Front. Comput. Sci., 2023, 17(6): 176349
https://doi.org/10.1007/s11704-023-2689-5

REVIEW ARTICLE

Large sequence models for sequential decision-making: a survey

Muning WEN[1,2*], Runji LIN[3,4*], Hanjing WANG[1,2], Yaodong YANG[5], Ying WEN[1], Luo MAI[6], Jun WANG[2,7], Haifeng ZHANG[3,4], Weinan ZHANG (✉)[1]

1 School of Electronics Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200241, China
2 Digital Brain Lab, Shanghai 201306, China
3 Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
4 School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
5 Institute for Artificial Intelligence, Peking University, Beijing 100091, China
6 School of Informatics, The University of Edinburgh, Edinburgh EH8 9JU, UK
7 Department of Computer Science, University College London, London WC1E 6BT, UK

Abstract Transformer architectures have facilitated the development of large-scale and general-purpose sequence models for prediction tasks in natural language processing and computer vision, e.g., GPT-3 and Swin Transformer. Although originally designed for prediction problems, it is natural to inquire about their suitability for sequential decision-making and reinforcement learning problems, which are typically beset by long-standing issues involving sample efficiency, credit assignment, and partial observability. In recent years, sequence models, especially the Transformer, have attracted increasing interest in the RL communities, spawning numerous approaches with notable effectiveness and generalizability. This survey presents a comprehensive overview of recent works aimed at solving sequential decision-making tasks with sequence models such as the Transformer, by discussing the connection between sequential decision-making and sequence modeling, and categorizing them based on the way they utilize the Transformer. Moreover, this paper puts forth various potential avenues for future research intending to improve the effectiveness of large sequence models for sequential decision-making, encompassing theoretical foundations, network architectures, algorithms, and efficient training systems.

Keywords sequential decision-making, sequence modeling, the Transformer, training system

1 Introduction

Large sequence models, which feature a significant volume of parameters and auto-regressive data processing, have recently been instrumental in prediction tasks and (self-)supervised learning [1] in natural language processing (NLP) [2] and computer vision (CV) [3], such as ChatGPT [4] and Swin Transformer [5]. Furthermore, these models, especially the Transformer [6], have garnered substantial interest from the reinforcement learning community in the past two years, spawning numerous approaches as outlined in Section 5. In addition, large sequence models have emerged in the field of sequential decision-making and reinforcement learning (RL) [7] with notable effectiveness and generalizability, as evidenced by Gato [8] and Video Pre-Training (VPT) [9]. These methods suggest the potential for constructing a large decision model for general purposes, that is, a large sequence model that can harness a vast number of parameters to perform hundreds or more sequential decision-making tasks, analogous to the way in which large sequence models have been leveraged for NLP and CV.

This survey focuses on most of the current works that leverage (large) sequence models, mainly the Transformer, for sequential decision-making tasks, while the application of various other types of foundation models in practical decision-making contexts could be found in the report by Yang et al. [10]. We offer an in-depth investigation of the role of sequence models in sequential decision-making problems, discussing their significance and how sequence models like the Transformer are related to solving such problems. While surveying how current works utilize sequence models to facilitate sequential decision-making, we also analyze major bottlenecks toward large decision models currently with regard to model size, data and computation, and explore potential avenues for future research in algorithms and training systems to improve performance.

In the rest of this survey, Section 2 presents the formulation of prediction and sequential decision-making problems. Section 3 introduces deep reinforcement learning (DRL) as a classical solution for sequential decision-making tasks and examines three long-lasting challenges in DRL: sample efficiency problem, credit assignment problem, and partial observability problem. Section 4 establishes the connection between sequence models and sequential decision-making, highlighting the promotion of sequence modeling regarding

Received November 15, 2022; accepted May 4, 2023

E-mail: wnzhang@sjtu.edu.cn

* These authors contributed equally to this work.

the three challenges raised in Section 3. Section 5 surveys most of the current works that leverage the Transformer architecture for sequential decision-making tasks and discusses how the Transformer enhances sequential decision-making in different settings as well as the potential for building large decision models. Section 6 discusses the current progress and potential challenges regarding the system support for training large decision models. Section 7 discusses current challenges and potential research directions from the perspectives of theoretical foundation, model architectures, algorithms, and training systems. Finally, Section 8 takes conclusions of this survey with the hope for more investigation into the emerging topic of large decision models.

2 Formulation

2.1 Prediction tasks

Prediction in deep learning refers to the output of a neural network after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome, e.g., image classification in CV and translation in NLP. For a classification task in CV, given an image x, the goal is to learn the estimation of the distributions $P(y|x)$, where y is a potential label of x. It is normally solved with discriminative models like Multi-layer Perceptron (MLP) or Convolution Neural Networks (CNNs) [11−13], extracting the high-dimensional representation $c(x)$ of the input image with convolution layers and estimating the distribution $P[y|c(x)]$. For a translation task in NLP, an input sentence $\mathbf{x}$ is decomposed into a sequence with n words $\{x_1,\ldots,x_n\}$ to predict an output sentence $\mathbf{y}=\{y_1,\ldots,y_n\}$. And the estimated distribution becomes $P(\mathbf{y}|\mathbf{x})=P(y_1,\ldots,y_n|x_1,\ldots,x_n)$. Besides, other NLP tasks like text generation, predicting the next potential word with previous contents, need to estimate only the distribution of $P(y_n|x_1,\ldots,x_n)$ instead of $P(\mathbf{y}|\mathbf{x})$. Both $P(\mathbf{y}|\mathbf{x})$ and $P(y_n|x_1,\ldots,x_n)$ could be modeled with sequence models like Recurrent Neural Networks (RNNs) [14] and their variants [15,16], which use their hidden states $h_{n-1}=h(x_{n-1},h_{n-2})$ to retain previous content and estimate the distribution of $P(y_n|x_n,h_{n-1})$ recursively.

2.2 The Transformer

As the state-of-the-art sequence model, the Transformer was originally designed for NLP tasks with an encoder-decoder structure. The encoder maps a sequence of tokens to latent representations, and then the decoder generates a sequence of desired outputs in an auto-regressive manner. Besides, the encoder and decoder could also be used alone as models like Bert [17] and GPT-3 [18], which leverage the encoder and decoder architectures, respectively. One of the most essential components in Transformer is the scaled dot-product attention, which captures the interrelationships of input sequences. The attention function is written as

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V})=\text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{1}$$

where $\mathbf{Q},\mathbf{K},\mathbf{V}$ correspond to the vector of queries, keys and values, which can be learned during training, and d_k represents the dimension of $\mathbf{Q}$ and $\mathbf{K}$. Self-attentions refer to cases when $\mathbf{Q},\mathbf{K},\mathbf{V}$ share the same set of inputs. With the help of the attention mechanism, the Transformer abandons the recursive process of RNNs and estimates the distribution of $P(\mathbf{y}|\mathbf{x})$ or $P(y_n|x_1,\ldots,x_n)$ more directly, enjoying higher computation efficiency. Moreover, although the Transformer is initially designed for NLP tasks, it has the potential to be applied to CV tasks as well. By splitting an image into fixed-size patches, embedding each of them, and feeding the resulting sequence of vectors to a Transformer encoder, recent works have demonstrated remarkable performance of the Transformer in image classification tasks [19].

2.3 Sequential decision-making tasks

Unlike prediction, sequential decision-making in deep learning refers to the process by which a neural network, known as an agent, infers a sequence of actions that can be used to interact with an environment and maximize its utility. In most cases, a sequential decision-making problem is represented as a Markov decision process (MDP), $\langle\mathcal{S},\mathcal{A},r,p,\gamma\rangle$, that satisfies the Markov property [7]

$$p(s_{t+1}|s_t,a_t)=p(s_{t+1}|s_0,a_0,\ldots,s_t,a_t). \tag{2}$$

This property states that the current state of the process completely captures all the relevant information about the system's history, and thus the future is independent of the past given the current state [7]. In MDPs, $\mathcal{S}$ is the state space of the environment and $\mathcal{A}$ is the action space of agents. $r_t=r(s_t,a_t)$ is the reward function quantifying the instant utility of an agent executing an action $a_t\in\mathcal{A}$ on a specific state $s_t\in\mathcal{S}$. $p=p(s_{t+1}|s_t,a_t)$ is the transition probability of performing action a_t on state s_t at timestep t and then transiting to state s_{t+1}. γ is the factor used to calculate discounted returns

$$G_t=\sum_{k=0}^{\infty}\gamma^k r_{t+k} \tag{3}$$

that starts from timestep t. At each timestep t, an agent takes an action a_t based on the environmental state s_t. After execution, it receives an instant reward $r(s_t,a_t)$ and observes a new state s_{t+1}, whose probability distribution is $p(s_{t+1}|s_t,a_t)$. Following this process infinitely long, the agent earns a discounted return of G_t. While $r(s_t,a_t)$ is the measurement of the instant utility of agents, $\mathbb{E}[G_t|s_t]$ is the expected cumulative utility starting in s_t, which is the objective of agents learning to maximize in sequential decision-making tasks. Figure 1 has demostrated the difference between sequential decition-making tasks and prediction tasks.

3 Deep RL for sequential decision-making

As a combination of deep neural networks and RL, deep reinforcement learning (DRL) has drawn much attention and emerged as a popular paradigm for solving sequential decision-making tasks [7]. In recent years, its high potential has been demonstrated by a series of notable achievements, such as AlphaGo [20] and AlphaStar [21], which have beaten human experts at Go chess and StarCraft II.

In nearly all value-based RL methods, an agent measures the quality of an action under a specific state by learning an

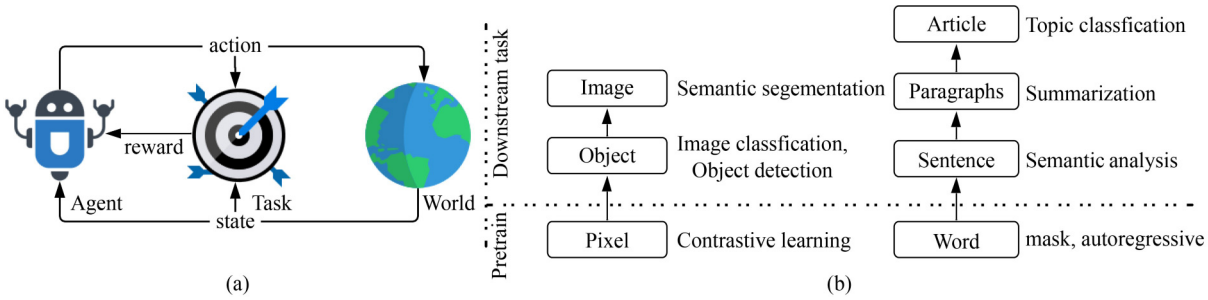


Fig. 1 The difference between sequential decision-making tasks and prediction tasks, such as CV and NLP. (a) A sequential decision-making task is a cycle of agent, task, and world, connected by interactions; (b) in prediction tasks, tasks form a hierarchical structure

action-value function $Q_\pi(s_t,a_t)$,

$$Q_\pi(s_t,a_t)=\mathbb{E}_\pi[G_t|s_t,a_t]. \tag{4}$$

Specifically, the action-value function $Q_\pi(s_t,a_t)$ approximates the expected return starting from s_t given that a_t is selected, assuming the agent follows its policy π thereafter. A fundamental property of the value function is that it satisfies a recursive relationship between the expected return from the current state and the expected return from the following state, so-called the Bellman Equation [7]:

$$Q_\pi(s_t,a_t)=r_t+\gamma\max_{a_{t+1}}Q_\pi(s_{t+1},a_{t+1}). \tag{5}$$

Through the utilization of the Bellman equation, Temporal-Difference (TD) methods [22] can learn from incomplete sequences of experience by approximating the authentic value of the current state with the sum of the observed reward and the estimated value of the subsequent state. Rather than waiting until the end of an episode as in Monte Carlo methods [23], TD methods thus update the value functions in a more efficient and incremental manner. More specifically, in TD learning, an agent updates its $Q_\pi(s_t,a_t)$ by minimizing the mean square TD error [22]:

$$\mathbb{E}_\pi[(r_t+\gamma\max_{a_{t+1}}Q_\pi(s_{t+1},a_{t+1})-Q_\pi(s_t,a_t))^2]. \tag{6}$$

In DRL, the Q function could be approximated with neural networks and trained with gradient descent. After learning an effective Q network, agents' policies that maximize $\mathbb{E}[G_t|s_t]$ can be simply obtained by

$$\pi(s_t)=\arg\max_{a_t}Q(s_t,a_t), \tag{7}$$

which is widely adopted in many value-based methods like DQN [24].

