双重算法推理

Danilo Numeroso比萨大学
danilo.numeroso@phd.unipi.it

Davide Bacciu比萨大学
davide.bacciu@unipi.it

Petar Veli˘ckovi´c
DeepM ind
petarv@deepmind.com

arXiv:2302.04496v1 [cs.LG] 9 Feb 2023

摘要

神经算法推理是机器学习领域一个新兴的方向，旨在将算法计算融入神经网络，通常通过训练神经模型来近似经典算法的步骤。在这一背景下，目前大量研究集中于学习可达性和最短路径图算法，并表明，对相似算法进行联合学习有助于提升泛化能力。然而，当目标转向更复杂的问题时，这类“相似”算法会变得更加难以寻找。在此，我们提出通过利用底层算法问题的对偶性来学习算法。许多算法都在求解优化问题。我们证明，在算法学习中同时学习这些优化问题的对偶定义，能够带来更好的学习效果和质量上更优的解。具体而言，我们利用最大流最小割定理，在合成生成图上同时学习这两种算法，展示了所提出方法的有效性。随后，我们将这一双重算法推理器部署到具有挑战性的脑血管分类任务中，以验证其现实世界的实用性；该任务可能取决于血管的流属性。在这一情境下使用我们的模型时，我们展示了明显的性能提升，并通过实证表明，同时学习最大流算法和最小割算法对于实现这一结果至关重要。

1 引言

学习执行类似算法的计算，是机器学习中的核心问题，长期以来人们从不同角度对此进行了广泛研究，例如学习推理（Khardon & Roth, 1997）、程序解释器（Reed & De Freitas, 2015）和自动定理证明（Rockt¨aschel & Riedel, 2017）。事实上，赋予神经网络推理能力，可能会大幅提升泛化能力，即神经网络超越训练数据支持范围进行泛化的能力；而这通常是当前神经模型面临的一项艰巨挑战（Neyshabur 等人, 2017）。神经算法推理（Velickovic & Blundell, 2021）是对这一长期问题的最新回应，旨在通过学习执行经典算法，训练神经网络展现一定程度的算法推理 能力。可以说，算法的设计具有通用性，能够针对满足一组严格前置条件的任意输入执行并返回“最优”答案。另一方面，神经网络则更加灵活，即能够适应几乎任何输入。因此，根本问题在于，神经模型能否继承算法的一些优良特性，并利用这些特性解决可能颇具挑战性的现实世界问题。

从历史上看，算法学习一直被作为一个简单的监督学习问题来处理（Graves 等人，2014年；Vinyals 等人，2015年），即学习输入与输出之间的映射；或者从强化学习的角度加以处理（Kool 等人，2019）。然而，近期的研究基于算法对齐这一理念（Xu 等人，2020年），指出学习模型的结构必须与目标算法“对齐”，以便降低优化难度。许多研究聚焦于图神经网络（GNN），探索其对图算法的学习能力（Bacciu 等人，2020年），例如 Bellman-Ford 算法（Bellman，1958）。Velickovic 等人（2020b）表明，训练图神经网络来执行经典图算法确实可行。此外，他们还指出，优化必须在图算法的所有中间步骤上进行，让网络学会复现输入的逐步变换，而不是学习从图到所需输出的映射。

自那以后，算法推理已成功应用于强化学习（Deac 等，2021）、物理仿真（Velickovic 等，2021）和二分图匹配（Georgiev & Li´o，2020）。

此外，Xhonneux 等（2021）证实了同时训练多个“相似”算法（多任务学习）的重要性。其依据是，许多经典算法共享子程序，例如 Bellman-Ford 算法和广度优先搜索（BFS），这有助于网络更有效地学习，并能在目标算法之间迁移知识。Ibarz 等（2022）进一步拓展了这一概念，他们构建了一种通用神经算法学习器，能够有效学习执行一组彼此无关的算法。然而，学习某些特定算法可能需要掌握输入数据非常具体的属性，而多任务学习对此未必有帮助。例如，学习用于求解最大流的*Ford-Fulkerson* 算法 （Ford 和 Fulkerson，1956），需要学会识别流网络中的关键（瓶颈）边，即那些边容量降低后会导致最大流减小的边。此外，在单任务情形下，也就是只需学习一种算法时，依赖多任务学习可能会不必要地增加训练阶段的计算负担。

受这些需求的启发，我们寻求其他学习设置，以减轻在多个算法上训练的需求，并提升算法推理器的推理能力。我们发现，目标算法问题的对偶性信息可能是一个不错的候选方案。对偶性这一概念从根本上使算法问题（例如线性规划）能够从两个角度来看待：原始问题和对偶问题。二者通常互为补充，即一个问题的解可能导出另一个问题的解。因此，我们提出将对偶信息直接纳入学习模型，既作为额外的监督信号，也作为输入特征（让网络在算法的后续步骤中复用其对偶预测），并将这一方法称为对偶算法推理（DAR）。据我们所知，尚无先前工作致力于在算法推理中利用对偶性。我们表明，通过训练算法推理器学习算法并优化对偶问题，我们能够放宽需要在多个算法上进行训练这一假设，同时保留多任务学习的所有优势。我们在合成生成的算法任务和真实世界预测性图学习问题上均展示了显著的性能提升。

2 问题陈述

我们研究图上的神经算法推理问题。具体而言，我们旨在学习图算法 A：$\mathbb{G} \to \mathbb{Y}$，这些算法接收图结构输入 $G = (V, E, \boldsymbol{x}_i, \boldsymbol{e}_{ij})$，其中$V$是节点集合， E 是边集合，节点特征为 $\boldsymbol{x}_i$，边特征为 $\boldsymbol{e}_{ij}$，并计算所需的输出 $\boldsymbol{y} \in \mathbb{Y}$。通常，算法 A 的输出空间取决于其作用范围。最一般的情况下，输出可以是 $\mathbb{R}^{|V|}$（节点级输出）、 $\mathbb{R}^{|V|\times|V|}$（边级输出）或 $\mathbb{R}$（图级输出）。我们主要考虑输出节点级和边级结果的算法类别，其中包括许多最知名的图问题，例如可达性、最短路径和最大流。从神经算法推理的角度来看，我们尤其关注学习一系列变换（算法的步骤 ）。因此，我们考虑一系列图 $\{G^{(0)}, \ldots, G^{(T-1)}\}$，其中每个元素都表示我们旨在学习的目标算法的中间状态。在每一步 t，我们都可以获取中间节点和边特征，即 $\boldsymbol{x}_i^{(t)}$， $\boldsymbol{e}_{ij}^{(t)}$，称为提示，以及中间目标 $\boldsymbol{y}^{(t)}$。与经典算法中的常见做法一样，某些中间目标可能会在算法的后续步骤中用作节点/边特征。因此，这些提示会作为附加特征/学习目标纳入训练，从而有效地学习整个步骤序列（算法轨迹）。

具体而言，我们专注于通过最大流的*Ford-Fulkerson* 算法神经执行来进行学习。不同于 Georgiev & Li´o （ 2020 年）通过学习 Ford-Fulkerson 算法来寻找二部图中的独立边集，我们旨在针对一般图学习 Ford-Fulkerson 算法。我们在附录中报告了 Ford-Fulkerson 算法的伪代码。

Ford-Fulkerson 算法带来了两个关键挑战：(i) 它包含两个子程序，即从 s 到 t寻找增广路径，以及在每一步 $\boldsymbol{F}^{(t)} \in \mathbb{R}^{|V|\times|V|}$ 更新流分配 t；(ii) $\boldsymbol{F}$必须遵守一组严格的约束，即边容量约束和流量守恒。前者规定，一个标量值 c_{ij}

（容量）被分配给每个（$i,\ j$）$\in E$，并且 $\boldsymbol{F}$必须满足：

$$\forall (i,j) \in E\ .\ \boldsymbol{F}_{ij} \leq c_{ij}, \tag{1}$$

也就是说，分配给一条边的流量不得超过其容量。后者规定，分配必须满足：

$$\forall i \in V \setminus \{s,t\} : \sum_{(i,j)\in E} \boldsymbol{F}_{ij} + \sum_{(j,i)\in E} \boldsymbol{F}_{ji} = 0 \quad \wedge \sum_{(s,j)\in E} \boldsymbol{F}_{sj} = - \sum_{(j,t)\in E} \boldsymbol{F}_{jt} \tag{2}$$

