# Mutually Adaptable Learning

Qi Tan, Yang Liu, *Senior Member, IEEE*, and Jiming Liu, *Fellow, IEEE*

***Abstract*—In real-world applications that involve complex data dependencies, it would be essential to proceed with machine learning tasks in an adaptable manner. This article presents a novel Mutually Adaptable Learning (MAL) approach that allows for, on the one hand, extracting the most crucial information from data, and on the other hand, maximally utilizing it through model learning, in a mutually adaptable manner. We elaborate our MAL approach by explaining how it determines the necessity for the adaptation of both features and the learning model, integratively adapts between feature selection and model learning, and optimally achieves the learning objective. To systematically validate the effectiveness of MAL, we conduct comprehensive experiments on challenging learning tasks from two representative domains: spatiotemporal prediction and chaotic behavioral prediction, where the complex data dependencies are general encountered. Results demonstrate that MAL outperforms existing learning methods. Moreover, we show that the formulated objective can be attained under an information-theoretic guarantee. With both empirical and theoretical supports, MAL offers an effective solution to the problem of feature and model adaptation to achieve desired learning objective for given complex tasks.**

***Index Terms*—Complex data dependency, Mutually Adaptable Learning (MAL), feature and model adaptation, information-theoretic analysis.**

## I. Introduction

With the availability of a vast amount of data and the rapid development of computing resources and storage devices, how to extract useful information from the available data and then make the best use of it is becoming increasingly important to various learning tasks. However, learning from data is never an easy mission due to complex dependencies and relationships among multiple variables caused by their nonlinear interactions (referred to as synergistic effects). Two typical application domains that show such complex dependencies and synergistic effects are spatiotemporal prediction [12], [13], [43], [48] and chaotic behavioral prediction [2], [6], [53], involving analyzing the underlying relationships among multiple variables over time and making the subsequent inference or prediction on the target variable (Fig. 1(a)).

Manuscript received 13 February 2023; revised 12 July 2023; accepted 22 July 2023. Date of publication 9 August 2023; date of current version 23 January 2024. This work was supported in part by the Ministry of Science and Technology of China under Grants 2021ZD0112501 and 2021ZD0112502, and in part by the General Research Fund from the Research Grant Council of Hong Kong SAR under Grants RGC/HKBU12201619, RGC/HKBU12202220, and RGC/HKBU12203122. *(Corresponding author: Jiming Liu.)*

The authors are with the Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR 999077, P. R. China (e-mail: csqtan@comp.hkbu.edu.hk; csygliu@comp.hkbu.edu.hk; jiming@comp.hkbu.edu.hk).

Digital Object Identifier 10.1109/TETCI.2023.3300183

In spatiotemporal prediction tasks such as traffic prediction [13], disease prediction [21], and climate forecasting [48] (Figs. 1(b) and 4 show an example of traffic prediction), there exist complex dependencies among data within and between temporal and spatial dimensions at multiple scales (e.g., temporally at yearly, monthly, weekly, daily, and hourly resolutions, as well as spatially at country, province, city, district, and street levels). As such spatiotemporal dependencies manifest at varying scales and are not directly observable, their impacts on the target variable are difficult to be quantitatively characterized by existing learning models with one-off, fixed structures.

Another typical and interesting application that well demonstrates the complexity of data relationships is predicting the dynamical behaviors of chaotic systems. In many representative chaotic systems such as the Lorenz system [6] and the double or triple pendulum system [2], we can observe highly nonlinear dynamics (as shown in Figs. 1(c) and 5), which are hard to predict, due to their extreme sensitivities to small perturbations in conditions. Existing time series models and deep recurrent neural network (DRNN) models have been designed to capture short-term dependencies or periodic long-term dependencies with predefined model structures, and thus have shown certain limitations in predicting such chaotic dynamics with irregular long-term dependencies and high-order correlations [53].

The above learning tasks have several critical computational challenges in common. Firstly, the observations in these tasks are complex — highly nonlinear, heterogeneous, and/or high-dimensional. As a result, it is difficult to determine which part of the observations is the most informative one with respect to the learning task. Further, the input variables and the target variable are nonlinearly coupled and interacting, resulting in complex synergistic effects and substantial irregularities of the relationships between observations and the target variable. This brings significant challenges to modeling intrinsic data dependencies. More importantly, the complexity of observations, the underlying data dependencies at multiple scales, and the sensitivities of target variable to small perturbations on observations make existing predefined one-off models too rigid to capture the subtleties of the data and the task.

To address the aforementioned challenges in an effective and efficient manner, we need to answer the following critical question: **Given a learning task, how can the learning procedure carry out the task with the right information, through the right model, at the right time?** In other words, how can the learning procedure, on the one hand, extract the most crucial information from the complex data, and on the other hand, utilize the extracted information to the maximum extent by constructing appropriate learning models, in a flexible and adaptable manner?

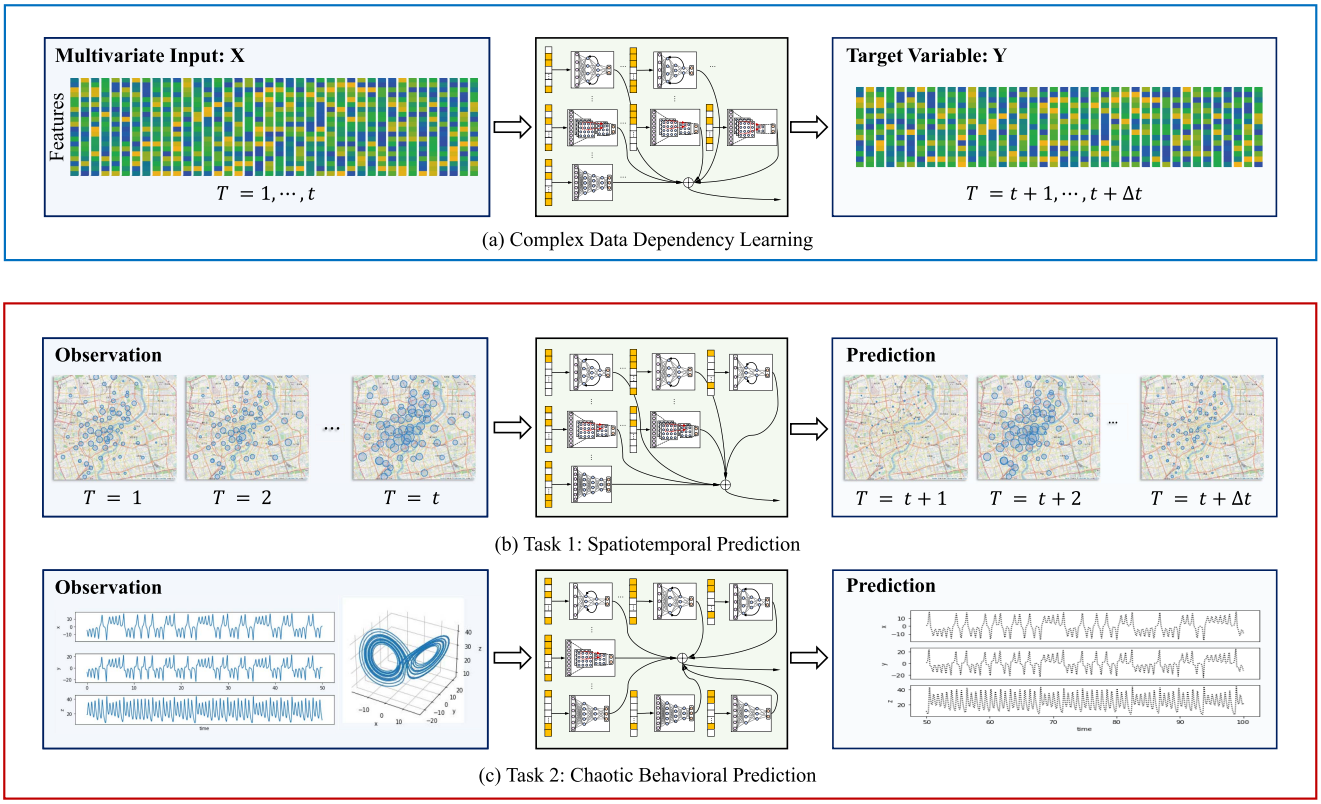



Fig. 1. Schematic illustration of the research problem studied in this paper. (a) An overview of the research problem: learning the complex dependencies and relationships among multiple variables over time (i.e., $\mathbf{X}$) and making the subsequent inference or prediction on the target variable $\mathbf{Y}$. (b) Spatiotemporal prediction, where the multi-scale dependencies of data along both spatial and temporal dimensions are hard to capture and thus bring challenges to the task. (c) Chaotic behavioral prediction, where the sensitivities to small perturbations caused by the system nonlinearity make the accurate prediction extremely difficult. Note that although the same MAL approach is used to learn the architectures for the tasks of spatiotemporal prediction and chaotic behavioral prediction, the constructed architectures on these two tasks could be different, as illustrated in the center of (b) and (c). The reason is that the intrinsic challenges of data dependency learning in these two tasks differ. The proposed MAL approach, therefore, needs to extract different kinds of information from data to make accurate predictions by selecting the most appropriate basic units and constructing the corresponding learning architectures for given tasks.

To answer this question in a systematic way, in this article, we design, demonstrate, and analyze a novel approach called Mutually Adaptable Learning (MAL), aiming at addressing the following three problems: **(P1)** How to design the learning method? **(P2)** How to validate the learning effectiveness? and **(P3)** How to guarantee the learning capacity?

### A. Related Work

This article aims to propose a novel MAL approach for complex data dependency learning via mutually adaptable feature selection and model update. In what follows, we review some related work in feature selection, automated model generation, and their applications in data dependency learning.

*1) Feature Selection:* Feature selection aims to select the most informative subset of features for the subsequent learning tasks. Typical feature selection methods can be categorized into three main categories: (i) filter methods, which rank the usefulness of features based on their scores in predefined statistical tests and select the features with the highest scores [37], [50]; (ii) wrapper methods, which usually use a search strategy, e.g., the genetic algorithm [25] or reinforcement algorithm [33], to identify the optimal feature set by evaluating the performance of prediction models on the selected feature set; and (iii) embedded methods, which integrate the processes of feature selection into the training process of the predictive models, and thus are generally more effective than filter methods and embedded methods, making embedded methods widely used [41], [44]. Most of the aforementioned feature selection methods are heuristically designed to search for the optimal feature subset in a one-way manner. However, the valuable features in terms of prediction vary from situation to situation. An informative feature may become redundant after incorporating another feature, or a less informative feature may become helpful when combined with another feature. Therefore, an adaptable way to select the optimal set of features, together with the procedure of model adaptation, is highly desired, especially when encountering learning tasks with complex observations.

Some recent studies focused on streaming feature selection problems for Big Data [1], [54], which could partially address the aforementioned challenge by dynamically deleting old features or including new features. One representative work along this direction was conducted by Zhou et al. [59]. They proposed an online group streaming feature selection method, which first selects features interacting with each other and then uses the regularization and elastic net for feature

grouping. However, these streaming feature selection methods do not explicitly consider the varying importance of features, which could be well handled by our approach via mutual adaptation.

*2) Automated Model Generation:* Automated model generation aims to automatically create a learning model with minimum manual efforts. The pipeline of automated model generation consists of two components: the search space and the optimization methods [15], [51]. The search space defines the scope of structures of models (traditional models or deep neural network (DNN) models) that can be explored while the optimization methods determine two types of hyper-parameters in models: hyper-parameters for training and those for model design. To generate complex DNN models, optimization methods further involve architecture optimization, which searches for the architecture with the best performance.

For DNN models, the technique used for automated model generation is called neural architecture search (NAS). The search space of NAS can be divided into four categories: entire-structured [60], cell-based [58], hierarchical [31], and morphism-based [8]. Regarding the hyper-parameter optimization in NAS, representative methods include grid and random search [3], [17], Bayesian optimization [42], and gradient-based optimization [35]. For architecture optimization, typical methods include random search [26], evolutionary algorithms [47], Bayesian optimization [22], reinforcement learning [60], and gradient descent for differential structures [32].

The high computational cost is one of the main challenges of NAS. Progressive automated learning has shown promising performance in reducing the computational cost, resulting in efficient NAS [9], [30]. Kiranyaz et al. proposed a progressive method for searching connections with fixed type of perceptrons [23], which, however, is still computationally expensive as it searches at the unit of perceptron and thus cannot scale well to large datasets with numerous features. Kiranyaz et al. further developed a self-organized variant with generative neurons that can adapt the nodal operator of each connection [24]. Xu et al. proposed to use random sampling from the super network to reduce the redundancy in exploration, thus improving the search efficiency [49]. However, these methods generally used the fixed structure between the cells and the homogeneous cell type for the upper and lower layers, and thus were less applicable when various kinds of dependencies at multiple scales exist and need to be captured via different types of cells or blocks. Our mutually adaptable learning approach can address this challenging issue by using a gradual adaption strategy, in which the most useful information, in each step, can be extracted and represented with properly selected units.

