GRAND+：可扩展的图形随机神经网络

冯文正[1], 董玉晓[1], 黄庭林[1], 尹子奇[2], 徐成[13] Evgeny Kharlamov[4], 唐杰[1]
1清华大学, 2北京理工大学
3腾讯公司，4博世人工智能中心
fwz17@mails.tsinghua.edu.cn,{yuxiaod,tl091,jietang}@tsinghua.edu.cn,ziqiyin18@gmail.com
alexcheng@tencent.com,evgeny.kharlamov@de.boosch.com

ABSTRACT

图神经网络（GNNs）已被广泛用于图的半监督学习。最近的一项研究表明，图随机神经网络（GRAND）模型可以为这个问题产生最先进的性能。然而，GRAND很难处理大规模的图，因为它的有效性依赖于计算上昂贵的数据增强程序。在这项工作中，我们提出了一个可扩展和高性能的GNN框架--GRAND+，用于半监督的图学习。为了解决上述问题，我们在GRAND+中开发了一个通用前推（GFPush）算法，预先计算一个通用的传播矩阵，并采用它以小批量的方式进行图数据增强。我们表明，GFPush的低时间和空间复杂度使GRAND+能够有效地扩展到大型图。此外，我们在GRAND+的模型优化中引入了一个置信度感知的一致性损失，促进了GRAND+的泛化优势。我们在七个不同规模的公共数据集上进行了广泛的实验。结果表明，GRAND+：1）能够扩展到大型图，并且比现有的可扩展GNN花费更少的运行时间；2）在所有数据集上比全批和可扩展GNN都能提供一致的准确性改进。

CCS概念

• **计算方法学→半监督学习设置**； -**信息系统**→社会网络。

关键字

图谱神经网络；可扩展性；半监督学习

ACM参考格式：

冯文正，董玉晓，黄庭林，尹子奇，徐成，Evegeny Kharlamov，唐杰。2022.GRAND+：可扩展的图随机神经网络-作品。在*2022年ACM网络会议（WWW'22）论文集中，2022年4月25-29日，虚拟活动，法国里昂。*ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3485447.3512044

该代码可在https://github.com/wzfhaha/GRAND-plus。唐杰是通讯作者。

WWW'22，2022年4月25-29日，虚拟活动，法国里昂。
ACM ISBN 978-1-4503-9096-5/22/04... $15.00
https://doi.org/10.1145/3485447.3512044

1 简介

图结构在我们的物理世界和虚拟世界中都很常见，如社会关系、化学键和信息扩散等。现实世界图数据固有的不完整性激发了人们对图上半监督学习问题的巨大兴趣[19, 33]。到目前为止，图神经网络（GNNs）已经被许多人认为是解决这个问题的*事实上的*方法[1, 11, 19, 28, 29]。简而言之，GNNs利用数据样本之间的图结构来促进模型预测，使它们能够产生比传统的半监督学习方法更突出的性能改进[32]。

然而，基于GNN的半监督学习解决方案仍然存在挑战。值得注意的是，GNN的泛化通常并不构成其优势，因为它们中的大多数只使用监督损失来学习参数[9, 19, 28, 29]。这种设置使得模型容易对有限的标记样本进行过度拟合，从而降低了对未见过的样本的预测性能。为了克服这个问题，图形随机神经网络（GRAND）[12]为GNNs设计了图形数据增强和一致性正则化策略。这些设计使其在Cora、Citeseer和Pubmed的半监督学习中比现有的GNN有明显的性能提升。

具体来说，GRAND开发了随机传播操作，以产生有效的结构数据增强。然后，对它进行训练，包括对已标记节点的监督损失和对未标记节点的不同扩增的一致性正则化损失。为了实现良好的图增强，GRAND中的随机传播提出使用混序邻接矩阵来传播特征矩阵。传播基本上需要在每个训练步骤中对邻接矩阵进行幂级迭代，这使得GRAND在扩展到大规模图时面临计算上的挑战。

为了解决这个问题，我们提出了GRAND+框架，用于大规模的图上半监督学习。GRAND+是一种可扩展的GNN一致性正则化方法。在GRAND+中，我们引入了有效的近似技术，以小批量的方式进行随机传播，解决了GRAND的可扩展性限制问题。此外，我们通过采用置信度感知损失来调节不同图形数据增强之间的一致性，从而改进GRAND。这种设计稳定了训练过程，并为GRAND+提供了良好的泛化能力。具体来说，GRAND+包括以下技术：

- *广义的特征传播*：我们提出一个广义的混序矩阵来进行随机特征传播。这种矩阵

它提供了一套可调整的权重来控制不同等级邻域的重要性，从而为处理复杂的现实世界图提供了一个灵活的机制。

- *高效的近似*：受最近基于矩阵近似的GNNs[5, 8]的启发，GRAND+采用了一种近似方法---广义前推（GFPush）--有效地计算出

广义的传播矩阵。这使得GRAND+能够按

以小批量的方式形成随机传播和模型学习，为该模型提供了显著的可扩展性。

- *信心感知的损失*：我们为GRAND+正则化框架设计了一个信心感知损失。这有助于在一致性训练中通过忽略高度不确定的未标记样本来过滤掉潜在的噪音，从而提高GRAND+的泛化性能。

我们在七个不同类型和规模的公共图数据集上进行了综合实验，以证明GRAND+的性能。总的来说，与10个GNN基线相比，GRAND+在三个基准数据集上产生了最好的分类结果，并在其他四个相对较大的数据集上超过了5个有代表性的可扩展GNN，而且效率很高。例如，GRAND+在Pubmed上达到了最先进的85.0%的准确性。在有1240万个节点的MAG-Scholar-C上，GRAND+比FastGCN和GraphSAINT快10倍，并且在运行时间相当的情况下，比PPRGo--以前在这个数据集上最快的方法--提供了4.9%的精度提升。