也就是说，从源点流出的流量不会被中间节点损耗或产生。这也意味着 $\boldsymbol{F} = -\boldsymbol{F}^T$，即反对称性。最大流问题的最优解 $\boldsymbol{F}^*$ 是使网络中的总流量最大化的解，也就是说 $\sum_{(s,\ j) \in E} \boldsymbol{F}_{sj}$。我们通过算法对齐这一概念，并仔细调整和重新缩放模型预测，展示了如何在模型架构中直接应对挑战 (i) 和 (ii)。

3 方法论

3.1 在算法推理中利用对偶性

在学习以神经网络执行经典图算法时，我们利用对偶性这一概念。具体而言，经典算法所求解的大多数问题（包括最大流）都可以表示为约束优化问题，例如线性规划或整数线性规划。在数学优化中，对偶原理确保任何优化问题都可以从两个角度来看待：“直接”解释（称为原始问题）和对偶问题；后者通常由原始问题的拉格朗日函数推导而来。对偶原理确保这两个问题的解要么通过上界/下界关系相联系（弱对偶），要么相等（强对偶）（Boyd 等人，2004）。因此，这两个问题彼此关联。

在神经算法推理的语境下，我们发现了若干值得考虑原始-对偶信息的原因。首先，通过纳入原始-对偶 目标，我们让网络从两个不同且互补的视角对任务进行推理。这可以显著简化对算法的学习，尤其是那些需要识别并推理输入数据中未明确编码的属性的算法。例如，要有效解决最大流问题，网络需要具备识别并推理关键边的能力。根据最大流最小割定理（Ford Jr & Fulkerson，2015年），这组边对应于分隔源节点 s 与汇点 t的最小割，即对偶问题。因此，正确识别最小割对于生成有效的最大流解至关重要。

其次，能够输出更好的逐步解，意味着神经网络执行的算法在整个轨迹中发生误差传播的可能性更小。对于更复杂的算法，这一点尤为重要，例如由多个相互关联的子程序组成的 Ford-Fulkerson 算法。其中，某个子程序的近似不准确，可能会对后续子程序的结果产生连锁负面影响。最后，在原始-对偶问题上联合学习可以视为多任务学习的一种实例，但放宽了必须有多个算法可供训练这一假设。

接下来，我们研究在最大流原始问题中加入最小割对偶信息后的双重算法推理。值得注意的是，已有严格证明表明，即使输入特征不包含有用信息，图神经网络也能够学习最小割（Fereydounian 等人，2022年）。这也意味着，求解最小割可以成为网络学习求解最大流过程中的一个有用“里程碑”。

3.2 架构

我们依托神经算法推理蓝图（Velickovic & Blundell，2021年），并以编码-处理-解码框架（Hamrick 等人，2018年）为基础。Ford-Fulkerson 算法的双重算法推理器（DAR）抽象架构如图1所示。由于该算法由两个子程序组成，我们引入两个处理器，使神经执行与算法的动态过程相一致。第一个处理器 P_{BF} 学习检索增广路径，而P_F 学习执行流更新操作 $\boldsymbol{F}^{(t)}$。 P_{BF} 和 P_F 均通过图

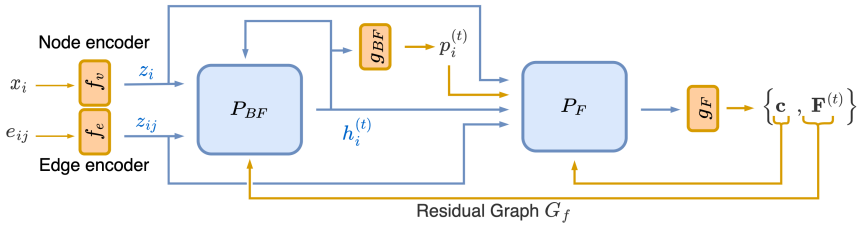


图1：用于 Ford-Fulkerson 算法的双重算法推理器（DAR）的高层架构。详细说明请参阅正文。

网络实现，并采用消息传递神经网络（MPNN）卷积（Gilmer 等人，2017年）：

$$\boldsymbol{h}_i^{(t+1)} = \psi_\theta\left(\boldsymbol{h}_i^{(t)}, \bigoplus_{(j,i)\in E} \phi_\theta\left(\boldsymbol{h}_i^{(t)}, \boldsymbol{h}_j^{(t)}, \boldsymbol{e}_{ij}^{(t)}\right)\right), \tag{3}$$

其中，ψ_θ 和 ϕ_θ 是采用 ReLU 激活的神经网络，而 $\bigoplus$ 是置换不变函数，即求和、求均值或取最大值。

直观而言，编码-处理-解码架构允许将算法步骤的学习与特定输入特征的使用解耦开来。通过学习得到的处理器，算法可以在潜在空间中以神经方式执行；该潜在空间是对原始算法所需输入特征的一种学习所得的表示。我们将展示如何利用这一特性，即使缺少输入特征，也能执行 Ford-Fulkerson 算法的步骤。

更具体地说，DAR 的计算流程包含两个线性 编码器，f_v 和 f_e，分别作用于节点特征 $\boldsymbol{x}_i^{(t)}$ 和边特征 $\boldsymbol{e}_{ij}^{(t)}$，以生成编码后的节点级和边级特征：

$$\boldsymbol{Z}_V^{(t)} = \{\boldsymbol{z}_i^{(t)} = f_v(\boldsymbol{x}_i^{(t-1)}) \mid \forall i \in V\} \quad , \quad \boldsymbol{Z}_E^{(t)} = \{\boldsymbol{z}_{ij}^{(t)} = f_e(\boldsymbol{e}_{ij}^{(t-1)}) \mid \forall (i,j) \in E\}.$$

这些编码后的表示将作为处理器网络 P_{BF} 的输入，用于计算潜在节点表示 $\boldsymbol{H}^{(t)}$：

$$\boldsymbol{H}^{(t)} = P_{BF}(\boldsymbol{Z}_V^{(t)}, \boldsymbol{Z}_E^{(t)}, \boldsymbol{H}^{(t-1)})$$

其中 $\boldsymbol{H}^{(0)} = \{\boldsymbol{0} \mid \forall i \in V\}$。在我们的 DAR 实例中，该处理器会执行 Bellman-Ford 步骤，以按照 Georgiev & Li´o（2020年）的方法，求出从 s到 t的最短增广路径。随后将 $\boldsymbol{H}^{(t)}$ 传递给解码器网络 g，生成增广路径 $\boldsymbol{p}^{(t)}$：

$$p_i^{(t)} = g_{BF}(\boldsymbol{z}_i^{(t)}, \boldsymbol{h}_i^{(t)}). \tag{4}$$

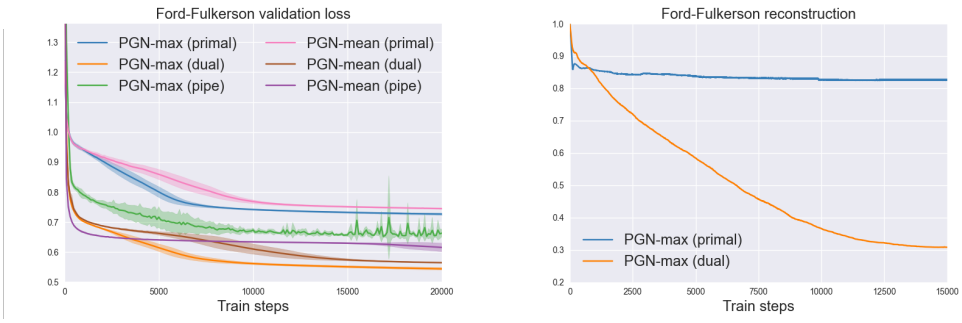
增广路径表示为一个向量，其中包含图中所有节点的前驱节点，即每个条目 $p_i^{(t)}$ 都是指向图中另一个节点 j 的指针。这样，我们就能够从任意节点（包括 t）沿路径回溯至源节点 s。随后将增广路径 $\boldsymbol{p}^{(t)}$ 作为输入特征传递给P_F。最后，算法的目标量，即流分配 $\boldsymbol{F}$和最小割 $\boldsymbol{c}$，预测如下：

$$\{\boldsymbol{F}^{(t)}, \boldsymbol{c}\} = g_F\big(P_F(\boldsymbol{Z}_V^{(t)} \cup \{\boldsymbol{p}^{(t)}\}, \boldsymbol{Z}_E^{(t)}, \boldsymbol{H}^{(t)})\big).$$