Furthermore, the structure searching process in the existing automated model generation methods is performed in a black-box manner. It is, therefore, difficult to understand the properties of each learning step and, thus, to theoretically guarantee the final outcome. Recent studies revealed that the evaluation of candidate architecture's learning performance in each step is essential in model searching [10], [56]. To address this problem, in the proposed approach, we develop an information-theoretic-based learning objective for model adaptation, so that the performance in each step can be analytically examined, and the achievability of the objective can be guaranteed.

*3) Feature Selection and Model Generation for Data Dependency Learning:* Learning complex data dependency has attracted much research attention due to its ubiquity in various domains such as spatiotemporal prediction [13], [16] and chaotic behavioral prediction [6], [7]. For an effective learning process, the selection of the feature space and the construction of the learning model are of critical importance [39], [57].

In addition to employing existing feature selection methods, many learning models utilize manually designed features. For example, in spatiotemporal prediction, regarding the spatial features, Li et al. incorporated the road network structure into the recurrent neural network (RNN) model, utilizing the states of the nearby locations for traffic forecasting [28]; regarding the temporal features, Zhang et al. constructed the recent, periodic, and long-term features for crowd flow prediction [55]. However, manual feature design requires domain-specific knowledge, which is generally difficult or even impossible to obtain in many scenarios [37], [39].

In terms of the learning model, various DNN models have demonstrated impressive performance in different applications of spatiotemporal prediction [28], [29], [40] and chaotic behavioral prediction [6], [7]. To enhance the applicability of learning models and to reduce the demand for experienced human experts, some recent studies have aimed to automate the process of learning model generation in complex learning problems. Li et al. applied NAS to spatiotemporal datasets and showed that the learned architecture outperforms the existing models [27]. However, the proposed model still required an initial spatiotemporal feature extractor, which needs extra manual effort and may exclude the feature subspace with potentially useful information.

Moreover, current studies only focus on the adaptation of model structure but ignore the progressive exploration of the intrinsic dependency among complex data, which is of great importance in various learning tasks. Without a clear understanding of the underlying relationships between the complex data and the learning task, it still remains a mystery how to dynamically extract the most crucial information from data and utilize it to the maximum extent by adaptively constructing appropriate learning models.

### B. Our Contributions

This article tackles the aforementioned challenges in complex data dependency learning by designing, demonstrating, and analyzing a novel MAL approach to address the three fundamental problems raised before Section I-A. Specifically, we summarize the contributions of this work as follows:

1) *Architectural Design of MAL:* To address **(P1)**, we elaborate our MAL approach, which is composed of two components: feature selection and model learning. Guided by the formulated learning objective and a residual learning strategy, the proposed MAL adapts between feature selection and model learning in an integrative way, so as to optimally achieve the target. Finally, we quantitatively

characterize the learning ability of the proposed design of MAL via information capacity analysis.

2) *Systematic Validation of MAL:* To address **(P2)**, we compare the performance of MAL with state-of-the-art learning methods through a set of experiments in two typical domains with complex data dependency: spatiotemporal prediction and chaotic behavioral prediction. Results demonstrate that MAL not only outperforms existing methods, but more importantly, offers an explicit guidance to construct desired architectures according to the given tasks and datasets.
3) *Information-Theoretic Analysis of MAL:* To address **(P3)**, we first rewrite the learning objective in an equivalent form from the information-theoretic perspective to measure the amount of information to be obtained from the data with respect to a given learning task. Armed with the information-theoretic measure, we show that the formulated objective is achievable and the learning procedure is able to converge.

### C. Organization of the Paper

The rest of the article is organized as follows. Section II elaborates the details of MAL, including the approach overview, the detailed procedure, and the learning behavior analysis. Section III demonstrates extensive experimentations on real-world learning tasks to validate the effectiveness of MAL. Section IV introduces the information-theoretic framework to re-formulate the objective of MAL, enabling the quantitative measurement of the amount of information to be obtained from the data, and proves the convergence of MAL. Section V concludes the article.

## II. Architectural Design of Mutually Adaptable Learning

In this section, we elaborate the architecture of MAL. First, we formally state the problem by providing its mathematical definition. Then we give an overview of the MAL architecture, followed by the specific procedure for adaptively learning features and models. After that, we present the optimization procedure. Finally, we analyze the capacity of the proposed approach in extracting useful information for prediction.

### A. Problem Statement

Let $\mathbf{X} \in \Re^{T_x \times D_x}$ denote the feature set of multiple covariates, where $T_x$ is the length of time lag of a sample and $D_x$ is the size of the covariate features; and $\mathbf{Y} \in \Re^{T_y \times D_y}$ denote the target variable, where $T_y$ is the horizon of prediction and $D_y$ is the dimension of the output. We aim to learn a model to accurately estimate $\mathbf{Y}$ using the features extracted from $\mathbf{X}$:

$$\min \|\mathbf{Y} - \hat{\mathbf{Y}}\|, \hat{\mathbf{Y}} = f(\hat{\mathbf{X}}), \hat{\mathbf{X}} = s(\mathbf{X}), \tag{1}$$

where $\|\cdot\|$ denotes the norm operator used to measure the difference between the ground truth and the prediction, $f(\cdot)$ is the mapping function, $s(\cdot)$ is the feature selection function, $\hat{\mathbf{X}}$ denotes the features selected from the given feature set, and $\hat{\mathbf{Y}}$ denotes the estimation of $\mathbf{Y}$. For example, in spatiotemporal prediction, $\mathbf{Y}$ is the value of variables to be predicted at time step $t$ and $\mathbf{X}$ is the covariate features up to time step $t-1$. The notations used in this article are described in Table I.

TABLE I
Notations and Descriptions

| Notations | Descriptions |
|---|---|
| $\mathbf{Y}$ | The target variable |
| $\mathbf{X}$ | The feature set of multiple covariates |
| $\hat{\mathbf{Y}}$ | The prediction of the target variable |
| $s(\cdot)$ | The feature selection function |
| $\hat{\mathbf{X}}$ or $\mathbf{X^a}$ | Selected feature set |
| $\hat{\mathbf{X}}^m$ | Selected feature set at learning step $m$ |
| $f^m$ | The learning mapping function at learning step $m$ |
| $c^m$ | Criterion value of prediction accuracy at learning step $m$ |
| $\mathbf{a}$ | The selection indicator vector |
| $\bar{\mathbf{a}}$ | The unselection indicator vector |
| $\theta$ | The parameter set of the model |
| $\Theta$ | The parameter set of the prediction function |
| $I(\cdot;\cdot)$ | The mutual information |
| $H(\cdot)$ | The entropy |
| $\mathbb{N}(\mu, \Sigma)$ | Gaussian distribution with mean $\mu$ and covariance $\Sigma$ |
| $\mathbf{h}$ | Learned hidden representation |
| $\lvert\cdot\rvert$ | Cardinality |
| $\mathrm{KL}(\cdot\|\|\cdot)$ | Kullback–Leibler divergence |
| $\mathbf{v}^{(m)}$ | Indicator vector for feature masking |
| $g_i^{(m)}$ | The $i_{th}$ base unit (BU) for the $m_{th}$ block |
| $W^{j\to m}$ | Connection from the $j_{th}$ block to the $m_{th}$ block |
| $o^{(m)}(\mathbf{h}^{(m)})$ | Output function of the $m_{th}$ block |

When encountering the data with complex dependency, an ideal learning approach should be able to identify and select the most important information from the data and make the best use of it by adaptively constructing the learning model. Here, the word "adaptive" or "adaptable" means that the features to be selected, the learning model, and the objective function should be able to self-update so as to optimally achieve the overall learning target. Specifically, the proposed mutually adaptable learning (MAL) can be defined as follows:

*Definition 1. Mutually Adaptable Learning (MAL):* Let $\hat{\mathbf{X}}^m$, $f^m$, and $c^m$ be the selected feature set, the learning mapping function, and the prediction accuracy at the learning step $m$, respectively. Subsequently, during the learning step $m+1$, MAL aims to update the selected feature set from $\hat{\mathbf{X}}^m$ to $\hat{\mathbf{X}}^{m+1}$ and the mapping function from $f^m$ to $f^{m+1}$, so that a better $c^{m+1}$ can be achieved (compared with $c^m$).

### B. Overview of the MAL Architecture

Fig. 2 provides an overview of the MAL architecture. Given the input variables $\mathbf{X}$, MAL aims to learn a model to predict the target variable $\mathbf{Y}$. To achieve this goal, the proposed MAL approach iteratively selects features using the formulation given in the upper box of Fig. 2(a) and adapts the learned model using the objective function shown in the bottom box of Fig. 2(a), so as to gradually refine the information extracted from data and capture the complex dependencies among the data. In each iteration, a new block with the newly selected features (the darker yellow/blue area on the right of Fig. 2(b)) will be constructed and added into the existing model and selected feature set (the lighter yellow/blue area on the left of Fig. 2(b)). Each block, consisting

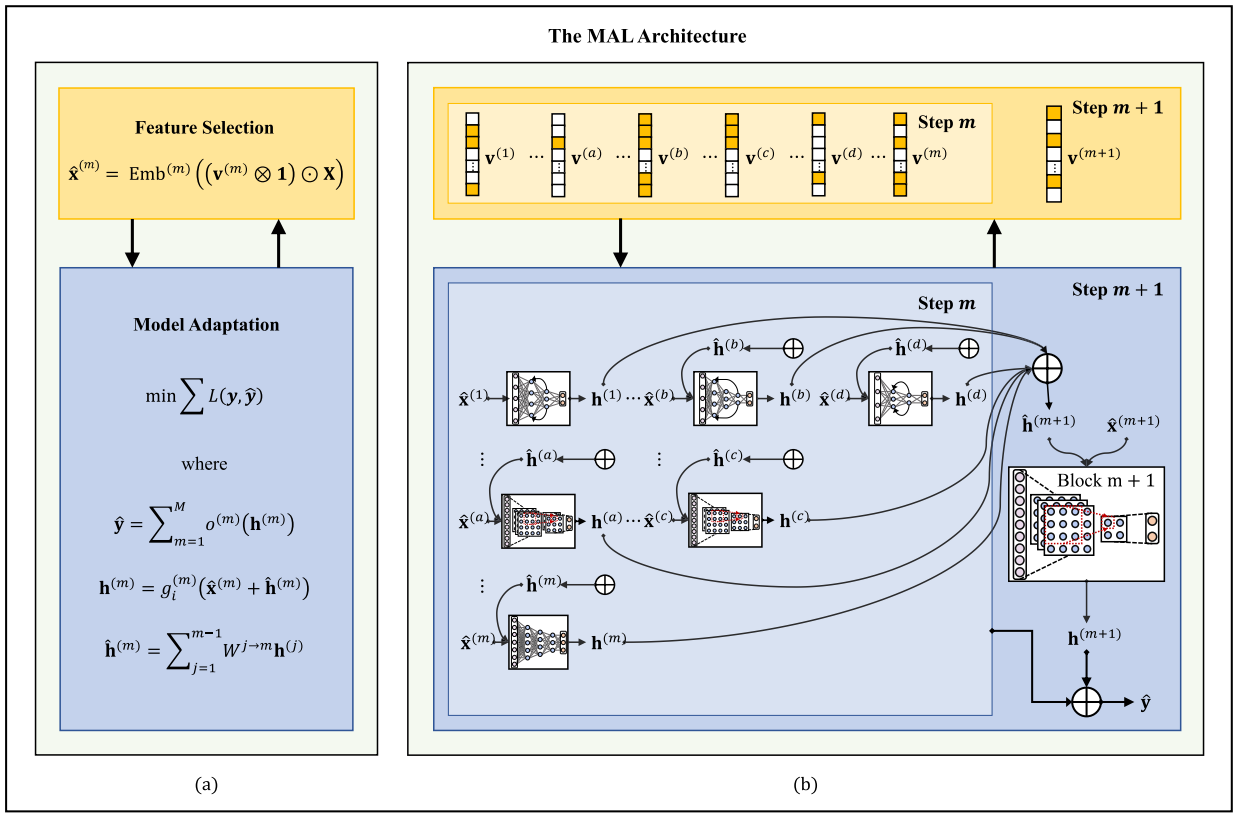



Fig. 2. Overview of the architecture of the proposed Mutually Adaptable Learning (MAL). MAL aims to learn a model from the given input variables to make prediction or inference on the target variable. (a) MAL iteratively selects features and adapt the learned model so as to gradually refine the information extracted from data and capture the complex dependencies among the data. The procedure will continue until convergence. (b) Illustration of the feature selection and model adaptation of the proposed MAL from step $m$ to step $m+1$. The symbol $\oplus$ in the bottom right box denotes the operation $\mathbf{h}^{(m)} = g_i^{(m)}(\hat{\mathbf{x}}^{(m)} + \hat{\mathbf{h}}^{(m)})$ given in the bottom left box of Model Adaptation.

of a feature indicator vector (the top box of Fig. 2(b)) and a base unit (BU) selected from a BU candidate pool (the bottom box of Fig. 2(b)), extracts the useful information from the input data and generates a representation $\mathbf{h}$ used for prediction.

