计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(7): 1670−1689, 2026

利用深度学习辅助算法设计——以排课问题与旅行商问题为例

隋京言[1] 丁诗哲[2,3] 夏博洋[2,3] 刘睿之[2,3] 丁振欣[2,3] 徐黎明[2,3] 王 睿[2,3] 黄旭林[2] 李捷鹏[2,3,5]
张海仓[2,3] 于春功[2,3] 王 超[2,3] 卜东波[2,3,4]

[1](聊城大学人工智能与计算机学院 山东聊城 252000)
[2](处理器芯片全国重点实验室(中国科学院计算技术研究所) 北京 100190)
[3](中国科学院大学 北京 100049)
[4](中原人工智能产业研究院 郑州 450046)
[5](国科大杭州高等研究院 杭州 310024)
(suijingyan@lcu.edu.cn)

Using Deep Learning to Assist Algorithm Design—Taking Course Scheduling Problem and Traveling Salesman Problem as Examples

Sui Jingyan[1], Ding Shizhe[2,3], Xia Boyang[2,3], Liu Ruizhi[2,3], Ding Zhenxin[2,3], Xu Liming[2,3], Wang Rui[2,3], Huang Xulin[2], Li Jiepeng[2,3,5], Zhang Haicang[2,3], Yu Chungong[2,3], Wang Chao[2,3], and Bu Dongbo[2,3,4]

[1](*School of Artificial Intelligence and Computer Science*, *Liaocheng University*, *Liaocheng*, *Shandong* 252000)
[2](*State Key Lab of Processors* (*Institute of Computing*, *Chinese Academy of Sciences*), *Beijing* 100190)
[3](*University of Chinese Academy of Sciences*, *Beijing* 100049)
[4](*Central China Artificial Intelligence Research Institute*, *Zhengzhou* 450046)
[5](*Hangzhou Institute for Advanced Study*, *UCAS*, *Hangzhou* 310024)

Abstract Deep learning has recently demonstrated strong capabilities in pattern discovery and structural inference, offering new possibilities for algorithm design—a field traditionally reliant on human experience and intuition. We focus on the core objective of automatically learning interpretable algorithmic rules under unsupervised settings, using two representative combinatorial optimization problems as cases: course scheduling problem (CSP) and traveling salesman problem (TSP). We propose two deep reinforcement–based methods, Neural-Greedy and Neural-3-OPT. For CSP, Neural-Greedy autonomously learns three optimal greedy rules from large collections of problem instances: prioritizing courses with the earliest finishing times, the latest starting times, and a hybrid of the two. Its underlying mechanism lies in parameter modulation and neuron cooperation, which precisely focuses on critical temporal features and enables the simultaneous acquisition of multiple rules. For TSP, Neural-3-OPT learns efficient 3-opt heuristic autonomously from large collections of problem instances, achieving optimality gaps as low as 0.00%, 0.08%, and 0.74% on random TSP20, TSP50, and TSP100 datasets, respectively. It significantly outperforms the industrial solver OR-Tools and maintains superior performance on the real-world TSPLIB benchmark. This advantage stems from an intelligent decision mechanism of early-stage precise selection, which prioritizes inefficient edges with substantial

收稿日期: 2026-01-05; **修回日期**: 2026-05-28

基金项目: 国家重点研发计划(2020YFA0907000); 国家自然科学基金项目(32270657, 32271297, 82130055, 62072435); 中国科学院青年创新促进会人才项目; 中国科学院计算技术研究所科学计算平台和 GPU 共享平台

This work was supported by the National Key Research and Development Program of China (2020YFA0907000), the National Natural Science Foundation of China (32270657, 32271297, 82130055, 62072435), the Project of Youth Innovation Promotion Association CAS, and the Numerical Calculations by ICT Computer-X center.

通信作者: 卜东波(dbu@ict.ac.cn)

impact on global solution quality and identifies reconnection strategies adaptive to the entire iterative process, thereby avoiding myopic greedy choices and ineffective iterations to achieve fast convergence and high-quality solutions. Overall, our results demonstrate that deep learning can not only approach optimal solution quality but also autonomously learn interpretable algorithmic rules, offering a data-driven paradigm for algorithm design and providing important insights into the development of AI-assisted algorithm design.

Key words algorithm design; deep learning; greedy rule; heuristic rule; course scheduling problem; traveling salesman problem; neural network

摘 要 深度学习在近年来表现出强大的模式挖掘与结构推断能力，为算法设计这一高度依赖经验与直觉的领域提供了新的可能性。聚焦“无监督条件下自主学习可解释性算法规则”这一核心目标，以排课问题和旅行商问题这2个经典组合优化问题为典型案例，提出2种基于深度强化学习的方法Neural-Greedy与Neural-3-OPT。针对排课问题，Neural-Greedy可从大量问题实例中自主学出3条最优贪心规则（优先选择最早下课的课程，优先选择最晚上课的课程及二者的混合策略）；其核心机理是通过参数调控与神经元协同实现时间节点关键特征的精准聚焦，进而完成多规则同步习得。针对旅行商问题，Neural-3-OPT可从大量问题实例中自主学出高效的3-opt启发式策略，在TSP20、TSP50和TSP100随机数据集上的最优性差距分别低至0.00%、0.08%和0.74%，显著优于工业级求解器OR-Tools，且在真实数据集TSPLIB上保持更优性能；其优势源于“初期精准选优”的智能决策机理，通过优先识别对全局解质量影响显著的低效边、识别适配迭代全程的重连策略，避免单步贪心选择与无效迭代，实现快速收敛与高质量求解。研究表明，深度学习不仅能逼近最优求解性能，更能自主学出具备可解释性的算法规则，为算法设计提供了“数据驱动”新范式，也为“AI辅助算法设计”的发展提供了重要思路。

关键词 算法设计；深度学习；贪心规则；启发式策略；排课问题；旅行商问题；神经网络

中图法分类号 TP311.13；TP309

DOI: 10.7544/issn1000-1239.202660017 **CSTR:** 32373.14.issn1000-1239.202660017

算法设计在诸多科学与工程问题中发挥核心作用，而贪心算法与启发式算法的设计往往高度依赖算法设计人员的经验、直觉与对问题结构的深刻理解。传统的算法设计流程通常由算法设计人员通过分析若干典型实例、发现结构模式、推测有效规则，再不断验证与修正，过程耗时且具有高度不确定性。近年来，随着深度学习在物理学规律发现、蛋白质结构预测、数学猜想辅助、矩阵乘法算法发现等任务中表现出类直觉式的模式抽象能力[1-10]，以数据驱动的方式辅助算法设计逐渐成为可能。

已有研究表明，神经网络能够学习任务的结构本质，如推断拓扑不变量、归纳物理公式、证明数学理论和预测蛋白质折叠结构等[1-10]。这些成果启发我们：深度学习是否也能够用于从问题实例中自主学习算法规则；特别是，能否在完全不依赖解监督的情况下，通过强化学习让模型自行探索并学习出有效的贪心规则或启发式策略。这一问题对于组合优化算法设计具有重要理论价值和应用前景。

近年来，深度学习与组合优化的交叉研究已形成若干代表性技术方向，但与本文目标存在本质差异。“Learning to optimize”[11]聚焦学习优化算法的参数更新规则，属于元优化层面；本文则聚焦算法设计规则的学习，属于算法设计方法论层面，关注如何让AI自主发现求解问题的有效规则。“Neural combinatorial optimization”[12-13]采用端到端方式直接预测解，追求求解性能但决策过程黑盒化；本文虽同样训练求解模型，但进一步分析模型预测背后的决策规则，并通过统计验证与机理分析确保规则的可解释性与有效性。“RL for heuristics discovery”[14]主要关注求解性能提升，将习得策略作为黑盒工具；本文则强调规则学习结果的可解释性验证，不仅证明规则有效，更揭示其决策机制，建立从数据到规则、从规则到机理的完整技术路径。

本文以2个经典组合优化问题——排课问题（course scheduling problem, CSP）与旅行商问题（traveling salesman problem, TSP）——为例进行研究。选择这2个问题的理由包括：1）其结构清晰但设计高质量的贪心算法与启发式策略并不容易；2）它们分别

代表区间调度类问题与路径优化类问题，具有典型性；3)均存在已知的最优算法或强竞争基线，可用于验证深度学习是否真正学到了有效规则。

在排课问题上，本文提出 Neural-Greedy 方法，通过深度强化学习训练一个神经网络模型，使其逐步选择课程并构造无冲突的最大集合。在无监督的设置下，模型自主学出了 3 条最优贪心规则。进一步分析显示，这些规则仅依赖课程的时间节点特征，而非课程时长、冲突数等非关键因素，模型通过参数自动聚焦关键特征并形成内在规则。

在旅行商问题上，本文提出基于深度强化学习的 Neural-3-OPT 方法，通过学习选择低效边与选择重连方式的策略，自动形成有效的 3-opt 启发式规则。实验表明，该方法不仅能在小规模 TSP 中达到最优解，在中大规模问题中也表现出比传统启发式(如最近邻、插入规则)和工业级求解器(OR-Tools)更优的性能，并能在真实 TSP 数据集 TSPLIB 上保持强泛化能力，证明其可适配实际场景需求。进一步的策略分析表明，Neural-3-OPT 具备显著的智能决策优势，其核心在于“初期精准选优”的决策机理，求解初始阶段即可精准筛选能大幅降低路径长度的改进操作，即优先识别对全局解质量影响显著的低效边，并匹配适配迭代全程的重连策略，避免单步贪心选择与无效迭代，从而在较少步数内快速锁定高质量解；相比之下，传统 3-opt 算法的改进操作(删边与重连)选择具有随机性，既无法聚焦关键低效边、筛选高效改进操作，也缺乏全局适配的重连逻辑，使得迭代过程中存在大量无增益操作，导致求解效率与解质量均受限。

本文的核心创新体现在 3 个递进层次。1)范式层提出“深度学习辅助算法设计”新范式，从人工设计规则转向 AI 自主学习规则，从“用模型求解”转向“学规则并解释规则”。2)能力层验证深度强化学习可在无标注解的条件下，自主习得可解释的算法设计规则；排课问题上习得 3 条通用最优贪心规则，TSP 问题上习得有效的 3-opt 启发式策略。3)机理解释层揭示神经网络学习算法设计规则的内在机制，通过参数可视化证明特征聚焦机制，通过决策轨迹分析揭示初期精准选优模式，解释模型如何学规则、为何有效，而非仅报告求解性能。与现有工作相比，本文填补了“让 AI 自主学出可解释的、有效的算法设计规则并验证其内在机制”这一空白，为突破传统算法设计对专家经验的依赖提供了可验证、可复现的技术路径。

本文的具体贡献有 4 个方面：

1)提出“深度学习辅助算法设计”新范式，验证了深度学习在无监督条件下，可从大量问题实例中自主学出有效且可解释的算法规则，为突破传统算法设计对专家经验与直觉的依赖提供了新路径；

2)提出求解排课问题的 Neural-Greedy 方法，该方法基于深度强化学习可自主学出 3 条基于时间节点的最优贪心规则，实现无冲突最大课程集合的高效构建；

3)提出求解旅行商问题的 Neural-3-OPT 方法，该方法基于深度强化学习可自主学出高效的 3-opt 启发式策略，其求解性能显著优于传统人工设计的启发式算法，且在真实世界实例中具备强泛化能力；

4)深入剖析了神经网络学习算法规则的可解释性，揭示了 Neural-Greedy 通过参数权重调控与神经元组合实现关键特征聚焦的核心机制，明确了 Neural-3-OPT“初期精准选优”的智能决策优势与性能提升机理。

1 背景知识

排课问题与旅行商问题是组合优化领域的经典 NP 难问题，其贪心规则与启发式规则的设计高度依赖专家经验与直觉，是算法设计领域的典型挑战[15]。二者不仅在计算机科学、运筹学等领域具有重要理论研究价值，还广泛落地于实际场景：排课问题则可服务于学校课程编排、企业会议室调度等资源分配场景；旅行商问题可转化为芯片电路布线、物流路径规划、DNA 测序排序等问题的求解框架[16]，具有广泛应用价值。本节将依次梳理这 2 类问题的定义、传统求解规则，以及适配“无监督规则学习”需求的神经网络与深度学习基础，为后文提出 Neural-Greedy、Neural-3-OPT 方法提供理论与技术支撑。

1.1 排课问题的定义及贪心规则

排课问题的定义为：给定n门课程的集合，第i门课程起始时间为s_i、结束时间为f_i($1 \leqslant i \leqslant n$)，需从集合中选择最大规模的课程子集，满足子集中任意两门课程的时间区间无重叠(无冲突)。图 1 所示为课程规模$n = 10$的排课问题实例。红色矩形代表构成最优解的 5 门无冲突课程。

需要说明的是，本文采用经典区间调度版本的排课问题进行研究。选择研究该简化版本的排课问题是基于以下考量：首先，聚焦规则学习的本质验证，该问题是贪心算法设计的教科书级案例，存在已知最优贪心规则，便于建立可验证的基准，若模型连最