不失一般性，我们选择将最小 *s-t* 割 $\boldsymbol{c}$ 表示为节点级特征，其中 $c_i = 0$ 表示 i 属于 s的节点簇，否则为 $c_i = 1$。请注意，最小 *s-t* 割包含所有满足 $c_i = 0$ 和 $c_j = 1$的边（i，j）。此外，$\boldsymbol{F}^{(t)}$ 会在算法的下一步（$\boldsymbol{F}^{(0)} = \boldsymbol{0}$）中再次用作输入特征。

为符合第 3.1 小节所述的最大流问题约束，我们特别关注流分配矩阵 $\boldsymbol{F}^{(t)}$ 的预测。具体而言，我们对 $\boldsymbol{F}$ 进行变换，以确保满足反对称性，即 $\boldsymbol{F}' = \boldsymbol{F} - \boldsymbol{F}^T$。为满足边容量约束，我们还根据双曲正切函数以及每条（i，j）$\in E$ 的实际容量值 c_{ij} 对矩阵进行缩放，如下所示：

$$\boldsymbol{F} = \tanh(\boldsymbol{F}) \odot C, \tag{5}$$



(a) Ford-Fulkerson 算法验证损失　　(b) Ford-Fulkerson 算法重建损失

图 2：**(a)** PGN 在合成数据上的 Ford-Fulkerson 算法验证损失。**(b)** 使用新编码器 l_{ij}，d_{ij}，ρ_{ij}，以及原始和对偶 PGN-max 重建 Ford-Fulkerson 算法的归一化损失曲线。该结果适用于 BVG 数据。

其中，对于图中的所有边，$C_{i,\ j} = c_{ij}$。需要注意的是，这只满足边容量的盒约束；然而，流量守恒仍可能遭到破坏，也就是说，从源点到目标点路径上的节点可能会保留部分流入（流出量小于流入量），反之亦然。为满足这一最后的约束，我们模拟整个神经算法直至终止，并应用校正过程来修正所有流量守恒误差。附录中给出了该过程的伪代码，以及更多细节和结果。

4 实验

为评估对偶算法推理方法的优势，我们在两个特定场景中测试学习模型。首先，我们在合成生成的图上训练并测试 DAR 流水线，以评估其在算法学习这一关键任务中的优势（第 4.1 节）。随后，为评估模型的通用性，我们在真实世界图学习任务上对其进行测试。具体而言，我们在一项具有生物学意义的血管分类任务中，将模型与若干图学习基线进行比较（Paetzold 等，2021）；该任务涉及大规模血管图（第 4.2 节）。需要强调的是，我们不会在真实世界数据上进一步重新训练神经推理器，因此模型必须运用在合成数据上获得的算法知识来解决新任务。

4.1 合成图

数据生成 我们考虑两类不同的图：（i）*2*-社区 图，其中各社区按照概率为 0.75 的*Erd˝os–R´enyi* 分布进行采样，社区内节点之间以 0.05 的概率相连；（ii）二部图 。为全面评估算法推理器的泛化能力，我们仅使用小型*2*-社区 图训练所有模型，并在规模大 4 倍的*2*-社区 图（分布外）以及规模大 4 倍的二部图 （分布族外）上进行测试。需要指出的是，二部图仅用于测试，不会在其上进行任何额外训练。为生成训练集、验证集和测试集，我们遵循标准 CLRS 基准（Veli˘ckovi´c 等，2022年）设置。具体而言，我们采样 1000 个每个包含 16 个节点的*2*-社区训练图。验证集用于评估分布内性能，因此包含 128 个同样具有 16 个节点的*2*-社区 图。为评估分布外和分布族外泛化能力，我们分别使用 128 个 2-社区测试样本和 128 个二部图样本，两者均包含 64 个节点。此外，我们生成 Ford-Fulkerson 算法所有中间步骤的数据，将其用作提示 和额外的训练目标，从而使网络能够在所有中间数据操作上进行训练。算法特征仍按照 CLRS-30 标准生成，包括：（i）输入：源节点 s、汇节点 t、边容量矩阵 $C \in \mathbb{N}^{|V|\times|V|}$ 以及供 Bellman-Ford 处理器使用的额外权重 $W \in [0,1]^{|V|\times|V|}$；（ii）提示 （算法步骤）：增广路径 $\boldsymbol{p}^{(t)}$ 和中间流分配 $\boldsymbol{F}^{(t)}$；（iii）输出（学习目标）：最终流矩阵 $\boldsymbol{F}$ 和最小割 $\boldsymbol{c}$。最后，容量采样为整数

表 1：在*2*-社区 和二部图 上预测最终 $\boldsymbol{F}$和中间流量$\bar{\boldsymbol{F}}^{(t)}$以及最小割 $\boldsymbol{c}$ （如适用）的平均绝对误差（MAE）和准确率。（原始）表示仅使用最大流进行训练。（对偶）表示使用原始-对偶头进行训练。（流水线）表示先学习最小割。（无算法）表示直接优化最大流，而不学习Ford-Fulkerson 算法。

	2-社区 （分布外）			二部图 （分布族外）		
模型	$\boldsymbol{F}$	$\bar{\boldsymbol{F}}^{(t)}$	$\boldsymbol{c}$	$\boldsymbol{F}$	$\bar{\boldsymbol{F}}^{(t)}$	$\boldsymbol{c}$
PGN-max *(原始)*	$0.266_{\pm 0.001}$	$0.294_{\pm 0.002}$	-	$0.56_{\pm 0.23}$	$0.82_{\pm 0.17}$	-
PGN-mean *(原始)*	$0.274_{\pm 0.001}$	$0.311_{\pm 0.004}$	-	$1.09_{\pm 0.47}$	$1.13_{\pm 0.18}$	-
MPNN-max *(原始)*	$0.263_{\pm 0.008}$	$0.289_{\pm 0.004}$	-	$0.75_{\pm 0.47}$	$0.78_{\pm 0.11}$	-
MPNN-mean *(原始)*	$0.278_{\pm 0.008}$	$0.313_{\pm 0.003}$	-	$0.75_{\pm 0.47}$	$0.92_{\pm 0.22}$	-
PGN-max (对偶)	$\mathbf{0.234_{\pm 0.002}}$	$\mathbf{0.269_{\pm 0.001}}$	$100\%_{\pm 0.0}$	$0.49_{\pm 0.22}$	$\mathbf{0.78_{\pm 0.29}}$	$100\%_{\pm 0.0}$
PGN-mean （对偶）	$0.240_{\pm 0.004}$	$0.285_{\pm 0.004}$	$100\%_{\pm 0.0}$	$1.10_{\pm 0.30}$	$1.05_{\pm 0.12}$	$99\%_{\pm 0.7}$
MPNN-max （对偶）	$0.236_{\pm 0.002}$	$0.288_{\pm 0.005}$	$100\%_{\pm 0.0}$	$0.71_{\pm 0.32}$	$0.98_{\pm 0.22}$	$100\%_{\pm 0.0}$
MPNN-mean (对偶)	$0.258_{\pm 0.008}$	$\mathbf{0.268_{\pm 0.002}}$	$100\%_{\pm 0.0}$	$0.81_{\pm 0.09}$	$1.06_{\pm 0.35}$	$100\%_{\pm 0.0}$
PGN-max (流水线)	$0.256_{\pm 0.001}$	$0.293_{\pm 0.003}$	$61\%_{\pm 0.1}$	$\mathbf{0.45_{\pm 0.18}}$	$\mathbf{0.77_{\pm 0.26}}$	$95\%_{\pm 0.1}$
PGN-mean （流水线）	$0.244_{\pm 0.001}$	$0.304_{\pm 0.001}$	$100\%_{\pm 0.0}$	$0.98_{\pm 0.44}$	$1.03_{\pm 0.32}$	$99\%_{\pm 0.8}$
MPNN-max （流水线）	$0.261_{\pm 0.002}$	$0.312_{\pm 0.005}$	$61\%_{\pm 0.3}$	$\mathbf{0.47_{\pm 0.23}}$	$0.95_{\pm 0.34}$	$90\%_{\pm 1.1}$
MPNN-mean (流水线)	$0.255_{\pm 0.002}$	$0.292_{\pm 0.002}$	$100\%_{\pm 0.0}$	$0.64_{\pm 0.35}$	$0.92_{\pm 0.20}$	$100\%_{\pm 0.0}$
随机	$0.740_{\pm 0.002}$	-	$50\%_{\pm 0.0}$	$1.00_{\pm 0.00}$	-	$50\%_{\pm 0.0}$
PGN-max （无算法）	$0.314_{\pm 0.013}$	-	-	$0.78_{\pm 0.02}$	-	-