### C. Feature Selection

In MAL, each block is intended to extract a subset of features that contain a specific type of useful information for prediction. Therefore, we formulate an indicator vector mask $\mathbf{v}^{(m)} \in \Re^D$ to indicate the importance of features/variables for the $m_{th}$ block and introduce a sparse regularizer on it, as shown in the top box of Fig. 2(a). Specifically, the embedding layer for the $m_{th}$ block is written as:

$$\hat{\mathbf{x}}^{(m)} = \mathrm{Emb}^{(m)}((\mathbf{v}^{(m)} \otimes \mathbf{1}) \odot X), \tag{2}$$

where $\mathrm{Emb}^{(m)}(\cdot)$ denotes the embedding function of the $m_{th}$ block, $\otimes$ is the Kronecker product, $\odot$ is the Hadamard product, and $\mathbf{1} \in \Re^{1\times T}$ is the vector of all 1 s. To select a compact set of useful features or variables from the high-dimensional space, we introduce the $l_1$ norm regularization on $\mathbf{v}^{(m)}$ during the training. In this work, we implement the $\mathrm{Emb}^{(m)}(\cdot)$ as a linear function. The extension to nonlinear cases is straightforward.

### D. Model Adaptation – Base Unit Selection

After feature selection, we build the BUs in each block to generate the hidden representation from the masked input features needed for the prediction. According to the universal approximation theorem [19], a neural network (NN) with $h$ hidden units can approximate the function with $h$ data points. Therefore, in the following, we use three typical NN models, i.e., the RNN (shown in the first row of the bottom box of Fig. 2(b)), the convolutional NN (CNN, shown in the middle row of the bottom box of Fig. 2(b)), and the multilayer perceptron (MLP, shown in the last row of the bottom box of Fig. 2(b)), as examples of BUs to demonstrate the feasibility and validity of our design. Note that other models, such as the graph neural networks, can also be used as BUs according to various learning requirements and scenarios. And it is not surprising that the inclusion of more types of BUs could further enhance the representation and learning capabilities of the architecture, as certain types of BUs have their own strengths in capturing or characterizing specific categories of information from complex data.

As each type of BU has its unique capacities in characterizing different dependencies, we attempt to automate the choice of BU according to the information to be captured. In various real-world learning tasks, the complex data dependencies are

generally hidden and difficult to be explicitly expressed. As a result, in our design, rather than formulating the explicit representation of data dependencies, we aim to automatically capture such dependencies via loss minimization, i.e., the BU with the smallest loss is the most efficient one to extract the synergistic information from input data and thus should be selected in the current step and integrated into the learning architecture. The rationale behind this design is that if the complex dependencies between data can be well captured, then the prediction should be accurate, that is, the prediction error should be small.

Specifically, during the learning, we pretrain all types of BUs in the first few epochs. After that, we adopt the "winner-gets-training" strategy, i.e., for each batch of training data, we first evaluate the loss (that is, the prediction error) for each type of BU on the current batch, and then only the BU with the smallest loss will be updated (using back-propagation) on the current batch of data. Finally, we select the candidate with the highest training frequency after multiple epochs as the final BU for this block. For instance, if the $i_{th}$ BU is chosen, the hidden representation is generated as:

$$\mathbf{h}^{(m)} = g_i^{(m)}(\hat{\mathbf{x}}^{(m)}), \tag{3}$$

where $g_i^{(m)}$ is the function parameterized by the $i_{th}$ BU for the $m_{th}$ block.

### E. Model Adaptation – Block Construction

To construct an expressive representation and reduce the redundancy between different blocks, we sequentially construct and connect the blocks to guarantee that each of them will take into account the structures of previous blocks during the learning, as shown in the right of the bottom box of Fig. 2(b). As the blocks are sequentially added, we label them based on their order of appearance as blocks $1, 2, \ldots, M$. In such a setting, the latter blocks can access the representations of the preceding blocks – as shown in the bottom box of Fig. 2(b) – block $m+1$ can access the representations of blocks $1, \ldots, m$. As a result, for blocks with index greater than 1 (i.e., when $m \geq 2$), the hidden representation in (3) is updated to the following:

$$\begin{aligned} \mathbf{h}^{(m)} &= g_i^{(m)}(\hat{\mathbf{x}}^{(m)} + \hat{\mathbf{h}}^{(m)}), \\ \hat{\mathbf{h}}^{(m)} &= \sum_{j=1}^{m-1} W^{j\to m}\mathbf{h}^{(j)}, \end{aligned} \tag{4}$$

where $W^{j\to m}$ denotes the connection from the $j_{th}$ block to the $m_{th}$ block. To control the complexity of the learning model, we remove the connections whose weight matrix's $F$-norm is smaller than a threshold $\delta_c$.

### F. Overall Optimization

As described in the last subsection, we iteratively construct a new block for generating informative representations and then integrate the outputs from multiple blocks to generate the overall prediction expressed as:

$$\hat{\mathbf{y}} = \sum_{m=1}^{M} o^{(m)}(\mathbf{h}^{(m)}), \tag{5}$$

where $o^{(m)}$ denotes the output function of the block $m$. In this study, we use the linear output function: $o^{(m)}(\mathbf{h}^{(m)}) = \mathbf{W}^{(m)}\mathbf{h}^{(m)}$, because it has an analytical solution for updating the output layer. Here $\mathbf{W}^{(m)}$ is a matrix. One can also extend this linear function to nonlinear ones according to various learning requirements.

We use a residual learning strategy, in which the existing blocks are fixed when learning the new block. The loss function for joint feature selection and model adaptation over all the training samples is given as follows:

$$\min \sum_{n=1}^{N} L(\mathbf{y}_n, \hat{\mathbf{y}}_n), \text{ where } \hat{\mathbf{y}}_n = \sum_{m=1}^{M} o^{(m)}(\mathbf{h}_n^{(m)}), \tag{6}$$

where $N$ is the number of training samples. We can then use the back-propagation to update the selected features and parameters of BUs within the block. After learning the new block, we selectively post-tune the existing blocks to make all blocks more integrative. Specifically, we update the blocks whose generated hidden representation has relatively high mutual information with that of the new block. Based on the following Proposition 1, adding a new block will not affect the existing blocks that are independent of the new block; updating only the highly related blocks is sufficient for the entire model. The detailed procedure of the proposed MAL is described in Algorithm 1. The code for implementing MAL and the datasets used in our experiments are available at https://github.com/tanqi-github/Mutually-Adaptable-Learning.

*Proposition 1:* Consider a block such that: $\mathbf{h}^{(1)} = g_{\theta^1}(\mathbf{x})$ and $\partial I(\mathbf{y}; \mathbf{h}^{(1)})/\partial\theta^1 = 0$. With fixed $\mathbf{h}^{(1)}$, the MAL learns $\mathbf{h}^{(2)}$ so that $\partial I(\mathbf{y}; [\mathbf{h}^{(1)}, \mathbf{h}^{(2)}])/\partial\theta^2 = 0$. If two blocks are independent, then $\partial I(\mathbf{y}; [\mathbf{h}^{(1)}, \mathbf{h}^{(2)}])/\partial\theta^1 = 0$. Here $I(\cdot;\cdot)$ denotes the

**Algorithm 1:** Mutually Adaptable Learning (MAL).

**Input:** Max number of blocks: $M$; Types of BUs: $\mathcal{U}$; Covariate features: $\mathbf{X}$; Target variable: $\mathbf{y}$
**Output:** Trained model: $\mathcal{L}$

```
Initialize the list of blocks: L ← [];
for m = 1 : M do
    /* Initialize BUs */
    for i = 1 : /U/ do
        Initialize parameters of BUs: u_i^m ∈ U[m];
        L^i ← [L, u_i^m];
        Train u_i^m with the objective min L(y, L^i(X));
    end
    /* Select the best BU */
    u_*^m = arg min_{u_i^m} L(y, L^i(X));
    L ← [L, u_*^m];
    /* Update the model */
    Post-tune L;
end
```

mutual information, and $\theta^1$ and $\theta^2$ are the parameters of these two blocks.

*Proof:*

$$\begin{aligned}\frac{\partial I(\mathbf{y};[\mathbf{h}^{(1)},\mathbf{h}^{(2)}])}{\partial\theta^1} &= \frac{\partial[I(\mathbf{y};\mathbf{h}^{(1)})+I(\mathbf{y};\mathbf{h}^{(2)}|\mathbf{h}^{(1)})]}{\partial\theta^1}\\ &= \frac{\partial I(\mathbf{y};\mathbf{h}^{(2)}|\mathbf{h}^{(1)})}{\partial\theta^1}\\ &= \frac{\partial I(\mathbf{y};\mathbf{h}^{(2)})}{\partial\theta^1} = \mathbf{0}.\end{aligned} \tag{7}$$

The second equality holds because $\frac{\partial I(\mathbf{y};\mathbf{h}^{(1)})}{\partial\theta^1}=0$. The third equality holds because we have $I(\mathbf{h}^{(1)};\mathbf{h}^{(2)})=0$, thus $I(\mathbf{y};\mathbf{h}^{(1)};\mathbf{h}^{(2)})=0$. As a result, we have $I(\mathbf{y};\mathbf{h}^{(2)}|\mathbf{h}^{(1)}) = I(\mathbf{y};\mathbf{h}^{(2)}) - I(\mathbf{y};\mathbf{h}^{(1)};\mathbf{h}^{(2)}) = I(\mathbf{y};\mathbf{h}^{(2)})$. ■

### G. Information Capacity Analysis

After presenting the structure and the optimization procedure of MAL, we analyze its information capacity, i.e., the capacity of extracting useful information to predict target variable. To achieve this goal, we first define the information capacity and per parameter information capacity as follows.

*Definition 2. Information capacity and Per parameter information capacity:* Information capacity is defined as $I(\mathbf{y};\mathbf{h})$ and per parameter information capacity is defined as $I(\mathbf{y};\mathbf{h})/n_c$, where $n_c$ is the parameter size of the BU.

*Information capacity:* We analyze the information capacity of different BUs in capturing the information along the temporal dimension. Let $\theta^i \in \Theta^i$ be the parameter set of the $i_{th}$ type of BU. The information capacity of a specific parameter set is defined as $c_{\theta^i} = \{I(g_{\theta^i}(\mathbf{x});\mathbf{x}_k), k=1,\dots,D\}$. For a fixed structure, the possible values of the information capacity that a BU can have are limited. Thus, we let $F_{\theta^i} = \{c_{\theta^i}, \theta^i \in \Theta^i\}$ be the feasible region of such information capacity. The learning process can be regarded as the procedure of searching the optimal information capacity $c^*$ from $F_{\theta^i}$ to optimize the learning objective. Therefore, one type of BU will be considered as better than the others if its feasible region covers the optimal information capacity with fewer parameters. Without loss of generality, in the following analysis we consider univariate input and a single hidden unit. We first introduce a Lemma to analyze the mutual information between the inputs and the learned representation for a linearized model.

*Lemma 1:* Assume $x_i \sim \mathbb{N}(0,\sigma_i^2)$; if $z=\sum_{i=1}^D \beta_i x_i, \beta_i > 0$, then $I(z;x_i) = \frac{1}{2}\log(1+\frac{\beta_i^2\sigma_i^2}{\sum_{j=1,j\neq i}^D \beta_j^2\sigma_j^2})$.

*Proof:* Because $z=\sum_{i=1}^D \beta_i x_i$, $z \sim \mathbb{N}(0,\sum_{i=1}^D \beta_i^2\sigma_i^2)$, then $H(z) = \frac{1}{2}\log(2\pi e\sum_{i=1}^D \beta_i^2\sigma_i^2)$. Therefore, we can rewrite $z=\beta_i x_i+\epsilon, \epsilon=\sum_{j=1,j\neq i}^D \beta_j x_j$, and thus $\epsilon \sim \mathbb{N}(0,\sum_{j=1,j\neq i}^D \beta_j^2\sigma_j^2))$. Note that $\epsilon$ can be regarded as the noise term in terms of $x_i$, thus $p(z|x_i) = \mathbb{N}(0,\sum_{j=1,j\neq i}^D \beta_j^2\sigma_j^2)$ and $H(z|x_i) = \frac{1}{2}\log(2\pi e\sum_{j=1,j\neq i}^D \beta_j^2\sigma_j^2)$. As a result, we have $I(z;x_i) = H(z) - H(z|x_i) = \frac{1}{2}\log(1+\frac{\beta_i^2\sigma_i^2}{\sum_{j=1,j\neq i}^D \beta_j^2\sigma_j^2})$. ■

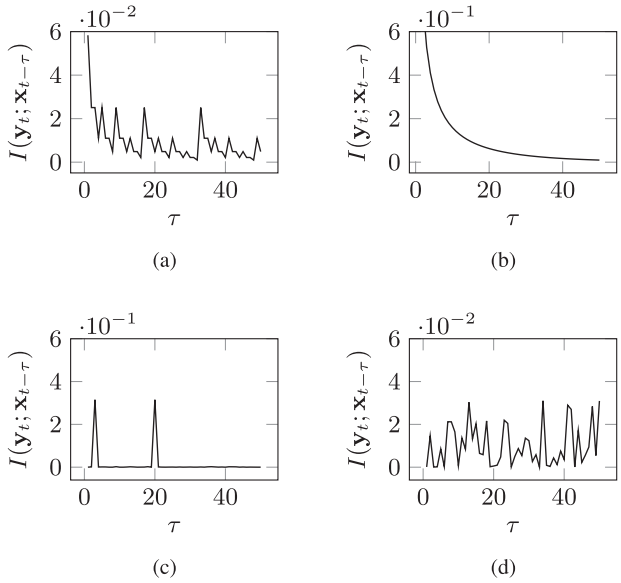



Fig. 3. Information capacity of various base units with specific settings. (a) CNN with left weighting $l=0.4$ and right weighting $r=0.6$. (b) RNN with $W_h=0.95$ and $W_x=1$. (c) MLP with sparse weightings. (d) MLP with random weightings.

