分类号_____ 密级_____
U D C _____ 编号 10736

西北师范大学

硕士学位论文

（专业学位）

深度强化学习求解软时间窗车辆路径问题的研究与仿真应用

研 究 生 姓 名：龙玉晶
指导教师姓名、职称：代祖华 副教授
实践指导教师姓名、职称：郭真 高级工程师

专 业 学 位 类 别：工程硕士
专 业 学 位 领 域：计算机技术
专 项 计 划：

二〇二三年五月

Research and Simulation Application of Deep Reinforcement Learning for Vehicle Routing Problem with Soft-time Windows

A Thesis Submitted to

Northwest Normal University

in partial fulfillment of the requirement

for the degree of

Master of Electronic Information

by

Long Yujing

Supervisor :Associate Professor Dai Zuhua

Supervisor for Practice Guiding: Senior Engineer Guozhen

May, 2023

摘 要

在城市交通规划中，车辆路径规划问题（Vehicle Routing Problem,VRP）可以帮助政府优化公共交通路线，实现减少交通拥堵、提高城市交通效率等社会功能。在其他领域如电子商务、医疗配送等，VRP 问题也同样具有重要的应用价值。软时间窗约束的车辆路径问题(Vehicle Routing Problem with Soft Time Windows, VRPSTW)是一个非常复杂的组合优化问题，传统的启发式算法往往需要大量的信息资源和计算时间才能找到较优解。而深度强化学习通过学习到更加高效的策略和规律来提高求解效率，从而在实时环境中进行求解并且处理大规模数据，因此可以更加迅速地找到最优质的解决方案。此外，深度强化学习还可以自适应地处理问题及各种不同情形下的实例而不需要重新设计算法，从而更加灵活地应对不同变种的 VRP 问题。因此，选择深度强化学习求解 VRPSTW 问题是非常科学合理且迅捷高效的。本文针对 VRPSTW 问题展开深入系统的研究，构建了两个基于深度强化学习的模型，并在实验层面上验证了算法的有效性。本文的主要工作如下：

（1）提出了一种基于改进注意力机制的 AM 模型求解 VRPSTW 问题。具体来说，本文提出的 AAM 模型运用在实际物流配送问题中，求解更具有现实意义的 VRPSTW 任务，采用基于 Actor-Critic 强化学习算法训练 AAM 模型。在 AAM 模型中，编码器产生所有输入节点的 embedding 信息，解码器中使用注意力机制来产生下一个输入的概率分布，用于选择下一个客户节点。其中编码器包括多头注意力层（Multi-Head Attention，MHA）和全连接前馈子层（Feed-Forward，FF）。解码器中根据一个上下文节点信息来表示当前节点解码的上下文信息。上下文节点主要通过编码器的 embedding 信息，解码器在时间步 t 之前的节点输出信息来加以构造。

（2）提出了一种基于动态注意力的 AAM 模型求解 VRPSTW 问题。为了提高 AAM 模型的求解速度，采用动态注意力机制来优化 AAM 模型，即 AAM-D 模型。在该模型中节点特征是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。为了更好地提取输入序列的信息，在 AAM 模型的基础上改进了编码器中节点更新的方式。在传统的注意力机制中，输入实例的特征经过编码器之后是固定的，不随模型的决策而改变。在 AAM-D 模型中，输入实例的特征表示是动态更新的，可以根据模型在不同构造步骤中的决策进行更

新，从而更好地反映实例的状态、捕捉实例的结构特征。

（3）设计了一个基于深度强化学习的 VRPSTW 实验平台。将第四章构建 AAM-D 模型运用起来，把训练好的模型集成到应用程序中，开发了一个求解 VRPSTW 问题的仿真实验平台。在这个系统中实现了节点数据生成与导出、节点位置可视化、最优路线方案生成、 结果可视化分析与导出等功能，为后面的研究者提供了一种更为方便的可视化实验平台。

关键词：深度强化学习；带软时间窗的车辆路径问题；组合优化问题；演员评论家算法；策略梯度算法

Abstract

In urban traffic planning, Vehicle Routing Problem (VRP) can help the government optimize public transport routes, reduce traffic congestion and improve urban traffic efficiency. In other fields, such as e-commerce and medical distribution, the VRP problem also has important application value. The Vehicle Routing Problem with Soft Time Windows (VRPSTW) is a very complex combinatorial optimization problem, and the traditional heuristic algorithm often needs a lot of computational resources and time to find the optimal solution. Deep reinforcement learning can improve solution efficiency by learning more efficient strategies and rules. It can be solved in real-time environment and handle large-scale data, so it can find better solutions faster. In addition, deep reinforcement learning can deal with problems adaptively and can handle different problem instances without the need to redesign the algorithm, so it can be more flexible to deal with different VRP problems. Therefore, it is very reasonable and effective to select deep reinforcement learning to solve VRPSTW problem. This paper conducts an in-depth study on VRPSTW problem, constructs two models based on deep reinforcement learning, and verifies the effectiveness of the algorithm experimentally. The main work are as follows:

(1) An AM model based on improved attention mechanism is proposed to solve VRPSTW problem. Specifically, our AAM model is applied in practical logistics distribution problems. In solving VRPSTW task with more realistic significance, the AAM model is trained by Actor-Critic based reinforcement learning algorithm. The encoder generates information about all the input nodes and the attention mechanism is used in the decoder to generate the probability distribution of the next input to select the location of the next customer. Encoders include Multi-Head Attention layer (MHA) and fully connected feedforward sublayer (FF). The decoder represents the context information decoded by the current node according to a context node information. The context node is mainly constructed through the embedding information of the encoder and the node output information of the decoder before the time step t.

(2) A AAM model based on dynamic attention is proposed to solve the VRPSTW problem. In order to improve the solving speed of AAM model, dynamic attention mechanism was used to optimize AAM model, namely AAM-D model. In this model,

node features are dynamically updated, which can be updated according to the decisions of the model in different construction steps, so as to better reflect the state of the instance. In order to extract the input sequence information better, the method of node updating in encoder is improved based on AAM model. In the traditional attention mechanism, the characteristics of the input instance are fixed by the encoder and do not change with the decision of the model. In the AAM-D model, the feature representation of the input instance is dynamically updated, which can be updated according to the decision of the model in different construction steps, so as to better reflect the state of the instance and capture the structural characteristics of the instance.

(3) A VRPSTW experimental platform based on deep reinforcement learning is designed. The AAM-D model constructed in Chapter 4 is applied, the trained model is integrated into the application program, and a simulation experiment platform is developed to solve the VRPSTW problem. In this system, node data generation and export, node position visualization, optimal route scheme generation, result visualization analysis and export and other functions are realized, which provides a more convenient visualization experiment platform for later researchers.

Keywords: Deep Reinforcement Learning; Vehicle Routing Problem with Soft-Time Windows; Combinatorial Optimization Problems; Actor-Critic Algorithm; Policy Gradient Algorithm

目　录

第 1 章　绪论

1.1　研究背景和意义

车辆路径问题的应用背景非常广泛，其中涉及到物流配送、城市交通规划、电子商务、医疗配送等多个社会领域[1]。VRP 问题是一种组合优化问题，涉及到配送中的路线规划、资源分配等多个方面[2]。这个问题在应用数学和计算机科学领域已经被研究了几十年，取得了一些具有突破性的成果，因为它与现实生活密切相关，具有重要的应用价值[3]。在物流配送过程中，VRP 问题可以帮助企业优化配送路线，降低生产成本，提高工作效率，最终达到提高客户满意度的目的[4]。在城市交通规划方面，VRP 问题可以帮助政府优化公共交通路线，减少交通拥堵的发生概率[5]，从而在整体上提高城市交通效率[6]。在其他领域如电子商务[7]、医疗配送等[8]，VRP 问题也同样具有重要的应用价值。然而，VRP 问题是一个计算复杂度很高的问题，即使是在简单的情况下也很难找到最优解，因此是组合优化问题中的一个 NP 难问题[9]。

根据求解的约束条件不同，车辆路径问题可以衍生出很多变种问题[10]。基础 VRP 问题是带容量的车辆路径问题，即 CVRP（The Vehicle Routing Problem With Capacity, CVRP）。在基本的 CVRP 问题中，对于每条路上的节点而言，节点的总需求不能超过车辆的负载容量。然而 CVRP 问题在实际的运作过程中由于客户需求的原因和工作任务的限制，要求必须在约定时间内完成车辆配送任务，对于这类具有弹性时间窗约束的 CVRP 问题可以称之为软时间窗约束的车辆路径问题(Vehicle Routing Problem with Soft Time Windows, VRPSTW)[11]。求解 VRPSTW 问题具有重要的现实意义，因为在实际的物流配送中，每个客户节点都有自己规定的派送时间窗口，若配送车辆未在规定时间内到达就会造成客户的不满和经济效益的损失，在这种情况下需要产生一定的惩罚后果。因此，VRPSTW 问题更符合实际情况，相应地也更具有实际应用价值。同时，VRPSTW 问题更具有挑战性，因为时间窗口约束增加了问题的复杂度，需要更高效的算法和更深入的研究来解决 VRPSWT 问题。因此，VRPSTW 问题的研究具有重要的理论价值和实践意义。

近年来深度强化学习的发展非常迅速[12]，其主要特点包括可以处理更加复杂的问题、自适应地处理问题、提高求解效率、处理实时问题和处理大规模数据等。这些特点使得深度强化学习在求解复杂的优化问题方面具有显著优势[13]。VRPSTW 问题是一个非常复杂的组合优化问题，传统的启发式算法往往需要大量

的计算资源和计算时间才能找到较优解[14]。而深度强化学习可以通过学习到更加高效的策略和运算规律来提高求解效率，可以在实时环境中进行求解并处理大规模数据，因此可以在更短时间内迅速找到更为优化的解决方案。此外，深度强化学习可以自适应地根据不同具体情况的实例处理问题而不需要重新设计算法，因此能做到更加灵活地应对不同的VRP问题。因此，选择深度强化学习求解VRPSTW问题是科学合理和精确高效的。

1.2 国内外研究现状分析

求解 VRPSTW 问题可以分为传统算法和深度强化学习算法这两大类[15]。

传统求解 VRPSTW 问题的算法包括：精确解算法、启发式算法和元启发式算法。精确解算法可以有效保证找到最优解，但计算复杂度很高，只适用于小规模问题。常用的精确解算法求解 VRP 问题包括分支定界法、动态规划法、割平面法等。启发式算法是一种近似求解方法，可以在较短的时间内找到较优解。常见的启发式算法求解 VRP 问题包括模拟退火算法、遗传算法、禁忌搜索算法等。元启发式算法结合了多种启发式算法，可以在不同的阶段使用不同的启发式算法，从而提高求解效率和求解质量。常见的元启发式算法包括混合启发式算法、多目标启发式算法等。

深度强化学习算法结合了深度神经网络和强化学习的特点，可以通过学习到更加高效的策略和运算规律来提高求解效率，并且在多变的实时环境中进行自适应求解，再次基础之上处理大规模数据，因此可以在更段时间内迅速找到更为优化的解决方案。事实证明近年来，深度强化学习在求解 VRP 问题方面取得了很好的效果[14]。

因此，按照这两类算法对国内外的研究现状分析如下。

1.2.1 基于传统算法求解 VRPSTW 国内外研究现状分析

求解 VRPSTW 问题的传统算法有精确解算法[15]、启发式算法[16]和元启发式算法[17]。精确算法可以得到问题的最优解，然而随着问题规模的扩大，计算复杂度也呈指数增长，导致问题无法在合理的时间内解决。在 VRPSTW 问题中可以通过启发式算法设计具有专业知识的手工特征来解决问题，因此启发式算法在实践中被广泛采用。局部搜索方法，通常称为邻域搜索（neighborhood search），是启发式算法的重要类别[18]，从初始解开始，局部搜索方法通过在初始解附近的空间中搜索来提高解的质量。在原始设计的基础上衍生出多种启发式算法，包括模拟退火[19]、

禁忌搜索[20]和 LNS[21]。本地搜索算法已被应用于不同变体的 VRP 问题，例如带时间窗的车辆路径问题[22]、拆分交付 VRP 问题[23]和电动 VRP 问题[24]。然而，当邻域空间中的解决方案没有改进时，搜索过程停止而导致算法陷入局部最优[25]。2019 年 Gutierrez 等人[26]提出了混合元启发式算法对 VRPSD 问题进行求解，该算法结合了启发式算法中的贪婪随机算法以及文化基因算法，实验表明与现有元启发式算法相比，该混合元启发式算法求解 VRPSD 问题时具有显著优势。

Marinakis 等人[27]对具有软时间窗的车辆路径问题进行了研究，进一步考虑到具有随机需求的客户节点，在此基础上设计了一种基于粒子群算法的混合方法来求解。除此之外，李阳等人[28] 针对 VRPSTW 问题提出了一种两阶段的混合变邻域分散搜索算法。实验表明，该算法的优点是求解稳定、平均误差小，但在求解大规模问题时算法的能力较差，耗时较长。Xu 等人[29]根据 VRPTW 问题特点，提出结合混合遗传算法和粒子群算法的方法，在路径解码阶段中利用粒子实数编码以减轻计算负担，同时为了避免求解结果陷入局部最优，算法利用了遗传算法中交叉算子的思想。Masrom 在求解 VRPTW 问题时考虑到时间窗约束对问题的影响，提出了基于膜计算的混合进化算法[30]，该混合进化算法将 GA 算法中的二进制编码改进为整数编码，避免了编码冗余等问题，能够提高算法的计算效率和实用性。戚远航等人[31]针对 VRPTW 问题设计了离散蝙蝠算法，利用离散蝙蝠算法鲁棒性好、寻优能力强等特性来进行求解。

综上所述，传统算法求解 VRPSTW 问题的不足有以下几点： 计算复杂度高，VRPSTW 问题是一个 NP-hard 问题，传统算法在求解大规模问题时需要耗费大量的计算资源和计算时间，难以在实际应用中得到广泛应用；求解质量难以保证，传统算法通常采用启发式方法求解，虽然可以在较短时间内找到较优解，但是无法保证找到全局最优解，所以求解质量难以得到保证；对问题的假设较多，传统算法通常对问题做出一些假设，如车辆的容量、路线的长度等，这些假设可能与实际情况不符，导致求解结果准确率不高；难以处理实时变化的问题，传统算法通常需要预先知道所有的问题参数，无法处理实时变化的问题，如客户需求的变化、交通状况的变化等；难以处理多目标问题，VRP 问题通常涉及多个目标，如最小化总路程、最小化总成本等，传统算法难以同时优化多个目标，需要进行权衡和取舍。

