MVMoE: Multi-Task Vehicle Routing Solver with Mixture-of-Experts

Jianan Zhou [1] Zhiguang Cao [2] Yaoxin Wu [3] Wen Song [4] Yining Ma [1] Jie Zhang [1] Chi Xu [1 5]

总结

摘要

- 在**某个特定问题**上独立地建模和训练的神经网络求解器缺乏足够的泛化性和实用性。
- 本文提出使用**混合专家系统**的多任务车辆路由问题求解器（Multi-task Vehicle Routing Solver with Mixture-of-Experts, MVMoE）。旨在不增加计算量的同时提升模型性能。
- 在MVMoE中进一步加入**分层门控机制**，控制模型性能和计算复杂度之间的权衡。
- 实验上，1）模型在10个训练中未见过的VRP变体中展现出zero-shot泛化能力；2）在real-world benchmark实例的few-shot设置中也得到合适的结果。3）门控机制在面对分布外数据时也展现出一定的优越性。
- Github：https://github.com/RoyalSkye/Routing-MVMoE.

研究问题

不同约束的VRP变体

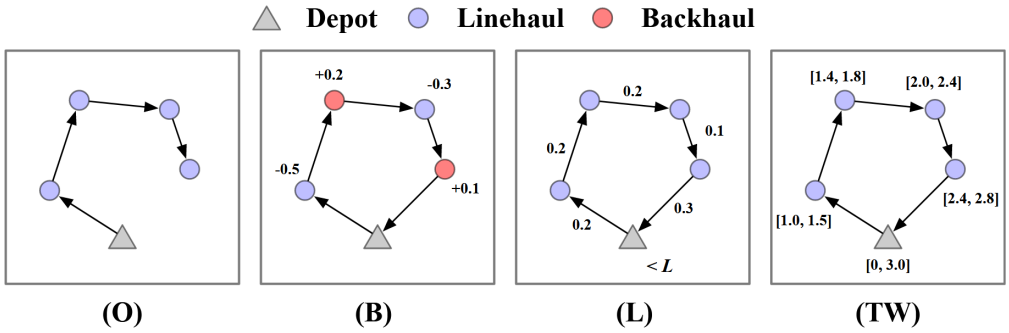

图1. 不同约束的子路径示例。

1. **开放路径约束 (Open Route - O)**：车辆服务完客户点后**无需返回仓库**（v_0）。

2. **回程约束 (Backhaul - B)**：允许linehauls与backhauls**无优先级混合访问。**

- **正需求 ($\delta_i>0$)**：linehauls（卸货）。
- **负需求 ($\delta_i<0$)**：backhauls（装货）。

3. **时长限制 (Duration Limit - L)**：单条路径总长度（成本）**不得超过预设阈值。**

4. **时间窗约束 (Time Window - TW)**：节点v_i必须在$[e_i, l_i]$内开始服务。

- 早到需等待至e_i。
- 所有车辆必须在l_0前返回仓库。

5. **容量限制(Capacity Constraint - C)**：即CVRP。

> **开放路径耦合效应**：当与(O)组合时，**免除仓库返回时间约束。**
> **约束组合特性**：多约束组合存在非线性交互（如O+TW），非简单叠加。5种基础约束可组合成16种VRP变体。