We now discuss and analyze the information capacity of the MLP, CNN, and RNN separately.

- *MLP:* As all of the features are first vectorized and then fed into the MLP, the length of the path from each feature to the hidden representation is the same, which makes it easy and flexible to capture the information in arbitrary temporal distance by adapting the weight of each feature as $\beta_i$ in Lemma 1.
- *CNN:* We consider the temporal causal convolutional network proposed in [36] as an example. We study a stack of dilated causal convolutional layers as shown in the Fig. 3 of [36], with 4 layers, exponential dilation (1,2,4 and 8), and the kernel size of 2. Without loss of generality, we consider univariate input and a single hidden unit. For the dilated causal network, the hidden state is calculated as $h=\sum_{\tau=1}\beta_\tau x_{t-\tau}$, $\beta_\tau = \prod_{d=1}^4 l_d^{(\mathbb{1}(\lceil\frac{\tau}{2^{d-1}}\rceil \bmod 2))} r_d^{(1-\mathbb{1}(\lceil\frac{\tau}{2^{d-1}}\rceil \bmod 2))}$, where $\mathbb{1}(\cdot)$ is the indicator function. Substituting it into the Lemma 1, we can obtain the mutual information between the hidden representation and time-lagged inputs.
- *RNN:* We consider the vanilla RNN as an example, whose iterative function can be written as $h_t = \tanh(W_h h_{t-1} + W_x x_t)$. As $h_t$ and $x_t$ are scalars, $W_h$ and $W_x$ are both scalars as well. According to Lemma 1, if we assume that $\sigma_i=1$, then we have $I(h_t;x_t) = \frac{1}{2}\log(1+\frac{W_x}{W_h})$ and $I(h_t;h_{t-1}) = \frac{1}{2}\log(1+\frac{W_h}{W_x})$. Combining these two equations we have

$$I(h_t;x_{t-\tau}) = \frac{1}{2}\log\left(1+\frac{(1-W_h)W_h^\tau}{(1-W_h^\tau)W_x}\right). \tag{8}$$

Fig. 3 shows the information capacity of various BUs with specific settings. We can observe that the information capacity

patterns for different BUs are different, allowing us to select appropriate BUs to capture different types of information among the data.

*Parameter size:* Assuming that the input feature dimension is $D$, the length of input feature is $T$, the hidden size is $s_h$, the kernel size of CNN is $s_k$, and the number of layers is $l$, then the parameter sizes of different BUs are:

- *MLP:* $Ds_hTl$.
- *CNN:* $Ds_hs_kl$.
- *RNN:* $(s_h + D)s_hl$.

*Per parameter information capacity:* Usually we have $s_h \ll D$ and $s_k \geq 3$, so the parameter size of an RNN is smaller than that of a CNN. As a result, an RNN is generally more efficient in capturing the short-range information while a CNN is effective in capturing long-range information. In contrast, an MLP is capable of characterizing both short-range and long-range information. However, its parameter sizes will be huge when the length of the input is large.

## III. Systematic Validation of MAL

In this section, we evaluate the effectiveness of the proposed approach by comparing it with the state-of-the-art methods on various tasks and datasets in two typical domains: spatiotemporal prediction and chaotic behavioral prediction, where complex nonlinear dependencies and relationships are frequently encountered. Note that the intrinsic challenges in spatiotemporal prediction and chaotic behavioral prediction are different. In spatiotemporal prediction, the multi-scale data dependencies are very hard to capture; while in chaotic behavioral prediction, the sensitivity to small perturbations in chaotic dynamical systems is the main challenge. Facing different challenges, we, therefore, select different baseline methods for performance comparison in these two different fields. More details will be provided in the following subsections.

### A. Spatiotemporal Prediction

Spatiotemporal prediction is one of the most representative application domains, as it usually shows complex dependencies of data along both spatial and temporal dimensions at multiple scales. Therefore, we first apply our approach to three typical spatiotemporal prediction tasks: traffic prediction, infectious disease prediction, and climate forecasting.

*Data description:* We first describe the datasets used in these three tasks.

- Traffic prediction: The traffic jam index data were collected by the Shanghai Urban and Rural Construction and Traffic Development Academy in April 2015. The dataset was released by the organizing committee of the Shanghai Open Data Apps (Season Information Technology Co. Ltd. Shanghai Open Data Apps (2015).[1] The dataset contains 2,160 time instants and 68 spatial regions.
- *Disease prediction:* Weekly estimates of influenza activity for 29 countries were released by Google.[2] We used the data from the years 2006 to 2015. We further collected the temperature data for these 29 countries.[3]
- *Climate forecasting:* This dataset was collected in the Central Region of the United States of America, and included the weekly air temperature data at 2 meters height and the precipitation data from the North American Regional Reanalysis (NARR).[4] We selected the data from the years 1980 to 2017 in the spatial region of the USA defined by the coordinates (Lon: −97.095E – −91.476E, Lat: 37.829 N – 40.2632 N).

*Baseline methods for comparison:* We compare the performance of our approach with the following representative methods.

- *Long short-term memory (LSTM) [18]:* It is a traditional RNN model to capture the long-range and short-range temporal dependencies of data.
- *Sparse group Lasso neural network (SGNN) [41]:* It trains a compact DNN using group sparsity for feature selection and prediction.
- *Dual-stage attention-based RNN (DARNN) [38]:* It uses the attention model first on the variable dimension and then on the temporal dimension to extract the important variables and time slices.
- *Temporal attention-augmented bilinear network (TABL) [45]:* It is composed of bilinear modules to capture the relationship in both spatial and temporal modes. The temporal attention is incorporated after the bilinear modules to select important time slices.
- *MLCNN:* Multi-level construal neural network (MLCNN) [11]: It is a multitask-like model that includes a shared recurrent component for the near and distant time horizon tasks to improve the performance of the current time horizon task.

*Settings and results:* We evaluate the one-step-ahead prediction accuracy of different methods, i.e., the temporal horizon of the target variable is 1. The length of the time lag of a sample feature is 48 (each represents 10 minutes) in traffic prediction, 60 (each represents one week) in disease prediction, and 60 (each represents one week) in climate forecasting.

The prediction of the target variable in each spatial location is regarded as one learning task. In each task, we use the data samples in the first 70% time steps for training the prediction model and the remaining 30% for testing. We compare the performance of different methods in terms of the RMSE and the mean absolute error (MAE). Table II shows the RMSE and MAE values of our approach and those of the aforementioned five models on three spatiotemporal prediction tasks. By selecting the useful features and making use of them to refine the learning model in an adaptable and integrative manner, the proposed approach achieves the best performance among all of the methods in all three tasks. Note that the values in the Google-Flu data range from 0 to 10555. Therefore, it is reasonable to have relatively large RMSE values of all methods in this dataset.

Fig. 4 illustrates the performance of our approach in the task of Shanghai traffic jam prediction. Fig. 4(a) shows the bubble

[1]http://soda.datashanghai.gov.cn/
[2]http://www.google.org/flutrends
[3]https://climateknowledgeportal.worldbank.org/
[4]https://www.esrl.noaa.gov/psd/data/gridded/data.narr.monolevel.html

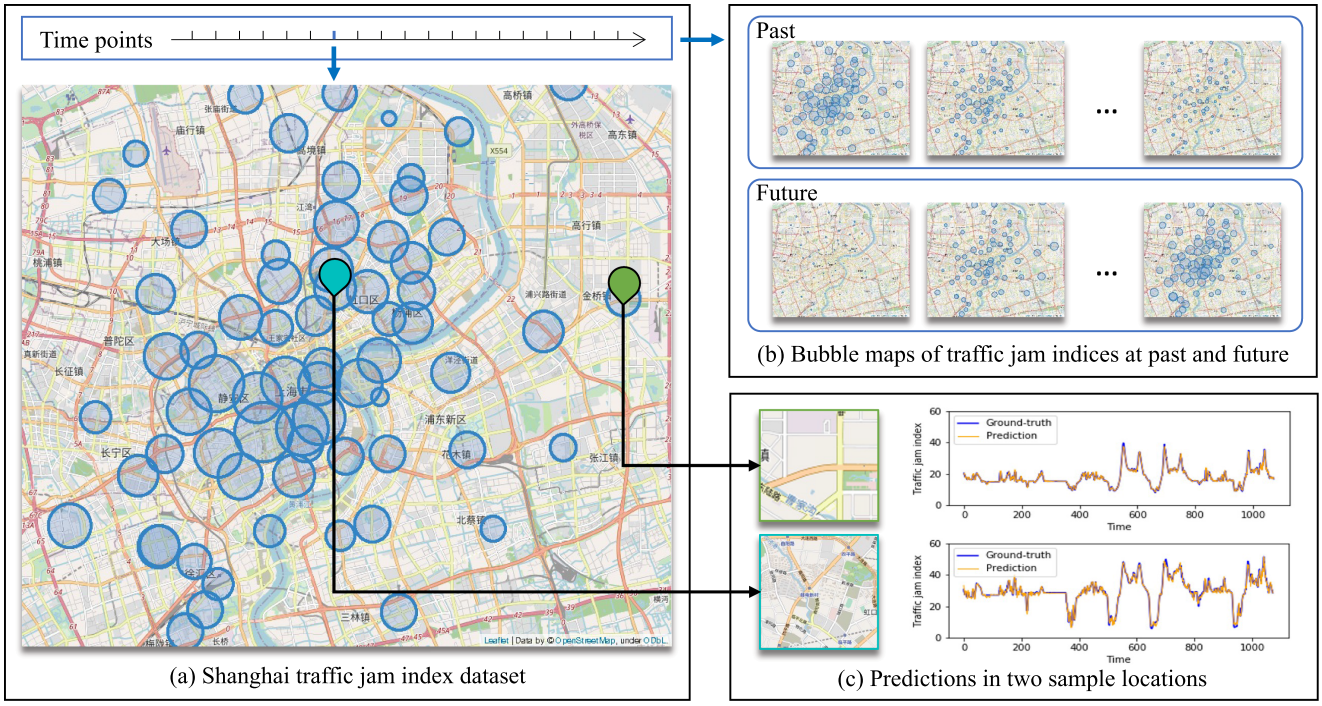



Fig. 4. Illustration of one typical spatiotemporal prediction task: traffic jam prediction in Shanghai. (a) The bubble map of traffic jam data in a certain 10 minutes on an enlarged map. (b) The bubble maps of the traffic jam data over time. (c) Comparison between the ground truth (the blue line) and MAL's prediction (the orange line) of traffic jam over time in two sample locations.

TABLE II
PERFORMANCE COMPARISON (IN TERMS OF RMSE AND MAE) BETWEEN THE PROPOSED MAL AND FIVE REPRESENTATIVE MODELS LSTM [18], SGNN [41], DARNN [38], TABL [45], AND MLCNN [11]) FOR THREE SPATIOTEMPORAL PREDICTION TASKS: TRAFFIC PREDICTION, DISEASE PREDICTION, AND CLIMATE FORECASTING

| Methods | Spatiotemporal Prediction | | | | | |
|---|---|---|---|---|---|---|
| | SH-Traffic | | Google-Flu | | US-Climate | |
| | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| LSTM-uni | 1.836 | 1.319 | 47.871 | 30.073 | 10.047 | 8.675 |
| LSTM-multi | 1.526 | 1.148 | 65.684 | 44.790 | 6.039 | 5.018 |
| SGNN | 1.772 | 1.358 | 52.849 | 33.227 | 3.641 | 2.880 |
| DARNN | 1.709 | 1.306 | 66.866 | 43.845 | 6.872 | 5.798 |
| TABL | 1.154 | 0.941 | 67.406 | 53.587 | 9.981 | 9.067 |
| MLCNN | 1.733 | 1.353 | 123.670 | 90.409 | 5.225 | 4.063 |
| MAL | **1.112** | **0.814** | **40.910** | **26.249** | **3.498** | **2.743** |

The best performances among all six methods for different tasks are highlighted in bold-face.

map of traffic jam indices of all locations in a certain 10 minutes on an enlarged map while Fig. 4(b) provides the maps at multiple time points. Fig. 4(c) demonstrates MAL's prediction of traffic jam over time in two sample locations, together with the ground truth. We can observe that the prediction (the orange line) is close to the ground truth (the blue line) in terms of both overall trend and individual values, reflecting the ability of the proposed approach in capturing complex dependencies of spatiotemporal data and making accurate predictions accordingly.