2 半监督的图形学习

2.1 问题

在基于图的半监督学习中，数据样本是或---

图$G = (V, E)$，其中每个节点$\in V$代表一个

数据样本，$E \in V \times V$是一组边，表示每对节点之间的关系-船舶。我们使用$A \in \{0, 1\}$。$|\square| \times |\square|$来代表$G$的邻接矩阵，每个元素$A(s, \square) = 1$表示$s$和$\square$之间存在一条边，否则$A(s, \square) = 0$。

$\tilde{}$

D是对角线度矩阵，其中$D(\square, \square) = \sum_{\square} A(\square, \square)$用来表示添加了自环连接的图$\square$。相应的邻接矩阵为$\tilde{A} = A + I$，程度矩阵为

是$\tilde{D} = D + I$

在这项工作中，我们关注的是分类问题，每个样本s都与1）特征向量$X_s \in X \in \mathbb{R}^{|\square| \times d_f}$和2）标签向量$Y_s \in Y \in \{0, 1\}^{|\square| \times \square}$代表了类的数量。$|V| \times \square$，$C$代表类的数量。在半监督设置中，只有有限的节点$L \in V$有观察到的标签（$0 < |\square| \ll |V|$），其余节点的标签$U = V - L$是未见的。半监督图学习的目的是根据图结构$\square$、节点特征$X$和观察到的标签$Y_{\square}$[1]，推断出未被标记的节点$U$的缺失标签$Y_{\square}$。

2.2 相关工作

图形神经网络（GNNs）已被广泛采用于半监督图形学习问题的修正。在这一部分，我们回顾了GNNs的进展，重点是它们对半监督图学习的大规模解决方案。

[1] 对于矩阵$M \in \mathbb{R}^{a \times \boxtimes}$，我们用$M_{\boxtimes} \in \mathbb{R}^{\boxtimes}$来表示其$\square$-th行向量，并让$M(\square, \square)$代表第1行和第1列的元素。

图卷积网络。图卷积网络（GCN）[19]将卷积操作概括为图。具体来说，第$\square$个GCN层定义为：：

$$H^{(l+1)} = \sigma(\hat{A} H^{(l)} W^{(l)}), \quad (1)$$

其中，$H^{(\square)}$表示第$\square$个隐藏节点的表示。

含有H的层 $^{(0)} = X$, $\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$是对称的归一化$\square$的邻接矩阵，$W^{(\square)}$表示第三层的权重矩阵，$\square(\cdot)$表示激活函数。在实践中，这个图形卷积过程将被重复多次，最终的表征通常被送入逻辑回归层进行分类。

简化图卷积。通过仔细观察方程1，我们可以发现图卷积包括两个操作：*特征传播*$\hat{A} H^{(\square)}$和*非线性变换*$\square(\cdot)$。Wu等人[29]通过去除隐藏层中的非线性变换来简化这一过程。由此产生的简化图卷积（SGC）被表述为：

$$\hat{Y} = \mathrm{softmax}(\hat{A}^N X W), \quad (2)$$

其中，$\hat{A}^N X$被认为是一个简化的N层图对X的卷积，W是指用于分类的可学习权重矩阵，$\hat{Y}$表示模型的预测。

具有混合顺序传播的GNNs。正如Li等人[23]所指出的，根据马尔科夫链收敛定理，$\hat{A}^N X$将随着N的增加而收敛到一个固定点，即过度平滑问题。为了解决这个问题，一种典型的方法[5, 20, 21]建议使用一个更复杂的混序矩阵来处理特征

传播。例如，APPNP[20]采用了截断的每-----

归纳的网页排名（ppr）矩阵 Π^{ppr} = $\square \sum \alpha(1 - \square)^n \hat{A}^n$，其中

的超参数 征兆 $\square = 0$

$\in (0, 1)$表示远距离传输概率，使模型能够保留本地信息，即使在

$N \to +\infty$.

可扩展的GNNs。大体上，有三类方法被提出来用于使GNNs可扩展：1) 节点采样方法

采样策略，以加快递归特征的速度。

聚合程序。代表性的方法包括Graph- SAGE[14]、FastGCN[7]和LADIES[34]；2）图的分割方法试图将原始大图分割成几个小的子图，并在子图上运行GNNs。这类方法包括Cluster-GCN[10]和GraphSAINT[31]；3）矩阵近似方法遵循SGC[29]的设计，将特征传播和非线性转换解耦，并利用一些近似方法来加速特征传播。所提出的GRAND+框架与基于矩阵逼近的方法高度相关，如PPRGo[5]和GBP[8]。我们将在第3.6节中分析它们的区别。

