计算机应用研究
Application Research of Computers
ISSN 1001-3695,CN 51-1196/TP

《计算机应用研究》网络首发论文

题目：基于端到端深度强化学习求解有能力约束的车辆路径问题
作者：葛斌，田文智，夏晨星，秦望博
DOI：10.19734/j.issn.1001-3695.2024.03.0101
收稿日期：2024-03-21
网络首发日期：2024-08-07
引用格式：葛斌，田文智，夏晨星，秦望博．基于端到端深度强化学习求解有能力约束的车辆路径问题[J/OL]．计算机应用研究.
https://doi.org/10.19734/j.issn.1001-3695.2024.03.0101

基于端到端深度强化学习求解有能力约束的车辆路径问题 *

葛 斌[1]，田文智[1]，夏晨星[1,2]，秦望博[1]

(1. 安徽理工大学计算机科学与工程学院，安徽 淮南 232001; 2. 合肥综合性国家科学中心能源研究院，合肥 230031)

摘 要：有能力约束的车辆路径问题(Capacitated Vehicle Routing Problem，CVRP)是现阶段供应链应用最常见的问题模型，现多采用启发式算法求解，但随着问题规模增大，启发式算法求解速度慢且无法保证解的质量。提出端到端深度强化学习(Deep Reinforcement Learning，DRL)网络框架对 CVRP 问题进行研究。首先利用边聚合图注意力网络编码器(Edge-Graph Attention Network Encoder，EGATE)对车辆路径规划问题的图表示进行特征嵌入编码；然后设计多头注意力解码器(Multi-head Attention Decoder，MAD)进行解码，并提出多解码策略以增加解的空间多样性；接着利用带回滚基线的基线 REINFORCE 算法对端到端网络模型进行训练，基线可自适应性更新以提升模型训练效果，并利用奖励函数归一化和 Adam 优化器对算法进行优化。最后通过对不同规模问题的实验以及与其他算法进行对比，验证了所提出端到端 DRL 框架的可行性与有效性，经过训练的模型在 CVRPLIB 公共数据集上的平均求解时间仅需 0.189s 即可得到较优解。

关键词：车辆路径问题；路径规划；端到端模型；深度强化学习；基线 REINFORCE 算法
中图分类号：TP399 doi: 10.19734/j.issn.1001-3695.2024.03.0101

Solving capacitated vehicle routing problems based on end to end deep reinforcement learning

Ge Bin[1], Tian Wenzhi[1], Xia Chenxing[1,2], Qin Wangbo[1]

(1. *School of Computer Science & Engineering*, *Anhui University of Science & Technology*, *Huainan Anhui* 232001, *China*; 2. *Institute of Energy*, *Hefei Comprehensive National Science Center*, *Hefei Anhui* 230031, *China*)

Abstract: The Capacitated Vehicle Routing Problem (CVRP) is the most prevalent problem model in supply chain applications at present, and researchers often use heuristic algorithms to solve it, but the solution speed is slow and the quality of the solution cannot be guaranteed. This paper proposes an end-to-end Deep Reinforcement Learning (DRL) network framework to study the CVRP problem. Firstly, using the Edge Graph Attention Network Encoder (EGATE) performs feature embedding encoding on the graph representation of Vehicle Routing Problems; Then, design a multi head attention decoder (MAD) to decode the encoded graph representation. Additionally, proposing a multi-decoding strategy to enhance the spatial diversity of the solutions. Continuing with the training of the end-to-end network model using the baseline REINFORCE algorithm with a rollout baseline, the adaptive updating of the baseline is employed to enhance the effectiveness of model training. Additionally, reward function normalization and optimization using Adam optimizer are utilized to further improve the algorithm. Finally, We validated the feasibility and effectiveness of the proposed end-to-end DRL framework through experiments on problems of different scales, comparing its performance against other algorithms. The average solution time of the trained model on the CVRPLIB public dataset is only 0.189 seconds to obtain a better solution.

Key words: vehicle routing problem; path planning; end to end model; deep reinforcement learning; baseline reinforce algorithm

0 引言

车辆路径问题(Vehicle Routing Problem,VRP)属于组合优化问题中典型的 NP-hard 问题[1]，由 George Dantzig 和 John Ramser 提出。有能力约束的车辆路径问题(Capacitated Vehicle Routing Problem, CVRP)是 VRP 问题中的一种基本问题，在运输和物流配送中起着关键作用。

求解 CVRP 问题的传统方法主要分为精确算法、近似求解法和启发式算法[2]。精确算法虽然能够求解出最优解，但是只在规模较小的求解情况下有效。近似求解法虽然能够快速完成计算，但无法保证解的质量。启发式算法如蚁群优化算法[3]、帝国竞争算法[4]、烟花算法[6]、蛙跳算法[7]、蝙蝠算法[8]，由于其高效的求解能力被广泛应用，但是启发式算法需要针对不同问题设定特定的启发措施。CVRP 涉及多个变量，如车辆位置、

收稿日期：2024-03-21；**修回日期：**2024-06-06 **基金项目：**国家重点研发计划(2020YFB1314103)
作者简介：葛斌(1973—)，男，安徽安庆人，教授，硕导，博士，主要研究方向为机器学习、智能规划，E—mail：bge@aust.edu.cn；田文智(2001—)，男，安徽阜阳人，硕士研究生，主要研究方向为智能规划、深度学习；夏晨星(1991—)，男，湖北大冶人，副教授，硕导，博士，主要研究方向为深度预测；秦望博(1998—)，男，安徽亳州人，硕士研究生，主要研究方向为深度预测.

剩余容量、客户位置及其需求等，这些变量的组合形成一个高维度的状态空间。上述三种传统求解方法在如此庞大的状态空间中寻找最优解往往面临计算复杂性和效率问题，尽管已有大量的求解策略，但仍然需要搭建更加高效的求解模型来进一步提升 CVRP 问题求解效率。

目前，DRL 模型在面对复杂、数据规模较大的路径规划情景展现出的高效能力广受研究者青睐，并取得突破性进展[9]。Vinyals 等人[10]介绍了指针网络(Pointer Network，PN)，该网络使用长短期记忆网络(Long-Short Term Memory，LSTM)作为编码器，注意力机制(Attention Mechanism，AM)作为解码器，离线训练该模型以解决车辆路径问题。受 Vinyals 等人的指引，Ma 等人[11]把 PN 与 GNN 相结合并提出图指针网络(graph pointer network，GPN)，利用 GNN 提取计算节点特征，再用 PN 进行解的构造，提升了大规模车辆路径问题的求解能力。由 Transformer[12]架构的启发，Kool 等人[13]借用 Transformer 提出新框架，其利用注意力机制对模型进行改进，超越了先前解决路径问题的优化性能。García-Torres R 等人[14]以注意力机制为核心开发了组合的深度构造器和扰动器来解决带约束的车辆路径规划问题，此外还提出内存有效算法，大幅降低内存复杂性。Hu 等人[15]提出了一种双向图神经网络(BGNN)，通过模仿学习依次生成下一个访问节点，并且能够与启发式搜索相结合以进一步提高性能。Zhu 等人[16]提出一种基于门控余弦的注意力模型(GCAM)来训练策略模型，加速了模型的收敛过程。上述学者虽然对车辆路径规划算法的研究采用 DRL 框架，但是在使用深度强化学习的求解路径规划问题时，对图表示信息考虑不充分、设计的解码方式较为单一，导致模型得到的初始解的质量较差，从而使求解结果不理想。