方法

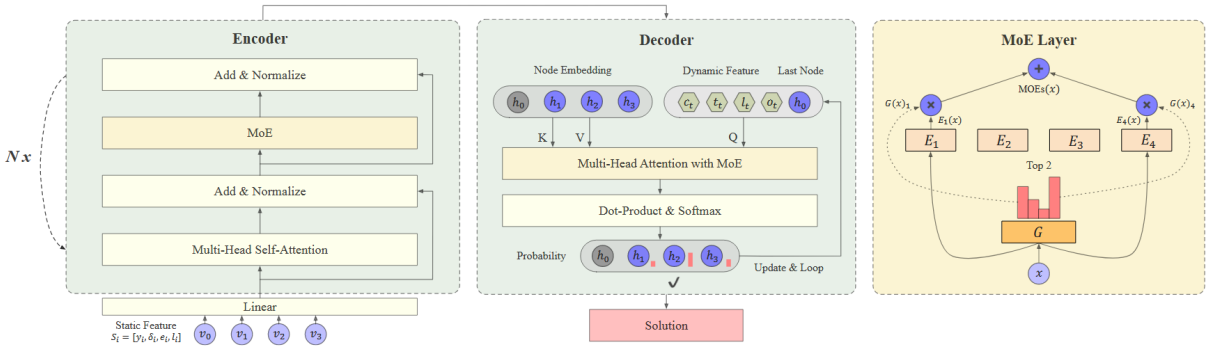

图2. MVMoe模型架构。具体来说，在编码器中用 MoE 替换了 FFN 层，并在解码器的多头注意力的最终线性层中用 MoE 进行替换。

编码器 (Encoder)

- **输入**：每个节点 v_i 的**静态特征** $\mathcal{S}_{i} = { y_i, \delta_i, e_i, l_i }$
  - y_i：坐标
  - δ_i：需求
  - e_i：时间窗开始时间
  - l_i：时间窗结束时间
- **输出**：d 维**节点嵌入向量** h_i

解码器 (Decoder) - 第 t 步

- **输入**：
  1. **节点嵌入**：编码器输出的所有 h_i
  2. **上下文表示**：
     - 上一个被选节点的嵌入 h_{last}
     - **动态特征** $\mathcal{D}_{t} = { c_t, t_t, l_t, o_t }$
       - c_t：车辆剩余容量
       - t_t：当前时间
       - l_t：当前部分路径长度
       - o_t：开放路径指示器（是否需要返回仓库）
- **输出**：
  - **节点概率分布**：所有有效节点的选择概率向量
  - **动作**：根据概率分布选择下一个节点，添加到当前部分解中

不同变体随机训练

- 每个训练批次随机选择一种VRP变体
- 当前VRP变体未使用的特征**置零填充**
  - **静态特征示例**（CVRP）：$\mathcal{S}_{i}^{(C)} = {y_i, \delta_i, 0, 0}$ → 时间窗特征(e_i,l_i)填零

    - **动态特征示例**（CVRP）：$\mathcal{D}_{t}^{(C)} = {c_t, 0, l_t, 0}$ → 当前时间t_t和开放路径o_t填零
- **损失函数**：$\min\limits _ {\Theta} \mathcal{L}=\mathcal{L}_{a}+\alpha\mathcal{L}_{b}$
    - $\mathcal{L}_{a}$：原始损失函数（例如REINFORCE损失）。
    - $\mathcal{L}_{b}$：与MoEs相关的损失函数（例如用于确保负载均衡的辅助损失）

门控机制

门控网络增加了额外的计算开销，主要是1）门控网络的前向传播；2）节点到专家的分配。
编码器层的门控次数是固定的，取决于编码器层数。解码器上门控次数由层数和解码次数决定。本文在解码器层提出分层门控机制来权衡模型性能和计算开销。