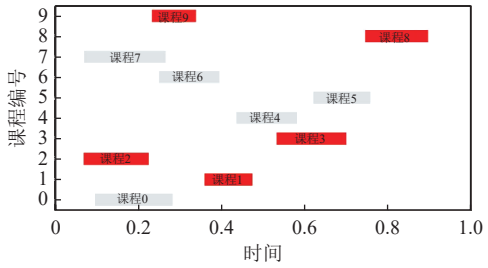


Fig. 1 An instance of CSP with 10 courses

图 1 包含 10 门课程的排课问题示例

优规则已知的问题都无法习得，则更难处理复杂场景；其次，控制变量验证方法——简化问题，可排除多约束耦合的干扰，清晰展示神经网络从纯时间特征中自主学习规则的能力。本文的核心贡献并非针对复杂约束排课问题的求解性能提升，而是验证“无监督规则学习”这一范式的可行性。选择结构清晰、规则已知的问题，恰恰是为了建立可严格验证的基准，使“模型是否真正学会规则”这一问题可被客观检验。

针对该问题，学界已提出动态规划法与贪心算法 2 类主流求解思路。其中，贪心算法因实现简洁、求解高效，成为首选方案，但优质贪心规则的设计高度依赖对问题结构的深刻洞察，需算法设计人员分析典型实例、推测规则、反复验证修正，过程耗时且不确定性强。算法设计领域曾探索过 5 类贪心规则：

规则 1 优先选择最早上课（s_i最小）的课程；

规则 2 优先选择时长最短（$f_i - s_i$最小）的课程；

规则 3 优先选择冲突最少的课程；

规则 4 优先选择最早下课（f_i最小）的课程；

规则 5 优先选择最晚上课（s_i最大）的课程。

其中，规则 1~3 被证实无法保证求出最优解，而规则 4 和 5 则被证实是该问题的 2 条最优贪心规则，可求出最大无冲突课程子集。

1.2 旅行商问题定义及启发式规则

旅行商问题的定义为：给定n座城市的坐标集合（或任意 2 座城市间的距离矩阵），求解一条从任意城市出发、遍历所有城市恰好 1 次、最终返回起始城市的最短回路。图 2 所示为城市规模$n = 50$的旅行商问题实例。蓝色圆点表示城市，灰色线条表示城市之间的路径，红色环路表示该示例的最优解。

旅行商问题的求解复杂度随城市规模呈指数增长：n座城市的可行路线有$(n-1)!/2$条[17]。具体而言，$n = 20$时，可行回路约为10^{16}条；$n = 50$时，达到约10^{62}条；$n = 100$时，更是突破10^{155}条。这直接导致精确算法在求解大规模问题时效率低下。

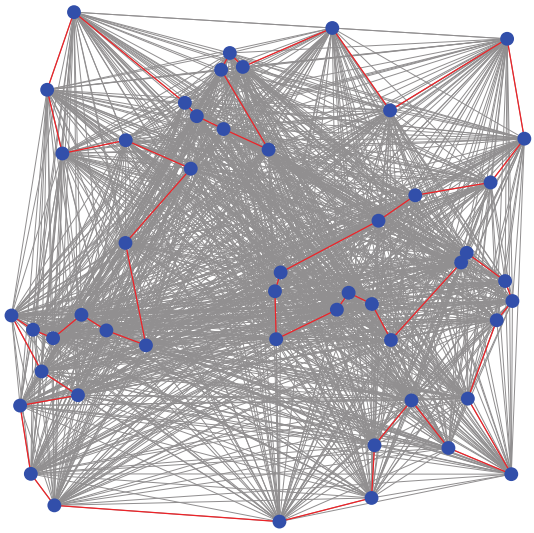


Fig. 2 An instance of TSP with 50 cities

图 2 含有 50 个城市的旅行商问题示例

对于小规模旅行商问题，可通过穷举法[18]、分支限界法[19]、动态规划法[20]等精确算法求解；但针对实际应用中更常见的大规模问题，上述方法因计算资源消耗过大、求解效率过低而不可行。此时，启发式算法成为主流选择——其核心优势在于以可接受的计算成本快速获取高质量近似解，但优质启发式规则的设计同样依赖专家经验与灵感，需经历“实例分析—规则推测—验证修正”的耗时过程，是算法设计的核心挑战之一。这一痛点也凸显了引言提出的“利用深度学习自主学习启发式策略”的必要性与价值。

当前主流的旅行商问题启发式方法包括最近邻算法[21]、插入算法[21]和局部搜索算法[18]等，其中局部搜索类的 2-opt 与 3-opt 规则因求解质量高、通用性强，被广泛应用[22-24]。图 3 为 2 类规则的核心思想示意图（图中线段长度不代表实际路径距离）：

1）2-opt 规则：从当前路径中删除 2 条低效边（图 3 中蓝线），将之替换为 2 条高效边（图 3 中红线），通过路径重构获得更短回路；

2）3-opt 规则：从当前路径中删除 3 条低效边（图 3 中蓝线），通过 3 条新边重新连接路径片段（图 3 中红线），生成更优回路。

理论与实践均证明，3-opt 的搜索空间更大，更易找到优质解[23-24]，但求解难度显著高于 2-opt：2-opt 仅存在 1 种有效重连方式，而 3-opt 的有效重连方式多达 7 种[25]。在实际迭代优化过程中，如何精准选择待删除的低效边、如何从 7 种重连方式中筛选最优方案，直接决定求解效率与解质量——这一核心痛点再次强化了传统启发式规则设计对“经验灵感”的依

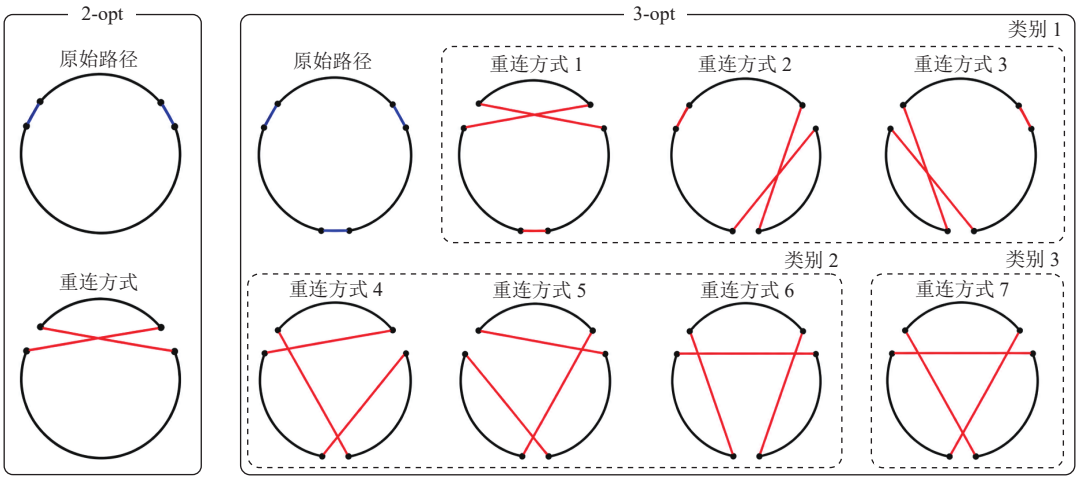


Fig. 3 2-opt and 3-opt heuristics for solving TSP[25]

图 3 求解旅行商问题的 2-opt 和 3-opt 启发式规则[25]

赖，也为后文 Neural-3-OPT 方法的提出提供了核心切入点：通过深度强化学习让模型自主学习“删边-重连”策略，替代人工决策。

1.3 神经网络与深度学习基础

随着深度学习技术的发展，卷积神经网络(CNN)[26]、循环神经网络(RNN)[27]、图卷积神经网络(GCN)[28]、图注意力神经网络(GAT)[29]、Transformer[30]等模型相继涌现，其在模式识别、特征抽象、复杂问题建模等领域表现出强大的类直觉式模式挖掘能力。与传统“手工设计规则”的求解思路不同，深度学习采用“数据驱动”范式，可通过学习大量问题实例的隐含特征，自动捕捉问题的本质规律，无需人工干预。

其中，深度强化学习的无监督学习特性，完美适配算法设计的核心场景：算法设计人员在面对未知解的问题时，仅能依托问题本身特征推导规则。这也正是本文选择深度强化学习框架的核心原因：通过训练神经网络模型，使模型仅通过学习问题原始特征，自主学习出适配的贪心规则或启发式策略。这一过程无需将已知解作为训练标签，既模拟了人类算法设计人员“从问题特征到规则构建”的思考过程，又突破了传统设计对经验的依赖，为“算法设计新范式”提供了关键技术支撑。

2 排课问题的研究方法、实验结果及讨论

2.1 排课问题的研究方法

本文提出的 Neural-Greedy 方法框架如图 4 所示，核心思路为将排课问题转化为序列决策问题，通过深度强化学习训练神经网络，使其从问题特征中自主学习选课策略，最终输出无冲突的最大课程集合。该框架采用“编码器-解码器”结构，具体设计如图 4 所示。

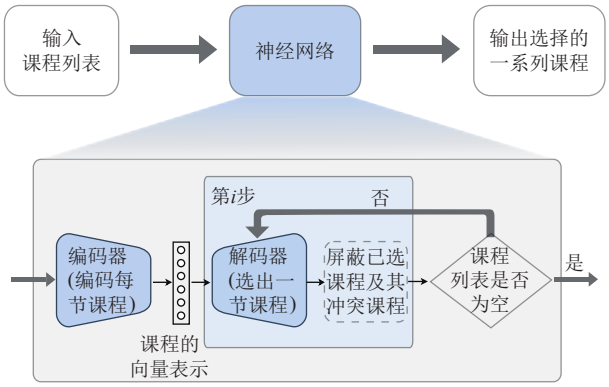


Fig. 4 Framework of Neural-Greedy method for solving CSP

图 4 求解排课问题的 Neural-Greedy 方法框架

2.1.1 马尔可夫决策过程建模

为适配强化学习训练范式，将排课问题建模为马尔可夫决策过程，各核心要素定义如下：

1)状态(State)：当前可选的课程集合(屏蔽已选课程及冲突课程后的剩余课程)。

2)动作(Action)：从当前可选课程中选择一门课程。

3)状态转移(Transition)：屏蔽已选课程及与其时间冲突的课程，更新可选课程集合。

4)奖励(Reward)：每选择一门课程获得+1 奖励，累计奖励等于最终选出的无冲突课程数量。

定义一个随机策略 $p_\theta(\pi|s)$，表示在状态 s 下生成选课序列 π 的概率，其参数化形式为：

$$p_\theta(\pi|s)=\prod_{t=1}^{k}p_\theta(\pi_t|s,\pi_{1:t-1}),$$

其中，θ表示神经网络可学习参数，s表示一个实例，$\pi=(\pi_1,\cdots,\pi_k)$表示长度为k的无冲突课程序列。

2.1.2 神经网络结构设计

模型采用基于 Transformer 的编码器[30]与基于指针网络（Ptr-Net）的解码器[31]，适配排课问题的特征建模与序列选择需求。

1）编码器。基于 Transformer 的多头注意力机制与全连接前馈层构建，删除位置编码模块。输入为每门课程的起始时间s_i与结束时间f_i，输出为课程的嵌入向量，实现特征的有效抽象。具体计算如下。

① 输入层

对于n门课程的排课问题，课程i的时间区间$\boldsymbol{x}_i=[s_i,f_i]$，线性映射到$d$维初始嵌入：

$$\boldsymbol{h}_i^0=\boldsymbol{W}_0\boldsymbol{x}_i+\boldsymbol{b}_0,$$

其中$\boldsymbol{W}_0\in\mathbb{R}^{d\times 2}$与$\boldsymbol{b}_0\in\mathbb{R}^d$为可学习参数。

② 注意力层

通过N个注意力层迭代更新节点嵌入，每层包含多头注意力子层（multi-head attention layer, MHA）与前馈子层（feed-forward layer, FF）。每个子层都添加了一个跨层连接和批量归一化（batch normalization, BN）。第$l\in\{1,2,\cdots,N\}$层的计算为：

$$\tilde{\boldsymbol{h}}_i=BN^l(\boldsymbol{h}_i^{l-1}+MHA_i^l(\boldsymbol{h}_{1:n}^{l-1})),$$

$$\boldsymbol{h}_i^l=BN^l(\tilde{\boldsymbol{h}}_i+FF^l(\tilde{\boldsymbol{h}}_i))。$$

③ 全局嵌入

将最终层所有节点嵌入取平均，得到编码全局信息的课程嵌入向量：

$$\bar{\boldsymbol{h}}=\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{h}_i^N。$$

2）解码器。采用指针网络结构，迭代生成选课序列。每个解码步利用编码器输出的课程嵌入向量与上一步选中课程的嵌入向量，计算当前可选课程的概率分布，按概率选择课程并更新可选课程集合，直至无课程可选。具体计算如下。

解码步t的上下文向量构造如下：

$$\boldsymbol{h}_{(c)}^N=\begin{cases}[\bar{\boldsymbol{h}},\boldsymbol{h}_{\pi_{t-1}}^N], & t>1,\\ [\bar{\boldsymbol{h}},\boldsymbol{v}_0], & t=1,\end{cases}$$