综上所述，传统算法在求解 VRP 问题时存在一些局限性和缺陷，需要进一步完善和改进。

1.2.2 基于深度强化学习算法求解 VRPSTW 国内外研究现状分析

近年来，深度强化学习在求解 VRPTW 问题方面取得了一些进展，目前已有一些使用端到端神经网络模型直接从数据中学习、求解组合优化问题的相关工作[14]。最开始采用神经网络求解组合优化问题的是 Bello 等人运用神经网络模型解决小实例的 TSP 问题[32]。最近，Sutskever 等人基于序列到序列模型[34]，针对 TSP 问题提出了使用监督学习训练的指针网络模型。Vinyals 等人利用指针网络来解决 TSP 并用强化学习技术训练网络[35]。与传统算法相比，实验证明了它在背包问题和 TSP 中的可扩展性。

在组合优化问题中，采用监督学习的方法通常需要为最优解生成标签，然而这些标签在实际问题中很难获得。不同于监督学习，深度强化学习是从奖励信号中学习而不需要提前标注。2020 年 Sheng 等人[37]提出指针网络（Ptr-Net）模型求解旅行商问题（Travel Salesman Problem，TSP），该指针网络模型相较于传统的启发式算法而言，在小规模实施的 TSP 问题上可以做到求解速度更快，这也是深度学习在组合优化问题上的首次应用。旅行商问题可以看成是车辆路径问题的简单形式，之后研究人员开始将指针网络（Ptr-Net）模型应用于 CVRP 的求解上。Kool 等人[39]针对 VRP 问题，提出了基于深度策略动态规划方法来进行求解，该方法结合了神经网络和动态规划思想并取得了一定成就。Kool 提出的该模型基于图神经网络（graph neural network，GNN）技术特点，采用 GNN 获得每个客户节点的特征向量，并采用注意力机制来计算每个客户节点被选中的概率，然后用动态规划算法得到概率结果，作为对部分解选择的策略，最后模型根据该策略构造最优解决方案。Bdeir 等人[40]基于深度 Q 网络（Deep Q-Network，DQN），提出 RP-DQN 模型求解 MDVRP 问题。RP-DQN 模型能够有效降低 MDVRP 问题计算的复杂性，RP-DQN 模型的编码器对静态客户节点信息进行编码，RP-DQN 模型的解码器对 MDVRP 问题中的动态节点特征信息进行编码，最后使用 Q-learning 强化学习算法对 RP-DQN 模型进行优化。实验表明在不同规模的实例中，RP-DQN 模型在 MDVRP 客户节点数量为 20、50、100 时，模型的优化效果均超过了 Kool 等人[39]的方法。

Chen 等人[42]针对车辆和无人机当天交付的问题（same-day delivery problem with vehicles and drones，SDDPVD），采用了与 Powell 相同的模型架构，并使用 Ulmer 等人[43]提出的路由策略来模拟路线上节点的更新和演变。在该模型中，将 SDDPVD 问题建模为马尔可夫决策过程，简化了 SDDPVD 问题的动作空间和状态空间，采用了 DQN 算法近似每个节点特征向量的值。Nazari 等人[44]针对 CVRP 问题构建了指针网络的强化学习模型 PtrNet-RL，该模型采用了端到端的深度学习思想，使用 REINFORCE 强化学习算法对模型进行训练。在构建指针网络阶段，

Ptr-Net 中的输入的信息为静态值与动态值。该 PtrNet-RL 模型不同于指针网络在训练模型时采用监督式方法，PtrNet-RL 模型使用 REINFORCE 算法对模型进行训练，分别在客户节点数为 10、20、50、100 的 CVRP 数据集上进行了实验，取得了比经典的启发式算法更好的效果。为有效解决车辆路径问题中的供需匹配难题，Zhang 等人[45]提出了由 Qrewriter 模块和 DDQN 模块构成 QRewriter-DDQN 模型。在该模型中先将车辆提前调度给需求级别高的客户，其中 DDQN 模块负责车辆和客户，将二者的 KL 散度分布作为激励函数，以此得到车辆和客户供需之间的动态变化，然后再运用 Qrewriter 模块改进 DDQN 的调度策略。为解决深度强化学习算法在构造解的过程中，无法修改之前决策这一问题，Xin 等人[46]针对 CVRP 问题，提出基于 REINFORCE 强化学习算法的模型，该模型采用分步思想的 SW-Ptr-Net 模型和近似思想 SW-AM 模型，有效提升了指针网络 Ptr-Net 模型和基于注意力机制的 AM 模型[48]，在求解 CVRP 问题上的优化效果。

另外，Kool 等人首次将 transformer 模型求解 CVRP 问题，transformer 的结构与大多数的 seq2seq 模型一样是由编码器和解码器组成的。在编码器中没有使用 transformer 模型的位置编码，从而使得客户节点的嵌入信息不受输入顺序影响。在解码器中，为提高模型的计算效率而使用了一个自注意力机制，最后模型采用 REINFORCE 强化学习算法对模型进行训练。通过在 CVRP 和 SDVRP 问题上的实验表明该模型取得了比指针网络模型更好的效果，而且求解效果与经典的运筹学求解器 LKH3 和 Gurobi 相差无几。Falkner 等人[47]针对 CVRPTW 问题，提出了由多个编码器和一个解码器组成的 JAMPR 模型，其中编码器采用自注意力机制，通过加入两个新的编码器产生上下文信息，并且增强了联合行动空间，解码器中使用 transform 模型中的多头注意力机制。在对 CVRPTW 问题的 3 种变体的实验中，JAMPR 模型的优化效果要优于现有的元启发式算法和 PtrNet-RL 模型。为有效地解决多车辆软时间窗的多车辆路径问题（multi- vehicle routing problem with soft time windows，MVRPSTW），Zhang 等人[50]提出了多智能体注意力模型（multi-agent attention model，MAAM），MAAM 模型使用 REINFORCE 算法对进行训练。实验结果表明，MAAM 模型的求解速度优于 Google OR-tools 求解器和传统的启发式算法。除此之外，Zhang 等人[49] 以 AM 模型为基础构建了一种具有多重关系的 MRAM 模型，该 MRAM 能够更好地获取车辆路径问题的动态特征。

传统的启发式算法往往需要大量的计算资源和计算时间才能找到较优解，而深度强化学习可以通过学习到更加高效的策略和规律来提高求解效率，可以在实时环境中进行求解并处理大规模数据，因此可以在更短时间内找到更为优质的解决方案。因此，选择深度强化学习求解 VRPSTW 问题是科学合理且迅捷高效的。

1.3 本文研究内容及论文组织结构

1.3.1 本文研究内容

本文的主要研究工作是基于深度强化学习框架求解 VRPSTW 问题。采用指针网络和注意力机制构建模型、使用 Actor-Critic 算法训练模型，通过使用策略参数中的预期回报梯度，来估计来改进策略、优化其参数，训练后的模型可以在实时中以一系列连续的动作产生解决方案。本文的主要研究工作如下：

（1）提出一种基于改进注意力机制的 AM 模型求解 VRPSTW 问题。具体来说，AAM 模型运用在实际物流配送问题中，求解更具有现实意义的 VRPSTW 任务中，采用基于 Actor-Critic 强化学习算法求解 VRPSTW 的 AAM 模型。编码器产生所有输入节点的 embedding 信息，解码器中使用注意力机制来产生下一个输入的概率分布，以选择下一个客户的位置。其中编码器包括多头注意力层（Multi-Head Attention，MHA）和全连接子层（Feed-Forward，FF）。解码器中根据一个上下文节点信息来表示当前节点解码的上下文信息。上下文节点主要通过编码器的 embedding 信息，解码器在时间步 t 之前的节点输出信息来加以构造。

（2）提出一种基于动态注意力的 AAM-D 模型求解 VRPSTW 问题。为了提高 AAM 模型的求解速度，采用动态注意力机制来优化 AAM 模型，在该模型中节点特征是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。为了更好地提取输入序列的信息，在 AAM 模型的基础上改进了编码器器中节点更新的方式。传统的注意力机制中，输入实例的特征经过编码器之后是固定的，不随模型的决策而改变。在 AAM-D 模型中，输入实例的特征表示是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态、捕捉实例的结构特征。

（3）设计一个基于深度强化学习的 VRPSTW 实验平台。将第四章构建 AAM-D 模型运用起来，将训练好的模型集成到应用程序中，开发了一个求解 VRPSTW 问题的仿真系统。在这个系统实现了节点数据生成与导出、节点位置可视化、最优路线方案生成、 结果可视化分析与导出等功能，为后面的研究者提供了一种更为方便的可视化实验平台。

1.3.2 论文组织结构

本文从 VRPSTW 问题的研究背景和研究现状出发，重点针对基于深度强化学习算法开展了深入的学习研究。论文结构安排具体如下：

第 1 章 绪论。本章中介绍了 VRPSTW 问题的研究背景和意义，对该领域的研究现状进行了分类总结并加以分析概括，最后阐述了本论文的主要研究工作和结构安排。

第 2 章 相关理论与技术。本章介绍了 VRPSTW 问题的定义、深度强化学习的相关理论知识、用到的深度强化学习算法以及基于 Encoder-Decoder 框架的 AM 模型，为后面的研究奠定了理论依据。

第 3 章 构建基于改进注意力机制的 AM 模型求解 VRPSTW 问题。本章首先概述了所提出的 AAM 模型的各个模块，然后详细介绍了 AAM 模型中各个模块的基本原理和计算公式，最后阐述了 AAM 模型求解 VRPSTW 的实验以及结果分析。

第 4 章 构建基于动态注意力的 AAM-D 模型求解 VRPSTW 问题。首先介绍了 AAM-D 模型的总体框架，以及 AAM-D 模型各个模块的作用、基本原理和计算公式，最后阐述了 AAM-D 模型求解 VRPSTW 的实验以及结果分析。

第 5 章 基于深度强化学习的 VRPSTW 实验平台设计与实现。将训练好的 AAM-D 模型集成到应用程序中，设计并实现了一个 VRPSTW 的仿真系统。在这个系统实现了节点数据生成与导出、节点位置可视化、最优路线方案生成与导出等功能，为后面的研究者提供了一种更为方便的可视化实验平台。

结 论 对文章的研究工作以及所提出的模型进行了总结，并针对模型的不足之处，对未来的研究工作进行了展望。

第 2 章　相关理论与技术

2.1 软时间窗的车辆路径问题的定义

假设有图$G=(V,E)$，其中节点集$V=\{v_0,v_1,...,v_n\}$表示客户和仓库，边集$E\subset V\times V$表示连接客户和仓库的路径，用$i=0$ 表示仓库节点，其余的则表示客户节点。从仓库出发有 m 辆车来完成个 n 个客户的货物交付，每个客户节点$i\in V$有以下特征：位置(x_i,y_i)，需求订单的权重q_i以及每个节点的时间窗约$TW(t_{e,i},t_{l,i})$，其中$t_{e,i}<t_{l,i}$。假设所有车辆有相同的容量和速度，所有的车辆要求在仓库开始和结束他们的路线；图中任意节点之间都有路径可达，边的权重为该边两个节点之间的距离$d_{i,j}$。

除此条件之外，VRPSTW 问题还需要满足以下约束条件：每个客户节点的需求不可拆分约束，即每个客户节点只能由一辆车提供服务，若该车辆中剩余的容量不满足客户节点需求，则分配其他车辆；容量约束，则表示当前该车辆的行驶路径上的客户节点，节点的总需求量不能超过车辆规定的最大容量，若该路径上的节点需求达到车辆容量约束，则表示该路径规划完成；车辆行驶约束指的是要求车辆从仓库出发，对所有客户节点服务完后必须回到仓库。

图 2-1 为$m=3,n=10$ 时的一个 VRPSTW 示例。

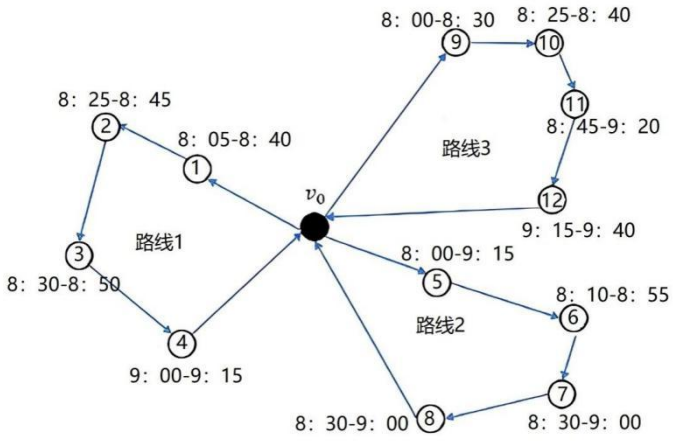


图 2-1 VRPSTW 实例

VRPSTW 的求解任务是获得代价最小且满足客户时间要求的车辆最优运行路径，m辆车对分布在不同位置的n个客户进行服务。由于每辆车都有一条独特的路线，因此将产生总数为 M 的路线，它们只在仓库相互连接。所有的距离都用平面上的欧几里得距离表示，所有车辆的速度都被假定为相同的（即行驶一单位的距离需要一单位的时间）。求解方案是找到一个总成本最小的解决方案$r[1,m]=$

$(r[1], r[2], \ldots, r[m])$，那么目标则是要求车辆在服务客户节点的过程中，找到一条花费成本最小且违背时间约束下总惩罚最小的一个顶点不相交的路线。因此构建目标函数如下：

$$Cost(r[1,m]) = d_{sum}(r[1,m]) + p_{sum}(r[1,m]) \quad (2\text{-}1)$$

其中$d_{sum}(r[1,m])$是所有车辆的总行驶距离，具体表达如下：

$$d_{sum}(r[1,m]) = \sum\sum ||r[m][b], r[m][b+1]||_2 \quad (2\text{-}2)$$

其中$b \in V$，$b = \{1,2,\ldots,n\}$, 表示客户节点，$b+1$ 代表 b 节点中的下一个点。$r[m][b]$表示车辆m 到节点 b 的位置，$r[m][b+1]$表示车辆m到节点 b 的下一个位置。$\sum\sum ||r[m][b], r[m][b+1]||_2$ 表示所有车辆 m 对应的路径方案上节点的欧氏距离之和。

其中$p_{sum}(r[1,m])$表示对应路线方案节点时间窗口约束的总惩罚，具体表达如公式（2-3）所示：

$$p_{sum}(r[1,m]) = \sum\sum \left[I_{（t_{e,i} > t_{a,i}）} + I_{（t_{l,i} > t_{a,i}）} \right] \quad (2\text{-}3)$$