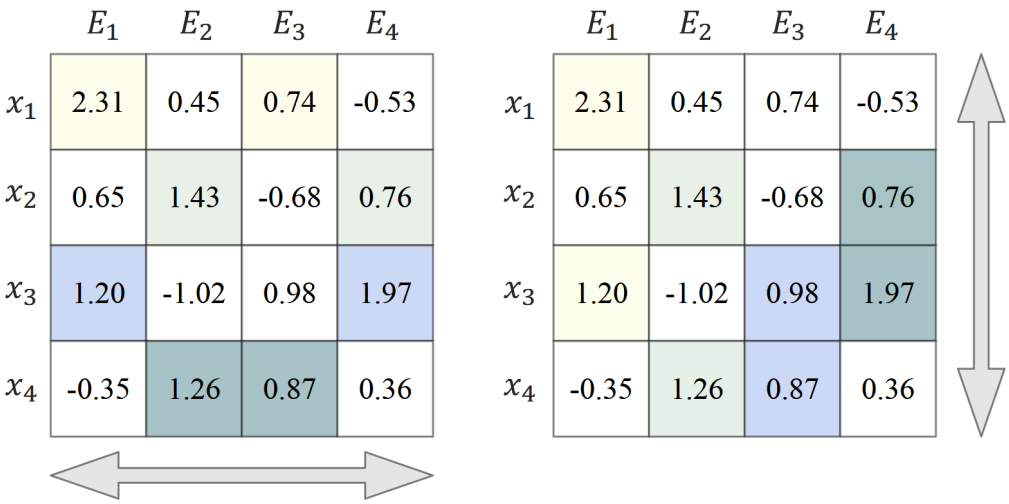

图3. 得分矩阵和门控算法的示例。彩色点代表被选择的节点或专家。
左边：Input-choice gating. 右边：Expert-choice gating.

- 节点级门控：门控网络参数$W_G\in \mathbf{R}^{d\times m}$，输入批次$X\in \mathbf{R}^{l\times d}$，得分矩阵$H=(X\cdot W_G)\in \mathbf{R}^{l\times m}$。
    - Input-choice gating：每个节点基于H选取$TopK$个专家。不保证负载均衡，可能导致某些专家欠拟合。通常会引入辅助损失(Auxiliary Loss)。
    - Expert-choice gating：每个专家基于H选取$TopK$个节点，$K=\frac{l\times \beta}{m}$。能够确保负载均衡，但可能有节点不被任一专家选中。

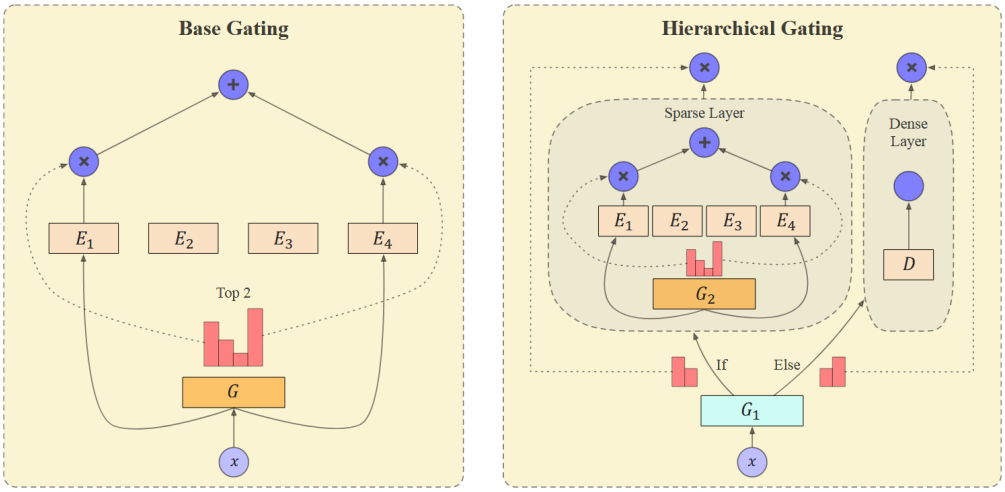

图4. 分层门控（右边）。

- 分层门控机制：在每一解码步骤上应用MoEs十分消耗计算资源，1）解码步骤T随着问题规模n的增加而上升；2）解码过程中问题特有的可行性约束必须被满足。因此，本文仅在部分解码步骤上应用MoEs。
    - 分层门控（右边）通过门控网络G_1将输入引导进稀疏层或稠密层。稀疏层中门控网络G_2再引导节点与专家选取。G_1根据问题级表征X_1进行选择，G_2根据节点级表征进行选择。

实验

考虑5中约束组合的16中VRP变体。设备：NVIDIA A100-80G，AMD EPYC 7513 CPU @ 2.6GHz.