其中，$\boldsymbol{v}_0\in\mathbb{R}^d$为可学习的初始占位向量，$[\cdot,\cdot]$表示水平拼接。

基于上下文向量计算查询向量（代表了“当前需要选什么课程”的语义），并与各课程的键向量（代表了“这门课程是什么”的语义）计算兼容性分数：

$$\boldsymbol{q}_{(c)}=\boldsymbol{W}_Q\boldsymbol{h}_{(c)}^N,$$

$$\boldsymbol{k}_i=\boldsymbol{W}_K\boldsymbol{h}_i^N,$$

$$u_{(c)i}=\frac{\boldsymbol{q}_{(c)}^{\mathrm{T}}\boldsymbol{k}_i}{\sqrt{d_k}},$$

其中$\sqrt{d_k}$是为缩放因子，d_k表示查询/键向量的维度。该缩放操作将点积的方差归一化为 1，防止因向量维度过大导致兼容性分数过大，进而使 Softmax 输出趋向极端（接近 0 或 1）并造成梯度消失，从而保障模型训练的稳定性。

计算选择概率之前屏蔽已选课程及与其时间冲突的课程，确保输出序列的无冲突性：

$$u_{(c)i}=-\infty,\quad \text{if } i\in\{\pi_1,\cdots,\pi_{t-1}\}$$

或

$$\exists\, t'<t:[s_i,f_i]\cap[s_{\pi_{t'}},f_{\pi_{t'}}]\neq\varnothing。$$

之后，通过 Softmax 归一化得到选择概率，兼容性分数经 tanh 裁剪以控制输出范围：

$$p_i=\frac{\exp(C\cdot\tanh(u_{(c)i}))}{\sum_{j=1}^{n}\exp(C\cdot\tanh(u_{(c)j}))},$$

其中$C=10$为裁剪超参数。

解码器在每一步输出概率最大的课程。

2.1.3 训练目标与优化算法

将目标函数设为累计奖励的期望，并将优化目标定义为最大化累计奖励。在这里，由于每选出一门课，累计奖励增加 1，因此累计奖励就等于选出的课程数量。所以，最大化累计奖励等价于最大化选出的不冲突的课程数量。

目标函数定义为：

$$\mathcal{L}(\theta|s)=\mathbb{E}_{p_\theta(\pi|s)}[L(\pi)],$$

其中$L(\pi)$为选课序列π的长度（即无冲突课程数量）。

采用强化学习中名为 REINFORCE 的策略梯度算法[32]训练模型，梯度计算公式为：

$$\nabla\mathcal{L}(\theta|s)=\mathbb{E}_{p_\theta(\pi|s)}[(L(\pi)-b(s))\nabla\log p_\theta(\pi|s)],$$

其中，$b(s)$表示基线模型的输出。该基线是通过确定性贪心展开（deterministic greedy rollout）方式得到的，即基于训练过程中当前最佳策略模型得到。引入该基线可降低训练过程中的方差，并加快训练速度。本文采用自适应矩估计优化器（adaptive moment estimation, ADAM）[33]作为优化器，完整训练流程如算法 1 所示。

算法 1. 求解 CSP 的 Neural-Greedy 方法训练过程。

输入：训练集，迭代轮数（*epoch*）E，每轮步数T，批次（*batch*）大小B，显著性系数α，学习率λ。

① 初始化神经网络及基线网络的参数θ，$\theta^{BL}\leftarrow\theta$；

② for $epoch = 1,\cdots,E$ 运行
③ for $step = 1,\cdots,T$ 运行
④ $s_i \leftarrow RandomInstance(), \forall i \in \{1,\cdots,B\}$;
⑤ $\pi_i \leftarrow SampleRollout(s_i, p_\theta), \forall i \in \{1,\cdots,B\}$;
⑥ $\pi_i^{BL} \leftarrow GreedyRollout\left(s_i, p_\theta^{BL}\right), \forall i \in \{1,\cdots,B\}$;
⑦ $\nabla\mathcal{L} = \sum_{i=1}^{B}\left(L(\pi_i) - L\left(\pi_i^{BL}\right)\right)\nabla_\theta \log p_\theta(\pi_i)$;
⑧ $\theta \leftarrow Adam(\theta, \nabla\mathcal{L})$;
⑨ end for
⑩ if 单侧配对 T 检验$(p_\theta, p_\theta^{BL}) < \alpha$
⑪ $\theta^{BL} \leftarrow \theta$;
⑫ end if
⑬ end for

2.2 排课问题数据集的构建

为验证“无监督学习算法规则”的核心目标，训练集与验证集仅包含问题本身的特征(无标注解)，测试集则包含问题特征与最优解(用于性能评估)，数据集构建细节如下：

1)实例生成。对任一个问题实例来说，每门课程的起始时间s_i与结束时间f_i在[0,1]区间随机生成，通过控制课程时长控制冲突密度(避免解长度过短)。具体来说，课程规模n= 10或n = 20时，课程时长限制为[0.1,0.2]；n= 50或n = 100时，课程时长限制为[0.05,0.1]。单个实例表示为$[[s_1,f_1],[s_2,f_2],\cdots,[s_n,f_n]]$。

2)测试集构建。每个测试集含 1 000 个实例。对于小规模问题(n = 10)，采用暴力枚举法获取唯一最优解，确保评估准确性。对于中大规模问题($n \geqslant 20$)采用动态规划法求出最优解(允许存在多个最优解)，只要模型输出任一个最优解即判定为求解正确。

2.3 排课问题的实验结果和讨论

2.3.1 实验设置与性能评估

实验针对n = 10，20，50，100四种课程规模展开，各规模模型均训练 100 轮(epoch)，每轮训练集和验证集分别包含 128 000 和 10 000 个随机生成的相应规模的实例。具体实验设置如表 1 所示。

测试阶段采用贪心解码策略(每步选择概率最大的课程)，评估指标包括“解的平均长度”“与最优解的平均差距”“求出最优解的比例”。其中，“解的平均长度”表示神经网络求出的测试集中所有问题的解的长度的平均值，“与最优解的平均差距”表示神经网络求出的测试集中所有问题解与最优解的差距的平均值。

2.3.2 实验结果与比较

实验结果如表 2 所示，模型在不同规模问题上均表现优异：小规模问题(n = 10，20)上实现 100% 最优解命中率，解的平均长度与最优解完全一致；中大规模问题(n = 50，100)上与最优解的平均差距仅为 0.256% 和 0.111%，最优解命中率分别达 98.9% 和 97.2%，验证了 Neural-Greedy 框架的有效性。

Table 1 Experimental Setup for CSP

表 1 排课问题实验设置表

分类	参数名称	参数值
网络结构	编码器	Transformer，多头注意力层数 N=3，头数 $head$=8，维度 d=128
	解码器	Ptr-Net，维度 d=128
训练设置	问题规模n	20/50/100
	学习率λ	1E−4
	批次大小B	512
	每轮步数T	2 500
	训练轮数E	100
	显著性系数α	0.05

Table 2 Experimental Results of CSP

表 2 排课问题的实验结果

问题规模	求出的解的平均长度	最优解的平均长度	与最优解的平均差距/%	求出最优解的比例/%
10	4.761	4.761	0.000	100
20	5.003	5.003	0.000	100
50	10.791	10.803	0.256	98.9
100	12.459	12.491	0.111	97.2

2.3.3 实验结果分析与讨论

为验证模型是否真正学习到有效贪心规则，针对n = 10的测试集展开规则统计分析(对比 1.1 节提及的 5 类经典贪心规则)，核心结论如下。

1)解码步级规则统计。如图 5 所示，模型在所有解码步中仅选择“最早下课的课程”(规则 4，1 535 次)和“最晚上课的课程”(规则 5，3 226 次)，未触发规则 1~3 或其他未知规则，表明模型学习的规则仅依赖时间节点特征(s_i，f_i)，与课程时长、冲突数量等无关。

2)实例级规则统计。如图 6 所示，1 000 个测试实例可分为 3 类：119 个实例全程采用规则 4(11.9%)、520 个实例全程采用规则 5(52.0%)、361 个实例混合采用规则 4 与规则 5(36.1%)，进一步验证模型仅学习时间节点相关规则。

结合图 7 的实例占比分布可进一步归纳模型在训练过程中自发习得的 3 条最优贪心规则：

规则 4 优先选择最早下课的课程。

规则 5 优先选择最晚上课的课程。

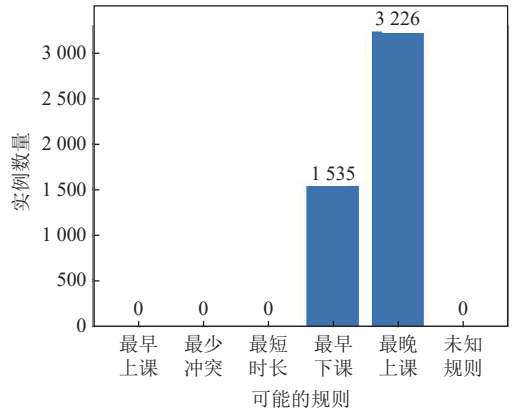


Fig. 5 Number distribution of rules used for solving CSP testset

图 5 求解排课问题测试集时使用的规则的数量分布

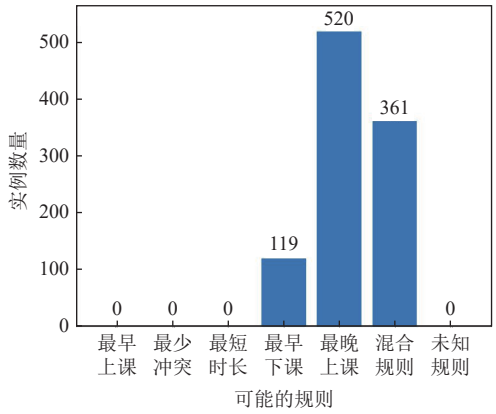


Fig. 6 Number distribution of CSP instances based on different greedy rules

图 6 不同贪心规则下排课问题实例的数量分布

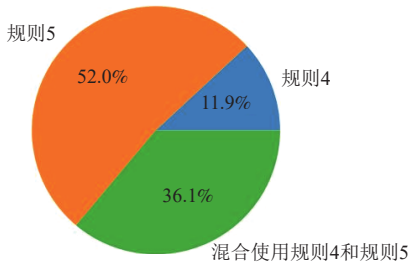


Fig. 7 Number percentage of CSP instances solved by different rules

图 7 基于不同规则求解的排课问题实例数量占比

规则 6 混合使用规则 4 与规则 5。

3)规则有效性验证。采用上述 3 类规则(规则 4、规则 5、规则 6)单独求解测试集，均实现 100% 最优解命中率，证明模型自发学习的 3 条规则均为最优贪心规则。