其中，$t_{e,i}$是客户 i 要求的到达时间，$t_{a,i}$表示车辆为客户 i 服务的时间，$t_{l,i}$是客户 i 要求的结束时间。其中，$I(t)$函数是该惩罚到达时间的一个线性惩罚函数，α_i是车辆在节点i处的早到惩罚系数，β_i是车辆在节点i处的晚到惩罚系数。在 VRPSTW 中，提前到达任何一个客户都被认为是早到惩罚，通常比晚到惩罚小很多。

早到惩罚函数表达如下：

$$I_{（t_{e,i} > t_{a,i}）} = I(t_{e,i} > t_{a,i}) * \alpha_i * (t_{e,i} - t_{a,i}) \quad (2\text{-}4)$$

晚到惩罚函数表达如下：

$$I_{（t_{l,i} > t_{a,i}）} = I(t_{l,i} > t_{a,i}) * \beta_i * (t_{l,i} - t_{a,i}) \quad (2\text{-}5)$$

表 2-1 是 VRPSTW 问题中符号的详细信息表。

表 2-1 VRPSTW 问题定义符号的详细信息

符号	说明
V	节点集合
V_0	仓库
i	客户 i
n	客户的数量
(x_i, y_i)	客户 i 的坐标
q_i	客户的需求

续表 2-1 VRPSTW 问题定义符号的详细信息

符号	说明
TW	时间约束
$t_{e,i}$	客户 i 要求的到达时间
$t_{l,i}$	客户 i 要求的结束时间
$t_{a,l}$	当前车辆到达 i 处的时间，即服务时间
E	边集
$d_{i,j}$	节点之间的距离
Q	车辆 j 的最大载重量
$Q^{'}$	车辆 j 当前的载重量
m	车辆的数量
π	解序列
L	解序列的总长度
t	当前的时间
α_i	车辆早到节点 i 约束时间的早到惩罚系数
β_i	车辆晚到节点 i 约束时间的晚到惩罚系数

2.2 深度强化学习原理

2.2.1 深度强化学习的组成元素

机器学习属于人工智能的子领域，深度学习、强化学习和深度强化学习都属于机器学习的下一级概念。深度学习是一种根据模型分类的方法，特指使用神经网络作为参数结构的模型。神经网络一般是多层的，先将底层数据转化成高层特征，再通过机器学习自身来产生好特征，从而使得机器学习向全自动的一个数据分析又迈近了一步。但是其转化过程难以推导，因此可解释性差。深度学习提供的是一种感知能力，通过神经网络来模仿人脑的机制从而抽象出一些数据特征。

强化学习通过任务分类的方法，强调与环境交互和基于环境而变动而取得最大化的预期收益。在具体任务中先给强化学习一个奖励函数，它的训练不需要标签，通过环境给出的奖惩来进行学习。强化学习最后得到的结果不是一个模型，

而是具有决策能力的智能体，此智能体的特点是不断试错与延迟奖励。

DRL 结合了深度学习和强化学习的特点，将深度学习的感知能力和强化学习的决策能力相结合用于解决复杂的决策问题。在深度强化学习中智能体能够通过与环境的交互来学习最优策略。在实际应用中，深度学习用于处理输入数据提取有用的特征，强化学习则用于指导智能体如何在环境中采取行动，并根据行动的结果来调整其策略。在具体的任务中，一般采用深度学习模型作为框架架构，采用强化学习作为决策步骤，模型的训练也常用到强化学习的算法。

在深度强化学习中有以下几个重要的组成元素： 策略，智能体在环境中采取行动的方式，可以是确定性的或随机的； 策略网络，用于根据当前状态输出智能体采取行动的概率分布，可以是基于值函数的策略网络或基于策略梯度的策略网络；值函数网络，用于评估智能体在某个状态下采取行动的好坏，可以是基于状态值函数的值函数网络或基于动作值函数的值函数网络； 训练算法，用于优化策略网络和值函数网络的参数，以使智能体在环境中表现更好。常用的训练算法包括 Q-learning、Actor-Critic、Policy Gradient 等；经验回放缓存，用于存储智能体与环境的交互经验，以便训练算法可以从中随机抽取样本进行训练，提高训练效率和稳定性；目标网络，用于稳定训练过程，将策略网络和值函数网络的参数从当前状态复制到目标网络中，以减少训练过程中的参数更新对目标函数的影响。

2.2.2 深度强化学习的基本架构

深度强化学习基本架构的主要目的是为了解决强化学习问题，如游戏玩法、机器人控制、自然语言处理等，用于让智能体在与环境的交互中学习如何做出最优的决策。这些问题通常需要模型能够在不断变化的环境中做出决策，从而得到最大化的预期奖励。

如图 2-2 所示，深度强化学习的基本架构包括以下几个部分：环境、状态、行动、奖励、智能体。

智能体（Agent）是指一个能够感知环境、做出决策并执行行动的实体。智能体的目标是通过与环境的交互来最大化累积奖励，从而学习如何做出最优的决策。环境（Enviroment）是智能体与之交互的环境，可以是真实世界或模拟环境。状态（State）是环境的某个特定时刻的状态，可以是离散的或连续的。行动（Action）是智能体在某个状态下采取的行动，可以是离散的或连续的。 奖励（Reward）是智能体在某个状态下采取某个行动后获得的奖励，可以是正数、负数或零。

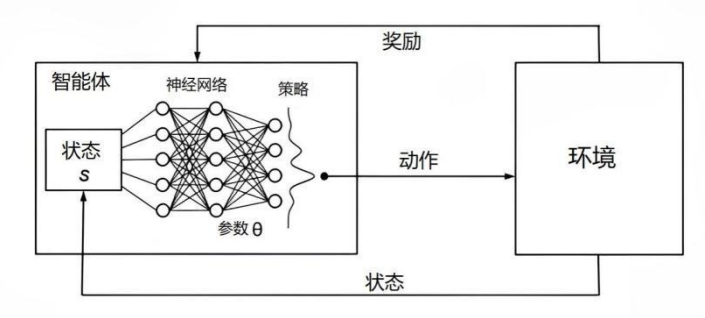


图 2-2 深度强化学习的基本架构

智能体（Agent）是指一个能够感知环境、做出决策并执行行动的实体。智能体的目标是通过与环境的交互来最大化累积奖励，从而学习如何做出最优的决策。环境（Enviroment）是智能体与之交互的环境，可以是真实世界或模拟环境。状态（State）是环境的某个特定时刻的状态，可以是离散的或连续的。行动（Action）是智能体在某个状态下采取的行动，可以是离散的或连续的。 奖励（Reward）是智能体在某个状态下采取某个行动后获得的奖励，可以是正数、负数或零。

2.2.3 马尔可夫决策过程

马尔可夫决策过程（Markov Decision Process，MDP）是深度强化学习中求解具体任务的理论依据，也是非常重要的数学模型。

在深度强化学习中，智能体需要在一个环境中做出一系列决策，以最大化长期累积奖励。MDP 提供了一种形式化的方法来描述这个过程，包括状态、行动、转移概率、奖励和策略等要素用来描述智能体在环境中做出决策的过程，MDP 主要由$\langle S, A, R, P, \gamma \rangle$构成。

S:状态集合，$s_t \epsilon S$为智能体在当前t时刻所处的状态;

A:动作集合，$a_t \epsilon A$为智能体在当前t时刻所采取的动作;

R:回报函数，代表当前状态下采取的动作所获得的奖励;

P: $S \times A \times S \to R$，表示状态转移矩阵;

γ:折扣因子，用来平衡即时回报与长期回报的重要程度。

在深度强化学习中，MDP 被广泛应用于强化学习算法的设计和优化。深度强化学习算法通常使用神经网络来表示策略或值函数，通过学习从状态到行动的映射来实现最优决策。MDP 提供了一种形式化的方法来描述这个映射关系，使得深度强化学习算法可以更加有效地学习最优策略。

此外，MDP 还提供一种用于评估 DRL 算法性能的方法，通过计算 Agent 在环境中的长期累积奖励，以此来评估算法的性能。这种评估方法可以帮助研究人员比较不同算法的性能并优化算法的设计。总之，MDP 在深度强化学习中起着非常重

要的作用，它提供了一种形式化的方法来描述 Agent 在环境中做出决策的过程，可以帮助 DRL 算法更有效地学习最优策略。

如图 2-3 所示，是 MDP 中智能体与环境交互的过程。

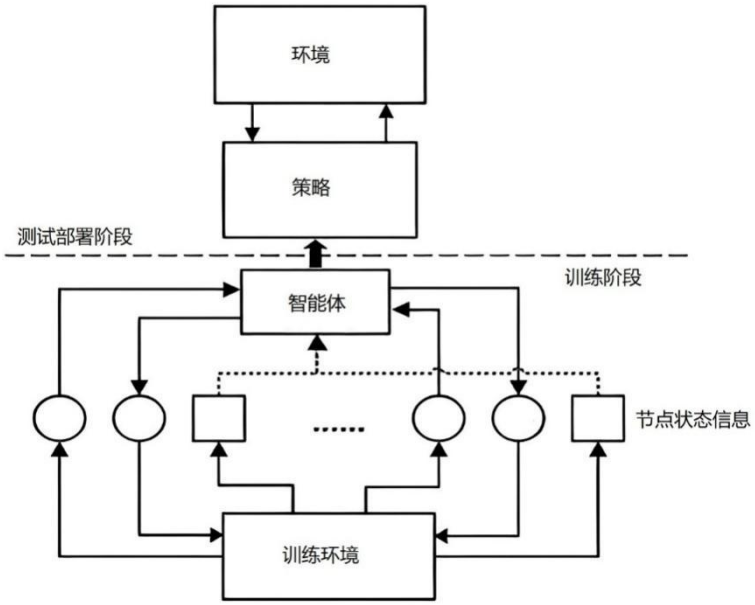


图 2-3 MDP 中智能体与环境交互的过程

马尔可夫决策过程可以用于求解车辆路径问题，对于解决 VRPSTW 问题具有重要意义。VRPSTW 涉及在有限数量的车辆和客户之间分配任务，以最小化总路程或总成本。求解 VRPSTW 问题是一个 NP 难问题，采用传统的优化算法求解 VRPSTW 问题需要耗费大量的计算时间和计算资源，MDP 提供了一种新的求解 VRPSTW 问题的方法。在 MDP 框架下，VRPSTW 问题可以被视为一个序列决策问题，智能体需要在每个时间步骤中选择一个动作，以最大化长期累积奖励。通过使用强化学习算法，可以训练一个智能体来学习如何做出最优决策，从而解决 VRPSTW 问题。

使用 MDP 求解 VRP 问题时,可以处理不同的 VRP 实例：传统的优化算法需要为每个 VRP 实例重新设计和优化算法，而 MDP 方法可以处理不同的 VRP 实例，只需要训练一个智能体即可。除此之外， MDP 方法可以自适应地处理问题变化，并处理大规模的 VRPSTW 问题，只需要重新计算奖励和验证解决方案的可行性即可。

在 MDP 中，智能体与环境交互的过程可以分为以下几个步骤：首先，在每个时间步骤，环境会向智能体提供当前的状态信息，包括智能体的所在位置、周围的环境信息等。其次，智能体根据当前的状态信息选择一个动作来执行。动作可以是向左或向右移动，或是进行其他操作。当智能体执行动作后，环境会根据智

能体的行为给出一个奖励信号，表示智能体的行为是正确的还是错误的，奖励可以是正数、负数或零。接着，环境根据智能体选择的动作，更新当前的状态信息，并返回给智能体。在某些情况下智能体执行动作后会到达一个终止状态，表示任务已经完成或者无法继续执行。在这种情况下智能体将不再执行动作。

智能体与环境交互的过程是一个不断迭代的过程，智能体根据当前环境的状态信息选择动作，环境根据智能体的行为给出奖励信号并更新状态信息，直到任务完成或者无法继续执行。Agent 的目标是通过不断地与环境进行交互，来学习到一个最优策略，使得在决策过程中累积的奖励以最大化。

2.3 深度强化学习算法方法

2.3.1 基于值函数的方法

基于值函数的方法（Value-based Methods）是强化学习中的一类方法，它的主要思想是通过学习一个值函数来指导智能体的决策。值函数表示在当前状态下，智能体采取某个动作后用于计算获得的长期累积奖励。基于值函数的方法包括 Q-learning、SARSA 等算法，这些算法通过更新值函数的估计值来指导智能体的决策。基于值函数的方法通常更适用于离散状态和动作空间，更容易处理确定性策略。

基于值函数的强化学习方法包括状态表示、值函数定义、值函数更新、策略改进这几个重要的方面。状态表示指将环境状态表示为一个向量或矩阵，以便于计算值函数。值函数通常包括状态价值函数和动作价值函数。状态价值函数表示在当前状态下智能体可以获得的长期累积奖励的期望值；动作价值函数表示在当前状态下采取某个动作可以获得的长期累积奖励的期望值。 值函数的更新使用强化学习算法，如 Q-learning、SARSA 算法等，这类算法是更新值函数的估计值，通常使用贝尔曼方程来更新值函数。 策略改进是根据更新后的值函数改进 Agent 的策略，使 Agent 能够更好地利用环境信息来获得更高的奖励。

值函数的方法在强化学习中得到了广泛的应用，例如在机器人控制、游戏智能等领域。优点是可以处理连续状态和动作空间，同时可以通过值函数的估计来指导智能体的决策，从而获得更好的性能和体验感。

2.3.2 基于策略函数的方法

基于价值的求解算法(如值函数近似法)在实际应用中存在一些不足，如算法难

以高效处理连续动作空间的任务，最终的求解结果不一定是全局最优解等。而基于策略函数的方法（Policy-based Methods）是强化学习中的一类方法，也称策略梯度法，它的主要思想是通过学习一个策略函数来指导智能体的决策。策略函数表示在当前状态下，智能体应该采取哪个动作来获得最大的长期累积奖励。基于策略函数的方法通常更适用于连续状态和动作空间，通常更容易处理随机策略。

策略梯度算法采用函数近似的方法构建策略网络，通过策略网络来选取动作，然后得到奖励值，参数的优化是沿着梯度方向进行，得到优化后的策略用于最大化奖励值。基于策略梯度的算法之中策略的学习目标是策略函数$\pi(a|s)$，通过求解策略函数$\pi(a|s)$的极大值以此来最优化策略π。这种方法使得 Agent 能够在不考虑价值函数的情况下直接在环境中选择动作，能够有效解决基于值函数强化学习方法求解问题时的缺点。

2.3.3 Actor-Critic 算法

基于值函数的方法和基于策略函数的方法都是 RL 中常用的方法，它们各有优缺点。基于值函数的方法可以处理大规模的状态空间和动作空间，处理确定性策略，可以通过值函数的估计值来指导智能体的决策。但是对于连续状态和动作空间需要进行离散化处理，这可能会导致信息损失，此外对于随机策略需要进行额外的处理。基于策略函数的方法可以处理连续状态和动作空间、处理随机策略，可以通过策略函数的估计值来指导智能体的决策。但是对于大规模的状态空间和动作空间则计算代价较高，可能会出现收敛到局部最优解的问题，需要进行更多的参数调整。

