. Streamlined Complexity: The Efficient Architecture Design

LOSEC is a low-complexity model featuring three key design strategies to achieve reduced computational complexity:

- **Formation of local semantic information across time and channel dimensions:** The local semantic capture module forms semantic information while lowering complexity. For the time dimension, patch H time series into N tokens, reducing complexity from $\mathcal{O}\left(H^2\right)$ to $\mathcal{O}\left(H^2/S^2\right)$, with N on the order of $\mathcal{O}\left(H/S\right)$. For the channel dimension, cluster C channels into K clusters, reducing complexity from $\mathcal{O}\left(C^2\right)$ to $\mathcal{O}\left(K^2\right)$.
- **Alternating capture over time and channel dimensions:** Instead of requiring global attention with a complexity of $\mathcal{O}\left(C^2H^2\right)$, alternating between dimensions reduces the complexity to $\mathcal{O}\left(\max\left(C^2, H^2\right)\right)$.
- **Dimension-specific attention mechanisms:** When capturing attention across time or channel dimensions, the remaining dimensions are not flattened but treated alternately as hidden dimensions, reducing complexity from $\mathcal{O}\left(C^2NPd\right)$ to $\mathcal{O}\left(C^2 \cdot \max\left(N, P, d\right)\right)$ for channel attention and from $\mathcal{O}\left(N^2CPd\right)$ to $\mathcal{O}\left(N^2 \cdot \max\left(C, P, d\right)\right)$ for time attention.

These three designs work together to streamline complexity in LOSEC. Specifically, the overall complexity is derived as follows: We employ a multi-head self-attention mechanism to capture local semantic information within the proposed LOSEC model. Consequently, the complexity is directly determined by the self-attention operation, which is expressed as $\mathcal{O}(n^2d)$, where n represents the sequence length, and d denotes the model's dimensionality [39]. For local semantic time capture, the time attention $\mathcal{T}^{cp}$, applied across hidden dimensions of length N and dimension d, incurs a computational cost of $\mathcal{O}\left(N^2d\right)$ per channel and patch. This results in a total complexity of $\mathcal{O}\left(CPN^2d\right)$. Similarly, the complexities of $\mathcal{T}^{cd}$ and $\mathcal{T}^{pd}$ are $\mathcal{O}\left(CdN^2P\right)$ and $\mathcal{O}\left(PdN^2C\right)$, respectively. Hence, the overall complexity for local semantic time capture is $\mathcal{O}\left(CN^2Pd\right)$. For local semantic channel capture, the channel attention $\mathcal{C}^{np}$, applied across hidden dimensions of length K and dimension d, incurs a computational cost of $\mathcal{O}\left(K^2d\right)$ per token and patch, yielding a total complexity of $\mathcal{O}\left(NPK^2d\right)$. Similarly, the complexities of $\mathcal{C}^{nd}$ and $\mathcal{C}^{pd}$ are $\mathcal{O}\left(NdK^2P\right)$ and $\mathcal{O}\left(PdK^2N\right)$, respectively. Therefore, the overall complexity for local semantic channel capture is $\mathcal{O}\left(K^2NPd\right)$. By integrating these two modules, the total computational complexity of the LOSEC model is $\mathcal{O}\left(NPd \cdot \max\left(CN, K^2\right)\right)$.

V. Experiments

A. Experimental Settings

Dataset. We utilize data collected from a real data center [25], obtained through various IoT devices with sampling intervals of one or two minutes. The data is processed using linear interpolation, Z-score normalization, and the box-and-whisker plot method to identify outliers. The resulting multivariate time series consists of one-minute intervals across 130 dimensions, totaling 274,662 samples.

TABLE I: Multivariate time series forecasting performance: The context length is set to 8, and the prediction length is set to 4 (best results in **bold**, second best in underlined, and third best in dotted underline).

MODEL	MAE	MSE	SMAPE	MASE	ARk
SUPERVISED					
Autoformer	0.139	0.180	26.244	0.184	8.75
Crossformer	0.137	0.190	23.863	0.181	8.75
DLinear	0.135	0.179	24.786	0.178	6.50
FEDformer	0.141	0.182	26.858	0.186	9.75
Informer	0.128	0.179	22.939	0.169	3.50
iTransformer	0.135	0.245	19.451	0.178	6.25
LightTS	0.168	0.204	34.194	0.222	12.50
NS-Transformer	0.130	0.182	22.971	0.171	5.75
PatchTST	0.136	0.249	**19.424**	0.179	7.25
TimesNet	0.130	0.187	22.580	0.171	5.00
Transformer	0.353	0.397	67.239	0.466	14.00
ZERO-SHOT					
Plag-Llama (Zero-Shot)	0.127	0.194	20.579	0.180	5.75
FINE-TUNING					
Plag-Llama (Full)	0.153	**0.176**	28.643	0.201	9.25
Plag-Llama (LOSEC Adapter)	**0.123**	0.177	21.980	**0.162**	**2.00**

Baselines. We establish the following recent SOTA methods as baselines: Autoformer [40], Crossformer [41], DLinear [23], FEDformer [42], Informer [43], iTransformer [33], LightTS [44], Non-stationary (NS)-Transformer [45], PatchTST [22], TimesNet [34], and Transformer [25]. The default parameters are configured as follows: the number of epochs is set to 50; the Huber loss hyperparameter δ is set to 0.005; and the learning rate is 3e-4. The patch size is 2, with a stride of 1. The number of clusters K is set to 64, the context length is set to 8, and the prediction length is set to 4. By default, the number of attention heads is 8, although it may vary based on the head dimension (using the greatest common divisor). The hidden layer sizes of the multi-layer perceptron (MLP) are set to 256, 128, 64, and 32, respectively. The Adam optimizer is employed, while other parameters adhere to their originally recommended values.

