CAFE and SOUP: Towards Adaptive VDI Workload Prediction

YAO ZHANG, School of Computer Science and Engineering, Southeast University, China and VMware Information Technology (China) Ltd, China
WENPING FAN, VMware Information Technology (China) Ltd, China
QICHEN HAO, VMware Information Technology (China) Ltd, China
XINYA WU, Beijing University of Posts and Telecommunications, China
MIN-LING ZHANG, School of Computer Science and Engineering, Southeast University, China and Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China

For Virtual Desktop Infrastructure (VDI) system, effective resource management is rather important where turning off spare virtual machines would help save running cost while maintaining sufficient virtual machines is essential to secure satisfactory user experience. Current VDI resource management strategy works in a *passive* manner by either reactively driving available capacity based on user demands or following manually configured schedules, which may lead to unnecessary running cost or unsatisfactory user experience. In this paper, we propose a first attempt towards proactive VDI resource management, where two adaptive learning approaches for VDI workload prediction are proposed by learning from multi-grained historical features. For *non-persistent* desktop pool, based on the aggregation session count of pool-sharing users, the CAFE approach induces pool-level workload predictive model by utilizing coarse-to-fine historical features extracted from aggregation workload data. For *persistent* desktop pool, based on the session connection status of individual users within the same pool, the SOUP approach induces user-level workload predictive model by incorporating encoded multi-grained features extracted from logon behavior of individual users into an aggregation pool-level model. Extensive experiments on data sets of real VDI customers and electricity load evidently verify the effectiveness of the proposed adaptive approaches for VDI workload prediction as well as other workload prediction tasks.

CCS Concepts: • **Computing methodologies → Supervised learning**; • **Information systems → Data stream mining**.

Additional Key Words and Phrases: VDI resource management, workload prediction, adaptive modeling, multi-grained features.

ACM Reference Format:
Yao Zhang, Wenping Fan, Qichen Hao, Xinya Wu, and Min-Ling Zhang. 2022. CAFE and SOUP: Towards Adaptive VDI Workload Prediction. *ACM Trans. Intell. Syst. Technol.* 1, 1, Article 1 (January 2022), 29 pages. https://doi.org/10.1145/3529536

Yao Zhang, School of Computer Science and Engineering, Southeast University, Nanjing, China, 210096, and VMware Information Technology (China) Ltd, China; email: yaoz@seu.edu.cn; Wenping Fan, VMware Information Technology (China) Ltd, China; email: wfan@vmware.com; Qichen Hao, VMware Information Technology (China) Ltd, China; email: qhao@vmware.com; Xinya Wu, Beijing University of Posts and Telecommunications, China; email: qwerwxy@bupt.edu.cn; Min-Ling Zhang (corresponding author), School of Computer Science and Engineering, Southeast University, Nanjing, China, 210096, and Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China; email: zhangml@seu.edu.cn.

2157-6904/2022/1-ART1 $15.00
https://doi.org/10.1145/3529536

1 INTRODUCTION

Virtual Desktop Infrastructure (VDI) is a desktop virtualization technology to provide and administer virtual desktops using virtual machines (VMs) hosted on centralized servers. The many benefits offered by VDI including remote access, cost saving, security and centralized management, which make VDI widely adopted in enterprise environments such as remote network, bring your own device (BYOD), task work and shift work. The global VDI market is continuously growing with a prediction of reaching 15.3 billion dollars in 2023 [36]. Traditionally, VDI solution is mostly deployed in on-premise environment where the physical servers and data centers are managed and owned by customers. In recent years, with the maturing of cloud computing, the cloud-based VDI solution is gaining increasing adoption where the virtual machines are handled by public cloud vendors. Because of the much lower initial investment, better resilience and scalability, cloud-based VDI solution is suitable for Small and Midsize Business (SMB) customers and larger enterprise customers with hybrid cloud infrastructure. One of the biggest advantages of such cloud infrastructure is cost-efficiency. Modern cloud vendors can charge per second for resource usage. Therefore, proactive provisioning of resources and power management is of great consequence to the cost and performance of a VDI system.

Building a proactive VDI resource management system is not an easy job. Virtual desktops are managed by desktop pools in VDI systems. Regarding the different requirements of user experience and cost-efficiency, there are usually two kinds of desktop pools: non-persistent pool and persistent pool. The non-persistent pool shares generic desktops among users where the desktop state is reset after the user session is logged off. Non-persistent pool is very cost-efficient and is mostly suitable for repetitive task workers who does not require customized desktop. Currently VDI admins can setup manual power management schedules and reactive provision policies for resource optimization purpose but these approaches are usually error-prone, time-consuming and sometimes cause unnecessary expense and poor user experience. On the contrary, in a persistent pool, each user is allocated with a dedicated desktop where all the changes will be saved after user session is logged off. A persistent desktop is just like a physical desktop belonging to the user which ensures the maximized VDI end user experience but the running cost is higher. Power management of the persistent pool is more challenging as each desktop is unique and needs its own power policy. This leads to the situation that most VDI admins are not willing to power off persistent desktops because its high risk of downgrading user experience due to boot storm and the long time wait for turning on and preparing desktops. To achieve proactive VDI power management, it is essential to build adaptive machine learning models to predict the desktop pool workload. Based on the predictive result of such models, VDI system can always maintain a suitable number of powered-on desktops to save infrastructure cost and improve user experience.

However, VDI workload prediction is a challenging task. Firstly, VDI resource management system is complicated where non-persistent pool and persistent pool are managed in different ways. Accordingly, two different models are needed to handle two types of VDI desktop pools. Secondly, the workload data of VDI desktop pools has very distinctive characteristics. The customers of VDI services are from many different industries. Meanwhile, as a cloud service, VDI was originally designed to provide the maximum flexibility for customer administrators to balance workload among desktop pools. Thus, the VDI desktop pools usually have frequent fluctuations. Moreover, according to the simplicity of reconfiguration and resource management, the lifespan of most VDI desktop pools is usually less than 12 months. So the data size of VDI desktop pool available for model training is usually between thousands to tens of thousands samples depending on the workload monitoring interval. Accordingly, it is difficult to achieve good performance with deep learning techniques. Thirdly, in order to cope with the incoming workload, VDI system needs

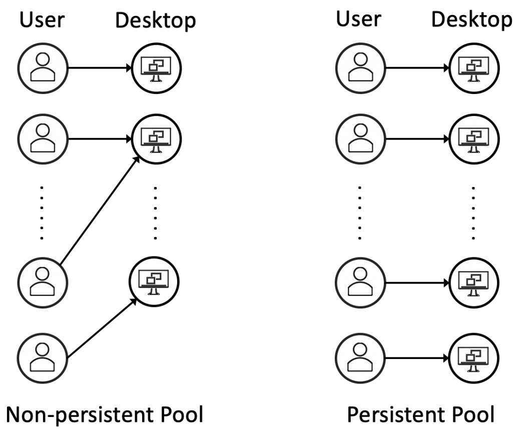


Fig. 1. VDI Non-persistent Pool and Persistent Pool.

about 15 to 30 minutes (the more the better) for preparation in most cases. Therefore, multi-steps prediction is a must which is much more challenging than normal one step prediction. Lastly, VDI workload prediction needs domain-specific evaluation metrics. For example, some VDI customers prefer more over predictions to ensure end user satisfaction rate but others can possible tolerate more under predictions for more cost saving. These differentiations make VDI workload prediction more challenging than common cloud workload prediction tasks.

Because of the challenging parts discussed in above section, the state-of-the-art solutions can not work well on VDI desktop pool workload prediction. In this paper, we introduce CAFE and SOUP, two adaptive learning approaches based on multi-grained features for VDI desktop pool workload prediction. Specifically, CAFE is designed for non-persistent pool where the aggregate session count of the pool is the most effective signal of the workload. Every minute CAFE records the aggregate session count of each non-persistent pool and leverages coarse-to-fine historical descriptions of the historical workload to generate multi-grained features. Correspondingly, the pool workload of the upcoming time span is considered as the target for prediction.[1] For persistent pool workload prediction, aggregate session count is not suitable as every specific user's dedicated desktop has to be managed individually. Accordingly, SOUP model is proposed to predict single user logon behavior accurately by utilizing encoded multi-granularity description. Specifically, we group all the users in the same pool and build a common model for them. For every single user, his/her VDI session connection status in one pool is collected per minute and aggregated every 30 minutes. Then, SOUP generates encoded multi-grained feature by employing coarse-to-fine historical descriptions of the half-hourly session count sequence. Correspondingly, the user's session count in the upcoming 30 minutes is regarded as the target for prediction. It is worth noting that the coarse-to-fine multi-granularity feature extraction used in CAFE and SOUP is an innovative framework to cope with long-term patterns and short-term fluctuations in the VDI workload data. Next, the Gradient Boosting Decision Tree (GBDT) [15, 16] regression models based on ensemble learning techniques are trained from examples formulated by encoded historical sequence with multi-granularity features. Comparative studies with well-defined domain specific metrics on real VDI customer data sets evidently verify the validity of coarse-to-fine multi-grained features for VDI workload prediction. Moreover, extended experiments are conducted on electricity load data prediction where the superior performance of CAFE demonstrates the potential applicability of

[1]In this paper, 30 or 60 minutes is used as the next time span which is practical for adaptive VDI resource management.

coarse-to-fine feature extraction method on other workload prediction and time-series prediction tasks.

The main contributions of this paper are as follows.

- Formalize the VDI workload prediction problem for VDI non-persistent pool and VDI persistent pool respectively. This problem can be viewed as the VDI domain-specific version of cloud service workload prediction problem.
- Propose CAFE, an adaptive learning approach based on coarse-to-fine multi-granularity features for VDI non-persistent pool workload prediction.
- Propose SOUP, an adaptive learning approach based on encoded multi-granularity features for VDI persistent pool workload prediction.
- Evaluate the effectiveness of CAFE and SOUP models by extensive experiments on real-world VDI customer workload data with general evaluation metrics as well as well-defined domain-specific metrics.
- Evaluate the performance of CAFE model on electricity load data and discuss the potential applicability of CAFE and SOUP models on other time-series data prediction tasks.

The rest of this paper is organized as follows. Section 2 gives technical details of the proposed CAFE and SOUP learning approaches. Section 3 discusses related work. Section 4 introduces the collected data sets in real VDI systems for workload prediction and the experimental results of comparative studies. Section 5 discusses the potential applicability of coarse-to-fine feature extraction method to other time-series tasks. Finally, we conclude the paper in Section 6.

2 THE PROPOSED APPROACHES

To provide accurate prediction of VDI workload, the deep understanding of VDI workload is very critical [4]. There are various perspectives to measure the VDI workload and in this paper we interpret the VDI workload from the desktop usage perspective which reflects the actual resource utilization of the customer deployment. Specifically, we use the user session count aggregately or individually to measure the VDI workload and build different predictive models for different types of desktop pools.

2.1 The CAFE Approach

The CAFE approach is designed to predict the workload of non-persistent pool where we choose to use the aggregate session count as the VDI workload indicator based on two considerations. Firstly, the data noise can be mitigated statistically by the effect of aggregating a great deal of individual sessions which could help improve the effectiveness of the learning model. According to [28, 46] and [42], there are similar characteristics in the time-series analysis of smart meter data and tourism data. Secondly, aggregate session count can be modeled only once for each VDI non-persistent pool which is very cost-efficient in view of model training.

2.1.1 Formulation. Firstly, we define the non-persistent pool's workload sequence as $\boldsymbol{x}_{t,n} = (x_t; \ldots; x_{t-n+1})$. Here, x_t denotes the value of the non-persistent pool workload at time t (in minute) and n denotes the workload sequence length. The workload to be predicted at time $t + \Delta t$ is denoted as $x_{t+\Delta t}$. The target is to train a predictive model $f_{\Delta t} : \boldsymbol{x}_{t,n} \mapsto \mathbb{R}$ with which we can acquire the prediction of the future workload $\hat{x}_{t+\Delta t}$ at time $t + \Delta t$ as :

$$\hat{x}_{t+\Delta t} = f_{\Delta t}(\boldsymbol{x}_{t,n}) \tag{1}$$

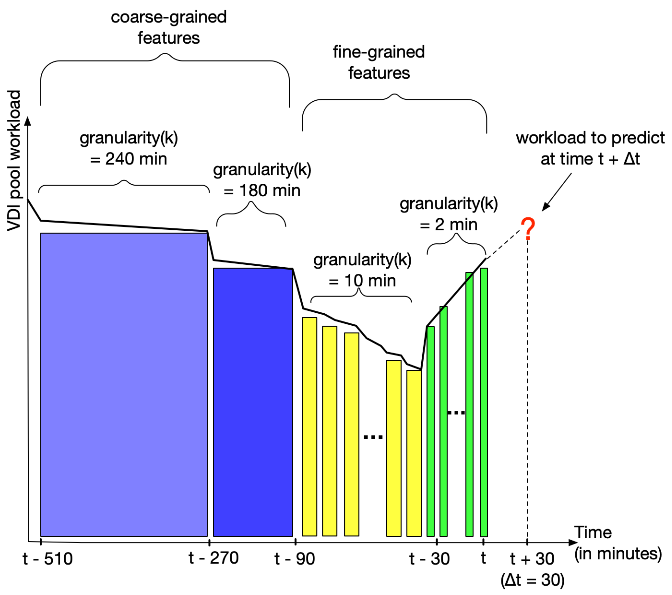


Fig. 2. Schematic diagram of multi-grained feature representation

Moreover, given the the feature vector $\boldsymbol{\mu}_t$ generated from $\boldsymbol{x}_{t,n}$ and a regression function $g(\cdot)$, the predictive model $f_{\Delta t}$ can be instantiated by:

$$f_{\Delta t} = g(\boldsymbol{\mu}_t) \tag{2}$$

In a manner similar to [20], $\boldsymbol{\mu}_t$ is designed to embody three feature vectors:

$$\boldsymbol{\mu}_t = (\boldsymbol{h}_t; \boldsymbol{s}_t; \boldsymbol{c}_t) \tag{3}$$

Here, $\boldsymbol{h}_t$ denotes the historical features in a multi-grained representation. To formulate $\boldsymbol{h}_t$, a coarse-to-fine description transformation $\lambda : \boldsymbol{x}_{t,n} \mapsto \boldsymbol{h}_t$ is employed by CAFE. Furthermore, $\boldsymbol{s}_t$ is the seasonal features obtained from $\boldsymbol{x}_{t,n}$ whilst $\boldsymbol{c}_t$ denotes a set of contextual features.