演员-评论家（Actor-Critic）算法是一种基于策略梯度的强化学习算法，用于解决连续动作空间的问题，它结合了策略评估和策略改进两个方面，可以同时学习策略和价值函数来提高学习效率。

在模型中，Actor-Critic 算法通常被用来解决连续动作空间和非线性策略的问题，因为它可以处理高维状态和动作空间，并且可以学习到更加复杂的策略。

Actor-Critic 算法包含两个网络：Actor 网络和 Critic 网络。Actor 网络负责学习策略，它输出动作的概率分布并根据这个概率分布选择动作。Critic 网络负责学习价值函数，它估计每个状态的价值，可以帮助 Actor 网络更好地选择动作。

在模型的训练过程中，Actor 网络的参数一般采用策略梯度方法进行更新，Critic 网络的参数一般通过 TD 误差进行更新。Actor-Critic 算法在模型中的作用是提高强化学习的效率和稳定性。由于 Actor-Critic 算法可以同时学习策略和价值函数，因此可以减少策略搜索的时间和次数，提高学习效率。此外，Critic 网络可以

提供更加准确的价值估计，帮助 Actor 网络更好地选择动作，从而提高学习稳定性。因此，Actor-Critic 算法在强化学习中被广泛应用，它能够结合基于值函数方法和基于策略函数方法的思想，在处理非线性策略的问题和连续动作空间时具有优势。

在 VRPSTW 问题中，需要找到一条最优路径，使得一辆车能够在满足约束条件的情况下依次访问多个客户节点并返回起点。 Actor-Critic 算法是一种基于值函数和策略函数的混合算法，它可以同时学习策略和值函数，从而提高学习效率和稳定性。在 VRPSTW 问题中，Actor-Critic 算法可以通过学习一个策略函数来生成一条路径，同时学习一个值函数来评估这条路径的质量。

Actor-Critic 算法中的 Actor 表示策略函数，用于生成路径；Critic 表示值函数，用于评估路径的质量。在 VRPSTW 问题中，Actor-Critic 算法可以通过学习一个策略函数来生成一条路径，同时学习一个值函数来评估这条路径的质量。Actor-Critic 算法可以通过优化策略函数和值函数来最大化总体奖励，从而找到最优路径。

相比于其他求解 VRPSTW 问题的算法，Actor-Critic 算法可以处理大规模的状态空间和动作空间，适用于 VRPSTW 问题求解大规模节点实例问题，除此之外，与传统启发式算法相比，AC 算法能够通过策略函数的估计值来指导智能体的决策。因此，Actor-Critic 算法在求解 VRPSTW 问题中具有不可替代的优势。

2.4 本章小结

本章阐述使用深度强化学习求解 VRPSTW 问题的理论技术及其原理，详细介绍了 VRPSTW 问题的定义以及深度强化学习的原理，包括深度强化学习的组成元素和框架，以及使用 MDP 求解 VRPSTW 问题的理论支持。并介绍了 DRL 相关算法，包括基于值函数的方法、基于策略函数的方法以及 Actor-Critic 算法。

第 3 章　基于改进注意力机制的 AM 模型求解 VRPSTW 问题

近年来，随着指针网络在组合优化问题上的成功[35]，基于机器学习求解组合优化问题求解模型逐渐被学术界关注[13]。在这些方法中基于深度强化学习的组合优化求解带来了较高的模型准确度和泛化性。Kool 等人基于 Transformer 模型提出基于注意力层的模型（Attention Model，简称 AM）[48]，采用 REINFORCE 方法训练模型来求解组合优化问题，取得了一定程度上的进展。

AM 模型在求解组合优化问题时还存在一些不足之处：AM 模型可以用于求解简单的 CVRP 问题，而现实中碰到的 VRP 问题往往带有多种约束条件。此外，AM 模型采用的 REINFORCE 算法使用蒙特卡罗采样来估计梯度，因此估计的梯度具有高方差，这会导致训练过程不稳定。为了更快地收敛模型并将该模型运用到现实生活中更为广泛的问题中，因章对 AM 模型进行了如下创新工作：

将 AM 模型运用在实际物流配送问题中，求解更具有现实意义的 VRPSTW 任务中。除此之外，为改进原模型训练时 REINFORCE 算法高方差的缺点，在模型的训练阶段采用 Actor-Critic 算法。最后将改进的 AM 模型与目前在 VRPSTW 问题上在的四个模型进行了对比试验，在三组不同规模的 VRPSTW 实例实验中，此型在计算 VRPSTW 问题上的路径总成本是最小的，优于其他四组模型的结果。

3.1 求解 VRPSTW 任务改进的 AM 模型（AAM 模型）

本节将详细介绍改进的 AM 模型（Advanced AM model，AAM）。AM[48]模型是基于编码器-解码器结构求解了 CVRP 问题，本文 AAM 模型运用在实际物流配送问题中，求解更具有现实意义的 VRPSTW 任务中。基于 Actor-Critic 强化学习算法求解 VRPSTW 的 AAM 模型。具体如图 3-1 所示。

输入客户节点的信息到编码器中。在 VRPSTW 任务中节点的信息有：位置信息、需求、时间窗约束。编码器产生所有输入节点的 embedding 信息，解码器中使用注意力机制来产生下一个输入的概率分布，以选择下一个客户的位置。其中编码器包括多头注意力层和全连接前馈子层。解码器中根据一个上下文节点信息来表示当前节点解码的上下文信息。上下文节点主要通过编码器的 embedding 信息，解码器在时间步 t 之前的节点输出信息来加以构造。在 VRPSTW 任务中编码器和解码器的工作原理和流程如下所示。

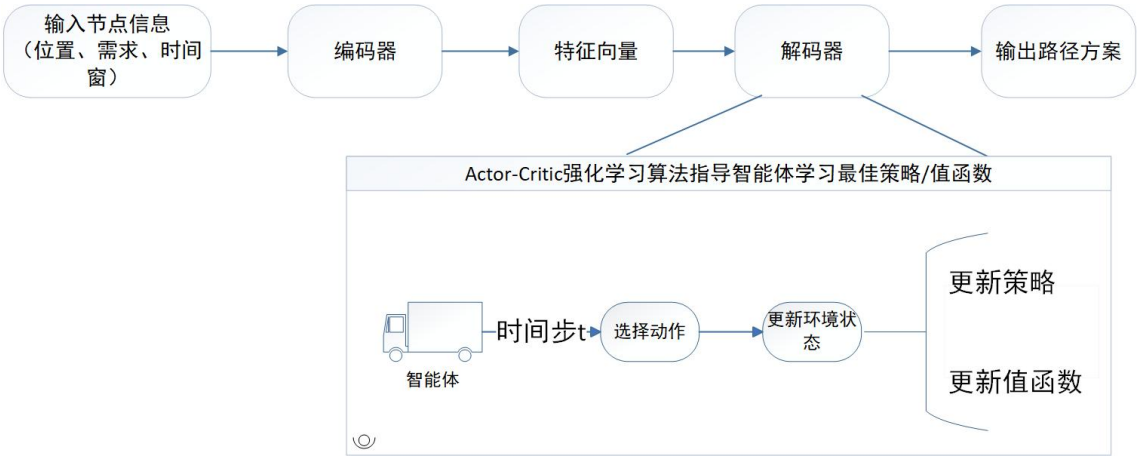


图 3-1 求解 VRPSTW 任务 AAM 模型框架

3.1.1 编码器

编码器产生所有输入节点的 embedding 信息，如图 3-2 所示为编码器结构图。

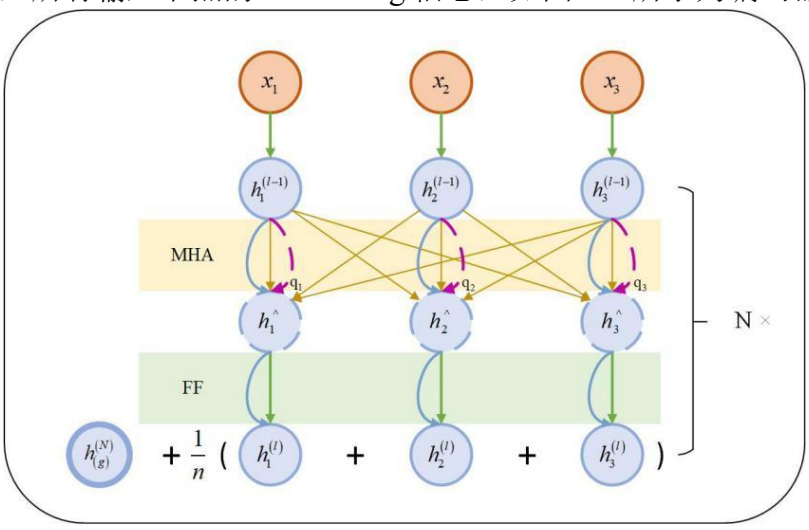


图 3-2 编码器结构图

编码器对于输入的 d_x维节点特征x_i，通过一个线性变化通过一个线性映射计算得到d_h维的节点 embedding 的$h_i^{(0)}$信息。

在 VRPSTW 任务中，d_x=4，由节点的位置、需求、时间窗决定的，其中d_h=128 。节点$h_i^{(0)}$的计算公式如（3-1）所示：

$$h_i^{(0)} = W^x x_i + b^x \tag{3-1}$$

之后节点$h_i^{(0)}$经过 N 层注意力层来更新节点 embedding，每一层包含两个子层： MHA 和 FF，对于每个子层，还添加跳跃链接(skip-connection)和批量归一化层(Batch Normalization，BN)。h_l和$h_i^{(1)}$的计算公式，如下所示：

$$\hat{h}_l = BN(h_i^{(l-1)} + MHA_i^l(h_1^{(l-1)},...,h_n^{(l-1)})) \tag{3-2}$$

$$h_i^{(l)} = BN(\hat{h}_l + FF^l(\hat{h}_l)) \tag{3-3}$$

定义维度d_v和d_k，将 embedding 映射为 key、value 和 query。对应的q_i、k_i、v_i计算公式如下所示：

$$q_i = W^Q h_i \tag{3-4}$$

$$k_i = W^K h_i \tag{3-5}$$

$$\mathrm{v}_i = W^V h_i \tag{3-6}$$

在计算注意力权重时，queries 和 keys 之间用内积表示它们之间的相似度，即 queries 和 keys 之间的相关性。内积可以通过将 queries 和 keys 的向量相乘，然后将结果相加得到。queries 和 keys 的兼容性u_{ij}计算如下所示：

$$\mathrm{u}_{\mathrm{ij}} = \begin{cases} \dfrac{q_i^T k_j}{\sqrt{d_k}}, & \text{如果i, } j\text{相邻} \\ -\infty, & \text{否则} \end{cases} \tag{3-7}$$

在计算注意力权重时，这个内积会被归一化，以便得到一个注意力分布，用于加权计算 values 向量的加权和。对u_{ij}进行 softmax 归一化计算得到注意力权重，公式如下所示：

$$\mathrm{a}_{ij} = \frac{e^{u_{ij}}}{\sum_{j'} e^{u_{ij'}}} \tag{3-8}$$

向量h_i'是计算 values 向量的加权和，即对v_j进行加权求和得到的节点的 embedding：

$$\mathrm{h}_i' = \sum_j a_{ij} v_j \tag{3-9}$$

多头注意力机制，它可以同时关注不同的位置和方面，从而提高模型的表现。在该环节中，将 queries 向量 Q、keys 向量 K 和 values 向量 V 分别拆分成多个头，每个头的维度为$d_k = d_v = d_h / M$，在 VRPSTW 任务中，令 M=8，则$d_k = d_v = 16$允许节点从不同的邻居接收不同类型的消息。因此，需要将公式(3-9)使用不同参数计算 8 次。

全连接前馈子层用于进一步提取数据特征，使用维度为$d_{ff} = 512$维的隐藏层和 ReLU 激活函数，全连接子层的计算公式如下所示：

$$FF(\hat{h}_i) = W^{ff,1} \bullet RELU(W^{ff,0} + b^{ff,0}) + b^{ff,1} \tag{3-10}$$

其中$\hat{h}_i$是前面多头注意力层的输出。

3.1.2 解码器

如图 3-3 所示，为 VRPSTW 任务在每一个时间步 t 中解码的过程。图中演示了三个节点信息的解码过程。

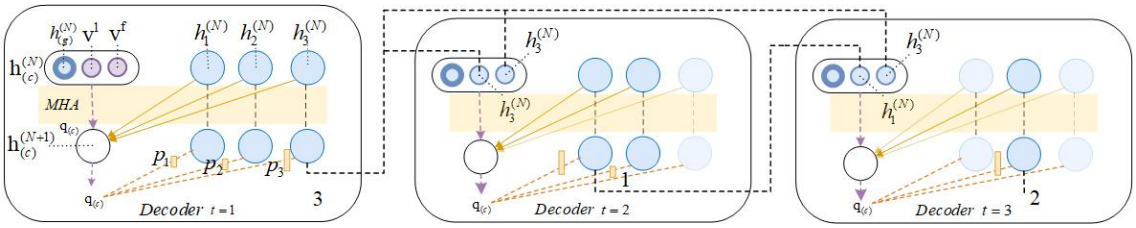


图 3-3 三个节点的解码过程

解码器在时间步 t 的上下文向量，包含编码器和解码器 t 时刻之前的信息。在 VRPSTW 问题中，解码器在时间步 t 时，上下文向量包含编码器生成的 $h_{(g)}^{(N)}$ 和解码器当前生成的最后一个节点和第一个节点的 embedding 信息。对于 $t=1$，使用两个可学习的 d_h 维参数 v^1 和 v^f 代替最后一个节点和第一个节点的 embedding，具体表达如下所示：

$$h_{(c)}^{(N)}=\begin{cases}[h_{(g)}^{(N)},h_{\pi_{t-1}}^{(N)},h_{\pi_1}^{(N)}],t>1\\ [h_{(c)}^{(N)},v_1,v_f],t=1\end{cases} \tag{3-11}$$

当 $t=1$ 时，$[h_{(c)}^{(N)},v_1,v_f]$ 表示将 $h_{(c)}^{(N)},v_1,v_f$ 水平拼接操作，再利用公式（3-12）将 $h_{(c)}^{(N)}$ 映射回 d_h 维。

$$q_{(c)}=W^Q h_{(c)},k_{(i)}=W^K h_{(i)},v_{(i)}=W^v h_{(i)} \tag{3-12}$$

然后再用多头注意力机制计算新的上下文节点的 embedding 信息 $h_{(c)}^{(N+1)}$。

在 VRPSTW 任务中，访问选择过的节点需要遮盖掉，具体的公式如下所示：

$$\mathrm{u_{ij}}=\begin{cases}\dfrac{q_{(c)}^T k_j}{\sqrt{d_k}}，\text{如果}j\neq\pi_t'\\ -\infty,\text{否则}\end{cases} \tag{3-13}$$