- 基线：
    - 传统求解器：节点数$n=50/100$，搜索时间限制为$20s/40s$。
        - HGS求解CVRP和VRPTW
        - LKH3求解CVRP、OVRP、VRPL、VRPTW
        - OR-Tools求解全部16种变体。
            - 采用平行最廉价插入法(Parallel Cheapest Insertion)求初始解
            - 使用引导式局部搜索(Guided Local Search)做局部搜索。
    - 神经求解器
        - POMO
        - POMO-MTL (1.25M)
- 训练：
    - 对于神经求解器：
        - 采用Adam优化器，学习率$1e^{-4}$，权重衰减$1e^{-6}$。最后10%数据，学习率衰减10倍。
        - batch size=128，epochs=5000 with 20000 trainning instances。
    - 对于多任务求解器：
        - 训练集数据：CVRP、OVRP、VRPB、VRPL、VRPTW、OVRPTW.
    - 对于MVMoE：
        - 专家数$m=4, K=\beta=2$。
        - 辅助损失权重$\alpha=0.01$。
        - 采用node-level input-choice gating。

实验结果

	Method	$n = 50$ Obj.	$n = 50$ Gap	$n = 50$ Time	$n = 100$ Obj.	$n = 100$ Gap	$n = 100$ Time	Method	$n = 50$ Obj.	$n = 50$ Gap	$n = 50$ Time	$n = 100$ Obj.	$n = 100$ Gap	$n = 100$ Time
CVRP	HGS	10.334	*	4.6m	15.504	*	9.1m	VRPTW	HGS	14.509	*	8.4m	24.339	*	19.6m
	LKH3	10.346	0.115%	9.9m	15.590	0.556%	18.0m		LKH3	14.607	0.664%	5.5m	24.721	1.584%	7.8m
	OR-Tools	10.540	1.962%	10.4m	16.381	5.652%	20.8m		OR-Tools	14.915	2.694%	10.4m	25.894	6.297%	20.8m
	OR-Tools (x10)	10.418	0.788%	1.7h	15.935	2.751%	3.5h		OR-Tools (x10)	14.665	1.011%	1.7h	25.212	3.482%	3.5h
	POMO	10.418	0.806%	3s	15.734	1.488%	9s		POMO	14.940	2.990%	3s	25.367	4.307%	11s
	POMO-MTL	10.437	0.987%	3s	15.790	1.846%	9s		POMO-MTL	15.032	3.637%	3s	25.610	5.313%	11s
	MVMoE/4E	**10.428**	**0.896%**	4s	**15.760**	**1.653%**	11s		MVMoE/4E	**14.999**	**3.410%**	4s	**25.512**	**4.903%**	12s
	MVMoE/4E-L	10.434	0.955%	4s	15.771	1.728%	10s		MVMoE/4E-L	15.013	3.500%	3s	25.519	4.927%	11s
OVRP	LKH3	6.511	0.198%	4.5m	9.828	*	5.3m	VRPL	LKH3	10.571	0.790%	7.8m	15.771	*	16.0m
	OR-Tools	6.531	0.495%	10.4m	10.010	1.806%	20.8m		OR-Tools	10.677	1.746%	10.4m	16.496	4.587%	20.8m
	OR-Tools (x10)	6.498	*	1.7h	9.842	0.122%	3.5h		OR-Tools (x10)	10.495	*	1.7h	16.004	1.444%	3.5h
	POMO	6.609	1.685%	2s	10.044	2.192%	8s		POMO	10.491	-0.008%	2s	15.785	0.093%	9s
	POMO-MTL	6.671	2.634%	2s	10.169	3.458%	8s		POMO-MTL	10.513	0.201%	2s	15.846	0.479%	9s
	MVMoE/4E	**6.655**	**2.402%**	3s	**10.138**	**3.136%**	10s		MVMoE/4E	**10.501**	**0.092%**	3s	**15.812**	**0.261%**	10s
	MVMoE/4E-L	6.665	2.548%	3s	10.145	3.214%	9s		MVMoE/4E-L	10.506	0.131%	3s	15.821	0.323%	10s
VRPB	OR-Tools	8.127	0.989%	10.4m	12.185	2.594%	20.8m	OVRPTW	OR-Tools	8.737	0.592%	10.4m	14.635	1.756%	20.8m
	OR-Tools (x10)	8.046	*	1.7h	11.878	*	3.5h		OR-Tools (x10)	8.683	*	1.7h	14.380	*	3.5h
	POMO	8.149	1.276%	2s	11.993	0.995%	7s		POMO	8.891	2.377%	3s	14.728	2.467%	10s
	POMO-MTL	8.182	1.684%	2s	12.072	1.674%	7s		POMO-MTL	8.987	3.470%	3s	15.008	4.411%	10s
	MVMoE/4E	**8.170**	**1.540%**	3s	**12.027**	**1.285%**	9s		MVMoE/4E	**8.964**	**3.210%**	4s	**14.927**	**3.852%**	11s
	MVMoE/4E-L	8.176	1.605%	3s	12.036	1.368%	8s		MVMoE/4E-L	8.974	3.322%	4s	14.940	3.941%	10s