### B. Chaotic Behavioral Prediction

In this part, we evaluate the performance of the proposed approach on chaotic behavioral prediction, which is another representative domain of complex data dependency learning demonstrating the highly nonlinear behaviors governed by the differential equations of chaotic systems. With the complex nonlinearity, even very similar initial states of the system could generate quite different future dynamics, making accurate prediction difficult.

*Task description:* We consider three classical chaotic systems in our evaluation. They are the Lorenz system, double pendulum system, and triple pendulum system.

- Lorenz system: It is a system governed by ordinary differential equations, with chaotic solutions for certain values of parameters and initial conditions. The differential equations can be written as: $\dot{x} = \sigma(y - x)$, $\dot{y} = x(\rho - z) - y$, $\dot{z} = xy - \beta z$. In this task, our target is to predict the coordinates of all three dimensions at different time instants.
- Double (triple) pendulum systems: A double (triple) pendulum system is a physical system with a pendulum connected to another one (two) pendulums at its end, showing highly nonlinear dynamical behavior with a strong sensitivity to initial conditions [2], [46], [52]. In these experiments, we consider the coordinates of the trajectory as the target variables to be predicted.

*Baseline methods for comparison:* We compare the performance of our approach with two methods, one being the classical LSTM, the other being the representative echo state network (ESN). The ESN [20] is a kind of recurrent neural networks that performs particularly well and has been widely used in chaotic system modeling. Among ESN and its variants [14], [34], the RC-ESN [6], which is a state-of-the-art method in the ESN family, has reported advanced performance in chaotic behavioral prediction. Therefore, we adopt the RC-ESN as a

TABLE III
PERFORMANCE COMPARISON (IN TERMS OF RMSE AND MAE) OF THE PROPOSED MAL AND TWO REPRESENTATIVE MODELS (LSTM [18] AND RC–ESN [6]) IN THREE CHAOTIC BEHAVIORAL PREDICTION TASKS: LORENZ SYSTEM, DOUBLE PENDULUM SYSTEM, AND TRIPLE PENDULUM SYSTEM

| Methods | Chaotic Behavioral Prediction | | | | | |
|---|---|---|---|---|---|---|
| | Lorenz | | Double Pendulum | | Triple Pendulum | |
| | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| LSTM | 9.285 | 7.308 | 0.861 | 0.688 | 1.049 | 0.863 |
| RC–ESN | 1.475 | 1.137 | 0.525 | 0.444 | 0.624 | 0.466 |
| MAL | **0.964** | **0.427** | **0.290** | **0.239** | **0.495** | **0.400** |

The best performances among all six methods for different tasks are highlighted in bold-face.

baseline method in our performance comparison. The RC-ESN utilizes the input data to drive an RNN, so as to induce a nonlinear response function in each node of this "reservoir" network. It then trains a linear combination of all these nonlinear response functions as the output of the model.

*Settings and results:* For the Lorenz system, we set $\sigma = 10, \rho = 28, \beta = 8/3$ in the differential equations to generate the chaotic time series. For the double and triple pendulum systems, we set the masses of the pendulums, i.e., $m_1$ and $m_2$, to be 1 and the lengths of the limbs, i.e., $l_1$ and $l_2$, to be 1 as well. For all three systems, we sample at a fixed discrete time step $dt = 0.01$. We use the first 15,000 time steps for training and the remaining 10,000 time steps for testing. Following the settings in [6], the reservoir size of the RC–ESN model is set to 5,000 and the size of the hidden layers of LSTM is set to 50. For our approach, we set the hidden size for each block as 400 and train only the output layer for prediction. For all methods, we predict the target variable at 20 time steps ahead.

Table III provides the comparison results of the prediction performance of LSTM, RC–ESN, and MAL on three chaotic systems. Equipped with the capacity of progressive adaptation of both features and models, MAL performs better than the classical LSTM and the popular RC–ESN in all three tasks.

On the left of Fig. 5(a)–(c), we visualize the dynamical behaviors of three typical chaotic systems: Lorenz system, double pendulum system, and triple pendulum system, respectively; on the right of these sub-figures, we show the corresponding prediction results generated by the proposed MAL (the orange line) together with the ground truth (the blue line) of the future time series of these three chaotic systems. Similar to the results shown in the traffic prediction, our forecasting on the chaotic behaviors is also close to the ground truth, showing MAL's capacity in capturing the complex and irregular long-term dependencies caused by the extreme sensitivities to small perturbations and the nonlinear coupling and interactions of variables in such challenging chaotic systems.

### C. Discussion

In addition to the quantitative evaluation of MAL's performance, we further analyze and discuss its behavior during the learning procedure. Specifically, we use the **traffic prediction task** as an example to investigate the progression of MAL's learning procedure.

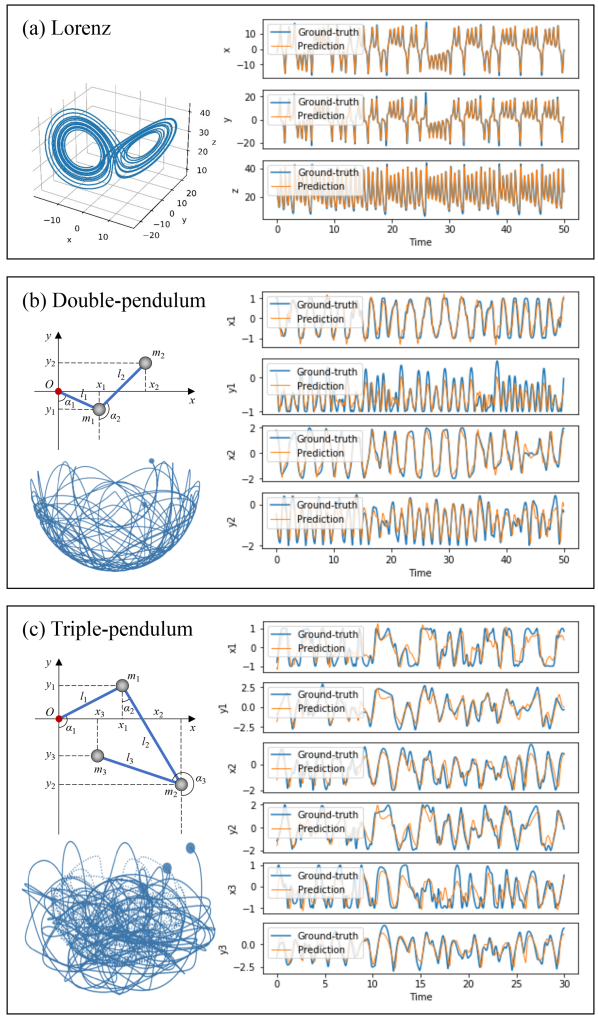



Fig. 5. Illustration of MAL's predictions of chaotic behaviors of (a) Lorenz system, (b) double pendulum system, and (c) triple pendulum system.

First, we observe the performance enhancement of MAL along with the progress of learning. Fig. 6 shows the normalized testing error (NTE) with the increase in block number at 14 uniformly selected locations. For the prediction of each location, the NTE with $M$ blocks, denoted as $NTE_M$, is calculated as $NTE_M = \frac{[\text{RMSE}]_M - \min([\text{RMSE}])}{\max([\text{RMSE}]) - \min([\text{RMSE}])}$, where [RMSE] is a $10 \times 1$ vector, with the $M$-th element, $[\text{RMSE}]_M$, being the RMSE value of the model with $M$ blocks ($M = 1, \ldots, 10$). As seen from Fig. 6, the testing error of the proposed MAL keeps decreasing as the number of blocks increases, showing the effectiveness of the proposed learning strategy. Moreover, the learning performance becomes stable when the number of blocks reaches 5, demonstrating the validity of our convergence analysis.

Then we explore the progression of block update during the learning procedure. Specifically, we examine the features selected by different blocks as well as the change of feature importance along the learning procedure. We use salience maps to reveal the selected features and their importance. Fig. 7 shows the selected features (the solid circles at the corresponding

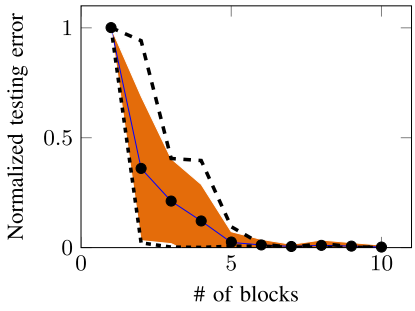



Fig. 6. Normalized testing error on 14 locations with the increase of block number in the traffic prediction task. The blue solid line indicates the mean value of the normalized testing error on 14 locations, and two dashed lines indicate the maximum and minimum normalized testing errors on 14 locations, respectively. The orange region shows the area within one standard deviation.

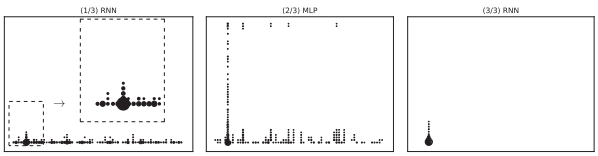



Fig. 7. Salience maps of three learned blocks. (a) The first block (with the RNN BU selected). (b) The second block (with the MLP BU selected). (c) The third block (with the RNN BU selected). The y-axis indicates the time lag and the x-axis indicates different features (i.e., locations). The solid circle indicates that the corresponding feature is significant (salience value $> 0.1$) while its size indicates the magnitude of the salience value. The plot with the dashed bounding box on the top right corner of the first subfigure is the magnified version of the small plot on the lower left corner.

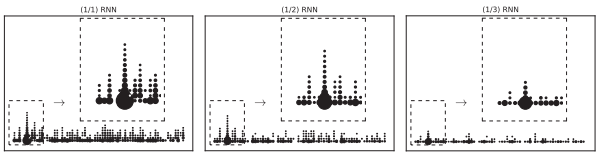



Fig. 8. Salience maps of the first learned block (with the RNN BU selected) at the (a) first iteration, (b) second iteration, and (c) third iteration. The y-axis indicates the time lag and the x-axis indicates different features (i.e., locations). The solid circle indicates that the corresponding feature is significant (salience value $> 0.1$) while its size indicates the magnitude of the salience value. The plot with the dashed bounding box on the top right corner of each subfigure is the magnified version of the small plot on the lower left corner of the same subfigure.

positions) and the feature importance (the size of the solid circle) in the first (Fig. 7(a)), second (Fig. 7(b)), and third (Fig. 7(c)) learned blocks. We can observe that different blocks select different types of features. The features selected by the first RNN block capture the global short-range dependencies; the features extracted by the second MLP block capture the long-term dependencies; and the features identified by the third RNN block capture the local short-range dependencies. Moreover, even existing blocks are able to update themselves during the learning procedure. Fig. 8 illustrates the selected features and their importance for the first RNN block in the first (Fig. 8(a)), second (Fig. 8(b)), and third (Fig. 8(c)) learning iterations. As the learning process continues, this RNN block gradually tends to update its ability to capture the short-term dependencies, as it is found that the features selected by the second MLP block are sufficient for characterizing the long-term dependencies.

Note that in our experiments, although the proposed MAL selects and incorporates the CNN as the BU in several cases, the number of times the CNN has been selected as the BU is fewer than that of RNN. The possible reason is that most of the applications used for validation are temporal tasks. Compared with CNN, RNN is more effective in capturing the temporal dependency and thus has been more frequently selected as the BU. While the RNN is relatively powerful in characterizing temporal information, CNN could be potentially beneficial in some scenarios. As shown in Fig. 3(a), the CNN captures both the periodic information with decay trend and the strong local information ($\tau < 3$). This indicates that the CNN, as a basic unit, is useful when the proposed MAL needs to extract the periodic patterns with trends or rich local patterns.

## IV. Information-Theoretic Analysis of MAL

In this section, we conduct the information-theoretic analysis of MAL, including the information-theoretic perspective of the learning objective, the information-characterized synergistic effects, and the convergence of the learning procedure.

### A. Information-Theoretic-Based Learning Objective

To quantitatively characterize the amount of information carried by the data and objectively determine the information needed for a given learning task, we examine the learning objective from an information-theoretic perspective.