While value-based methods learn to approximate the action values and then make decisions based on the estimates, policy-based methods, also known as policy gradient methods, learn the policy π that selects actions directly without consulting a value function [7]. During training, policy gradient methods such as REINFORCE [23] optimize the policy by maximizing the expected return below through gradient ascent.

$$\mathbb{E}_\pi[\log\pi(a_t|s_t)G_t]. \tag{8}$$

Combining the value-based and policy-based methods, actor-critic methods [25] learn a state-value function $V_\pi(s_t)$ as a critic to evaluate the quality of an actor given a state s_t, i.e., the expected return commencing from s_t following the policy π:

$$V_\pi(s_t)=\mathbb{E}_\pi[G_t|s_t]. \tag{9}$$

Similar to the Q function, $V_\pi(s_t)$ also satisfies the recursive relationship between the preceding and following states,

$$V_\pi(s_t)=\mathbb{E}_\pi[r_t+\gamma V_\pi(s_{t+1})], \tag{10}$$

and thus could be optimized by minimizing the mean square TD error as well:

$$\mathbb{E}_\pi[(r_t+\gamma V_\pi(s_{t+1})-V_\pi(s_t))^2]. \tag{11}$$

While updating the critic, the actor is optimized through policy gradient with the advantage function replacing the discounted return:

$$\mathbb{E}_\pi[\log\pi(a_t|s_t)A_\pi(s_t,a_t)], \tag{12}$$

where the advantage function $A_\pi(s_t,a_t)$ measures how well the selected action is compared with the actor's average performance.

$$\begin{aligned}A_\pi(s_t,a_t)&=Q_\pi(s_t,a_t)-V_\pi(s_t)\\&=r_t+\gamma V_\pi(s_{t+1})-V_\pi(s_t).\end{aligned} \tag{13}$$

In model-based RL methods, a *model* can be employed to predict how the environment will respond to agents' actions under a given state, by estimating the MDP's dynamics, $p(s_{t+1},r_t|s_t,a_t)$ [7]. The learning process for the model resembles a supervised learning task, but with data collected through real-time interaction with the environment. Once the model is trained, it can be leveraged to generate action sequences via planning methods such as model predictive control (MPC) [26], or to generate imagined data as supplements to further enhance the value approximation or policy with direct RL, like what (deep) Dyna-Q does [7,27].

However, despite DL having scaled RL to previously intractable problems, DRL is still not as widely applied in the real world as supervised or unsupervised learning. Several existing problems involving sample efficiency, credit assignment and partial observability have prompted extensive discussions [28–30].

3.1 Sample efficiency problem

Poor data efficiency is one of the major restrictions of RL [28]. In supervised learning, training data is labeled with ground truth y so that models can learn to approximate the final distribution $P(y|x)$ of data from the beginning, which means models are fitting the same distribution during the training process. Unlike supervised learning, conventional RL optimizes agents in a trial-and-error manner [7], which means the data distribution changes according to the current policy

during the training process. Such a paradigm needs a series of loops to improve the quality of collected data and models alternately, i.e., data collection, model optimization, and data collection with optimized models. For example, at the kth training epoch, agents collect dataset $\mathcal{D}_k$ with π_k, train a world model or value network with $\mathcal{D}_k$, update the policy and get π_{k+1}, which is used to collect $\mathcal{D}_{k+1}$ for the next epoch. Since the policy π for data collection is updated continuously, the collected dataset $\mathcal{D}$ is changing as well, which means the corresponding world model or value network is approximating a new data distribution in each training epoch and so is the policy. Further, for environments with sparse rewards, the dilemma of poor sample efficiency will be more pronounced in the early training stages [31,32], since the initial random policies make it difficult to explore positive rewards and improve the quality of the dataset $\mathcal{D}$ and models. Therefore, to guarantee the stability and effectiveness of the learning process, massive interactions with environments are indispensable in each epoch to explore enough positive rewards and fully reveal the new distribution. However, these interactions can be expensive or even impossible due to safety concerns in real-world applications (e.g., autonomous driving [33], industrial scenario [34]). Moreover, even slight differences between simulators and real environments (i.e., the reality gap) can lead to the vulnerability [35] of trained RL agents, constraining the current application of RL to a certain set of tasks.

3.2 Credit assignment problem

Mostly, the consequences of an action do not manifest immediately, requiring RL algorithms to capture the cause-and-effect relationship between a sequence of decisions and resulting rewards, known as the credit assignment problem [29], whose solution is crucial for effective and efficient algorithms. While the simplest way to estimate the credit of a given state involves averaging its discounted sum of future rewards through Monte Carlo methods, such methods may suffer from high variance estimations and inefficient learning due to the randomness of trajectories [36]. To mitigate the variance, many RL approaches place more emphasis on TD methods with learned value approximation [22]. But the approximation is likely to introduce bias, which spawns TD(λ) methods to balance the bias-variance trade-off [22]. In most of the aforementioned methods, they rely solely on time as a metric of relevance: the more recent the decision, the more credit or blame it receives from a future result, which is heuristic in general and can hence be further improved by learning [7,36,37].

3.3 Partial observability problem

In many real-world environments, it is common for parts of the state information to be unavailable and needed to be inferred by combining current observations with historical or other agents' observations [30]. This loss of state information can significantly confuse agents' decisions and hinder the development of effective decision-making agents. For instance, in the case of an auto-driving car, providing only one image of a moment as the observation is insufficient to infer the speed of other vehicles, which is a crucial factor in deciding the next move. Or in multi-agent settings, each agent's observations and experiences are often partial and potentially different from those of other agents, necessitating communication between agents to estimate the complete state of the system and make decisions. This loss of full-state visibility expands the Markov decision processes (MDPs) to the partially observable Markov decision processes (POMDPs) [30] for single-agent systems and decentralized partially observable Markov Decision Processes (Dec-POMDPs) for multi-agent systems [38]. A common approach for addressing the partial observability problem is to model a sequence of observations with RNNs, expecting the missing information can be reconstructed during the training process [30]. However, information from early observations might be continuously diluted and even forgotten with the recursive function $h_n = h(x_n, h_{n-1})$ in RNNs, harming agents' performance when modeling long sequences [6].

4 Sequential decision-making as sequence modeling problems

Fortunately, the challenges mentioned in Section 3 could be addressed by treating sequential decision-making problems as sequence modeling problems and then be solved by sequence models. In order to overcome these challenges, several researchers have attempted to simplify sequential decision-making tasks by transforming them into supervised learning problems, specifically, sequence modeling problems. Imitation learning (IL), such as behavioral cloning (BC) [39] and generative adversarial imitation learning (GAIL) [40], trains agents with the supervision of expert demonstrations, integrating advances in representation learning and transfer learning, e.g., the BC-Z [41] or multi-modal interactive agent (MIA) [42]. However, the performance of IL depends heavily on high-quality expert data which is costly to obtain and conflicts with the increasing data requirements as the model size grows. Upside-down reinforcement learning (UDRL) [43] is a novel approach that transforms conventional reinforcement learning (RL) into a purely supervised learning paradigm. Compared with value-based RL, it reverses the roles of actions and returns during learning. Specifically, it employs undiscounted desired returns as network inputs, serving as commands to guide the agent's behavior. Thus, unlike conventional value-based RL, which learns a value model to evaluate the quality of each action and select the optimal one, UDRL learns to search for a sequence of actions that satisfy specific desired returns. By training the agent with pure SL on all past trajectories, UDRL circumvents the issues of sensitive discounted factors and the deadly trials arising from the combination of function approximation, bootstrapping, and off-policy training in traditional RL [7,43]. Moreover, despite classical methods still being more effective in environments with perfect Markov properties, experimental results demonstrate that UDRL surprisingly exceeds conventional baselines, such as DQN and A2C, in non-Markovian environments [43]. These results suggest that the general principles of UDRL are not restricted to Markovian environments only, indicating a promising direction for

addressing sequential decision-making in a broader context.

As a representative work, Decision Transformer (DT) [44] frames RL problems as sequence modeling problems, which enables drawing upon the simplicity and scalability of the Transformer. Based on the concept of UDRL, DT feeds a sequence of states, previous actions and desired returns to a GPT-like network and infers actions to achieve the desired returns, where the Transformer is served as a policy model. Different from DT and UDRL, Trajectory Transformer (TT) [45] maps transition sequences to shifted transition sequences entirely, incorporating states, actions and instant rewards, where the Transformer is served as a world model that captures the full dynamics of environments. Although DT is a model-free method while TT is a model-based method, both approaches share a common foundation: treating each temporal trajectory as a continuous sequence of transitions and modeling it with the Transformer. Based on this foundation, the Transformer could be used to infer future states, actions, and rewards, thus unifying many of the components that are typically required in IL, model-based RL, model-free RL, or goal-conditioned RL [45], e.g., predictive dynamics models in model-based methods, actor and critic in actor-critic (AC) algorithms [25], and behavior policy approximation in IL. Figure 2 compares the paradigms between conventional RL, IL, UDRL, DT, and TT.

4.1 Improving sample efficiency

As mentioned in Section 3.1, conventional RL in a trial-and-error manner suffers from poor sample efficiency that limits its application in the real-world environment, because of the distribution shift at each epoch and inefficient exploration in the early training stages. One of the directions to bypass this dilemma is pre-training [46−48], leveraging previous experience from offline data to pre-train a suboptimal policy in a supervised manner and then fine-tuning it for downstream tasks, which appears in multiple recent works with the Transformer [46,49,50]. In this way, the trial-and-error process could start from the near-final stages with the suboptimal policy and leave aside most of the upfront exploration and interaction, reducing the online sampling epochs. Especially for sparse reward settings, it can help skip the harrowing exploration in the early training stages. With the strong generalizability of the Transformer architecture that has been validated in many practical results [8,18,51], we can not only leverage the experience from the same task but also experience from many similar or related tasks, or even directly fine-tune the policies learned from other similar tasks [52−56]. From this perspective, the samples produced in different tasks could be stored for reuse when pre-training for new tasks, which improves the efficiency of sample utilization implicitly and largely.

4.2 Effective credit assignment

As discussed in Section 3.2, numerous conventional RL algorithms rely heavily on time as the primary metric for determining the cause-and-effect relationship between actions and rewards, which suffer from the bias-variance trade-off and require further improvement. Various works have explored better credit assignment through state association or learning additional reward functions to facilitate reward propagation over long horizons [36,57,58]. In contrast, sequence models naturally embed this property in their architecture without requiring the explicit learning of extra reward functions [44,45,49]. Furthermore, instead of assuming that recent actions receive more credit or blame, the attention mechanism can directly model the cause-and-effect relevance with undiscounted return sequences. Experiments conducted by Chen et al. [44] have confirmed that this approach is more effective than conventional TD learning algorithms.