表1. MVMoe模型性能对比。

- POMO在每个单独问题上的表现优于多任务求解器，但平均性能（泛化能力）差。
- MVMoE优于POMO-MTL，表明MVMoE在多任务VRP求解方面具有优势。

	Method	$n = 50$ Obj.	$n = 50$ Gap	$n = 50$ Time	$n = 100$ Obj.	$n = 100$ Gap	$n = 100$ Time	Method	$n = 50$ Obj.	$n = 50$ Gap	$n = 50$ Time	$n = 100$ Obj.	$n = 100$ Gap	$n = 100$ Time
OVRPB	OR-Tools	5.764	0.332%	10.4m	8.522	1.852%	20.8m	OVRPL	OR-Tools	6.522	0.480%	10.4m	9.966	1.783%	20.8m
	OR-Tools (x10)	5.745	*	1.7h	8.365	*	3.5h		OR-Tools (x10)	6.490	*	1.7h	9.790	*	3.5h
	POMO-MTL	6.116	6.430%	2s	8.979	7.335%	8s		POMO-MTL	6.668	2.734%	2s	10.126	3.441%	9s
	MVMoE/4E	**6.092**	**5.999%**	3s	**8.959**	**7.088%**	9s		MVMoE/4E	**6.650**	**2.454%**	3s	**10.097**	**3.148%**	10s
	MVMoE/4E-L	6.122	6.522%	3s	8.972	7.243%	9s		MVMoE/4E-L	6.659	2.597%	3s	10.106	3.244%	9s
VRPBL	OR-Tools	8.131	1.254%	10.4m	12.095	2.586%	20.8m	VRPBTW	OR-Tools	15.053	1.857%	10.4m	26.217	2.858%	20.8m
	OR-Tools (x10)	8.029	*	1.7h	11.790	*	3.5h		OR-Tools (x10)	14.771	*	1.7h	25.496	*	3.5h
	POMO-MTL	8.188	1.971%	2s	11.998	1.793%	8s		POMO-MTL	16.055	8.841%	3s	27.319	7.413%	10s
	MVMoE/4E	**8.172**	**1.776%**	3s	**11.945**	**1.346%**	9s		MVMoE/4E	**16.022**	**8.600%**	4s	**27.236**	**7.078%**	11s
	MVMoE/4E-L	8.180	1.872%	3s	11.960	1.473%	9s		MVMoE/4E-L	16.041	8.745%	4s	27.265	7.190%	10s
VRPLTW	OR-Tools	14.815	1.432%	10.4m	25.823	2.534%	20.8m	OVRPBL	OR-Tools	5.771	0.549%	10.4m	8.555	2.459%	20.8m
	OR-Tools (x10)	14.598	*	1.7h	25.195	*	3.5h		OR-Tools (x10)	5.739	*	1.7h	8.348	*	3.5h
	POMO-MTL	14.961	2.586%	3s	25.619	1.920%	12s		POMO-MTL	6.104	6.306%	2s	8.961	7.343%	8s
	MVMoE/4E	**14.937**	**2.421%**	4s	**25.514**	**1.471%**	13s		MVMoE/4E	**6.076**	**5.843%**	3s	**8.942**	**7.115%**	9s
	MVMoE/4E-L	14.953	2.535%	4s	25.529	1.545%	12s		MVMoE/4E-L	6.104	6.310%	3s	8.957	7.300%	9s
OVRPBTW	OR-Tools	8.758	0.927%	10.4m	14.713	2.268%	20.8m	OVRPLTW	OR-Tools	8.728	0.656%	10.4m	14.535	1.779%	20.8m
	OR-Tools (x10)	8.675	*	1.7h	14.384	*	3.5h		OR-Tools (x10)	8.669	*	1.7h	14.279	*	3.5h
	POMO-MTL	9.514	9.628%	3s	15.879	10.453%	10s		POMO-MTL	8.987	3.633%	3s	14.896	4.374%	11s
	MVMoE/4E	**9.486**	**9.308%**	4s	**15.808**	**9.948%**	11s		MVMoE/4E	**8.966**	**3.396%**	4s	**14.828**	**3.903%**	12s
	MVMoE/4E-L	9.515	9.630%	3s	15.841	10.188%	10s		MVMoE/4E-L	8.974	3.488%	4s	14.839	3.971%	10s
VRPBLTW	OR-Tools	14.890	1.402%	10.4m	25.979	2.518%	20.8m	OVRPBLTW	OR-Tools	8.729	0.624%	10.4m	14.496	1.724%	20.8m
	OR-Tools (x10)	14.677	*	1.7h	25.342	*	3.5h		OR-Tools (x10)	8.673	*	1.7h	14.250	*	3.5h
	POMO-MTL	15.980	9.035%	3s	27.247	7.746%	11s		POMO-MTL	9.532	9.851%	3s	15.738	10.498%	10s
	MVMoE/4E	**15.945**	**8.775%**	4s	**27.142**	**7.332%**	12s		MVMoE/4E	**9.503**	**9.516%**	4s	**15.671**	**10.009%**	11s
	MVMoE/4E-L	15.963	8.915%	4s	27.177	7.473%	11s		MVMoE/4E-L	9.518	9.682%	4s	15.706	10.263%	10s