Metrics. We employ commonly used metrics in time series forecasting, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Symmetric Mean Absolute Percentage Error (SMAPE), and Mean Absolute Scaled Error (MASE). Additionally, we calculate the average rank (ARk) across these metrics.

B. Multivariate Time Series Forecasting Performance

We compare our proposed methods with SOTA models in the data center multivariate time series forecasting task. The results for three categories: supervised, zero-shot, and fine-tuning models are summarized in Table I. Leveraging the powerful forecasting capabilities of Lag-Llama and the proposed transfer block, Plag-Llama demonstrates strong zero-shot performance, achieving the second-best MAE and third-best SMAPE, along with an average rank of 5.75. Although it falls short of the best performance, it still ranks third

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

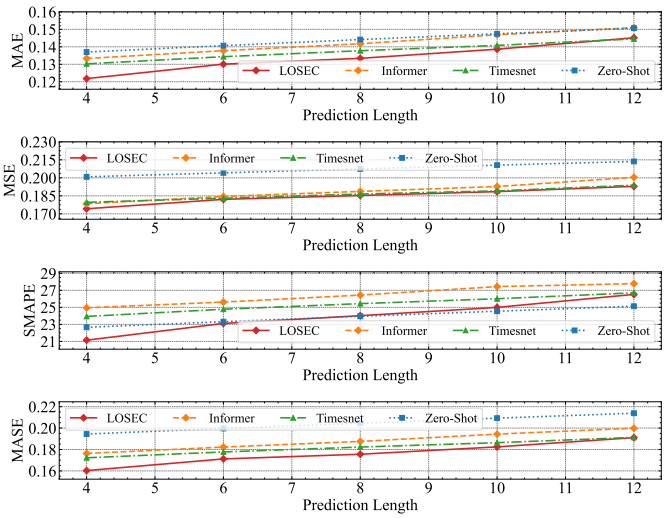

Fig. 4: Multivariate time series performance comparison versus different prediction lengths.

among the supervised SOTA models. This superior zero-shot performance indicates that the Plag-Llama framework is a promising solution to address data scarcity challenges and expedite deployment within data centers. It is important to note that the original Lag-Llama aims to align the overall distribution with the ground truth; however, the simple average layer effectively transfers these probabilistic capabilities to point forecasting. This is attributed to the symmetric Student's t-distribution adopted in Lag-Llama, where learning optimal mean values is beneficial for probabilistic forecasting. However, full fine-tuning significantly undermines the performance of Plag-Llama due to overfitting, resulting in an average rank of 9.25, which is even worse than its zero-shot performance. Empowered by the proposed local semantic information capture framework, the LOSEC adapter consistently demonstrates superior performance compared to other SOTA models, exhibiting an average improvement of 3.34% over the second-best model.

Based on the above results, we further explore the performance of the models versus different prediction lengths. We select the two best supervised models, zero-shot Plag-Llama, and the LOSEC adapter for this comparison. As the prediction length increases, models face greater challenges, as illustrated in Fig. 4. LOSEC consistently outperforms other models, except for the SMAPE metric, where it slightly underperforms compared to zero-shot Plag-Llama when the prediction length exceeds 10. These results demonstrate that LOSEC can effectively capture local semantic information across both channel and time dimensions, showcasing SOTA performance in multivariate time series forecasting tasks.

C. Few-Shot Ability

To assess the few-shot capabilities of our proposed models in comparison to SOTA methods, we train on the last 20%, 40%, 60%, and 80% of the dataset using both supervised learning from scratch and fine-tuning techniques. The results, presented in Table II, show that LOSEC consistently outperforms SOTA models across various few-shot ratios, achieving results comparable to the top-performing supervised models, specifically PatchTST and TimesNet. However, PatchTST and TimesNet exhibit instability, with performance occasionally declining as more historical data is introduced, suggesting sensitivity to noisy data. Besides, our proposed LOSEC model employs an adapter fine-tuning approach that leverages its generalization ability, adjusting only a subset of parameters for new tasks rather than training entirely from scratch. Although full fine-tuning of Plag-Llama achieves slightly better performance with less data, its overall results are limited by overfitting. Reducing the data size somewhat alleviates this issue, but the underlying problem persists. In contrast, the proposed LOSEC method fundamentally addresses this limitation by resolving it at the model architectural level.