通过 3 个典型实例可视化模型的选课决策流程，直观呈现规则的应用过程，如图 8~10 所示。

图 8 为规则 4 的应用实例，模型在每个解码步均

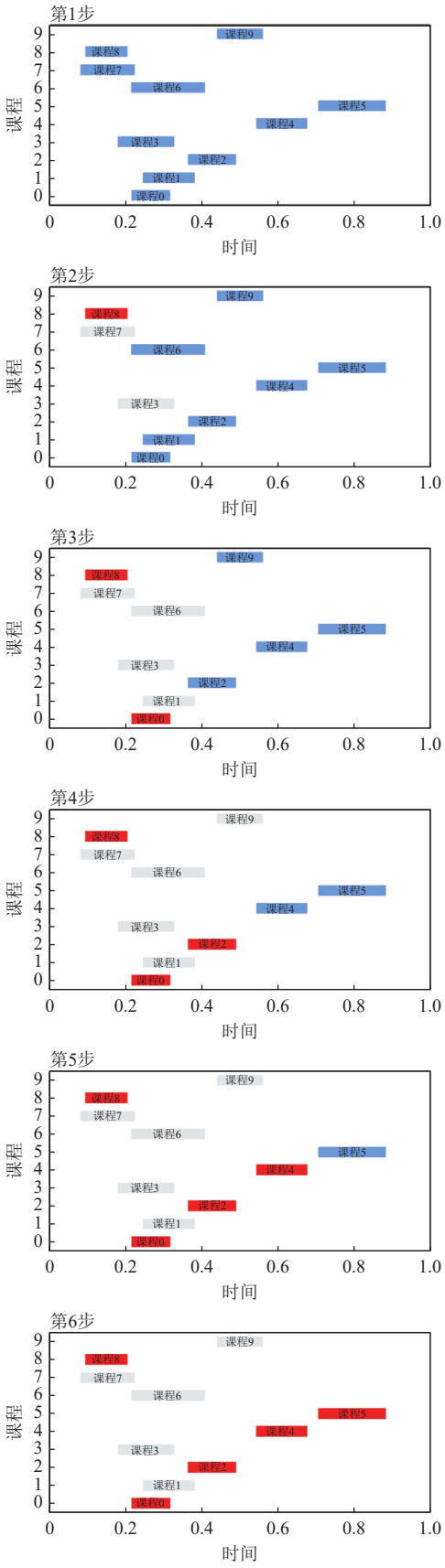


Fig. 8 Prioritize selecting the course that finishes earliest based on rule 4

图 8 基于规则 4 优先选择最早下课的课程

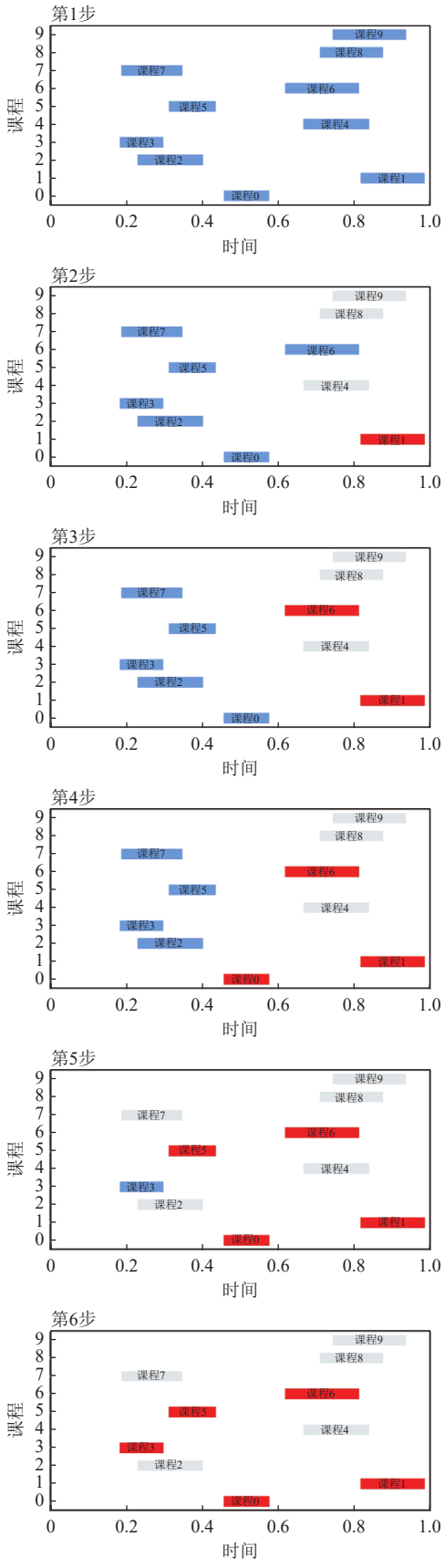


Fig. 9 Prioritize selecting the course that starts latest based on rule 5

图 9 基于规则 5 优先选择最晚上课的课程

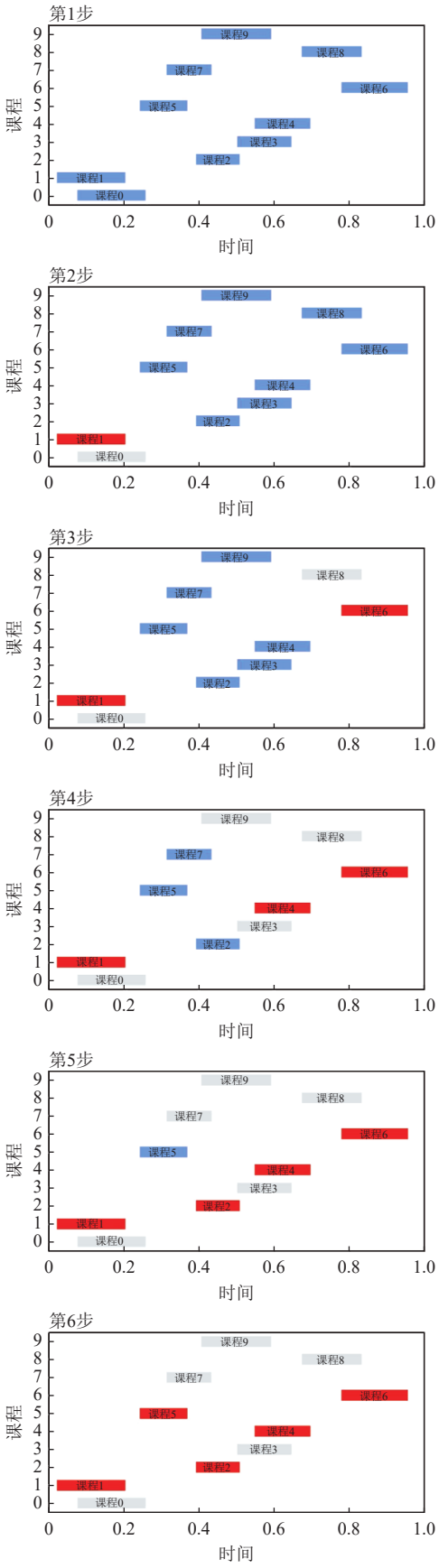


Fig. 10 Mixing use of rule 4 and rule 5 based on rule 6

图 10 基于规则 6 混合使用规则 4 和规则 5

选择当前可选课程中最早下课的课程(如第 2 步选课程 8、第 3 步选课程 0 等)，最终输出无冲突最优解。该类实例占测试集的 11.9%，验证了模型对规则 4 的有效习得。

图 9 展示了规则 5 的应用过程，模型每一步均选择当前可选课程中最晚上课的课程(如依次选课程 1、6、0 等)，对应实例占测试集的 52.0%，验证了模型对规则 5 的有效习得。

图 10 为规则 6 的混合决策实例：模型在部分解码步骤采用规则 4(如第 2 步选最早下课的课程 1)，部分步骤采用规则 5(如第 3~5 步选最晚上课的课程 6、4、2)。该类实例占测试集的 36.1%，验证了模型对规则 6 的有效习得。

2.3.4　可解释性分析：特征聚焦机制揭示

前述分析已证实，模型学出的 3 条贪心规则仅依赖课程的上课与下课时间特征。为揭示模型“聚焦时间节点特征学习规则”的内在机制，采用简化的全连接网络展开分析(便于参数可视化)，核心发现如下：

1)网络参数可视化。输入层与隐藏层以及隐藏层与输出层的参数均呈现“正负交替”特征，表明神经元通过权重调控放大关键时间特征和抑制无关特征，实现时间节点特征的精准聚焦。

2)子网络拆分验证。将全连接网络拆分为 2 个子网络(屏蔽不同神经元)，可分别稳定输出规则 4(优先最早下课)和规则 5(优先最晚上课)的决策逻辑，证明模型可通过不同神经元的组合学习差异化规则，进而实现多规则的同步习得。

具体来说，实验构建仅含 1 个隐藏层(5 个神经元)的全连接网络，如图 11 所示。首先采用强化学习方式训练网络。结果显示，该模型求解性能显著下降：针对规模 $n=10$ 的问题，解与最优解的差距达 34.12%；针对 $n=5$ 的问题，解与最优解的差距达 21.10%。推测原因为单隐藏层全连接网络的学习能力有限，难以在无监督条件下精准习得规则。因此，后续改用监督学习方式训练该网络，以保障模型具备基础的规则学习能力，支撑后续机制分析。

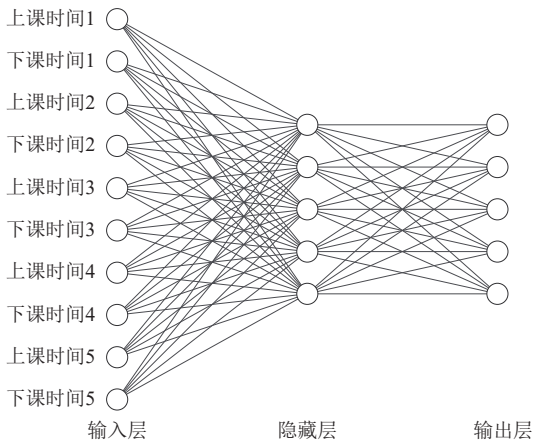


Fig. 11　Illustration of a fully connected neural network for solving CSP

图 11　求解排课问题的全连接神经网络示意图

监督学习实验针对 $n=5$ 的排课问题展开，网络输入层的 10 个神经元分别对应 5 门课程的上课时间 s_i 与下课时间 f_i，课程特征为 $[s_1,f_1,s_2,f_2,\cdots,s_5,f_5]$，取值范围限定于[0,1]；标签向量为 $[O_1,O_2,\cdots,O_5]$，其中 $O_i=0/1$，1 表示课程 i 属于最优解，0 则相反。训练集的最优解通过暴力枚举法获取，且仅保留具有唯一最优解的实例。

模型训练完成后，采用迭代策略求解排课问题：每一步选择网络输出概率最大的课程，屏蔽与其时间冲突的课程，将剩余课程重新输入网络，直至无课程可选，具体流程如图 12 所示。

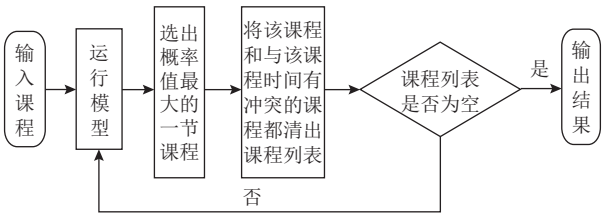


Fig. 12　Illustration of iterative process of solving CSP by the fully connected neural network

图 12　利用全连接神经网络求解排课问题迭代过程示意图

实验结果表明，该全连接网络同样学出 3 条贪心规则(规则 4~6)，模型准确率达 91.26%(准确率定义为：网络输出概率最大的课程属于最优解的比例)。需说明的是，本文未通过增加隐藏层或神经元数量提升准确率，而是在准确率与网络复杂度间权衡，以“较高准确率+简单网络结构”的组合，保障机制分析的直观性与有效性。

图 13 直观呈现了输入层与隐藏层间的参数分布特征：横轴对应输入层的 10 个神经元(分别映射 5 门课程的 s_i 与 f_i)，纵轴为参数值；1 种颜色对应 1 个隐藏层神经元，共 5 种颜色，每种颜色的 10 个圆点代表该隐藏层神经元与输入层所有神经元的参数大小，对应颜色的曲线则表征参数变化趋势(如红色圆点及曲线对应隐藏层 3 号神经元与输入层各神经元的参数匹配关系及权重变化规律)。从图 13 可清晰观察到，各颜色对应的参数呈现显著的“正负交替”特

征。基于这一观察,我们推测输入层向隐藏层传递特征时,可能对同一课程的s_i与f_i进行差异化加权——对某一特征赋予正权重(激活)以及对另一特征赋予负权重(抑制),从而实现时间特征的对比选择。这一参数模式与贪心规则仅依赖单一时间特征(最早下课或最晚上课)的特性相吻合,为理解神经网络如何聚焦关键特征提供了一种可能的解释视角。此外,不同隐藏层神经元对应的曲线波峰、波谷位置存在差异,说明不同神经元会针对性地“强调”特定课程的时间节点特征,实现对关键时间信息的精准筛选,为规则学习奠定基础。

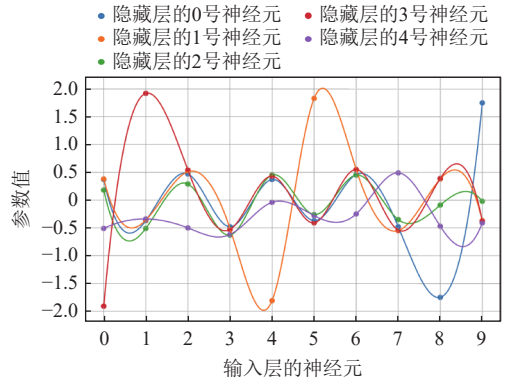


Fig. 13 Visualization of parameters between the input layer and hidden layer of fully connected neural network for solving CSP

图 13 求解排课问题的全连接神经网络的输入层与隐藏层之间的参数可视化

同理,图 14 展示了隐藏层与输出层间的参数分布,横轴为隐藏层的 5 个神经元,纵轴为参数值。参数分布同样呈现大小交替的特征,表明隐藏层神经元会进一步筛选与放大关键时间特征,最终实现对时间节点特征的精准聚焦。

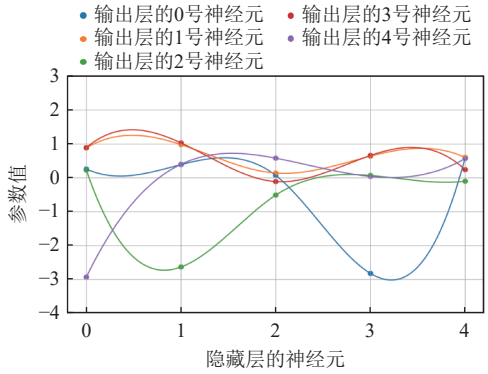


Fig. 14 Visualization of parameters between the hidden layer and output layer of fully connected neural network for solving CSP

图 14 求解排课问题的全连接神经网络的隐藏层与输出层之间的参数可视化

为探究模型同时习得规则 4 与规则 5 的原因,本文对全连接网络进行结构拆分,以一个随机生成的$n=5$的排课问题([0.102, 0.330, 0.329, 0.535, 0.756, 0.973, 0.354, 0.603, 0.467, 0.789])为例开展实验。

拆分得到 2 个子网络,如图 15 所示:子网络 1 屏蔽隐藏层神经元 3,剩余神经元 0、1、2、4,其输出的课程选择概率中,最早下课课程[0.102,0.330]的概率最高(40.52%);子网络 2 屏蔽隐藏层神经元 1,剩余神经元 0、2、3、4,其输出的课程选择概率中,最晚上课课程[0.756,0.973]的概率最高(40.81%)。