3.2 模型训练的 Actor-Critic 强化学习算法

为了减少 REINFORCE 算法的方差，更快地收敛模型，本文采用 Actor-Critic 算法对模型进行训练。

演员-评论家算法将策略评估和策略改进结合在一起，通过同时学习策略和值函数来提高学习效率。其中 Actor 网络负责更新策略，Critic 网络负责更新动作值函数。

那么，策略梯度定理中的评价函数更新如下所示：

$$q_w(\mathrm{s},a)=q^{\pi_\theta}(s,a) \tag{3-14}$$

最终，模型的输出数据是一个车辆路径的解决方案，其中包括每个客户节点的访问顺序，即车辆的路线。Actor 网络用于学习策略，它输出动作的概率分布，预测在任何给定决策步骤下的下一个动作的概率分布，根据这个概率分布选择动作。该网络的任务是负责更新新策略π_θ，根据评论家网络的动作值函数更新参数θ。Critic 网络用于学习价值函数，它估计每个状态的价值并帮助 Actor 网络更好地选择动作，估计从给定状态开始的任何问题实例的奖励。该网络的任务是负责更新动作值$q_w(s,a)$，根据函数近似法更新参数w。

评论家网络是用来估计从给定状态开始的任何问题实例的奖励。在 VRPSTW 任务中，评论家网络会接收当前状态作为输入，并输出一个值函数，用于估计在该状态下采取行动的期望奖励。这个值函数可以被用来计算优势函数，从而更新策略网络的参数。

在这个模型中，评论家网络和演员网络是同时训练的，以最大化期望奖励。演员网络是用来预测在任何给定的决策步骤上下一个动作的概率分布的神经网络。演员网络将当前状态作为输入，并输出一个概率分布，该分布表示在当前状态下采取每个可能动作的概率。这个概率分布是由模型的参数化随机策略 π 来参数化的。在训练过程中，策略梯度方法使用预期奖励相对于策略参数的梯度的估计来迭代地改进策略，以最大化预期奖励。演员网络的目标是学习一个最优的策略以最大化预期奖励。

算法 1 详细描述了演员-评论家（Actor-Critic）算法求解 VRPSTW 问题的过程。

Algorithm 1：演员-评论家（Actor-Critic）算法流程

Initialize θ, s value arbitrarily
for each episode $a \in \pi_\theta$ do：
　　for t = 1 to T-1 do：
　　　　Sample reward r=R，sample trasition $s^{'}\sim P$;
　　　　Sample action $a^{'}\sim\pi_\theta(s^{'},a^{'})$;
　　　　$\delta^{\pi_\theta}=r+\gamma V^{\pi_\theta}(s^{'})-Q^{W}(s,a)$;
　　　　$\theta\leftarrow\theta+\alpha\Delta_\theta \log\pi_\theta(s,a)Q^{W}(s,a)$;　　//v_t作为$Q(s,a)$的无偏估计
　　　　$w\leftarrow w+\beta\delta(s,a)$;
　　　　$a\leftarrow a',s\leftarrow s'w$;
　　end for
end for

演员-评论家 Actor-Critic 算法在 VRPSTW 任务中的工作步骤如下：

图 3-4 是基于 Actor-Critic 算法的 AAM 模型在求解 VRPSTW 任务的流程图。

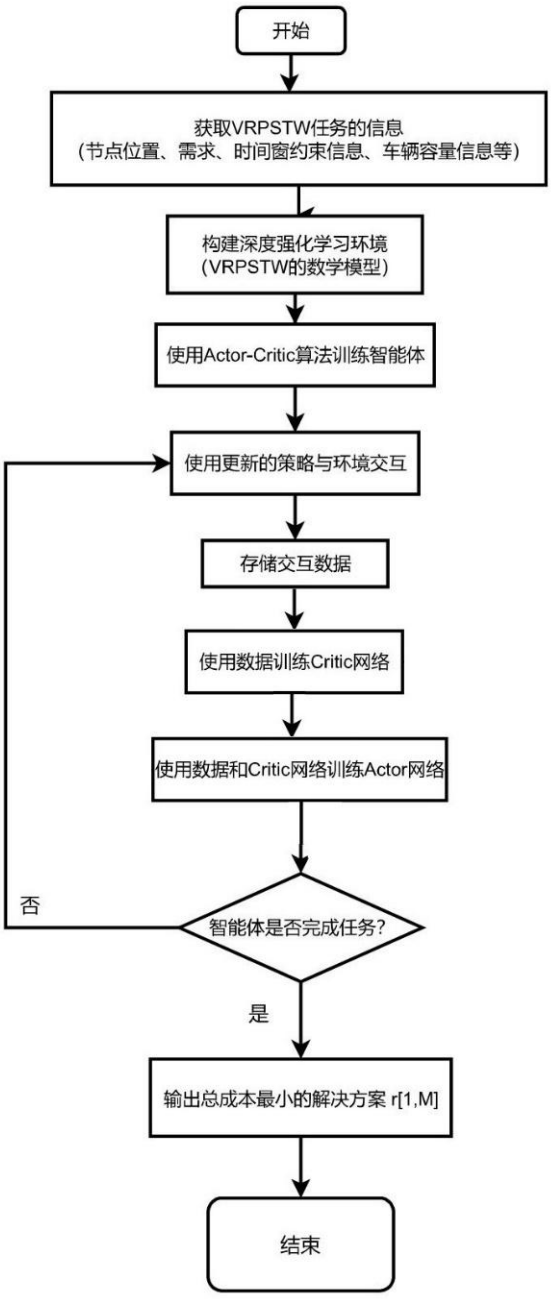


图 3-4 求解 VRPSTW 任务的流程图

首先，在初始化阶段，算法初始化策略网络和值函数网络，并设置初始状态。然后，在策略评估阶段，根据当前状态和策略网络选择一个动作，并根据环境反馈的奖励更新值函数网络。其次，在策略改进阶段，根据值函数网络的输出和当前状态更新策略网络的参数。接下来，重复迭代操作，需要重复执行步骤策略评估和策略改进直到达到停止条件（例如，达到最大迭代次数或找到满意的解决方

案）。

Actor-Critic 算法是一种基本的策略梯度算法，它使用 actor 网络来输出动作概率分布，使用 critic 网络来估计状态值函数。其中 AC 算法的训练是串行的，即每个时间步只更新一次网络参数。

3.3 实验结果与分析

3.3.1 数据实例说明

VRP 问题属于组合优化问题中的 NP 问题，随着节点数量的增加会导致问题的复杂度呈指数级增长。对于一个含有 n 个节点的问题，其解空间大小为 O(n!)，即存在 n!种不同的排列方式。因此，当节点数量增加到 100 个时，问题的规模会非常之大，需要使用高效的算法和技术来解决。此外，VRP 问题还涉及到车辆的容量、时间窗口等约束条件，这些条件的增加也会进一步增加问题的复杂度。因此，在处理 VRPSTW 问题中根据问题的复杂度设置三种不同规模大小的实例，分别是具有 20 个节点的小规模实例、具有 50 个节点的中等规模实例和具有 100 个节点的大规模实例。以下是对三种规模实例数据的具体参数说明。

（1） 20 个客户的实例——VRPSTW20

在小规模实例中，客户的位置、需求和客户节点对应的时间窗口数据生成，是基于著名基准问题集 Solomon 的要求生成的。在 VRPSTW20 实例中，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成，车辆容量设置为 60。早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.1 和 0.5。

（2） 50 个客户的实例——VRPSTW50

在中等规模的情况下随机生成 50 个客户节点的信息，车辆容量设置为 750，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成。早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.1 和 0.5。

（3） 100 个客户的实例——VRPSTW100

在大等规模的情况下，此实例中随机生成 100 个客户节的位置信息，车辆容量设置为 1000，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成。早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.1 和 0.5。

3.3.2 实验环境与参数说明

AAM 模型的实验平台如表 3-1 所示。

AAM 模型基于 Actor-Critic 算法进行训练，模型训练的迭代次数为 50 个 epoch，在每个时期处理 128000 个小规模实例和 640000 个中规模和大规模实例。对于小规模实例和中等规模实例，batch 大小设置为 512，对于大型实例，batch 大小设置为 256。其中嵌入层的维度设置为 128，对于解码器中的隐向量维度设置为 256。为了控制模型参数在每次迭代中的更新幅度使用 Adam 优化器，并采用一个平滑的学习率衰减策略，即在每个 epoch 中，学习率按照一个指数衰减的方式进行更新，以控制模型参数的更新幅度。

因此，学习率的更新公式为：

$$\eta_t = \left(\frac{1}{1+\gamma_t}\right) * \eta_t - 1 \tag{3-15}$$

其中η_t表示第 t 个 epoch 的学习率，γ 是一个衰减因子，t 表示当前的 epoch 数。学习率的初始值为 1e-4，每个 epoch 的衰减倍率为 0.96。

表 3-1 AAM 模型的实验平台

设备	参数	设备	参数
CPU	Intel(R) Xeon(R) Silver 4210R CPU @2.40GH	GPU	NVIDIA RTX 309
显存	24GB	内存	64GB
操作系统	Ubuntu 18.04	Python	3.8
TensorFlow	2.4.1		

3.3.3 实验结果与分析

该模型使用的基线对比模型如下：

RL 模型[44]：基于指针网络和注意力机制的模型，RL 模型的编码器读取输入序列并将其转换为固定大小的向量表示，RL 模型的解码器将该向量表示转换回输出序列。使用 REINFORCE 算法来训练模型。

JAMPR 模型[47]：用于 CVRPTW 问题的深度学习模型。它采用了更加全面的状态和动作空间提供关于节点、路径和车辆的增强特征嵌入，以及联合动作空间的注意力机制，能够同时处理多个路径，从而更好地解决车辆路径问题。

LHK3[52]：基于 Lin-Kernighan 启发式的离线优化器，用于解决 TSP 和 VRP 等组合优化问题。LKH3 算法在实践中已经被证明是一种有效的优化器，能够在较短

的时间内找到具有高质量的解决方案。

Google OR-Tools[53]（简称 GORT）：是用于优化问题的软件套件，包括车辆路径规划、调度和约束编程。它提供了一组库和求解器，可用于各种编程语言，包括 C ++、Python、Java 和.NET。OR-Tools 库包括广泛的算法和技术，用于解决优化问题，例如线性规划、混合整数规划和局部搜索。它被广泛用于工业和学术界解决复杂的优化问题。

表 3-2 实验结果

模型	VRPSTW20		VRPSTW50		VRPSTW100	
	Cost	Time	Cost	Time	Cost	Time
RL	3766.88	0.05s	7189.43	0.12s	5570.30	3.29s
JAMPR	3041.24	0.12s	7327.09	0.38s	2785.88	3.80s
LHK3	1862.40	0.10s	3055.94	0.24s	8497.29	8.63s
GORT	2577.08	0.22s	4344.88	1.76s	8924.56	1.20s
AAM	1716.60	0.86s	2691.55	3.07s	2682.39	8.98s

如上所示，表 3-2 是本文模型以及对照组的实验结果。通过五种模型对三组不同规模的 VRPSTW 实例进行评估，我们的 AAM 模型在 VRPSTW20 的小规模实例上相比于其他四组模型的结果，其计算所得的代价是最小的。在 VRPSTW50 的中规模实例上相比于其他四组模型的结果，其计算所得的代价也是最小的，其中 RL 模型虽然在计算时间上是最快的，但是代价要远远高于其他四组模型。在 VRPSTW100 的大规模实例上，使用改进的 AAM 模型计算所得的代价优于其他四组模型的结果。综上所述， AAM 模型，在计算 VRPSTW 问题上的路径总成本是最小的，优于其他四组模型的结果。

3.4 本章小结

本章提出了一种 AAM 框架用于求解 VRPSTW 问题，通过使用 Actor-Critic 强化学习算法来训练模型，同时学习策略和值函数来提高学习效率。实验表明，五种模型在三组不同规模的 VRPSTW 实例实验中， AAM 模型在计算 VRPSTW 问题上的路径总成本是最小的，优于其他四组模型的结果。

第 4 章　基于动态注意力的 AAM 模型求解 VRPSTW 问题

为了提高 AAM 模型的求解速度，本章采用动态注意力机制来优化 AAM 模型（Advanced Attention Mode based on dynamic attention mechanism，AAM-D），在该模型中节点特征是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。为了更好地提取输入序列的信息，在 AAM 模型的基础上改进了编码器器中节点更新的方式。传统的注意力机制中输入实例的特征经过编码器之后是固定的，不随模型的决策而改变。在改进的 AAM-D 模型中输入实例的特征表示是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态、捕捉实例的结构特征。

4.1 基于动态注意力机制优化的 AAM 模型(AAM-D)

为了更好地提取输入序列的信息、反映实例的状态、捕捉实例的结构特征，AAM-D 模型在前一个模型上的编码器的基础上进行了改进，AAM-D 模型整体的网络结构图如图 4-1 所示。

该模型的数据流是从输入实例开始通过编码器和解码器交替构建部分解决方案，并使用注意力机制动态更新节点嵌入，最终输出 VRPSTW 的解决方案，其中包括每个客户节点的访问顺序，即车辆的路线。