针对 CVRP 问题所存在的问题以及深度强化学习框架目前所存在的局限性，本文提出一种端到端的深度强化学习网络框架 EGATE-MAD(A Edge-Graph Attention Network Encoder and Multi-head Attention Decoder)用于高效求解 CVRP 问题。在该框架中，首先将 CVRP 问题中的边信息和节点信息通过 EGATE 模块进行信息聚合编码，可获得更加完整的图结构信息；接着通过多解码器模块对嵌入信息进行解码，并在解码时使用多解码器策略来增加解的空间多样性，以提高模型的求解精度；然后设计改进基线 REINFORCE 算法来训练 EGATE-MAD 模型；最后通过实验验证所提出的端到端深度强化学习框架在 CVRP 问题上的高效性。

1　问题描述

CVRP 问题如图 1 所示。存在单个配送中心为多个客户节点配送所需货物，配送中心拥有一定数量的车辆，运输车辆的最大容量和配送中心的坐标位置已知；每个客户节点的需求量以及位置坐标已知；目标是通过最优车辆路径规划实现总路径最小化。

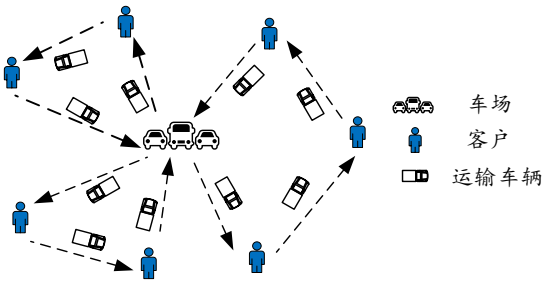


图 1　CVRP 示意图

Fig. 1　Schematic diagram of CVRP

为了便于分析和研究，对该问题提出四点假设：

1)配送中心只有一个，所有车辆均从配送中心出发且最终返回配送中心；

2)在安排车辆运输时，已经获取了各客户节点的位置信息；

3)每辆运输车的载货容量相同；

4)每个客户点的需求均小于车辆最大容量。

本文通过无向图 $G=(V,E,W)$ 定义 CVRP 问题，$\boldsymbol{n}_i$ 表示节点，其中 $i\in V=\{0,1,\cdots m\}$，m 表示节点的数量，n_i 表示第 i 个节点的坐标，下标 $i=0$ 表示配送中心节点，$i(i>0)$ 表示第 i 个客户节点，$a_{ij}\in E,i,j\in V$ 代表节点 i 到节点 j 之间的边，$e_{ij}\in W$ 表示 a_{ij} 的距离信息。结果集 $\hat{\pi}=(\hat{\pi}_1,\cdots,\hat{\pi}_m)$ 表示路径规划序列，$\hat{\pi}_t\in\{1,\cdots,m\}$，$\hat{\pi}_t\neq\hat{\pi}_v$，$\forall t\neq t'$。每辆车都有各自需求大小 $D>0$，每一个客户节点 i 存在一个需求 σ_i，$0<\sigma_i<D$。并定义车场节点的需求 σ_0 为 0。

根据上述定义，端到端网络模型对 CVRP 问题进行求解，如图 2 所示。端到端模型捕捉 CVRP 问题中客户和车辆之间复杂的空间关系，通过处理图结构数据，能够有效地表示 CVRP 实例中的节点信息(客户)和边信息(路径)，即编码器将所有输入实例的节点特征编码映射到高维特征。在给定的问题实例 s 情况下，解码器根据编码器的输出在每一个时间步长 t 根据解码得到概率矩阵，采用搜索策略选择下一个节点从而产生结果序列 $\hat{\pi}$，不仅可以满足问题约束，而且使总路程长度最小化，总路程长度被定义为

$$L(\hat{\pi}\,|\,s)=\left\|\boldsymbol{n}_{\hat{\pi}_m}-\boldsymbol{n}_{\hat{\pi}_1}\right\|_2+\sum_{t=1}^{m-1}\left\|\boldsymbol{n}_{\hat{\pi}_t}-\boldsymbol{n}_{\hat{\pi}_{t+1}}\right\|_2 \tag{1}$$

其中，$\|\cdot\|_2$ 表示 L2 范式，本文端到端网络模型为实例 s 定义一个随机策略 $p(\hat{\pi}\,|\,s)$，基于链式概率规则，序列 $\hat{\pi}$ 的选择概率由网络模型参数集 θ 计算：

$$p_\theta(\hat{\pi}\,|\,s)=\prod_{t=1}^{m}p_\theta(\hat{\pi}_t\,|\,s,\hat{\pi}_{t'},\forall t'<t) \tag{2}$$

采用回滚基线的 REINFORCE 强化学习算法，通过模拟和经验积累，逐步优化路径规划策略 $p(\hat{\pi}\,|\,s)$ 以达到收敛。后续将介绍上述端到端模型的搭建以及强化学习算法的设计。

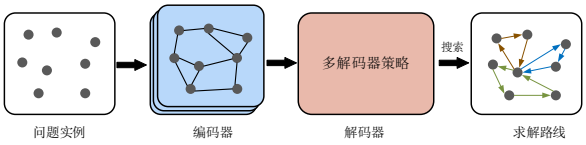


图 2　端到端模型求解 CVRP 问题示意图

Fig. 2　Schematic diagram of end-to-end model solving CVRP

2　端到端模型建立

由于 GAT 其对图拓扑结构具有强大的表示能力，文献[17][19]在解决组合优化问题时均采用其进行编码，鉴于此，本文的编码器 EGATE(Edge-Graph Attention Network Encoder)基于图注意力网络(Graph Attention Network，GAT)设计。同时解码器是基于 Transformer 进行构建，但由于 Transformer 的输出维度预先固定，不能根据输入维度而变化，因此不能直接用于解决组合优化问题(CVRP 问题是组合优化问题的一种)。Vinyals 等人[9]提出的 PN 使用注意力机制，基于 softmax 概率分布，在每个解码步骤从输入序列中选择一个成员作为输出。PN 使 Transformer 模型能够应用于组合优化问题，其中输出序列的长度由源序列决定。考虑到这一想法，解码器的设计遵循 PN 的方式来输出节点，其中每个节点在每个解码时间步长通过使用 softmax 概率分布作为“指针”的概率值。