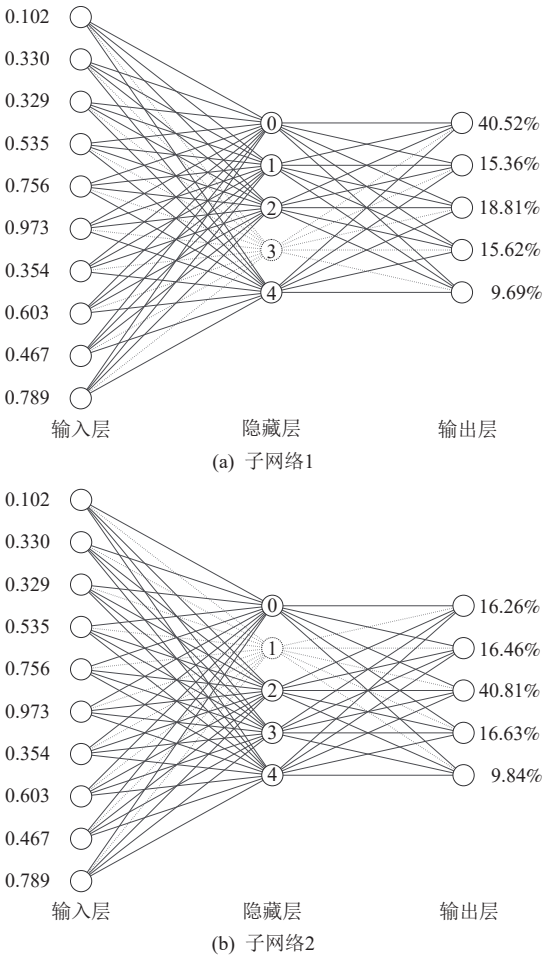


Fig. 15 Illustration of two subnetworks after splitting the fully connected neural network for solving CSP

图 15 求解排课问题的全连接神经网络拆分后的 2 个子网络示意图

综上,神经网络之所以能够习得贪心规则,核心在于通过参数权重调控实现关键特征聚焦,并将规则逻辑隐含于网络参数与结构之中;而多规则学习的实现,则依赖于不同神经元组合对差异化时间特征的选择性放大。

3 旅行商问题的研究方法、实验结果及讨论

3.1 旅行商问题的研究方法

旅行商问题的经典启发式求解思路为改进式方法：先通过随机生成或简单启发式规则获取初始解，再利用 1.2 节提及的 2-opt 或 3-opt 操作迭代优化初始解，直至满足停止条件（如达到预设迭代步数、获得高质量解）。

3.1.1 马尔可夫决策过程建模

本研究将该改进过程建模为马尔可夫决策过程，具体定义如下：

1）状态（State）：由当前解S_t和截至当前最好的解S'_t共同构成，记为$\bar{S}_t=(S_t,S'_t)$。

2）动作（Action）：选出 3 条待删除边a_0，a_1，a_2与重连方式$T_{3\text{-opt}}$，记为$A_t=(a_0,a_1,a_2,T_{3\text{-opt}})$。

3）状态转移（Transaction）：对当前解执行 3-opt 操作，使其生成新的解，从而当前状态$\bar{S}_t$转移为新状态$\bar{S}_{t+1}$。

4）奖励（Reward）：定义为截至当前状态最好的解S'_t与下一状态最好的解的长度改进量，记为$R_t=L(S'_t)-\min(L(S'_t),L(S_{t+1}))$，其中$L(\cdot)$表示解的长度，即路径的长度。

5）回报（Returns）：固定步数的奖励的折扣累加和记为$G_t=\sum_{t'=t}^{T-1}\gamma^{t'-t}R_{t'}$，其中折扣因子$\gamma\in(0,1]$。

3.1.2 神经网络结构设计

本研究采用含编码器-解码器结构的神经网络定义随机策略$p(\pi|s)$，用于学习启发式规则并迭代优化解，该方法被命名为 Neural-3-OPT[25]。其求解流程如图 16 所示：对于给定旅行商问题实例，首先通过随机生成或简单启发式规则获取初始解；随后利用神经网络学出的启发式规则迭代改进当前解，直至达到预设迭代步数。每个改进步骤中，Neural-3-OPT 通过编码器计算每个节点与当前解的特征嵌入向量，再将其输入 2 个解码器：解码器 1 依次筛选 3 条待删除的低效边，解码器 2 选择适配全局的重连方式；最终执行 3-opt 操作完成解的改进。

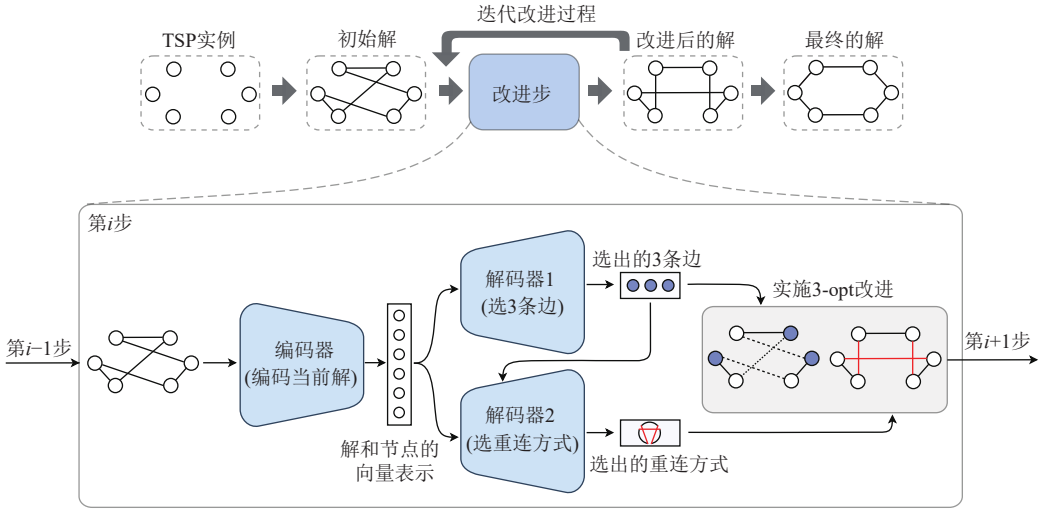


Fig. 16 Framework of Neural-3-OPT method for solving TSP

图 16 求解旅行商问题的 Neural-3-OPT 方法框架

具体网络结构设计如下：

1）编码器。由稀疏图卷积网络（sparse graph convolutional network, K-GCN）与双向长短时记忆网络（bidirectional long short-term memory network, bi-LSTM）[14] 组成。在输入旅行商问题的图结构与一个可行解后，稀疏图卷积网络通过聚合图中每个节点的K个最近邻的嵌入来编码每个节点的嵌入；随后，双向长短时记忆网络对可行解路径进行编码，以获得更准确的节点的嵌入与路径的嵌入。

具体而言，对于一条具有n个节点的可行路径$\{\boldsymbol{x}_i\}_n$，其中$\{\boldsymbol{x}_i\}\in[0,1]^2$表示节点的二维坐标，步骤如下：

①输入层

利用线性映射将节点的二维坐标$\boldsymbol{x}_i$映射到d维空间得到节点的初始嵌入$\boldsymbol{x}_i^0$：

$$\boldsymbol{x}_i^0=\boldsymbol{W}_0\boldsymbol{x}_i+\boldsymbol{b}_0,$$

其中$\boldsymbol{W}_0\in\mathbb{R}^{d\times 2}$与$\boldsymbol{b}_0\in\mathbb{R}^d$是可学习的参数。

②聚合图中每个节点的K个最近邻的嵌入来编码每个节点的嵌入

根据旅行商问题的图结构生成一个稀疏图G_K来记录每个节点的K个最近邻，再将G_K与$\boldsymbol{x}_i^0$输入到N个K-GCN层中。

在K-GCN层中，将第$l\in\{1,\cdots,N\}$层产生的节点嵌入表示为$\boldsymbol{x}_i^l$，将下一层的节点嵌入表示为$\boldsymbol{x}_i^{l+1}$，计算图节点嵌入$\boldsymbol{x}_i^{l+1}$：

$$\boldsymbol{x}_i^{l+1}=\boldsymbol{x}_i^l+ReLU\left(\sum_{j\in\mathcal{N}(i)}e_{ij}(\boldsymbol{W}^l\boldsymbol{x}_j{}^l+\boldsymbol{b}^l)\right),$$

其中$\mathcal{N}(i)$是第i个节点的K个最近邻；$\boldsymbol{W}^l\in\mathbb{R}^{d\times d}$与$\boldsymbol{b}^l\in\mathbb{R}^d$是第$l$个K-GCN层的可学习的参数；$e_{ij}$是$G_K$的归一化后的边权重，表示如下：

$$e_{ij}=\frac{\tilde{d}_{ij}}{\sum_j\tilde{d}_{ij}},$$

$$\tilde{d}_{ij}=\frac{\epsilon}{\epsilon+d_{ij}},$$

其中ϵ是一个小实数，d_{ij}是第i个节点与第j个节点之间的欧几里德距离。

③使用bi-LSTM编码可行解路径

使用bi-LSTM对可行解序列进行编码，此时可行解序列由K-GCN得到的节点嵌入构成的序列$\{\boldsymbol{x}_i^N\}_n$来表示。bi-LSTM包括2个LSTM，分别用于从左到右与从右到左编码可行解序列，从而得到更准确的节点嵌入$\boldsymbol{o}_i$与路径嵌入$\boldsymbol{h}_n$，表示为：

$$\boldsymbol{o}_i=\tanh((\boldsymbol{W}_f\boldsymbol{h}_i^{\rightarrow}+\boldsymbol{b}_f)+(\boldsymbol{W}_b\boldsymbol{h}_i^{\leftarrow}+\boldsymbol{b}_b)),$$

$$\boldsymbol{h}_n=\boldsymbol{h}_n^{\rightarrow}+\boldsymbol{h}_n^{\leftarrow},$$

其中$\boldsymbol{o}_i$和$\boldsymbol{h}_n$分别表示最终的节点嵌入和路径嵌入，$\boldsymbol{h}_i^{\rightarrow}\in\mathbb{R}^d$和$\boldsymbol{h}_n^{\rightarrow}$分别表示一个LSTM从左向右编码可行解序列过程中的隐向量和第n步的输出，$\boldsymbol{h}_i^{\leftarrow}\in\mathbb{R}^d$和$\boldsymbol{h}_n^{\leftarrow}$分别表示另一个LSTM从右向左编码可行解序列过程中的隐向量和第n步的输出，$\boldsymbol{W}_f,\boldsymbol{W}_b\in\mathbb{R}^d$、$\boldsymbol{b}_f$，$\boldsymbol{b}_b\in\mathbb{R}^d$是可学习的参数。

由于状态$\bar{S}_t=(S_t,S_t')$包含2个路径：当前路径S_t与截至当前最好的路径S_t'，因此Neural-3-OPT使用相同结构的编码器来分别编码它们的嵌入。$\boldsymbol{h}_n\in\mathbb{R}^d$表示路径$S_t$的嵌入，$\boldsymbol{h}_n'\in\mathbb{R}^d$表示路径$S_t'$的嵌入。

2)解码器。解码器$\pi_{\theta,\phi}$包括2个神经网络：解码器1为边选择器π_θ、解码器2为重连方式选择器π_ϕ。Neural-3-OPT将选择重连方式视为一个分类问题，使神经网络从重连方式的多个备选方式中选择合适的一个。解码器将编码器输出的节点嵌入$\{\boldsymbol{o}_i\}_n$、路径嵌入$\boldsymbol{h}_n$与$\boldsymbol{h}_n'$作为输入，利用边选择器依次输出3条边，然后利用重连方式选择器输出一个适当的重新连接方式。

Neural-3-OPT使用链式规则将3-opt操作的概率公式分解为：

$$\pi_{\theta,\phi}(A|\bar{S})=\pi_\theta(a_{<3}|\bar{S})\cdot\pi_\phi(T_{3\text{-opt}}|a_{<3},\bar{S})=\prod_{i=0}^{2}p_\theta(a_i|a_{<i},\bar{S})\cdot p_\phi(T_{3\text{-opt}}|a_{<3},\bar{S}),$$

其中a_i是第i个选定的边，$a_{<i}$表示第i步之前选定的边的集合，$T_{3\text{-opt}}$是选出的重连方式。

在边选择器π_θ中，利用节点嵌入$\{\boldsymbol{o}_i\}_n$、路径嵌入$\boldsymbol{h}_n$与$\boldsymbol{h}_n'$来计算查询向量$(\boldsymbol{q}_0,\boldsymbol{q}_1,\boldsymbol{q}_2)$，基于指针机制依次输出边$(a_0,a_1,a_2)$。这些查询向量的计算方式为：

$$\boldsymbol{q}_0=(\boldsymbol{W}_s\boldsymbol{h}_n+\boldsymbol{b}_s\|\boldsymbol{W}_{s'}\boldsymbol{h}_n'+\boldsymbol{b}_{s'})+\max(\boldsymbol{x}_i^N),$$

$$\boldsymbol{q}_i=\tanh((\boldsymbol{W}_q\boldsymbol{q}_{i-1}+\boldsymbol{b}_q)+(\boldsymbol{W}_o\boldsymbol{o}_{i-1}+\boldsymbol{b}_o)),$$