Formally speaking, given the historical workload sequence of a non-persistent desktop pool: $S = \{x_1, x_2, \ldots, x_T\}$, we can define the training data set $\mathcal{D} = \{(\boldsymbol{\mu}_t, x_{t+\Delta t}) \mid l \leq t \leq T - \Delta t\}$ where l denotes the minimum feasible starting time of $\boldsymbol{x}_{t,n}$ and T denotes the sequence ending time. Finally, by invoking specified regression learning method $\mathcal{L}$ on $\mathcal{D}$, i.e. $g \hookleftarrow \mathcal{L}(\mathcal{D})$, we can induce the regressor $g : \boldsymbol{\mu}_t \mapsto \mathbb{R}$.

2.1.2 *Coarse-to-fine Feature Extraction.*

As mentioned above, feature vector $\boldsymbol{h}_t$ is used to represent the historical pattern of the pool workload sequence $\boldsymbol{x}_{t,n}$. We define a coarse-to-fine multi-granularity representation model $\lambda : \boldsymbol{x}_{t,n} \mapsto \boldsymbol{h}_t$ to describe $\boldsymbol{x}_{t,n}$ in a coarse-to-fine form.

Specifically, let $\boldsymbol{k} = (k_1, \ldots, k_\alpha)$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_\alpha)$ be the granularities and corresponding action scopes in multi-grained representation where α denotes the total number of transformation layer in λ. Each transformation layer contains m_i granules of length k_i and its action scope γ_i is defined by $\gamma_i = k_i * m_i$. Given the input sequence $\boldsymbol{x}_{t,n}$, the historical feature vector $\boldsymbol{h}_t$ can be formulated as:

$$\boldsymbol{h}_t = \lambda(\boldsymbol{x}_{t,n}, \boldsymbol{k}, \boldsymbol{\gamma}) = (\boldsymbol{L}_1, \boldsymbol{L}_2, \ldots, \boldsymbol{L}_\alpha)^T \boldsymbol{x}_{t,n}, \text{where } \boldsymbol{L}_i = [l^i_{pq}]_{n \times m_i}, \; m_i = \frac{\gamma_i}{k_i}. \tag{4}$$

Table 1. $\boldsymbol{k}$ and $\boldsymbol{m}$ in coarse-granularity description

i	1	2	3	4
k_i	n	$\frac{n}{2}$	$\frac{n}{6}$	$\frac{n}{8}$
m_i	1	1	1	1

Table 2. $\boldsymbol{k}$ and $\boldsymbol{m}$ in fine-granularity description

i	5	6	7
k_i	$\frac{\Delta t}{3}$	$\frac{\Delta t}{15}$	1
m_i	6	15	1

Here, $\boldsymbol{L}_i = [l^i_{pq}]_{n \times m_i}$ is the transformation matrix of the i_{th} grain-layer where its elements are defined by mean aggregation as follows:

$$l^i_{pq} = \begin{cases} \frac{1}{k_i}, & (q-1)k_i + 1 \le p \le qk_i, \\ & 1 \le q \le m_i. \\ 0, & otherwise. \end{cases} \tag{5}$$

Figure 2 shows a toy example of the multi-grained feature representation. For the ancient historical data, two coarse granularities ($k = 180\ min$ and $k = 240\ min$) are used to extract features. For the recent historical data, two fine granularities ($k = 2\ min$ and $k = 10\ min$) are used respectively.

In this paper, seven coarse-to-fine grain-layer transformations ($\alpha = 7$) are applied to the pool workload sequence to generate the historical feature vector $\boldsymbol{h}_t$. CAFE employs four *coarse-granularity* layer transformations based on hour-level aggregations. The intent is to eliminate the less-relevant minor fluctuation in the ancient data and focus more on macro pattern extraction to minimize the noise impact during formulating regression model. In these four transformations, we enforce $m = 1$ so that the workload trend is described by a consistent average manner in the whole action scope. Table 1 summarizes the corresponding setting of k_i and γ_i.

On the other hand, *fine-granularity* description excels in grasping the frequent unexpected change in the more recent periods. In this paper, we use three fine-granularity layer transformations by utilizing minute-level aggregations. It is worth noting that, the value of granularity k_i and scope γ_i become smaller when the more recent periods are considered and the prediction time span Δt ($\Delta t \ge 30$) is used as the basic unit to calculate γ_i. The configurations of all three fine-granularity transformations can be found in Table 2.

Seasonal features and contextual features are also important for model learning. From the visualization of training data in Figure 4, the weekly and daily seasonal patterns can be easily captured. This reflects the domain knowledge that the VDI workload pattern could change every few weeks. For example, in the retail domain, the VDI workload climbs in the holiday seasons and drops in the slack seasons. Let d and w denote the maximum days and weeks for seasonal features extraction[2] respectively, we can define the seasonal feature vector $\boldsymbol{s}_t$ as:

$$\boldsymbol{s}_t = (s_{t-1440*1}; \dots, s_{t-1440*d}; s_{t-1440*7*1}; \dots; s_{t-1440*7*w}), \text{where } s_a = \frac{\sum_{i=1}^{\Delta t} x_{a+i}}{\Delta t}. \tag{6}$$

Finally, two static contextual features are considered in CAFE. We use $minute\_of\_the\_day \in$

[2]In this paper, we set $d = 6$ and $w = 5$.

$\{1, 2, \ldots, 1440\}$ to represent the index of the current minute in a day and $day\_of\_the\_week \in \{1, 2, \ldots, 7\}$ to indicate the day of the week. The contexture features can be defined as follows:

$$\boldsymbol{c}_t = (minute\_of\_the\_day, day\_of\_the\_week). \tag{7}$$

2.1.3 Learning Method. We choose GBDT as our learning method $\mathcal{L}$ for following reasons. Firstly, most VDI desktop pools have a medium-length lifespan which is usually less than 12 months. So the available workload data points for training are limited. As a widely used prediction model, GBDT has recorded good performance as well as generalization ability on middle size data set which is a perfect fit for VDI workload data. Secondly, the training time of GBDT is much shorter than deep learning methods. This provides the flexibility of model re-training with newly collected data on demand to increase the prediction accuracy. Last but not least, GBDT has less hype parameters which can save a lot of model tuning effort.

2.2 The SOUP Approach

As discussed in Section 1, VMs in the persistent pool are usually kept running all the time to avoid user's wait for desktop launch, which results in enormous cost for public cloud customers. However, it is not possible to manage the persistent pool wisely without the ability of controlling each VM's power in an adaptive manner. In this subsection, we propose SOUP, a single user logon model which aims to predict whether the specific user will logon in the near future. Based on the prediction results of SOUP, the persistent pool can power on specific desktops for specific users on demand. Consequently, VMs can be shut down safely to save cost with no fear of sacrificing user experience. Intuitively, different models should be built for every individual user for the purpose of logon prediction. However, it is found that a huge amount of VDI users have very limited logons which is inadequate and highly noise sensitive for building an accurate model. To cope with this characteristic of the VDI use logon data, SOUP adopts a group-based model for all the users in one persistent pool. Specifically, the data of users in the same pool are grouped together to build a common model, which coincides with the fact that the users assigned to the same pool have similar behavior patterns. This method can be regarded as a special case of cluster-based aggregate forecasting of time-series data which has been proven that the impact of noise can be mitigated statistically [27]. Furthermore, employing a general model for all the users in a pool is much more efficient and can also deal with the problem of insufficient logon samples.

2.2.1 Formulation. Single user logon prediction is the process of using historical user connection status to predict if this user will logon in the future period. Formally speaking, let Δt be the granularity interval (in minute, $\Delta t \leq 60$) and $x_t \in \{0, 1\}$ indicate if user session has ever connected at time range $[t - \Delta t, t)$. Similar to the CAFE model, the user connection status sequence can be represented as $\boldsymbol{x}_{t,n} = (x_t; \ldots; x_{t-n+1})$ with sequence length n. Let x_{t+1} be the user logon status we want to predict at the future period $[t, t + \Delta t)$. The aim is to learn the predictive model $f_{\Delta t} : \boldsymbol{x}_{t,n} \mapsto \{0, 1\}$, where the future connection status $\hat{x}_{t+1}$ during time $[t, t + \Delta t)$ can be obtained as follows:

$$\hat{x}_{t+1} = f_{\Delta t}(\boldsymbol{x}_{t,n}) \tag{8}$$

Furthermore, the predictive model $f_{\Delta t}$ is instantiated as:

$$f_{\Delta t} = g(\boldsymbol{\mu}_t) \tag{9}$$

where $\boldsymbol{\mu}_t$ is the feature vector generated from $\boldsymbol{x}_{t,n}$ and $g(\cdot)$ is a classification function. Speficically, $\boldsymbol{\mu}_t$ is composed of components from different categories including *historical*, *seasonal* and *contextual* features:

$$\boldsymbol{\mu}_t = (\boldsymbol{h}_t; \boldsymbol{s}_t; \boldsymbol{c}_t) \tag{10}$$

Here, $\boldsymbol{h}_t$ is the historical features extracted from user connection status sequence, $\boldsymbol{s}_t$ denotes the seasonal representation of the data and $\boldsymbol{c}_t$ is the contextual feature vector.

2.2.2 Encoded Multi-grained Features. Different from the practice in CAFE, where historical feature granularity is set according to the time remoteness, SOUP employs coarse-to-fine feature in describing time-series seasonal pattern. Moreover, encoding is applied on the multi-grained description to improve the prediction accuracy.

Let $S_j = \{x_1, x_2, \ldots, x_T\}$ be the full-length connection status sequence of one user j, which in turn leads to data set $\mathcal{D}_j = \{(\boldsymbol{\mu}_t, x_{t+1}) \mid l \leq t \leq T-1\}$. Here, l is the minimum feasible starting time of $\boldsymbol{x}_{t,n}$ and T is the ending time of the sequence. Based on training data set $\mathcal{D} = \cup_{j=1}^{\delta} \mathcal{D}_j$, where δ is the total number of users in one VDI pool, the classifier $g : \boldsymbol{\mu}_t \mapsto \{0,1\}$ can be derived by invoking some classification learning method $\mathcal{L}$ on $\mathcal{D}$, i.e. $g \hookleftarrow \mathcal{L}(\mathcal{D})$.