取自 $U(0,\ 10)$，随后针对*2*-社区 图通过最小-最大归一化进行缩放；而对于二部图，容量则采样为 0 或 1。

消融研究与神经网络架构 我们通过训练一个不含额外最小割头 的相同 DAR 架构，评估对偶的贡献（因此，对偶信息不会在图1的 P_F 中反馈回来）。为了深入分析，我们还考虑一种先于 Ford-Fulkerson 算法学习最小割的神经网络架构。具体而言，我们引入第三个处理器，该处理器仅针对最小割进行训练，其输出随后作为额外特征，用于图1所示的架构。此外，我们比较了两种不同类型的处理器：（i）全连接的消息传递神经网络（MPNN）（Gilmer 等人，2017），它实现公式 3，并在所有节点对之间交换消息；（ii）指针图网络（PGN）（Velickovic 等人，2020a），它只在节点与其由算法输入 和提示 定义的邻居之间交换消息。对于所有处理器，我们尝试了公式 3 中不同的聚合算子，即 $\bigoplus = \{$最大值、平均值、求和$\}$。我们使用随机梯度下降优化器训练所有模型 20,000 个训练轮数，并对 5 次运行的结果取平均。我们还使用教师强制 ，衰减因子为 0.999。这样一来，在训练阶段早期 为网络提供真实的提示，而在大部分训练过程中则让网络预测参与其中。为了选择最优超参数，例如学习率和隐藏维度，我们采用双层随机搜索方案：第一层在较大的取值范围内采样超参数，第二层则根据第一层的结果“细化”搜索。我们根据 $\boldsymbol{F}$上的验证误差选择最佳超参数。图 2(a) 展示了汇总后的验证损失曲线。有关模型选择的更多细节，请参阅附录。

结果分析 我们在表 1 中报告了 Ford-Fulkerson 模拟的结果。具体而言，我们使用平均绝对误差（MAE）作为评估最终流分配预测的指标$\boldsymbol{F}$，其计算方式如公式 5 所示。同样，我们还测量所有中间流分配 $\bar{\boldsymbol{F}}^{(t)}$ 的平均表现，以展示模型在所有步骤中对算法的模仿效果；表 1 中将其称为 $\bar{\boldsymbol{F}}^{(t)}$ 。在适用情况下，我们也报告最小割的准确率，即针对对偶 模型和流水线 模型。为更全面地评估所有模型，我们加入了一个随机基线，该基线随机采样 $\boldsymbol{F}$，并按照公式 5 对其重新缩放；此外还加入了一个直接输出流矩阵 $\boldsymbol{F}$、而不学习 Ford-Fulkerson 算法的图神经网络（标记为无算法）。首先，表 1 显示这些模型的表现明显优于两种基线方法，这表明借助算法推理来学习最大流 （即学习 Ford-Fulkerson 算法）更为有效。更重要的是，我们发现，在*2*-社区 图和二部图 上，纳入对偶问题预测的模型始终优于原始 基线方法。对偶架构也

表 2：对 $\boldsymbol{F}$ 预测结果的定性分析。使用平均绝对误差（MAE）作为预测值与真实最大流值之间的回归误差。为简便起见，我们仅报告表现最佳的模型（PGN）的结果。

	原始		*dual*		流水线	
指标	**PGN-max**	**PGN-mean**	**PGN-max**	**PGN-mean**	**PGN-max**	**PGN-mean**
$\lvert\boldsymbol{F}-\boldsymbol{F}^*\rvert$	7.86 ± 0.47	8.68 ± 0.21	$\mathbf{0.34 \pm 0.04}$	0.41 ± 0.01	7.58 ± 0.10	0.38 ± 0.08

与原始模型相比，在所有中间步骤中对算法的模仿效果更好，这一点可由更低的$\bar{\boldsymbol{F}}^{(t)}$佐证。这表明，尽管对偶最小割信息很容易学到（Fereydounian 等人，2022年），但它仍能帮助模型降低预测误差。以下观察也进一步印证了这一发现：每当最小割预测不准确时，例如*2*-社区图上的 {PGN、MPNN}-最大聚合器流水线，对 $\boldsymbol{F}$和 $\boldsymbol{F}^{(t)}$ 的预测也会随之变差。根据我们的实验，就*2*-社区 图而言，采用最大聚合器的对偶 PGN 架构表现最佳，能够准确预测测试集中所有图的最小割。相比之下，先学习最小割的方式稳定性较差（但仍优于原始基线），这印证了先前关于多任务学习有效性的研究发现。

在二部图上测试分布族外 数据时，性能差距也会增大；此时，对偶和采用最大聚合器的流水线都极具竞争力。我们注意到，在二部图上，记录到的均值和标准差都更高。虽然容量仅采样为 0 或 1，这意味着预测出错的机会更多，从而凸显了这种现象，但这也表明，在任意图分布上的泛化仍然是一项艰巨的任务。

定性分析 为进一步评估 DAR 的性能，我们开展了一项定性研究，结果如表 2 所示。对于*2*-社区 图，我们评估预测的流矩阵 $\boldsymbol{F}$与最优最大流解的接近程度，而不考虑中间节点的误差。这可以衡量网络预测图中的最大流值并将其用于预测解的能力。为此，我们忽略中间节点误差，仅测量从源节点 s 流出并进入汇节点 t的流信号，即 $\sum_{(s,j)\in E} \boldsymbol{F}_{sj}$ 和$\sum_{(j,t)\in E} \boldsymbol{F}_{jt}$。因此，我们取二者之间的最大值（绝对值），并将该值与真实值最大流值 $\boldsymbol{F}^*$进行比较。从表 2 可以观察到，所有对偶架构都得出了能够反映输入图中真实最大流量的解，即平均而言，与最优解的 MAE 为 ≈ 0.30。该分析进一步证实了我们的观点：DAR 模型能够将知识从对偶问题积极迁移至原始问题，从而得到更准确且在定性上更优的解。以下事实也支持了这一观点：对于最小割结果较差的原始架构和对偶架构，它们都以较大幅度偏离了最优解（比较表 1 中的 PGN-max流水线 最小割结果与表 2 中较高的 MAE）。

4.2 真实世界图

基准描述 我们通过考察一项真实世界边分类任务，评估 DAR 流水线的通用性及其潜在影响；在这项任务中，关于最大流概念的先验知识可能会有所帮助。我们在脑血管图（BVG）基准（Paetzold 等，2021）上测试原始架构和 DAR 架构。该基准包含 9 个大型真实世界图，其中边表示血管，节点表示脑网络中血管分支的分叉点。任务是根据以下特征，将每条边分为三类：毛细血管、静脉 和动脉： 血管长度 l_{ij}；分叉点之间的最短距离 d_{ij}；以及曲率 ρ_{ij}。请注意，这三类可以根据血管半径加以区分，或者等效地，根据可通过血管的血流量加以区分。因此，能够模拟整个脑网络中的血流，很可能有助于有效完成这项任务。此外，还需注意，这项分类任务存在严重的类别不平衡：95% 的样本为毛细血管，4% 为静脉，只有 1% 为动脉。

我们在三个 BVG 图上测试模型，分别是 CD1-E-1（规模最大，包含 5,791,309 条边）、CD1-E-2（包含 2,150,326 条边）和 CD1-E-3（包含 3,130,650 条边）。BVG 数据还包含一幅用于验证的合成脑血管图，其中有 3159 个节点和 3234 条边。

算法重构 与我们的合成任务相比，主要区别在于，此处我们需要估算血管直径/半径，这一数值与血管的容量有关，即有多少血液（流量）可以通过该边。因此，容量 是学习目标，而不是要输入算法推理器的特征之一。在这里，我们利用编码-处理-解码架构的通用性，学习重构 Ford-Fulkerson 算法的神经执行。具体而言，我们复用在*2*-社区 图上预训练的 PGN-max 网络（第 4.1 节）。

由于容量 不再是输入特征，我们从 f_v 中移除容量 编码器，并在 f_v中引入三个新的编码器层，分别对应血管图基准测试的每项特征，即 l_{ij}，d_{ij}，ρ_{ij}。因此，除了新引入的编码器层外，我们冻结了预训练模型中的所有参数。由此，我们只训练 l_{ij}，d_{ij}，ρ_{ij} 的权重，以便在缺少容量输入信息的情况下学习 Ford-Fulkerson 算法的步骤。换言之，模型学习利用 l_{ij}，d_{ij}，ρ_{ij} 来估计网络中的边流量，作为边容量的代理信息；这正是我们在 BVG 任务中的主要目标。我们在 BVG 基准测试提供的合成血管图上执行这些算法重构学习步骤。源节点和汇节点 s，t 选为两个随机节点，且它们之间的最短距离等于图的直径。我们使用 Adam 优化器（Kingma 和 Ba，2015年）和 1e-5 的学习率，训练 15000 轮以重构该算法。图 2(b) 比较了原始模型和 DAR 模型在该任务上的损失曲线。