Overall, these results demonstrate that Plag-Llama performs strongly in multivariate time series forecasting, especially in zero-shot and few-shot scenarios. Our analysis further highlights that while full fine-tuning of Plag-Llama can lead to overfitting, LOSEC effectively mitigates this issue by leveraging a more effective adapter fine-tuning approach. By adjusting only a subset of parameters, LOSEC achieves better generalization without the pitfalls of overfitting, yielding more stable results across varied data contexts. Taken together, these experiments emphasize LOSEC's effectiveness from multiple perspectives, showcasing its advantages over SOTAs and traditional full fine-tuning methods.

D. Visualization of Local Semantic Capture

1) ***LOSEC over Channel****:* Fig. 5a presents a t-SNE visualization of time series data along the channel dimension, where each point represents a channel within a sample, and each color denotes a distinct channel type. For simplicity, we display only the first 10 channels. It is evident that channels 2, 3, 5, 6, 7, 8, and 9 are closely positioned in the t-SNE space, suggesting they may form a cohesive cluster. In contrast, channels 0, 1, and 4 are more distinct from this cluster and from each other. The similarity matrices in Fig. 5b and 5c further confirm these observations, highlighting strong correlations between certain channels. In Fig. 5b, dark regions or bands appear in the similarity map, indicating clusters of channels with consistent similarity patterns. These dark regions correspond to groups of channels with higher mutual correlations, while channels outside these areas or in lighter regions exhibit lower correlation, implying more diverse or independent feature sets. Fig. 5c provides more detailed correlations among the first 10 channels, where channels 2, 3, 5, 6, 7, 8, and 9 show high similarity scores, while channels 0, 1, and 4 are notably less similar to both each other and the other channels. This observation aligns with conclusions from Fig. 5a.

The visualizations above reveal strong correlations within specific channel clusters, while distinct features separate different clusters. These findings highlight the necessity of capturing inter-channel correlations without interference from unrelated channels, emphasizing the importance of capturing local semantic information across channels. Notably, local

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

TABLE II: Comparison of few-shot abilities: The context length is set to 8, and the prediction length is set to 4 (best results in **bold**, second best in underlined, and third best in dotted underline).

DATA %	MODEL	MAE	MSE	SMAPE	MASE	ARk	DATA %	MODEL	MAE	MSE	SMAPE	MASE	ARk
20 %	Autoformer	0.141	0.181	26.745	0.186	7.50	40 %	Autoformer	0.141	0.184	26.375	0.186	8.50
	Crossformer	0.143	0.189	23.991	0.188	8.00		Crossformer	0.156	0.190	28.696	0.206	10.25
	DLinear	0.135	0.177	25.151	0.179	6.00		DLinear	0.137	0.172	26.154	0.180	5.75
	FEDformer	0.145	0.186	27.870	0.191	9.50		FEDformer	0.142	0.184	27.218	0.188	9.00
	Informer	0.134	0.182	24.174	0.176	6.25		Informer	0.130	0.180	23.389	0.172	4.25
	iTransformer	0.133	0.207	21.653	0.175	5.50		iTransformer	0.132	0.223	19.925	0.174	5.75
	LightTS	0.373	0.369	70.688	0.492	12.00		LightTS	0.170	0.208	31.831	0.224	11.50
	NS-Transformer	0.132	0.181	23.919	0.174	4.50		NS-Transformer	0.130	0.181	23.153	0.172	4.00
	PatchTST	**0.130**	0.208	**20.527**	**0.172**	3.50		PatchTST	0.132	0.227	**19.729**	0.175	6.25
	TimesNet	0.131	0.179	23.847	0.173	**3.00**		TimesNet	0.130	0.181	23.012	0.171	3.00
	Transformer	0.484	0.557	92.126	0.639	13.00		Transformer	0.407	0.445	80.859	0.537	13.00
	Plag-Llama (Full)	0.156	**0.166**	31.457	0.205	8.50		Plag-Llama (Full)	0.140	**0.168**	29.339	0.184	7.00
	Plag-Llama (LOSEC Adapter)	0.132	0.179	24.885	0.173	3.75		Plag-Llama (LOSEC Adapter)	**0.128**	0.178	23.576	**0.168**	**2.75**
DATA %	**MODEL**	**MAE**	**MSE**	**SMAPE**	**MASE**	**ARk**	**DATA %**	**MODEL**	**MAE**	**MSE**	**SMAPE**	**MASE**	**ARk**
60 %	Autoformer	0.140	0.181	26.287	0.184	7.00	80 %	Autoformer	0.139	0.180	26.245	0.184	6.00
	Crossformer	0.161	0.193	28.054	0.212	9.50		Crossformer	0.210	0.209	37.375	0.277	10.75
	DLinear	0.141	**0.172**	28.392	0.186	6.75		DLinear	0.218	0.198	49.875	0.288	11.00
	FEDformer	0.142	0.183	26.933	0.187	8.50		FEDformer	0.142	0.183	27.261	0.188	7.75
	Informer	0.130	0.179	23.411	0.172	3.25		Informer	0.130	0.180	23.278	0.171	3.50
	iTransformer	0.132	0.227	19.713	0.174	5.75		iTransformer	0.134	0.236	19.541	0.176	5.75
	LightTS	0.172	0.212	32.919	0.227	11.25		LightTS	0.153	0.205	27.751	0.202	9.50
	NS-Transformer	0.130	0.181	23.274	0.172	4.50		NS-Transformer	0.130	0.182	22.981	0.172	4.50
	PatchTST	0.133	0.235	**19.559**	0.176	6.25		PatchTST	0.134	0.239	**19.520**	0.177	6.25
	TimesNet	0.129	0.180	23.026	0.170	**2.50**		TimesNet	0.130	0.183	22.876	0.171	4.00
	Transformer	0.367	0.405	71.127	0.485	13.00		Transformer	0.387	0.423	77.318	0.511	13.00
	Plag-Llama (Full)	0.172	0.180	35.518	0.226	9.75		Plag-Llama (Full)	0.148	0.180	29.369	0.195	7.50
	Plag-Llama (LOSEC Adapter)	**0.129**	0.180	23.835	**0.169**	3.00		Plag-Llama (LOSEC Adapter)	**0.127**	**0.178**	22.695	**0.167**	**1.50**