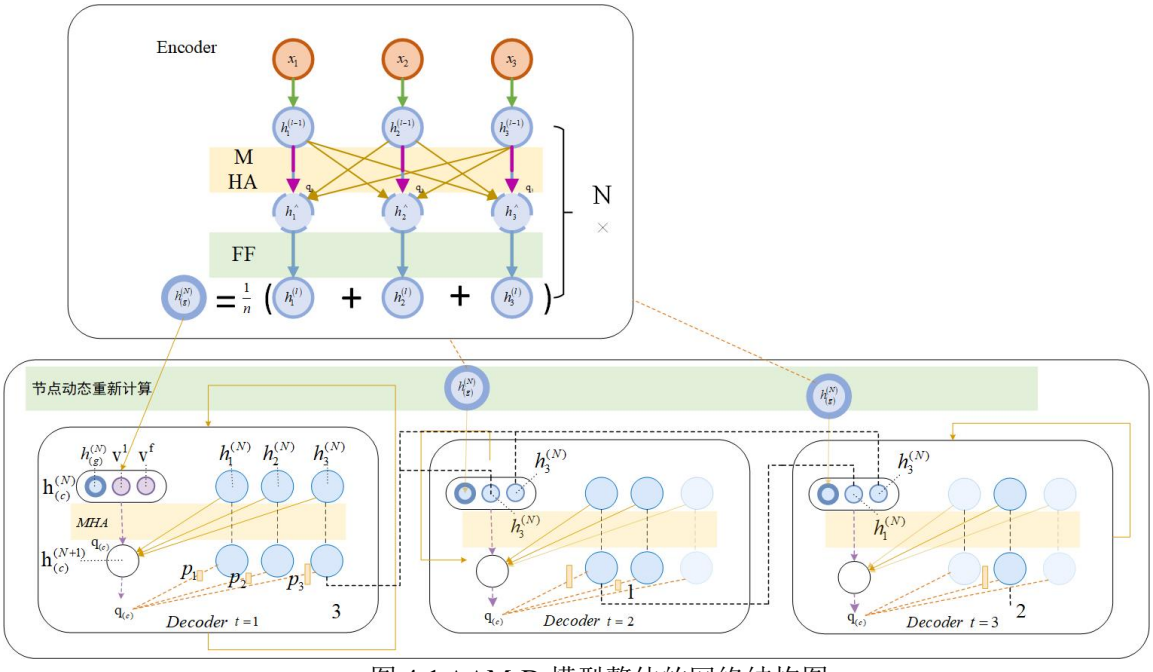


图 4-1 AAM-D 模型整体的网络结构图

在传统的注意力机制中输入实例的特征表示是固定的，不随模型的决策而改变。而在具有动态更新的注意力机制中输入实例的特征表示是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。具有动态更新的注意力机制可以帮助模型更好地捕捉实例的结构特征，从而提高模型的性能。在本文中动态注意力机制被用于车辆路径问题的解决，以帮助模型更好地构建可行解。

在该模块中每个位置的特征表示不仅依赖于自身的特征，还依赖于上下文信息。它通过引入一个上下文向量，将每个位置的特征表示与上下文向量进行加权求和，得到最终的特征表示。上下文向量的计算可以通过对之前位置的特征表示进行加权平均得到，在 VRPSTW 任务中节点的嵌入向量会在车辆返回到车站时立即重新计算。

4.2 注意力机制动态更新节点

在 AAM-D 模型中编码器对于输入的 d_x维节点特征x_i，通过一个线性变化通过一个线性映射计算得到d_h维的节点 embedding 的$h_g^{(0)}$ 信息。

在 VRPSTW 任务中，d_x=4, d_h=128。节点$h_g^{(0)}$ 的计算公式如（4-1）所示：

$$h_g^{(0)} = W^x x_i + b^x \tag{4-1}$$

之后节点$h_g^{(0)}$ 经过 N 层注意力层来更新节点 embedding，每一层包含两个子层：多头注意力层和全连接前馈层。$\hat{h}_g$ 和$h_g^{(1)}$ 的计算公式，如下所示：

$$\hat{h}_g = BN(h_g^{(l-1)} + MHA_i^l(h_1^{(l-1)},...,h_n^{(l-1)})) \tag{4-2}$$

$$h_g^{(l)} = BN(\hat{h}_g + FF^l(\hat{h}_g)) \tag{4-3}$$

定义维度d_v和d_k，将 embedding 映射为 key、value 和 query。对应的q_i、k_i、v_i计算公式如下所示：

$$q_i = W^Q h_g \tag{4-4}$$

$$k_i = W^K h_g \tag{4-5}$$

$$v_i = W^V h_g \tag{4-6}$$

在计算注意力权重时，queries 和 keys 之间用内积表示它们之间的相似度，即 queries 和 keys 之间的相关性。内积可以通过将 queries 和 keys 的向量相乘，然后将结果相加得到。

queries 和 keys 的兼容性u_{ij}计算如下所示：

$$u_{ij}=\begin{cases}\dfrac{q_i^T k_j}{\sqrt{d_k}}, & \text{如果i, } j\text{相邻}\\ -\infty, & \text{否则}\end{cases} \tag{4-7}$$

在计算注意力权重时，这个内积会被归一化，以便得到一个注意力分布，用于加权计算 values 向量的加权和。对 u_{ij} 进行 softmax 归一化计算得到注意力权重，公式如下所示：

$$a_{ij}=\frac{e^{u_{ij}}}{\sum_{j'} e^{u_{ij'}}} \tag{4-8}$$

向量 h_i' 是计算 values 向量的加权和，即对 v_j 进行加权求和得到的节点的 embedding：

$$h_g'=\sum_j a_{ij} v_j \tag{4-9}$$

在该模块中，每个位置的特征表示不仅依赖于自身的特征，还依赖于上下文信息。在 VRPSTW 任务中，节点的嵌入向量会在车辆返回到车站时立即重新计算得到h_g。解码器在时间步 t 的上下文向量，包含编码器和解码器 t 时刻之前的信息。在时间步 t 时，上下文向量包含编码器生成的 $h_{(g)}^{(N)}$ 和解码器当前生成的最后一个节点和第一个节点的 embedding 信息。对于$t=1$，使用两个可学习的 d_h 维参数 v^1 和 v^f 代替最后一个节点和第一个节点的 embedding。

4.3 实验与结果分析

4.3.1 数据实例说明

在 AAM-D 模型中，以下是对三种规模实例数据的具体参数说明。

VRPSTW20 小规模实例，客户的位置、需求和客户节点对应的时间窗口数据生成，是基于著名基准问题集 Solomon 的要求生成的。在 VRPSTW20 实例中，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成，车辆容量设置为 60，早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.15 和 0.6。

VRPSTW50 中等规模实例，随机生成 50 个客户节点的信息。车辆容量设置为 750，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成，早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.15 和 0.6。

VRPSTW100 大规模实例，此实例中随机生成 100 个客户节的位置信息，车辆容量设置为 1000，客户节点地位置在 [0,10]×[0,10]这个区间生成，每个客户的需求是从 [0,10]区间范围随机产生的，时间窗口从[0,100]随机生成。早到惩罚系数和晚到惩罚系数α_i，β_i 分别为 0.15 和 0.6。

4.3.2 实验环境与参数说明

模型的求解效果不仅与模型的合理设计与算法公式的正确有关，同样与实验采用的参数有关。实验参数对于深度强化学习算法对问题的求解质量有着重要的作用，本节将从实验所用的数据集、模型与强化学习算法的超参数等方面介绍本实验所需的实验参数，以确保本文的实验具有可重复性。

学习率是 AAM-D 模型在训练过程中非常重要的超参数，本文在学习率的选取上考虑了两种常见的学习率：静态学习率和动态学习率。其中静态学习率设置为$1e-4$。静态学习率值小，不需要在训练过程中进行动态变化，但在训练前期收敛速度较慢，且训练后期中易出现震荡。动态学习率设置初始学习率为$1e-3$，每个 epoch 的衰减倍率为 0.96。动态学习率的优点为训练前期可以以较大的学习率使模型在短时间内收敛，并在训练后期以较小的学习率求解量优值。但动态学习率依赖于学习率的初始设计，相对于静态学习率难度较大。因此本文实验均采取静态学习率。

AAM-D 模型基于 Actor-Critic 算法进行训练，模型训练的迭代次数为 50 个 epoch，在每个时期处理了 128000 个小规模实例和 640000 个中规模和大规模实例。对于小规模实例和中等规模实例，batch 大小设置为 512，对于大型实例，batch 大小设置为 256。其中嵌入层的维度设置为 128，对于解码器中的隐向量维度设置为 256。为了控制模型参数在每次迭代中的更新幅度使用了 Adam 优化器，并采用了一个平滑的学习率衰减策略，即在每个 epoch 中学习率按照一个指数衰减的方式进行更新，以控制模型参数的更新幅度。

AAM-D 模型的实验平台如表 4-1 所示。

表 4-1 AAM 模型的实验平台

设备	参数	设备	参数
CPU	Intel(R) Xeon(R) Silver 4210R CPU @2.40GH	GPU	NVIDIA RTX 309
显存	24GB	内存	64GB
操作系统	Ubuntu 18.04	Python	3.8
TensorFlow	2.4.1		

4.3.3 实验结果

在该实验中，直接将 AAM-D 模型与前面提到的五种模型进行对比，实验结果如表 4-2 所示。

表 4-2 AAM-D 模型以及对照组的实验结果

模型	VRPSTW20		VRPSTW50		VRPSTW100	
	Cost	Time	Cost	Time	Cost	Time
RL	3564.88	0.08s	3507.16	0.98 s	5170.30	3.09s
JAMPR	2941.14	0.53s	7077.09	0.41s	3885.88	3.20s
LHK3	2862.40	0.40s	3055.94	0.24s	8497.29	8.63s
GORT	2977.08	0.52s	4344.88	1.76s	3436.51	0.59s
AAM	1916.60	0.86s	2641.55	3.77s	2582.54	8.98s
AAM-D	1844.35	0.78s	1791.92	3.08 s	2082.39	7.45s

通过六种模型对三组不同规模的 VRPSTW 实例进行评估，改进的 AAM-D 模型在求解 VRPSTW 问题上，求解质量和求解速度优于 AAM 模型。

在 VRPSTW20 的小规模实例上，相比于其他五组模型的结果，其计算所得的代价是最小的，时间上要优于 AAM 模型。在 VRPSTW50 的中规模实例上，相比于其他五组模型的结果，其计算所得的代价也是最小的，其中 RL 模型虽然在计算时间上是最快的，但是代价要远远高于其他五组模型。在 VRPSTW100 的大规模实例上，使用 AAM-D 计算所得的代价优于其他五组模型的结果。

综上所述， 改进的 AAM-D 模型在计算路径总成本上优于其他五组模型的结果，求解速度比 AAM 模型有所提高。

4.4 本章小结

为了提高 AAM-D 的求解速度，本章提出了一种基于动态注意力的 AM 模型，即 AAM-D 模型，在该模型中节点特征是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。实验表明，六种模型在三组不同规模的 VRPSTW 实例实验中，改进的 AAM-D 模型，在计算路径总成本上优于其他五组模型的结果，求解速度优于 AAM 模型。

第5章　基于深度强化学习的VRPSTW实验平台设计与实现

5.1 引言

VRPSTW 是一种实际应用非常广泛的组合优化问题，它涉及到物流、运输、配送等领域。现实生活中，由于 VRP 问题的复杂性，开发一个 VRPSTW 实验平能够提供一个模拟环境，用于测试和评估不同的车辆路径规划算法。通过将这些算法在相同的环境中进行比较，可以更好地理解它们的优缺点，并选择最适合特定应用场景的算法。除此之外，实验平台提供一个快速、低成本的方法来测试新的路径规划算法，而不必在现实世界中构建车辆测试环境。这降低了研究和开发过程的成本，并允许更多人参与到相关研究和开发工作中。

本节中通过开发一个基于深度强化学习的 VRPSTW 实验平台，帮助企业和组织更好地规划和管理物流和配送，从而提高效率和降低成本。为研究者提供一个便利的测试环境，可以帮助促进相关领域的研究进展。通过分析实验结果，研究者能够发现问题，并提出新的探索性研究，以进一步推进该领域的发展。

该 VRPSTW 的实验平台具有以下功能和特性： 优化路线规划，实验平台可以根据不同的需求和限制条件，自动规划最优的路线，从而减少运输成本和时间；实时监控和调整，实验平台可以实时监控车辆的位置和运输情况，及时调整路线和配送计划，以应对突发情况； 数据分析和决策支持，实验平台可以收集和分析大量的运输数据，为企业和组织提供决策支持和业务优化建议。

开发一个基于深度强化学习的 VRPSTW 实验平台可以帮助企业和组织更好地管理物流和配送，提高效率和降低成本，从而获得更大的商业价值。

本章结合第四章提出基于动态注意力机制优化的 AAM-D 模型，将训练好的模型集成到应用程序中，设计并实现了一个 VRPSTW 的实验平台，为后面的研究者提供了一种更为方便的可视化实验平台。

5.2 开发环境

在本节中主要介绍实现这样的平台所需的开发环境及说明。其中两个训练好的模型是基于 TensorFlow 进行编写的，需要把训练好的模型集成到该实验平台中。因此，模型的训练采用 python 开发环境，web 的实验平台采用 java web 开发，以下是实现该实验平台所需的开发环境说明。

1. 模型训练所需的 python 开发环境

开发语言：python 3.8

主要的工具包：TensorFlow 2.2

python 开发环境 IDE：PyCharm COMMUNITY 2019.1

JRE: 11.0.2+9-b159.34 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

运行环境：windows 10，64-bit，CPU 4 核，内存 16GB

GPU 型号：RTX 2080 Ti * 1 卡

云平台：AutoDL

2. 实验平台搭建所需的 java web 开发环境

开发语言：java 11.0.15

JRE: 11.0.2+9-b159.34 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

服务器：tomcat 8.0

运行环境：windows 10，64-bit，CPU 4 核，内存 14GB，机带 RAM 8.00 GB

处理器：Intel(R) Celeron(R) CPU N3150 @1.60GHz 1.60 GHz

5.3 功能需求分析

车辆路径规划问题的实验平台通常需要包括仿真系统、路网地图构建工具、路径规划算法等组成部分。现在常见的车辆路径规划问题的实验平台主要有：CARLA、SUMO（Simulation of Urban MObility）和 ROS（Robot Operating System）。

CARLA 是一个开源的城市驾驶车辆模拟器，支持高级传感器和控制器，并且提供了各种行为应用程序接口（API），以便测试和评估车辆路径规划算法。SUMO 是一个免费的道路交通模拟器，能够模拟城市道路的交通情况，可以支持车辆路径规划算法的仿真实验。ROS 是一个机器人操作系统，也可以用于车辆路径规划问题的仿真实验。它支持 ROS Navigation Stack 软件包，其中包含多种路径规划算法。

针对现有的实验平台的功能和特点，我们开发的是基于深度强化学习的 VRPSTW 实验平台，一款针对为研究者提供的一种更为方便可视化实验平台，本文的实验平台中的某些操作模块平台更加灵活或者更加易于使用，也提供了更好的实验环境和数据分析工具。