The prediction accuracy, denoted by $c$, measures the gap between the prediction made by the learning model and the ground truth. Probabilistically, the prediction accuracy can be expressed in the form of the likelihood function $q(\mathbf{Y}|\mathbf{X}, \Theta)$. Here we use $\Theta$ to denote the parameter set of the prediction function that generates the prediction from $\mathbf{X}$. From this aspect, the learning objective can be regarded as maximizing the following log-likelihood:

$$\max_{\Theta} \mathbb{E}[\log q(\mathbf{Y}|\mathbf{X}, \Theta)]. \tag{9}$$

We can further express $\Theta$ as $\Theta = \{\mathbf{a}, \theta\}$, in which $\mathbf{a} \in \{0, 1\}^D$ is the selection indicator vector corresponding to a set of selected features, with $\mathbf{a}_j = 1$ indicating that the $j^{th}$ feature is selected and 0 otherwise, and $\theta$ is the parameter set of the model. Therefore, we can also denote $\hat{\mathbf{X}}$ (the set of selected features) as $\mathbf{X}^{\mathbf{a}}$ and the set of unselected features as $\mathbf{X}^{\bar{\mathbf{a}}}$. In the following proposition, we connect the proposed learning objective with the mutual information and conditional entropy.

*Proposition 2:* Optimizing the formulated learning objective, i.e., $\max_{\Theta} \mathbb{E}[\log q(\mathbf{Y}|\mathbf{X}, \Theta)]$, is equivalent to jointly optimizing $\max_{\mathbf{a}}(I(\mathbf{Y}; \mathbf{X}^{\mathbf{a}}))$ and $\max_{\theta}(H(\mathbf{Y}|\mathbf{X}^{\mathbf{a}}) - H(\mathbf{Y}|f_{\theta}(\mathbf{X}^{\mathbf{a}})))$, where $I(\mathbf{Y}; \mathbf{X}^{\mathbf{a}})$ denotes the mutual information between the target variable and the selected feature set, and $H(\mathbf{Y}|\mathbf{X}^{\mathbf{a}})$ and

$H(\mathbf{Y}|f_\theta(\mathbf{X^a}))$ denote the conditional entropy of $\mathbf{Y}$ given $\mathbf{X^a}$ and that given $f_\theta(\mathbf{X^a})$, respectively.

*Proof:* According to [5], the scaled log-likelihood in (9) can be rewritten as:

$$\begin{aligned} l &= \mathbb{E}[\log q(\mathbf{Y}|\mathbf{X}_i^{\mathbf{a}}, \theta)] \\ &= \mathbb{E}[\log \frac{q(\mathbf{Y}|\mathbf{X}_i^{\mathbf{a}}, \theta)}{p(\mathbf{Y}|\mathbf{X}_i^{\mathbf{a}})} + \log \frac{p(\mathbf{Y}|\mathbf{X}_i^{\mathbf{a}})}{p(\mathbf{Y}|\mathbf{X}_i)} + \log p(\mathbf{Y}|\mathbf{X}_i)] \\ &= H(\mathbf{Y}|\mathbf{X^a}) - H(\mathbf{Y}|f_\theta(\mathbf{X^a})) - I(\mathbf{X}^{\bar{\mathbf{a}}}; \mathbf{Y}|\mathbf{X^a}) - H(\mathbf{Y}|\mathbf{X}) \\ &= H(\mathbf{Y}|\mathbf{X^a}) - H(\mathbf{Y}|f_\theta(\mathbf{X^a})) + I(\mathbf{Y}; \mathbf{X^a}) - H(\mathbf{Y}). \end{aligned} \tag{10}$$

In (10), the term $H(\mathbf{Y})$ is a constant. Therefore, the objective of maximizing $\mathbb{E}[\log q(\mathbf{Y}|\mathbf{X}_i^{\mathbf{a}}, \theta)]$ is equivalent to maximizing $H(\mathbf{Y}|\mathbf{X^a}) - H(\mathbf{Y}|f_\theta(\mathbf{X^a})) + I(\mathbf{Y}; \mathbf{X^a})$. ■

Building upon the connection between the learning objective and the mutual information, we will further show that the learning objective is achievable (i.e., bounded). The objective function in (10) can be further rewritten as:

$$\begin{aligned} & H(\mathbf{Y}|\mathbf{X^a}) - H(\mathbf{Y}|f_\theta(\mathbf{X^a})) + I(\mathbf{Y}; \mathbf{X^a}) - H(\mathbf{Y}) \\ &= H(\mathbf{Y}) - H(\mathbf{Y}|f_\theta(\mathbf{X^a})) - H(\mathbf{Y}) \\ &= I(\mathbf{Y}; f_\Theta(\mathbf{X})) - H(\mathbf{Y}). \end{aligned} \tag{11}$$

Therefore, the original objective is equivalent to $\max_\Theta I(\mathbf{Y}; f_\Theta(\mathbf{X}))$. To derive the upper bound of $I(\mathbf{Y}; f_\Theta(\mathbf{X}))$, we first introduce the following lemma:

*Lemma 2:* For three random variables $z_1, z_2, z_3$, if $z_1$ and $z_3$ are conditionally independent given $z_2$, i.e., $(z_1 \perp\!\!\!\perp z_3)|z_2$, then $I(z_1; z_3) \leq I(z_1; z_2)$.

*Proof:* The following equality holds:

$$\begin{aligned} I(z_1; [z_2, z_3]) &= I(z_1; z_2) + I(z_1; z_3|z_2) \\ &= I(z_1; z_3) + I(z_1; z_2|z_3). \end{aligned} \tag{12}$$

Because $(z_1 \perp\!\!\!\perp z_3)|z_2$, we know that $I(z_1; z_3|z_2) = 0$. Thus we have $I(z_1; z_3) = I(z_1; z_2) - I(z_1; z_2|z_3) \leq I(z_1; z_2)$. Using a similar procedure, we obtain $I(z_1; z_3) \leq I(z_2; z_3)$. ■

Because $f_\Theta(\mathbf{X})$ is generated from $\mathbf{X}$, then $(f_\Theta(\mathbf{X}) \perp\!\!\!\perp \mathbf{Y})|\mathbf{X}$. According to Lemma 1, we have $I(\mathbf{Y}; f_\Theta(\mathbf{X})) \leq I(\mathbf{Y}; \mathbf{X})$, i.e., $I(\mathbf{Y}; f_\Theta(\mathbf{X}))$ is bounded by $I(\mathbf{Y}; \mathbf{X})$. Therefore, we only need to prove that $I(\mathbf{Y}; \mathbf{X})$ is upper bounded. In the general case, the temporal data with multiple variables can be concatenated into vector representations, and thus, in the following analysis, we use $\mathbf{y}$ and $\mathbf{x}$ to denote the target variable and input variables, respectively. Assume that $\mathbf{y}$ is the observation of the latent state $\mathbf{g}_y$ and $\mathbf{x}$ is the observation of the latent state $\mathbf{g}_x$; then, we have the following theorem.

*Theorem 1:* $I(\mathbf{y}; \mathbf{x})$ is bounded if any of the three terms, $p(\mathbf{x}|\mathbf{g}_x)$, $p(\mathbf{y}|\mathbf{g}_y)$, or $p(\mathbf{g}_y|\mathbf{g}_x)$, is stochastic.

*Proof:* According to the Markov blanket [4], in the model of $\mathbf{x} \leftarrow \mathbf{g}_x \rightarrow \mathbf{g}_y \rightarrow \mathbf{y}$, we have $(\mathbf{x} \perp\!\!\!\perp \mathbf{y})|\mathbf{g}_x$ and $(\mathbf{g}_x \perp\!\!\!\perp \mathbf{y})|\mathbf{g}_y$. According to Lemma 1, we have $I(\mathbf{x}; \mathbf{y}) \leq I(\mathbf{x}; \mathbf{g}_x)$, $I(\mathbf{x}; \mathbf{y}) \leq I(\mathbf{y}; \mathbf{g}_x)$, $I(\mathbf{g}_x; \mathbf{y}) \leq I(\mathbf{g}_x; \mathbf{g}_y)$ and $I(\mathbf{g}_x; \mathbf{y}) \leq I(\mathbf{y}; \mathbf{g}_y)$. In summary, $I(\mathbf{x}; \mathbf{y}) \leq \min[I(\mathbf{x}; \mathbf{g}_x), I(\mathbf{y}; \mathbf{g}_x), I(\mathbf{g}_x; \mathbf{y}), I(\mathbf{g}_x; \mathbf{g}_y), I(\mathbf{y}; \mathbf{g}_y)]$. If any of the three terms, $p(\mathbf{x}|\mathbf{g}_x)$, $p(\mathbf{y}|\mathbf{g}_y)$, or $p(\mathbf{g}_y|\mathbf{g}_x)$, is stochastic, then the corresponding mutual information $I(\mathbf{x}; \mathbf{g}_x)$, $I(\mathbf{y}; \mathbf{g}_y)$, or $I(\mathbf{g}_x; \mathbf{g}_y)$ is bounded; so $I(\mathbf{x}; \mathbf{y})$ is bounded. ■

### B. Characterization of Synergistic Effects

In this subsection, we first show the existence of synergistic effects in complex data, which is a critical challenge in learning tasks. Based on that, we further prove that MAL is able to capture such synergistic effects in a progressive way.

*Proposition 3:* In learning tasks with complex dependency, there exist synergistic effects such that $I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2]) > I(\mathbf{y}; \mathbf{x}_1) + I(\mathbf{y}, \mathbf{x}_2)$.

*Proof:* Consider $\begin{bmatrix} \mathbf{y} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \sim \mathbb{N}\left( \begin{bmatrix} \mu_y \\ \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_y & \Sigma_{y1} & \Sigma_{y2} \\ \Sigma_{1y} & \Sigma_1 & \Sigma_{12} \\ \Sigma_{2y} & \Sigma_{21} & \Sigma_2 \end{bmatrix} \right)$. Without loss of generality, assume $dim(\mathbf{y}) = 1$. We then have

$$\begin{aligned} I(\mathbf{y}; \mathbf{x}_1) &= H(\mathbf{y}) - H(\mathbf{y}|\mathbf{x}) \\ &= \frac{1}{2} \ln (2\pi e \Sigma_y) - \frac{1}{2} \ln \left(2\pi e (\Sigma_y - \Sigma_{1y}^T \Sigma_1^{-1} \Sigma_{1y})\right) \\ &= -\frac{1}{2} \ln \left(1 - \frac{\Sigma_{1y}^T \Sigma_1^{-1} \Sigma_{1y}}{\Sigma_y}\right). \end{aligned} \tag{13}$$

Similarly, we have:

$$I(\mathbf{y}; \mathbf{x}_2) = -\frac{1}{2} \ln \left(1 - \frac{\Sigma_{2y}^T \Sigma_2^{-1} \Sigma_{2y}}{\Sigma_y}\right),$$

$$I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2]) = -\frac{1}{2} \ln \left(1 - \frac{\begin{bmatrix} \Sigma_{1y} \\ \Sigma_{2y} \end{bmatrix}^T \begin{bmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_2 \end{bmatrix}^{-1} \begin{bmatrix} \Sigma_{1y} \\ \Sigma_{2y} \end{bmatrix}}{\Sigma_y}\right). \tag{14}$$

Next, we use two typical cases to show the existence of synergistic effects.

*Case 1:* We show the existence of the synergistic effects even if $\mathbf{x}_1$ and $\mathbf{x}_2$ are independent. Let $\Sigma_y = 1$, $\Sigma_1 = \mathbf{I}$, $\Sigma_2 = \mathbf{I}$, $\Sigma_{12} = \mathbf{0}$ (independence of $\mathbf{x}_1$ and $\mathbf{x}_2$), $\Sigma_{y1} = 0.2 * \mathbf{1}$, and $\Sigma_{y2} = 0.2 * \mathbf{1}$. We then have $I(\mathbf{y}; \mathbf{x}_1) + I(\mathbf{y}; \mathbf{x}_2) = -\frac{1}{2} \ln(1 - 0.04|\mathbf{x}_1|) - \frac{1}{2} \ln(1 - 0.04|\mathbf{x}_2|)$ and $I(\mathbf{y}; \mathbf{x}_2|\mathbf{x}_1) = -\frac{1}{2} \ln(1 - 0.04(|\mathbf{x}_1| + |\mathbf{x}_2|))$, where $|\mathbf{x}_1|$ and $|\mathbf{x}_2|$ are the dimensions of $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. Without loss of generality, let $|\mathbf{x}_1| = |\mathbf{x}_2| = 10$, then $I(\mathbf{y}; \mathbf{x}_1) + I(\mathbf{y}; \mathbf{x}_2) = 0.5108$ and $I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2]) = 0.8047$. Obviously, we have $I(\mathbf{y}; \mathbf{x}_1) + I(\mathbf{y}; \mathbf{x}_2) < I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2])$ in this case, i.e., the synergistic effects exist.

*Case 2:* We show that the synergistic effects also exist even if $\mathbf{y}$ and any of the $\mathbf{x}_1$, $\mathbf{x}_2$ are independent. Let $\Sigma_y = 1$, $\Sigma_1 = \mathbf{I}$, $\Sigma_2 = \mathbf{I}$, $\Sigma_{12} = 0.2 * \mathbf{I}$, $\Sigma_{y1} = 0.2 * \mathbf{1}$, and $\Sigma_{y2} = \mathbf{0}$ (independence of $\mathbf{y}$ and $\mathbf{x}_2$). We then have $I(\mathbf{y}; \mathbf{x}_2) = 0$ and $I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2]) = -\ln(1 - \Sigma_{y1}(\mathbf{I} - \Sigma_{12}\Sigma_{12})^{-1}\Sigma_{y1}^T) = 0.8959$. Again, we have $I(\mathbf{y}; \mathbf{x}_1) + I(\mathbf{y}; \mathbf{x}_2) < I(\mathbf{y}; [\mathbf{x}_1, \mathbf{x}_2])$, showing the existence of synergistic effects in this case. ■

*Remark:* We further analyze the condition under which the data inter-relation $\Sigma_{12}$ gives rise to the synergistic effects. Without loss of generality, we assume that $x_1$, $x_2$, and $y$ are univariate variables with unit variance, for simplicity of notation.