3 大+框架

在本节中，我们简要回顾了图随机神经网络（GRAND），并介绍了其用于大规模半监督图学习的可扩展解决方案GRAND+。

3.1 图形随机神经网络

最近，Feng等人[12]介绍了用于半监督节点分类的图神经网络（GRAND）。GRAND是一个GNN

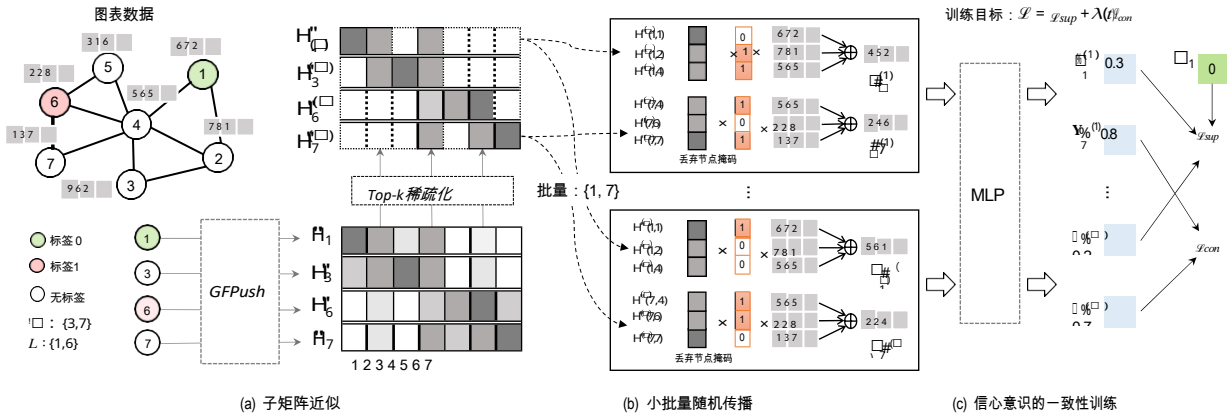


图1：GRAND+的说明。(a) GRAND+采用*Generalized Forward Push* (*GFPush*)和*Top-k稀疏化*，对$L \cup U$中的节点进行传播矩阵Π的相应行近似。(b) 然后利用得到的稀疏化行近似进行小批量随机传播，为批量中的节点生成增强值。(c) 最后，将计算出的特征增量送入MLP进行置信度感知的一致性训练，该训练同时采用监督损失L_{sup}和置信度感知的一致性损失L_{con}进行模型优化。

一致性正则化框架，优化了不同增量中未标记节点的预测一致性。

具体来说，它设计了*随机传播*-一种混合顺序的传播策略--来实现图数据的增强。首先，用DropNode--一种dropout的变体，随机地丢弃节点特征X。然后，由此产生的被破坏的特征矩阵用混序矩阵在图上传播。而不是PPR矩阵、

进行传播。形式上，随机传播策略被模拟为：

$$\bar{X} = \Pi_{\text{征}}^{\text{avg}} \cdot \text{diag}(z) \cdot X,\ z_i \sim \text{Bernoulli}(1-\delta), \tag{3}$$

where $z \in \{0,1\}^{|V|}$表示从Bernoulli(1-δ)中抽取的随机DropNode掩码，而δ代表DropNode概率。这样一来，每个节点丢失的信息就由其邻域来补偿。在图形数据的同质性假设下，产生的矩阵X可以被看作是原始特征矩阵X的有效数据增强。由于DropNode的随机性，这种方法在理论上可以为每个节点产生指数级的许多增强值。

在GRAND的每个训练步骤中，随机传播程序被执行了M次，导致M增强的特征矩阵$\{\bar{X}^{(s)} \mid 1 \le s \le M\}$。然后将所有增强的特征矩阵输入MLP，得到$M$的预测结果。在优化过程中，GRAND的训练既包括对已标记数据的标准分类损失，也包括对未标记节点集U的额外*一致性正则化*损失[4]，即：

$$\frac{1}{M \cdot |U|}\sum_{s\in U}\sum_{m=1}^{M}\left\|\hat{Y}_s^{(s)} - \bar{Y}_s\right\|_2^2,\quad \bar{Y}_s = \frac{1}{M}\sum_{s=1}^{M}\hat{Y}_s^{(s)}, \tag{4}$$

其中$\hat{Y}_s^{(s)}$是MLP对节点s的预测概率，当使用$\bar{X}^{(s)}$作为输入。一致性损失提供了一个额外的regu-

通过强制神经网络对不同的无标签数据进行类似的预测，从而达到拉长的效果。通过运行
穹顶传播和一致性正则化，GRAND实现了
比传统的GNN有更好的泛化能力[12]。

GRAND 的可扩展性。在实践中，当n较大时，对增强矩阵$\hat{A}^n$的第n幂进行计算是不可行的[25]。为了避免这个问题，GRAND 采用幂级迭代法直接计算整个增强特征矩阵 X（在公式 3 中），即迭代计算并求出 $\hat{A}$ 和 $\hat{A}^n$-diag(z) - X 的乘积为 $0 \le n < N$。这个过程是用稀疏密集矩阵乘法实现的，具有线性时间

M次，在每个训练步骤中产生不同的特征增强。因此，T训练步骤的总复杂度变成了$O(nMT(|V|+|E|))$。然而，它需要对
复杂度为$O(n(|V|+|E|))$。
非常昂贵的。

3.2 GRAND+的概述

我们提出GRAND+来实现基于图的半监督学习的可扩展性和准确性。它遵循GRAND的一般一致性正则化原则，并包括一些技术，使其可以扩展到大型图，同时保持GRAND的灵活性和概括能力。

简而言之，我们在GRAND+中开发了一种高效的近似算法--广义for-ward push（GFPush），以预先计算传播矩阵所需的行向量，并以小批量的方式进行随机传播，而不是用功率迭代来传播特征。该程序的时间复杂度由一个预定义的超参数控制，避免了GRAND所面临的可扩展性限制。此外，GRAND+采用了一种新的置信度感知损失来实现一致性正则化，这使得
训练过程更加稳定，导致更好的概括性
表现比GRAND好。

传播矩阵。在GRAND+中，我们提出以下基因

侵蚀的混序矩阵用于特征传播：

$$\Pi = \sum_{n=0}^{N} w_n \cdot P^n,\quad P = D^{-1}A \tag{5}$$

其中 $\sum_{n=0}^{N} w_n = 1$，$w_n \ge 0$，P是行规范化的邻接性矩阵。与GRAND中使用的传播矩阵不同的是

和其他GNN，Π的形式采用了一组可调整的权重
通过调整□，≤ N级混合不同顺序的邻接矩阵，模型可以灵活地操纵其重要性。

□

的不同阶数的邻域，以适应现实世界中不同结构属性的不同图形。

训练管道。为了实现快速训练，GRAND+放弃了直接计算整个增强特征矩阵X的功率迭代方法，而是为每个节点单独计算每个增强特征向量。理想情况下，节点□的增强特征向量X□的计算方法是：

$$\bar{X}_s = \sum_{v \in N_\pi^s} z_v \cdot \Pi(s, □) \cdot X_v ,\ z_v \sim \text{Bernoulli}(1 - □)。 \quad (6)$$