Following the above explanations, $\boldsymbol{s}_t$ is defined as the seasonal representation of user connection status. From the real customer data, we observe two major seasonal patterns with different intervals: daily and weekly. Let $\boldsymbol{\zeta}_{\psi}^{d}$ be the *daily* feature vector where ψ indicates the seasonal timing with $\psi = t - \frac{1440*days}{\Delta t}$ and $\boldsymbol{\zeta}_{\psi}^{w}$ be the *weekly* feature vector of seasonal timing ψ with $\psi = t - \frac{1440*7*weeks}{\Delta t}$. If we use α and β to denote the day and week count respectively[3] where $\frac{1440*(7\beta+1)}{\Delta t} \leq n$, the seasonal feature vector $\boldsymbol{s}_t$ can be specified as:

$$\boldsymbol{s}_t = (\boldsymbol{\zeta}^{d}_{t-\frac{1440}{\Delta t}}; \ldots; \boldsymbol{\zeta}^{d}_{t-\frac{1440*\alpha}{\Delta t}}; \boldsymbol{\zeta}^{w}_{t-\frac{1440*7}{\Delta t}}; \ldots; \boldsymbol{\zeta}^{w}_{t-\frac{1440*7*\beta}{\Delta t}}). \tag{11}$$

To generate the feature vector $\boldsymbol{\zeta}_{\psi}^{d}$ and $\boldsymbol{\zeta}_{\psi}^{w}$ from input connection status sequence $\boldsymbol{x}_{t,n}$, we introduce model $\lambda_{\psi}^{d} : \boldsymbol{x}_{t,n} \mapsto \boldsymbol{\zeta}_{\psi}^{d}$ and $\lambda_{\psi}^{w} : \boldsymbol{x}_{t,n} \mapsto \boldsymbol{\zeta}_{\psi}^{w}$ to depict the daily and weekly characteristic in coarse-to-fine manner. The transformation defined in these two models follows similar form with diverse parameters. Specifically, we define the common model as λ_{ψ}. Figure 3 gives the concrete example for the model. Let k_i be the length of granule, m_i be the number of granules that are taken into account and $\boldsymbol{e}_i$ be the encoding vector of m_i dimensions which transforms the i_{th} granularity layer feature vector into a numerical value. With the granularity vector $\boldsymbol{k} = (k_1, \ldots, k_{\gamma})$, granule number vector $\boldsymbol{m} = (m_1, \ldots, m_{\gamma})$, and the encoding vectors $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{\gamma})$, the seasonal feature vector $\boldsymbol{s}_t$ can be generated from $\boldsymbol{x}_{t,n}$ as follows:

$$\begin{aligned}\boldsymbol{s}_t &= \lambda_{\psi}(\boldsymbol{x}_{t,n}, \boldsymbol{k}, \boldsymbol{m}, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_{\gamma}) \\ &= ({\boldsymbol{e}_1}^T I({\boldsymbol{L}_{\psi}^{1}}^T \boldsymbol{x}_{t,n}); \ldots; {\boldsymbol{e}_{\gamma}}^T I({\boldsymbol{L}_{\psi}^{\gamma}}^T \boldsymbol{x}_{t,n})), \text{where } \boldsymbol{L}_{\psi}^{i} = [l_{pq}^{i,\psi}]_{n \times m_i}.\end{aligned} \tag{12}$$

$\boldsymbol{L}_{\psi}^{i}$ denotes the i_{th} grain-layer transformation performed on the input sequence $\boldsymbol{x}_{t,n}$ around seasonal time ψ. Here, we utilize sum aggregation with elements of $\boldsymbol{L}_{\psi}^{i}$ specified as follows:

$$l_{pq}^{i,\psi} = \begin{cases} 1, & z + (q-1)k_i + 1 \leq p \leq z + qk_i, \\ & \qquad 1 \leq q \leq m_i, \\ 0, & \qquad otherwise. \end{cases} \qquad \text{where } z = \psi - \lfloor \frac{m_i}{2} \rfloor k_i \tag{13}$$

I is an element-wise indicator function (defined in Eq.(14)) transforming each resulting multi-grained feature v_{ξ} into an indicator value $\{0,1\}$, describing whether user is connected in the granule scope.

[3]In this paper, we set $\alpha = 7$ and $\beta = 4$.

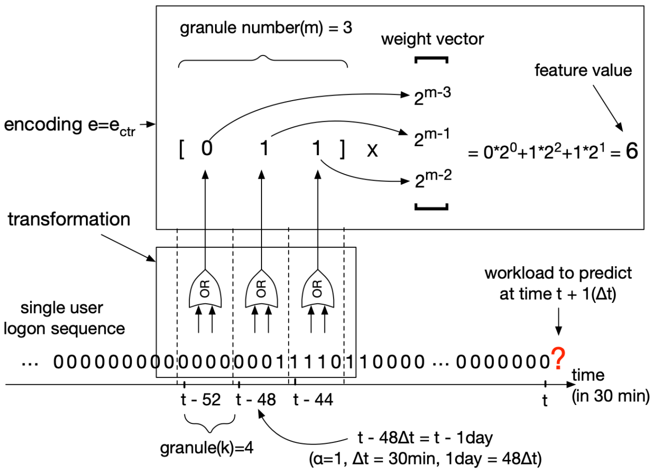


Fig. 3. SOUP seasonal multi-grained feature extraction example for previous 1 day ($\alpha = 1$) of layer 4 ($i = 4$) with granular parameter $k = 4, m = 3$ and central-spread encoding e_{ctr}

Table 3. $\boldsymbol{k}^d$, $\boldsymbol{m}^d$ and $\boldsymbol{e}_i^d$ in daily description with $\gamma^d = 6$

i	1	2	3	4	5	6
k_i^d	1	2	3	4	8	1
m_i^d	3	3	3	3	3	$\lceil \frac{2*mean(std_{user\_logon\_time})}{\Delta t} \rceil$
$\boldsymbol{e}_i^d$	$\boldsymbol{e}_{ctr}$	$\boldsymbol{e}_{ctr}$	$\boldsymbol{e}_{ctr}$	$\boldsymbol{e}_{ctr}$	$\boldsymbol{e}_{ctr}$	$\boldsymbol{e}_{seq}$

$$I(\boldsymbol{v}) = (\mathbb{1}_{v\neq 0}(v_1); \mathbb{1}_{v\neq 0}(v_2); \ldots; \mathbb{1}_{v\neq 0}(v_{m_i})) \\ \text{where } \mathbb{1}_{v\neq 0}(v_\xi) = \begin{cases} 1, & if\ v_\xi \neq 0, \\ 0, & otherwise. \end{cases} \tag{14}$$

The encoding transformation vector $\boldsymbol{e}_i$ has two types: central-spread encoding ($\boldsymbol{e}_{ctr}$) and sequential encoding ($\boldsymbol{e}_{seq}$). The central-spread encoding, which is given in "Encoding" section in Figure 3, emphasizes the grain value at the particular seasonal timing ψ while sequential encoding focuses on the sequential correlation of features generated with the same granularity. These two encodings introduce the temporal correlation of neighboring granularity features from different perspectives, which make the feature more informative. Given the grain number m_i, the two types of encoding vector are customized as follows:

$$\boldsymbol{e}_{ctr}(m_i) = (\ldots; 2^{m_i-3}; 2^{m_i-1}; 2^{m_i-2}; \ldots) \tag{15}$$

$$\boldsymbol{e}_{seq}(m_i) = (2^0; 2^1; \ldots; 2^{m_i-1}) \tag{16}$$

A concrete example of the process of calculating a central-spread encoding feature ($e = e_{ctr}, k = 4, m = 3, \alpha = 1$) is illustrated in Figure 3. Firstly, the data in one granule with length of 120 minutes ($4 \times 30min$) are transformed to the indicator values as in Eq.(14). Then we calculate the dot product of the indicator value vector (formed by 3 indicator values around time $t - \alpha\ day$) and the weight vector of the central-spread encoding e_{ctr}. The final result is the encoded multi-grained feature value of this layer ($i = 4$).

Table 4. $\boldsymbol{k}^w$, $\boldsymbol{m}^w$ and $\boldsymbol{e}_i^w$ in weekly description with $\gamma^w = 2$

i	1	2
k_i^w	1	$\frac{1440}{\Delta t}$
m_i^w	$\lceil \frac{2*mean(std_{user\_logon\_time})}{\Delta t} \rceil$	1
$\boldsymbol{e}_i^w$	$\boldsymbol{e}_{seq}$	$\boldsymbol{e}_{seq}$

Table 5. Contextual feature list for SOUP

Feature	Description
hour_index	index of current hour within day
day_index	index of current day within week
has_session_day	if user has session this day
current_session_count	user's session count in last minute
session_count_mean	user's average session count of day
last_no_session_dur	user's last no session duration in pool

For the daily pattern representation, SOUP employs six coarse-to-fine grain-layer transformations with both central spread and sequential encoding. Firstly, we utilize multi-grained layer transformations with central spread encoding. The aim is to retrieve the elaborate daily pattern of connection status as well as avoid the randomness of user logon time. In addition, we make use of one more fine granularity layer with sequential encoding to further illustrate the sequential correlation of the grained features from nearby range. Accordingly, the granule count is set to standard deviation of users logon time in VDI pool. The grain-layer configuration $\boldsymbol{k}^d$, $\boldsymbol{m}^d$ and encoding vectors $\boldsymbol{e}_i^d$ are summarized in Table 3.

For the weekly pattern representation, SOUP applies hour-level and day-level transformations. The hour-level transformation is to catch the weekly patterns of user connection status that are not included in daily description. For example, users usually logon late on Monday as it's the first workday of one week. In day-level transformations, we would like to extract the regularity that whether user has connected sessions on the same week day in last β weeks. Configurations of $\boldsymbol{k}^w$, $\boldsymbol{m}^w$ and $\boldsymbol{e}_i^w$ are summarized in Table 4.

Except for the seasonal description, SOUP also utilizes historical features to describe the connection status change in short-term. In single user logon model, the predicted value is more likely to be impacted by the connection status in recent period. For example, a user logged off just several minutes ago is less possible to logon immediately. Accordingly, let θ be the short-term interval we want to consider[4], the historical feature vector is set as follows:

$$\boldsymbol{h}_t = (x_t; \boldsymbol{e}_{seq}(\theta)^T(x_t, x_{t-1}, ..., x_{t-\theta+1})) \tag{17}$$

The contextual features selected in SOUP include both static features, such as *hour_index* and *day_index*, and dynamic features generated from input sequence $\boldsymbol{x}_{t,n}$. The detailed features are listed in Table 5.

[4]In this paper, we set $\theta = 3$

Table 6. Power data $\boldsymbol{k}$ and $\boldsymbol{m}$ configuration in fine-granularity description

i	5	6	7
k_i	$\frac{\Delta t}{4}$	$\frac{\Delta t}{8}$	1
m_i	8	8	1

Table 7. Power data $\boldsymbol{k}$ and $\boldsymbol{m}$ configuration in coarse-granularity description

i	1	2	3	4
k_i	n	$\frac{n}{2}$	$\frac{n}{6}$	$\frac{n}{8}$
m_i	1	1	1	1

2.2.3 Learning Method. For the similar reason of limited data size, short training time and less hype parameters, SOUP chooses the same GBDT learning method with CAFE.

2.3 Parameters Tuning

To explore the generalization ability of coarse-to-fine feature extraction method, we summarize the common practice for parameters tuning when dealing with a new cloud workload of time series prediction task.

First of all, we suggest to start from CAFE and SOUP's default parameters listed in Table 1, 2, 3 , 4 if it could fit the data sample interval and data size. From our experience, even if the default parameters can not work well, they can help to lay the foundation for future finer tuning.

If the results of the default parameters are not satisfied, below tips can be used to help further tune the coarse-to-fine transformation parameters:

- Layer number (length of $\boldsymbol{k}$ and $\boldsymbol{m}$): the total layer number is suggested to set in range $[6, 8]$, it applies for both CAFE and SOUP granularity feature to ensure the feature efficiency.
- Action scope ($\gamma = k * m$):
  (1) Use domain knowledge in action scope configuration to judge how long will recent data fluctuations, global trend and seasonal patterns impact the future prediction.
  (2) Limit the historical (non-seasonal) feature action scope within 24 hours. The patterns that occur in previous more than 24 hours should be captured in daily and weekly features.
- Granularity (k):
  (1) Except for the latest data point ($k = 1$), the granularity must be in multiples of the sampling interval.
  (2) Use finer granularities for recent and small action scope and coarser granularities for ancient and big action scope.
- Leverage the learner's capability, such as the feature importance in decision tree based learning algorithms, to adjust the the total granular count and feature count in each granular layer.

To evaluate the generalization ability of the coarse-to-fine feature extraction method, we apply the above parameter tuning practice as well as the whole CAFE approach on an electricity load prediction task. The data sets are from the Europe countries' electricity load data between 2015 and 2020, which is collected every 15 minutes from the energy consumption monitoring system, and the prediction is made for future $\Delta t = 120$ and $\Delta t = 240$ minutes.

In the 120 minutes prediction task ($\Delta t = 120$), we can't achieve CAFE's default finest granular layer settings ($\frac{\Delta t}{15} = 8$ minute) as it's smaller than the sampling interval (15 minutes). Following the rules in granularity settings, the fine granules for the electricity load data are set as $\frac{\Delta t}{4}$ and $\frac{\Delta t}{8}$, which are $1 * 15$ minutes and $2 * 15$ minutes for the prediction when $\Delta t = 120$. In terms of their action scopes, as we have limited domain knowledge in energy consumption field, the action scope employs CAFE's default settings: $2\Delta t$ and Δt. The parameter configuration of the fine granularity feature description is summarized in Table 7. For coarse granularity, we completely follow CAFE's default settings where the max action scope is $n = 24 * 60$ minutes. The $\boldsymbol{k}$ and $\boldsymbol{m}$ configuration of coarse description is given in Table 6 which depicts the global trend of power load during last 24 hours with a consistent average aggregation over the whole action scope.

In Section 5, the dataset and parameters settings are further detailed and the evaluational experiment is conducted by the concrete comparison with other time-series prediction algorithms.

3 RELATED WORK

Generally, the VDI workload prediction problem belongs to the cloud services workload prediction problem. According to [32], the workload of cloud services is the total of work performed by virtual machines demanded by the end-users or applications. As the accurate workload prediction is of crucial importance to improve the operational efficiency, cost and QoS (Quality of Service) satisfaction of cloud services [23], an increasing number of researchers attempt to solve the problem with various statistical and machine-learning models.

Formally speaking, cloud services workload data can be regarded as time series data [31] as it is a collection of observations made chronologically. During the past decade, classical statistical methods have been successfully used in time series prediction [26], to name a few, in [3, 13, 24, 29], authors used Autoregressive Integrated Moving Average (ARIMA) based models for various cloud workload predictions; in [34, 37, 39], holt-winters exponential smoothing [6, 7] models are deployed to predict cloud workload and resource provision; [8] leverages facebook Prophet [20] to predict Microsoft Azure VM workload for cloud resource management. While these statistical methods show advantages when dealing with the data of small size and simple patterns, they can not handle the complex data with long term patterns and short term fluctuation which is commonly seen in cloud workload data. Moreover, statistical methods can not conduct multi-steps forward prediction but have to predict single step several times which results in pool performance in these scenarios.