因此，我们通过 PGN-max 的原始和对偶模型，在 CD1-E-*X*上模拟 Ford-Fulkerson 算法的单步运算，并提取每个节点隐藏的学习表示，再将这些表示求和以得到边嵌入。这些边嵌入将作为额外输入特征，用于下文所述、经训练以解决脑血管分类问题的图神经网络。最-后，我们强调，该方法可以轻松导出这些嵌入，因为重建得到的编码器和处理器不会再使用真实数据进行训练。

神经网络架构我们采用 BVG 基准测试论文中的图神经网络作为基线方法，即图卷积网络（GCN）（Kipf 和 Welling，2017年）、GraphSAGE（Hamilton 等人，2017年）以及采用 GraphSAGE 卷积（C– SAGE）的 ClusterGCN（Chiang 等人，2019年）。其通用架构由若干个带 ReLU 激活函数的图卷积层和一个线性模块组成。此外，我们还使用从 PGN-max（原始）和 PGN-max（对偶）提取的嵌入来训练一个简单的线性分类器（LC），以评估这些嵌入相对于原始 l_{ij}，d_{ij}，ρ_{ij} 特征增加了多少信息。我们还将这些表示与 GraphSAGE 和 C–SAGE 结合使用。具体而言，在最终线性层之前，我们将这些嵌入与 GraphSAGE 和 C–SAGE 学习得到的嵌入拼接起来。作为额外的合理性检查，我们还在三个数据集上分别训练 Node2Vec（Grover 和 Leskovec，2016年），并以相同方式拼接其学习得到的嵌入。所有模型均采用 300 轮训练的早停策略，并使用 BVG 论文中给出的最优超参数；为完整起见，我们在附录中报告这些参数。最后，我们对 3 次试验的结果取平均值。

结果分析 BVG 基准测试的结果见表 3。由于学习问题高度不平衡，我们使用平衡准确率（各类别召回率的平均值）和 ROC 曲线下面积作为评估性能的指标。

从线性分类器（LC）的表现来看，算法推理器嵌入（包括原始和对偶嵌入）包含丰富的信息。与简单特征相比，在三个数据集上，其平衡准确率平均提升了 16.6%，ROC 提升了 10.5%。对偶嵌入的表现也优于原始嵌入，两项指标均持续提升，足以证明这一点。图 2(b) 提示，这可能是因为对偶嵌入中的算法重构效果更好，从而产生了信息更丰富的表示。LC 的表现也清楚表明，算法推理器能够在多大程度上将从合成算法任务中获得的知识有效迁移到未见过的真实世界预测图学习任务中。

将学习得到的嵌入与图神经网络架构结合使用时，我们注意到，与原始（即未经过算法增强的）图神经网络相比，性能有了显著提升。采用对偶 嵌入的 C– SAGE 在全部三个数据集上均取得最佳表现，并且在 CD1-E-3 和 CD1-E-2 上始终保持明显的性能优势。有趣的是，对偶嵌入的表现始终优于 Node2Vec 嵌入。考虑到 Node2Vec 直接在 CD1-E-*X* 数据上训练，而 DAR 仅对这些数据进行推理，这一点尤为突出。造成这一性能差距的原因可能是，Node2Vec 本质上通过随机游走拓宽了图神经网络的局部感知野，以期捕获全局图特征

表 3：大型脑血管图上的平衡准确率（平衡准确率）和 ROC 曲线下面积（ROC）性能指标。LC 指线性分类器。除标准架构外，我们还考虑了若干变体，其中最终线性分类层会接收额外的 Node2Vec（Grover 和 Leskovec，2016年）学习得到的嵌入，以及从 PGN-maxprimal 和对偶架构中提取的嵌入。

模型	CD1-E-3 平衡准确率	CD1-E-3 ROC	CD1-E-2 平衡准确率	CD1-E-2 ROC	CD1-E-1 平衡准确率	CD1-E-1 ROC
LC	39。3%$_{\pm 0.2}$	52。3%$_{\pm 0.6}$	36.9%$_{\pm 0.5}$	55.9%$_{\pm 0.1}$	45。5%$_{\pm 0.1}$	61.7%$_{\pm 0.0}$
线性分类器 *(N2V)*	43.9%$_{\pm 0.2}$	55.5%$_{\pm 0.1}$	71。9%$_{\pm 0.1}$	62。6%$_{\pm 0.0}$	46。1%$_{\pm 0.1}$	60。0%$_{\pm 0.0}$
线性分类器 *(原始)*	48.6%$_{\pm 0.4}$	59.4%$_{\pm 0.4}$	58.7%$_{\pm 0.1}$	63.8%$_{\pm 0.2}$	45.3%$_{\pm 0.1}$	59。9%$_{\pm 0.1}$
线性分类器 *(对偶)*	53。8%$_{\pm 0.3}$	66。2%$_{\pm 0.2}$	67。3%$_{\pm 0.1}$	71.8%$_{\pm 0.0}$	48。1%$_{\pm 0.5}$	62。1%$_{\pm 0.3}$
GCN	58.1%$_{\pm 0.5}$	67。9%$_{\pm 0.2}$	74.6%$_{\pm 1.7}$	78。7%$_{\pm 0.1}$	59.0%$_{\pm 0.2}$	67。9%$_{\pm 0.2}$
SAGE	63。5%$_{\pm 0.2}$	70。9%$_{\pm 0.3}$	73.9%$_{\pm 0.6}$	82.5%$_{\pm 0.2}$	64.7%$_{\pm 0.7}$	74。2%$_{\pm 0.2}$
SAGE *(N2V)*	65.0%$_{\pm 0.1}$	71。9%$_{\pm 0.1}$	84.1%$_{\pm 1.9}$	82.5%$_{\pm 0.4}$	65.9%$_{\pm 0.1}$	74.8%$_{\pm 0.1}$
SAGE *(原始)*	64.5%$_{\pm 0.2}$	72.0%$_{\pm 0.2}$	83.8%$_{\pm 0.4}$	83。7%$_{\pm 0.4}$	66。2%$_{\pm 0.5}$	74.8%$_{\pm 0.3}$
SAGE *(对偶)*	66.7%$_{\pm 0.4}$	75.0%$_{\pm 0.2}$	85。2%$_{\pm 0.1}$	85。5%$_{\pm 0.2}$	66.4%$_{\pm 0.3}$	74.8%$_{\pm 0.1}$
C–SAGE	68.6%$_{\pm 0.8}$	74。2%$_{\pm 0.5}$	81.8%$_{\pm 0.5}$	85.6%$_{\pm 0.2}$	59.3%$_{\pm 0.9}$	68。3%$_{\pm 0.5}$
C–SAGE *(N2V)*	68。6%$_{\pm 0.2}$	74.1%$_{\pm 0.1}$	84.8%$_{\pm 0.2}$	84。8%$_{\pm 0.5}$	67.4%$_{\pm 0.6}$	**75.9**%$_{\pm \mathbf{0.2}}$
C–SAGE *(原始)*	67。3%$_{\pm 0.2}$	73。6%$_{\pm 1.9}$	82.5%$_{\pm 1.9}$	84.0%$_{\pm 1.5}$	67.7%$_{\pm 0.1}$	**75.8**%$_{\pm \mathbf{0.2}}$
C–SAGE （对偶）	**70.2**%$_{\pm \mathbf{0.2}}$	**76。3**%$_{\pm \mathbf{0.1}}$	**85.6**%$_{\pm \mathbf{0.2}}$	**86.7**%$_{\pm \mathbf{0.3}}$	**68.1**%$_{\pm \mathbf{0.2}}$	**75.8**%$_{\pm \mathbf{0.1}}$

。相反，DAR 采用了一种更具理论依据的方法，基于图流模拟。这意味着，学习得到的潜在空间编码了重建流分配所需的信息，进而编码边容量；这些信息对于特定任务更具参考价值。DAR 模型还展现出很强的泛化能力。事实上，网络仅在含有 16 个节点的图上训练，却能从含有数百万个节点的图中提取有意义的表示，并在性能上明显优于基线方法。这也可能指明了一条值得探索的途径，以实现可持续的数据高效 图学习模型。