这里我们用Π_s来表示Π的行向量，对应于节点□，N□被用来表示Π，Π(□, □)表示Π的第□个元素。这种范式允许我们在每个训练步骤中只为一批节点生成增强的特征，从而使我们能够使用高效的小批量梯度下降法进行优化。

然而，在实践中很难计算出Π_s的确切形式。技巧。为了解决这个问题，我们开发了几种有效的方法

来逼近GRAND+中的Π_s。近似过程由两个阶段组成。在第一阶段，我们提出了一种有效的方法--*通用前推法*（*GFPush*）来计算行向量的误差约束近似值。在第二阶段，我们采用顶层*稀疏化*策略来截断Π，使其仅包含Π□的最大元素。得到的稀疏化行近似于

$\tilde{\Pi}^{(□)}$被用来计算X□，作为Π□的替代（公式6）。为了提高效率，需要预先计算出相应的行训练中使用的所有节点的近似值。除了标记的节点，GRAND+还要求无标签的节点执行一致的

在训练过程中，GRAND+采用了正则化技术。为了进一步提高效率，GRAND+没有使用全部的U，而是对一个较小的未标记节点的子集$U' \subseteq U$进行一致性正则化。如图1所示，GRAND+的训练流水线包括三个步骤：

- *子矩阵近似。*我们得到一个稀疏化的行近似通过GFPush对每个节点$s \in L \cup U'$进行建模$\Pi^{(k)}$和顶部-□稀疏化。由此产生的稀疏化子矩阵被用于以支持随机传播。
- *小批量的随机传播。*在每个训练步骤中，我们从$L \cup U'$中抽出一批节点，用近似的行向量为该批中的每个节点生成多个增强值。
- *信心意识到的一致性训练。*我们将增强的特征送入MLP以获得相应的预测，并通过监督损失和置信度感知的一致性损失对模型进行优化。

3.3 子矩阵近似

广义前推（GFPush）。可以看出，行归一化的邻接矩阵$^{-1}$也是□上的反向随机行走过渡概率矩阵[8]，其中行向量P□表示从节点□开始的随机行走过渡概率。基于这一事实，我们提出一种有效的算法

称为Generalized Forward Push (GFPush)，以近似于行向量。的r Π_s= □=0□□P□，误差有限。GFPush的灵感来自于

前推[2]算法用于逼近个性化的PageRank向量，同时具有更高的灵活性，能够逼近广义的混序矩阵Π。GFPush的核心思想是模拟一个N步的随机行走概率扩散过程，从s用一系列的剪枝操作进行ac- celeration。为了达到这个目的，我们应该在每一步n（$0 \le n \le N$）保持一对向量：1）*储备向量*$q^{(n)} \in R^{|□|}$，表示在第n步保留的概率质量；2）*剩余向量*$r^{(n)} \in R^{|□|}$、代表超过步骤□的扩散概率质量。

算法1显示了GFPush的伪代码。在开始时、$r^{(0)}$和$q^{(0)}$都被初始化为指标向量$e^{(□)}$，其中$e^{(□)}_s$=1，$e^{(□)}$=0，对于□≠□，意味着随机行走开始时□，概率质量为1。其他储备和残留物向量（即$r^{(n)}$和$q^{(n)}$，$1 \le n \le N$）被设置为 0→。然后，该算法算法以□步迭代更新储备和残值向量。

在第n迭代中，算法对满足$r^{(n-1)}$ > d□ · r_m□□的节点□进行*推送*操作（算法1的第5-9行）。这里d□=D(v, □)代表□的度数，r_m□□是一个预定的阈值。在推送操作中，v的残差$r^{(n-1)}$是均匀的□

分布到它的邻域，并将结果存储到□的残余向量$r^{(n)}$中。同时，储备向量 $q^{(n)}$也被更新为与 $r^{(n)}$相同。在完成对□的推送操作后，我们将$r^{(□-1)}$重置为0，以避免重复更新。

为了更直观地了解这一程序，我们可以观察到$r^{(n-1)}$/d□是随机行走的条件概率。从□到邻接节点□，条件是它到达□时有上一步的概率$r^{(□-1)}$。因此，每次推送操作在□上，可以被看作是一个单步随机行走的概率扩散从v到其邻域的过程。为了保证效率，GFPush只对残差值大于$d_v \cdot r_{max}$的节点v进行推动操作。因此，当第□迭代完成后，$q^{(n)}$可以被看作是对□步数的随机行走的近似描述

观念向量P^n。而$\tilde{\Pi}$□= $\sum_{□=}$ □$q^{(□)}$相应地被认为是作为算法所返回的Π_s的近似值。

理论分析。我们有以下定理，关于时间复杂度、内存复杂度和近似度的界限

GFPush的错误。

定理1。*算法1的时间复杂度为*O(N /r_{max})，*内存复杂度为*O(N /r_{max})，*并返回*Π_s*作为*Π□*的近似值，其误差界限为：* ‖Π□ - Π□‖$_1$ ≤ □ · (2|E| + |□|) · r_{max}。

证明。见附录A.2。 □

定理1表明，GFPush的近似精度和运行成本与□□□□呈负相关。在实践中，我们可以使用□□□□来控制效率和近似精度之间的权衡。

顶层稀疏化（Top-k Sparsification）。为了进一步降低训练成本，我们对...进行顶部k稀疏化。在这个过程中，只保留了Π中最大的元素，其他条目被设置为0。这样，该模型只考虑了最多k的 □ ~□