表2. Zero-shot 泛化表现。

- 在10个未见过的VRP变体上的表现，说明MVMoE的Zero-shot 泛化能力优于POMO-MTL。

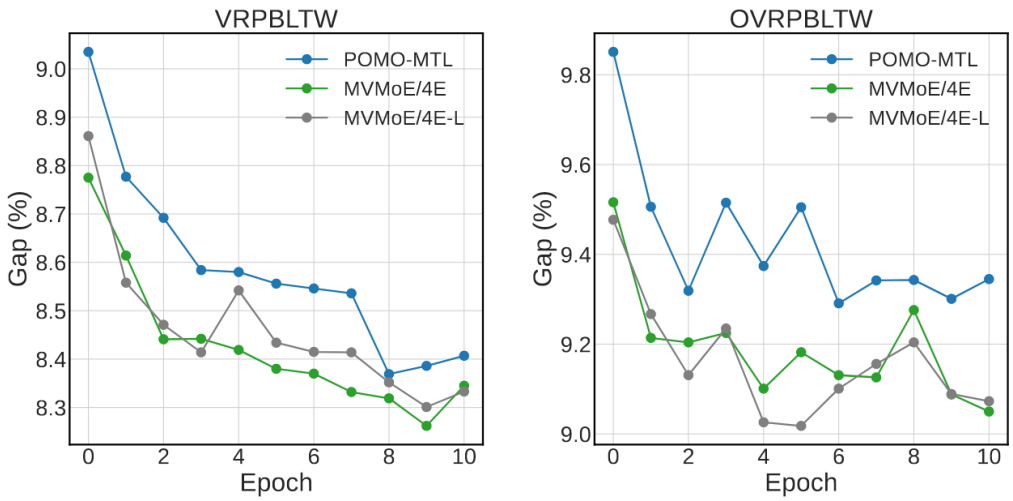

图5. Few-shot 泛化表现对比。

消融实验

探索不同MoE设置对Zero-shot泛化能力的影响。epochs减少至2500，问题规模$n=50$，使用MVMoE/4E。

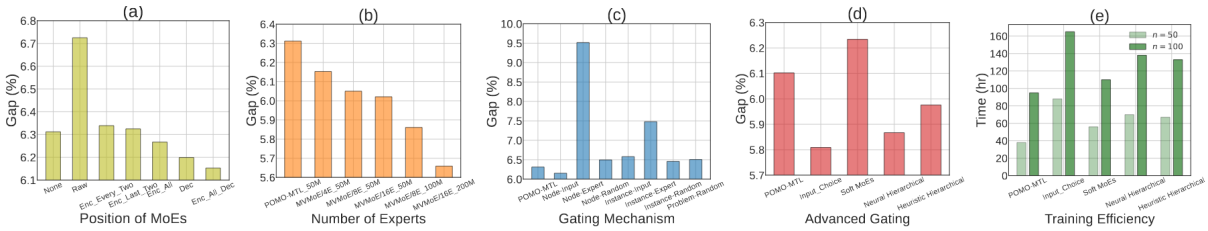

图6. 消融实验结果。

- MoEs位置：Encoder和Decoder处有利于zero-shot泛化。
  - 原始特征处理层
  - 编码器层
  - 解码器层
- 专家数量：先统一使用50M实例进行训练，再使8E额外训练50M实例（总100M），16E额外训练150M实例（总200M）。
  - 专家数量8
  - 专家数量16
- 门控机制：
  - 层次
    - 节点级
    - 实例级
    - 问题级
  - 算法
    - input-choice
    - expert-choice
    - random gating

结论

为建立一个解决VRP问题的更通用和强大的神经网络求解器，本文提出了MVMoE(Multi-task Vehicle Routing Solver with MoEs)以及适用于MVMoE的分层门控机制。第一次尝试建立一个大型的VRP模型。MVMoE也展现出了zero-shot、few-shot的强大泛化能力。但相比于大语言模型的参数规模，MVMoE仍然还是小得多。
未来的研究方向：

1. 解决大规模VRPs问题的可扩展（scalable）基于MoE模型的发展
2. 针对不同问题的通用表征的探索
3. 门控机制可解释性的探索
4. 对MoEs的scaling laws的研究

笔记

- 跨规模泛化

> Fu, Z.-H., Qiu, K.-B., and Zha, H. Generalize a small pretrained model to arbitrarily large tsp instances. In AAAI, volume 35, pp. 7474–7482, 2021.
> Hou, Q., Yang, J., Su, Y., Wang, X., and Deng, Y. Generalize learned heuristics to solve large-scale vehicle

> routing problems in real-time. In ICLR, 2023.
> Son, J., Kim, M., Kim, H., and Park, J. Meta-SAGE: Scale meta-learning scheduled adaptation with guided exploration for mitigating scale shift on combinatorial optimization. In ICML, 2023.