5 结论

我们提出了**双重算法推理**（DAR），这是一种神经算法推理方法，在学习经典算法时利用对偶信息。不同于其他方法，我们放宽了必须联合学习多个算法这一假设，并表明，将算法所针对问题的对偶纳入学习，为学习算法推理器提供了宝贵的信息来源。我们证明，同时学习原始-对偶最大流-最小割问题，能够显著提升预测质量，模型的定量和定性评估结果也印证了这一点。此外，双重算法推理器展现出更强的泛化能力：它们能够在不同图分布系列之间实现积极的知识迁移，并能为大规模图提取有信息量的表示，尽管训练时仅使用了玩具合成图。在此背景下，我们还首次展示了如何通过算法重构，利用算法推理解决更经典的图学习任务。最后，我们指出了若干可能受益于对偶推理方法的问题和算法。首先，最大流和最小割可能代表一大类满足强对偶的原始/对偶问题。在这类问题中，与最大流和最小割的情形类似，可以利用对偶解恢复原始最优值（反之亦然）。这类问题的例子包括最短路径和最小费用流问题。更有意思的是，我们可以将这一方法推广到弱原始-对偶问题，即对偶目标函数是原始目标函数的近似。即便在弱对偶的情况下，对偶信息也很有价值，众多利用原始-对偶关系的算法便证明了这一点（Balinski，1986；Pourhassan 等人，2017年）。特别值得研究的问题包括旅行商问题（Cormen 等人，2009），其对偶形式包括学习输出一棵 1-树（Bazaraa 和 Goode，1977）；以及加权顶点覆盖问题，其对偶可有效用于开发 2-近似启发式算法（Pourhassan 等人，2017年）。我们认为，本文展示的结果应能有力推动这一方向的后续研究，将分析扩展到其他原始-对偶问题对，例如本节所提出的问题。

致谢

本研究部分由 TAILOR 项目资助，该项目由欧盟地平线2020研究与创新计划根据协议编号 952215 资助。

参考文献

Davide Bacciu、Federico Errica、Alessio Micheli 和 Marco Podda。图深度学习入门。神经网络，129：203–221，2020年。

Michel L Balinski。用于指派问题的竞争式（对偶）单纯形法。《数学规划》，34(2)：125–141，1986年。

Mokhtar S Bazaraa 和 Jamie J Goode。旅行商问题：一种对偶方法。《数学规划》，13(1)：221–237，1977年。

Richard Bellman。《应用数学季刊》，关于路径规划问题，16(1)：87–90，1958年。

Stephen Boyd、Stephen P Boyd 和 Lieven Vandenberghe。《凸优化》。剑桥大学出版社，2004年。

Wei-Lin Chiang、Xuanqing Liu、Si Si、Yang Li、Samy Bengio 和 Cho-Jui Hsieh。Cluster-GCN：一种用于训练深层大规模图卷积网络的高效算法。载于*KDD*，第257–266页。美国计算机协会，2019年。

Thomas H Cormen、Charles E Leiserson、Ronald L Rivest 和 Clifford Stein。《算法导论》第3版，第1111–1117页。麻省理工学院出版社，2009年。

Andreea Deac、Petar Velickovic、Ognjen Milinkovic、Pierre-Luc Bacon、Jian Tang 和 Mladen Nikolic。神经算法推理器是隐式规划器。载于*NeurIPS*，第15529–15542页，2021年。

Mohammad Fereydounian、Hamed Hassani、Javid Dadashkarimi 和 Amin Karbasi。图神经网络生成的图函数的精确类别。*CoRR*，abs/2202.08833，2022年。

Lester Randolph Ford 和 Delbert R Fulkerson。《网络中的最大流》。《加拿大数学杂志》，8:399–404，1956年。Lester Randolph Ford Jr 和 Delbert Ray Fulkerson。《网络流》。网络流，普林斯顿大学出版社，2015年。Dobrik Georgiev 和 Pietro Li´o。神经二分匹配。图表示学习及后续研讨会，国际机器学习大会。，2020年。Justin Gilmer、Samuel S. Schoenholz、Patrick F. Riley、Oriol Vinyals 和 George E. Dahl。用于量子化学的神经消息传递。在国际机器学习大会，第70卷机器学习研究论文集，第1263–1272页。PMLR，2017年。Alex Graves、Greg Wayne 和 Ivo Danihelka。神经图灵机。*arXiv*预印本 *arXiv:1410.5401*，2014年。Aditya Grover 和 Jure Leskovec。node2vec：面向网络的可扩展特征学习。在*KDD*，第855–864页。美国计算机协会，2016年。William L. Hamilton、Zhitao Ying 和 Jure Leskovec。大型图上的归纳式表示学习。在*NIPS*，第1024–1034页，2017年。Jessica B. Hamrick、Kelsey R. Allen、Victor Bapst、Tina Zhu、Kevin R. McKee、Josh Tenenbaum 和 Peter W. Battaglia。人类和机器进行物理构建时的关系归纳偏置。在认知科学学会年会。cognitivesciencesociety.org，2018年。

Borja Ibarz、Vitaly Kurin、George Papamakarios、Kyriacos Nikiforou、Mehdi Bennani、R´obert Csord´as、Andrew Joseph Dudzik、Matko Bo˘snjak、Alex Vitvitskyi、Yulia Rubanova、Andreea Deac、Beatrice Bevilacqua、Yaroslav Ganin、Charles Blundell 和 Petar Veli˘ckovi´c。通用神经算法学习器。在 Bastian Rieck 和 Razvan Pascanu（编）所编的第一届图学习会议论文集，第198卷机器学习研究论文集，第2:1–2:23页。PMLR，2022年12月9日至12日。网址：https://proceedings.mlr.press/v198/ ibarz22a.html。

Roni Khardon 和 Dan Roth。学习推理。*ACM*学报（*JACM*），44(5):697–725，1997年。

Diederik P. Kingma 和 Jimmy Ba。Adam：一种随机优化方法。发表于*ICLR*（海报），2015年。

Thomas N. Kipf 和 Max Welling。基于图卷积网络的半监督分类。发表于*ICLR*（海报）。OpenReview.net，2017年。

Wouter Kool、Herke van Hoof 和 Max Welling。注意力，学习求解路径规划问题！发表于*ICLR*（海报）。OpenReview.net，2019年。

Behnam Neyshabur、Srinadh Bhojanapalli、David McAllester 和 Nati Srebro。探索深度学习中的泛化。神经信息处理系统进展，30，2017年。

Johannes C. Paetzold、Julian McGinnis、Suprosanna Shit、Ivan Ezhov、Paul B¨uschl、Chinmay Prabhakar、Anjany Sekuboyina、Mihail I. Todorov、Georgios Kaissis、Ali Ert¨urk、Stephan G¨unnemann 和 Bjoern H. Menze。《全脑血管图：用于图学习和神经科学的数据集与基准》。发表于*NeurIPS*数据集与基准，2021年。

Mojgan Pourhassan、Tobias Friedrich 和 Frank Neumann。《进化算法中最小权顶点覆盖对偶形式的应用》。发表于第*14*届美国计算机协会*/SIGEVO*遗传算法基础会议论文集，第37–44页，2017年。

Scott Reedand Nando De Freitas。《神经程序员解释器》。*arXiv*预印本 *arXiv:1511.06279*，2015年。

Tim Rockt¨aschel 和 Sebastian Riedel。端到端可微证明。神经信息处理系统进展，30，2017年。

Petar Velickovic 和 Charles Blundell。神经算法推理。*Patterns*，2(7):100273，2021年。Petar Velickovic、Lars Buesing、Matthew C. Overlan、Razvan Pascanu、Oriol Vinyals 和 Charles Blundell。指针图网络。载于*NeurIPS*，2020a年。Petar Velickovic、Rex Ying、Matilde Padovano、Raia Hadsell 和 Charles Blundell。图算法的神经执行。载于*ICLR*。OpenReview.net，2020b年。Petar Velickovic、Matko Bosnjak、Thomas Kipf、Alexander Lerchner、Raia Hadsell、Razvan Pascanu 和 Charles Blundell。推理调制表示。*CoRR*，abs/2107.08881，2021年。Petar Veli˘ckovi´c、Adri`a Puigdom`enech Badia、David Budden、Razvan Pascanu、Andrea Banino、Misha Dashevskiy、Raia Hadsell 和 Charles Blundell。CLRS 算法推理基准。*arXiv preprint arXiv:2205.15659*，2022年。Oriol Vinyals、Meire Fortunato 和 Navdeep Jaitly。指针网络。载于*NIPS*，第2692–2700页，2015年。Louis-Pascal A. C. Xhonneux、Andreea Deac、Petar Velickovic 和 Jian Tang。如何迁移算法推理知识以学习新算法？载于*NeurIPS*，第19500–19512页，2021年。Keyulu Xu、Jingling Li、Mozhi Zhang、Simon S. Du、Ken-ichi Kawarabayashi 和 Stefanie Jegelka。神经网络能够推理什么？载于*ICLR*。OpenReview.net，2020年。