We then use $\sigma_*$ for a univariate variable to replace $\Sigma_*$, and we have $\sigma_{11} = \sigma_{22} = \sigma_{33} = 1$. We then have:

$$I(y;x_1) + I(y;x_2) = -\frac{1}{2}\ln\left(1-\sigma_{y1}^2-\sigma_{y2}^2+\sigma_{y1}^2\sigma_{y2}^2\right),$$
$$I(y;[x_1,x_2]) = -\frac{1}{2}\ln\left(1-\frac{\sigma_{y1}^2+\sigma_{y2}^2-2\sigma_{12}\sigma_{y1}\sigma_{y2}}{(1-\sigma_{12}^2)}\right). \tag{15}$$

Letting $\frac{\sigma_{y1}^2+\sigma_{y2}^2-2\sigma_{12}\sigma_{y1}\sigma_{y2}}{(1-\sigma_{12}^2)} > \sigma_{y1}^2+\sigma_{y2}^2-\sigma_{y1}^2\sigma_{y2}^2$, we have

$$(\sigma_{y1}^2+\sigma_{y2}^2-\sigma_{y1}^2\sigma_{y2}^2)\sigma_{12}^2 - 2\sigma_{y1}\sigma_{y2}\sigma_{12} + \sigma_{y1}^2\sigma_{y2}^2 > 0, \tag{16}$$

which is a general quadratic function of $\sigma_{12}$. The roots for the quadratic function are $\frac{\sigma_{y1}\sigma_{y2}\pm\sigma_{y1}\sigma_{y2}\sqrt{(1-\sigma_{y1}^2)(1-\sigma_{y2}^2)}}{\sigma_{y1}^2+\sigma_{y2}^2-\sigma_{y1}^2\sigma_{y2}^2}$.

In fact, what we are interested in is the synergistic gain of integrating two information sources from different steps of MAL for prediction, i.e., $G_Y(\mathbf{x}_1,\mathbf{x}_2) = I(\mathbf{x}_2;\mathbf{y}|\mathbf{x}_1) - I(\mathbf{x}_2;\mathbf{y})$.[5] Therefore, in the following, we demonstrate that MAL can capture such synergistic effects with respect to the adaptation of features and the model.

*Theorem 2:* For $\mathbf{h}_1 = \arg\max I(\mathbf{y};\mathbf{h}_1)$, there exists $\mathbf{h}_2 = \arg\max I(\mathbf{y};\mathbf{h}_2|\mathbf{h}_1)$, such that $I(\mathbf{y};[\mathbf{h}_1,\mathbf{h}_2]) > I(\mathbf{y};\mathbf{h}_1) + I(\mathbf{y};\mathbf{h}_2)$. Here $\mathbf{h}_1$ and $\mathbf{h}_2$ denote the latent representations learned by MAL at two consecutive iterations, respectively.

*Proof:* Consider that $\begin{bmatrix}\mathbf{y}\\ \mathbf{x}\end{bmatrix} \sim \mathbb{N}\left(\begin{bmatrix}\mu_y\\ \mu_x\end{bmatrix}, \begin{bmatrix}\Sigma_y & \Sigma_{yx}\\ \Sigma_{xy} & \Sigma_x\end{bmatrix}\right)$, $\mathbf{h}_1 = \mathbf{\Phi}_1\mathbf{x}$, and $h_2 = \mathbf{\Phi}_2\mathbf{x}$, where $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$ are matrices. We then have

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{h}_1\\ \mathbf{h}_2\end{bmatrix} \sim \mathbb{N}\left(\begin{bmatrix}\mu_y\\ \mathbf{\Phi}_1\mu_x\\ \mathbf{\Phi}_2\mu_x\end{bmatrix}, \begin{bmatrix}\Sigma_y & \Sigma_{yx}\mathbf{\Phi}_1^T & \Sigma_{yx}\mathbf{\Phi}_2^T\\ \mathbf{\Phi}_1\Sigma_{xy} & \mathbf{\Phi}_1\Sigma_x\mathbf{\Phi}_1^T & \mathbf{\Phi}_1\Sigma_x\mathbf{\Phi}_2^T\\ \mathbf{\Phi}_2\Sigma_{xy} & \mathbf{\Phi}_2\Sigma_x\mathbf{\Phi}_1^T & \mathbf{\Phi}_2\Sigma_x\mathbf{\Phi}_2^T\end{bmatrix}\right). \tag{17}$$

Substituting the covariance into (13) and (14), we can obtain $I(\mathbf{y};[\mathbf{h}_1,\mathbf{h}_2])$, $I(\mathbf{y};\mathbf{h}_1)$ and $I(\mathbf{y};\mathbf{h}_2)$. Using a similar procedure to that of the proof of Proposition 3 and the remark following it, we can prove the existence of synergistic effects and the corresponding condition for their emergence. ■

We further show that such synergistic effects will be taken into consideration by MAL in adapting the existing model. Consider a model $\mathbf{z} = \mathbf{\Phi}_1\mathbf{h}_1 + \mathbf{\Phi}_2\mathbf{h}_2$, $|\mathbf{h}_1| = |\mathbf{h}_2| = k$. We have:

$$I(\mathbf{y};\mathbf{z}) = H(\mathbf{y}) - H(\mathbf{y}|\mathbf{z}) = \frac{1}{2}\log\left[\frac{\det(\Sigma_y)}{\det(\Sigma_y - \Sigma_{yz}\Sigma_z^{-1}\Sigma_{zy})}\right]$$
$$= -\frac{1}{2}\log\left(1-\frac{v}{\Sigma_y}\right), \tag{18}$$

where $v = (\mathbf{\Phi}_1\Sigma_{1y} + \mathbf{\Phi}_2\Sigma_{2y})^T(\mathbf{\Phi}_1\Sigma_1\mathbf{\Phi}_1^T + \mathbf{\Phi}_2\Sigma_2\mathbf{\Phi}_2^T + 2\mathbf{\Phi}_1\Sigma_{12}\mathbf{\Phi}_2^T)^{-1}(\mathbf{\Phi}_1\Sigma_{1y} + \mathbf{\Phi}_2\Sigma_{2y})$, $\Sigma_i$ $(i=1,2)$ is the variance of $\mathbf{h}_i$, and $\Sigma_{ij}$ is the covariance of $\mathbf{h}_i$ and $\mathbf{h}_j$. For simplicity, we introduce a regularization term such that $\mathbf{\Phi}_1 + \mathbf{\Phi}_2 = I$. We then have:

$$\frac{\partial v}{\partial \mathbf{\Phi}_1}|_{\mathbf{\Phi}_1=I} = 2(\Sigma_{1y}-\Sigma_{2y})\Sigma_1^{-1}\Sigma_{1y} - 2\Sigma_{1y}\Sigma_1^{-1}\Sigma_{1y} + 2\Sigma_{1y}\Sigma_1^{-1}\Sigma_{12}\Sigma_1^{-1}\Sigma_{1y}. \tag{19}$$

If $\frac{\partial v}{\partial \mathbf{\Phi}_1}|_{\mathbf{\Phi}_1=I} < 0$, i.e.,

$$\Sigma_{12} < (\Sigma_{y1}\Sigma_1^{-1})^{-1}(\Sigma_{y2}\Sigma_1^{-1}\Sigma_{1y})[(\Sigma_{y1}\Sigma_1^{-1})^{-1}]^T, \tag{20}$$

then the optimal solution for $\mathbf{\Phi}_1, i.e., \arg_{\mathbf{\Phi}_1}\max I(\mathbf{y};\mathbf{z})$, will fall into the region where $\beta_2 > \mathbf{0}$. This condition is similar to that of data synergy in (16), indicating that the model adaptation in MAL will explore the synergy within the data to generate synergistic representations by updating not only the new block but also the existing blocks.

### C. Convergence of MAL

In this subsection, we show the feasibility of MAL by proving its convergence.

*Proposition 4:* For $\forall\epsilon > 0, \exists M_\epsilon > 0$ such that $M$ $(M > M_\epsilon)$ learning blocks can achieve:

$$|I(\mathbf{y};\mathbf{x}) - I(\mathbf{y};\{g_{\theta_i}(\mathbf{x}), i=1,\ldots,M\})| < \epsilon, \tag{21}$$

given that each learning block can approximate the function with $h$ data points.

*Proof:* Assuming that the existing model has $m$ learning blocks, then MAL aims to improve the performance by adding a new learning block, which is parameterized using the function $T_{\theta_{m+1}}(\mathbf{x})$. Let $\mathbf{q}$ and $\mathbf{r}$ be the functions of the existing blocks and of the new block, respectively, defined as: $\mathbf{q} = [T_{\theta_i}(\mathbf{x}), i = 1:m]$, $\mathbf{x} \sim P_x$ and $\mathbf{r} = T_{\theta_{m+1}}(\mathbf{x})$, $\mathbf{x} \sim P_x$. We then have

$$I(\mathbf{y};\{T_{\theta_i}, i=1,\ldots,m+1\}) = \mathrm{KL}(P_{yqr}||P_yP_{qr})$$
$$= \mathbb{E}_{P_{yqr}}\left[\log\frac{P_{yqr}}{P_yP_{qr}}\right] \approx \mathbb{E}_{\hat{P}_{yqr}}\left[\log\frac{P_{yqr}}{P_yP_{qr}}\right], \tag{22}$$

where $P_*$ denotes the true distribution of the random multivariate variable and $\hat{P}_*$ denotes the empirical distribution from the samples. As $\mathbf{q}$ and $\mathbf{r}$ are both generated from the empirical distribution of $\mathbf{x}$, we have the following upper bound for $I(\mathbf{y};\{T_{\theta_i}, i=1,\ldots,m+1\})$:

$$I(\mathbf{y};\{T_{\theta_i}, i=1,\ldots,m+1\}) \le I(\mathbf{y};\mathbf{x}), \tag{23}$$

which is equivalent to the following:

$$\mathbb{E}_{\hat{P}_{yqr}}\left[\log\frac{P_{yqr}}{P_yP_{qr}}\right] \le \mathbb{E}_{\hat{P}_{yx}}\left[\log\frac{P_{yx}}{P_yP_x}\right]. \tag{24}$$

The equality holds when $\frac{P_{r|yq}}{P_{r|q}} = \frac{P_{y|x}}{P_{y|x}}$. We then show that the objective function is monotonically increasing. Assuming the function $T$ is bijective, we have:

$$I(\mathbf{y};[\mathbf{q},\mathbf{r}]) = \mathrm{KL}(P_{yqr}||P_yP_{qr})$$
$$= \mathbb{E}\left[\log\frac{P_{yq}}{P_yP_q}\right] + \mathbb{E}\left[\log\frac{P_{r|yq}}{P_{r|q}}\right]$$
$$= I(\mathbf{y};\mathbf{q}) + \mathrm{KL}(P_{ry|q}||P_{r|q}P_{y|q})$$
$$\ge I(\mathbf{y};\mathbf{q}). \tag{25}$$

[5] This gain occurs because $I(\mathbf{y};[\mathbf{x}_1,\mathbf{x}_2]) - I(\mathbf{y};\mathbf{x}_1) - I(\mathbf{y};\mathbf{x}_2) = I(\mathbf{x}_2;\mathbf{y}|\mathbf{x}_1) - I(\mathbf{x}_2;\mathbf{y})$.

If each block can approximate the function with $h$ data points, then we can choose a function $\hat{R}$ that is able to approximate the $h$ data points of the empirical distribution. We then have $\mathrm{KL}(P_{ry|q}||P_{r|q}P_{y|q}) > 0$. Thus, there exists a positive value $a$ such that $\mathrm{KL}(P_{ry|q}||P_{r|q}P_{y|q}) \geq a$, i.e., the gap $|I(\mathbf{y};\mathbf{x}) - I(\mathbf{y};\{g_{\theta_i}(\mathbf{x}), i = 1,\ldots,m\})|$ will decrease by at least the value of $a$ in each iteration. If we further add $(I(\mathbf{y};\mathbf{x}) - I(\mathbf{y};\mathbf{q}) - \epsilon)/a$ blocks, the gap will be smaller than $\epsilon$. Therefore, if we set $M = m + (I(\mathbf{y};\mathbf{x}) - I(\mathbf{y};\mathbf{q}) - \epsilon)/a$, then the inequality in (21) will hold. ■

We further discuss the lower bound of the convergence rate. Let $\bar{\mathbf{y}} = \mathbf{y} - o(\mathbf{q})$, where $o(\cdot)$ is the output function as defined in (5), then we have $H(\mathbf{y}|\mathbf{q}) = H(\bar{\mathbf{y}})$. As mentioned above, $R$ fits at least $s_h$ data points with $s_h$ hidden units. Subsequently, the conditional entropy decreases by $s_h/N^m$ on average, where $N^m$ is the sample size with residual greater than 0. In each iteration, the function $R$ is learned to optimize the following objective: $\min_R H(\mathbf{y}|\mathbf{q},\mathbf{r}) = H(\mathbf{y}|\mathbf{q}) - I(\mathbf{y},\mathbf{r}|\mathbf{q})$. Therefore, the objective conditional entropy $H(\mathbf{y}|\mathbf{q},\ldots)$ decreases at least exponentially with the rate $\frac{H(\mathbf{y}|\mathbf{q},\mathbf{r})}{H(\mathbf{y}|\mathbf{q})} = 1 - s_h/N^m$.