More recently, machine-learning models are gaining more attention for cloud service workload prediction where both supervised learning models and deep learning models are broadly explored by researchers. A lot of literatures focused on using the mainstream supervised learning models like Support Vector Machine (SVM), Support Vector Regression (SVR) [19] and Random Forest (RF) [43] for cloud workload prediction such as [2, 5, 33, 41, 51]. For instance, in [2], an innovative tuned support vector regression (TSVR) method was proposed for cloud workload prediction. TSVR uses a hybrid genetic algorithm (GA) and particle swarm optimization (PWO) to select SVR parameters and demonstrate good performance on simulated data by Google cloud traces. In [41], Singh et al. proposed a web application workload prediction model with Linear Regression, ARIMA and SVR for different kinds of data sets. Their results showed SVR performs well on short-term non-seasonal workload data. In [5], Cetinski et al. introduced an Advanced Model for Efficient Workload Prediction in the Cloud (AME-WPC) which uses Random Forest method for cloud workload prediction. While these supervised learning models show good performance on several cloud workload prediction scenarios, their limitations are also discussed in different literatures. In [33], authors indicated that the accuracy of SVM depends on the workload data with seasonal and increasing patterns. In [2], it was discussed that SVM and SVR lack of systematic way of parameter tuning.

Deep learning models have the strength on long-term prediction especially when the data size and model layers are big and deep enough. There have been a number of explorations of leveraging deep learning models to predict cloud workload. In [25], Kumar et al. explored using LSTM for workload prediction on three data sets of web server logs and achieved high accuracy. In [17], Gao et al suggested to use m-gap prediction (m steps forward prediction) on cloud workload for better resource management. Their experiments show that LSTM perform better than ARIMA but worse than Bayesian Ridge Regression (BRR). In [38], Ruan et al proposed a workload prediction method (CrystalLP) for cloud storage based on LSTM. The experiments on the I/O data of a search engine demonstrate LSTM outperform Arima, SVR and simple RNN. Meanwhile, the disadvantages of deep learning models for cloud workload data prediction have also been extensively discussed in many literatures. For example, in [17], authors indicated that LSTM has a good fit of the non-linearities but does not perform well on short-term task when data size is not big enough. In [35], authors discussed that the training time of LSTM is much longer than other approaches. It is also reported in some literatures [18] that LSTM should only be used if statistical method fails on time-series data prediction.

Besides exploring the different learning models, several researchers also investigated the effective feature extraction methods for cloud workload prediction. For example, in [5], authors introduced a Two-phase Pattern Matching (TPM) method to find similar daily patterns in the training data. But longer term patterns like monthly patterns are not discussed. In [40], Shishira et al. proposed a Feature Extraction Model (FEM) to extract labels features from raw cloud workload data. However, there has not been any attempt of leveraging coarse-to-fine multi-granularity methods for feature extraction.

VDI workload prediction is an emerging domain-specific cloud services workload prediction problem. The majority of the state-of-the-art researches still focus on VDI workload analysis while very few literatures on VDI workload prediction are available. In [48], Xu et al. discussed the virtual desktop user workload on simulated data. In [14], characteristics of desktop workload on single server are discussed in detail. In [4], authors analyze the important information of VDI users behavior, VDI CPU usage and VDI storage I/O from a real-world cloud virtual desktop service. In [50], Yao et al. proposed CAFE which uses multi-grained feature and GBDT to predict non-persistent VDI workload adaptively which achieve superior performance against other methods. In [12], Fan et al. introduced a Multi-instance Multi-label based approach to prediction the VDI single user workload and verify its effectiveness on real VDI customer data.

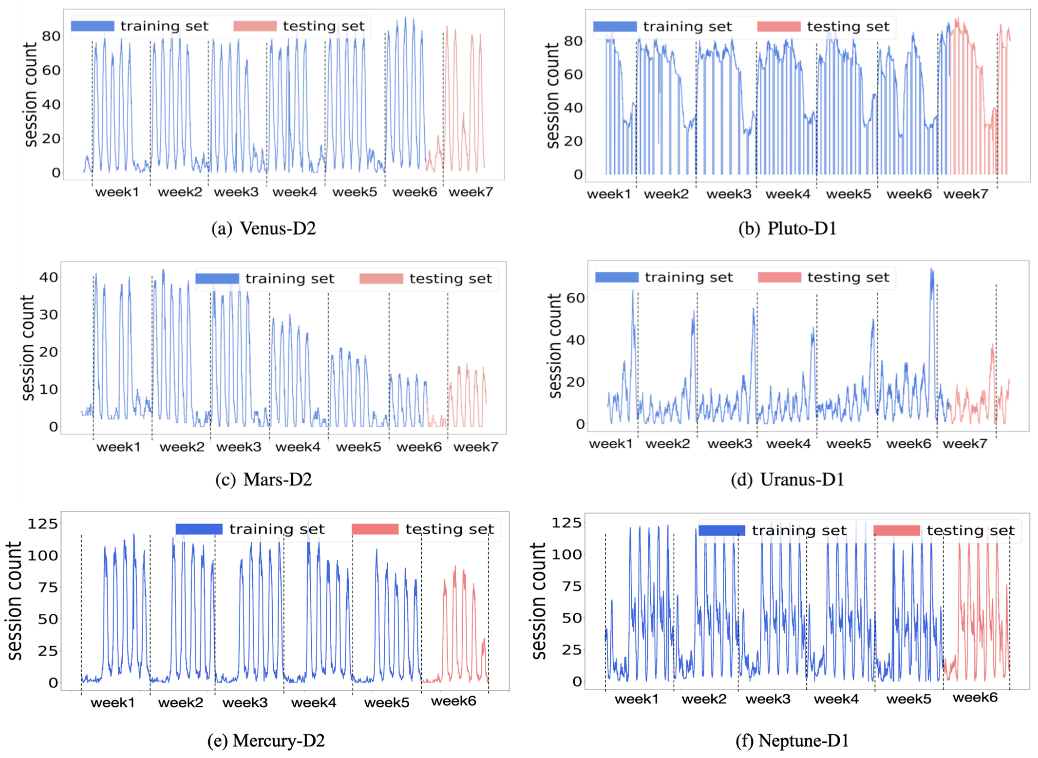
4 EXPERIMENTS ON VDI WORKLOAD PREDICTION

4.1 CAFE Experiments

4.1.1 Data Sets. In this paper, we use the real VDI customer data for experiments. For CAFE, the collected data sets include 6 non-persistent VDI pools from 6 different customers over several months across 2018 and 2019. The number of samples in the training data is from 51840 to 59040. The data of each pool is divided into two divisions where every division consists of training set of 5 or 6 weeks' customer data and test set of 1 week's customer data. The twelve data sets are named as `Venus-D1`, `Venus-D2`, `Uranus-D1`, `Uranus-D2`, `Pluto-D1`, `Pluto-D2`, `Mars-D1`, `Mars-D2`, `Mercury-D1`, `Mercury-D2`, `Neptune-D1` and `Neptune-D2` respectively. Table 8 shows the detailed statistics of the twelve data sets, where `min-max` and `mean±std` refer to the minimum/maximum session count, mean value and standard deviation of the session counts in the VDI pools.

Table 8. Characteristics of the real-world VDI non-persistent pool data sets where `min-max` and `mean±std` stand for the minimum/maximum session count, mean value and standard deviation of the session counts respectively.

Data sets	Training set			Testing set		
	Period	min-max	mean±std	Period	min-max	mean±std
Venus-D1	1/06-10/07, 2018	0-82	26.26±26.66	11/07-17/07, 2018	0-85	27.09±27.35
Uranus-D1	1/06-10/07, 2018	0-74	11.63±11.81	11/07-17/07, 2018	0-38	9.59±7.61
Pluto-D1	1/06-10/07, 2018	0-91	51.15±27.88	11/07-17/07, 2018	0-94	56.01±32.29
Mars-D1	1/06-10/07, 2018	0-110	17.81±19.36	11/07-17/07, 2018	0-39	12.26±13.51
Mercury-D1	11/02-17/03, 2019	0-141	35.60±40.76	18/03-24/03, 2019	0-123	33.33±40.55
Neptune-D1	14/07-18/08, 2018	0-126	43.11±33.75	19/08-25/08, 2018	0-124	42.04±33.96
Venus-D2	1/07-10/08, 2018	0-91	28.35±28.09	11/08-17/08, 2018	0-86	26.09±26.83
Uranus-D2	1/07-10/08, 2018	0-74	13.09±12.24	11/08-17/08, 2018	0-61	10.76±11.46
Pluto-D2	1/07-10/08, 2018	0-99	52.01±30.98	11/08-17/08, 2018	0-85	46.14±29.35
Mars-D2	1/07-10/08, 2018	0-42	9.92±11.41	11/08-17/08, 2018	0-17	4.72±5.41
Mercury-D2	09/03-12/04, 2019	0-128	32.23±37.90	13/04-19/04, 2019	0-92	23.24±19.24
Neptune-D2	08/10-11/11, 2018	0-144	48.24±36.24	12/11-18/11, 2018	3-139	48.35±37.85



(a) Venus-D2 (b) Pluto-D1

(c) Mars-D2 (d) Uranus-D1

(e) Mercury-D2 (f) Neptune-D1

Fig. 4. Ground-truth workload of the real-world VDI non-persistent pool data sets.

Figure 4 shows the ground-truth workload of the six different pools in which we can observe different patterns. The rich diversity of the real-world data sets provide a solid basis for comprehensive experimental studies.

- `Venus` is from a traditional national enterprise customer of energy domain. `Venus-D1` has 57600 samples in the training set and 10080 samples in the testing set. `Venus-D2` has 59040 samples in the training set and 10080 samples in the testing set. Both data sets show clear working day pattern where the working days (from Monday to Friday) have much higher workload than the weekend (from Saturday to Sunday). This conforms to the usual working schedule of the traditional enterprise. Additionally, the testing set of `Venus-D2` has an unusual lower workload peak on Wednesday.
- `Uranus` is from a university customer. `Uranus-D1` has 57600 samples in the training set and 10080 samples in the testing set. `Uranus-D2` has 59040 samples in the training set and 10080 samples in the testing set. We can see the obvious a weekend pattern in `Uranus` which has higher workload in weekend(especially in the Sunday) than in working days. Moreover, it is worth noting that the testing set of `Uranus-D1` has an uncommon lower workload peak on Sunday.
- `Pluto` is from a customer of construction domain. `Pluto-D1` has 57600 samples in the training set and 10080 samples in the testing set. `Pluto-D2` has 59040 samples in the training set and 10080 samples in the testing set. `Pluto` has a normal working day pattern but it shows a special characteristic that the workload change of drops and climbs in a vigorous way.
- `Mars` is from a large customer of service provider enterprise domain. `Mars-D1` has 57600 samples in the training set and 10080 samples in the testing set. `Mars-D2` has 59040 samples in the training set and 10080 samples in the testing set. Besides the common working day patter, `Mars` shows a drift pattern whose average workload declines a little week by week in `Mars-D2`.
- `Mercury` is from a customer of automative domain. Both `Mercury-D1` and `Mercury-D2` has 50400 samples in the training set and 10080 samples in the testing set. `Mercury` shows clear working day pattern and it workload change pattern is very smooth where the session count shifts regularly and slowly. Furthermore, testing set of `Mercury-D2` has unusual lower workload peak on Friday.
- `Neptune` is from a customer of transportation and logistics domain. `Neptune-D1` has 50400 samples in the training set and 10080 samples in the testing set. `Neptune-D2` has 48960 samples in the training set and 10080 samples in the testing set. `Neptune` has a working day pattern with the speciality of double workload peak where the second session count peak has about half workload of the first peak. Additionally, a notable higher peak on Friday is observed in the testing set of `Neptune-D1`.

4.1.2 Comparing Approaches. In CAFE experiments, four state-of-the-art time-series data prediction approaches are selected as the comparison methods including Prophet [44], Holt-Winters [7], NF-GBDT and LSTM. Prophet can automatically detect seasonalities and excels at the prediction of large scale time-series data. Holt-Winter is a traditional exponential smoothing model based statistical algorithm for time-series data forecasting. NF-GBDT is the method that applies GBDT model directly on naive time-series features [28]. Here, naive features refer to the way of using raw time-series data as the historical features. Accordingly, we can regard NF-GBDT as a degenerated version of CAFE where multi-grained features are not employed.[5] The comparison with NF-GBDT serves to show the advantages of multi-grained features used in CAFE. LSTM has been used in

[5]CAFE utilizes GBDT as the regression method.

many cloud workload prediction problems and perform well on fitting long term patterns on large data sets.