4.3 Long horizon for partial observability

As described in Section 3.3, conventional RL methods have

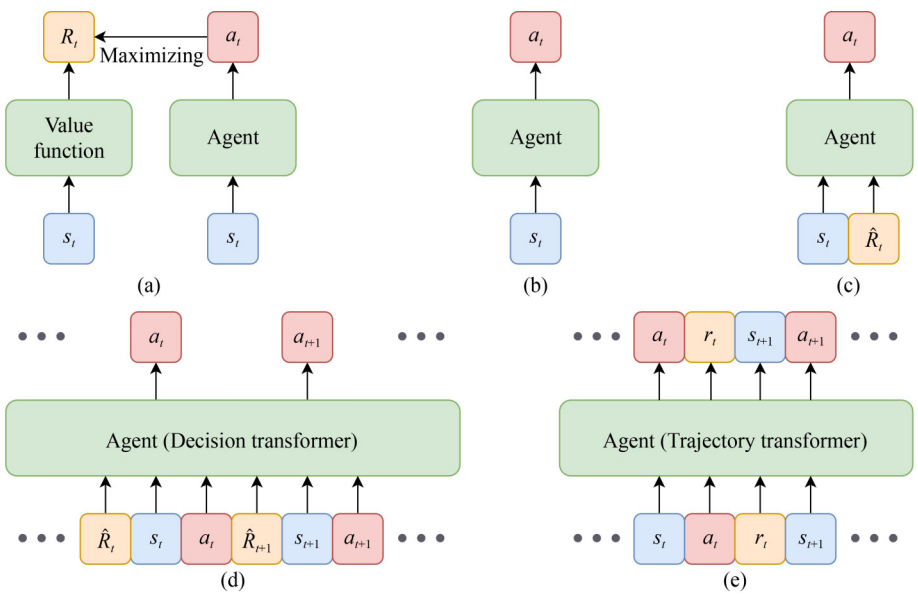


Fig. 2 Paradigm comparison of conventional RL, IL, UDRL, DT and TT. (a) is a representative method of conventional RL, where R_t indicates the estimated cumulative rewards with discount starting from s_t. (b) is a classic method in IL, i.e. Behavioral Cloning. In (c) and (d), $\hat{R}_t$ is the desired cumulative reward without discount. In (e), r_t means the instant rewards after executing a_t.

often relied on RNNs and their variants [15,16] to recover the information lost from historical or other agents' observations [30]. However, these methods still suffer from shortsightedness due to the limited capacity of hidden states, leading to the gradual dilution of early observations during recursion. For instance, given a sequence of observations $\{o_1,\ldots,o_n\}$, an RNN models a policy as $\pi(\cdot|o_n,h_{n-1})$, which prioritizes the last elements with limited capacity in the sequence and is difficult to build up long-term dependencies. Compared with RNN-based methods, transformer-based methods model the policy as $\pi(\cdot|o_1,\ldots,o_n)$, where the attention mechanism enables selective focus on specific parts of the sequence when making decisions [44,50,54]. Specifically, the impact of an early observation, such as o_1, does not have to be diluted since the impact is determined by corresponding attention weights which are continually updated during training. Ablation experiments conducted by Wen et al. [54] compare the performance of transformer-based and RNN-based policies and validate that the Transformer architectures enjoy a longer horizon than RNNs.

5 How the Transformer helps sequential decision-making

The backbone architectures in machine learning have gone through several iterations, and the family of Transformers achieves a big convergence in the era of large-scale pre-trained models. From the linear model, Gaussian mixture model (GMM) and support vector machine (SVM) in the classical ML stage, to MLP, RNN and CNN in the DL stage. With the appearance of a series of recent works listed in Table 1, the Transformer has shown tremendous potential in the field of sequential decision-making, as its variants rapidly become dominant for large-scale pre-training models in NLP [17,18], CV [5,63,64], and multi-modal domains [65−68].

In Section 5.1, we explore the empirical and theoretical advantages of the Transformer architecture as well as how it has become a popular choice in many state-of-the-art NLP or CV models. We then examine the development of the Transformer in the field of sequential decision-making, which can be divided into two parts. The first part focuses on recent works converting the reinforcement learning problem into sequential form to leverage sequence modeling for specific reinforcement learning settings, which will be surveyed in Section 5.2. The second part concentrates on leveraging diverse data to pre-train a large-scale sequence model for various downstream sequential decision-making tasks, inspired by the tremendous success of NLP and CV, which will be discussed in Section 5.3. Finally, in Section 5.4, we discuss the potential of building a large decision model and relevant characteristics that must be carefully considered.

5.1 The rise of the Transformer

Transformers have a shown substantial impact on the progress of a large variety of machine learning tasks since the efficient

Table 1 Detailed comparison between different transformer-based methods for sequential decision-making

Method	Sequence	Prediction	Discretized tokens	Benefit	Notes
UPDeT [52]	s	a	No	Multi-task; few-shot learning; interpretability	Model-free; online; multi-agent
PIT [53]	s	Q values	No	Multi-task; few-shot learning; credit assignment	Model-free; online; multi-agent
DT [44]	rtg-s-a	a	No	Long sequence; POMDP; credit assignment	Model-free; offline
TT [45]	s-a-r(-rtg)	s-a-r	Yes	Long sequence; POMDP; sparse-reward	Model-based; offline
GDT [59]	$\psi(s,a)$-s-a	a	No	HIM problems	Model-free; offline
PDT [46]	s-a	a	No	Few-shot learning	Model-free; pre-train
MADT [50]	s-a	a	No	Multi-task; long sequence	Model-free; offline; multi-agent
ODT [49]	rtg-s-a	a	No	Few-shot learning	Model-free; online
MAT [54]	s	a	No	Monotonic improvement; multi-task; few-shot learning	Model-free; online; multi-agent
MGDT [55]	s-a-r-rtg	a-r-rtg	Yes	Multi-task; few-shot learning	Model-free; offline
TrMRL [60]	s	a	No	Multi-task; few-shot learning	Model-free; online; meta-learning
PG-AR [61]	s	a	No	Monotonic improvement	Model-free; online; multi-agent
Prompt-DT [56]	rtg-s-a	a	No	Multi-task; few-shot learning	Model-free; offline
BooT [62]	s-a-r-rtg	s-a-r-rtg	Yes	Data augmentation	Model-based; offline

expansion of model size helps harness massive amounts of data. Scale is a significant ingredient in achieving excellent results. Therefore, the model size is growing faster than ever before: benefiting from the Transformer architecture, large language models have scaled up from 340 million [17] to 1.6 trillion parameters [69] in a few years. As a result, the Transformers have outperformed previous standard networks (CNN and RNN) on numerous benchmarks and become general choices in the state-of-the-art model [5,70], e.g., image classification, semantic segmentation, text classification, text generation, question answering, image caption, etc. [68]. Despite there being some attempts to build large pre-trained models based on CNN [70], Transformer architectures still take the dominant position in the field of large models. From empirical results and theoretical perspectives, Transformers have advantages in high parallelization [17,18], scalability [18,63,71], and appropriate inductive bias [72,73]. In general, the advantages of Transformers supported by empirical evidence and theoretical analysis are summarized as follows.

Scaling law The existence of the scaling law in Transformer architecture indicates that the loss scales as a power-law with model size, the amount of data, and the training computation [71,74]. There are several detailed and adequate experiments showing that the capacity of Transformer architectures increases smoothly following power law and the bigger models are more sample efficient in a series of tasks, such as ViT-G in CV benchmarks [63] and GPT in NLP benchmarks [71]. This finding has encouraged researchers to scale up their models to pursue higher performance.

Higher throughput In the domain of sequence modeling, Transformer architectures exhibit superior throughput compared to RNNs, which possess inherent sequentiality: each hidden state is dependent on the previous hidden state. This fundamental characteristic limits their ability to be parallelized across multiple GPUs, resulting in a considerable slowdown during training [14]. For instance, supposing a sequence with a length of n, a recurrent layer has to execute n operations sequentially to backpropagate gradients for one training epoch. In contrast, the Transformer architecture offers more computational efficiency and parallelizability by avoiding sequential computation over time. Instead, it performs self-attention operations across the entire sequence at once, reducing the number of operations required for gradient backpropagation [6]. This property helps Transformer-based methods to be trained at larger scale scenarios with acceptable computing budgets.

Long-term interaction modeling ability In terms of long sequence inputs, MLP suffers from the linear increase of the input layer dimension, vanilla CNN is limited by the local convolution kernel, and RNN is limited by an exponential decay of mutual information in the temporal distance [75], which leads to difficulty in accurately modeling interactions between the long-spanning pairs. However, the attention mechanism enables the Transformer to efficiently handle very long sequences [18], which is discussed in Section 4.3 as well.

More stable training process RNN frequently suffers from vanishing and exploding gradient problems [76]. On the contrary, Transformers are more robust in training. Researchers [71] observe the insensitivity of Transformers to some architectural hyper-parameters, which is vital for the training of large models considering the expensive training cost of conducting a hyper-parameter search.

Efficient inductive bias Edelman et al. [73] reveal that the inductive bias of self-attention is a creation of sparse variables to capture features of the input sequences. Olsson et al. [72,77] demonstrate that the Transformer not only memorizes data patterns but also tries to conduct abstract reasoning. Researchers also have provided theoretical analysis for the features of the Transformer, e.g., the inductive bias, sample complexity, and the generalization bound of the attention mechanism [78]. The others focus on measuring the model expressivity of the Transformer under the framework of universal function approximation and Turing completeness [67,79].

5.2 RL with the Transformer

Due to the noticeable effectiveness of DT and TT, many Transformer-based variants have recently emerged for sequential decision-making tasks, spanning from offline RL, model-based RL, meta RL, multi-agent RL, and goal-conditioned RL to agent architecture in the general RL setting. RL is suitable for the sequence modeling method, as a sequence of transition (trajectory) data includes information like environment states, actions decided by agents, and how the action affects the world, i.e., transition dynamics to the next stage, and task-specific rewards to measure the performance of behaviors. The major differences among these methods are listed in Table 1, such as the components in the sequence, how to process the sequence elements, benefits from sequence modeling, and specific reinforcement learning settings.

5.2.1 Offline RL

Offline reinforcement learning [80] focuses on leveraging static datasets collected by behavior policy in various qualities without further interaction to train a better policy or evaluate it [81]. The sequence model provides a new perspective to tackle offline RL problems at the trajectory level. Because of the high similarity of the approach to using offline datasets with prediction tasks, this is the first area where sequence models are applied in RL. Decision Transformer (DT) [44] adopts the reward condition from UDRL to boost the performance of the policy, and models a sequence of return-to-go, states, and actions. After supervised learning on offline data, DT demonstrates strong generalization to decode the better action when conditioned on an appropriately high return-to-go. However, it lacks a guiding principle to find an appropriately high return-to-go to achieve expert performance. To alleviate this issue, Multi-Game Decision Transformer (MGDT) introduces an expert classifier to conduct discriminator-guided generation for expert action. Trajectory Transformer (TT) [45] learns a world model to predict the future trajectory from offline data and chooses the desired action by planning through beam search during execution. Extended from TT, Bootstrapped Transformer (BooT) [62] boosts the sequence model training process with bootstrapping data argumentation. Although offline RL has advantages in data efficiency,

sometimes an online fine-tuning process is necessary to achieve further performance improvements after offline learning. However, DT is conservative due to the supervising manner, which impedes the exploration of the online process. For this purpose, the Online Decision Transformer (ODT) [49] appends DT with hindsight return relabeling and entropy terms to encourage exploration.