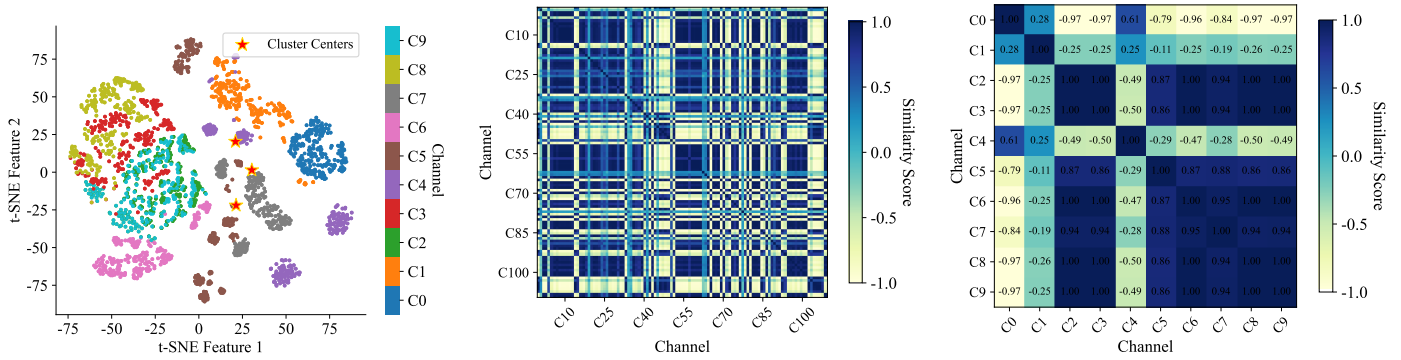

Fig. 5: Visualization of input channel correlations: (a) t-SNE clustering for the first 10 channels; (b) similarity map for all channels; (c) similarity map for the first 10 channels.

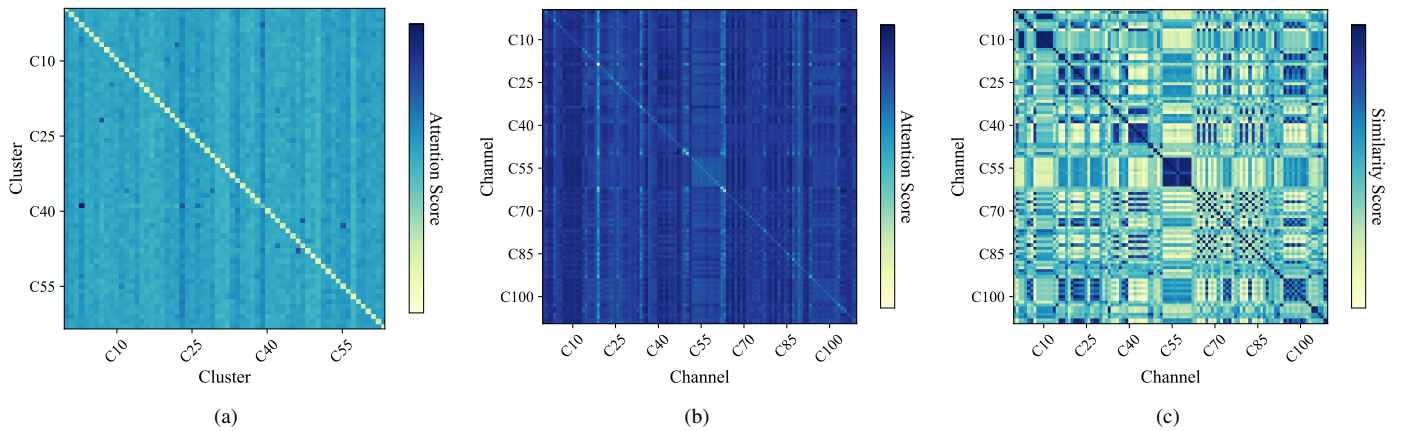

Fig. 6: Visualization of LOSEC attention maps over the channel dimension: (a) cluster attention map; (b) channel attention map; (c) similarity map for future time steps.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