> Luo, F., Lin, X., Liu, F., Zhang, Q., and Wang, Z. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In NeurIPS, 2023.
> Drakulic, D., Michel, S., Mai, F., Sors, A., and Andreoli, J.M. BQ-NCO: Bisimulation quotienting for generalizable neural combinatorial optimization. In NeurIPS, 2023.

- 跨分布泛化

> Zhang, Z., Zhang, Z., Wang, X., and Zhu, W. Learning to solve travelling salesman problem with hardness-adaptive curriculum. In AAAI, 2022.
> Geisler, S., Sommer, J., Schuchardt, J., Bojchevski, A., and Günnemann, S. Generalization of neural combinatorial solvers through the lens of adversarial robustness. In ICLR, 2022.
> Bi, J., Ma, Y., Wang, J., Cao, Z., Chen, J., Sun, Y., and Chee, Y. M. Learning generalizable models for vehicle routing problems via knowledge distillation. In NeurIPS, 2022.
> Jiang, Y., Cao, Z., Wu, Y., Song, W., and Zhang, J. Ensemble-based deep reinforcement learning for vehicle routing problems under distribution shift. In NeurIPS, 2023.

- 跨VRP变体泛化

> Wang, C. and Yu, T. Efficient training of multi-task neural solver with multi-armed bandits. arXiv preprint arXiv:2305.06361, 2023.
> Liu, F., Lin, X., Zhang, Q., Tong, X., and Yuan, M. Multi-task learning for routing problem with cross-problem zero-shot generalization. arXiv preprint arXiv:2402.16891, 2024.
> Lin, Z., Wu, Y., Zhou, B., Cao, Z., Song, W., Zhang, Y., and Senthilnath, J. Cross-problem learning for solving vehicle routing problems. In IJCAI, 2024.