2.1　编码器

将图 $G=\langle V,E,W\rangle$ 作为输入，经过编码器模块，得到融合的节点嵌入信息，如图 3 所示。输入的节点特征由一个二维坐标 $\boldsymbol{n}_i$ 和采用欧几里德算法得到的边信息特征 $\boldsymbol{e}_{ij},i,j\in\{1,2,\cdots,m\}$ 组成。之后再通过全连接层(图 3 中的 FC Layer)将维度扩展到 d_x 维和 d_e 维。嵌入操作是由一个 W_i 和 b_i 组成的可学习线性变换执行，即

$$\boldsymbol{x}_i^{(0)}=BN\left(\boldsymbol{W}_0\boldsymbol{n}_i+\boldsymbol{b}_0\right),\forall i\in\{1,\cdots,m\} \tag{3}$$

$$\hat{\boldsymbol{e}}_{ij}=BN\left(\boldsymbol{W}_1\boldsymbol{e}_{ij}+\boldsymbol{b}_1\right),\forall ij\in\{1,\cdots,m\} \tag{4}$$

其中，用 $\boldsymbol{x}_i^{(l)}$ 来表示 EGAT 模块中第 l 层嵌入，$l\in\{1,\cdots,L\}$，$BN(\cdot)$ 表示批归一化处理。$\boldsymbol{x}_i^{(0)}\left(\boldsymbol{x}_1^0,\boldsymbol{x}_2^0,\cdots,\boldsymbol{x}_i^0\right)$，$\boldsymbol{x}_i^{(0)}\in R^{d_x}$ 表示 EGAT 中的第一层输入，随着层数的增加，$\boldsymbol{x}_i^{(l)}$ 随之变换，而边信息特征 $\hat{\boldsymbol{e}}=\{\hat{\boldsymbol{e}}_{11},\hat{\boldsymbol{e}}_{12},\cdots,\hat{\boldsymbol{e}}_{mm}\},\hat{\boldsymbol{e}}_{ij}\in R^{d_e}$ 将保持不变。

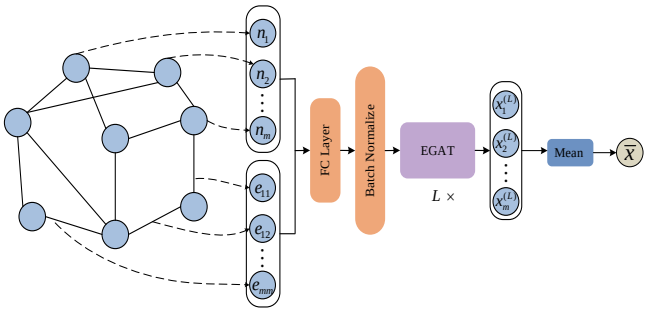


图 3　边聚合图注意力网络编码器结构

Fig. 3　Edge aggregation graph attention network encoder structure

图 4 详细描述了 EGAT 如何将节点嵌入信息和边嵌入信息进行聚合。将扩展嵌入信息 $\left(\boldsymbol{x}_i^{(l)},\hat{\boldsymbol{e}}_{ij}\right)$ 作为输入。注意力权重向量计算公式为

$$\boldsymbol{h}_{concat,i,j}=concat\left(\boldsymbol{x}_i^{(l)},\boldsymbol{x}_j^{(l)},\hat{\boldsymbol{e}}_{ij}\right) \tag{5}$$

$$\alpha_{i,j}=Leaky\,\mathrm{Re}\,LU\left(\boldsymbol{W}_L*\boldsymbol{h}_{concat,i,j}\right) \tag{6}$$

$$\tilde{\boldsymbol{w}}_{i,j}=\frac{\exp(\alpha_{i,j})}{\sum_j\exp(\alpha_{i,j})} \tag{7}$$

其中 $\boldsymbol{W}_L$ 为可学习的权重矩阵，然后，通过类似于 GAT 的注意力机制更新每个节点信息的嵌入以产生 EGAT 的输出：

$$\boldsymbol{x}_i^{(l+1)}=\boldsymbol{x}_i^{(l)}+\sum_j\tilde{\boldsymbol{w}}_{ij}\otimes\boldsymbol{n}_j^{(l)} \tag{8}$$

其中，$\tilde{\boldsymbol{w}}_{ij}$ 代表可以进行学习的权重矩阵，$\boldsymbol{x}_i^{(l)}$ 经过一轮 EGAT 模块后得到融合边的嵌入信息 $\boldsymbol{x}_i^{(l+1)}$。最后，将最高层输出的节点嵌入 $\boldsymbol{x}_i^{(L)}$ 被进一步送到平均池化层，得到表示整个解决方案的编码器的最终输出 $\bar{\boldsymbol{x}}$。

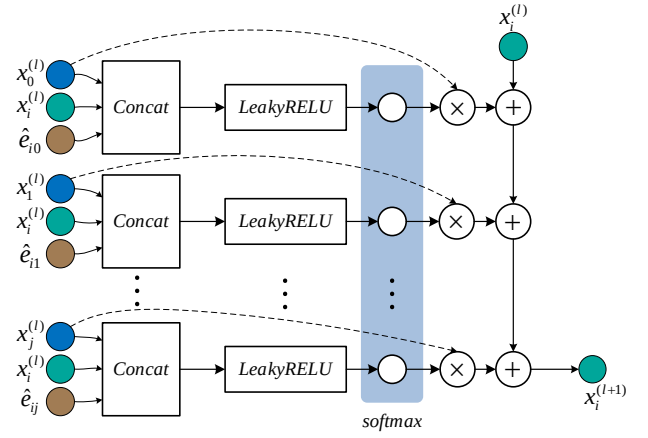


图 4　边、点信息聚合网络结构图

Fig. 4　Structure diagram of edge and node information aggregation network

2.2　解码器

解码器采取类似于 Transformer 模型的结构，但不使用残差连接、批归一化处理以及全连接层，而是使用两个注意力子层代替。在此基础之上设计了多解码器结构，即具有相同结构但不共享参数的多个解码器，不同的解码器使用索引 $d\in\{1,\cdots,D\}$ 来表示。图 5 说明了单个解码器的解码过程。

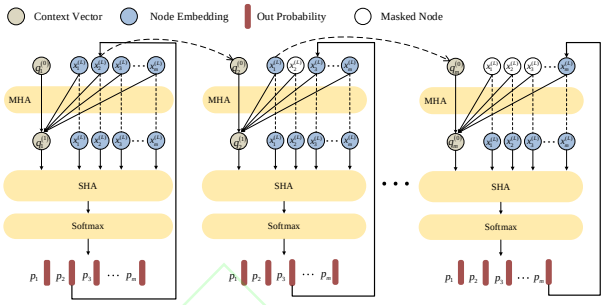


图 5　多头注意力解码器结构

Fig. 5　Structure of Multi head Attention Decoder

其中，第一层通过多头注意力机制来计算上下文向量，第二层通过自注意力层和 softmax 层来输出将要选择节点的概率分布。解码器在每个时间步长 $t\in\{1,\cdots,m\}$，会根据节点嵌入信息，和先前的输出节点 $\hat{\pi}_{t'}$ 的信息生成 $\hat{\pi}_t$，$(t<t')$。在 t 时刻，解码器的上下文信息输入 $\boldsymbol{q}_t^{(0)}$ 是通过第一个输出的节点信息 $\hat{\pi}_1$ 和最后一个输出的节点信息 $\hat{\pi}_{t-1}$ 计算而来。在 $t=1$ 时，$\boldsymbol{q}_1^{(0)}$ 可根据编码器输的出全局图嵌入信息 $\bar{\boldsymbol{x}}$ 和一个可学习的 d_x 维的参数向量 $\vec{v}$ 进行计算：

$$\boldsymbol{q}_t^{(0)}=\begin{cases}\bar{\boldsymbol{x}}+\boldsymbol{W}_x\left(\boldsymbol{x}_{\pi_1}^{(L)}\,\|\,\boldsymbol{x}_{\pi_{t-1}}^{(L)}\right),t>1\\ \bar{\boldsymbol{x}}+\vec{\boldsymbol{v}},t=1\end{cases} \tag{9}$$