其中$\boldsymbol{W}_s,\boldsymbol{W}_{s'}\in\mathbb{R}^{\frac{d}{2}\times d}$、$\boldsymbol{W}_q,\boldsymbol{W}_o\in\mathbb{R}^{d\times d}$、$\boldsymbol{b}_s,\boldsymbol{b}_{s'}\in\mathbb{R}^{\frac{d}{2}}$与$\boldsymbol{b}_q,\boldsymbol{b}_o\in\mathbb{R}^d$是可学习的参数。指针机制参数化为：

$$p_\theta(a_i|a_{<i},\bar{S})=\text{Softmax}(C\tanh(u^i)),$$

其中

$$u_j^i=\begin{cases}\boldsymbol{v}^{\mathrm{T}}\tanh\left(\boldsymbol{K}_{\boldsymbol{o}_j}+\boldsymbol{Q}_{q_j}\right), & \text{if } j>a_{i-1},\\ -\infty, & \text{其他},\end{cases}$$

这里$\boldsymbol{K},\boldsymbol{Q}\in\mathbb{R}^{d\times d}$与$\boldsymbol{v}\in\mathbb{R}^d$为学习的注意力参数，$C$为超参数。

在重连方式选择器π_ϕ中，采用特征逐层线性调制网络(feature-wise linear modulation network, FiLM)网络[34]来融合当前解的嵌入与边的嵌入作为条件信息，以计算备选方式的概率分布，进而基于概率分布选出合适的重连方式。FiLM网络是一种有效的信息融合方法，它将加法与乘法交互结合成条件仿射变换，与单一的加法或乘法相比，它能更有效地整合多个信息源。

重连方式选择器采用了L个FiLM网络层，每个层通过神经网络学习2个函数$f_\phi^{(l)}$与$g_\phi^{(l)}(l\in\{1,\cdots,L\})$，这2个函数将3条选定的边的嵌入之和$\boldsymbol{h}^{(c)}$映射到FiLM网络的参数$\boldsymbol{\gamma}^{(l)},\boldsymbol{\beta}^{(l)}\in\mathbb{R}^d$：

$$\boldsymbol{\gamma}^{(l)}=f_\phi^{(l)}(\boldsymbol{h}^{(c)}),\boldsymbol{\beta}^{(l)}=g_\phi^{(l)}(\boldsymbol{h}^{(c)}),$$

然后$\boldsymbol{\gamma}^{(l)}$与$\boldsymbol{\beta}^{(l)}$再与各层的输入信息$\boldsymbol{h}_m$融合，融合方式为：

$$\boldsymbol{h}_m^{(l+1)}=FiLM^{(l)}\left(\boldsymbol{h}_m^{(l)}|\boldsymbol{\gamma}^{(l)},\boldsymbol{\beta}^{(l)}\right)=\boldsymbol{\gamma}^{(l)}\odot\boldsymbol{h}_m^{(l)}+\boldsymbol{\beta}^{(l)},$$

其中第1层的输入$\boldsymbol{h}_m^{(0)}$是路径嵌入$\boldsymbol{h}_n$。

之后，将L层的输出$\boldsymbol{h}_m^{(L)}$输入Softmax层得到重连方式的概率分布，表示如下：

$$p_\phi\left(T_{3\text{-opt}}|a_{<3},\bar{S}\right)=Softmax(\boldsymbol{W}_m\boldsymbol{h}_m^{(L)}+\boldsymbol{b}_m),$$

其中$\boldsymbol{W}_m \in \mathbb{R}^{3\times d}$与$\boldsymbol{b}_m \in \mathbb{R}^3$是可学习的参数。

最终重连方式选择器输出适配全局的重连方式。

3)价值网络。将一个旅行商问题当前状态下的路径嵌入输入价值网络，价值网络输出一个实数值来估计当前状态的价值，该实数值用于优化策略网络。价值网络的计算为：

$$V_\psi(\bar{S}) = \boldsymbol{W}_r ReLU\left(\boldsymbol{W}_z\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^N + (\boldsymbol{W}_v\boldsymbol{h}_n + \boldsymbol{b}_v \| \boldsymbol{W}_{v'}\boldsymbol{h}'_n + \boldsymbol{b}_{v'})\right) + \boldsymbol{b}_z\right) + \boldsymbol{b}_r,$$

其中$\boldsymbol{W}_v, \boldsymbol{W}_{v'} \in \mathbb{R}^{\frac{d}{2}\times d}$、$\boldsymbol{W}_z \in \mathbb{R}^{d\times d}$、$\boldsymbol{W}_r \in \mathbb{R}^{1\times d}$、$\boldsymbol{b}_v, \boldsymbol{b}_{v'} \in \mathbb{R}^{\frac{d}{2}}$，$\boldsymbol{b}_z \in \mathbb{R}^d$与$\boldsymbol{b}_r \in \mathbb{R}$都是可学习的参数。

3.1.3 训练目标与优化算法

模型训练采用强化学习中的行动者-评论者(Actor-Critic)算法[35]。行动者为策略网络，负责生成动作；评论者为价值网络，负责评估当前状态的期望回报。其中，策略网络由上述神经网络定义的随机策略$p(\pi|s)$构成，其目标函数为回报的期望，即：

$$J(\theta,\phi|\bar{S}) = E_{\pi_{\theta,\phi}}[G_t|\bar{S}]。$$

优化目标为最大化该期望回报(即最大化累计路径改进长度)，其梯度公式为：

$$\nabla J(\theta,\phi) \approx \frac{1}{B}\frac{1}{T}\left[\sum_{b=1}^{B}\sum_{t=0}^{T-1}\nabla_{\theta,\phi}\log\pi_{\theta,\phi}(A_t^b \mid \bar{S}_t^b)\mathcal{A}_t^b\right],$$

其中$\mathcal{A}_t^b = G_t^b - V_\psi\left(\bar{S}_t^b\right)$为优势函数，$G_t^b$为回报，$V_\psi\left(\bar{S}_t^b\right)$为价值网络估计的状态值。

价值网络$V_\psi(\bar{S})$输出当前状态的实值估计，其目标函数为回报与状态估计值的均方误差：

$$\mathcal{L}(\psi) = \frac{1}{B}\frac{1}{T}\left[\sum_{b=1}^{T}\sum_{t=0}^{T-1}\|G_t^b - V_\psi(\bar{S}_t^b)\|_2^2\right],$$

优化目标是最小化$\mathcal{L}(\psi)$。

求解旅行商问题的Neural-3-OPT方法训练过程的算法采用ADAM作为优化器，训练过程如算法2所示。

算法2. 求解TSP的Neural-3-OPT方法训练过程。

输入：训练集，迭代轮数(*epoch*)E，轨迹长度T_e，批次(*batch*)大小B，邻域大小K，迭代步数$\mathbb{T}$，学习率λ。

① 初始化策略网络(Actor)$\pi_{\theta,\phi}$和价值网络(Critic)V_ψ的参数θ,ϕ和ψ；

② for $epoch = 1,\cdots,E$ 运行

③ $T \leftarrow T_e$；

④ for $batch = 1,\cdots,N_B$运行

⑤ $t \leftarrow 0$；

⑥ 随机初始化$\bar{S}_0^b, \forall b \in \{1,\cdots,B\}$；

⑦ while $t < \mathbb{T}$

⑧ $$g_{\theta,\phi} \leftarrow \frac{1}{B}\left[\frac{1}{T}\sum_{b=1}^{B}\sum_{i=0}^{T-1}\nabla_{\theta,\phi}\log\pi_\theta(a_{<3}|\bar{S}_i^b)\times \pi_\phi(T_{3\text{-opt}}|a_{<3},\bar{S}_i^b)A_i^b + \beta_H\nabla_{\theta,\phi}H(\pi_{\theta,\phi})\right];$$

⑨ $$g_\psi \leftarrow \frac{1}{BT}\left[\beta_V\sum_{b=1}^{B}\sum_{i=0}^{T-1}\nabla_\psi\|G_i^b - V_\psi(\bar{S}_i^b)\|_2^2\right];$$

⑩ $\theta,\phi,\psi \leftarrow Adam(\lambda, -g_{\theta,\phi}, g_\psi)$；

⑪ end while

⑫ end for

⑬ end for

3.2 旅行商问题数据集构建

与排课问题一致，旅行商问题的训练集与验证集仅包含问题本身的特征，不包含任何解信息。本研究方法的核心目标为：从大量问题实例中，在不依赖先验解信息的前提下自主学习启发式规则。对于任意旅行商问题实例，每个城市的特征由横坐标x_i和纵坐标y_i构成；所有城市坐标均在$[0,1]^2$范围内随机生成。对于规模为n的实例，其特征可表示为$[[x_1,y_1],[x_2,y_2],\cdots,[x_n,y_n]]$。测试集同时包含问题特征与最优解，其中最优解通过精确求解器Concorde[36]计算获得，为后续性能评估提供基准。

3.3 旅行商问题的实验结果和讨论

3.3.1 实验设置与性能评估

本研究分别在城市规模为20、50、100(记为TSP20/50/100)的随机生成的旅行商问题实例上训练模型使其学习启发式规则。其中TSP20模型训练200轮(*epoch*)，TSP50和TSP100模型各训练300轮。每轮的训练集和验证集分别包含5 120个和256个对应规模的随机实例。具体实验设置如表3所示。

测试阶段采用规模匹配的测试集(各10 000个

Table 3 Experimental Setup for TSP

表3 旅行商问题实验设置

分类	参数名称	参数值
网络结构	编码器	3层K-GCN + bi-LSTM，维度d=128
	解码器	Ptr-Net + FiLM，维度d=128
训练设置	问题规模n	20/50/100
	学习率λ	1E−3
	批次大小B	512
	训练轮数E	200（TSP20），300（TSP50/TSP100）
	轨迹长度T_e	8~20
	邻域大小K	8
	迭代步数$\mathbb{T}$	200

实例)，以“最优性差距(optimality gap)”作为性能评估指标，即模型输出解与精确解的长度差值占精确解长度的比例。

3.3.2 实验结果与比较

表 4 实验结果显示：Neural-3-OPT 为 TSP20 测试集所有实例找到了最优解(最优性差距为 0.00%)；针对 TSP50 与 TSP100 实例，最优性差距分别低至 0.08%和 0.74%。此外，该方法的性能显著优于人工设计启发式算法(最近插入法、随机插入法、最远插入法)与谷歌 OR-Tools 求解器[37]。上述结果验证了“利用深度学习自主学习启发式规则，辅助算法设计”的可行性。

Table 4 Experimental Results of TSP

表 4 旅行商问题的实验结果

方法		TSP20		TSP50		TSP100	
		解的平均长度	平均最优性差距/%	解的平均长度	平均最优性差距/%	解的平均长度	平均最优性差距/%
精确算法	Concorde	3.84	0.00	5.70	0.00	7.76	0.00
传统的启发式算法	OR-Tools	3.85	0.37	5.80	1.83	7.99	2.90
	最近插入法	4.33	12.91	6.78	19.03	9.46	21.82
	随机插入法	4.00	4.36	6.13	7.65	8.52	9.69
	最远插入法	3.93	2.36	6.01	5.53	8.35	7.59
基于深度学习的启发式算法	Neural-3-OPT	**3.84**	**0.00**	**5.70**	**0.08**	**7.82**	**0.74**

注：黑体数值表示最优对比结果。

为进一步验证习得启发式规则的泛化性，本研究利用 TSP100 训练的模型求解 TSPLIB 中的真实世界实例[38]。表 5 结果显示：Neural-3-OPT 在规模 51~280 的真实实例上，平均最优性差距仅为 2.93%，优于谷歌 OR-Tools 求解器(3.79%)。这表明从随机实例中习得的启发式规则可有效迁移至真实场景，验证了方法的实用价值。

Table 5 Experimental Results of Real-world TSP Instances

表 5 真实世界的旅行商问题的实例的实验结果

实例名称	最优路径长度	路径长度	
		OR-Tools	Neural-3-OPT
eil51	426	439	**436**
berlin52	7 542	7 944	**7 572**
pr76	108 159	110 948	**108 277**
rd100	7 910	8 221	**8 077**
eil101	629	650	**640**
lin105	14 379	15 363	**15 229**
ch130	6 110	6 329	**6 213**
pr144	58 537	59 286	60 851
ts225	126 643	127 763	129 381
a280	2 579	2 742	2 808
平均最优性差距	0.00%	3.79%	**2.93%**

注：黑体数值表示最优对比结果。

3-opt 操作中，删除 3 条边后存在 7 种可行重连方式(图 3)。Neural-3-OPT 通过神经网络学出的策略选择重连方式，而非依赖人工设计策略。为验证习得策略的优越性，本研究将其与 3 种人工设计重连策略对比：随机选择重连方式(随机策略)、固定选择某一重连方式(固定策略)、贪心选择当前步改进最大的重连方式(贪心策略)。

表 6 结果显示：Neural-3-OPT 在 TSP100 上的最优性差距仅为 0.74%，显著优于随机策略(4.26%)、固定策略(2.04%)与贪心策略(9.97%)。这表明神经网络学出的 3-opt 启发式规则，通过适配迭代全程的重连策略而非单步贪心选择，具备更强的全局求解能力。