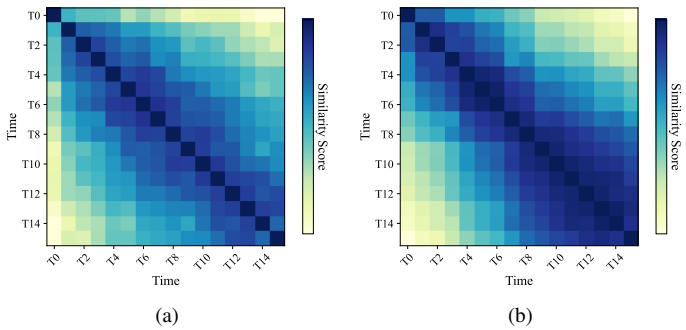

Fig. 7: Visualization of similarity maps over the time dimension: (a) similarities between context time steps; (b) similarities between future time steps applying LOSEC without patching.

semantic information within the channel dimension is visually manifested as dark regions in the similarity map, representing clusters of related channels.

To validate that LOSEC effectively captures local semantic information in the channel dimension, we also visualize LOSEC's attention maps over this dimension. Fig. 6a shows attention maps between clusters, while Fig. 6b reconstructs these maps through a cluster-channel mapping matrix. Fig. 6c shows the future time steps similarity map across channels. The reconstructed channel attention map closely aligns with this similarity map, confirming that LOSEC effectively captures channel correlations through clustering.

*2) **LOSEC over Time**:* Fig. 7a illustrates the similarity map between context time steps, revealing strong correlations between adjacent steps, with particularly high correlations observed among nearby steps. This indicates that consecutive time steps are inherently related, reflecting natural temporal continuity, where each step exhibits contextual dependencies on its neighbors. The patching strategy adopted in LOSEC groups consecutive time steps, enabling the model to fully leverage adjacent correlations, i.e., local semantic information. This approach facilitates the extraction of local semantic features across multiple time steps, enhancing the model's ability to capture correlated features, such as temporal continuity and dependencies over short intervals. Furthermore, it aggregates useful information while reducing noise, offering significant advantages over step-level attention. Fig. 7b depicts the similarity map between future time steps after applying the LOSEC module without patching. Although patching is not employed here, the map still reveals distinct high correlations among adjacent time steps compared to context time steps. This further validates the commonality of such correlations and highlights the advantages of the patching mechanism in our experiments.

VI. Ablation Studies

In this section, to further validate the effectiveness of the proposed LOSEC model and clarify the roles of its individual modules, we conduct ablation studies on context length, attention across channel and time dimensions, local semantic channel capture, and local semantic time capture.

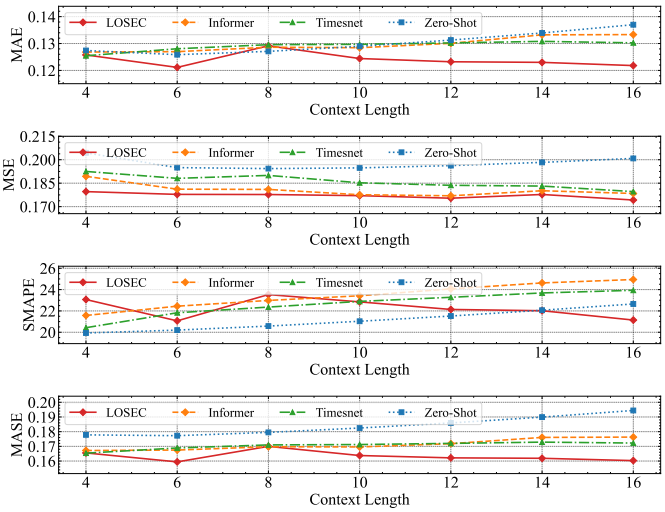

Fig. 8: Ablation study on context length: Multivariate time series performance comparison versus different context lengths.

TABLE III: Ablation study on attention across channel and time dimensions in the LOSEC adapter: Channel+Time refers to the original LOSEC, Time refers to attention only over time, Channel refers to attention only over the channel, and None refers to the absence of attention.

MODEL	MAE	MSE	SMAPE	MASE	ARk
LOSEC					
Channel+Time	**0.123**	0.177	**21.980**	**0.162**	**1.25**
Time	0.127	**0.176**	23.858	0.168	2.25
Channel	0.437	0.427	92.403	0.576	5.00
None	0.130	0.178	23.468	0.171	3.50
SOTA					
Informer	0.128	0.179	22.939	0.169	3.00

A. Influence of Context Length

We investigate model performance across different context lengths, comparing the top two supervised models, zero-shot Plag-Llama, and the LOSEC adapter. As shown in Fig. 8, LOSEC consistently outperforms other models on most metrics, except for SMAPE, where it achieves comparable performance. Furthermore, as context length increases, the performance of zero-shot Plag-Llama, Informer, and TimesNet deteriorates significantly, while LOSEC maintains stable and superior performance. A longer context length provides more valuable historical information, but it also introduces additional noise and interference. Consequently, these models struggle with longer contexts, whereas LOSEC demonstrates robustness and an enhanced ability to capture relevant information amid interference.