Based on the domain knowledge, two different time spans: $\Delta t = 30$ minutes and $\Delta t = 60$ minutes are chosen for the CAFE experiments where smaller Δt is useful during real-time monitoring which needs immediate mitigation actions and bigger Δt provides more value to scenarios like boot storm detection and overall capacity optimization. With the maximum action scope n set as 1440 minutes (24 hours), the granularity vector $\boldsymbol{k}$ is then set as $(1440, 720, 240, 180, 10, 2, 1)$ for $\Delta t = 30$ minutes and $(1440, 720, 240, 180, 20, 4, 1)$ for $\Delta t = 60$ minutes. Correspondingly, the action scope vector $\boldsymbol{\gamma}$ is configured as $(1440, 720, 240, 180, 60, 30, 1)$ and $(1440, 720, 240, 180, 120, 60, 1)$ for $\Delta t = 30$ minutes and $\Delta t = 60$ minutes respectively. To better utilize the weekly seasonly pattern, the seasonal parameter of Holt-Winters is set as 168 (24*7) hours. For Nf-gbdt and LSTM, 168-dimensional naive features $(max_{t-168*60+1 \le i \le t-167*60}\, x_i,\ \ldots, max_{t-60+1 \le i \le t}\, x_i)$ are used for the reason that maximum workload is more important for VDI capacity planning.

4.1.3 Evaluation Metrics. In the CAFE experiment, we employ four evaluation metrics in total. Two of them are commonly used regression metrics: *Normalized Mean Absolute Error* (NMAE) and *Normalized Root Mean Square Error* (NRMSE). The other two are customized metrics based on domain knowledge: *Over Prediction Rate* (OPR) and *Under Prediction Rate* (UPR). Following is the definition of each metric where we use $S = (s_1, \ldots, s_m)$ as the ground-truth workload sequence and $\hat{S} = (\hat{s}_1, \ldots, \hat{s}_m)$ as the predicted workload sequence:

$$\mathrm{NMAE}(S, \hat{S}) = \frac{\mathrm{MAE}(S, \hat{S})}{\|S\|_1} = \frac{\sum_{t=1}^{m} |s_t - \hat{s}_t|}{\sum_{t=1}^{m} |s_t|} \tag{18}$$

$$\mathrm{NRMSE}(S, \hat{S}) = \frac{\mathrm{RMAE}(S, \hat{S})}{\|S\|_2} = \sqrt{\frac{\sum_{t=1}^{m} (s_t - \hat{s}_t)^2}{\sum_{t=1}^{m} {s_t}^2}} \tag{19}$$

$$\mathrm{OPR}(S, \hat{S}) = \frac{\sum_{t=1}^{m} |s_t - \hat{s}_t| \times \frac{sign(\hat{s}_t - s_t)+1}{2}}{\sum_{t=1}^{m} |s_t|} \tag{20}$$

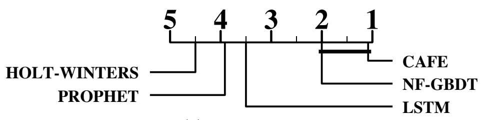
$$\mathrm{UPR}(S, \hat{S}) = \frac{\sum_{t=1}^{m} |s_t - \hat{s}_t| \times \frac{sign(s_t - \hat{s}_t)+1}{2}}{\sum_{t=1}^{m} |s_t|} \tag{21}$$

Here, $sign(\cdot)$ represents the signed function. Generally speaking, NMAE and NRMSE are more scale-robust to the values of the prediction than MAE and RMSE. OPR and UPR are introduced from the VDI domain practice. OPR is a measurement of resource waste by considering the cases that the predicted values are greater than the ground-truth ones, which results in more desktops powered on than needed. On the contrary, UPR measures the impact to the user experience by considering the cases that the predicted values are less than the ground-truth ones, which causes the shortage of powered on desktops and users have to wait new resource provision. OPR and UPR can clearly indicate the impact of resource waste and bad user experience from the workload prediction. To build a feasible VDI power management system, OPR and UPR need to be carefully balanced based on different customer requirements.

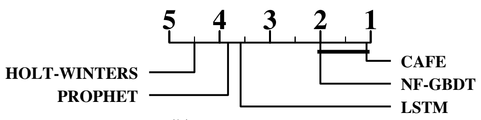
4.1.4 Experimental Results. The experimental results of CAFE are summarized in Table 9 and Table 10 respectively where the best performance on each data set is shown in boldface. Furthermore, we employ the *Friedman test* [10] for statistical analysis which can systematically reveal the relative performance among the comparing approaches.

Table 9. Performance on the real-world VDI data sets in term of NMAE, NRMSE, OPR and UPR (Δt = 30).

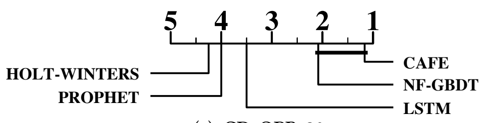
Data Sets	NMAE					NRMSE				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Venus-D1	**0.0625**	0.2981	0.8134	0.0969	0.2363	**0.0765**	0.2702	0.6779	0.1185	0.2230
Venus-D2	**0.0953**	0.4131	0.7693	0.1764	0.2984	**0.1189**	0.3704	0.6869	0.2268	0.3838
Uranus-D1	**0.1224**	0.4991	0.6282	0.2595	0.7679	**0.1473**	0.4836	0.7010	0.2679	0.9280
Uranus-D2	**0.1043**	0.4030	0.8191	0.1864	0.3653	**0.1048**	0.3767	0.7211	0.1709	0.3330
Pluto-D1	**0.0859**	0.2324	0.4404	0.1284	0.2880	**0.1321**	0.2791	0.4607	0.1757	0.3107
Pluto-D2	0.0775	0.2464	0.3490	**0.0640**	0.1310	0.1152	0.2791	0.3696	**0.0747**	0.1532
Mars-D1	**0.0714**	0.2611	1.2227	0.1564	0.4687	**0.0728**	0.2721	1.0293	0.1913	0.4465
Mars-D2	**0.1260**	0.3331	0.9367	0.3081	0.3653	**0.1332**	0.3251	0.8201	0.3100	0.3394
Mercury-D1	**0.0938**	0.5029	0.2171	0.1226	0.1657	**0.0882**	0.3823	0.2183	0.1086	0.1415
Mercury-D2	**0.1967**	0.5029	0.6193	0.2272	0.2707	**0.2505**	0.3823	0.4899	0.2651	0.3065
Neptune-D1	**0.0749**	0.2623	0.2368	0.1108	0.1264	**0.0844**	0.2691	0.2589	0.1166	0.1390
Neptune-D2	**0.0950**	0.2808	0.1055	0.1070	0.1467	**0.0975**	0.2844	0.1127	0.1190	0.1505
Data Sets	OPR					UPR				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Venus-D1	**0.0287**	0.1705	0.5877	0.0381	0.1186	**0.0338**	0.1277	0.2257	0.0588	0.1177
Venus-D2	**0.0699**	0.2902	0.2045	0.1287	0.2506	**0.0255**	0.1229	0.5648	0.0477	0.0479
Uranus-D1	**0.0843**	0.3360	0.5258	0.2110	0.7493	**0.0381**	0.1631	0.1024	0.0485	0.1862
Uranus-D2	**0.0607**	0.2398	0.6906	0.1280	0.2813	**0.0435**	0.1632	0.1285	0.0584	0.0839
Pluto-D1	**0.0267**	0.1140	0.3634	0.0651	0.0903	**0.0592**	0.1184	0.0770	0.0633	0.1977
Pluto-D2	**0.0211**	0.1232	0.2873	0.0350	0.0941	0.0564	0.1232	0.0617	**0.0290**	0.0369
Mars-D1	**0.0514**	0.2231	1.1949	0.0570	0.2462	**0.0199**	0.0380	0.0278	0.0994	0.2225
Mars-D2	0.0567	0.2090	0.6985	**0.0281**	0.1264	**0.0693**	0.1242	0.2382	0.2800	0.2389
Mercury-D1	**0.0600**	0.2149	0.1291	0.0903	0.1139	0.0338	0.2880	0.0880	**0.0323**	0.0518
Mercury-D2	**0.1587**	0.2149	0.2634	0.1758	0.2184	**0.0379**	0.2880	0.3559	0.0515	0.0523
Neptune-D1	**0.0022**	0.6344	0.1176	0.0325	0.0234	**0.0531**	0.1988	0.1192	0.0783	0.1030
Neptune-D2	0.0547	0.1549	**0.0408**	0.0710	0.1169	0.1048	0.1259	0.0647	**0.0360**	0.2977



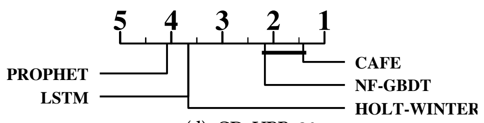
(a) CD-NMAE-30



(b) CD-NRMSE-30



(c) CD-OPR-30



(d) CD-UPR-30

Fig. 5. Comparison of CAFE (control algorithm) against other comparing approaches with the Bonferroni-Dunn test (Δt = 30). Approaches not connected with CAFE in the CD diagram are considered to have significantly different performance from the control algorithm (CD=1.6125 at 0.05 significance level).

Table 10. Performance on the real-world VDI data sets in term of NMAE, NRMSE, OPR and UPR (Δt = 60).

Data Sets	NMAE					NRMSE				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Venus-D1	**0.0760**	0.2835	0.1011	0.1085	0.2478	**0.0758**	0.2600	0.0969	0.1140	0.2271
Venus-D2	**0.1589**	0.4062	0.2435	0.2045	0.3324	**0.2005**	0.3716	0.3053	0.2635	0.3917
Uranus-D1	**0.1639**	0.5111	0.4064	0.3321	0.8049	**0.1833**	0.4935	0.4529	0.3440	0.9074
Uranus-D2	**0.1463**	0.4074	0.2426	0.2187	0.4812	**0.1558**	0.3839	0.2092	0.2008	0.4181
Pluto-D1	**0.1244**	0.2255	0.1705	0.1647	0.3002	**0.1687**	0.2711	0.2567	0.2108	0.3170
Pluto-D2	**0.0885**	0.2232	0.2776	0.0961	0.1479	**0.1018**	0.2728	0.3655	0.1053	0.1782
Mars-D1	**0.0982**	0.2270	0.2296	0.1823	0.5670	**0.0957**	0.2418	0.2165	0.2324	0.4878
Mars-D2	**0.1550**	0.5351	1.8100	0.3767	0.3864	**0.1583**	0.5392	1.9407	0.3721	0.3298
Mercury-D1	**0.0862**	0.5053	0.2088	0.1392	0.2601	**0.0812**	0.3822	0.2128	0.1178	0.2055
Mercury-D2	**0.1907**	0.5966	0.5490	0.2565	0.3076	**0.2496**	0.4927	0.4774	0.2904	0.3192
Neptune-D1	**0.0749**	0.2577	0.2209	0.1071	0.2940	**0.0844**	0.2656	0.2407	0.1121	0.2853
Neptune-D2	**0.0910**	0.2752	0.0999	0.1388	0.2437	**0.0932**	0.2798	0.1052	0.1436	0.2288
Data Sets	OPR					UPR				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Venus-D1	**0.0305**	0.1568	0.0654	0.0441	0.1409	0.0455	0.1267	**0.0357**	0.0644	0.1069
Venus-D2	**0.1101**	0.3024	0.1608	0.1508	0.3007	0.0489	0.1038	0.0827	0.0537	**0.0316**
Uranus-D1	**0.1237**	0.3820	0.3310	0.2821	0.7854	0.0402	0.1291	0.0754	0.0499	**0.0195**
Uranus-D2	**0.0864**	0.2626	0.1987	0.1368	0.3657	0.0599	0.1448	**0.0439**	0.0819	0.1155
Pluto-D1	**0.0396**	0.0881	0.1127	0.0802	0.1092	0.0848	0.1374	**0.0578**	0.0844	0.1910
Pluto-D2	**0.0330**	0.1274	0.2448	0.0469	0.1010	0.0555	0.0958	**0.0329**	0.0492	0.0469
Mars-D1	0.0686	0.1748	0.1425	**0.0438**	0.3675	**0.0296**	0.0522	0.0872	0.1385	0.1994
Mars-D2	0.0679	0.4378	1.7545	**0.0611**	0.1956	0.0871	0.0973	**0.0554**	0.3156	0.1907
Mercury-D1	**0.0538**	0.2168	0.1283	0.0920	0.1523	**0.0324**	0.2883	0.0804	0.0472	0.1078
Mercury-D2	**0.1556**	0.2660	0.2990	0.1992	0.2330	**0.0351**	0.3304	0.2500	0.0572	0.7463
Neptune-D1	**0.0022**	0.6590	0.1086	0.0347	0.1231	**0.0531**	0.1918	0.1124	0.0724	0.1710
Neptune-D2	0.0490	0.1544	**0.0474**	0.0858	0.1663	0.0978	0.1208	**0.0525**	0.0530	0.0775