因此，该实验平台的功能模块图如图 5-1 所示。

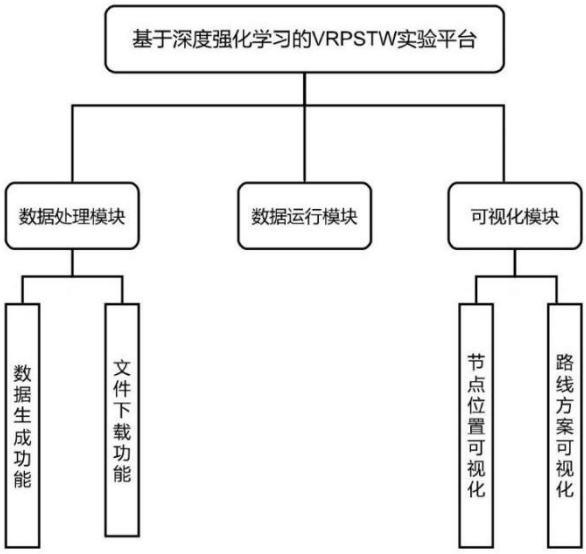


图 5-1 实验平台的功能模块图

数据处理模块：数据生成功能、文件下载功能（节点信息数据下载、最优路线方案下载）。

数据运行模块：根据制定生成的文件对其加载制定好的模型进行求解。

可视化模块：节点位置可视化、路线方案可视化。

5.4 系统实现

5.4.1 系统功能实现

在数据生成功能模块中，主要针对 VRPSTW 问题生成指定规模的实例数据。前端给用户提供指定节点的个数 n，生成对应的 VRPSTW_n.txt 示例数据文件，生成的节点信息能够更好的为后续研究者提供数据支持。

图 5-2 所示为 VRPSTW 实验平台根据用户生成实例的界面图，在该页面中相应的数据生成成功之后，会有对应的弹窗信息提示。在数据生成功能模块中，首先需要确定需求点的数量，其中每个需求点都有一个对应的需求量，需求点可以代表客户、仓库、节点的时间窗口等。然后，根据需求点的坐标，可以计算出需求点之间的欧几里得距离矩阵。后台会确定车辆数量和容量，根据用户输入的需求确定总共有多少辆车，后台在生成每辆车的容量，以及其他限制条件等。

如图 5-3 所示，为文件下载功能页面。在文件下载功能页面中主要实现了两个功能：节点信息文件下载以及最优路线结果下载。节点信息文件下载是根据用户在数据生成模块中下载相关的文件。最优路线结果下载是根据用户在数据运行模块中，下载对应 VRPSTW 实例的结果。

图 5-2 VRPSTW 实例生成的界面图

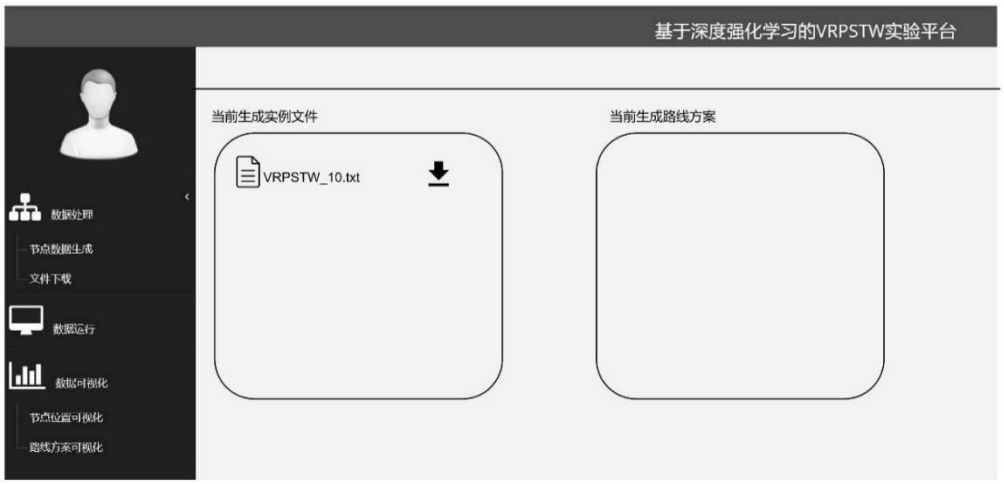
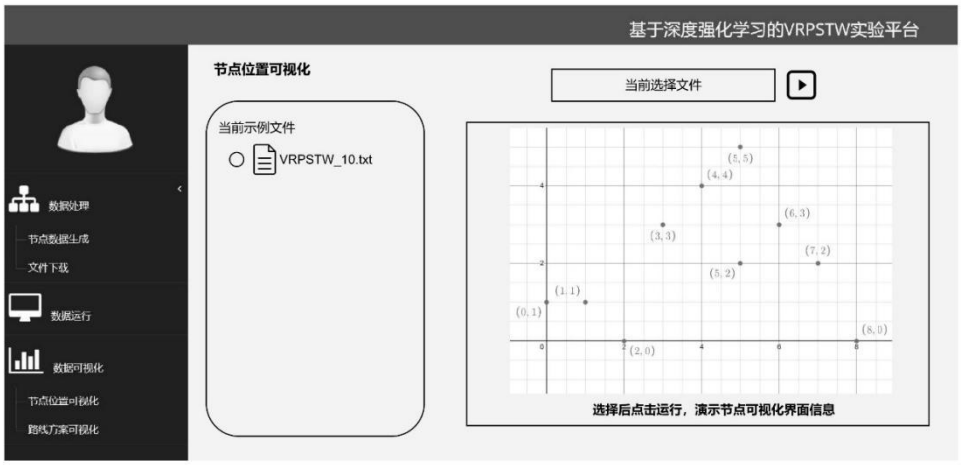


图 5-3 文件下载功能界面图

在数据运行功能页面中主要实现了最优路线结果的生成。用户可以将自己的 VRP 实例上传到平台上进行求解，用户也可以选择自己在数据生成模块中选择生成的 VRPSTW 实例，为该实例生成对应的结果。程序运行完毕之后，会弹出对应的弹窗提示用户。文件上传或选择成功后，用户可以在平台上进行求解设置。这些设置包括选择求解算法、设置迭代次数、设置终止条件等。设置完毕后，用户可以点击“开始运行”按钮，启动求解器开始求解 VRP 实例。根据实例的大小和复杂程度不同，求解时间可能会有所不同。求解完成后，用户可以查看求解结果。这些结果包括车辆路径、总成本、服务时间窗口等指标，并可以通过可视化工具展示车辆路径图、需求点分布图等。生成的最优路线结果会保存在系统后台中，用户如需下载该结果，可以到文件下载模块中下载对应的 txt 文件。图 5-4 所示为数据运行功能页面。

图 5-4 数据运行功能界面图

在该 VRPSTW 实验平台，可以通过该功能直观地展示 VRP 实例的需求点分布和车辆路径。在节点位置可视化功能页面中主要实现了节点信息可视化的功能，为用户更为直观的理解该问题。点击“结果分析”按钮，在弹出的菜单中选择“需求点分布图”。选择“需求点分布图”后，平台会根据 VRP 实例文件中每个需求点的坐标，绘制出需求点在地图上的分布情况。用户可以通过放大、缩小、移动等方式查看地图。同时，在地图上可以显示各个需求点的编号和需求量信息。图 5-5 是对应的 VRPSTW_10 实例的可视化界面，为该实例的节点信息可视化界面。



5-5 VRPSTW_10 节点位置可视化界面图

在路线方案可视化页面中主要实现了路线方案可视化功能，为用户更为直观的演示结果。点击“结果分析”按钮，在弹出的菜单中选择“车辆路径图”。选择“车辆路径图”后，平台会根据求解结果，绘制车辆路径。用户可以通过不同颜色的线条来区分不同车辆的路径。同时，在地图上也可以显示需求点的编号和车辆编号等详细信息。用户可以选择自己在数据运行模块中对应的 VRPSTW 实例，为该实例的路线方案可视化。图 5-6 所示是对应的 VRPSTW_10 实例路线方案的

可视化结果界面。

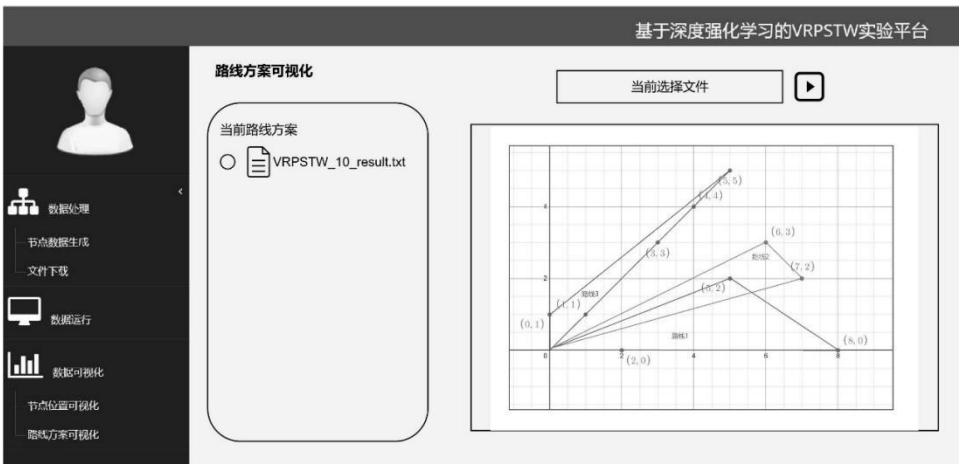


图 5-6 VRPSTW_10 实例路线方案的可视化结果界面

5.4.2 代码模块结构与算法流程

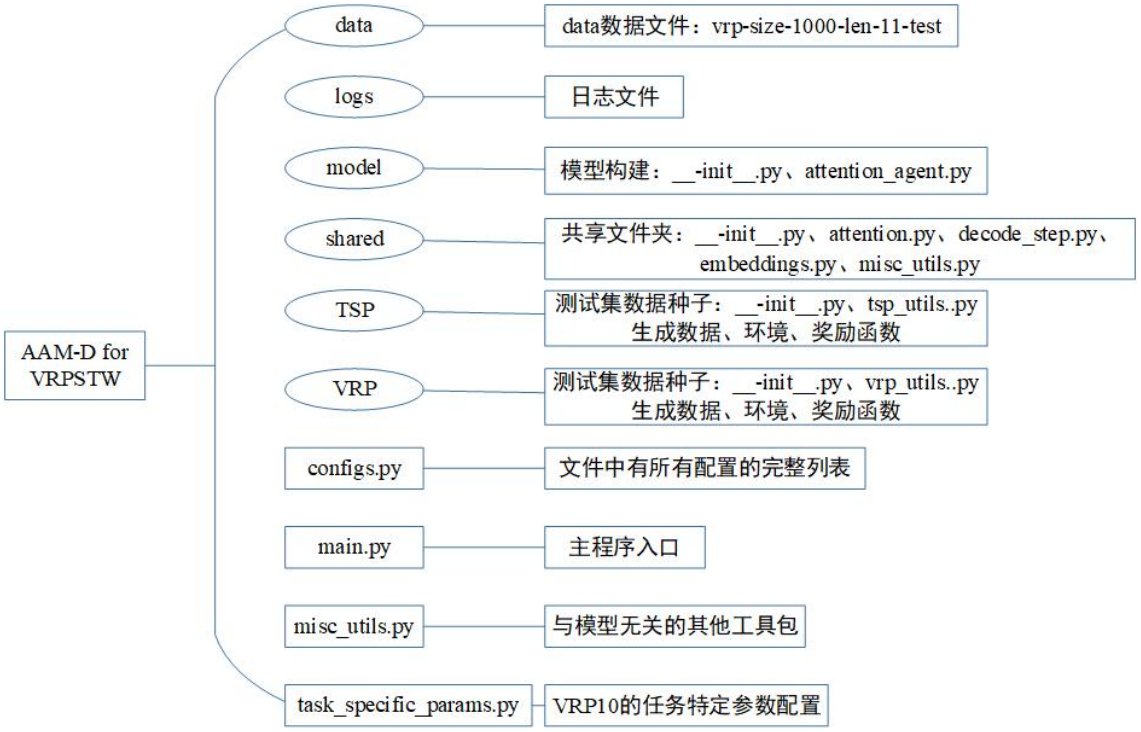


图 5-7 AAM-D for VRPSTW 代码结构组织图

模型的代码是基于 TensorFlow 框架，训练好的模型需要集成到 web 应用程序中，主要需要如下几个步骤：

首先，将基于 TensorFlow 训练好的模型导出为 SavedModel 格式，需要通过使用 TensorFlow 的 SavedModelBuilder API 来完成。然后，使用 TensorFlow Serving 将 SavedModel 部署到 Web 服务器上。TensorFlow Serving 是一个用于部署机器学习

模型的高性能开源系统，它可以轻松地将 TensorFlow 模型部署到生产环境中。接着，在 Web 应用程序中使用 REST API 或 gRPC API 与 TensorFlow Serving 进行通信，以便使用模型进行推理。最后，在 Web 应用程序中使用 JavaScript 来呈现模型的输出。

如图 5-7 所示，为使用 AAM-D 模型求解 VRPSTW 问题，采用 TensorFlow 框架的代码结构组织图。在该工程中，根据需要实现功能的不同，将文件划分为十部分：data 数据文件夹、logs 日志文件夹、model 模型文件夹、shared 共享组件文件夹、TSP 问题文件夹、VRP 问题文件夹、configs 参数配置文件夹、main 主程序文件、模型工具包、特定任务参数配置文件。

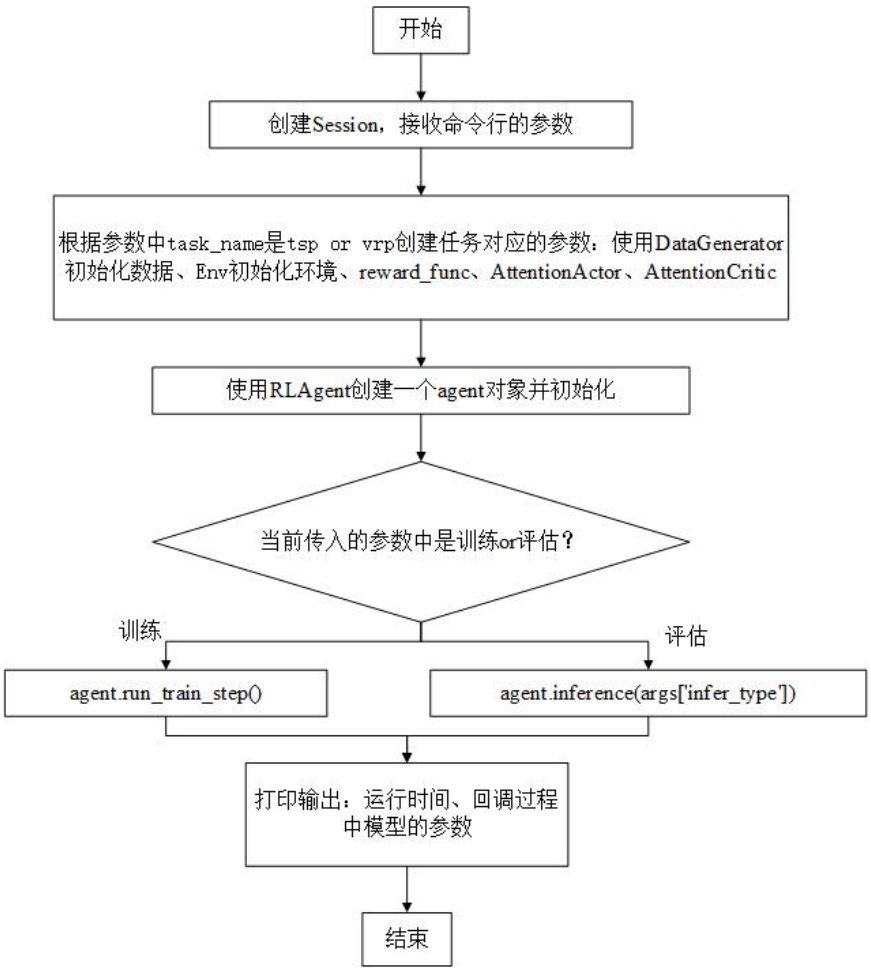


图 5-8 TensorFlow 训练模型的流程图

如图 5-8 所示，为用 TensorFlow 训练模型的具体流程图。在该程序中，用户需要创建 session 用于接收命令行的参数，模型初始化后根据参数的不同进行解析，对模型分别进行训练或评估环节，最后在控制台打印相关参数。

