《Overcoming Catastrophic Forgetting in Graph Neural Networks with Experience Replay》的复现及改进

林伟兴

摘要

图神经网络（GNNs）[1] 近年来受到了广泛的研究关注，因为它们在各种与图相关的学习任务上表现出卓越的性能。目前的大多数工作集中在静态或动态图设置上，解决单一特定任务，例如节点/图分类、链路预测等。基于这个工作，引发了一个问题：GNNs 能否被应用于持续学习一系列任务？为此，产生了对持续图学习（Continual Graph Learning，CGL）范式的探索，论文提出了基于经验回放的框架 ER-GNN [2] 用于 CGL，以缓解现有 GNNs 中的灾难性遗忘问题。ER-GNN 将先前任务的知识存储为经验，并在学习新任务时重放它们，以减轻灾难性遗忘问题。论文提出了三种经验节点选择策略：特征均值、覆盖最大化和影响最大化，以指导经验节点的选择过程。对三个基准数据集的大量实验证明了 ER-GNN 的有效性，并为增量图（非欧几里得）结构学习提供了启示。

关键词：图神经网络；增量学习；经验回放

1 引言

近年来，应用深度学习方法进行图数据分析任务引起了极大的研究兴趣。已经开发了许多模型来解决各种与图相关的学习任务，包括节点分类、链路预测、更广泛的图分类等。早期的努力主要集中在将网络/图中的节点编码成低维向量空间，同时以无监督的方式保持拓扑结构和节点属性信息。然而，研究人员最近已经从在类似欧几里得的领域（例如图像、文本）上开发复杂的深度学习模型转向了非欧几里得的图结构数据。这反过来产生了许多显着的图神经网络（GNNs）——例如 GCN [3]，GraphSAGE [4]，GAT [5] 等。

尽管在图神经网络（GNNs）方面取得了显著的突破，但现有的模型——无论是在静态还是动态图设置中——主要专注于单一任务。对于 GNNs 而言，按顺序学习多个任务仍然是一个基本挑战。一个自然的问题是，这些流行的 GNNs 在学习一系列与图相关的任务时表现如何，将其称为持续图学习（Continual Graph Learning，CGL）。以 Cora 引文数据为例进行说明，如图 1 所示。起初，引文网络中有几个节点，代表一些属于一组类别的论文。可以在当前图上训练一个节点分类模型。然而，真实的引文网络随着时间的推移自然演变。因此，将一组新的节点（其中一些标记，其他未标记）从新类别添加到图中。我们期望相同的模型能够对新节点进行分类。然而，这种训练过程很容易导致所谓的灾难性遗忘现象（参见图 1 底