其中，$\boldsymbol{W}_x$ 是一个可学习的矩阵，$\boldsymbol{q}_t^{(0)}$ 表示解码器第一层的上下文信息，之后，通过多头(H 个)注意力机制产生的新的上下文信息 $\boldsymbol{q}_t^{(1)}$。

针对解码器的多头注意力机制，本文定义维度 d_v，通过节点嵌入信息和上下文向量 $\boldsymbol{q}_t^{(0)}$ 来计算 $\boldsymbol{k}_i\in R_v^d$，$\boldsymbol{v}_i\in R_v^d$，$\boldsymbol{q}\in R_v^d$：

$$\begin{cases}\boldsymbol{q}=\boldsymbol{W}^Q\boldsymbol{q}_t^{(0)},\\ \boldsymbol{v}_i=\boldsymbol{W}^V\boldsymbol{x}_i^{(L)},i\in\{1,2,\cdots,m\}\\ \boldsymbol{k}_i=\boldsymbol{W}^K\boldsymbol{x}_i^{(L)},\end{cases} \tag{10}$$

其中，$\boldsymbol{W}^K\in R^{d_v\times d_x},\boldsymbol{W}^Q\in R^{d_v\times d_x}$，$\boldsymbol{W}^V\in R^{d_v\times d_x}$，$(d_v=d_x/H)$ 为可学习的权重矩阵。然后，$\boldsymbol{q}$ 和 $\boldsymbol{K}=\{\boldsymbol{k}_1,\cdots\boldsymbol{k}_m\}$ 用来计算第一层解码器中 t 时刻的注意力系数 $u_{i,t}^{(1)}\in R,i\in\{1,\cdots,m\}$。

$$u_{i,t}^{(1)}=\begin{cases}\dfrac{\boldsymbol{q}^T\boldsymbol{k}_i}{\sqrt{d_v}}if\ i\neq\hat{\pi}_{t'}\ \forall t'<t,\\ -\infty,otherwise.\end{cases} \tag{11}$$

其中，将在 t 时刻已经被选择的节点的注意力系数设置为 $-\infty$。之后，使用 softmax 激活函数将注意力系数 $u_{i,t}^{(1)}$ 进行归一化处理，如式(12)所示。

$$\hat{u}_{i,t}^{(1)}=soft\max\left(u_{i,t}^{(1)}\right) \tag{12}$$

接下来 H 头注意力机制通过式(13)进行操作，第 H 头归一化注意力系数表为

$$\boldsymbol{q}_t^{(1)}=\boldsymbol{W}_f\cdot\left(\|_{h=1}^H\sum_{i=1}^m\left(\hat{u}_{i,t}^{(1)}\right)^h\boldsymbol{v}_i^h\right) \tag{13}$$

其中，$\boldsymbol{W}_f$ 是可训练权重矩阵。多头注意力机制有助于增强注意力学习过程的稳定性。

第二层是一个以上下文信息向量 $\boldsymbol{q}_t^{(1)}$ 作为输入的自注意

力机制层，使用式(12)来计算此层的时刻 t 下的注意力系数 $\hat{u}_{i,t}^{(2)} \in R, i \in \{1,\cdots,m\}$，将注意力系数用 $tanh$ 函数划分在 $[-C,C]$，$C=10$。对于任意节点 $i \in \{1,\cdots,m\}$，都可以通过式(14)获得它的选择概率 $p_{i,t}$。最后，根据概率分布 $p_{i,t}$，可使用采样或贪婪策略来预测要访问的下一个节点，直至形成一次路径规划任务。

$$p_{i,t} = p_\theta(\hat{\pi}_t \mid s, \hat{\pi}_{t'} \forall t' < t) = soft\max(u_{i,t}^{(2)}) \tag{14}$$

本文对车辆剩余容量、解码器上下文向量与掩码进行如下设置。

车辆剩余容量更新：屏蔽已经服务过的需求节点，因此，不需要更新已服务需求节点的需求。解码器在 t 时刻选择下一个节点 $\hat{\pi}_t$，剩余的车辆容量用 D_t' 表示，并且按照以下方式更新：

$$D'_t = \begin{cases} D'_{t-1} - \delta'_{\hat{\pi}_t}, \hat{\pi}_t = i, i \in \{1,\cdots,m\} \\ D, \hat{\pi}_t = 0 \end{cases} \tag{15}$$

解码器上下文向量更新：解码器 t 时刻的上下文向量 $\boldsymbol{q}_t^{(0)}$ 由图嵌入信息 $\bar{\boldsymbol{x}}$，节点嵌入信息 $\hat{\pi}_t$ 和车辆剩余容量 D'_{t-1} 组成，如式(16)所示。

$$\boldsymbol{q}_t^{(0)} = \begin{cases} \bar{\boldsymbol{x}} + \boldsymbol{W}_x\left(\boldsymbol{n}_{\hat{\pi}_{t-1}}^{(L)} \| D'_{t-1}\right), t > 1 \\ \bar{\boldsymbol{x}} + \boldsymbol{W}_x\left(\boldsymbol{n}_0^{(L)} \| D'_t\right), t = 1 \end{cases} \tag{16}$$

掩码更新：掩码由两部分组成：需求节点掩码和车场节点掩码，掩码操作屏蔽已经服务的需求节点或需求节点需求大于车辆剩余容量时的需求节点。即当 $\delta'_i > D'_{t-1}$ 或 $i \neq \hat{\pi}_{t'} \forall t' < t, i \in \{1,\cdots,m\}$ 时，将需求节点的注意力系数进行如下更改：$u_{i,t}^{(0)}, u_{i,t}^{(1)} = -\infty$。对于车场掩码，如果车场不是下一个要访问的节点，或者刚从仓库节点离开，即当 $t=1$ (车辆刚离开车场节点)或者 $\hat{\pi}_{t-1} = 0$，(车辆在车场节点)作如下更新：$u_{0,t}^{(0)}, u_{0,t}^{(1)} = -\infty$。