算法1： GFPush

输入 ：自环增强图G，传播步骤N，节点□，阈值rm□，权重系数□$_n$，0≤n≤N。

输出： 节点□的过渡向量Π□的近似值。

$r^{(n)} \leftarrow 0 \rightarrow$ 对于n = 1, …, □; $r^{(0)} \leftarrow e^{(□)}$ ($e^{(□)}$ = 1, $e^{(□)}$ = 0 对于$v \neq$ □)。

$q^{(n)} \leftarrow 0 \rightarrow$ 对于n = 1, …, □; $q^{(0)} \leftarrow e^{(□)}$。

对于□ = 1 : □
do

  如果*存在节点v，且*$r^{(n-1)}$>d□$_{□}$$rm$□□**做**

    对于*每个*u∈N$_v$**做**

      /* N□是□在图□ 中的邻居集。 */

      $r_{□}^{(n)} \leftarrow r_{□}^{(n)} + r_{□}^{(□-1)}$ /d□□。

      $q_{□}^{(n)} \leftarrow r_{□}^{(n)}$。

    结束

    $r_{□}^{(□-1)} \leftarrow 0$。

    /* 对□ 进行推操作。 */

  结

结束 $_s$ **束**

← -□ $_{□=}$ w_n - $q^{(n)}$

返回 Π~$_s$ 。

在随机传播中，每个节点的重要邻域，根据聚类假设[6]，这仍然被认为是有效的。类似的技术也被PPRGo[5]所采用。我们将在第4.5节中实证检验□的效果。

并行化。 在GRAND+中，我们需要对$L \cup U'$中的所有节点进行行向量近似。可以很容易地检查出，在GFPush中，不同的行近似是相互独立计算的。因此，我们可以启动多个工作器，同时对多个向量进行近似。这个过程是这样实现的

在我们的实现中采用多线程编程。

3.4 小批量随机传播

GRAND+采用Π的稀疏行近似值来执行

以小批量的方式进行随机传播。具体来说，在

在第19个训练步骤中，我们随机抽取一批有标签的节点L_s来自□，而一批未标记的节点□来自U'。然后，我们计算节点s∈L□ ∪U□的增强特征向量X□：

$$\bar{X}_{□} = \sum_{□∈N(□)} z_v \cdot Π^{(□)}(s, □) \cdot X_v \,,\ z_v \sim \text{Bernoulli}(1 - □)、 \quad (7)$$

其中N□ $^{(□)}$ 表示Π$_s$ 的非零指数，X$_v$∈R $^{~(□)}$ □□ 是结点□的特征向量。在每个训练步骤中，我们生成□

重复这个过程M次，就可以得到增强的特征向量{X$^{(□)}$ |1 ≤ m ≤ □}。让b = |□□ | + |□$_t$ | 表示批次大小。那么每个批次的时间复杂度以O(□ - □ - □□)为界。

可学习表征的随机传播。 在公式7中，增强的特征向量X$_s$是用原始特征X计算的。然而，在一些实际应用中（如图像或文本分类），X的维度可能非常大，这将产生巨大的计算成本。为了缓解这个问题，我们可以采用线性层将每个X□转化为低维的隐藏表示H$_v$∈R^{dh} 首先，然后用H进行随机传播：

$$\bar{X}_{□} = \sum_{□∈N(□)} z_v \cdot Π^{~(□)}(s, □) \cdot H_v \,,\ H_v = X_v \cdot W^{(0)}、 \quad (8)$$

其中W$^{(0)}$∈R$^{df \times dh}$表示可学习的变换矩阵。这样，该程序的计算复杂度就降低到O(□-□-□$_h$)，其中dh≪$_{df}$表示H□的维度。

预测。 在训练期间，增强的特征向量X$^{(□)}$ $_{□}$

被送入一个MLP模型，以获得相应的输出：

$$\hat{Y}_s^{(□)} = \text{MLP}(X_s, Θ)、 \quad (9)$$

其中Y$^{\hat{}}$ $^{(m)}$ $_{□}$ ∈[0, 1]^{C}表示□的预测概率。Θ代表MLP的参数。

3.5 信心意识的一致性训练

GRAND+在训练过程中采用监督分类损失和一致性正则化损失来优化模型参数。监督下的损失被定义为以下方面的平均交叉熵

标记的节点的多个增量：

$$L_{sup} = -\frac{1}{|L□| \cdot M} \sum_{s∈Lt} \sum_{m=1}^{M} Y_s \cdot \log(\hat{Y}_{□}^{(□)}) 。 \quad (10)$$

信心意识的一致性损失。 受半监督学习的最新进展启发[4]，GRAND采用了一个附加的一致性损失来优化未标记数据的多次增强的预测一致性，这在提高泛化能力方面被证明是有效的。GRAND+也遵循了这一思想，同时采用了新的置信度感知的一致性损失来进一步提高有效性。

具体来说，对于节点s∈$_U$□，我们首先通过取其M预测概率的平均值来计算分布中心，即：、

$\bar{Y}_{□}$= $\frac{1}{□}$ $\sum_{□=1}$ $\hat{Y}_{□}^{(m)}$ /□□。然后我们将*锐化*[17]的技巧应用于$\bar{Y}_{□}$来 "猜 "出节点□的伪标签$\tilde{Y}_{□}$。从形式上看，节点j的第s类上的猜测概率是通过以下方式获得的：

$$\tilde{Y}(□, □) = \bar{Y}(□, □)^{\frac{1}{□}} / \sum_{□=0}^{1□-1} \bar{Y}(□, □)^{\frac{1}{□}} \quad (11)$$