部），其中在学习新任务后，分类器被更新并覆盖——这很可能导致以前任务的分类性能显著下降。为了解决这种问题而进行的学习过程我们称之为持续学习。

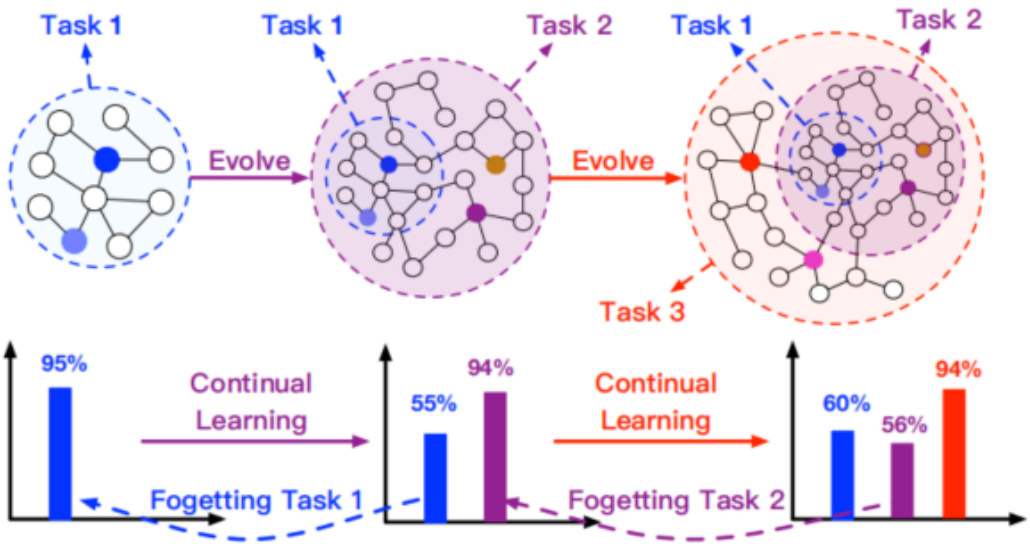

图 1. 一个连续图学习（CGL）的示例。节点表示论文，边表示两篇论文之间存在引用关系。任务是将每个节点分类到预定义的几个类别（例如，主题）。从左到右展示了引文网络的演变过程。不同颜色的直方图表示每个对应任务的节点分类性能。

持续学习 [6]，也称为终身学习、顺序学习或增量学习，最近引起了显著的研究关注。其目标是逐渐扩展已获得的知识以进行未来学习，这与人类智能非常相似。持续学习侧重于按顺序学习多个任务，其两个一般目标是：（i）学习新任务不导致对以前任务的灾难性遗忘和（ii）模型可以利用先前任务的知识来促进学习新任务。灾难性遗忘是神经网络中更一般问题的直接结果，即所谓的“稳定性-可塑性”困境。稳定性表示先前获得的知识的保留，可塑性指的是整合新知识的能力。这种稳定性与可塑性的权衡是人工和生物神经智能系统的一个重要方面。持续学习的现有研究主要侧重于图像分类和强化学习任务，已产生了一些成功的方法，例如 iCaRL [?,7]，EWC [8]，LwF [9] 等。然而，尽管进行了广泛的研究并取得了有望的结果，关于持续图学习的研究却令人惊讶地很少。主要原因有三点：（i）图（非欧几里得数据）不是独立且同分布的数据；（ii）图可能是不规则的、嘈杂的，并且在节点之间展现更复杂的关系；以及（iii）除了节点特征信息之外，图中的拓扑结构在处理与图相关的任务时起着至关重要的作用。为了弥补这一差距，在这项工作中，我们致力于解决这个问题。

针对增量学习问题，原文的主要贡献有以下几点：

- 提出了持续图学习（CGL）范式，并对节点分类问题提出了一个新的持续学习问题。与先前的图神经网络（GNN）工作的主要区别在于，原文的目标是学习多个连续的任务，而不是单一任务。
- 对持续节点分类任务进行了实证研究，展示了在连续学习一系列任务时，现有的 GNNs 在灾难性遗忘方面面临困境。
- 为了解决灾难性遗忘问题，原文开发了一个基于经验回放的通用框架，可以轻松与任何流行的 GNN 模型结合使用。除了两种直观的经验选择方案外，提出了一种基于影响函数的新策略。
- 进行了广泛的实验评估，使用三个基准数据集展示了原文的框架相对于几种最先进的 GNNs

的优越性。

2 相关工作

2.1 图神经网络

图神经网络（Graph Neural Networks，GNNs）最近已成为学习图表示的强大模型。许多优秀的 GNN 模型已被提出，以利用底层图结构信息。它们也潜在地有益于许多现实世界的应用，涵盖从节点分类和链路预测到交通预测和更好推荐的多个领域。目前的大多数 GNNs 可以分为两组：空间方法和频谱方法。空间方法直接从节点的邻域中聚合节点表示，而频谱方法的基本思想是在谱域中学习图表示，其中学到的滤波器基于傅里叶变换（。尽管这些 GNN 在许多与图相关的应用中取得了巨大成功，但它们只学习单一任务。也就是说，它们无法推广到需要在保持模型在先前任务上性能的同时进行持续学习的场景。在这项工作中，我们研究了图学习中的一个新颖但基本的问题，即如何在一系列任务上训练 GNN，其中每个任务都是典型的与图相关的问题，比如节点分类。最重要的是，学到的 GNN 能够成功克服遗忘问题，并允许我们回顾先前模型的行为。