图 6 显示了多解码器的结构，其中的单个解码器模块如图 5 所示。它们具有相同结构但是不共享参数，每个单独的解码器都可以生成一个排列 $\pi^d = (\hat{\pi}_1,\cdots,\hat{\pi}_m)$，$d \in (1,\cdots D)$，从排列的排列池中择优选择最终解决方案以增加解空间的多样性。

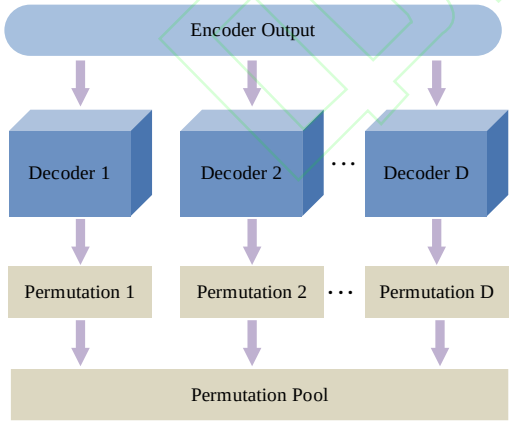


图 6　多解码策略

Fig. 6　Multiple decoding strategies

3　回滚基线 REINFORCE 算法

本文引入改进的基线 REINFORCE 算法来训练端到端模型。该算法通过计算单智能体的累计回报估计策略梯度并训练单智能体策略。将损失函数定义为 $Loss(\theta \mid s) = E_{\hat{\pi} \sim p_\theta(\hat{\pi} \mid s)}[L(\hat{\pi} \mid s)]$。参数 θ 通过一种带回滚基准 b 的 REINFORCE 算法进行优化，如式(17)所示。

$$\nabla_\theta Loss(\theta \mid s) = E_{\hat{\pi} \sim p_\theta(\hat{\pi} \mid s)}[(L(\hat{\pi} \mid s) - b)\nabla_\theta \log p_\theta(\hat{\pi} \mid s)] \tag{17}$$

Kool 等人[13]在解决路径规划问题时提出了带回滚基准的 REINFORCE 算法，并证明了其有效性。算法流程如图 7 所示。该算法基于 actor-critic 算法[21]，actor 网络指的是前文的图注意力网络，而 critic 网络是由多个一维卷积层组成，并且与 actor 共享相同的编码器网络。actor-critic 算法中的 critic 网络被基线 actor 网络所取代，该网络可以被描述为双 actor 结构。在每个历元结束时，使用解码策略来比较当前训练策略和基线策略的结果。然后，根据现有研究中对评估实例进行在显著性水平为 α(α=0.05)的 T 检验，如果策略网络输出的解显著优于基准网络，则对基准网络参数 θ 进行回滚更新。在训练过程中，本文使用了包括奖励函数归一化和 Adam 优化器学习率退火在内的“代码级优化”来提高基线 REINFORCE 算法的训练能力。

每个小批量中的奖励函数归一化：在标准的基线方法实现中，不是将奖励直接从环境中输入目标，而是执行某种基于折扣的缩放方案。本文没有执行这种基于折扣的缩放方案，相反，通过减去每个小批量中的平均值并除以标准偏差来归一化奖励 $L(\hat{\pi} \mid s)$。

Adam 优化器学习率退火：在训练过程中使用了学习率衰减，并在每个历元结束时更新了学习率：

$$l_{new} = l_{old} \cdot (\beta)^{epoch} \tag{18}$$

其中，l_{new} 和分别为更新前后的学习率，β 是退火率。图 7 是算法整个基线算法流程。

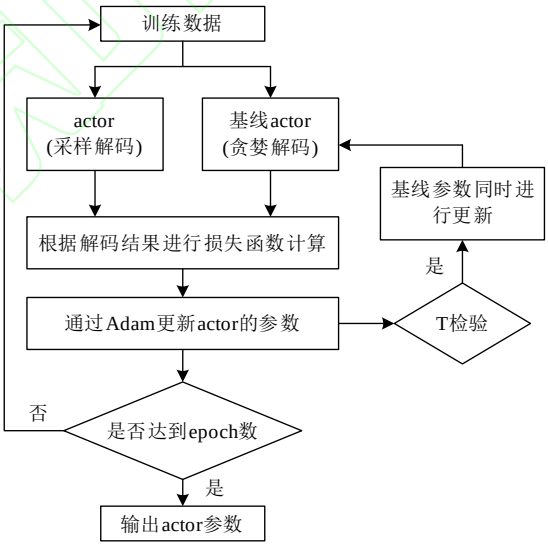


图 7　回滚基线 REINFORCE 算法流程

Fig. 7　Rollback baseline REINFORCE algorithm process

用于组合优化问题的有效搜索算法包括波束搜索、邻域搜索、树搜索、采样搜索、贪婪搜索等。本文使用采样搜索算法以及贪婪搜索算法进行训练。在采样搜索时，在解码的每个时刻 $t \in \{1,\cdots,m\}$，随机策略 $p_{\theta(\hat{\pi}_t \mid S, \hat{\pi}_{t'}, \forall t' < t)}$ 根据当前概率分布对要选择的节点进行采样，为了提高采样的多样性，本文采用 Kool 等人[13]所提到的方法，使用温度超参数来修改采样方程，修改后的方程如下：

$$p_{i,t} = p_\theta(\hat{\pi}_t \mid s, \hat{\pi}_{t'} \forall t' < t) = soft\max\left(\frac{u_{i,t}^{(2)}}{\lambda}\right) \tag{19}$$

其中，采取 Kool 等人[13]的设定，取超参数 λ=2，1.8，1.2(分别对应 CVRP20、50、100)确保采样的多样性，使 $p_{i,t}$ 的分布略微分散，防止模型盲目自信。在训练过程中，通常需要随机采样来探索环境，以获得更好的模型性能，而贪婪解码总是选择概率最大的节点，虽然搜索速度大幅提高，但是模型性能会降低。

4　实验

4.1　实验环境设置

为证明本文所提出端到端深度强化学习框架的效果，本文在文献[22]定义的车辆路径问题环境下进行实验，该环境自提出以来被用于训练及测试国内外各类 DRL 方法来解决车辆路径规划相关问题[22~25]。

分别对三个问题规模为 20、50 和 100 的 CVRP 问题进行实验，实验所需训练数据、验证数据、和测试数据都是在单位正方形 $[0,1]\times[0,1]$ 中随机均匀绘制。生成节点个数为 21，51，101 的实例，其中第一个节点为车场节点，因此，每种路径规划问题实例的规模为 20，50，100，以及对应的车的容量设为 30，40，50。需求节点的需求从 $\{1,2,\cdots,9\}$ 进行均匀采样，本文将需求节点的需求通过公式 $\sigma_i'=\sigma_i/10$ 进行标准化处理，使需求分布在[0,1]范围内，因此，对应的车容量变为 3，4，5。在训练阶段，为基线 REINFORCE 强化学习算法生成节点规模为 20 和 50 的实例各 409600 个，节点规模为 100 的实例 384000 个。训练迭代次数为 100。对于验证数据和测试数据，使用与其他研究相同数据分布的实例数据。此外，采用来自于公共数据库 CVRPLIB 的基准测试数据。