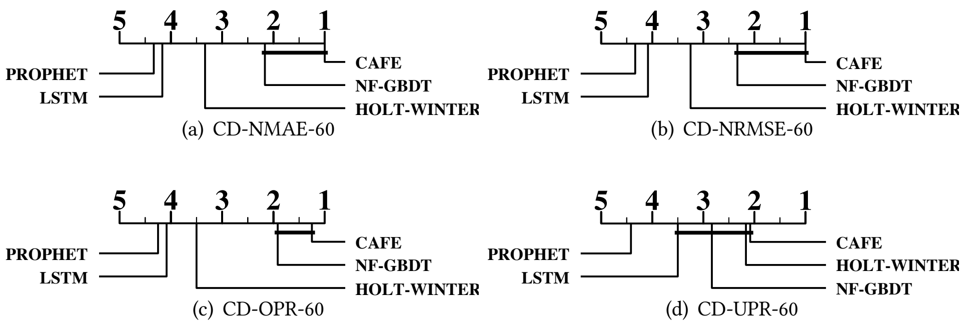
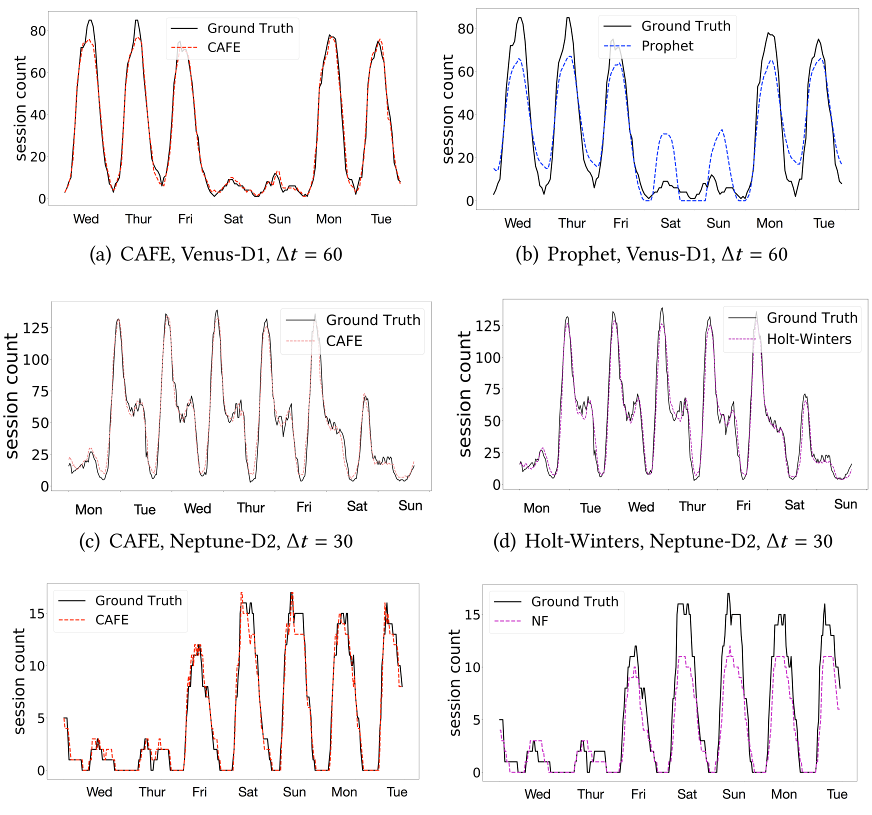


Fig. 6. Comparison of CAFE (control algorithm) against other comparing approaches with the Bonferroni-Dunn test (Δt = 60). Approaches not connected with CAFE in the CD diagram are considered to have significantly different performance from the control algorithm (CD=1.6125 at 0.05 significance level).



(a) CAFE, Venus-D1, $\Delta t = 60$

(b) Prophet, Venus-D1, $\Delta t = 60$

(c) CAFE, Neptune-D2, $\Delta t = 30$

(d) Holt-Winters, Neptune-D2, $\Delta t = 30$

(e) CAFE, Mars-D2, $\Delta t = 30$

(f) NF-GBDT, Mars-D2, $\Delta t = 30$

Fig. 7. Illustrative prediction results of CAFE and comparing approaches on several data sets over one week.

Table 11 and Table 12 report the Friedman statistics F_F and the corresponding critical values in terms of each evaluation metric for $\Delta t = 30$ and $\Delta t = 60$ respectively. It is obvious that the null hypothesis of equal performance is rejected at 0.05 significance level. Accordingly, post-hoc Bonferroni-Dunn test [11] is performed to compare the relative performance among the comparing approaches. The CD diagrams are presented in Figure 5 and Figure 6 for $\Delta t = 30$ and $\Delta t = 60$ respectively, where the average rank of each approach is marked along the axis (the smaller the better). Based on the reported experiment results, the following observations can be made:

- Out of a total of 96 configurations (24 data sets $\times$ 4 evaluation metrics), CAFE achieves 1st ranks in 78 (81.3%) cases and 2nd ranks in 13 (13.5%) cases. In the prediction of the shorter time span ($\Delta t = 30$), CAFE ranks 1st in 41 (85.4%) and 2nd in 5 out of 48 (10.4%) cases. In the prediction of the longer time span prediction ($\Delta t = 60$), CAFE ranks 1st in 37 (77.0%) and 2nd in 8 out of 48 (16.7%) cases. It is impressive that CAFE achieves 1st rank in all cases in terms of NMAE and NRMSE when $\Delta t = 60$.
- It is remarkable that CAFE achieves the lowest average rank in terms of all evaluation metrics, except on UPR with $\Delta t = 60$.
- From the results shown in Figure 7, it is worth mentioning that CAFE achieves promising generalization performance on diverse data sets: a) On data set `Venus-D1` ($\Delta t = 60$) as shown

Table 11. Friedman statistics F_F in terms of each evaluation metric and the critical value at 0.05 significance level (# comparing algorithms k = 5, # data sets N = 12, Δt = 30).

Evaluation metric	F_F	critical value
NMAE	44.38	2.5837
NRMSE	42.88	
OPR	25.84	
UPR	12.23	

Table 12. Friedman statistics F_F in terms of each evaluation metric and the critical value at 0.05 significance level (# comparing algorithms k = 5, # data sets N = 12, Δt = 60).

Evaluation metric	F_F	critical value
NMAE	42.15	2.5837
NRMSE	32.28	
OPR	28.6	
UPR	6.8	

in Figure 7(a) and 7(d)), CAFE demonstrates far better generalization performance than Prophet on Saturday and Sunday workload prediction; b) On data set `Neptune-D2` (Δt = 30) as shown in (Figure 7(b) and 7(e)), CAFE shows superior performance on fitting the fine granularity change of the peak workload on Wednesday while Holt-Winters does not adapt to the frequent change very well; c) On data set `Mars-D2` (Figure 7(c) and 7(f)), CAFE shows its significant capability of balancing coarse and fine granularity change, while Nf-gbdt severely under-predicts the peak workload.

4.2 SOUP Experiments

4.2.1 Data Sets. SOUP's experimental data sets are collected from four persistent pools of different customers from Aug. 15th to Apr. 17th, 2019. For each pool, the data is divided into two divisions. Each division uses 10 weeks' data as training set and succeeding 1 week's data as testing set. The eight data sets are named as `Cocoa-D1`, `Cocoa-D2`, `Avocado-D1`, `Avocado-D2`, `Orange-D1`, `Orange-D2`, `Lemon-D1` and `Lemon-D2` respectively.

Table 13 shows details of the eight data sets, where *session duration mean*, *logon time entropy mean* and *session count mean* denote the average value of the corresponding statistics of users. Through the statistics in Table 13, it is obvious that the data set of different customers exhibit different patterns. Due to these diverse properties, the selected real-world data sets serve as a solid basis for evaluating the effectiveness of comparing approaches.

- `Cocoa` is from a large customer of service provider enterprise domain. `Cocoa-D1`, `Cocoa-D2`, `Cocoa-D3`, `Cocoa-D4` have 432000, 414000, 352800, 344448 samples in training set and 32008, 34963, 28602, 29187 samples in testing set respectively. For the four `Cocoa` data sets, a considerable user count can be observed where each user logon regularly on working day and keep the session during working hours.
- `Avocado` is from a customer of service industry. `Avocado-D1`, `Avocado-D2`, `Avocado-D3`, `Avocado-D4` have 201600, 205200, 178080, 182160 samples in training set and 16388, 17891, 16123, 16846 samples in testing set respectively. `Avocado` has a small number of assigned users with shorter average session duration.

Table 13. Characteristics of the real-world VDI data sets where *session duration mean*, *logon time entropy mean* and *session count mean* are user-related statistics

Data sets	User count	Session duration mean(hr)	Logon time entropy mean	Session count mean	Training period	Testing period
Cocoa-D1	120	5.88	1.54	58.00	15/08/2018 – 28/10/2018	29/10/2018 – 04/11/2018
Avocado-D1	56	3.15	1.61	58.79		
Orange-D1	102	10.87	1.12	11.05		
Lemon-D1	91	88.99	1.06	10.98		
Cocoa-D2	115	5.79	1.72	65.25	10/10/2018 – 23/12/2018	24/12/2018 – 30/12/2018
Avocado-D2	57	3.43	1.71	49.07		
Orange-D2	90	10.23	1.17	10.84		
Lemon-D2	98	118.01	1.21	14.15		
Cocoa-D3	105	5.69	1.57	63.00	09/12/2018 – 16/02/2019	17/02/2019 – 23/02/2019
Avocado-D3	53	3.89	1.61	44.85		
Orange-D3	79	12.09	1.07	10.91		
Lemon-D3	93	220.43	1.19	17.09		
Cocoa-D4	104	5.67	1.76	57.94	01/02/2019 – 10/04/2019	11/04/2019 – 17/04/2019
Avocado-D4	55	3.43	1.80	48.42		
Orange-D4	75	10.95	1.04	9.35		
Lemon-D4	95	142.43	1.25	17.49		

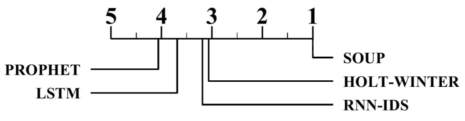
- `Orange` belongs to a customer from basic industry. `Orange-D1`, `Orange-D2`, `Orange-D3`, `Orange-D4` have 367200, 324000, 265440, 248400 samples in training set and 31749, 29059, 24345, 23765 samples in testing set respectively. Different with `Cocoa` and `Avocado`, `Orange`'s average user session count is much less than the number of working days. It indicates that most users just have 1 session in a week.
- `Lemon` is from a manufacturing industry customer. `Lemon-D1`, `Lemon-D2`, `Lemon-D3`, `Lemon-D4` have 327600, 352800, 312480, 314640 samples in training set and 18235, 25849, 20215, 20566 samples in testing set respectively. `Lemon` has the similar pattern that users have fewer sessions that the number of working days. Furthermore, its average user session duration is as long as 118 hours, this is a commonly seen 24 shift pattern in manufacturing enterprise where three teams work 24-hr shifts to provide the whole day coverage.

4.2.2 Comparing Approaches. In the experiment, we compare SOUP with four approaches including Prophet [44], Holt-Winters [7], RNN-IDS [49] and LSTM [21]. Prophet is proven efficient in large scale time-series forecasting with the ability of detecting seasonalities automatically. Holt-Winters is a traditional statistical method for seasonal time-series data prediction using an exponential smoothing model. RNN-IDS is an intrusion detection system based on recurrent neural network which achieves high accuracy and superior performance in time-series prediction task. LSTM (Long Short-Term Memory network) is a type of recurrent neural network used in deep learning. It's designed to handle sequence dependence and is good at time-series prediction problems. These approaches investigate the time series from various aspects, thereby help to verify the effectiveness of SOUP in VDI user login prediction.

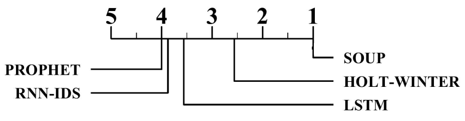
The experiments are performed with granularity interval $\Delta t = 30$ minutes and maximum action scope $n = 1344$ ($\frac{60}{30} * 24 * 7 * 4$) which looks into previous 4 weeks historical data for pattern

Table 14. The value of $\frac{b}{a}$ used in CSAG for four persistent pool datasets. Specially, a and b correspond to the cost of VM charges and the cost of employee's time wasted respectively.

Dataset	Cocoa	Avocado	Orange	Lemon
$\frac{b}{a}$	160	80	120	320



(a) CD-AUC



(b) CD-CSAG

Fig. 8. Comparison of SOUP (control algorithm) against other comparing approaches with the Bonferroni-Dunn test, CD=1.3964 at 0.05 significance level.

extraction. For SOUP, GBDT is utilized as the regression method to learn model from the training samples. For Prophet, we enable the daily and weekly seasonality as they are both observed in the data. For Holt-Winters, the seasonal parameter is configured as 336 ($\frac{60}{30} * 24 * 7$) hours. For LSTM, we specify the input shape of 3D tensor $(batch\_size, timesteps, input\_dim)$ as $(256, 72, 1)$ which achieves best AUC score on the data sets. For evaluation, we exclude the "$x_t = 1$" samples from testing set as "$x_t = 1$" indicates that the user is currently connected to VDI system making the logon prediction unnecessary. This conforms to the practical scenario where VDI system makes logon prediction and launches desktops only for offline users. Finally, all the four comparing methods use the same group-based model with SOUP where a single model is trained from the logon data of all the users in one persistent pool.

4.2.3 Evaluation Metrics. Two metrics are used for performance evaluation, including *Area Under the ROC Curve* (AUC) and *Cost Saving Absolute Gain* (CSAG). AUC is a single-value metric which attempts to summarize an ROC curve to evaluate the quality of a classifier. After excluding "$x_t = 1$" samples in testing set, we observe an imbalanced distribution where number of negative cases ($x_{t+1} = 0$) is 100 times of positive cases ($x_{t+1} = 1$). When dealing with imbalanced data, AUC is more feasible to represent model performance than precision and recall.