2.2 持续学习

在过去几年中,已经提出了几种方法来解决灾难性遗忘。大致可以区分三种研究方向 [10]:(i) 基于经验回放的方法；（ii）基于正则化的方法；（iii）基于参数隔离的方法。第一类方法以原始格式或在生成模型中压缩的形式存储样本。从先前任务中存储的样本在学习新任务时被重新播放，以避免显著的遗忘。这些样本/伪样本可以用于排练，即近似联合训练先前和当前任务，或者用于限制优化。第二类方法在损失函数中提出了一个额外的正则化项，以在有新数据可用时巩固先前的知识。第三类方法试图通过将不同的参数子集分配给不同的任务来防止可能的对先前任务的任何遗忘。当对体系结构的规模没有约束时，可以通过冻结在每个先前任务之后学到的参数集并为新任务增加新分支来实现。或者，在固定架构下，方法通过识别用于先前任务的部分并在训练新任务时屏蔽它们来进行。这些方法在图像分类和强化学习任务中取得了巨大的成功。然而，它们尚未在图结构数据上进行过研究，这激发了在本文中的研究。原文的方法属于基于经验回放的方法。此外，提出了一种基于影响函数的新经验选择策略，除了两种直观的经验选择方案。

3 本文方法

3.1 问题定义

持续节点分类(即任务增量学习)问题的设置假设存在一组任务：$T = \{T_1, T_2, \ldots, T_i, \ldots, T_M\}$，这些任务按顺序出现，其中每个 $T_i \in T$ $|T| = M$ 都是一个节点分类任务。形式上，节点分类任务被定义为：
(节点分类) 对于每个任务 T_i，我们有训练节点集 D_i^{tr} 和测试节点集 D_i^{te}。节点分类旨在学习一个在 D_i^{tr} 上的任务特定分类器，其预期将 D_i^{te} 中的每个节点正确分类为相应的类别 $y_i^l \in Y_i$ ，其中 $Y_i = \{y_i^1, y_i^2, ..., y_l^i, ..., y_i^L\}$ 是标签集，L 是任务 T_i 中的类别数量。

在持续图学习设置中，我们不再关注单个任务 T_i，而是需要学习一系列节点分类任务集合 T。也就是说目标是学习一个由 θ 参数化的模型 f_θ，该模型可以逐步学习这些任务。具体而言，我们期望分类器 f_θ 不仅在当前任务上表现良好，而且在先前任务上克服灾难性遗忘。

3.2 损失函数定义

受 CLS(Complementary Learning Systems) [11] 理论启发，原文提出了一个新颖而通用的框架，被称为 ER-GNN，该框架从当前任务中选择并保留经验节点，并在将来的任务中重新播放它们。原文的 ER-GNN 框架概述如 Algorithm 1 所示。

在学习任务 Ti 时，获取其训练集 D_i^{tr} 和测试集 D_i^{te}。随后，我们从经验缓冲区中选择示例。然后，将训练集 D_i^{tr} 和经验节点一起输入到分类器 f_θ。节点分类任务的一个自然损失函数选择是交叉熵损失函数：

$$\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}) = -(\sum_{(\boldsymbol{x}_i, y_i)\in\mathcal{D}} (y_i \log f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \quad + (1-y_i)\log(1-f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)))).$$

为了保持缓冲区中数据与当前数据的节点数目之间的平衡，对损失函数进行一个加权求和，令损失函数为：

$$\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}, B) = \beta\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_i^{\text{tr}}) + (1-\beta)\mathcal{L}_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, B).$$

其中，$\beta = |B|/(|\mathcal{D}_i^{\text{tr}}| + |B|)$,

3.3 节点经验回放

原文采用了三种方法进行经验回放 [12] 节点的选择：Mean of Feature (MF)、Coverage Maximization (CM)、Influence Maximization (IM)。