基于上述设定，实验在 GPU：RTX A4000(16GB)、CPU：12 vCPU Intel(R) Xeon(R) Gold 5320 CPU @ 2.20GHz、32GB 的配置上使用 PyTorch 进行构造，并用 Python3.7 实现。表 1 列出了训练过程的相关超参数的值。

表 1　超参数设定

Tab. 1　Hyperparameter settings

参数	值
编码器层数 l	4
学习率衰减 β	0.96
基线学习率 φ	$1\times10^{-3}(m=20)$ $3\times10^{-4}(m=50,100)$
多头注意力头数 H	8
节点嵌入维度 d_x	128
边嵌入维度 d_e	64

4.2　实验结果分析

实验结果如表 2、表 3 所示。表 2 列出了所提框架在不同规模(CVRP20、50、100)的生成实例上的训练结果。分别与近年来较为热门的算法：AM[13]、CDCP[14]、GCAM[16]、Wu's[23]、AM-D[24]、L2I[26]、LKH3[27]、HGS[28]、Gurobi 进行比较，其中，Gurobi 为组合优化问题的高效求解器。表 3 选取公开数据集 CVRLIB 中的不同实例对已经训练好的模型进行测试并与蚁群算法(ACO)、超启发式算法 HH_PG[29]进行对比。其中，距离(越小越好)；Gap 值：与现有最优结果的差距(CVRP20、50、100 的最优结果分别来自于 Gurobi、LKH3[27]、L2I[26])，越小越好；OPT：最优值；时间：总耗时，单位为秒(越少越好)，值为 10000 个实例求解时间的平均值。

表 2 不同方法在各个规模 CVRP 上的对比实验结果

Tab. 2　Comparative experimental results of different methods on different scales of CVRP

方法	CVRP20			CVRP50			CVRP100		
	距离	Gap(%)	时间	距离	Gap(%)	时间	距离	Gap(%)	时间
Gurobi	6.10	0.00	-	-	-	-	-	-	-
L2I	6.12	0.33	-	10.35	0.00	-	15.57	0.06	-
HGS	-	-	-	-	-	-	15.56	0.00	6h11m
LKH3	6.14	0.66	2h	10.38	0.29	7h	15.65	0.58	13h
AM(Sampling)	6.25	2.46	6m	10.62	2.61	28m	16.23	4.31	2h
AM-D(Greedy)	6.28	2.95	3s	10.78	4.15	25s	16.40	5.40	2m39s
GCAM(Sampling)	6.28	2.95	6m	10.64	2.50	20m	16.29	4.69	1h
Wu's	6.12	0.33	2h	10.45	0.97	4h	16.03	3.02	5h
CDCP	6.13	0.49	4h53m	10.47	1.16	10h12m	15.85	1.86	19h22m
Ours(Greedy)	6.26	2.60	3s	10.80	4.10	7s	16.69	6.68	17s
Ours(Sampling)	6.18	1.46	15m	10.55	1.64	1h	16.10	3.47	4h

*表中“-”表示源文献未提供该结果

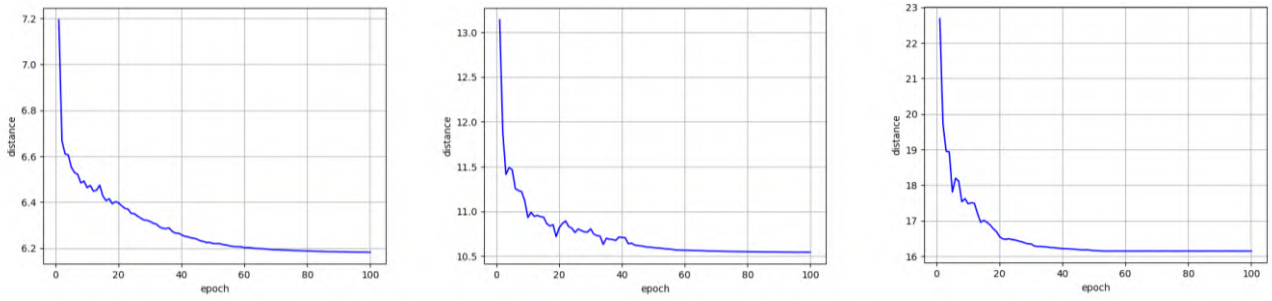
由表 2 可以看出，采样解码策略(Ours(Sampling))取得了优异的效果，优于 AM、AM-D、GCAM 算法，仅次于 CDCP、Wu's。但是在求解时间上，以规模 20 的实验结果为例，训练时间远少于他们。贪婪解码(Ours(Greedy))贪婪的选择概率最大的节点作为下一个访问节点，又由于神经网络进行并行计算能够同时处理多个算例，从而让贪婪解码能够以非常快的速度求解问题。例如 CVRP20 中，Ours(Greedy)耗时为 3s，而 Ours(Sampling)耗时 15m。此外，训练迭代曲线如图 8 所示，可以看出各个规模的训练在 100 个 epoch 内都可以得到很好的收敛。图 9 为 CVRP20、50、100 训练的可视化结果，其中，每个回路代表车辆从车场出发，运输完成后返回车场节点。

表 3 中的实验结果进一步证明了本文所提模型的求解能力，该实验选取公开数据集 CVRLIB 中的不同实例对已经训练好的模型进行测试。实验使用规模为 50 的已训练模型，采用贪婪解码策略对每个实例进行单次求解，并且与蚁群算法(ACO)、超启发式算法 HH_PG[29]进行对比。由表 3 可知，虽然 HH_PG 算法在求解质量上优于本文方法(2.80% vs 0.032%)，但是 HH_PG 算法的结果是针对每个算例进行单独执行 20 次并选出最优解。相反，本文所提框架对每个算例只进行单次求解，求解时间远优于 HH_PG 算法和 ACO 算法(0.189s vs 142.14s)。因此，本文所提框架在求解质量和运行时间上较为均衡，并且可以在本地进行训练，在实际应用中进行实时求解。

4.3　消融实验

为证明所设计端到端深度强化学习框架中模块的有效性，在本小节完成了消融实验工作。需要证明该框架中的两个重要组成部分均有效，Modle1：边信息的融入模块；Model2：多解码器策略模块。为此分别设计了四个实验，分别为基模型 Base、Base+Model1、Base+Model2、Base+Model1+Model2。四种实验使用相同的参数，在三种问题规模(20、50、100)的数据集上进行训练，并且分别使用相应规模的测试数据来进