5.2.2 Model-based RL

Model-based RL [82] utilizes historical data to build a world model to improve data efficiency and conduct safe planning. Sequence models use a historical sequence to predict the future and thus effectively reduce cumulative error. TransDreamer [83] and Dreamer with Transformers [84] inherit the learning framework from Dreamer [85], a notable MBRL algorithm, and simply change the backbone network architecture for agents and world models from RNN to Transformer. Benefiting from long-range modeling capability in Transformer, TransDreamer significantly surpasses Dreamer in benchmarks requiring complex memory. Compared with learning a latent state representation and assuming that state distribution follows a prior distribution in TransDreamer, TT discretizes the continuous state and action into a sequence of discrete tokens, which represents a fixed width or quantile range of the original continuous space. Therefore, TT outputs an arbitrary probabilistic distribution of the next token conditioned on the historic discrete token sequence, significantly reducing the dynamic prediction error.

5.2.3 Meta RL

Meta RL aims to train on diverse tasks to allow agents to adapt to new tasks quickly without much interaction in the environment. Pre-trained Decision Transformer (PDT) [46] combines DT with semi-supervised learning to reduce the demand for labeled data through pre-training on massive unlabeled data, in which reward is regarded as the label in RL. Multi-Game Decision Transformer [55] has no special design for meta RL pre-training, but simply uses a mixed dataset including several Atari trajectories with diverse performance. However, MGDT demonstrates rapid adaptation ability in out-of-distribution tasks with 1% data to finetune. Prompting Decision Transformer (Prompt-DT) [56] leverages a prompting framework to enable rapid adaptation in offline RL, in which segments of task-specific demonstration are concatenated with input to guide agents to understand the new task. TrMRL (the Transformer for Meta Reinforcement Learning) [60] employs a Transformer architecture to create an episodic memory to contextualize the policy, which is called the memory reinstatement mechanism. Generalized Decision Transformer (GDT) [59] proposes a unified framework for hindsight information matching and a bi-directional DT which performs well in an offline one-shot imitation learning setting.

5.2.4 Multi-agent RL

Multi-agent RL is proposed for the interactive scenario with several smart agents. Sequence models in multi-agent RL generally treat agents as a sequence, rather than a transition trajectory. Therefore, the interactions among agents can be captured by sequence modeling, which brings extra benefits, such as a monotonic improvement guarantee. Based on the DT, multi-agent decision transformer (MADT) [50] extends it into multi-agent systems by directly applying the same architecture to independent agents with shared parameters. While Fu et al. [61] analyze the monotonic improvement property of auto-regressive policies in conventional multi-agent RL methods and propose the Auto-Regressive Policy Gradient (PG-AR) paradigm, multi-agent transformer (MAT) [54], which is inspired by the Advantage Decomposition Theorem, incorporates the entire Transformer architecture and auto-regressive decision process into online multi-agent RL algorithms for monotonic improvement of joint policies and achieves state-of-the-art performance.

5.2.5 Goal-conditioned RL

Goal-condition RL [86−88] learns a general policy function to finish a series of simple tasks, for instance, to reach different goal states. TT can also be used in goal-conditioned RL by conditioning the goal state tokens in the planning process. Despite most of the sequence models in RL being GPT-style auto-regressive models, FlexiBiT [89] uses a BERT-style bi-directional Transformer as backbone architecture to model the entire trajectory. Instead of predicting the next token from history, FlexiBiT is trained to predict some masked tokens given other tokens as context. Therefore, FlexiBiT is competent in goal-conditioned RL because it can predict the next action conditioned on the goal state by masking the intermediate sub-sequence. FlexiBiT provides a unified way to treat distinct RL tasks as different mask schemes, such as behavior cloning, offline RL, inverse dynamics, waypoint conditioning, and goal-conditioning. However, the current performance of the masked model is not satisfactory enough in general. Text-Conditioned Decision [90] trains an agent to follow the instructions with the goal to take action.

5.2.6 Agent architecture

Since the attention mechanism has some unique advantages, for instance, flexible input length and permutation invariance, the agents' backbone architecture based on Transformer enhances performance, which easily plugs into any conventional RL methods. Universal Policy Decoupling Transformer (UPDeT) [52] leverages the Transformer architecture to fit tasks with different observation and action configuration requirements. Population Invariant agent with Transformer (PIT) [53] utilizes the Transformer architecture to achieve coordination transfer in universal scenarios.

5.3 Scalable pre-trained decision models

The huge amount of multi-modal interaction data on the Internet could be used to train a general model, helping agents understand their tasks and make various decisions according to humans' instructions in real-world applications [51]. While detailed comparisons between these Transformer-based sequence modeling methods are shown in Table 2.

5.3.1 Pre-training for sequential decision-making

The essential differences between prediction and sequential decision-making problems make the current success of large

Table 2 Detailed comparison between different pre-trined decision models, with abbreviations: Language model (LM), language and vision model (LVM)

Methods	Knowledge domain	Downstream task indicator	What to pre-train	How to pre-train	How to use pre-trained model
Xland [91]	Online tasks	Predicates	Policy	RL	Zero-shot; finetune
MIA [42]	Offline human demo	Text	Policy	BC	Zero-shot; finetune
Gato [8]	Offline expert demo; multi-modal data	Prompt	Policy	BC	Zero-shot; finetune
SayCan [92]	Pre-trained LM	Text	Perception	SL; RL	Zero-shot
Minedojo [51]	Internet video; Pre-trained LVM;	Text	Reward	SL	Online RL
VPT [9]	Internet video; manual annotation	–	Policy; world model	BC	Finetune
LM-Nav [93]	Pre-trained LVM; pre-trained LM	Text	Perception	SL	Search method
Inner Mono. [94]	Pre-trained LM; pre-trained VM	Text	Perception	SL; BC	Zero-shot

sequence models in NLP or CV cannot be directly transferred to the latter. Because the sequential decision-making process involves a feedback loop, subtle changes in behavior would lead to severe data distribution shifts. Therefore, new algorithms are demanded to learn stable representation, mitigate distribution shifts, and improve data efficiency.

We cannot expect that pre-training a single model would lead to strong generalization ability in all out-of-distribution tasks. Therefore, how to learn a universal and consistent representation for all the downstream tasks and minimize the distance between the training data distribution and the evaluation data distribution are the major issues that remain unsolved for effective large decision models with a reliable theoretical guarantee.

In general, representation learning and how to deal with distribution shifts are significant in pre-training, thus attracting interest from both CV and NLP. Self-supervised learning contributes profoundly to the development of large models in representation learning. NLP adopts the masking mechanism [17] and auto-regressive process [18], while CV develops contrastive learning, such as SimCLR [95] and MoCo [96]. These methods help not only the utilization of unlabeled data but also produce a stable, informative, and consistent representation of data to speed up the downstream tasks. Prompting [56], a recently proposed training paradigm in NLP is challenging the typical pre-train and fine-tune paradigm. Briefly, prompting methods transfer downstream tasks into some prompt templates. In essence, prompts convert the evaluation distribution into the training distribution, therefore being a promising solution for the zero-shot setting.

However, this issue is more challenging in the sequential decision-making domain. First, as Fig. 1 shows, the relationship between pre-trained tasks and downstream tasks is hierarchical in prediction problems, while it is cyclic in sequential decision-making problems. This difference makes learning what kind of representation and how to organize the downstream tasks remains an open problem. Second, since the decision made by the agent would affect the world, the sequential decision-making problems suffer from severe distribution shifts, which impede generalization [97]. This problem is abstracted as the auto-induced distributional shift [98], which means the output of a system causes a change in input data. Although researchers provided a theoretical analysis framework of the distribution shift from the difference between behavior policy and training policy [99], we should consider more factors, such as the different world dynamics and task objectives in downstream tasks.

5.3.2 Data collection for pre-training

Data, model size, and computing are the three main performance bottlenecks, according to empirical findings [71]. There are two potential research topics for expanding the available data in the sequential decision-making domain, while model size and computation are discussed in Section 5.1 and Section 6, respectively.

The first focuses on creating a procedural framework to generate a wide spectrum of tasks and scenarios in simulators to eliminate bottlenecks from a limited number of human-designed tasks [91]. Massively diverse and flexible tasks provide essential knowledge to develop skills in logical reasoning, understanding, planning, and memory to solve new complex sequential decision-making problems. However, since all the training tasks are generated in the same format, how to eliminate the gap between training tasks and downstream tasks remains an open problem. Proposing efficient algorithms to transfer the skills from simulation to the real world [100] for large model settings or constructing realistic but scalable simulators to reflect the real world as much as possible [101] are promising directions.

The second way focuses on leveraging diverse, large but static datasets without further interaction to train a sequential decision-making system, termed offline reinforcement learning [80]. This paradigm greatly extends the boundaries of applications, as interaction with the environment is infeasible, expensive, and unsafe in most real-world tasks, e.g., automatic driving, recommendation systems, and robotics. Although offline algorithms have the aforementioned advantages, offline RL faces a series of challenges, including smoothly transitioning from offline pre-training to online fine-tuning to achieve better performance [102,103], hyper-parameter sensitivity, and a lack of an efficient evaluation method to search for better hyper-parameters and examine policy robustness.

5.3.3 Recent advances in scaling pre-trained decision models As shown in Table 2, there are several attempts in pre-trained decision models trying to answer the aforementioned questions. Researchers utilize diverse datasets from distinct knowledge domains, including online interactions with environments or offline demonstrations, to pre-train different components in decision systems. These methods are characterized by what kind of component in decision systems is pre-trained and how to use the pre-trained model.

Pre-training for policy A policy with ideal initialization can mitigate the exploration problem better than learning from scratch, which is verified in many scenarios, e.g., Go and Starcraft. Therefore, pre-training a policy on enough diverse and massive experience data can improve data efficiency in the new downstream tasks. In an important attempt, Gato [8] pre-trains a single large model on multi-modal data to master hundreds of tasks, including sequential decision-making tasks, image captions, chitchat, etc. By simple imitation learning on expert demonstrations, Gato successfully pushes the model parameter scale in the sequential decision-making domain to the billion level. Also, it avoids some primary challenges in sequential decision-making, such as learning ability with suboptimal offline data and high data efficiency in the online fine-tuning process. VPT pre-trains a single sequence model to imitate human player behavior from massive YouTube videos. The pre-trained model served as a general behavioral prior, showing zero-shot capabilities and making exploration easier and more efficient in fine-tuning. MGDT focuses more on policy transferring or few-shot adaptation across multiple tasks with the strong generalizability of Transformer architectures. Crucially, Gato, VPT, and MGDT all show scaling law in the RL field, indicating the pursuit of large decision models is promising.

Pre-training for reward function The reward function plays a key role in sequential decision-making systems, and defines the target of tasks or the preferences over a series of different policies. On account of the importance of the reward function, pre-training a reward function for downstream tasks helps RL work on completely new tasks without human design. Minedojo [51] pre-trains a large language and vision model to approximate reward functions and guide the online reinforcement learning to generalize into unseen task instructions.

Pre-training for world model In a specific setting, a world model simulates the environment that agents interact with, which is a reusable component shared by a series of tasks. Although TT provides a promising tool to train a world in a sequential manner for robotics or other similar scenarios, to the best of our knowledge, there is no study to fill this gap. However, an inverse world model has been introduced to increase the diversity and quantity of offline data. The inverse world model in VPT [9] expands the sources of data from laboratories to large-scale realistic internet information produced by humans. Specifically, VPT collects a relatively small dataset with game videos played by volunteers labeled with action sequences and a large set of videos without action labels from YouTube. Then an inverse dynamic world model is trained on the small dataset to label massive YouTube videos, and human demonstrations with action labels are used to pre-train a policy.