Mean of Feature (MF)：直观地说，每个类别中最具代表性的节点是那些与平均特征向量最接近的节点。对于每个任务，我们计算每个类别的原型，并选择与该原型最接近的前 e 个节点以形成经验。

$$\mathbf{c}_l = \frac{1}{|S_l|}\sum_{(\boldsymbol{x}_i, y_i)\in S_l} x_i, \mathbf{c}_l = \frac{1}{|S_l|}\sum_{(\boldsymbol{x}_i, y_i)\in S_l} \boldsymbol{h}_i,$$

Coverage Maximization (CM): 当每个类别中的经验节点 e 的数量较小时，最大化属性/嵌入空间的覆盖可能是有帮助的。为了最大化属性/嵌入空间的覆盖，根据在固定距离 d 内同一任务中来自其他类别的节点数量对每个类别中的节点进行排名：

$$\mathcal{N}(v_i) = \{v_j | \text{dist}(v_i - v_j) < d, \mathcal{Y}(v_i) \neq \mathcal{Y}(v_j)\}$$

Influence Maximization (IM)：在每个任务 T_i 的训练过程中，我们可以从训练集 D_i^{tr} 中移除一个训练节点 v_*，得到新的训练集 D_{i*}^{tr}。然后，我们可以计算最优参数 θ_* 为：

$$\theta_\star = \arg\min_{\theta\in\Theta}(\mathcal{L}'_{\mathcal{T}_i}(f_{\boldsymbol{\theta}}, \mathcal{D}_{i\star}^{\text{tr}}, B)).$$

4 复现细节

4.1 与已有开源代码对比

本文对原论文 [2] 的开源代码进行了改进，并在整体架构上借鉴了源代码，以将原论文的算法扩展到其他关于图的数据集。主要工作涵盖以下四个方面：

1. 原论文结果复现：在构建原文代码的环境的基础上，我们对原文中的图数据集进行了复现。通过这一过程，我们成功地再现了原文的结果，并与原文进行了基本的比较验证；

2. 数据集处理：我们选择了常见的图数据集 Arxiv 和 Products，并进行了细致的数据处理，以确保这些数据集能够适用于图增量学习任务。这一阶段还包括将数据集划分为训练集和测试集，以便更好地评估模型的性能；

3. 算法实现：我们编写了代码来实现核心算法，主要目标是计算三种经验选择方法对应的前 e 个节点。这些节点将在增量学习的经验回放过程中被使用，从而进一步优化模型的性能；

4. 新数据集上的增量学习：我们将经过处理的数据集应用于 ergnn 的方法，进行了图增量学习训练。随后，我们在测试集上对模型的性能进行了全面的测试，以验证模型在处理新数据集上的增量学习任务时的效果。这一系列步骤旨在确保模型的鲁棒性和泛化能力。

4.2 实验环境搭建

在本研究中，我们主要选择了 PyTorch 框架来构建神经网络模型。所采用的 Python 版本为 3.7.10，PyTorch 版本为 1.9.1，对应的 CUDA 版本为 11.6。在复现过程中，我们面临的主要挑战是 DGL 版本与 CUDA 的适配问题。为了解决这一问题，我们采取了一系列措施，其中包括更新 CUDA 版本以及对原始 DGL 代码的更新。这些调整旨在确保相应的 DGL 函数能够顺利运行 ERGNN 框架的增量学习代码，为研究的顺利进行提供了必要的技术支持。这一过程也反映了在开展深度学习研究时，技术调整和适配是不可避免的一部分。

4.3 创新点

对于图神经网络中的另外两个常见数据集 Arxiv 和 Products，我们进行了详细的数据处理，以确保这些数据集能够适用于图增量学习任务。在进行数据处理的同时，我们将数据集划分为训练集和测试集，以便更好地评估模型的性能。随后，我们采用了 ergnn 的方法对处理后的数据集进行图增量学习训练，并在测试集上进行了充分的性能测试。这一系列步骤的目的是确保模型在应对图增量学习任务时表现出色。