B. Influence of Attention across Channel and Time Dimension

To investigate the effects of attention mechanisms across channel and time dimensions, we sequentially remove time and channel attention to create three configurations: time-only, channel-only, and no-attention. We also use the best-performing supervised model, Informer, as the baseline SOTA. When channel attention is removed, LOSEC shows an 8.54% decrease in SMAPE and a 3.73% average decrease, yet it still outperforms the SOTA model, highlighting the importance

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2025.3541967

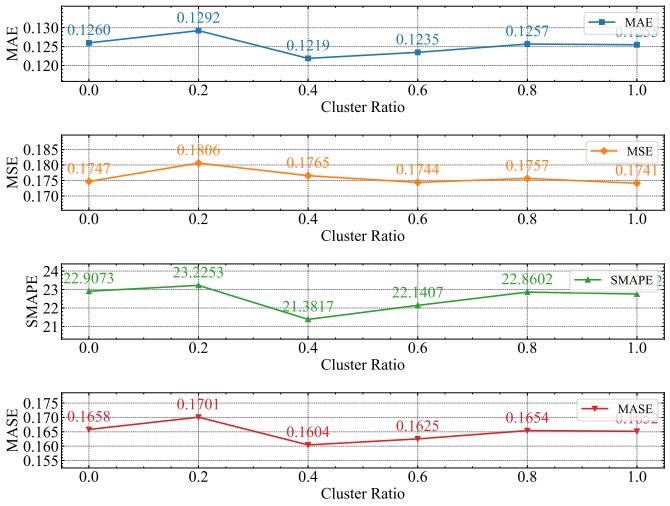

Fig. 9: Ablation study on local semantic channel capture: Multivariate time series performance comparison versus different cluster ratios.

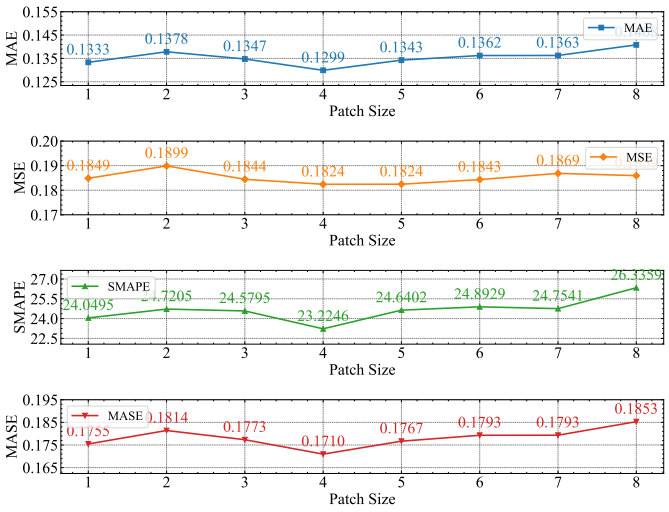

Fig. 10: Ablation study on local semantic time capture: Multivariate time series performance comparison versus different patch sizes.

of capturing inter-channel correlations. Separately, removing time attention results in a significant degradation in LOSEC's performance, reducing it to below the configuration without any attention. This result confirms the essential role of time attention. Although channel attention captures inter-channel correlations, it may miss channel-specific information when considered alone, leading to diminished performance. This information loss has minimal impact on LOSEC's performance, as time attention preserves complete information with low complexity, allowing LOSEC to sustain strong performance with reduced complexity. With both attention mechanisms removed, LOSEC's performance, while still comparable to SOTA, decreases substantially compared to the original LOSEC (by 4.65% on average). These results validate LOSEC's robust time series forecasting capabilities and highlight the necessity and effectiveness of leveraging attention mechanisms to capture local semantic information across both channel and time dimensions.

C. Influence of Local Semantic Channel Capture

As shown in Fig. 9, we vary the cluster ratio from 0.0 to 1.0 with intervals of 0.2, where 0.0 represents all channels fused into a single channel, and 1.0 indicates that each channel remains individual without clustering. Our findings reveal that a fully unclustered approach (1.0) is not optimal, highlighting interference and noise among channels. Purely global attention cannot capture these inter-channel correlations effectively, which helps explain why channel-dependent models may underperform compared to channel-independent ones. Therefore, LOSEC's approach of clustering channels to capture local semantic information is essential, as it reduces interference and noise among channels. Subsequent experimental results further underscore the LOSEC model's effectiveness and superiority over SOTA methods. Clustering all channels into one (0.0) is also suboptimal, as it discards inter-channel information, retaining only averaged features. Nevertheless, even at a 0.0 clustering ratio, LOSEC performs well, surpassing SOTA models. These results indicate that averaging channels is not necessarily a poor choice. Additionally, some channel information is preserved through time attention, which helps retain performance. Therefore, LOSEC achieves SOTA performance even with a single-channel cluster, maintaining remarkably low complexity. Furthermore, in our experiments, LOSEC achieves the best performance when the cluster ratio is within $[0.4, 0.6]$, though this may vary with different datasets and tasks. This parameter can be selected to balance performance and complexity depending on the specific task.