CVRP（容量约束车辆路径问题）

- **图结构**
  设图 $ g = [V, E] $，其中
  - $V = {v_0, v_1, \dots, v_n}$：节点集合
    - $ v_0 $：仓库（depot）
    - ${v_i}_{i=1}^n$：顾客节点（共 $ n $ 个）
  - $ E $：边集合，包含任意两节点 $v_i, v_j $（ $ i ≠ j $ ）之间的边 $ e(v_i, v_j)$。

- **容量约束**
  每个顾客节点 $ v_i $ 有需求 $ d_i ≥ 0 $，每辆车的最大容量为 $ Q $。

- **路径结构**
  解（即路径方案）$ T $ 由多个子路径（sub-tours）组成：
  - 每个子路径表示一辆车从仓库 $ v_0 $ 出发，访问若干顾客节点后返回 $ v_0 $。
  - 需满足：
    1. **唯一性**：每个顾客节点被恰好访问一次。
    2. **容量限制**：每个子路径中所有顾客节点的总需求 $ \sum d_i ≤ Q $。

- **成本函数**
  在欧几里得空间中，路径成本 $ c(T) $ 定义为所有子路径的总长度（即边的欧氏距离之和）。

MoE（混合专家层）

- 包含m个专家${E_1, E_2, \ldots, E_m}$，每个专家是**线性层或FFN**，相互之间参数不共享。
- 门控网络G决定输入如何分配给专家。

> [\text{MoE}(x) = \sum_{j=1}^{m} G(x)_j \cdot E_j(x)\ G(x)_j：第 j 个专家的权重 \ E_j(x)：第 j 个专家的输出]

参考文献(后续阅读)

1. Wang C, Yu T. Efficient training of multi-task neural solver with multi-armed bandits. CoRR. 2023 Jan 1.
2. Lin Z, Wu Y, Zhou B, Cao Z, Song W, Zhang Y, Jayavelu S. Cross-problem learning for solving vehicle routing problems. arXiv preprint arXiv:2404.11677. 2024 Apr 17.
3. **Liu F, Lin X, Wang Z, Zhang Q, Xialiang T, Yuan M. Multi-task learning for routing problem with cross-problem zero-shot generalization. InProceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2024 Aug 25 (pp. 1898-1908).**
4. Kaplan J, McCandlish S, Henighan T, Brown TB, Chess B, Child R, Gray S, Radford A, Wu J, Amodei D. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361. 2020 Jan 23.
5. Floridi, L. and Chiriatti, M. Gpt-3: Its nature, scope, limits, and consequences. Minds and Machines, 30:681–694, 2020.
6. **Joshi, C. K., Cappart, Q., Rousseau, L.-M., and Laurent, T. Learning tsp requires rethinking generalization. In International Conference on Principles and Practice of Constraint Programming, 2021.**
7. **Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In ICLR, 2017.**
8. Puigcerver, J., Riquelme, C., Mustafa, B., and Houlsby, N. From sparse to soft mixtures of experts. In ICLR, 2024.
9. Xue, F., Zheng, Z., Fu, Y., Ni, J., Zheng, Z., Zhou, W., and You, Y. OpenMoE: An early effort on open mixture-of-experts language models. arXiv preprint arXiv:2402.01739, 2024.
10. Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. The Journal of Machine Learning Research, 23(1):5232–5270, 2022b.
11. Yuksel, S. E., Wilson, J. N., and Gader, P. D. Twenty years of mixture of experts. IEEE transactions on neural networks and learning systems, 23(8):1177–1193, 2012.
12. Fedus, W., Dean, J., and Zoph, B. A review of sparse expert models in deep learning. arXiv preprint arXiv:2209.01667, 2022a.
13. **Luo, F., Lin, X., Liu, F., Zhang, Q., and Wang, Z. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In NeurIPS, 2023.**