其中，0 < τ ≤ 1 是一个超参数，用于控制尖锐度。猜测的伪标签。随着□的减少，$\bar{Y}$□ 被强制变为更加尖锐，并最终收敛于单热分布。那么，在无标签节点批上的置信度感知的一致性损失□□被定义为：

$$L□□_n = \frac{1}{|□□| \cdot |U_t|} \sum_{s∈Ut} I(\max(\bar{Y}_s) ≥ \gamma) \sum_{□=1}^{M} D(\bar{Y}_{□}, \hat{Y}_{□}^{(□)})、 \quad (12)$$

其中I(max($\bar{Y}$□)≥□)是一个指标函数，如果max($\bar{Y}$□)≥γ成立则输出1，否则输出0。0 ≤ □ < 1 是一个预定的阈值。D(p, □)是一个距离函数，衡量p和□之间的分布差异。这里我们主要考虑D的两个选项：□$_2$ *距离*和*KL分歧*。与 GRAND 使用的一致性损失相比（参见公式 4），L□□$_n$最大的优势在于它在优化过程中只考虑由阈值 τ决定的 "高度自信 "的无标签节点。这种机制可以通过过滤掉不确定的伪标签来减少潜在的训练噪声进一步提高模型的实际性能。结合L□□$_n$和L$_{sup}$，可得模型优化的最终损失被定义为：：

$$L = L_{□□□} + □ \cdot (□)^{L} \cdot □□□、 \quad (13)$$

算法2：GRAND+

输入 :图□，特征矩阵$X \in R^{|V| \times d_f}$，有标签的节点集□，无标签的节点集U和观察标签$Y \in R^{|□| \times □}$。

输出：分类概率$\hat{Y}^{(□nf)}$。

1 从U中抽取一个无标签节点的子集U'。
2 **对于**$s \in L \cup U'$**做**
3 $\Pi_s \leftarrow$ GFPush(G, □)。
4 通过对Π_s进行顶-□稀疏化得到$\tilde{\Pi}^{(□)}$。
/* 用泛化的方式逼近行向量。 */
5 **结束**
6 **for** □ = 0 : □ **do**
7 对一批有标签的节点$□_t \subseteq L$和一批无标签的节点进行采样$U_t \subseteq U'$。
8 **对于**$s \in L_□ \cup U_□$**做**
9 **对于**□ = 1 : □**做**
10 生成增强的特征向量$\bar{X}_□$。用方程7生成增强的特征向量X□。
11 用$\hat{Y}^{(□)}$ = MLP($X^{(□)}$, Θ) 预测类的分布。
12 **结束**
13 **结束**
14 通过公式 10 计算 L_{sup}，通过公式 12 计算 $L_{□□n}$。通过小
15 批量梯度下降来更新参数Θ：Θ = Θ - ∇_Θ($L_{sup} + \lambda L_{con}$)。
/* 用提前停止的方式停止训练。 */
16 **结束**
17 推断分类概率$\hat{Y}^{(□nf)}$ = MLP(Π - (1 - □) - X, Θ)。
18 **返回** $\hat{Y}^{(□nf)}$。

其中，□(□)是一个线性预热函数[13]，随着训练步骤□的增加，它从0线性增加到最大值□□□。

模型推断。训练结束后，我们需要推断出未标记节点的预测结果。在推理过程中，GRAND+采用幂级迭代的方式计算未标记节点的精确预测结果：

$$\hat{Y}^{(□n□)} = \text{MLP}(\sum_{□=0}^{N} w_n (□) - (1-□)\text{-X}, \Theta), \quad (14)$$

其中我们用(1-□)重新标定X，以使其与训练中使用的DropNode扰动特征的期望值。

请注意，与GRAND不同的是，上述功率迭代过程只需要在GRAND+中执行一次，而计算成本在实践中是可以接受的。与获得预测值相比

与训练中的GFPush相比，这种推理策略在理论上可以提供更准确的预测。算法2显示了GRAND+的整个训练和推理过程。

3.6 模型分析

复杂度分析。我们对GRAND+不同学习阶段的时间复杂度进行了详细分析。根据定理1，近似阶段（算法2的第2-5行）的复杂度为O((|U'| + |L|) - N /□□$_a$□)。至于训练阶段（算法2的第6-15行），T训练步骤的总复杂度为O(k -b -M -□)，这对大图来说实际上是有效的，因为b和□通常远小于图的大小。推理阶段（算法2第17行）的复杂度为O((|V| + |E |) - N)，与节点和边的数量之和成线性关系。

GRAND+ vs. PPRGo 和 GBP。与GRAND+类似，PPRGo[5]和GBP[8]也采用矩阵逼近的方法来扩展GNNs。然而，GRAND+在几个关键方面与这两种方法不同。PPRGo通过使用Forward Push[2]来扩大APPNP的规模。

表1：数据集的统计数据。

数据集	节点	边缘	课堂	特点
Cora	2,708	5,429	7	1,433
Citeseer	3,327	4,732	6	3,703
公共医学杂志	19,717	44,338	3	500
琥珀-CS	593,486	6,217,004	18	100
睿迪特（Reddit）公司	232,965	11,606,919	41	602
Amazon2M	2,449,029	61,859,140	47	100
MAG-Scholar-C	10,541,560	265,219,904	8	2,784,240

近似于PPR矩阵。与PPRGo相比，由于采用了广义的传播矩阵Π和GFPush算法，GRAND+在实际应用中更加灵活。GBP也拥有这个

通过使用广义的PageRank矩阵[22]进行特征分析，来确定优点。

传播。然而，它直接近似于传播的乘通过双向传播[3]对原始特征的结果进行处理，其

计算复杂度与原始特征尺寸呈线性关系，因此难以处理具有高维特征的数据集。此外，与PPRGo和GBP的设计不同的是在一般的监督分类问题上，GRAND+会对其进行统计分析。