D. Influence of Local Semantic Time Capture

We also investigate the impact of time patching, as shown in Fig. 10, where the patch size ranges from 1 to 8, with a patch size of 1 indicating no patching. A larger patch size captures more global information and reduces complexity but may result in the loss of local details. Our results indicate that the optimal patch size is approximately 4 in this experiment, which outperforms no patching by an average of 2.47%. The best patch size varies depending on the dataset and task, and its selection should balance complexity and accuracy. Based on these results, we conclude that LOSEC's approach to capturing local information over time through patching is both crucial and instrumental in enhancing model performance.

VII. Conclusion

In this paper, we have investigated large models empowered by local semantic capture for multivariate time series forecasting in IoT-enabled data centers, an approach that addresses data scarcity, facilitates rapid deployment, and achieves superior performance. We have adapted Lag-Llama for multivariate point time series forecasting, incorporating zero-shot, few-shot and fine-tuning capabilities, by proposing the framework Plag-Llama. Additionally, we have designed a novel Local Semantic Capture (LOSEC) model for adapter fine-tuning, which captures local semantic information across patched time and clustered channels. The LOSEC model can also be seamlessly integrated into other SOTA models or algorithms, particularly

those lacking the ability to effectively construct and capture semantic information. Extensive experiments have demonstrated the superior performance of the proposed models, while ablation studies have validated the rationale and effectiveness of our design philosophy, particularly the construction and capture of semantic information. In the future, we will apply our models to multivariate time series forecasting in various domains to further validate their superiority and contribute to achievements in other areas.

REFERENCES

[1] Y. Sun, B. Cheng, J. Li, G. Jiang, T. Zhang, and H. Zhou, "Plag-llama for IoT-enabled data centers: A multivariate time series forecasting approach," in *GLOBECOM 2024 - 2024 IEEE Global Communications Conference*.

[2] J. Wang, H. Du, D. Niyato, J. Kang, S. Cui, X. Shen, and P. Zhang, "Generative AI for integrated sensing and communication: Insights from the physical layer perspective," *IEEE Wireless Communications*, vol. 31, no. 5, pp. 246–255, Oct. 2024.

[3] C. Zhang, G. Sun, J. Li, Q. Wu, J. Wang, D. Niyato, and Y. Liu, "Multi-objective aerial collaborative secure communication optimization via generative diffusion model-enabled deep reinforcement learning," *IEEE Transactions on Mobile Computing*, pp. 1–18, 2024.

[4] P. Zhao, L. Yang, Z. Kang, and J. Lin, "On Predicting the PUE with Gated Recurrent Unit in Data Centers," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, Dec. 2019, pp. 1664–1670.

[5] H. D. Vu, K. S. Chai, B. Keating, N. Tursynbek, B. Xu, K. Yang, X. Yang, and Z. Zhang, "Data Driven Chiller Plant Energy Optimization with Domain Knowledge," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: Association for Computing Machinery, Nov. 2017, pp. 1309–1317.

[6] Y. Xu, H. Zhou, T. Ma, J. Zhao, B. Qian, and X. Shen, "Leveraging multiagent learning for automated vehicles scheduling at nonsignalized intersections," *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11 427–11 439, Jul. 2021.

[7] Y. Xu, B. Qian, K. Yu, T. Ma, L. Zhao, and H. Zhou, "Federated learning over fully-decoupled RAN architecture for two-tier computing acceleration," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 3, pp. 789–801, Mar. 2023.

[8] J. Xue, K. Yu, T. Zhang, H. Zhou, L. Zhao, and X. Shen, "Cooperative Deep Reinforcement Learning Enabled Power Allocation for Packet Duplication URLLC in Multi-Connectivity Vehicular Networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8143–8157, Aug. 2024.

[9] G. Jiang, Y. Sun, B. Cheng, Y. Wang, and H. Zhou, "Leveraging multi-task learning for energy consumption prediction in IoT-based data center," in *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*, Oct. 2023, pp. 943–948.

[10] G. Sun, W. Xie, D. Niyato, H. Du, J. Kang, J. Wu, S. Sun, and P. Zhang, "Generative AI for advanced UAV networking," Apr. 2024.

[11] J. Wang, H. Du, Y. Liu, G. Sun, D. Niyato, S. Mao, D. I. Kim, and X. Shen, "Generative AI based secure wireless sensing for ISAC networks," *arXiv preprint arXiv:2408.11398*, 2024.

[12] H. Xue and F. D. Salim, "PromptCast: A new prompt-based learning paradigm for time series forecasting," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2023.

[13] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan, and Q. Wen, "Time-LLM: Time series forecasting by reprogramming large language models," *arXiv preprint arXiv:2310.01728*, Oct. 2023.

[14] A. Garza and M. Mergenthaler-Canseco, "TimeGPT-1," *arXiv preprint arXiv:2310.03589*, Oct. 2023.

[15] K. Rasul, A. Ashok, A. R. Williams, H. Ghonia, R. Bhagwatkar, A. Khorasani, M. J. D. Bayazi, G. Adamopoulos, R. Riachi, N. Hassen, M. Biloš, S. Garg, A. Schneider, N. Chapados, A. Drouin, V. Zantedeschi, Y. Nevmyvaka, and I. Rish, "Lag-llama: Towards foundation models for probabilistic time series forecasting," *arXiv preprint arXiv:2310.08278*, Feb. 2024.