Table 6 Experimental Results of Different Strategies to Select Reconnecting Type

表 6 选择重连方式的不同策略的实验结果

重连策略	TSP50		TSP100	
	最优路径长度	平均最优性差距/%	最优路径长度	平均最优性差距/%
随机策略	5.72	0.33	8.04	4.26
固定策略	5.71	0.18	7.92	2.04
贪心策略	5.87	2.99	8.54	9.97
Neural-3-OPT	**5.70**	**0.08**	**7.82**	**0.74**

注：黑体数值表示最优对比结果。

泛化性测试结果表明：利用 TSP20/50/100 训练的模型可有效求解 TSP200 规模实例，验证了模型的良好泛化性能，进一步佐证了神经网络自主学习启发式规则的可行性。

3.3.3 实验结果分析与讨论

为深入揭示 Neural-3-opt 启发式算法的核心特性与性能优势根源，本节从 3 方面展开递进式分析：首

先通过宏观对比验证其求解质量优势；其次结合具体实例拆解求解过程，定位关键差异环节；最后量化分析改进策略差异，揭示性能提升机理，并补充说明算法的可扩展性。

3.3.3.1 Neural-3-OPT 与传统 3-opt 的求解质量对比

为验证 Neural-3-OPT 的求解质量优势，本研究在相同迭代步数约束下，对 2 种算法进行规模化对比实验：分别求解 TSP20/50/100 测试集（各含 10 000 个实例），通过可视化每步改进后的路径长度均值，直观呈现两者的性能差异。

图 17 中，红色虚线表示测试集最优解的平均值，橙色实线为 Neural-3-OPT 的路径长度变化曲线，蓝色实线为传统 3-opt 的结果：图 17(a)对应 500 步内求解 TSP20 的结果，图 17(b)对应 1 000 步内求解 TSP50 的结果，图 17(c)对应 2 000 步内求解 TSP100 的结果。

实验结果表明，2 种算法存在显著性能差距：1)相同改进步数下，Neural-3-OPT 的求解质量始终优于传统 3-opt；2)Neural-3-OPT 具备快速收敛特性，可在极少步数内逼近最优解，而传统 3-opt 往往需要更多的迭代步数，且得到的解质量仍低于 Neural-3-OPT。具体而言：TSP20 实例中，Neural-3-OPT 约 20 步即可接近最优解，传统 3-opt 需 200 步且仍存在明显差距；TSP50 实例中，Neural-3-OPT 约 50 步收敛至高质量解，传统 3-opt 需 1 000 步；TSP100 实例中，Neural-3-OPT 约 100 步逼近最优解，传统 3-opt 需 2 000 步且差距显著。

3.3.3.2 基于实例对求解过程差异拆解

为定位性能优势的根源，本研究从 TSP20 测试集中随机选取实例，对比 2 种算法的求解过程细节，重点分析路径长度变化趋势与待删除边选择策略的差异。

图 18 呈现了 2 种算法的路径长度动态变化，红色虚线为最优路径长度，橙色实线为 Neural-3-OPT 的路径长度变化，蓝色实线为传统 3-opt 的结果。可见，Neural-3-OPT 的求解过程具有“初期快速下降、快速收敛”的特征，求解初期路径长度即快速降低，约 20 步即可获得高质量解；而传统 3-opt 的路径长度下降趋势先快后缓，需约 175 步才能收敛至较差解，且最终解质量显著低于 Neural-3-OPT。

图 19 进一步对比了 2 种算法在同一可行解上的单步改进效果：图 19(a)为初始可行解（路径长度 10.890），图 19(b)为 Neural-3-OPT 改进一步后结果（路径长度 10.291），图 19(c)为传统 3-opt 改进一步后结果（路径长度 10.591）。图中黑色圆点代表城市，蓝色

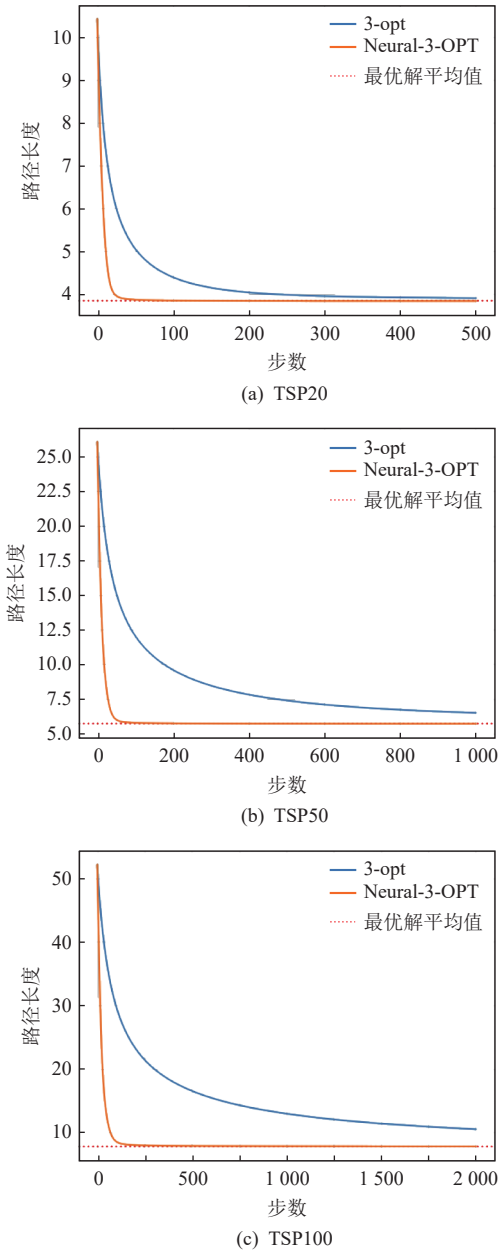


Fig. 17 Comparing solution quality between Neural-3-OPT and 3-opt algorithms

图 17 Neural-3-OPT 与 3-opt 算法求解质量对比

实线为可行解路径，红色虚线为待删除的 3 条边。核心差异在于待删除边的选择策略：Neural-3-OPT 优先删除总长度较长（对解质量影响更大）的 3 条边，而传统 3-opt 删除的边总长度较短（影响较小），直接导致单步改进量更小。这一差异揭示了 Neural-3-OPT 性能优势的关键：通过更精准的待删除边选择，实现“更少步数、更高质量”的求解效果。

3.3.3.3 Neural-3-OPT 与传统的 3-opt 改进策略的差异分析

为进一步分析 2 种算法的改进策略差异，本研究

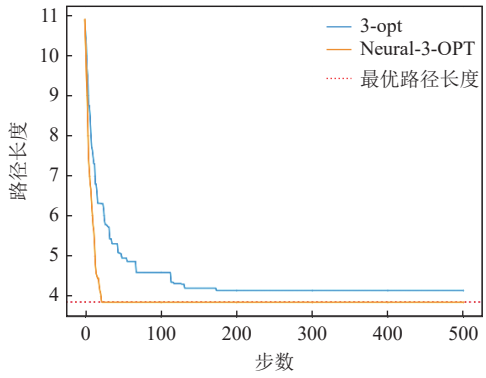


Fig. 18 Performance comparison of Neural-3-OPT and 3-opt algorithms on a random instance

图 18 Neural-3-OPT 与 3-opt 算法在随机实例上的表现对比

统计了每步改进量(路径长度减少的绝对数值)与改进程度(减少的值占改进前路径长度的百分比)，结果如图 20 所示。其中，图 20(a)(b)的横轴为改进步数，纵轴为路径长度的改进量；图 20(c)(d)的纵轴为路径长度的改进程度(百分比)，用于消除初始路径长度差异的影响，更客观地反映改进效率。

统计结果清晰呈现了 2 种算法的核心策略差异：

如图 20(a)所示，Neural-3-OPT 采用“初期精准选优”的高效策略，在迭代初期优先选择可大幅降低路径长度的操作，从而快速收敛至高质量解；而传统 3-opt 的改进操作选择具有随机性，如图 20(b)所示，即使经过多步迭代仍可能出现路径长度的大幅波动，且存在大量无效改进步骤(改进量为 0)。

从改进程度来看，随着迭代步数增加，路径长度逐渐递减，单步改进量自然降低，但 Neural-3-OPT 在迭代初期的每一步均尽可能最大化改进程度，充分利用每一次改进机会(见图 20(c))；传统 3-opt 则无明确的改进优先级，仅随机接受有效改进操作，且无效步骤占比更高(见图 20(d))，导致整体改进效率低下。

需要说明的是，“初期精准选优”是基于大规模群体统计与单个实例轨迹验证确立的决策模式：TSP20/50/100 三个规模各 10 000 个测试实例均呈现该特征(图 17、图 20)，且随机选取的案例显示，求解初期的每一步均呈现显著的路径长度下降(图 18)并倾向于选择对全局质量影响大的低效边(图 19)。该模式在观测范围内普遍存在，既非偶然的统计平均，也非个别特例的表现，表明神经网络习得了可复现的、系统性的初期选优策略。当然，该策略的完整数学刻画(如价值函数对边特征的精确依赖关系)仍有待深入，本文通过性能对比与决策过程分析确立其存在及效果，为神经网络学习启发式规则提供可验证的行为层面解释。

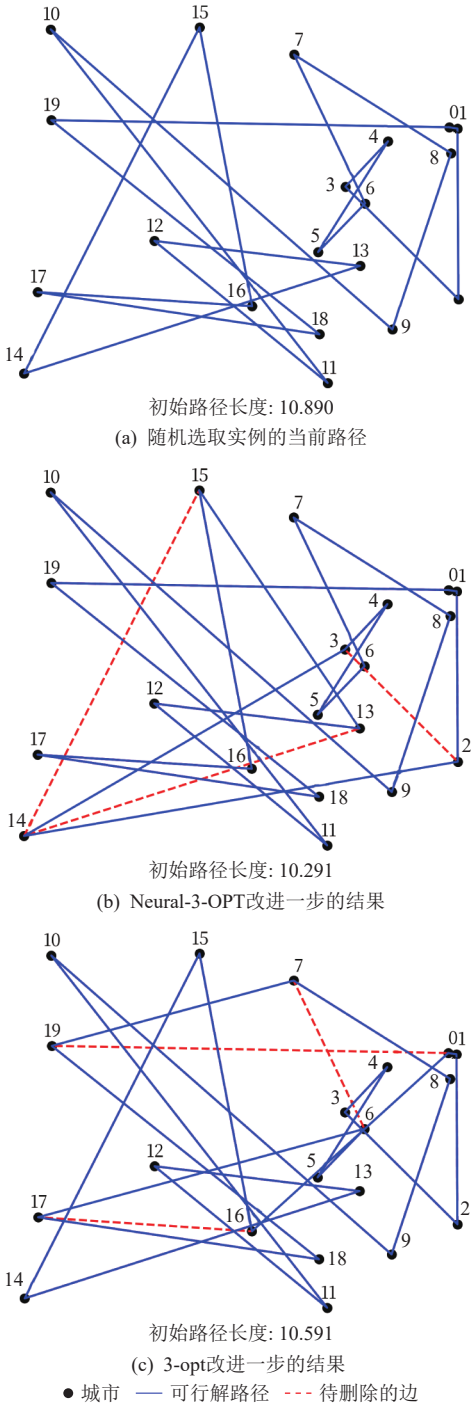
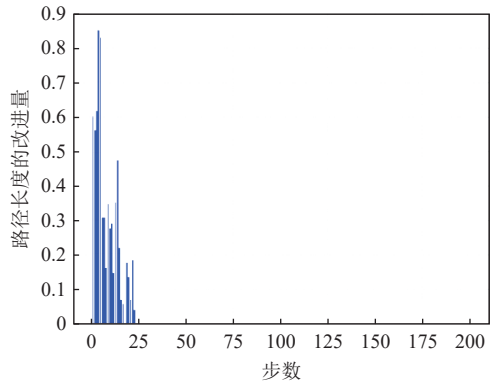


Fig. 19 Performance comparison of one-step improvement between Neural-3-OPT and 3-opt algorithms

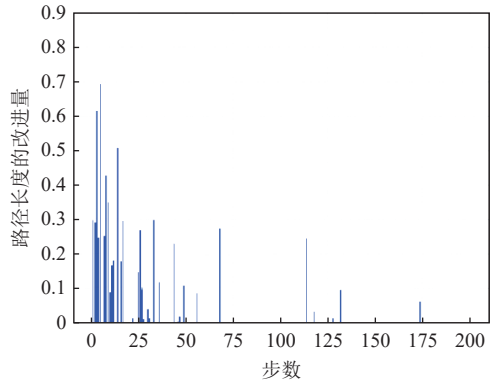
图 19 Neural-3-OPT 与 3-opt 算法随机改进一步的表现对比

3.3.3.4 算法的可扩展性说明

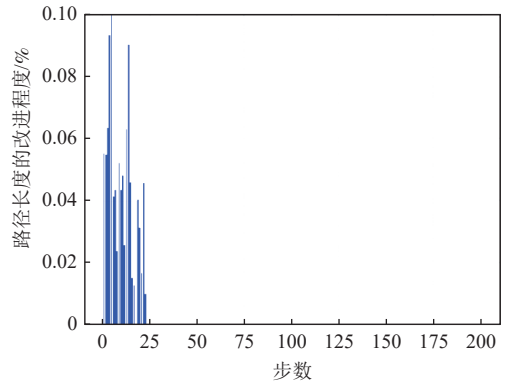
Neural-3-OPT 的网络架构具备良好的可扩展性，可轻松推广至 k-opt($k \geqslant 4$)等更复杂的改进规则。具体扩展思路为：将解码器 1 从“依次筛选 3 条待删除边”扩展为“依次筛选 k 条待删除边”，同时将解码器