5.5 本章小结

本章主要介绍了基于深度强化学习的 VRPSTW 实验平台的实现意义、实验开发环境介绍、平台功能需求分析、平台各功能实现的页面展示、基于 TensorFlow 框架系统详细实现说明。详述了开发的实验平台的具体功能，可以支持哪些实验操作、处理什么类型的数据等等。强调了实验平台的前沿性和创新性特点。与其他类似的实验平台进行比较，说明本文开发的实验平台的优势和不同之处，本文的实验平台中的某些操作模块平台更加灵活或者更加易于使用。该实验平台提供了更好的实验环境和数据分析工具。基于深度强化学习的 VRPSTW 实验平台为用户提供数据生成功能、文件下载功能（节点信息数据下载、最优路线方案下载）、数据运行功能、节点位置可视化功能、路线方案可视化功能。为研究 VRPSTW 问题提供一个实验平台，让后续研究者更为方便直观地进行研究。也详述了本文开发的实验平台与前面相关的研究工作之间的关系，阐述了实验平台所具有的实际价值。

结 论

本文主要针对 VRPSTW 问题展开深入的研究，构建了两个基于深度强化学习的模型，并在实验上验证了算法的有效性。本文的主要工作如下：

（1）提出一种基于改进注意力机制的 AM 模型求解 VRPSTW 问题。具体来说，AAM 模型运用在实际物流配送问题中，求解更具有现实意义的 VRPSTW 任务中，采用基于 Actor-Critic 强化学习算法求解 VRPSTW 的 AAM 模型。编码器产生所有输入节点的 embedding 信息，解码器中使用注意力机制来产生下一个输入的概率分布，以选择下一个客户的位置。其中编码器包括多头注意力层和全连接子层。解码器中根据一个上下文节点信息来表示当前节点解码的上下文信息。上下文节点主要通过编码器的 embedding 信息，解码器在时间步 t 之前的节点输出信息来加以构造。

（2）提出一种基于动态注意力的 AAM 模型求解 VRPSTW 问题。为了提高 AAM 模型的求解速度，采用动态注意力机制来优化 AAM 模型，在该模型中节点特征是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态。为了更好地提取输入序列的信息，在 AAM 模型的基础上改进了编码器器中节点更新的方式。传统的注意力机制中，输入实例的特征经过编码器之后是固定的，不随模型的决策而改变。在 AAM-D 模型中，输入实例的特征表示是动态更新的，可以根据模型在不同构造步骤中的决策进行更新，从而更好地反映实例的状态、捕捉实例的结构特征。

（3）本文设计了一个基于深度强化学习的 VRPSTW 实验平台。将第四章构建 AAM-D 模型运用起来，将训练好的模型集成到应用程序中，开发了一个求解 VRPSTW 问题的仿真系统。在这个系统实现了节点数据生成与导出、节点位置可视化、最优路线方案生成、 结果可视化分析与导出等功能，为后面的研究者提供了一种更为方便的可视化实验平台。

本文构建的 AAM 模型和 AAM-D 模型，与 RL 模型、LHK3、JAMAR 模型和 Google OR Tools 相比，在求解精度上有很大的优势，但是在求解时间和模型训练时间上仍然存在不足之处。

本文提出来的模型都是基于 Actor-Critic 强化学习算法进行模型训练的，不同于其他模型使用到的 REINFORCE 算法。Actor-Critic 算法它结合了策略评估和策略改进两个方面，可以同时学习策略和价值函数。因此，需要同时训练两个神经网络，即 Actor 和 Critic，这增加了算法的复杂度和训练时间。除此之外，Actor-Critic

算法中的 Critic 网络需要对每个状态进行价值估计，这需要更多的计算资源和时间。Actor-Critic 算法的性能很大程度上取决于环境的复杂度和稳定性，如果环境变化很大或者不稳定，算法的性能会受到很大影响。AC 算法是一种基本的策略梯度算法，未来的研究工作者可采用 A2C 算法和 A3C 算法来训练网络以优化模型。A2C 算法利用多核 CPU 或 GPU 进行并行化，从而加速训练。 A3C 算法使用异步更新的方式来进一步加速训练，利用多台计算机进行并行化，从而进一步加速训练。

参考文献

[1] Braekers K, Ramaekers K, Van Nieuwenhuyse I. The vehicle routing problem: State of the art classification and review[J]. Computers & industrial engineering, 2016, 99: 300-313.

[2] Ma ń dziuk J. New shades of the vehicle routing problem: Emerging problem formulations and computational intelligence solution methods[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2018, 3(3): 230-244.

[3] Delarue A, Anderson R, Tjandraatmadja C. Reinforcement learning with combinatorial actions: An application to vehicle routing[J]. Advances in Neural Information Processing Systems, 2020, 33: 609-620.

[4] Konstantakopoulos G D, Gayialis S P, Kechagias E P. Vehicle routing problem and related algorithms for logistics distribution: a literature review and classification[J]. Operational research, 2020: 1-30.

[5] Basso R, Kulcsár B, Sanchez-Diaz I. Electric vehicle routing problem with machine learning for energy prediction[J]. Transportation Research Part B: Methodological, 2021, 145: 24-55.

[6] Zhang M, Pratap S, Zhao Z, et al. Forward and reverse logistics vehicle routing problems with time horizons in B2C e-commerce logistics[J]. International Journal of Production Research, 2021, 59(20): 6291-6310.

[7] Moussavi S E, Mahdjoub M, Grunder O. A matheuristic approach to the integration of worker assignment and vehicle routing problems: Application to home healthcare scheduling[J]. Expert Systems with Applications, 2019, 125: 317-332.

[8] Harwood S, Gambella C, Trenev D, et al. Formulating and solving routing problems on quantum computers[J]. IEEE Transactions on Quantum Engineering, 2021, 2: 1-17.

[9] Khoufi I, Laouiti A, Adjih C. A survey of recent extended variants of the traveling salesman and vehicle routing problems for Unmanned Aerial Vehicles[J]. Drones, 2019, 3(3): 66: 1-30.

[10] Oyola J. The capacitated vehicle routing problem with soft time windows and stochastic travel times[J]. Revista Facultad de Ingeniería, 2019, 28(50): 19-33.

[11] Gronauer S, Diepold K. Multi-agent deep reinforcement learning: a survey[J]. Artificial Intelligence Review, 2022, 55(2): 895-943.

[12] 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.

[13] 牛鹏飞, 王晓峰, 芦磊, 张九龙.强化学习在车辆路径问题中的研究综述[J]. 计

算机工程与应用, 2022, 58(1):41-45.

[14] 顾世民. 基于深度强化学习的车辆路径规划研究[D]. 福建工程学院, 2021.

[15] Heßler K. Exact algorithms for the multi-compartment vehicle routing problem with flexible compartment sizes[J]. European Journal of Operational Research, 2021, 294(1): 188-205.

[16] Guo J, Long J, Xu X, et al. The vehicle routing problem of intercity ride-sharing between two cities[J]. Transportation Research Part B: Methodological, 2022, 158: 113-139.

[17] He M, Wei Z, Wu X, et al. An adaptive variable neighborhood search ant colony algorithm for vehicle routing problem with soft time windows[J]. IEEE Access, 2021, 9: 21258-21266.

[18] Hintsch T. Large multiple neighborhood search for the soft-clustered vehicle-routing problem[J]. Computers & Operations Research, 2021, 129: 105132.

[19] Wang J, Weng T, Zhang Q. A two-stage multiobjective evolutionary algorithm for multiobjective multidepot vehicle routing problem with time windows[J]. IEEE Transactions on Cybernetics, 2018, 49(7): 2467-2478.

[20] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Reinforcement learning, 1992: 5-32.

[21] Dai H, Dai B, Song L. Discriminative embeddings of latent variable models for structured data[C]//International conference on machine learning. PMLR, 2016: 2702-2711.

[22] Duan J, He Z, Yen G G. Robust multiobjective optimization for vehicle routing problem with time windows[J]. IEEE Transactions on Cybernetics, 2021, 52(8): 8300-8314.

[23] Wang J, Zhou Y, Wang Y, et al. Multiobjective vehicle routing problems with simultaneous delivery and pickup and time windows: formulation, instances, and algorithms[J]. IEEE transactions on cybernetics, 2015, 46(3): 582-594.

[24] Hof J , Schneider M , Goeke D . Solving the battery swap station location-routing problem with capacitated electric vehicles using an AVNS algorithm for vehicle-routing problems with intermediate stops[J]. Transportation Research Part B: Methodological, 2017, 97(MAR.):102-112.

[25] Erdelić T, Carić T. A survey on the electric vehicle routing problem: variants and solution approaches[J]. Journal of Advanced Transportation, 2019, 2019.

[26] Salavati-Khoshghalb M, Gendreau M, Jabali O, et al. A hybrid recourse policy for the vehicle routing problem with stochastic demands[J]. EURO Journal on Transportation and Logistics, 2019, 8(3): 269-298.

[27] Marinakis Y, Iordanidou G R, Marinaki M. Particle swarm optimization for the vehicle routing problem with stochastic demands[J]. Applied Soft Computing, 2013, 13(4): 1693-1704.

[28] 李阳, 范厚明, 张晓楠, 等. 随机需求车辆路径问题及混合变邻域分散搜索算法求解[J]. 控制理论与应用, 2017, 34(12): 1594-1604.

[29] Xu S H, Liu J P, Zhang F H, et al. A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows[J]. Sensors, 2015, 15(9): 21033-21053.

[30] Masrom S, Abidin S Z Z, Omar N, et al. Dynamic parameterizations of particle swarm optimization and genetic algorithm for facility layout problem[J]. ARPN Journal of Engineering and Applied Sciences, 2017, 12(10): 3195-3201.

[31] 戚远航, 蔡延光, 蔡颢, 等. 带时间窗的车辆路径问题的离散蝙蝠算法[J]. 电子学报, 2018, 46(3): 672-679.

[32] Bello I, Pham H, Le Q V, et al. Neural combinatorial optimization with reinforcement learning[J]. arXiv preprint arXiv:1611.09940, 2016.

[33] Hopfield J J, Tank D W. “Neural” computation of decisions in optimization problems[J]. Biological cybernetics, 1985, 52(3): 141-152.

[34] Sutskever I, Vinyals O, Le Q V. Sequence to Sequence Learning with Neural Networks[J]. arXiv preprint arXiv:1409.3215, 2014.

[35] Vinyals O, Fortunato M, Jaitly N. Pointer Networks[J]. arXiv e-prints, 2015: arXiv: 1506.03134.

[36] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[37] Sheng Y, Ma H, Xia W. A pointer neural network for the vehicle routing problem with task priority and limited resources[J]. Information Technology and Control, 2020, 49(2): 237-248.

[38] Xu K, Wu L, Wang Z, et al. Graph2seq: Graph to sequence learning with attention-based neural networks[J]. arXiv preprint arXiv:1804.00823, 2018.

[39] Kool W, van Hoof H, Gromicho J, et al. Deep policy dynamic programming for vehicle routing problems[C]//Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 19th International Conference, CPAIOR 2022, Los Angeles, CA, USA, June 20-23, 2022, Proceedings. Cham: Springer International Publishing, 2022: 190-213.

[40] Bdeir A, Boeder S, Dernedde T, et al. RP-DQN: an application of q-learning to vehicle routing problems[C]//KI 2021: Advances in Artificial Intelligence: 44th German Conference on AI, Virtual Event, September 27 – October 1, 2021, Proceedings 44. Springer International Publishing, 2021: 3-16.

[41] Kool W, Van Hoof H, Welling M. Attention, learn to solve routing problems![J]. arXiv preprint arXiv:1803.08475, 2018.

[42] Chen X, Ulmer M W, Thomas B W. Deep Q-learning for same-day delivery with vehicles and drones[J]. European Journal of Operational Research, 2022, 298(3): 939-952.

[43] Ulmer M W, Goodson J C, Mattfeld D C, et al. On modeling stochastic dynamic vehicle routing problems[J]. EURO Journal on Transportation and Logistics, 2020, 9(2).

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

[45] Zhang W, Wang Q, Li J, et al. Dynamic fleet management with rewriting deep reinforcement learning[J]. IEEE Access, 2020, 8: 143333-143341.

[46] Xin L, Song W, Cao Z, et al. Step-wise deep learning models for solving routing problems[J]. IEEE Transactions on Industrial Informatics, 2020, 17(7): 4861-4871.

[47] Falkner J K, Schmidt-Thieme L. Learning to Solve Vehicle Routing Problems with Time Windows through Joint Attention[J]. arXiv e-prints, 2020: arXiv: 2006.09100.

[48] Kool W, Van Hoof H, Welling M. Attention, learn to solve routing problems![J]. arXiv preprint arXiv:1803.08475, 2018.

[49] Xin L, Song W, Cao Z, et al. Multi-decoder attention model with embedding glimpse for solving vehicle routing problems[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(13): 12042-12049.

[50] Zhang K, He F, Zhang Z, et al. Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach[J]. Transportation Research Part C: Emerging Technologies, 2020, 121: 102861.

[51] Xu Y, Fang M, Chen L, et al. Reinforcement learning with multiple relational attention for solving vehicle routing problems[J]. IEEE Transactions on Cybernetics, 2021, 52(10): 11107-11120.

[52] Lin S, Kernighan B W. An effective heuristic algorithm for the traveling-salesman problem[J]. Operations research, 1973, 21(2): 498-516.

[53] Kruk S. Practical Python AI Projects Mathematical Models of Optimization Problems with Google OR-Tools[M]. Apress, 2018.

[54] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.