CSAG serves as a domain specific metric measuring the cost saving gain by user logon prediction. In current solution, the VDI system admin usually keeps all VM powered on no matter the end user is connected or not. Let TN be the number of true negatives where user is correctly predicted as not connected and a be the cost of VM charges of time range Δt, the overall cost saving can be represented as $a *$ TN. Similarly, let FN be the number of false negatives where user actually logon but predicted not, and b be the cost of employee's time wasted in waiting for VM to be powered on and desktop to be ready, the penalty cost of the incorrect prediction is $b *$ FN. For simplicity, the definition of CSAG is as follows:

$$\text{CSAG} = \text{TN} - \frac{b}{a}\text{FN} \tag{22}$$

Here, we use the pricing of the on-demand AWS EC2 t3.large VM (Windows) [1] as a and 10 minutes' employee salary of four different customers' industries as b. The configuration of $\frac{b}{a}$ for the four customers is detailed in Table 14. Comparing with common metrics like accuracy and recall, CSAG is more intuitive from VDI resource management perspective.

Table 15. Performance on the real-world VDI data sets in term of AUC Score and Cost Saving Absolute Gain.

Comparing	**AUC Score**							
Methods	Cocoa-D1	Cocoa-D2	Avocado-D1	Avocado-D2	Orange-D1	Orange-D2	Lemon-D1	Lemon-D2
SOUP	**0.9588**	**0.9084**	**0.9681**	**0.9690**	**0.9203**	**0.8650**	**0.9124**	**0.9054**
Prophet	0.7169	0.6528	0.8553	0.8224	0.5923	0.5658	0.6565	0.5625
Holt-Winters	0.6948	0.6649	0.7963	0.8538	0.7203	0.6643	0.7634	0.7732
RNN-IDS	0.7846	0.6883	0.7780	0.8010	0.6976	0.6859	0.8640	0.8302
LSTM	0.6607	0.6737	0.7389	0.7688	0.7024	0.7134	0.8180	0.8226
	Cocoa-D3	Cocoa-D4	Avocado-D3	Avocado-D4	Orange-D3	Orange-D4	Lemon-D3	Lemon-D4
SOUP	**0.9494**	**0.9477**	**0.9646**	**0.9503**	**0.9259**	**0.879**	**0.9533**	**0.9285**
Prophet	0.6431	0.7747	0.7541	0.788	0.591	0.6909	0.5395	0.681
Holt-Winters	0.8223	0.7731	0.8769	0.8015	0.876	0.6665	0.8792	0.6846
RNN-IDS	0.7451	0.7625	0.7453	0.759	0.8077	0.8013	0.813	0.8574
LSTM	0.6478	0.6626	0.673	0.7312	0.8189	0.8244	0.8253	0.847
Comparing	**Cost Saving Absolute Gain**							
Methods	Cocoa-D1	Cocoa-D2	Avocado-D1	Avocado-D2	Orange-D1	Orange-D2	Lemon-D1	Lemon-D2
SOUP	**20940**	**20371**	**13282**	**15731**	**25565**	**24025**	**10452**	**17862**
Prophet	3490	2476	9122	12709	19590	22138	2361	4452
Holt-Winters	12099	11420	11569	14832	23265	22282	6770	12788
RNN-IDS	4819	2109	4345	8139	17562	16667	8132	10928
LSTM	2370	1952	5121	10239	21541	22622	8270	14775
	Cocoa-D3	Cocoa-D4	Avocado-D3	Avocado-D4	Orange-D3	Orange-D4	Lemon-D3	Lemon-D4
SOUP	**16448**	**16567**	**13140**	**12936**	**20532**	**20699**	**13789**	**11496**
Prophet	-206	6186	7212	6921	15943	16614	-441	-119
Holt-Winters	5582	3388	11175	7541	19078	18055	9927	202
RNN-IDS	2692	2280	6467	5795	17614	18514	4955	7580
LSTM	1545	183	5124	5528	17620	18274	8248	7859

Table 16. Friedman statistics F_F for SOUP in terms of each evaluation metric and the critical value at 0.05 significance level (# comparing algorithms $k = 5$, # data sets $N = 16$).

Evaluation metric	F_F	critical value
AUC	19.41	2.5252
CSAG	25.25	

4.2.4 Experimental Results. Table 15 summarizes the detailed experimental results in which the best performance on each data set is shown in boldface. For CSAG, the threshold yielding best CSAG score is utilized for each comparing approach. To analyze the relative performance among the comparing approaches in a systematic manner, *Friedman test* [10] is employed as the statistical test for performance comparison. The Friedman statistics F_F and the critical values of each evaluation metric are reported in Table 16. Accordingly, we perform post-hoc *Bonferroni-Dunn test* [11] to compare the relative performance among the comparing approaches and control algorithm. The CD diagrams are presented in Figure 8, where the average rank of each approach is marked along the axis (the smaller the better).

Table 17. Characteristics of the OPSD data sets where `min-max` and `mean±std` stand for the minimum/maximum session count, mean value and standard deviation of the session counts respectively.

Data sets	Period	min-max	mean±std
Dataset-1	29/01-22/07, 2018	4274-10836	7229±1442
Dataset-2	08/07-29/12, 2019	4167-10817	7075±1383
Dataset-3	01/02-26/07, 2015	3382-10013	6810±1344
Dataset-4	01/02-26/07, 2019	4312-10328	7107±1310

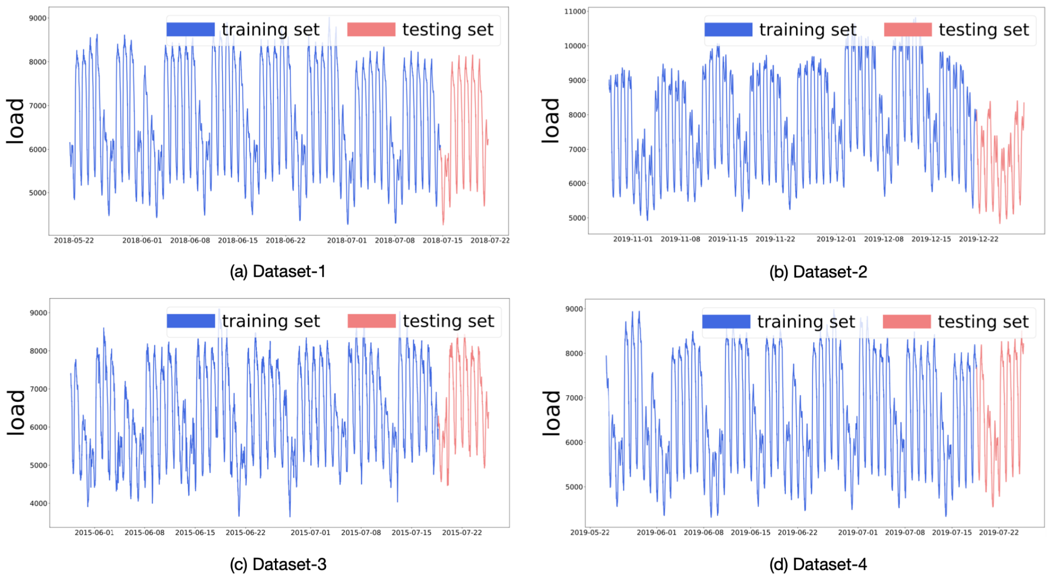


Fig. 9. Ground-truth electricity workload of the OPSD data sets. Only part of the training data is shown in the figure.

Based on the experiment results, the following observations can be made:

- Among the 32 configurations (16 data sets × 2 evaluation metrics), SOUP ranks 1st in 100% cases. As per the diverse properties of experimental data sets, SOUP demonstrates desirable generalization ability in user logon prediction. Furthermore, for `Division D2` in which testing set includes Christmas holiday, SOUP keeps stable superiority against comparing approaches.
- Apparently, the null hypothesis that SOUP and comparing approaches are of equal performance is rejected at 0.05 significance level.

4.3 Experiments on Electricity Load Prediction

To evaluate the generalization ability of coarse-to-fine multi-grained feature extraction method, we conduct the CAFE experiments of electricity load prediction tasks on "Open Power System Data" (OPSD) [9]. We use the same comparing approaches (Prophet, Holt-Winters, NF-GBDT and LSTM) and evaluation metrics (NMAE, NRMSE, OPU and UPR) with the CAFE experiments on VDI workload prediction.

Table 18. Performance on the OPSD data sets in term of NMAE, NRMSE, OPR and UPR (Δt = 120*minutes*).

Data Sets	NMAE					NRMSE				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Dataset-1	**0.0251**	0.0634	0.0750	0.0292	0.0451	**0.0346**	0.0760	0.098	0.0398	0.0555
Dataset-2	**0.0122**	0.0448	0.0722	0.0146	0.0351	**0.0159**	0.0547	0.0954	0.0206	0.0402
Dataset-3	**0.0197**	0.0463	0.0699	0.0239	0.0385	**0.0236**	0.0544	0.0928	0.0288	0.0470
Dataset-4	**0.1021**	0.265	0.2013	0.1253	0.1813	**0.1250**	0.2939	0.2362	0.1594	0.2229
Data Sets	**OPR**					**UPR**				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Dataset-1	**0.0123**	0.0530	0.0418	0.0140	0.0180	0.0128	**0.0103**	0.0332	0.0152	0.0271
Dataset-2	0.0041	0.0317	0.0368	0.0052	**0.0035**	**0.0082**	0.0131	0.0355	0.0094	0.0316
Dataset-3	0.0030	0.0283	0.0274	0.0057	**0.0013**	**0.0167**	0.0180	0.0424	0.0182	0.0372
Dataset-4	**0.0955**	0.2628	0.1922	0.1191	0.1735	0.0065	**0.0022**	0.0090	0.0062	0.0078

4.3.1 Data Sets. The electricity load data are collected from "Open Power System Data" (OPSD) from 2015 to 2020. We select 4 data sets and name them as `Dataset-1`, `Dataset-2`, `Dataset-3` and `Dataset-4`. Each data set contains 25 weeks of data and the sample interval is 15 minutes. We use the first 24 weeks for training and the last 1 week for testing. Table 17 shows the detailed statistics of the 4 data sets, where `min-max` and `mean±std` refer to the minimum/maximum, mean and standard deviation of the electricity load. Figure 9 shows the ground-truth value of the 4 data sets where we can find some patterns. First, all 4 data sets show the working-day pattern where the working days (from Monday to Friday) have much higher electricity load than the weekend (from Saturday to Sunday). Additionally, Sunday almost has the least load but has some days of anomaly in every data set. Second, all the data sets display a slowly and regularly change pattern. It is interesting that `Dataset-2` shows a two-peek on every day. Moreover, `Dataset-3` exhibits more short term fluctuations than other three data sets. Last but not least, notable variances can be observed in the testing sets. For instance, `Dataset-2` has much lower load in the whole week as it is Christmas. `Dataset-1` shows a drift pattern where the average electricity load declines slowly every week.

4.3.2 Coarse-to-fine Parameters. As the sample interval of electricity load raw data is 15 minutes, we perform two group of experiments with $\Delta t = 120$ *minutes* and $\Delta t = 240$ *minutes*. As discussed in the Section 2.3, the coarse-to-fine feature extraction parameters are as in Table 6 and Table 7. We use the same maximum action scope n set as 1440 minutes (24 hours), the granularity vector $\boldsymbol{k}$ is then set as $(1440, 720, 240, 180, 30, 15, 15)$ for $\Delta t = 120$ minutes and $(1440, 720, 240, 180, 60, 30, 15)$ for $\Delta t =$ 240 minutes. Meanwhile, the action scope vector $\boldsymbol{\gamma}$ is configured as $(1440, 720, 240, 180, 240, 120, 15)$ and $(1440, 720, 240, 180, 480, 240, 15)$ for $\Delta t = 120$ minutes and $\Delta t = 240$ minutes respectively.

4.3.3 Experimental Results. The experimental results are summarized in Table 18 and Table 19 respectively where the best performance on each data set is shown in boldface. Based on the reported experiment results, the following observations can be made:

- Out of a total of 32 configurations (8 data sets × 4 evaluation metrics), CAFE achieves 1st ranks in 22 (61.1%) cases and 2nd ranks in 6 (16.7%) cases. In the prediction of the shorter time span ($\Delta t = 120$), CAFE ranks 1st in 12 (66.7%) and 2nd in 3 out of 18 (16.7%) cases. In the

Table 19. Performance on the OPSD data sets in term of NMAE, NRMSE, OPR and UPR ($\Delta t = 240 minutes$).