## V. Conclusion

In this article, we investigated a fundamental yet challenging problem in machine learning: given a learning task, how can the learning approach carry out the task with the right information through the right model at the right time? To answer this question, we proposed a novel learning approach, MAL, which refines the feature selection and model learning in an iterative way. In so doing, the optimum of the objective function with respect to the given learning task and dataset is guaranteed to be attained. Then we systematically evaluated the validity of the proposed approach on two challenging domains with complex data dependencies and relationships: spatiotemporal prediction and chaotic behavioral prediction. To analyze the learning behavior of MAL, we further re-formulated the learning objective from the information-theoretic perspective to quantify the information of data features and the model capacity. Based on our information-theoretic analysis, we demonstrated that the developed objective is progressively achievable and the learning approach is able to converge.

In our future work, we will extend the current work along the following three directions. First, in the current design, we only adopt three basic structures (i.e., RNN, MLP, and CNN) as the primitive units. In fact, the design is flexible to involve more advanced structures such as the graph neural networks. In the future, we will enrich the type of basic units and explore the integration of various advanced structures into our design. Second, when more advanced structures are involved, and a large number of basic units are selected to learn complex dependencies, the training cost might be increased exponentially. Therefore, we plan to design novel learning strategies or optimization algorithms to accelerate the model training, making it practically efficient. Last but not the least, we will extend our design to more complex applications in various scenarios by taking into account domain-specific factors, so as to enable tailor-made solutions to real-world challenges.

## References


[1] H. M. Abdulwahab, S. Ajitha, and M. A. N. Saif, "Feature selection techniques in the context of Big Data: Taxonomy and analysis," *Appl. Intell.*, vol. 52, no. 12, pp. 13568–13613, 2022.

[2] A. Asseman, T. Kornuta, and A. Ozcan, "Learning beyond simulated physics," in *Proc. NIPS Workshop Model. Decis.- Mak. Spatiotemporal Domain*, 2018.

[3] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[5] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 27–66, 2012.

[6] A. Chattopadhyay, P. Hassanzadeh, and D. Subramanian, "Data-driven predictions of a multiscale lorenz 96 chaotic system using machine-learning methods: Reservoir computing, artificial neural network, and long short-term memory network," *Nonlinear Process. Geophys.*, vol. 27, no. 3, pp. 373–389, 2020.

[7] P. Chen, R. Liu, K. Aihara, and L. Chen, "Autoreservoir computing for multistep ahead prediction based on the spatiotemporal information transformation," *Nat. Commun.*, vol. 11, no. 1, 2020, Art. no. 4568.

[8] T. Chen, I. J. Goodfellow, and J. Shlens, "Net2net: Accelerating learning via knowledge transfer," in *Proc. 4th Int. Conf. Learn. Representations*, 2016.

[9] X. Chen, L. Xie, J. Wu, and Q. Tian, "Progressive differentiable architecture search: Bridging the depth gap between search and evaluation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1294–1303.

[10] Y. Chen et al., "Contrastive neural architecture search with neural architecture comparators," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9497–9506.

[11] J. Cheng, K. Huang, and Z. Zheng, "Towards better forecasting by fusing near and distant future visions," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 3593–3600.

[12] L. Deng, D. Lian, Z. Huang, and E. Chen, "Graph convolutional adversarial networks for spatiotemporal anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2416–2428, Jun. 2022.

[13] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, and S. He, "Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 890–897.

[14] C. Gallicchio, A. Micheli, and L. Pedrelli, "Design of deep echo state networks," *Neural Netw.*, vol. 108, pp. 33–47, 2018.

[15] X. He, K. Zhao, and X. Chu, "Automl: A survey of the state-of-the-art," *Knowl. Based Syst.*, vol. 212, 2021, Art. no. 106622.

[16] Z. He, C.-Y. Chow, and J.-D. Zhang, "STCNN: A spatio-temporal convolutional neural network for long-term traffic prediction," in *Proc. IEEE 20th Int. Conf. Mobile Data Manage.*, 2019, pp. 226–233.

[17] J. Y. Hesterman, L. Caucci, M. A. Kupinski, H. H. Barrett, and L. R. Furenlid, "Maximum-likelihood estimation with a contracting-grid search algorithm," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 3, pp. 1077–1084, Jun. 2010.

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

[19] K. Hornik et al., "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[20] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[21] N. P. Jewell, J. A. Lewnard, and B. L. Jewell, "Predictive mathematical models of the COVID-19 pandemic: Underlying principles and value of projections," *JAMA*, vol. 323, no. 19, pp. 1893–1894, 2020.

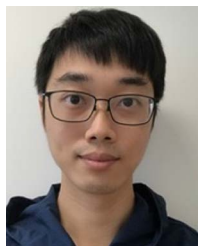
[22] K. Kandasamy, W. Neiswanger, J. Schneider, B. Póczos, and E. P. Xing, "Neural architecture search with Bayesian optimisation and optimal transport," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2020–2029.

[23] S. Kiranyaz, T. Ince, A. Iosifidis, and M. Gabbouj, "Progressive operational perceptrons," *Neurocomputing*, vol. 224, pp. 142–154, 2017.

[24] S. Kiranyaz, J. Malik, H. B. Abdallah, T. Ince, A. Iosifidis, and M. Gabbouj, "Self-organized operational neural networks with generative neurons," *Neural Netw.*, vol. 140, pp. 294–308, 2021.

[25] R. Leardi, "Genetic algorithms in feature selection," in *Genetic Algorithms in Molecular Modeling*. New York, NY, USA: Elsevier, 1996, pp. 67–86.

[26] L. Li and A. Talwalkar, "Random search and reproducibility for neural architecture search," in *Proc. 35th Uncertainty Artif. Intell.*, 2019, pp. 367–377.
[27] T. Li, J. Zhang, K. Bao, Y. Liang, Y. Li, and Y. Zheng, "AutoST: Efficient neural architecture search for spatio-temporal prediction," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 794–802.
[28] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.
[29] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for GEO-sensory time series prediction," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3428–3434.
[30] C. Liu et al., "Progressive neural architecture search," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 19–35.
[31] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.
[32] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable Architecture Search," in *Proc. 7th Int. Conf. Learn. Representations*, 2019.
[33] K. Liu, Y. Fu, P. Wang, L. Wu, R. Bo, and X. Li, "Automating feature subspace exploration via multi-agent reinforcement learning," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 207–215.
[34] Q. Ma, L. Shen, and G. W. Cottrell, "DeePr-ESN: A deep projection-encoding echo-state network," *Inf. Sci.*, vol. 511, pp. 152–171, 2020.
[35] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Gradient-based hyperparameter optimization through reversible learning," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 2113–2122.
[36] A. v. d. Oord et al., "Wavenet: A generative model for raw audio," in *Proc. 9th ISCA Speech Synthesis Workshop*, vol. 125, 2016.
[37] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.
[38] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2627–2633.
[39] G. Roffo, S. Melzi, U. Castellani, A. Vinciarelli, and M. Cristani, "Infinite feature selection: A graph-based feature filtering approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4396–4410, Dec. 2021.
[40] E. Santana, M. S. Emigh, P. Zegers, and J. C. Principe, "Exploiting spatio-temporal structure with recurrent winner-take-all networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3738–3746, Aug. 2018.
[41] S. Scardapane, D. Comminiello, A. Hussain, and A. Uncini, "Group sparse regularization for deep neural networks," *Neurocomputing*, vol. 241, pp. 81–89, 2017.
[42] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016.
[43] Q. Tan, Y. Liu, and J. Liu, "Demystifying deep learning in predictive spatiotemporal analytics: An information-theoretic framework," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3538–3552, Aug. 2021.
[44] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. Ser. B. Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.
[45] D. Tran, A. Iosifidis, J. Kanniainen, and M. Gabbouj, "Temporal attention-augmented bilinear network for financial time-series data analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1407–1418, May 2019.
[46] V. Tran, E. Brost, M. Johnston, and J. Jalkio, "Predicting the behavior of a chaotic pendulum with a variable interaction potential," *Chaos*, vol. 23, no. 3, 2013, Art. no. 033103.
[47] L. Xie and A. L. Yuille, "Genetic CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2017, pp. 1388–1397.
[48] J. Xu, X. Liu, T. Wilson, P.-N. Tan, P. Hatami, and L. Luo, "Muscat: Multi-scale spatio-temporal learning with application to climate modeling," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2912–2918.
[49] Y. Xu et al., "PC-darts: Partial channel connections for memory-efficient architecture search," in *Proc. 8th Int. Conf. Learn. Representations*, 2020.
[50] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proc. 14th Int. Conf. Mach. Learn.*, 1997, pp. 412–420.
[51] Q. Yao et al., "Taking human out of learning applications: A survey on automated machine learning," 2018, *arXiv:1810.13306*.
[52] B. Yesilyurt, "Equations of motion formulation of a pendulum containing n-point masses," 2020, *arXiv:1910.12610*.
[53] R. Yu, S. Zheng, A. Anandkumar, and Y. Yue, "Long-term forecasting using higher ordertensor RNNS," 2019, *arXiv:1711.00073*.
[54] E. A. K. Zaman, A. Mohamed, and A. Ahmad, "Feature selection for online streaming high-dimensional data: A state-of-the-art review," *Appl. Soft Comput.*, vol. 127, 2022, Art. no. 109355.
[55] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, "Predicting citywide crowd flows using deep spatio-temporal residual networks," *Artif. Intell.*, vol. 259, pp. 147–166, 2018.
[56] Y. Zhang, B. Li, Y. Tan, and S. Jian, "Evaluation ranking is more important for NAS," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2022, pp. 1–8.
[57] X. Zheng et al., "Evolving fully automated machine learning via life-long knowledge anchors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 9, pp. 3091–3107, Sep. 2021.
[58] Z. Zhong, J. Yan, W. Wu, J. Shao, and C. Liu, "Practical block-wise neural network architecture generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2423–2432.
[59] P. Zhou, N. Wang, and S. Zhao, "Online group streaming feature selection considering feature interaction," *Knowl.-Based Syst.*, vol. 226, 2021, Art. no. 107157.
[60] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.



**Qi Tan** received the Ph.D. degree from the Department of Computer Science, Hong Kong Baptist University, Hong Kong, in 2021. His research interests include spatiotemporal data analytics and complex network modeling. He has published high-quality papers in reputable journals, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, *Neurocomputing*, and *Malaria Journal.*

**Yang Liu** (Senior Member, IEEE) received the B.S. and M.S. degrees in automation from the National University of Defense Technology, Changsha, China, in 2004 and 2007, respectively and the Ph.D. degree in computing from The Hong Kong Polytechnic University, Hong Kong, in 2011. During 2011–2012, he was a Postdoctoral Research Associate with the Department of Statistics, Yale University, New Haven, CT, USA. He is currently an Assistant Professor in Computer Science with Hong Kong Baptist University, Hong Kong. His research interests include artificial intelligence, machine learning, applied mathematics, as well as their applications in high-dimensional/heterogeneous data analytics, multimedia content analysis, and computational epidemiology. He has authored and coauthored more than 80 peer-reviewed papers in reputable venues, including top-tier journals such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON AFFECTIVE COMPUTING, T-AMD, T-IST, PR, *NeuroImage*, and *Lancet Discovery Science*, as well as top-tier conferences such as AAAI, IJCAI, SIGIR, and ACMMM.

**Jiming Liu** (Fellow, IEEE) received the M.Eng. and Ph.D. degrees from McGill University (Centre for Intelligent Machines), Montreal, QC, Canada. His research interests include data analytics and machine learning, complex networks and systems, web intelligence, autonomy-oriented computing paradigm, and AI-enabled epidemiology and practice. He is the Chair Professor in Computer Science with Hong Kong Baptist University, Hong Kong. Prof. Liu was the Editor-in-Chief of *Web Intelligence Journal* (IOS), and an Associate Editor for IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *Data and Information Analytics* (AIMS), and *Computational Intelligence* (Wiley), among others. He was the Chair of IEEE Computer Society Technical Committee on Intelligent Informatics (TCII).