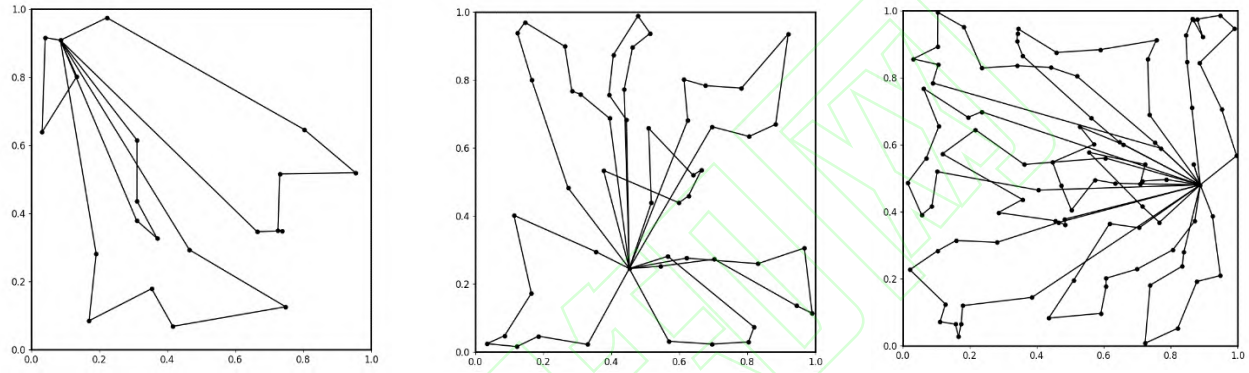
行测试并比较，消融实验结果详情见表 4。从表 4 中可以得出以下结论：当完整模型缺少 Model1、Model2 中任意模块或同时缺少两种模块时，平均距离均增加。因此，本文所提出的端到端深度学习框架中两个重要组成部分均有效。



(a)CVRP20　　(b)CVRP50　　(c)CVRP100

图 8　各规模训练迭代曲线

Fig. 8　Iteration curves for training at different scales



(a)CVRP20　　(b)CVRP50　　(c)CVRP100

图 9　各规模训练可视化结果

Fig. 9　Visualization results of training at different scales

表 3 端到端深度强化学习框架应用于部分 CVRLIB 数据集的结果

Tab. 3　Results of applying end-to-end DRL models to partial CVRLIB datasets

Problem	OPT	OURS			ACO			HH_PG		
		距离	Gap(%)	时间(s)	距离	Gap(%)	时间(s)	距离	Gap(%)	时间(s)
A-n32-k5	784	789	0.63	0.16	865	10.33	15.12	784	0	0.73
A-n33-k6	742	760	1.94	0.18	832	12.13	16.02	742	0	0.93
A-n36-k5	799	830	3.99	0.15	921	21.71	25.58	799	0	10.37
A-n37-k6	949	971	2.31	0.18	1079	18.13	29.85	949	0	12.64
A-n39-k5	822	838	1.94	0.10	960	16.79	19.41	822	0	11.87
A-n44-k6	937	984	5.01	0.15	1034	9.53	23.42	937	0	114.67
A-n45-k7	1146	1182	3.14	0.26	1332	16.23	23.59	1146	0	114.98
A-n48-k7	1073	1123	4.65	0.28	1306	21.71	25.58	1073	0	6.29
A-n54-k7	1167	1176	0.77	0.16	1337	14.57	30.03	1167	0	221.02
A-n60-k9	1354	1365	4.29	0.22	1582	16.84	30.70	1354	0	250.80
A-n62-k8	1288	1324	0.81	0.15	1489	15.61	36.20	1289	0.078	262.30
A-n63-k10	1314	1352	2.89	0.18	1601	21.84	37.10	1315	0.076	281.25
A-n65-k9	1174	1246	6.13	0.18	1440	22.66	38.72	1174	0	266.12
A-n69-k9	1159	1182	1.98	0.26	1388	19.76	42.23	1159	0	250.67
A-n80-k10	1763	1791	1.59	0.22	2205	25.07	52.28	1769	0.340	327.48
Avg	1098	1128	2.80	0.189	1291	17.52%	29.72	1099	0.032	142.14

表 4 消融实验结果

Tab. 4　Results of ablation experiment

实验	模块		CVRP20	CVRP50	CVRP100
	Model1	Model2			
Base			6.235	10.721	16.323
Base+Model1	√		6.204	10.712	16.241
Base+Model2		√	6.217	10.652	16.305
Base+Model1+Model2	√	√	6.184	10.552	16.102

5　结束语

本文提出一种端到端深度强化学习的网络框架来解决车辆路径规划问题。在编码器部分，以图注意力网络为基础，编码器将节点信息和边信息充分融合得到编码信息。在解码器部分，使用基于 Transformer 的解码器进行解码，采用多解码器策略来增加解的空间多样性。接着为端到端网络模型设计了基于回滚基线 REINFORCE 的强化学习算法进行训练，并使用奖

励函数归一化和 Adam 优化器进行优化。最后通过随机生成的大量实例以及公开数据集进行实验并与现有的深度强化学习方法、启发式方法作对比。此外，还设置了消融实验证明模块的有效性。实验结果表明，所提框架在各种实例上的求解时间和求解质量更为均衡，可以实现线下训练和线上实时求解，并且边信息融入和多解码器结构模块对整体框架均有效。

本研究考虑在固定分布和规模上进行深度模型训练，由于车辆路径规划问题的随机性较强，对于不同规模、不同分布、甚至未知规模或者分布的实例，模型的求解能力会降低。未来，在本文的研究基础上，将考虑如何提高模型在不同规模或者分布上的泛化能力。

参考文献：

[1] 蔡延光, 王世豪, 戚远航, 等. 帝国竞争算法求解 CVRP [J]. 计算机应用研究, 2021, 38 (03): 782-786. (Cai Yanguang, Wang Shihao, Qi Yuanhang, *et al.* Empire CVRP competition algorithm [J]. Application Research of Computers, 2021, 38 (03): 782-786.)

[2] 李凯文, 张涛, 王锐等. 基于深度强化学习的组合优化研究进展 [J]. 自动化学报, 2021, 47 (11): 2521-2537. (Li Kaiwen, Zhang Tao, Wang Rui, *et al.* Research progress on combinatorial optimization based on deep reinforcement learning [J]. Journal of Automation, 2021, 47 (11): 2521-2537.)

[3] 孙志国, 肖硕, 吴毅杰等. 基于迁移学习和参数优化的干扰效能评估方法 [J/OL]. 电子与信息学报, 1-10. (2023-10-30) [2024-01-25]. http://kns. cnki. net/kcms/detail/11. 4494. TN. 20231027. 1505. 009. html. (Sun Zhiguo, Xiao Shuo, Wu Yijie, *et al.* Interference Efficiency Evaluation Method Based on Transfer Learning and Parameter Optimization [J/OL]. Journal of Electronics and Information Science, 1-10. (2023-10-30) [2024-01-25] http://kns. cnki. net/kcms/detail/11. 4494. TN. 20231027. 1505. 009. html.)

[4] 丁增良, 陈珏, 邱禧荷. 一种应用于旅行商问题的莱维飞行转移规则蚁群优化算法 [J]. 计算机应用研究, 2024, 41 (05): 1420-1427. (Ding Zengliang, Chen Jue, Qiu Xihe. An Ant Colony Optimization Algorithm for Levy Flight Transfer Rules Applied to Traveling Salesman Problems [J]. Computer Application Research, 2024, 41 (05): 1420-1427.)