通过采用随机传播和一致性正则化来提高泛化能力，对半监督设置进行了重大改进。

4 实验

4.1 实验设置

基线。在我们的实验中，我们将GRAND+与五个最先进的全批次GNNs-GCN[19]，GAT[28]，APPNP[20]、GCNII[9]和GRAND[12]，以及五个有代表性的可扩展GNN--FastGCN[7]、GraphSAINT[31]、SGC[29]、GBP[8]、和PPRGo[5]。对于GRAND+，我们实现了三个变体，分别是传播矩阵Π的不同设置（参考方程5）：

- GRAND+ (P): *截断的ppr 矩阵* $\Pi^{ppr} = \sum^{N} \alpha(1-□)^n P^n$。
- GRAND+（A）：*平均集合矩阵* $\Pi^{平均数} = \sum_{□=0} -□ P^□/(□+1)$.
- GRAND+ (S)：*单阶矩阵* $\Pi^{单} = P^□$ □= 。

数据集。实验在七个不同规模的公共数据集上进行，包括三个广泛采用的基准图--Cora、Citeseer和Pubmed[30]，以及四个相对较大的图--AMiner-CS[12]、Reddit[14]、Amazon2M[10]和MAG-scholar-C[5]。对于Cora、Citeseer和Pubmed，我们使用公共数据拆分[19, 28, 30]。对于AMiner-CS、Reddit、Amazon2M和MAG- Scholar-C，我们使用20×#classes节点进行训练，30×#classes节点进行验证，其余节点进行测试。表1中总结了相应的统计数据。更多关于设置和可重复性的细节可以在附录A.1中找到。

4.2 基准数据集的结果

为了评估GRAND+的有效性，我们在Cora、Citeseer和Pubmed上将其与10个GNN基线进行比较。按照社区惯例，基线模型在这三个基准上的结果取自以前的工作[9, 12, 28]。对于GRAND+，我们用随机种子进行了100次试验，并报告了平均精度和相应的标准差。

表2：基准的分类准确率（%）。

类别	方法	科拉	馨予	医学博士
全批次的GNNs	GCN	81.5 ± 0.6	71.3 ± 0.4	79.1 ± 0.4
	GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
	APPNP	84.1 ± 0.3	71.6 ± 0.5	79.7 ± 0.3
	GCNII	85.5 ± 0.5	73.4 ± 0.6	80.3 ± 0.4
	盛大的	85.4 ± 0.4	75.4 ± 0.4	82.7 ± 0.6
	快速GCN	81.4 ± 0.5	68.8 ± 0.9	77.6 ± 0.5
可扩展的GNNs	GraphSAINT	81.3 ± 0.4	70.5 ± 0.4	78.2 ± 0.8
	SGC	81.0 ± 0.1	71.8 ± 0.1	79.0 ± 0.1
	英镑	83.9 ± 0.7	72.9 ± 0.5	80.6 ± 0.4
	PPRGo	82.4 ± 0.2	71.3 ± 0.3	80.0 ± 0.4
我们的方法	GRAND+ (P)	**85.8 ± 0.4**	**75.6 ± 0.4**	84.5 ± 1.1
	Grand+ (a)	85.5 ± 0.4	75.5 ± 0.4	**85.0 ± 0.6**
	Grand+ (s)	85.0 ± 0.5	74.4 ± 0.5	84.2 ± 0.6

表3：大型图形的准确率（%）和运行时间（s）。

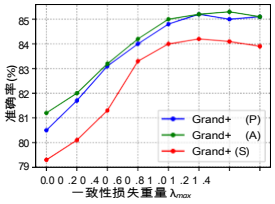
方法	AMiner-CS Acc	术语	Reddit Acc	报道	亚马逊2.5亿美元 Acc	访谈	MAG。帐户	术语
盛大的	53.1±1.1	750	OOM	-	OOM	-	OOM	-
快速GCN	48.9±1.6	69	89.6±0.6	158	72.9±1.0	239	64.3±5.6	4220
GraphSAINT	51.8±1.3	39	92.1±0.5	39	75.9±1.3	189	75.0±1.7	6009
SGC	50.2±1.2	9	92.5±0.2	31	74.9±0.5	69	-	>24h
英镑	52.7±1.7	21	88.7±1.1	370	70.1±0.9	280	-	>24h
PPRGo	51.2±1.4	11	91.3±0.2	233	67.6±0.5	160	72.9±1.1	434
GRAND+ (P)	53.9±1.8	17	93.3±0.2	183	75.6±0.7	188	77.6±1.2	653
GRAND+ (A)	54.2±1.7	14	**93.5±0.2**	174	75.9±0.7	136	**80.0±1.1**	737
GRAND+ (S)	**54.2±1.6**	10	92.8±0.2	62	**76.2±0.6**	80	77.8±0.9	483

的试验。结果显示在表2中。我们可以看到，最好的GRAND+变体在三个数据集上的表现一直优于所有的基线。值得注意的是，GRAND+（A）在Pubmed上比GRAND提高了2.3%（绝对差异）。在Cora（85.8±0.4 vs. 85.4±0.4）和Citeseer（75.6±0.4 vs. 75.4±0.4）上，GRAND+（P）比GRAND的改进也具有统计学意义（经t检验，p值为0.01）。这些结果表明，GRAND+取得了强大的*泛化性能*。

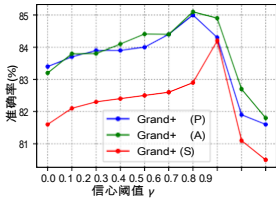
4.3 大图上的结果

为了证明GRAND+的可扩展性，我们进一步将其与四个大图上的五个可扩展的GNN基线进行比较，即AMiner- CS、Reddit、Amazon2M和MAG-Scholar-C。请注意，MAG-Scholar-C的特征维度是巨大的（即每个节点2.8M的特征）。为了使GRAND+能够处理这个问题，在随机传播之前加入了一个可学习的线性层，将高维节点特征转化为低维的隐藏向量（参见公式8）。为了进行公平的比较，我们对所有方法进行了仔细的超参数选择（参见附录A.1）。我们对每个模型进行了10次随机拆分，并报告了它在试验中的平均精度和平均运行时间（包括预处理时间、训练时间和推理时间）。结果总结在表3中。