[16] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, "A survey of large language models," Oct. 2024.

[17] V. Lialin, V. Deshpande, and A. Rumshisky, "Scaling down to scale up: A guide to parameter-efficient fine-tuning," *arXiv preprint arXiv:2303.15647*, Mar. 2023.

[18] Z. Han, C. Gao, J. Liu, J. Zhang, and S. Q. Zhang, "Parameter-efficient fine-tuning for large models: A comprehensive survey," Apr. 2024.

[19] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. D. Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 2790–2799.

[20] H. Zhao, H. Tan, and H. Mei, "Tiny-attention adapter: Contexts are more important than the number of parameters," Oct. 2022.

[21] J. Chen, J. E. Lenssen, A. Feng, W. Hu, M. Fey, L. Tassiulas, J. Leskovec, and R. Ying, "From similarity to superiority: Channel clustering for time series forecasting," Mar. 2024.

[22] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *The Eleventh International Conference on Learning Representations*, Sep. 2022.

[23] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, pp. 11 121–11 128, Jun. 2023.

[24] Y. Wang, Y. Sun, B. Cheng, G. Jiang, and H. Zhou, "DQN-based chiller energy consumption optimization in IoT-enabled data center," in *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*, Oct. 2023, pp. 985–990.

[25] Y. Sun, Y. Wang, G. Jiang, B. Cheng, and H. Zhou, "Deep learning-based power usage effectiveness optimization for IoT-enabled data center," *Peer-to-Peer Networking and Applications*, Mar. 2024.

[26] Y. Sun, B. Cheng, G. Jiang, Y. Wang, and H. Zhou, "Cooling capacity prediction for safety-guaranteed optimization in IoT-enabled data center," in *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*, Oct. 2023, pp. 920–925.

[27] C. Chang, W.-C. Peng, and T.-F. Chen, "LLM4TS: Two-stage fine-tuning for time-series forecasting with pre-trained LLMs," *arXiv preprint arXiv:2308.08469*, Oct. 2023.

[28] Y. Liu, H. Zhang, C. Li, X. Huang, J. Wang, and M. Long, "Timer: Generative pre-trained transformers are large time series models," in *Forty-First International Conference on Machine Learning*, Jun. 2024.

[29] M. Jin, Q. Wen, Y. Liang, C. Zhang, S. Xue, X. Wang, J. Zhang, Y. Wang, H. Chen, X. Li, S. Pan, V. S. Tseng, Y. Zheng, L. Chen, and H. Xiong, "Large models for time series and spatio-temporal data: A survey and outlook," Oct. 2023.

[30] Y. Jiang, Z. Pan, X. Zhang, S. Garg, A. Schneider, Y. Nevmyvaka, and D. Song, "Empowering time series analysis with large language models: A survey," Feb. 2024.

[31] M. Jin, Y. Zhang, W. Chen, K. Zhang, Y. Liang, B. Yang, J. Wang, S. Pan, and Q. Wen, "Position paper: What can large language models tell us about time series analysis," Feb. 2024.

[32] T. Lei, J. Bai, S. Brahma, J. Ainslie, K. Lee, Y. Zhou, N. Du, V. Zhao, Y. Wu, B. Li, Y. Zhang, and M.-W. Chang, "Conditional adapters: Parameter-efficient transfer learning with fast inference," *Advances in Neural Information Processing Systems*, vol. 36, pp. 8152–8172, Dec. 2023.

[33] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "Itransformer: Inverted transformers are effective for time series forecasting," *arXiv preprint arXiv:2310.06625*, 2023.

[34] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-variation modeling for general time series analysis," in *The Eleventh International Conference on Learning Representations*, Sep. 2022.

[35] L. Zhao and Y. Shen, "Rethinking channel dependence for multivariate time series forecasting: Learning from leading indicators," Feb. 2024.

[36] D. Li and H. Zhang, "Improved regularization and robustness for fine-tuning in neural networks," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 27 249–27 262.

[37] K. Li, W. Gu, M. Xue, J. Xiao, D. Shi, and X. Wei, "Atten-adapter: A unified attention-based adapter for efficient tuning," in *2023 IEEE International Conference on Image Processing (ICIP)*, Oct. 2023, pp. 1265–1269.

[38] S. Chen, L. Li, G. Wang, M. Pang, and C. Shen, "Federated Learning with Heterogeneous Quantization Bit Allocation and Aggregation for Internet of Things," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[40] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 22 419–22 430.

[41] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, Sep. 2022.

[42] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proceedings of the 39th International Conference on Machine Learning*. PMLR, Jun. 2022, pp. 27 268–27 286.

[43] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, pp. 11 106–11 115, May 2021.

[44] D. Campos, M. Zhang, B. Yang, T. Kieu, C. Guo, and C. S. Jensen, "LightTS: Lightweight time series classification with adaptive ensemble distillation," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 171:1–171:27, Jun. 2023.

[45] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9881–9893, Dec. 2022.