[5] 唐捷凯, 胡蓉, 钱斌等. 混合帝国竞争算法求解带多行程批量配送的多工厂集成调度问题 [J]. 电子学报, 2022, 50 (07): 1621-1630. (Tang Jiekai, Hu Rong, Qian Bin, *et al.* Hybrid Empire Competition Algorithm for Multi factory Integrated Scheduling Problem with Multi itinerary Batch Delivery [J]. Journal of Electronics, 2022, 50 (07): 1621-1630.)

[6] 魏振春, 傅宇, 马仲军等. 带时间窗的无线可充电传感器网络多目标路径规划算法 [J]. 电子学报, 2022, 50 (08): 1819-1829. (Wei Zhenchun, Fu Yu, Ma Zhongjun, *et al.* Multi objective path planning algorithm for wireless rechargeable sensor networks with time windows [J]. Journal of Electronics, 2022, 50 (08): 1819-1829.)

[7] 蔡劲草, 王雷, 雷德明. 基于蛙跳算法的分布式装配混合流水车间调度 [J]. 华中科技大学学报: 自然科学版, 2023, 51 (12): 37-44. (Cai Jincao, Wang Lei, Lei Deming. Distributed Assembly Hybrid Flow Shop Scheduling Based on Frog Jump Algorithm [J]. Journal of Huazhong University of Science and Technology: Natural Science Edition, 2023, 51 (12): 37-44.)

[8] 谢良波, 李宇洋, 王勇等. 基于自适应蝙蝠算法的室内 RFID 定位算法 [J]. 通信学报, 2022, 43 (08): 90-99. (Xie Liangbo, Li Yuyang, Wang Yong, *et al.* Indoor RFID positioning algorithm based on adaptive bat algorithm [J]. Journal of Communications, 2022, 43 (08): 90-99.)

[9] 雷坤, 郭鹏, 王祺欣, 等. 基于 end-to-end 深度强化学习的多车场车辆路径优化 [J]. 计算机应用研究, 2022, 39 (10): 3013-3019. (Lei Kun, Guo Peng, Wang Qixin, *et al.* End-to-end deep reinforcement learning framework for multi-depot vehicle routing problem [J]. Application Research of Computers, 2022, 39 (10): 3013-3019.)

[10] Vinyals O, Fortunato M, Jaitly N. Pointer networks [EB/OL]. (2017-10-02) [2024-2-11]. https://proceedings. neurips. cc/paper_files/paper/2015/file/29921001f2f04bd3baee84a12e98098f-Paper. pdf.

[11] Ma Qiang, Ge Suwen, He Danyang, *et al.* Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning [EB/OL]. (2019-11-12) [2024-2-15]. https://arxiv. org/pdf/1911. 04936. pdf.

[12] Vaswani A, Shazeer N, Parmar N, *et al.* Attention is all you need [EB/OL]. (2017-12-06) [2024-2-23]. https://proceedings. neurips. cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper. pdf.

[13] Kool W, Van Hoof H, Welling M. Attention, learn to solve routing problems! [C]// Proc of International Conference on Learning Representations 2018.

[14] García-Torres R, Macias-Infante A A, Conant-Pablos S E, *et al.* Combining constructive and perturbative deep learning algorithms for the capacitated vehicle routing problem [EB/OL]. (2022-11-25) [2024-1-29]. https://arxiv. org/pdf/2211. 13922. pdf.

[15] Hu Yujiao, Zhang Zhen, Yao Yuan, *et al.* A bidirectional graph neural network for traveling salesman problems on arbitrary symmetric graphs [J]. Engineering Applications of Artificial Intelligence, 2021, 97: 104061.

[16] Zhu Tianyu, Shi Xinli, Xu Xiaoping, *et al.* An accelerated end-to-end method for solving routing problems [J]. Neural Networks, 2023, 164: 535-545.

[17] Drori I, Kharkar A, Sickinger W R, *et al.* Learning to solve combinatorial optimization problems on real-world graphs in linear time [C]// Proc of the 19th IEEE International Conference on Machine Learning and Applications. 2020: 19-24.

[18] Zhao Jiuxia, Mao Minjia, Zhao Xi, *et al.* A hybrid of deep reinforcement learning and local search for the vehicle routing problems [J]. IEEE Trans on Intelligent Transportation Systems, 2020, 22 (11): 7208-7218.

[19] Gao Lei, Chen Mingxiang, Chen Qichang, *et al.* Learn to design the heuristics for vehicle routing problem [EB/OL]. (2020-02-20) . https://arxiv. org/pdf/2002. 08539. pdf.

[20] Scarselli F, Gori M, Tsoi A C, *et al.* The graph neural network model [J]. IEEE Trans on neural networks, 2008, 20 (1): 61-80.

[21] Engstrom L, Ilyas A, Santurkar S, *et al.* Implementation matters in deep policy gradients: A case study on ppo and trpo [EB/OL]. (2020-5-25) [2024-2-16]. https://arxiv. org/pdf/2005. 12729. pdf.

[22] Nazari M, Oroojlooy A, Takáč M, *et al.* Reinforcement learning for solving the vehicle routing problem [C]// Proc of the 32th International Conference on Neural Information Processing Systems. 2018: 9861-9871.

[23] Wu Yaoxin, Song Wen, Cao Zhiguang, *et al.* Learning improvement heuristics for solving routing problems [J]. IEEE Trans on neural networks and learning systems, 2021, 33 (9): 5057-5069.

[24] Peng Bo, Wang Jiahai, Zhang Zizhen. A deep reinforcement learning algorithm using dynamic attention model for vehicle routing problems [C]// Proc of the 11th International Conference on automation, information and Computing. Singapore: Springer Press, 2020: 636-650.

[25] Falkner J K, Schmidt-Thieme L. Learning to Solve Vehicle Routing Problems with Time Windows through Joint Attention [EB/OL]. (2020-6-16) [2024-2-23]. https://arxiv. org/pdf/2006. 09100. pdf.

[26] Lyu Hao, Zhang Xingwen, Yang Shuang. A learning-based iterative method for solving vehicle routing problems [C/OL]/ Proc of International conference on learning representations. 2019. (2019-12-20) [2024-2-27]. https://dblp. uni-trier. de/rec/conf/iclr/LuZY20. html.

[27] Helsgaun K. An effective implementation of the Lin–Kernighan traveling salesman heuristic [J]. European journal of operational research, 2000, 126 (1): 106-130.

[28] Vidal T. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood [J]. Computers & Operations Research, 2022, 140: 105643-105643.

[29] 张景玲，余孟凡，赵燕伟等. 策略梯度的超启发算法求解带容量约束车辆路径问题 [J/OL]. 控制理论与应用, 1-14. (2023-11-24) [2023-12-05]. (Zhang Jingling, Yu Mengfan, Zhao Yanwei, *et al*. Hyperheuristic algorithm based on policy gradient for solving vehicle routing problems with capacity constraints [J/OL]. Control Theory and Applications, 1-14. (2023-11-14) [2023-12-05].)