Pre-trained multi-modal perception model for sequential decision-making To attain agents with general skills, basic common sense is indispensable, such as the ability to recognize objects from pictures, understand semantics from text, and decompose a task into steps [104]. Multi-modal algorithms [92−94] improve data efficiency by transferring knowledge from off-the-shelf pre-trained large sequence models in the language or vision domains, rather than cultivating basic ability in a trial-and-error manner. SayCan combines a value function and a pre-trained language model to control a real robot, following task instructions [92]. Specifically, the value function figures out what action can be completed, while the pre-trained language model figures out whether this action is appreciated to achieve the task goal. Huang et al. [105] decompose a complex task into several simple goals, speeding up the learning process of downstream tasks and showing strong generalization. Inner Monologue [94] implements a closed-loop feedback control system for robotics based on a pre-trained language model and a collection of perception models, thinking of completing the entire task as a conversation. Perception models provide scene information. An agent driven by the language model decides what to do next and inquires for human feedback to give the correct response, and the human describes the tasks and interacts with agents. It is observed that all participants in a conversation give information in language form. LM-Nav [93] achieves impressive performance in open real-world robotic navigation tasks, powered by large pre-trained models of language, vision, and action. The language model is responsible for converting navigation commands into a series of landmarks, and the vision-and-language model grounds the landmarks in the topological map. The knowledge from the pre-trained model significantly eliminates the bottleneck caused by limited language-annotated robot data.

5.4 The next step: large decision models

Gato [8] and VPT [9] have shown the potential of building large decision models for general purposes in the field of sequential decision-making, like what large sequence models have done for NLP and CV tasks. However, to build a large decision model, some modifications in architecture are significant with increasing data and model size, while naively scaling up models might fail as the number of parameters increases. That is, with the same volume of data and parameters, the network architecture can be the determining factor to improve the performance of large decision models. In this section, some important characteristics are listed since they can serve as consultative principles when designing network architecture for large decision models in the future. Noticed that the Transformer and its variants are suggested to be promising candidates recently, but any other model architectures [106−108] meeting the requirements below are still worth an exploration.

5.4.1 Multi-task

To take full advantage of high-capacity models, how to utilize

data from diverse tasks is critical for generalization. Some techniques in model architecture have been investigated, e.g., transfer learning can be accomplished with the mixture of experts (MoE) [109] and modularization [110,111]. Related research can help large models in the sequential decision-making domain attain better general intelligence.

5.4.2 Sparse activation

When decision models are scaled to extreme sizes, a computing request involving the full set of parameters can be incredibly expensive and inefficient. However, for dense models, each piece of inference or training data activates the entire set of model parameters, resulting in high training and inference costs and latency. Therefore, the parse activation [5,112] methods are proposed to balance model size and performance. For each piece of data, only a subset of the parameters is activated to process the input in a sparse model during training and testing. Even if sparse models usually have more parameters when compared to their equal-quality dense alternatives [113], their training or inference cost and latency are significantly reduced.

5.4.3 Multi-modality

Data in different modalities provides information from distinct perspectives. Model architectures supporting multi-modality are capable of a broader range of applications and more complicated interactions. Vision endows agents with the ability to observe, make a reasonable response [91], and form general knowledge about the shape of objects [9,93]. Natural language instructs agents and deepens their understanding of the new tasks [105], divides the high-level goal into detailed steps [51,93,94] and provides a natural interface for agents to cooperate with humans [42].

6 Training systems

In this section, we discuss the systems that can support the training of large decision models based on sequence models. Sequence models, especially with Transformer architectures, have achieved substantial improvements in accuracy and generalizability by scaling up, usually following scaling laws [63,71,74].

Model and data scaling Although based on highly different test suits and tasks (image classification [63], language translation [74]), prior research reports that Transformers exhibit highly predictable scaling patterns in many of these tasks. As proposed, the test loss of the model when saturating (given enough training data) follows a general form:

$$\hat{\mathcal{L}}(N_c) = \alpha(N_c/N)^{-\beta} + \mathcal{L}_\infty, \tag{14}$$

where α, β, L_∞ are fitted parameters depending on tasks and data. N_c is the number of non-embedding parameters. N is a fixed normalization term for N_c and L_∞ represents the irreducible part of the loss in scaling due to data noise. Besides the model saturating law revealed by Eq. (14), empirical results [71] show that models have better data efficiency and can benefit from a larger dataset when scaling up. In the language translation tasks [71], the number of training data that saturates models when scaling can be fitted by a sub-linear power-law (dataset volume $D \sim N^{0.74}$). To get the most out of scaling, the sizes of the model and data are required to be expanded simultaneously, imposing new challenges to the design of efficient training systems due to issues such as massive memory and computational budget demanded.

Scaling decision models While there exists a fruitful line of work on scaling behaviors of vision and language Transformers, that of large decision-making Transformers is still under-explored. Even though both are trained in a supervised manner, recent researches [8,42] on large decision models report different scaling patterns, not to mention their reinforced counterparts, which often have a stronger dependency on resource-demanding online data generation. Efficient training of large decision models is not a trivial problem, and the lack of a handy training toolkit and systems for large decision models is holding back more research forces from entering this area.

6.1 Existing challenges

6.1.1 Hybrid parallelism

Gigantic sequence models can contain trillions of parameters. These parameters consume tremendous amounts of memory and must be distributed to multiple devices. The distributed execution is usually achieved through a hybrid parallelism scheme that combines data parallelism, model parallelism, and pipeline parallelism, as shown in Fig. 3. To train large sequence models, training systems must have effective ways to optimize hybrid parallelism schemes and distribute computation to multiple devices.

6.1.2 Large datasets and massive environments

Sequence models need to be pre-trained using offline datasets and fine-tuned (i.e., few-shot learning) for downstream tasks by interacting with environment simulators online, as shown in Fig. 4. The pre-training datasets can be as large as 500 billion tokens [18]. The parallel execution of environment simulators is also challenging. These simulators need to be parallelized using thousands of CPUs (and even GPUs), thus producing a sufficient workload for fine-tuning sequence models.

6.2 Hybrid parallelism systems

For training large Transformer architectures, many hybrid parallelism systems have been proposed. For example, GPipe [114] introduces pipeline parallelism for the Transformer, DeepSpeed partitions the states of the optimizer to reduce communication overhead, and Colossal-AI [115] attempts to automatically parallelize the training of gigantic neural networks. Though promising, these systems fail to fully support large decision models for several reasons.

6.2.1 Lack of designs for synchronizing model checkpoints

Existing hybrid parallelism systems are often designed for offline training scenarios where models are trained for a long time, and model checkpoints are deployed into inference servers only once. In the scenarios of training decision models, the models need to be continuously checkpointed and repetitively deployed to the inference servers (i.e., model synchronization); otherwise, the inference servers will have

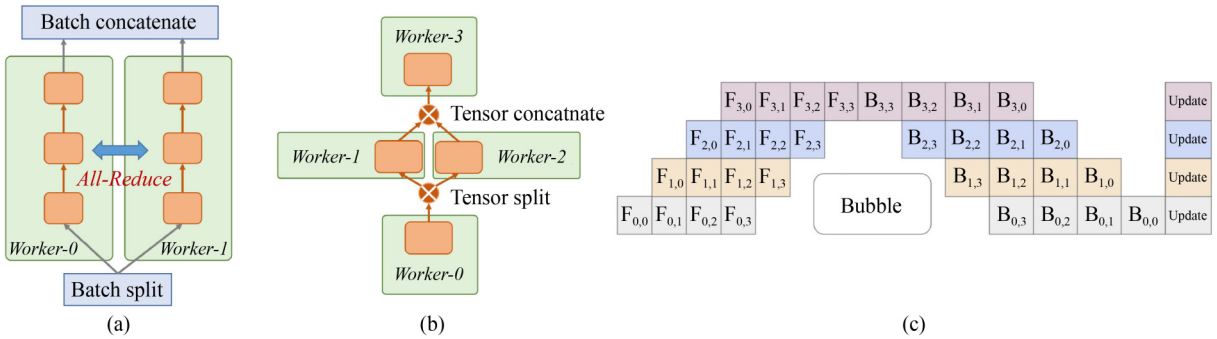


Fig. 3 (a) shows a data-paralleled three-layer model with a parallel size of 2. Data Parallelism (DP) creates replicas of the entire model across the cluster, with each device holding one (or more) of these replicas. (b) illustrates the same three-layer model being assigned to 4 physical devices under Model Parallelism (MP), with a layer-wise (vertical) slicing schema and a horizontal slicing scheme on the second layer (the 2nd layer being internally sliced and assigned to worker-1 and worker-2). MP splits the model either horizontally (inside a layer, where Tensor Parallelism is often involved since parameters like weights are sliced, e.g., split matrix multiplication into operations into sub-matrices) or vertically (layer-level slice). (c) GPipe [114]: A 4-layer model assigned to 4 physical devices (the vertical axis) with a parallel parallelism schema. Parallel Parallelism (PP) combines DP and MP by slicing the model vertically into chunks, mapping them to different devices, and splitting the mini-batch input into micro-batches fed into the pipeline sequentially to reduce bubbles (device under-utilized periods). **Hybrid Parallelism**: Though PP has already been a hybrid of DP and MP, it can be further integrated with DP inside a parallel schema by serving multiple homogeneous pipelines (parameters can differ depending on the synchronization schema), orchestrated as a hybrid parallelism schema. A hybrid parallelism schema is often a combination of DP, MP and PP to have fine-grained placement and execution plans based on diverse IO, memory, and computing characteristics of different parallelism methods with an overall optimization goal of efficiency

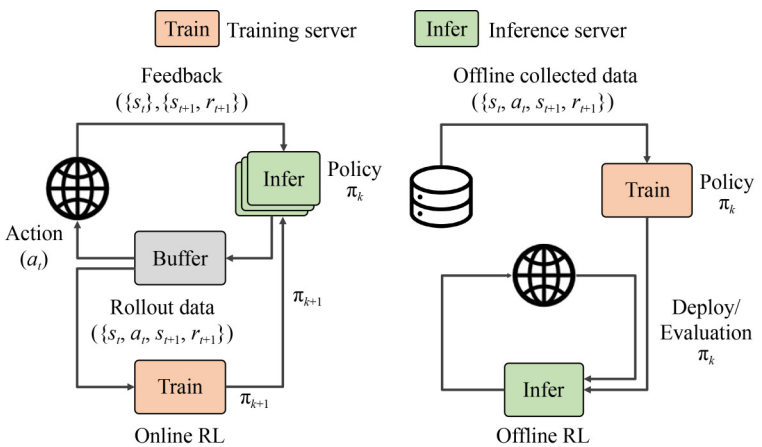


Fig. 4 The data-flow comparison between the paradigms of offline RL and online RL, where offline pre-training relies on large datasets and online fine-tuning requires parallelizing massive environments to accelerate online interaction and data collection. Moreover, the online fine-tuning phase imposes more communication pressure due to strict parameter synchronization requirements between inference and training servers

stale models that offer sub-optimal performance when interacting with environment simulators. As large decision models can have trillions of parameters, continuously checkpointing and deploying large model checkpoints can incur severe network bottlenecks, making the training of large decision models prohibitively expensive.

6.2.2 Lack of designs for handling simulation environments

Existing hybrid parallelism systems are originally designed for processing training datasets stored on disks, and they are ill-suited to processing simulation environments that continuously produce training samples in memory. Different from on-disk training datasets, simulation environments can return highly complex nested observations, and those observations are produced at dynamic rates (i.e., observations are produced at a different rate at the beginning of training because initial decision models often offer insufficient performance). Integrating these environments into existing offline-oriented hybrid parallelism systems requires non-trivial research and implementation efforts.