A 伪代码

A.1 FORD-FULKERSON 伪代码

我们考虑一种标准的 Ford-Fulkerson 算法实现，如算法1所示。

算法 1 Ford-Fulkerson 算法

输入： $G=(V,E,c)$, s, t;
$f_{uv}=0$： $\forall(u,v) \in E$
当 $\exists$增广路径 $p=s,\ldots,t$ 在 G_f**时**;
  $df=$最小值$\{c_{uv}$： $(u,\ v) \in p\}$;
  对于每个 $(u,v) \in p$ **执行**
    $f_{uv}=f_{uv}-df$
    $f_{vu}=f_{vu}+df$
  结束循环
结束 while 循环
返回 f

A.2 流量校正过程伪代码

校正过程如算法 2 所示。我们区分负节点 i^-，若 $(\sum_{(i,\ j)\in E} F_{ij}+\sum_{(j,\ i)\in E} F_{ji}) < 0$，即节点保留了部分流量；以及正节点 i^+，即流入量与流出量之和为正的节点。因此，我们通过两个简单步骤来强制满足流量守恒约束。首先，所有负节点 i^- 都会向源点 s 送回一定量的流量，其数值等于约束违规幅度，且不考虑边容量。正节点则会减少流向汇点 t的流出量。此步骤后，流量守恒得以满足，但图中出现了容量违规。因此，第二步是重新施加容量约束。从源点 s出发，我们寻找存在容量违规的路径，并将流量值限制在最大容量以内。接着，我们重新调整从 s 发送的流量，直到所有容量违规都得到校正，且不再出现流守恒误差。

算法 2流量校正算法

输入： $G=(V,\ E)$, $\boldsymbol{F}$: 流矩阵， C: 容量矩阵，s: 源点，t: 汇点
$V^-=\{v \mid v \in V \wedge \sum_{(i,\ v)\in E} \boldsymbol{F}_{iv}+\sum_{(v,\ i)\in E} \boldsymbol{F}_{vi}<0\}$
$V^+=\{v \mid v \in V \wedge \sum_{(i,\ v)\in E} \boldsymbol{F}_{iv}+\sum_{(v,\ i)\in E} \boldsymbol{F}_{vi}>0\}$
对每个 $v- \in V^-$，**执行**
  $\varepsilon=|\sum_{(i,\ v)\in E} \boldsymbol{F}_{iv}+\sum_{(v,\ i)\in E} \boldsymbol{F}_{vi}|$
  寻找一条从 v^- 到 s的路径，并将 ε 的流量发送回 s
结束循环
对于每个 $v+ \in V^+$，**执行**
  $\varepsilon=|\sum_{(i,\ v)\in E} \boldsymbol{F}_{iv}+\sum_{(v,\ i)\in E} \boldsymbol{F}_{vi}|$
  查找一条从 v^+ 到 t 的路径，并将流出量减少 ε
结束循环
$\varepsilon_s=|\sum_{(s,\ i)\in E} \boldsymbol{F}_{si}-\sum_{(s,\ i)\in E} C_{si}|$
当 $\varepsilon_s>0$ **时执行**
  寻找一条从 s 到 t 的路径，并通过施加容量约束来减少流出量
  重新计算 ε_s
结束循环
返回 $\boldsymbol{F}$

表4：第一级和第二级随机搜索的详情。

级别	超参数	最小值	最大值	分布
第 1 级	学习率	1e-5	1e-1	对数均匀
	权重衰减	1e-5	1e-1	对数均匀
	隐藏维度	16	512	均匀
第 2 级	学习率	1e-3	1e-2	对数均匀
	权重衰减	1e-3	4e-3	对数均匀
	隐藏维度	60	100	均匀

B 超参数优化

B.1 合成图

对于双层随机搜索的每一层，我们都会采样 $n = 50$ 个配置。表4报告了第一层和第二层随机搜索的详情；第二层搜索根据原始模型的验证结果对第一层搜索进行细化。原始模型的最佳超参数为：隐藏维度 = 68、学习率 = 0.009341、权重衰减 = 0.003420。对偶模型的最佳超参数为：隐藏维度 = 65、学习率 = 0.009868、权重衰减 = 0.001734。流水线 与对偶共用超参数。

B.2 脑血管基准测试

BVG 数据的最优超参数见表5。对于 CD1-E-1，我们仅对 C–SAGE 的超参数进行最少量的优化，因为 Paetzold 等人（2021）报告的最优超参数表现不佳。具体而言，我们通过分别在 $\{2, 3, 4\}$ 和 $\{64, 128\}$ 上进行网格搜索，优化了层数和隐藏单元数。

表5：BVG 基准测试使用的最优超参数。请注意，对于 CD1-E-3，C–SAGE 需要更多训练轮数。

数据集	GCN	SAGE	C–SAGE	Node2Vec
CD1-E-3	学习率=$3 \cdot 10^{-3}$ 层数=3 隐藏单元数=256 丢弃率=0.4 训练轮数=1500	学习率=$3 \cdot 10^{-3}$ 层数=4 隐藏单元数=128 丢弃率=0.4 训练轮数=1500	学习率=$3 \cdot 10^{-3}$ 层数=4 隐藏单元数=128 丢弃率=0.2 训练轮数=5000	学习率=$1 \cdot 10^{-2}$ 游走长度=40 每个节点的游走次数=10 隐藏单元数量=128 训练轮数=5
CD1-E-2	同上	同上	学习率=$3 \cdot 10^{-3}$ 层数=4 隐藏单元数=128 丢弃率=0.2 训练轮数=1500	同上
CD1-E-1	同上	同上	学习率=$3 \cdot 10^{-3}$ 层数=3 隐藏单元=64 丢弃率=0.2 训练轮数=1500	同上

C 补充结果

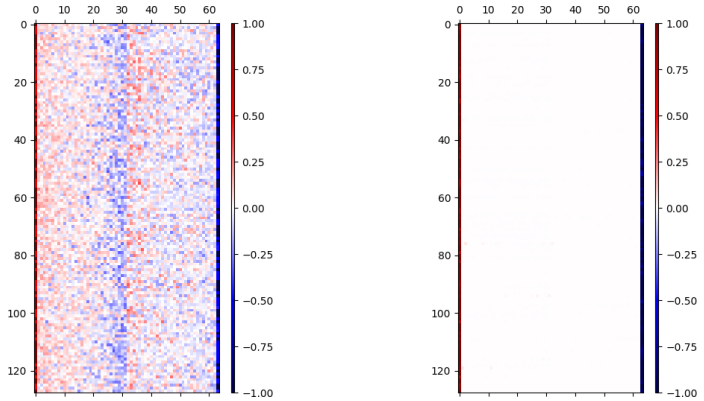
定性研究 在此，我们针对表 1 中的所有模型，在合成图上开展了额外的定性研究，涵盖*2*-社区 图和二部图。为进一步强化表 2 中的定性发现，我们旨在衡量最优 最大流值在多大程度上可以从

表 6：根据学习得到的图表示 $\boldsymbol{h}_g$ 预测最大流值的 R^2 分数。 R^2 指标输出的分数范围为（$-\infty$, 1], 其中 1 为可能的最佳分数。

图	原始		*dual*		流水线	
	PGN-max	**PGN-mean**	**PGN-max**	**PGN-mean**	**PGN-max**	**PGN-mean**
2-社区	0.69 ± 0.12	-0.59 ± 0.33	$\mathbf{1.00 \pm 0.00}$	0.93 ± 0.01	0.80 ± 0.05	-0.13 ± 0.29
二部图	-0.79 ± 0.81	-115.6 ± 64	$\mathbf{0.98 \pm 0.01}$	-115.2 ± 64	-0.69 ± 0.33	$\mathbf{0.98 \pm 0.01}$