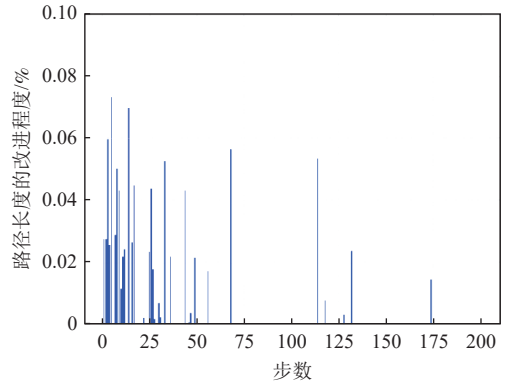
(a) Neural-3-OPT每步改进时路径长度的变化量



(b) 3-opt每步改进时路径长度的变化量



(c) Neural-3-OPT每步改进时路径长度的变化程度



(d) 3-opt每步改进时路径长度的变化程度

Fig. 20 Comparing improvement strategies for Neural-3-OPT and 3-opt algorithms

图 20 Neural-3-OPT 与 3-opt 算法的改进策略对比

2 的重连方式选择器适配 *k*-opt 操作对应的可行重连方式即可。这一扩展性源于算法的核心设计逻辑：通过神经网络自主学习“待删除边选择”与“重连方式匹配”的启发式规则，而非依赖人工设计的固定逻辑，为求解更复杂的组合优化问题提供了灵活的扩展方案。

4 结 论

本文围绕“深度学习辅助算法设计”这一核心主题，以排课问题和旅行商问题这 2 类经典组合优化问题为研究载体，探索了无监督条件下深度强化学习自主学出贪心规则和启发式规则的可行性、有效性与内在机理，初步构建了“数据驱动型算法设计”研究范式与技术框架，主要结论如下：

1）本文证实深度强化学习能够在无监督条件下，仅从大量问题实例中自主学出有效且可解释的算法规则，为突破传统算法设计的经验依赖瓶颈提供了新路径。

2）针对排课问题提出的 Neural-Greedy 算法，通过深度强化学习可自主学出 3 条最优贪心规则，充分验证了深度强化学习在区间调度类问题中自主学习贪心规则的适用性。

3）针对旅行商问题提出的 Neural-3-OPT 方法，通过深度强化学习可自主学出高效的 3-opt 启发式策略，兼具优异性能、泛化能力与可扩展性，充分验证了深度强化学习在路径优化类问题中自主学习启发式规则的适用性。

4）本文通过可解释性分析深入揭示了深度学习自主学出算法规则的内在机理，提升了方法的可信度与可解释性。对于 Neural-Greedy 框架，其核心在于通过参数调控与神经元协同作用，实现对时间节点关键特征的精准聚焦，进而完成多规则的同步习得；对于 Neural-3-OPT 框架，其性能优势源于“初期精准选优”的智能决策机理，通过优先识别对解质量影响显著的低效边与重连方式，避免无效迭代，实现快速收敛与高质量求解，这与传统 3-opt 算法的随机改进策略形成鲜明对比。

未来研究可进一步扩展至更多类型的组合优化问题与动态场景，探索更为通用的算法学习框架，以推动深度学习辅助算法设计范式在更多实际工程场景中落地应用，为组合优化问题的智能求解提供更完善的解决方案。

作者贡献声明：隋京言提出了算法思路和实验方案、完成实验并撰写论文；丁诗哲、夏博洋、刘睿之、丁振欣、徐黎明、王睿、黄旭林、李捷鹏、张海仓、于春功、王超在不同环节提供了各自的建议和帮助；卜东波提出指导意见并修改论文。

参 考 文 献

[1] Wu Yadong, Zhang Pengfei, Shen Huitao, et al. Visualizing neural network developing perturbation theory[J]. Physical Review A, 2018, 98(1): 010701

[2] Zhang Pengfei, Shen Huitao, Zhai Hui. Machine learning topological invariants with neural networks[J]. Physical Review Letters, 2018, 120(6): 066401

[3] Sun Ning, Yi Jinmin, Zhang Pengfei, et al. Deep learning topological invariants of band insulators[J]. Physical Review B, 2018, 98(8): 085402

[4] Jumper J, Evans R, Pritzel A, et al. Highly accurate protein structure prediction with AlphaFold[J]. Nature, 2021, 596(7873): 583−589

[5] Tunyasuvunakool K, Adler J, Wu Z, et al. Highly accurate protein structure prediction for the human proteome[J]. Nature, 2021, 596(7873): 590−596

[6] Baek M, Dimaio F, Anishchenko I, et al. Accurate prediction of protein structures and interactions using a three-track neural network[J]. Science, 2021, 373(6557): 871−876

[7] Ju Fusong, Zhu Jianwei, Shao Bin, et al. CopulaNet: Learning residue co-evolution directly from multiple sequence alignment for protein structure prediction[J]. Nature Communication, 2021, 12(1): 2535

[8] Davies A, Veličković P, Buesing L, et al. Advancing mathematics by guiding human intuition with AI[J]. Nature, 2021, 600(7887): 70−74

[9] Fawzi A, Balog M, Huang A, et al. Discovering faster matrix multiplication algorithms with reinforcement learning[J]. Nature, 2022, 610(7930): 47−53

[10] Tenachi W, Ibata R, Diakogiannis F I. Deep symbolic regression for physics guided by units constraints: Toward the automated discovery of physical laws[J]. The Astrophysical Journal, 2023, 959(2): 99

[11] Li K, Malik J. Learning to optimize[C]//Proc of the 5th Int Conf on Learning Representations. Red Hook: Curran Associates, Inc., 2017: 2249−2261

[12] Bello I, Pham H, Le Q V, et al. Neural combinatorial optimization with reinforcement learning[J]. arXiv preprint, arXiv: 1611.09940, 2017

[13] Kool W, van Hoof H, Welling M. Attention, learn to solve routing problems![J]. arXiv preprint arXiv: 1803.08475, 2019

[14] d O Costa P R, Rhuggenaath J, Zhang Y, et al. Learning 2-opt heuristics for the traveling salesman problem via deep reinforcement learning[C]//Proc of the 12th Asian Conf on Machine Learning. Cambridge: PMLR, 2020: 465−480

[15] Cook W J. In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation[M]. Princeton: Princeton University Press, 2012

[16] La Maire B F, Mladenov V M. Comparison of neural networks for solving the travelling salesman problem[C]//Proc of the 11th Symp on Neural Network Applications in Electrical Engineering. Belgrade: Serbia, 2012: 21−24

[17] Matai R, Singh S P, Mittal M L. Traveling Salesman Problem, Theory and Applications[M]. Rijeka: InTech, 2010

[18] Gutin G, Punnen A P. The Traveling Salesman Problem and Its Variations[M]. Beilin: Springer, 2006

[19] Little J D, Murty K G, Sweeney D W, et al. An algorithm for the traveling salesman problem[J]. Operations Research, 1963, 11(6): 972−989

[20] Held M, Karp R M. A dynamic programming approach to sequencing problems[J]. Journal of the Society for Industrial and Applied Mathematics, 1962, 10(1): 196−210

[21] Rosenkrantz D J, Stearns R E, Lewis II P M. An analysis of several heuristics for the traveling salesman problem[J]. SIAM Journal on Computing, 1977, 6(3): 563−581

[22] Lin S. Computer solutions of the traveling salesman problem[J]. Bell System Technical Journal, 1965, 44(10): 2245−2269

[23] Johnson D S. Local optimization and the traveling salesman problem[C]//Proc of the 17th Int Colloquium on Automata, Languages, and Programming. Berlin: Springer, 1990: 446−461

[24] Johnson D S, McGeoch L A. Local Search in Combinatorial Optimization[M]. Princeton: Princeton University Press, 2003

[25] Sui Jingyan, Ding Shizhe, Liu Ruizhi, et al. Learning 3-opt heuristics for traveling salesman problem via deep reinforcement learning[C]// Proc of the 13th Asian Conf on Machine Learning. Cambridge: PMLR, 2021: 1301−1316

[26] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J] Proceedings of the IEEE, 2002, 86(11): 2278−2324

[27] Elman J L. Finding structure in time[J]. Cognitive Science, 1990, 14(2): 179−211

[28] Bresson X, Laurent T. Residual gated graph convnets[J]. arXiv preprint, arXiv: 1711.07553, 2017

[29] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[C]//Proc of the 6th Int Conf on Learning Representations. Red Hook: Curran Associates, Inc., 2018: 2920−2931

[30] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems 30. Red Hook: Curran Associates, Inc., 2017: 5999−6009

[31] Vinyals O, Fortunato M, Jaitly N. Pointer networks[C]//Advances in Neural Information Processing Systems 28. Red Hook: Curran Associates, Inc., 2015: 2692−2700

[32] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8(3): 229−256

[33] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint, arXiv: 1412.6980, 2014

[34] Perez E, Strub F, De Vries H, et al. FiLM: Visual reasoning with a general conditioning layer[C]//Proc of the AAAI Conf on Artificial Intelligence. Palo Alto: AAAI, 2018: 3942−3949

[35] Mnih V, Badia A P, Mirza M, et al. Asynchronous methods for deep reinforcement learning[C]//Proc of the 33rd Int Conf on Machine Learning. Cambridge: PMLR, 2016: 1928−1937

[36] Applegate D, Bixby R, Chvatal V, et al. Concorde TSP solver[EB/OL]. 2006[2021-04-03]. https://www.math.uwaterloo.ca/tsp/concorde.html

[37] Laurent Perron & Vincent Furnon. OR-Tools[EB/OL]. 2019[2021-04-15]. https://developers.google.com/optimization

[38] Gerhard R. TSPLIB—A traveling salesman problem library[J]. ORSA Journal on Computing, 1991, 3(4): 376−384



Sui Jingyan, born in 1982. PhD. Her main research interests include algorithm design, machine learning, deep learning, combinatorial optimization and bioinformatics.

隋京言, 1982 年生。博士。主要研究方向为算法设计、机器学习、深度学习、组合优化和生物信息。



Ding Shizhe, born in 1998. PhD. His main research interests include machine learning and combinatorial optimization.

丁诗哲, 1998 年生。博士。主要研究方向为机器学习、组合优化。



Xia Boyang, born in 1998. Master. His main research interests include machine learning and combinatorial optimization.

夏博洋, 1998 年生。硕士。主要研究方向为机器学习、组合优化。



Liu Ruizhi, born in 1996. PhD. His main research interests include open-source EDA, AI for EDA, algorithm design, and combinatorial optimization.

刘睿之, 1996 年生。博士。主要研究方向为开源 EDA、AI 启发的 EDA、算法设计、组合优化。



Ding Zhenxin, born in 1994. Master. His main research interests include algorithm design, machine learning, and combinatorial optimization.

丁振欣, 1994 年生。硕士。主要研究方向为算法设计、机器学习、组合优化。



Xu Liming, born in 1997. Master. Her main research interests include algorithm design, machine learning, and combinatorial optimization.

徐黎明, 1997 年生。硕士。主要研究方向为算法设计、机器学习、组合优化。



Wang Rui, born in 2001. PhD candidate. Her main research interests include algorithm design, machine learning, deep learning, and combinatorial optimization.

王　睿, 2001 年生。博士研究生。主要研究方向为算法设计、机器学习、深度学习、组合优化。



Huang Xulin, born in 1998. Master. His main research interests include algorithm design, machine learning, deep learning, and protein design.

黄旭林, 1998 年生。硕士。主要研究方向为算法设计、机器学习、深度学习、蛋白质设计。



Li Jiepeng, born in 2000. Master. His main research interests include algorithm design, large language model, and reinforcement learning.

李捷鹏, 2000 年生。硕士。主要研究方向为算法设计、大语言模型、强化学习。



Zhang Haicang, born in 1987. PhD. His main research interests include machine learning, protein design, and protein structure prediction.

张海仓, 1987 年。博士。主要研究方向为机器学习、蛋白质设计和蛋白质结构预测。



Yu Chungong, born in 1979. Master. His main research interests include bioinfomatics and protein structure prediction.

于春功, 1979 年生。硕士。主要研究方向为生物信息学、蛋白质结构预测。



Wang Chao, born in 1987. PhD. His main research interests include algorithm design, deep learning, knowledge graph, and automonous driving.

王　超, 1987 年生。博士。主要研究方向为算法设计、深度学习、知识图谱、自动驾驶。



Bu Dongbo, born in 1973. PhD, professor, PhD supervisor. His main research interests include algorithm design, bioinformatics, protein structure prediction, and deep learning.

卜东波, 1973 年生。博士，研究员，博士生导师。主要研究方向为算法设计、生物信息学、蛋白质结构预测、深度学习。