6.3 Distributed RL systems

Distinct from building hybrid parallelism systems, practitioners have also made parallel efforts in designing distributed RL systems. Ray allows multiple RL tasks (e.g., simulating environments or training RL models) to be dynamically dispatched to CPUs and GPUs. Impala [116] adopts an Actor-Learner architecture where actors (consisting of an inference model and an environment simulator) produce trajectories in parallel, and learners replicate RL models on multiple GPUs. Seed-RL [117] further speeds up actors by allowing GPUs to be effectively used in model inference.

However, there are non-trivial challenges for existing variants of distributed RL systems to accommodate the training tasks of large decision models. We observe several reasons for this problem.

6.3.1 Lack of end-to-end performance optimization

Training a large decision model requires a complex pipeline. Specifically, the model requires (1) processing large training datasets first, (2) interaction with environments, and (3) using hybrid parallelism to partition large model states finally. This pipeline requires various techniques to optimize hardware performance: there are techniques for using GPUs to speed up dataset processing and environment simulation or parallelize the computation of large tensors. These days, all these techniques are applied in an isolated manner, and they are not coordinated in existing RL systems. Such systems thus lack end-to-end performance optimization, leaving underlying hardware resources inefficiently utilized.

6.3.2 Lack of automatic resource management

An enabling scenario for large decision models is multi-task pre-training. These tasks need to be driven by different environments, and the pre-trained models can adopt a mixture-of-expert architecture. These days, users must *manually* allocate GPU resources to different environments, and further reserve GPUs for pre-trained models. This manual resource allocation is, however, tedious and often sub-optimal, and we anticipate future distributed RL systems to realize fully automatic resource management, making them capable of supporting large-scale multi-task pre-training.

7 Discussion and future prospects

7.1 Theoretical foundation

Although converting RL problems into sequence modeling problems has yielded numerous benefits recently, it has also resulted in a loss of theoretical guarantees for policy optimization, in contrast to traditional RL approaches. While satisfactory performance has been achieved in some experiments, this superiority is heavily dependent on the generalization ability of network architectures, data quality, and specific problem scenarios. For instance, DT-type algorithms might experience significant degradation in environments with high randomness destabilizing the desired returns. The lack of effective theoretical analysis and guarantees for policy optimization constrains further improvement of decision models. Therefore, it is highly meaningful to research the organic integration of sequence modeling methods with traditional RL methods that offer theoretical guarantees in the future.

7.2 Network architectures

In terms of network architecture, most of the RL methods directly rely on vanilla Transformers from NLP and CV without customized design, leaving ample room for performance improvement. Developing customized Transformer architectures primarily involves defining sequences, designing tokens, targeted attention calculation, and employing MoE layers. Consequently, promising research directions include integrating RL-specific semantics into the token design, combining attention masks with the Markov properties, and allocating MoE specifically to sequential decision-making tasks. Additionally, in-context learning is an important feature of large language models that often require lengthy sequences to emerge. Leveraging the Markov properties of sequential decisions to reduce computation complexity from quadratic to linear is a highly valuable research problem with the potential to facilitate in-context learning. Lastly, the recent surge of diffusion models yields novel implications for modeling decision sequences.

7.3 Algorithms

Despite recent advancements, many RL algorithms with sequence models remain domain-specific. Therefore, a unified framework capable of encompassing various RL scenarios is an area of future research that requires attention. Notably, GPT and multi-modal BeITv3 [118] have demonstrated a trend toward unifying upstream and downstream tasks and achieved remarkable results. Although UniMask [119] is trying to unify upstream and downstream tasks in RL, it still falls short in performance. Thus, the development of a unified modeling approach in sequential decision-making domains will continue to be a critical issue.

In the context of large-scale pre-training, effectively incorporating multi-modal knowledge of vision and language into sequential decision-making is of utmost importance. While semantic common-sense information plays a critical role in enhancing the efficiency and effectiveness of sequential decision-making for general purposes, ChatGPT [4] achieves a remarkable breakthrough as a powerful knowledge base. However, the integration between sequential decision-making and perception modalities still lacks naturalness. For example, LM-Nav and SayCan manually design the fusion mechanism of multiple outputs from large perception models, but fail to perform joint training. While Gato performs joint training of multiple modalities, it lacks alignment between modalities in terms of tasks. It would be interesting to explore the possibility of learning an extra module to splice cross-modal large models together, such as an adapter or an inverse dynamic model. Furthermore, the emergence of in-context learning and chain-of-thought abilities [120] in large language models may be the ingredients for creating general self-improving agents without the need for supervised knowledge from humans.

7.4 Efficient training systems

Training Efficiency represents a significant impediment that hinders the development of large decision models, which requires additional efforts and extensive exploration in future research aimed at designing efficient training systems.

7.4.1 Requirements for offline pre-training

Although the offline pre-training of decision models shares similarities in data flow and control flow with language and vision transformers, data loading might impede the scalability of the former. The offline datasets for large-scale models may exceed the capacity of host memory or even the hard drives of a compute node, necessitating to be served over networks. In NLP and CV tasks, training datasets can be shuffled prior to the training phase and sequentially read during the epochs, and thus they can be efficiently cached and accelerated due to underlying space locality. However, decision models' performance and stability depend on the data distribution, consequently requiring runtime sampling. The random access

behavior of sampling might cause a high miss rate for vanilla caching policies and poor performance in data loading. Therefore, future researches for training systems with efficient data placement and caching are crucial for pre-training large decision models.

7.4.2 Requirements for online training

Large decision models impose unique workload characteristics in online training due to the RL paradigm. During the online learning phase, since these models periodically interact with the environment and collect mini-batches of training data, these models have to switch repeatedly between inference and training modes. As a result, while more researchers and industries have been utilizing high-end GPUs shipped with large GPU memory to accommodate model parameters, their computing resources are often underutilized in distributed training.

Modern GPUs are designed for parallel and batch execution, whereas most existing environments are CPU-oriented and executed sequentially. Although the overall environmental throughput can be greatly extended in multi-core systems, a throughput gap may still exist between many CPU-served environments and GPU-served models. Therefore, extra abstraction layers and implementation are required to efficiently parallelize and distribute CPU environments.

Besides, since GPU-served models and CPU-served environments have bi-directional dependencies on each other, their overall performance should be optimized from a systematic view. For example, larger batches may lead to high peak utilization of devices, while the latency from batching, environment scheduling, and network communication can result in a poor average utilization rate of devices. Moreover, frequent communication and synchronization for mode coordination can be expensive in large-scale training. The parameter server, a common component for the asynchronous training paradigm, can easily become a bottleneck for massive parameters and large clusters. Therefore, joint efforts in training system design and algorithms are indispensable to address these issues.

8 Conclusions

In this survey, we explored the current progress of leveraging the sequence modeling methods for sequential decision-making tasks. Tackling sequential decision-making problems via sequence modeling can be a promising solution to address those long-lasting issues in conventional RL methods, involving sample efficiency, credit assignment, and partial observability. Besides, sequence models can bridge the gap between RL and offline self-supervised learning in terms of data efficiency and transferability.

We conclude that model architecture for large decision models should be designed with the awareness of support for multi-modality, multi-task transferability, and sparse activation, while the algorithms should address the concerns about both the quality and quantity of data. And the overall training efficiency should be systematically optimized via parallelism. Following a series of discussions about the theoretical foundation, network architecture, algorithm design and training system support, this survey provides potential research directions toward building a large decision model. We hope this survey could inspire more investigation into this trending topic and ultimately empower more real-world applications, e.g., robotics, automatic vehicles, and the automated industry.

Acknowledgements The SJTU team was partially supported by "New Generation of AI 2030" Major Project (2018AAA0100900), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and the National Natural Science Foundation of China (Grant No. 62076161). Muning Wen is supported by Wu Wen Jun Honorary Scholarship, AI Institute, Shanghai Jiao Tong University.

Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

References

1. Liu X, Zhang F, Hou Z, Mian L, Wang Z, Zhang J, Tang J. Self-supervised learning: generative or contrastive. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(1): 857–876
2. Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems. 2014, 3104−3112
3. Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. Communications of the ACM, 2017, 60(6): 84–90
4. Qin C, Zhang A, Zhang Z, Chen J, Yasunaga M, Yang D. Is ChatGPT a general-purpose natural language processing task solver? 2023, arXiv preprint arXiv: 2302.06476
5. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B. Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021, 9992−10002
6. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017, 6000−6010
7. Sutton R S, Barto A G. Reinforcement Learning: An Introduction. 2nd ed. Cambridge: MIT Press, 2018
8. Reed S, Zolna K, Parisotto E, Colmenarejo S G, Novikov A, Barth-Maron G, Gimenez M, Sulsky Y, Kay J, Springenberg J T, Eccles T, Bruce J, Razavi A, Edwards A, Heess N, Chen Y, Hadsell R, Vinyals O, Bordbar M, de Freitas N. A generalist agent. 2022, arXiv preprint arXiv: 2205.06175
9. Baker B, Akkaya I, Zhokhov P, Huizinga J, Tang J, Ecoffet A, Houghton B, Sampedro R, Clune J. Video PreTraining (VPT): learning to act by watching unlabeled online videos. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
10. Yang S, Nachum O, Du Y, Wei J, Abbeel P, Schuurmans D. Foundation models for decision making: problems, methods, and opportunities. 2023, arXiv preprint arXiv: 2303.04129
11. Kruse R, Mostaghim S, Borgelt C, Braune C, Steinbrecher M. Multi-layer perceptrons. In: Kruse R, Mostaghim S, Borgelt C, Braune C, Steinbrecher M, eds. Computational Intelligence: A Methodological Introduction. 3rd ed. Cham: Springer, 2022, 53−124
12. LeCun Y, Boser B, Denker J S, Henderson D, Howard R E, Hubbard W, Jackel L D. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1989, 1(4): 541–551
13. Sarker I H. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Computer Science, 2021, 2(6): 420
14. Goodfellow I, Bengio Y, Courville A. Deep Learning. Cambridge: MIT Press, 2016

15. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780
16. Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing. 2014, 1724−1734
17. Devlin J, Chang M W, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 2019, 4171−4186
18. Brown T B, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler D M, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D. Language models are few-shot learners. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 1877−1901
19. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N. An image is worth 16x16 words: transformers for image recognition at scale. In: Proceedings of the 9th International Conference on Learning Representations. 2021
20. Silver D, Huang A, Maddison C J, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D. Mastering the game of go with deep neural networks and tree search. Nature, 2016, 529(7587): 484–489
21. Vinyals O, Babuschkin I, Czarnecki W M, Mathieu M, Dudzik A, Chung J, Choi D H, Powell R, Ewalds T, Georgiev P, Oh J, Horgan D, Kroiss M, Danihelka I, Huang A, Sifre L, Cai T, Agapiou J P, Jaderberg M, Vezhnevets A S, Leblond R, Pohlen T, Dalibard V, Budden D, Sulsky Y, Molloy J, Paine T L, Gulcehre C, Wang Z Y, Pfaff T, Wu Y H, Ring R, Yogatama D, Wünsch D, Mckinney K, Smith O, Schaul T, Lillicrap T, Kavukcuoglu K, Hassabis D, Apps C, Silver D. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature, 2019, 575(7782): 350–354
22. Sutton R S. Learning to predict by the methods of temporal differences. Machine Learning, 1988, 3(1): 9–44
23. Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 1992, 8(3): 229–256
24. Mnih V, Kavukcuoglu K, Silver D, Rusu A A, Veness J, Bellemare M G, Graves A, Riedmiller M, Fidjeland A K, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D. Human-level control through deep reinforcement learning. Nature, 2015, 518(7540): 529–533
25. Konda V R, Tsitsiklis J N. Actor-critic algorithms. In: Proceedings of the 13th Conference on Neural Information Processing Systems. 1999
26. Camacho E F, Alba C B. Model Predictive Control. Advanced Textbooks in Control and Signal Processing. Springer London, 2013
27. Peng B, Li X, Gao J, Liu J, Wong K F, Su S Y. Deep Dyna-Q: integrating planning for task-completion dialogue policy learning. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. 2018, 2182−2192
28. Botvinick M, Ritter S, Wang J X, Kurth-Nelson Z, Blundell C, Hassabis D. Reinforcement learning, fast and slow. Trends in Cognitive Sciences, 2019, 23(5): 408–422
29. Sutton R S. Temporal credit assignment in reinforcement learning. University of Massachusetts Amherst, Dissertation, 1984
30. Hausknecht M J, Stone P. Deep recurrent q-learning for partially observable MDPs. In: Proceedings of 2015 AAAI Fall Symposium Series. 2015, 29−37
31. McFarlane R. A survey of exploration strategies in reinforcement learning. McGill University, 2018
32. Hao J, Yang T, Tang H, Bai C, Liu J, Meng Z, Liu P, Wang Z. Exploration in deep reinforcement learning: from single-agent to multiagent domain. 2021, arXiv preprint arXiv: 2109.06668
33. Zhou M, Luo J, Villella J, Yang Y, Rusu D, Miao J, Zhang W, Alban M, Fadakar I, Chen Z, Huang A C, Wen Y, Hassanzadeh K, Graves D, Chen D, Zhu Z, Nguyen N, Elsayed M, Shao K, Ahilan S, Zhang B, Wu J, Fu Z, Rezaee K, Yadmellat P, Rohani M, Nieves N P, Ni Y, Banijamali S, Rivers A C, Tian Z, Palenicek D, bou Ammar H, Zhang H, Liu W, Hao J, Wang J. SMARTS: scalable multi-agent reinforcement learning training school for autonomous driving. In: Proceedings of the Conference on Robot Learning. 2020
34. Qin R J, Zhang X, Gao S, Chen X H, Li Z, Zhang W, Yu Y. NeoRL: a near real-world benchmark for offline reinforcement learning. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
35. Jakobi N, Husbands P, Harvey I. Noise and the reality gap: the use of simulation in evolutionary robotics. In: Proceedings of the 3rd European Conference on Artificial Life. 1995, 704−720
36. Harutyunyan A, Dabney W, Mesnard T, Heess N, Azar M G, Piot B, van Hasselt H, Singh S, Wayne G, Precup D, Munos R. Hindsight credit assignment. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 1120
37. Schulman J, Moritz P, Levine S, Jordan M, Abbeel P. High-dimensional continuous control using generalized advantage estimation. 2015, arXiv preprintarXiv: 1506.02438
38. Oliehoek F A, Amato C. A Concise Introduction to Decentralized POMDPs. Cham: Springer, 2016
39. Torabi F, Warnell G, Stone P. Behavioral cloning from observation. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. 2018, 4950−4957
40. Ho J, Ermon S. Generative adversarial imitation learning. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 4572−4580
41. Jang E, Irpan A, Khansari M, Kappler D, Ebert F, Lynch C, Levine S, Finn C. BC-Z: zero-shot task generalization with robotic imitation learning. In: Proceedings of the Conference on Robot Learning. 2021, 991−1002
42. Interactive Agents Team. Creating multimodal interactive agents with imitation and self-supervised learning. 2021, arXiv preprint arXiv: 2112.03763
43. Srivastava R K, Shyam P, Mutz F, Jaśkowski W, Schmidhuber J. Training agents using upside-down reinforcement learning. 2019, arXiv preprint arXiv: 1912, 0287, 7
44. Chen L, Lu K, Rajeswaran A, Lee K, Grover A, Laskin M, Abbeel P, Srinivas A, Mordatch I. Decision transformer: reinforcement learning via sequence modeling. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 15084−15097
45. Janner M, Li Q, Levine S. Offline reinforcement learning as one big sequence modeling problem. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 1273−1286
46. Cang C, Hakhamaneshi K, Rudes R, Mordatch I, Rajeswaran A, Abbeel P, Laskin M. Semi-supervised offline reinforcement learning with pre-trained decision transformers. In: Proceedings of the International Conference on Learning Representations. 2022
47. Wang Z, Chen C, Dong D. Lifelong incremental reinforcement learning with online Bayesian inference. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(8): 4003–4016
48. Wang Z, Chen C, Dong D. A dirichlet process mixture of robust task

models for scalable lifelong reinforcement learning. IEEE Transactions on Cybernetics, 2022, doi: 10.1109/TCYB.2022.3170485

49. Zheng Q, Zhang A, Grover A. Online decision transformer. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 27042−27059
50. Meng L, Wen M, Yang Y, Le C, Li X, Zhang W, Wen Y, Zhang H, Wang J, Xu B. Offline pre-trained multi-agent decision transformer: one big sequence model tackles all SMAC tasks. 2021, arXiv preprint arXiv: 2112.02845
51. Fan L, Wang G, Jiang Y, Mandlekar A, Yang Y, Zhu H, Tang A, Huang D A, Zhu Y, Anandkumar A. MINEDOJO: building open-ended embodied agents with internet-scale knowledge. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
52. Hu S, Zhu F, Chang X, Liang X. UPDeT: universal multi-agent reinforcement learning via policy decoupling with transformers. 2021, arXiv preprint arXiv: 2101.08001
53. Zhou T, Zhang F, Shao K, Li K, Huang W, Luo J, Wang W, Yang Y, Mao H, Wang B, Li D, Liu W, Hao J. Cooperative multi-agent transfer learning with level-adaptive credit assignment. 2021, arXiv preprint arXiv: 2106.00517
54. Wen M, Kuba J G, Lin R, Zhang W, Wen Y, Wang J, Yang Y. Multi-agent reinforcement learning is a sequence modeling problem. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022, 16509−16521
55. Lee K H, Nachum O, Yang M, Lee L, Freeman D, Xu W, Guadarrama S, Fischer I, Jang E, Michalewski H, Mordatch I. Multi-game decision transformers. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
56. Xu M, Shen Y, Zhang S, Lu Y, Zhao D, Tenenbaum J B, Gan C. Prompting decision transformer for few-shot policy generalization. In: Proceedings of the International Conference on Machine Learning. 2022, 24631−24645
57. Ferret J, Marinier R, Geist M, Pietquin O. Selfattentional credit assignment for transfer in reinforcement learning. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence. 2021, 368
58. Mesnard T, Weber T, Viola F, Thakoor S, Saade A, Harutyunyan A, Dabney W, Stepleton T S, Heess N, Guez A, Moulines E, Hutter M, Buesing L, Munos R. Counterfactual credit assignment in model-free reinforcement learning. In: Proceedings of the 38th International Conference on Machine Learning. 2021, 7654−7664
59. Furuta H, Matsuo Y, Gu S S. Generalized decision transformer for offline hindsight information matching. In: Proceedings of the 10th International Conference on Learning Representations. 2022
60. Melo L C. Transformers are meta-reinforcement learners. In: Proceedings of the International Conference on Machine Learning. 2022, 15340−15359
61. Fu W, Yu C, Xu Z, Yang J, Wu Y. Revisiting some common practices in cooperative multi-agent reinforcement learning. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 6863−6877
62. Wang K, Zhao H, Luo X, Ren K, Zhang W, Li D. Bootstrapped transformer for offline reinforcement learning. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
63. Zhai X, Kolesnikov A, Houlsby N, Beyer L. Scaling vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, 1204−1213
64. Goyal P, Caron M, Lefaudeux B, Xu M, Wang P, Pai V, Singh M, Liptchinsky V, Misra I, Joulin A, Bojanowski P. Self-supervised pretraining of visual features in the wild. 2021, arXiv preprint arXiv: 2103.01988
65. Radford A, Kim J W, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, Krueger G, Sutskever I. Learning transferable visual models from natural language supervision. In: Proceedings of the 38th International Conference on Machine Learning. 2021, 8748−8763
66. Ramesh A, Pavlov M, Goh G, Gray S, Voss C, Radford A, Chen M, Sutskever I. Zero-shot text-to-image generation. In: Proceedings of the 38th International Conference on Machine Learning. 2021, 8821−8831
67. Dehghani M, Gouws S, Vinyals O, Uszkoreit J, Kaiser Ł. Universal transformers. In: Proceedings of the 7th International Conference on Learning Representations. 2019
68. Wang W, Bao H, Dong L, Bjorck J, Peng Z, Liu Q, Aggarwal K, Mohammed O K, Singhal S, Som S, Wei F. Image as a foreign language: BEiT pretraining for all vision and vision-language tasks. 2022, arXiv preprint arXiv: 2208.10442
69. Fedus W, Zoph B, Shazeer N. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. 2021, arXiv preprint arXiv: 2101.03961
70. Kolesnikov A, Beyer L, Zhai X, Puigcerver J, Yung J, Gelly S, Houlsby N. Big transfer (BiT): general visual representation learning. In: Proceedings of the 16th European Conference on Computer Vision. 2020, 491−507
71. Kaplan J, McCandlish S, Henighan T, Brown T B, Chess B, Child R, Gray S, Radford A, Wu J, Amodei D. Scaling laws for neural language models. 2020, arXiv preprint arXiv: 2001.08361
72. Kharitonov E, Chaabouni R. What they do when in doubt: a study of inductive biases in seq2seq learners.2020, arXiv preprint arXiv: 2006.14953
73. Edelman B L, Goel S, Kakade S, Zhang C. Inductive biases and variable creation in self-attention mechanisms. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 5793−5831
74. Ghorbani B, Firat O, Freitag M, Bapna A, Krikun M, Garcia X, Chelba C, Cherry C. Scaling laws for neural machine translation. In: Proceedings of the 10th International Conference on Learning Representations. 2022
75. Shen H. Mutual information scaling and expressive power of sequence models. 2019, arXiv preprint arXiv: 1905.04271
76. Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. In: Proceedings of the 30th International Conference on Machine Learning. 2013, 1310−1318
77. Olsson C, Elhage N, Nanda N, Joseph N, DasSarma N, Henighan T, Mann B, Askell A, Bai Y, Chen A, Conerly T, Drain D, Ganguli D, Hatfield-Dodds Z, Hernandez D, Johnston S, Jones A, Kernion J, Lovitt L, Ndousse K, Amodei D, Brown T, Clark J, Kaplan J, McCandlish S, Olah C. In-context learning and induction heads. 2022, arXiv preprint arXiv:2209.11895
78. Wei C, Chen Y, Ma T. Statistically meaningful approximation: a case study on approximating turing machines with transformers. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022
79. Pérez J, Marinković J, Barceló P. On the Turing completeness of modern neural network architectures. In: Proceedings of the 7th International Conference on Learning Representations. 2019
80. Levine S, Kumar A, Tucker G, Fu J. Offline reinforcement learning: tutorial, review, and perspectives on open problems. 2020, arXiv preprint arXiv: 2005.01643
81. Li L. A perspective on off-policy evaluation in reinforcement learning. Frontiers of Computer Science, 2019, 13(5): 911–912
82. Moerland T M, Broekens J, Plaat A, Jonker C M. Model-based reinforcement learning: a survey. Foundations and Trends® in Machine Learning, 2023, 16(1): 1–118
83. Chen C, Wu Y F, Yoon J, Ahn S. TransDreamer: reinforcement