5 实验结果分析

本部分对实验所得结果进行分析，详细对实验内容进行说明，实验结果进行描述并分析。

实验结果详见表 1 和表 2。表 1 呈现了在复现过程中采用了原文中的 Cora、Citeseer 和 Reddit 三个图数据集所得到的增量学习结果。我们不仅仅复现了这些结果，还将它们与原文中的实验结果进行了详尽的比较。令人鼓舞的是，我们的复现结果基本上与原文提供的实验结果相一致，从而充分证明了我们成功实现了代码的复现工作。

Datasets	Cora	Citeseer	Reddit
原文PM	95.66%	81.83%	95.36%
复现PM	95.54%	82.01%	95.21%
原文FM	21.03%	17.08%	23.09%
复现FM	22.05%	16.58%	23.21%

表 1. 复现结果

	nodes	edges	classes	tasks	PM	FM	Time
Arxiv	169343	1166243	40	20	88.65%	14.43%	11h
Products	2449028	61859036	46	223	95.77%	12.25%	108h

表 2. 新数据集的信息及实验结果

这一成功的代码复现过程不仅验证了我们对算法的理解和实施的准确性，也为后续的实验工作奠定了坚实的基础。此外，通过与原文结果的一致性验证，我们进一步确保了实验的可靠性和重复性，为该研究领域的深入探索提供了可信的实验基础。这也强调了科学研究中复现的重要性，使得研究社区能够建立在可靠的实验基础上展开深入的讨论和进一步的创新。

表 2 所呈现的内容构成了本文的主要创新点，具体体现在将原文提出的 ergnn 方法成功地应用于另外两个广泛使用的图数据集，即 Arxiv 和 Products，以进行图增量学习训练。这一创新的实验设计旨在探索 ergnn 方法在不同数据背景下的适用性和性能表现。在这一过程中，我们不仅仅进行了模型的训练，还在测试数据集上对其进行了详尽的测试，以全面评估该方法在不同场景下的图增量学习性能。

这一实验设置不仅为我们提供了对 ergnn 方法在新领域中表现的深入理解，同时也为我们了解模型的泛化能力以及对不同数据集的适应性提供了重要线索。通过在 Arxiv 和 Products 数据集上进行的增量学习实验，我们期望进一步拓展对该方法在真实世界应用中的理解，从而为图数据的增量学习任务提供更加全面和可靠的解决方案。这一系列工作将有助于推动图神经网络领域的研究，促使更多实用性强、适用范围广的增量学习方法的发展。

6 总结与展望

图增量学习作为图神经网络领域的前沿研究方向，其核心目标是使模型在面对新任务时能够灵活适应而不丧失先前任务的学习成果。通过采用经验回放等策略，该领域已经在保留模型知识、提高迁移性能等方面取得了显著进展，对于实现长期模型演进和适应动态环境具有重要意义。未来的研究方向可聚焦于处理动态图的挑战，将图增量学习扩展到更多领域如医疗、金融等，以解决实际问题。同时，考虑对抗性学习，提高模型在面对恶意攻击时的稳健性，也是一个重要的发展方向。此外，提升图增量学习模型的可解释性，使其决策过程更具可理解性，将进一步增强模型在实际应用中的可信度。综合而言，图增量学习将在未来持续推动图神经网络领域的创新，为更加复杂、动态的现实场景提供可靠而灵活的解决方案。

参考文献

[1] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Deep neural networks for learning graph representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[2] Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4714–4722, 2021.

[3] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.

[4] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

[5] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[6] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE international conference on computer vision*, pages 1320–1328, 2017.

[7] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[8] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[9] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[10] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

[11] Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20(7):512–534, 2016.

[12] Tim de Bruin, Jens Kober, Karl Tuyls, and Robert Babuška. Improved deep reinforcement learning for robotics through distribution-based experience retention. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3947–3952. IEEE, 2016.