Data Sets	NMAE					NRMSE				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Dataset-1	**0.0284**	0.1015	0.1278	0.0300	0.0407	**0.0367**	0.1170	0.1614	0.0396	0.0527
Dataset-2	0.0170	0.0832	0.1312	**0.0131**	0.0229	0.0222	0.1012	0.1653	**0.0167**	0.0288
Dataset-3	**0.024**	0.0771	0.1212	0.0250	0.0362	**0.0286**	0.0944	0.1584	0.0301	0.0443
Dataset-4	**0.1237**	0.3048	0.2261	0.1559	0.1757	**0.1502**	0.3343	0.2622	0.1989	0.2174
Data Sets	OPR					UPR				
	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM	CAFE	Prophet	Holt-Winters	NF-GBDT	LSTM
Dataset-1	**0.0134**	0.0895	0.0691	0.0165	0.0184	0.0150	**0.0119**	0.0586	0.0136	0.0224
Dataset-2	**0.0036**	0.0681	0.0663	0.0069	0.0213	0.0134	0.0151	0.0649	0.0063	**0.0017**
Dataset-3	**0.0024**	0.0558	0.0529	0.0035	0.0035	0.0215	**0.0213**	0.0683	0.0215	0.0327
Dataset-4	**0.1129**	0.3046	0.2041	0.1486	0.1647	0.0108	**0.0001**	0.0220	0.0072	0.0110

prediction of the longer time span prediction ($\Delta t = 240$), CAFE ranks 1st in 10 (55.5%) and 2nd in 3 out of 18 (16.7%) cases. It is worth noting that CAFE achieves 1st rank in all cases in terms of NMAE and NRMSE when $\Delta t = 120$.

- CAFE performs not well on UPR when $\Delta t = 240$. By analysis, we can see that the main reason is CAFE has pretty good balance between OPR (all rank 1st) and UPR while Prophet has un-acceptable high OPR which makes its UPR relatively low. In the VDI domain, the balance between OPR and UPR is very important for the VDI desktop pool administrator to measure the cost saving against user experience satisfaction.

5 GENERALIZATION ABILITY OF COARSE-TO-FINE FEATURE EXTRACTION

As one of the most important contributions of this paper, coarse-to-fine feature extraction was proposed specifically in VDI domain and its effectiveness has been proven by the experiment results of non-persistent and persistent desktop pool workload prediction. Additionally, the parameters tuning practice has been discussed in Section 2.3. Furthermore, in the extended experiments in Section 4.3.3, CAFE achieves convincing generalization performance on the electricity load prediction which clearly proves the effectiveness of the coarse-to-fine feature extraction approach. Generally, it is highly potential that this method can be tuned to adapt workload prediction or time-series prediction tasks in an explainable way under controllable effort. It is worth mentioning that we do not have any domain knowledge of electricity load data and just tune the coarse-fine parameters with some common sense. Accordingly, we have reason to believe that with more guidance of related domain knowledge, CAFE and SOUP can be optimized to perform even better in various workload prediction and time-series prediction tasks.

6 CONCLUSION

In this paper, an extension to our earlier research [50] is presented where two adaptive learning approaches named CAFE and SOUP are proposed for VDI workload prediction on non-persistent and persistent pools. With the multi-grained features, selected seasonal and contextual features generated from the historical workload time-series data, CAFE and SOUP train aggregated model by leveraging the GBDT regressor. Comprehensive experiments on real VDI customer data demonstrate that CAFE and SOUP achieve preferable performance than the comparing models and show superior generalization capability on dealing with data sets with varying characteristics. Furthermore, extended experiments of electricity load prediction on OPSD data are conducted where CAFE's superior performance validates the potential applicability of using coarse-to-fine feature extraction method in other workload prediction and time series data prediction tasks.

During the procedure of designing and evaluating CAFE and SOUP, we reveal several practicable insights: 1) Aggregated model shows superiority in VDI workload prediction for its capability of adopting the global behaviors of all pool-sharing users as well as its cost-efficient training. 2) For different types of VDI pools, different multi-grained features are required. For non-persistent pool, multi-grained features are generated from pool-level aggregated workload historical data. For persistent pool, multi-grained features generated from individual users are incorporated into one pool-level aggregated model. 3) The design of multi-grained features must be well-balanced between long-term and short-term characteristics to improve the model's generalization ability of handling macro and micro patterns in data sets with diverse properties.

Future works may include exploring advanced machine learning techniques on VDI workload data sets, such as deep learning for large-scale data sets [30, 52], feature augmentation for contextual features exploitation [22, 45], group supervision modeling for pool-sharing users [47], etc., and experimenting further on longer prediction interval. We also plan to further evaluate coarse-to-fine feature extraction method on more workload prediction tasks in other domains. From engineering perspective, it is beneficial to study the possibility of enable online training and testing in VDI workload prediction models.

REFERENCES

[1] Amazon Web Services. 2020. Amazon EC2 on-demand pricing. https://aws.amazon.com/ec2/pricing/on-demand/.
[2] Masoud Barati and Saeed Sharifian. 2015. A hybrid heuristic-based tuned support vector regression model for cloud load prediction. *The Journal of Supercomputing* 71, 11 (2015), 4235–4259.
[3] Rodrigo N Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya. 2014. Workload prediction using ARIMA model and its impact on cloud applications' QoS. *IEEE transactions on cloud computing* 3, 4 (2014), 449–458.
[4] Emiliano Casalicchio, Stefano Iannucci, and Luca Silvestri. 2015. Cloud desktop workload: A characterization study. In *2015 IEEE International Conference on Cloud Engineering*. IEEE, 66–75.
[5] Katja Cetinski and Matjaz B Juric. 2015. AME-WPC: Advanced model for efficient workload prediction in the cloud. *Journal of Network and Computer Applications* 55 (2015), 191–201.
[6] Chris Chatfield. 1978. The Holt-winters forecasting procedure. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 27, 3 (1978), 264–279.
[7] Chris Chatfield and Mohammad Yar. 1988. Holt-Winters forecasting: some practical issues. *Journal of the Royal Statistical Society: Series D (The Statistician)* 37, 2 (1988), 129–140.
[8] Mustafa Daraghmeh, Anjali Agarwal, Ricardo Manzano, and Marzia Zaman. 2021. Time series forecasting using Facebook Prophet for cloud resource management. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 1–6.
[9] Open Power System Data. 2020. Open Power System Data time series. https://data.open-power-system-data.org/time_series/2020-10-06/.
[10] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research* 7 (2006), 1–30.
[11] Olive Jean Dunn. 1961. Multiple comparisons among means. *Journal of the American statistical association* 56, 293 (1961), 52–64.
[12] Wenping Fan, Yao Zhang, Qichen Hao, Xinya Wu, and Min-Ling Zhang. 2021. BAMBOO: A multi-instance multi-label approach towards VDI user logon behavior modeling. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 2367–2373. https://doi.org/10.24963/ijcai.2021/326 Main Track.
[13] Wei Fang, ZhiHui Lu, Jie Wu, and ZhenYin Cao. 2012. Rpps: A novel resource prediction and provisioning scheme in cloud data center. In *2012 IEEE Ninth International Conference on Services Computing*. IEEE, 609–616.
[14] Dror G Feitelson. 2015. *Workload modeling for computer systems performance evaluation.* Cambridge University Press.

[15] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[16] Jerome H Friedman. 2002. Stochastic gradient boosting. *Computational statistics & data analysis* 38, 4 (2002), 367–378.

[17] Jiechao Gao, Haoyu Wang, and Haiying Shen. 2020. Machine learning based workload prediction in cloud computing. In *2020 29th international conference on computer communications and networks (ICCCN)*. IEEE, 1–9.

[18] Felix A Gers, Douglas Eck, and Jürgen Schmidhuber. 2002. Applying LSTM to time series predictable through time-window approaches. In *Neural Nets WIRN Vietri-01*. Springer, 193–200.

[19] Steve R Gunn et al. 1998. Support vector machines for classification and regression. *ISIS technical report* 14, 1 (1998), 5–16.

[20] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at Facebook. In *Proceedings of the eighth international workshop on data mining for online advertising*. 1–9.

[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[22] Bin-Bin Jia and Min-Ling Zhang. 2022. Multi-dimensional classification via selective feature augmentation. *Machine Intelligence Research* 19, 1 (2022), 38–51.

[23] Dionatra F Kirchoff, Miguel Xavier, Juliana Mastella, and César AF De Rose. 2019. A preliminary study of machine learning workload prediction techniques for cloud applications. In *2019 27th Euromicro international conference on parallel, Distributed and Network-Based Processing (PDP)*. IEEE, 222–227.

[24] Anoop S Kumar and Somnath Mazumdar. 2016. Forecasting HPC workload using ARMA models and SSA. In *2016 International Conference on Information Technology (ICIT)*. IEEE, 294–297.

[25] Jitendra Kumar, Rimsha Goomer, and Ashutosh Kumar Singh. 2018. Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for cloud datacenters. *Procedia Computer Science* 125 (2018), 676–682.

[26] Jitendra Kumar and Ashutosh Kumar Singh. 2018. Workload prediction in cloud using artificial neural network and adaptive differential evolution. *Future Generation Computer Systems* 81 (2018), 41–52.

[27] Peter Laurinec and Mária Lucká. 2016. Comparison of representations of time series for clustering smart meter data. In *Proceedings of the world congress on engineering and computer science*, Vol. 1.

[28] Peter Laurinec and Mária Lucká. 2017. New clustering-based forecasting method for disaggregated end-consumer electricity load using smart grid data. In *2017 IEEE 14th international scientific conference on informatics*. IEEE, 210–215.

[29] Shengming Li, Ying Wang, Xuesong Qiu, Deyuan Wang, and Lijun Wang. 2013. A workload prediction-based multi-vm provisioning mechanism in cloud computing. In *2013 15th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 1–6.

[30] Xin-Chun Li, De-Chuan Zhan, Jia-Qi Yang, and Yi Shi. 2021. Deep multiple instance selection. *Science China Information Sciences* 64, 3 (2021), 1–15.

[31] Chunhong Liu, Chuanchang Liu, Yanlei Shang, Shiping Chen, Bo Cheng, and Junliang Chen. 2017. An adaptive prediction approach based on workload pattern discrimination in the cloud. *Journal of Network and Computer Applications* 80 (2017), 35–44.

[32] Mohammad Masdari and Afsane Khoshnevis. 2020. A survey and classification of the workload forecasting methods in cloud computing. *Cluster Computing* 23, 4 (2020), 2399–2424.

[33] Ali Yadavar Nikravesh, Samuel A Ajila, and Chung-Horng Lung. 2015. Towards an autonomic auto-scaling prediction system for cloud resource provisioning. In *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, 35–45.

[34] J Prassanna and Neelanarayanan Venkataraman. 2021. Adaptive regressive holt–winters workload prediction and firefly optimized lottery scheduling for load balancing in cloud. *Wireless Networks* 27, 8 (2021), 5597–5615.

[35] Kashifuddin Qazi and Igor Aizenberg. 2018. Cloud datacenter workload prediction using complex-valued neural networks. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*. IEEE, 315–321.

[36] ReportLinker. 2017. Global desktop virtualization market analysis (2017-2023). https://www.reportlinker.com/p05207394.

[37] Maurizio Rossi and Davide Brunelli. 2015. Forecasting data centers power consumption with the Holt-Winters method. In *2015 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS) Proceedings*. IEEE, 210–214.

[38] Li Ruan, Yu Bai, Shaoning Li, Shuibing He, and Limin Xiao. 2021. Workload time series prediction in storage systems: a deep learning based approach. *Cluster Computing* (2021), 1–11.

[39] Ashraf A Shahin. 2017. Using multiple seasonal holt-winters exponential smoothing to predict cloud resource provisioning. *arXiv preprint arXiv:1701.03296* (2017).

[40] SR Shishira and A Kandasamy. 2021. A novel feature extraction model for large-scale workload prediction in cloud environment. *SN Computer Science* 2, 5 (2021), 1–7.

[41] Parminder Singh, Pooja Gupta, and Kiran Jyoti. 2019. Tasm: technocrat arima and svr model for workload prediction of web applications in cloud. *Cluster Computing* 22, 2 (2019), 619–633.

[42] Haiyan Song and Gang Li. 2008. Tourism demand modelling and forecasting—A review of recent research. *Tourism management* 29, 2 (2008), 203–220.

[43] Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. 2003. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences* 43, 6 (2003), 1947–1958.

[44] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72, 1 (2018), 37–45.

[45] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2020. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478* (2020).

[46] Tri Kurniawan Wijaya, Samuel François Roger Joseph Humeau, Matteo Vasirani, and Karl Aberer. 2014. *Individual, aggregate, and cluster-based aggregate forecasting of residential demand.* Technical Report. School of Computer and Communication Sciences, EPFL.

[47] Miao Xu and Lan-Zhe Guo. 2021. Learning from group supervision: the impact of supervision deficiency on multi-label learning. *Science China Information Sciences* 64, 3 (2021), 1–13.

[48] Yingxiao Xu, Prasad Calyam, David Welling, Saravanan Mohan, Alex Berryman, and Rajiv Ramnath. 2013. Human-centric composite-quality modeling and assessment for virtual desktop clouds. *ZTE Communications* 11, 1 (2013), 27–36.

[49] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, and Xinzheng He. 2017. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access* 5 (2017), 21954–21961.

[50] Yao Zhang, Wen-Ping Fan, Xuan Wu, Hua Chen, Bin-Yang Li, and Min-Ling Zhang. 2019. CAFE: adaptive VDI workload prediction with multi-grained features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5821–5828.

[51] Wei Zhong, Yi Zhuang, Jian Sun, and Jingjing Gu. 2018. A load prediction model for cloud computing using PSO-based weighted wavelet support vector machine. *Applied Intelligence* 48, 11 (2018), 4072–4083.

[52] Lingxue Zhu and Nikolay Laptev. 2017. Deep and confident prediction for time series at uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 103–110.