learning with transformer world models. 2022, arXiv preprint arXiv: 2202.09481

84. Zeng C, Docter J, Amini A, Gilitschenski I, Hasani R, Rus D. Dreaming with transformers. In: Proceedings of the AAAI Workshop on Reinforcement Learning in Games. 2022
85. Hafner D, Lillicrap T P, Ba J, Norouzi M. Dream to control: learning behaviors by latent imagination. In: Proceedings of the 8th International Conference on Learning Representations. 2020
86. Kaelbling L P. Learning to achieve goals. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence. 1993, 1094−1099
87. Rudner T G J, Pong V H, McAllister R, Gal Y, Levine S. Outcome-driven reinforcement learning via variational inference. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 13045−13058
88. Liu M, Zhu M, Zhang W. Goal-conditioned reinforcement learning: problems and solutions. In: Proceedings of the 31st International Joint Conference on Artificial Intelligence. 2022, 5502−5511
89. Carroll M, Lin J, Paradise O, Georgescu R, Sun M, Bignell D, Milani S, Hofmann K, Hausknecht M, Dragan A, Devlin S. Towards flexible inference in sequential decision problems via bidirectional transformers. 2022, arXiv preprint arXiv: 2204.13326
90. Putterman A L, Lu K, Mordatch I, Abbeel P. Pretraining for language-conditioned imitation with transformers. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021
91. Open Ended Learning Team, Stooke A, Mahajan A, Barros C, Deck C, Bauer J, Sygnowski J, Trebacz M, Jaderberg M, Mathieu M, McAleese N, Bradley-Schmieg N, Wong N, Porcel N, Raileanu R, Hughes-Fitt S, Dalibard V, Czarnecki W M. Open-ended learning leads to generally capable agents. 2021, arXiv preprint arXiv: 2107.12808
92. Ahn M, Brohan A, Brown N, Chebotar Y, Cortes O, David B, Finn C, Fu C, Gopalakrishnan K, Hausman K, Herzog A, Ho D, Hsu J, Ibarz J, Ichter B, Irpan A, Jang E, Ruano R J, Jeffrey K, Jesmonth S, Joshi N J, Julian R, Kalashnikov D, Kuang Y, Lee K H, Levine S, Lu Y, Luu L, Parada C, Pastor P, Quiambao J, Rao K, Rettinghouse J, Reyes D, Sermanet P, Sievers N, Tan C, Toshev A, Vanhoucke V, Xia F, Xiao T, Xu P, Xu S, Yan M, Zeng A. Do as I can, not as I say: grounding language in robotic affordances. 2022, arXiv preprint arXiv: 2204.01691
93. Shah D, Osiński B, Ichter B, Levine S. LM-Nav: robotic navigation with large pre-trained models of language, vision, and action. In: Proceedings of the 6th Conference on Robot Learning. 2023, 492−504
94. Huang W, Xia F, Xiao T, Chan H, Liang J, Florence P, Zeng A, Tompson J, Mordatch I, Chebotar Y, Sermanet P, Jackson T, Brown N, Luu L, Levine S, Hausman K, Ichter B. Inner monologue: embodied reasoning through planning with language models. In: Proceedings of the Conference on Robot Learning. 2022, 1769−1782
95. Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 149
96. He K, Fan H, Wu Y, Xie S, Girshick R. Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, 9726−9735
97. Levine S. Understanding the world through action. In: Proceedings of the 5th Conference on Robot Learning. 2022, 1752−1757
98. Krueger D, Maharaj T, Leike J. Hidden incentives for auto-induced distributional shift. 2020, arXiv preprint arXiv: 2009.09153
99. Kumar A, Fu J, Tucker G, Levine S. Stabilizing off-policy Q-learning via bootstrapping error reduction. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 1055
100. Kaspar M, Osorio J D M, Bock J. Sim2Real transfer for reinforcement learning without dynamics randomization. In: Proceedings of 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020, 4383−4388
101. Tancik M, Casser V, Yan X, Pradhan S, Mildenhall B P, Srinivasan P, Barron J T, Kretzschmar H. Block-NeRF: scalable large scene neural view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, 8238−8248
102. Nair A, Gupta A, Dalal M, Levine S. AWAC: accelerating online reinforcement learning with offline datasets. 2020, arXiv preprint arXiv: 2006.09359
103. Mao Y, Wang C, Wang B, Zhang C. MOORe: model-based offline-to-online reinforcement learning. 2022, arXiv preprint arXiv: 2201.10070
104. Zhou Z H. Rehearsal: learning from prediction to decision. Frontiers of Computer Science, 2022, 16(4): 164352
105. Huang W, Abbeel P, Pathak D, Mordatch I. Language models as zero-shot planners: extracting actionable knowledge for embodied agents. In: Proceedings of the International Conference on Machine Learning. 2022, 9118−9147
106. Bai S, Kolter J Z, Koltun V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. 2018, arXiv preprint arXiv: 1803.01271
107. Tolstikhin I, Houlsby N, Kolesnikov A, Beyer L, Zhai X, Unterthiner T, Yung J, Steiner A, Keysers D, Uszkoreit J, Lucic M, Dosovitskiy A. MLP-mixer: an all-MLP architecture for vision. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 24261−24272
108. Jaegle A, Borgeaud S, Alayrac J B, Doersch C, Ionescu C, Ding D, Koppula S, Zoran D, Brock A, Shelhamer E, Hénaff O J, Botvinick M M, Zisserman A, Vinyals O, Carreira J. Perceiver IO: a general architecture for structured inputs & outputs. In: Proceedings of the 10th International Conference on Learning Representations. 2022
109. Shazeer N, Mirhoseini A, Maziarz K, Davis A, Le Q V, Hinton G E, Dean J. Outrageously large neural networks: the sparsely-gated mixture-of-experts layer. In: Proceedings of the 5th International Conference on Learning Representations. 2017
110. Yang R, Xu H, Wu Y, Wang X. Multi-task reinforcement learning with soft modularization. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 400
111. Fernando C, Banarse D, Blundell C, Zwols Y, Ha D, Rusu A A, Pritzel A, Wierstra D. PathNet: evolution channels gradient descent in super neural networks. 2017, arXiv preprint arXiv: 1701.08734
112. Lepikhin D, Lee H, Xu Y, Chen D, Firat O, Huang Y, Krikun M, Shazeer N, Chen Z. GShard: scaling giant models with conditional computation and automatic sharding. In: Proceedings of the 9th International Conference on Learning Representations. 2021
113. Rajbhandari S, Li C, Yao Z, Zhang M, Aminabadi R Y, Awan A A, Rasley J, He Y. DeepSpeed-MoE: advancing mixture-of-experts inference and training to power next-generation AI scale. In: Proceedings of the International Conference on Machine Learning. 2022, 18332−18346
114. Huang Y, Cheng Y, Bapna A, Firat O, Chen M X, Chen D, Lee H, Ngiam J, Le Q V, Wu Y, Chen Z F. GPipe: efficient training of giant neural networks using pipeline parallelism. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 10
115. Li S, Fang J, Bian Z, Liu H, Liu Y, Huang H, Wang B, You Y. Colossal-AI: a unified deep learning system for large-scale parallel training. 2021, arXiv preprint arXiv: 2110.14883
116. Espeholt L, Soyer H, Munos R, Simonyan K, Mnih V, Ward T, Doron Y, Firoiu V, Harley T, Dunning I, Legg S, Kavukcuoglu K. IMPALA: scalable distributed deep-RL with importance weighted actor-learner

architectures. In: Proceedings of the 35th International Conference on Machine Learning. 2018, 1406−1415

117. Espeholt L, Marinier R, Stanczyk P, Wang K, Michalski M. SEED RL: scalable and efficient deep-RL with accelerated central inference. In: Proceedings of the 8th International Conference on Learning Representations. 2020

118. Ozbulak U, Lee H J, Boga B, Anzaku E T, Park H, Van Messem A, De Neve W, Vankerschaver J. Know your self-supervised learning: A survey on image-based generative and discriminative training. 2023, arXiv preprint arXiv: 2305.13689

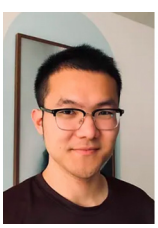
119. Carroll M, Paradise O, Lin J, Georgescu R, Sun M, Bignell D, Milani S, Hofmann K, Hausknecht M, Dragan A, Devlin S. UniMASK: unified inference in sequential decision problems. 2022, arXiv preprint arXiv: 2211.10869

120. Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E H, Le Q V, Zhou D. Chain-of-thought prompting elicits reasoning in large language models. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022

Muning Wen is currently working toward his PhD degree at Shanghai Jiao Tong University, China. His research interests include reinforcement learning and multi-agent system. He has been serving as a reviewer at NeurIPS.

Runji Lin is currently pursuing his MSc degree at the School of Artificial Intelligence, University of Chinese Academy of Sciences, China. His research interests include reinforcement learning, multi-agent system, and game theory.

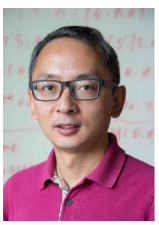
Hanjing Wang is currently a PhD Candidate of Shanghai Jiao Tong University, China. His research interests include scalable reinforcement learning and machine learning systems.



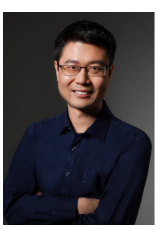
Yaodong Yang is currently an assistant professor at Peking University, China. His research is about reinforcement learning and multi-agent systems. He has maintained a track record of more than forty publications at top conferences and journals, along with the best system paper award at CoRL 2020 and the best blue-sky paper award at AAMAS 2021. Before joining Peking University, he was an assistant professor at King's College London. Before KCL, he was a principal research scientist at Huawei UK.

Ying Wen is a tenure-track assistant professor at John Hopcroft Center for Computer Science at Shanghai Jiao Tong University, China. His research interests include machine learning, multi-agent systems and human-centered interactive systems, etc. He has been serving as a PC member at ICML, NeurIPS, ICLR, AAAI, IJCAI, ICAPS and a reviewer at TIFS, Operational Research, etc. He was granted Best Paper Award in AAMAS 2021 Blue Sky Track and Best System Paper Award in CoRL 2020.

Luo Mai is an assistant professor (UK Lecturer) in the School of Informatics at the University of Edinburgh, UK. He is a member of the Institute of Computing Systems Architecture where he is leading the Edinburgh System-X Group. His research group designs scalable, adaptive and efficient system software to support emerging data-centric applications and utilize novel computing platforms.



Jun Wang is a chair professor of Computer Science at University College London, UK, and the founding director of MSc Web Science and Big Data Analytics. His main research interests are in the areas of AI and intelligent systems, including (multi-agent) reinforcement learning, deep generative models, and their diverse applications on information retrieval, recommender systems and personalization, data mining, smart cities, bot planning, computational advertising etc. He has served as an Area Chair in ACM CIKM and ACM SIGIR.



Haifeng Zhang is an associate professor at the Institute of Automation, Chinese Academy of Sciences (CASIA), China. His research areas include reinforcement learning, game AI, game theory and computational advertising. He has published research papers on international conferences ICML, NeurIPS, AAAI, IJCAI, AAMAS etc. He has served as a reviewer for AAAI, IJCAI, TNNLS, Acta Automatica Sinica, and co-chair for IJCAI competition, IJTCS, DAI Workshop, etc.

Weinan Zhang is now an associate professor at Shanghai Jiao Tong University, China. His research interests include reinforcement learning, deep learning and data science with various real-world applications. He has published over 150 research papers on international conferences and journals and has been serving as an area chair or (senior) PC member at ICML, NeurIPS, ICLR, KDD, AAAI, IJCAI, SIGIR, etc., and a reviewer at JMLR, TOIS, TKDE, TIST, etc.