学习到的节点表示中线性解码出来。具体来说，我们为测试集中的所有图计算真实的最优最大流值 f_g^*。因此，对于所有模型，我们通过神经网络执行 Ford-Fulkerson 算法，收集学习到的节点嵌入 $\boldsymbol{h}_i$，并通过最大池化操作获得图表示 $\boldsymbol{h}_g$。接着，我们在 $\{\boldsymbol{h}_g\}$ 上拟合线性模型，并计算 R^2 分数来评估预测质量。表 6 显示，利用对偶 嵌入，我们几乎可以完美地线性解码最优最大流 值，即在两种图分布上平均为 ≥ 0.99 。这再次证实，同时学习最小割和最大流是有益的，并能产生信息量更丰富的表示。

强制流守恒 如主论文第 3.2 节所述，算法推理器的最大流输出$\tilde{}$（公式 5）是实际正确解 $\boldsymbol{F}$ 的一个含噪版本（$\boldsymbol{F}$），因为它可能不满足流量守恒约束（公式 2）。我们可以通过算法 2 强制实现流守恒。图 3 以直观示例展示了算法 2 的校正过程如何修正流守恒，并说明可以从神经网络输出中提取正确解。图中，负节点 i^- （见 A.2 小节）以蓝色表示，正节点 i^+ 以红色表示。源点 s 应始终为“正”节点（流量始终流出），而汇点应为“负”节点（流量始终流入）。图 3(b) 中的白色节点表示满足公式 2 的节点。



(a) 校正前的流量　　(b) 校正后的流量

图 3：该图展示了每个测试图中所有节点（横轴）的（归一化）流守恒误差（纵轴）。(a) 显示应用算法 2 前预测的 $\boldsymbol{F}$ 守恒误差。(b) 显示校正后的 $\boldsymbol{F}$ 守恒误差。通过算法 2，我们能够校正所有内部误差（白色表示误差为零）； s 和 t 分别是唯一发送和接收流的节点。

结合 Node2Vec 和对偶嵌入 在表 3 中，Node2Vec 和对偶嵌入是解决脑血管分类任务时信息量最大的节点嵌入。因此，我们通过拼接这两种嵌入，并训练神经网络来解决 BVG 任务，开展了一项额外实验；方法与第 4.2 小节中分别使用原始/对偶/N2V 嵌入时相同。实验结果见表 7。与分别使用这两种嵌入相比，组合嵌入取得了更好的结果，这表明 Node2Vec 和对偶表示编码了两种不同的信息，且可能可以有效结合，即结构信息（前者）和血流信息（后者）。

表 7：Node2Vec 与对偶嵌入拼接的其他 BVG 结果。可与正文中的表 3 进行比较。

模型	CD1-E-3 平衡准确率	CD1-E-3 ROC	CD1-E-2 平衡准确率	CD1-E-2 ROC	CD1-E-1 平衡准确率	CD1-E-1 ROC
线性分类器 *(双重-N2V)*	53.9%$_{\pm 0.1}$	66。3%$_{\pm 0.1}$	77.8%$_{\pm 0.1}$	74。3%$_{\pm 0.1}$	48.2%$_{\pm 0.2}$	62。1%$_{\pm 0.1}$
SAGE *(双重-N2V)*	70。1%$_{\pm 0.3}$	76.5%$_{\pm 0.1}$	89.2%$_{\pm 0.2}$	86。3%$_{\pm 0.2}$	67.5%$_{\pm 0.2}$	76.0%$_{\pm 0.1}$
C–SAGE *(双重-N2V)*	70。4%$_{\pm 0.1}$	76.7%$_{\pm 0.1}$	88.3%$_{\pm 0.3}$	85.0%$_{\pm 0.3}$	67.0%$_{\pm 0.2}$	75.8%$_{\pm 0.3}$

表 8：所有模型在*2*-社区图上的验证结果。

模型	F	$\bar{F}_{(t)}$	c
PGN-max *(原始)*	0.218 ± 0.002	0.272 ± 0.006	-
PGN-mean *(原始)*	0.199 ± 0.001	0.281 ± 0.004	-
MPNN-max *(原始)*	0.215 ± 0.003	0.271 ± 0.003	-
MPNN-mean *(原始)*	0.235 ± 0.008	0.304 ± 0.005	-
PGN-max (对偶)	$\mathbf{0.183 \pm 0.001}$	0.250 ± 0.002	$100\% \pm 0.0$
PGN-mean (对偶)	$\mathbf{0.184 \pm 0.001}$	0.266 ± 0.004	$100\% \pm 0.0$
MPNN-max (对偶)	0.185 ± 0.001	$\mathbf{0.246 \pm 0.002}$	$100\% \pm 0.0$
MPNN-mean（对偶）	$\mathbf{0.184 \pm 0.001}$	$\mathbf{0.246 \pm 0.002}$	$100\% \pm 0.0$
PGN-max（流水线）	0.189 ± 0.002	0.262 ± 0.003	$98.2\% \pm 0.1$
PGN-mean *(流水线)*	0.193 ± 0.001	0.264 ± 0.002	$100\% \pm 0.0$
MPNN-max （流水线）	0.188 ± 0.003	0.257 ± 0.003	$98.3\% \pm 0.1$
MPNN-mean （流水线）	0.204 ± 0.001	0.278 ± 0.002	$100\% \pm 0.0$
随机	0.553 ± 0.003	-	$50\% \pm 0.0$
PGN-max （无算法）	0.256 ± 0.002	-	-

D 验证结果

为了更好地评估模型的算法泛化能力，我们在表 8 中报告了合成数据上的验证结果。此外，我们在图 4 中给出了展示 MPNN-* 模型损失的额外图表，类似于 PGN 在图 2(a) 中的展示。最后，我们还报告了 C–SAGE 模型所有变体（即原始、原始、对偶 和流水线）在脑血管分类任务上的验证表现（见图 5）。

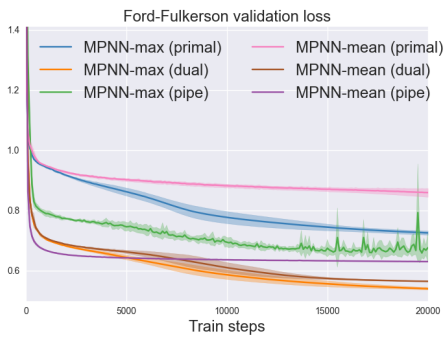


图 4：MPNN 处理器的 Ford-Fulkerson 算法验证损失。

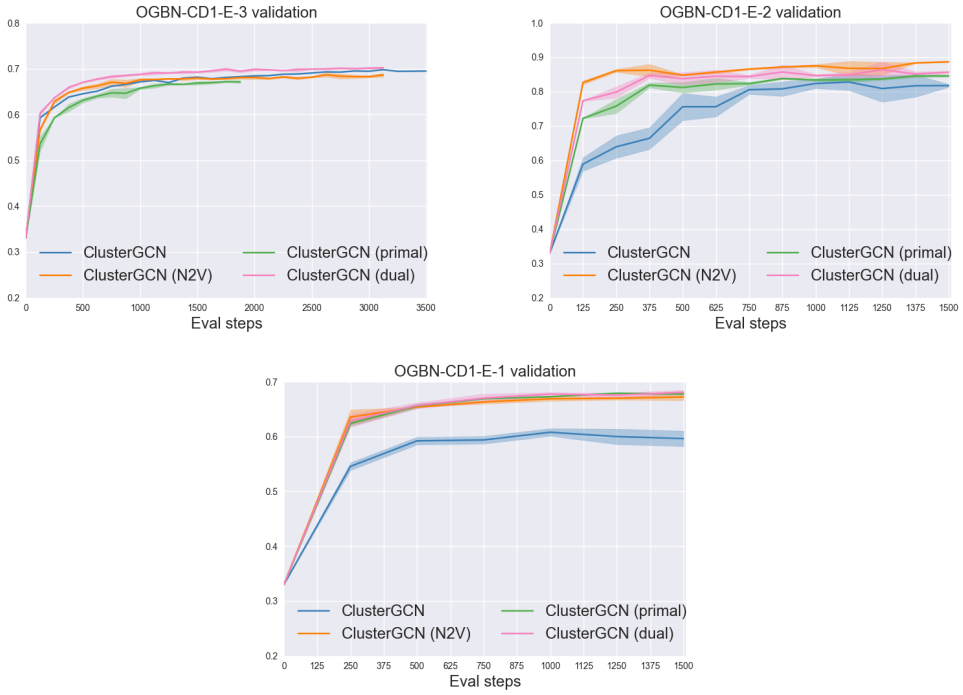


图 5：C–SAGE 各变体在 CD1-E-*X* 上的平衡准确率验证表现。CD1-E-3 图中的截断线表示早停。