我们从两个角度来解释表3的结果。首先，结合表2的结果，我们注意到GRAND+的三个变体在这些数据集中表现出很大的差异：



(a) 准确度为：□□□□。



(b) 准确度（相对于）□

图2: 的影响 和□在Pubmed上。

□□
□

在Cora和Citeseer上，GRAND+（P）比GRAND+（A）和GRAND+（S）取得了更好的结果；在Pubmed、Reddit和MAG- Scholar-C上，GRAND+（A）超过了其他两个变体；在AMiner-CS和Amazon2M上，GRAND+（S）得到了最佳的分类结果。这表明传播矩阵在这个任务中起着关键作用，并进一步表明GRAND+可以通过调整广义混序矩阵Π来*灵活地*处理不同的图。

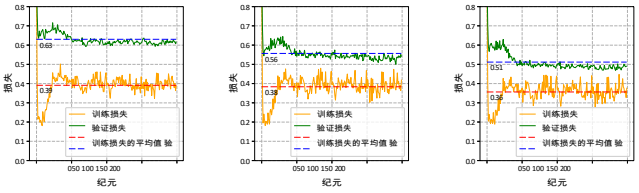
其次，我们观察到GRAND+在准确率上一直超过所有的基线方法，并在四个数据集上获得有效的运行时间。重要的是，在最大的图MAG-Scholar-C上，GRAND+可以在10分钟左右成功地完成训练和预测，而SGC和GBP需要24小时以上才能完成，因为这两种方法在预处理步骤中被设计为直接支持高维的原始特征。与FastGCN和GraphSAINT相比，GRAND+（S）分别实现了8倍和12倍的加速。与过去该数据集上最快的模型PPRGo相比，GRAND+ (S)在运行时间相当的情况下，准确率提高了4.9%。这些结果表明GRAND+在大图上的*扩展性很好*，并进一步强调了其*卓越的性能*。

我们还报告了GRAND在AMiner-CS上的准确性和运行时间。请注意，由于内存不足的错误，它不能在其他三个大数据集上执行。我们可以看到，GRAND+在AMiner-CS上的运行时间比GRAND快40倍以上，证明了所提出的近似技术在*提高效率*方面的有效性。

4.4 概括性的改进

在本节中，我们定量研究了所提出的置信度感知的一致性损失□□□□对模型的生成能力的好处。在GRAND+中，□□□□主要由两个超参数主导：*置信度阈值*γ（参照公式12）和*最大一致性损失权重*□□$_a$（参照公式13）。

我们首先分析了γ和□□$_a$□对GRAND+的分类性能的影响。具体来说，我们调整了γ和□的值。λ_{max}，其他超参数固定，观察GRAND+在测试集上的准确性变化。图2说明了Pubmed数据集的结果。从图2（a）可以看出，当λ_{max}从0增加到0.8时，准确性明显提高。当□□$_{ax}$大于0.8时，精度趋于稳定。这表明，一致性损失可以真正促进GRAND+的性能。从图2（b）中，我们观察到，当γ为0时，模型的性能从γ的扩大中获益。



(a) □□□□ = 0.0　(b) □□□□ = 1.0, □ = 0.0　(c) □□□□ = 1.0, □ = 0.6

图3：Pubmed上的训练和验证损失。

小于0.7，这突出了信心的重要性。

机制。如果□设置得过大（即>0.7），性能就会下降，因为在这种情况下，太多的未标记样本被忽略，削弱了一致性正则化的效果。

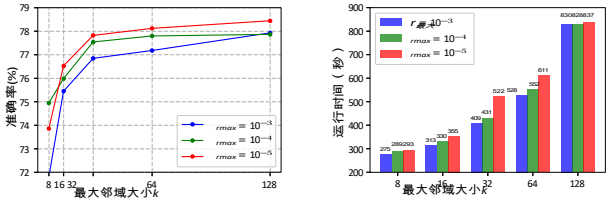
图2显示了置信度感知的一致性损失可以显著提高模型的性能。我们通过分析训练集和验证集的交叉熵损失，进一步研究它对泛化能力的好处。这里我们用*泛化差距*[16]*来*衡量模型的泛化能力，即*训练损失和验证损失之间的差距*。一个较小的泛化

化差距意味着该模型具有更好的泛化能力。图3报告了GRAND+（A）在Pubmed上的训练和验证损失。我们可以看到，当我们在训练过程中不使用一致性损失（λ_{max}=0）时，泛化差距相当大，表明存在严重的过拟合问题。而当我们把□□$_a$□改为1.0时，差距就会变小。当我们把□□$_a$□和γ设置为适当的值（即λ_m□□=1.0，γ=0.6），泛化差距进一步减小。这些观察结果表明，建议的一致性训练和信心机制确实有助于GRAND+的泛化能力。

4.5 参数分析

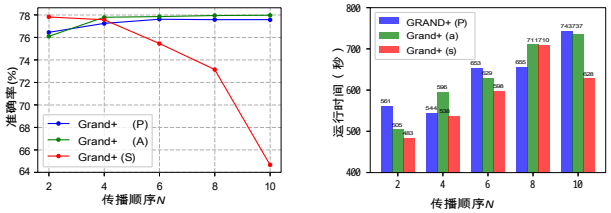
阈值r_{ma}和邻域尺寸。GRAND+使用GF- Push和top-□稀疏化来逼近Π的多个行向量，以进行小批量的随机传播（参见第3.4节）。这个过程的近似误差主要受两个超参数的影响--GFPush的阈值□□$_a$和稀疏化的最大邻域尺寸k。我们进行了详细的实验，以更好地了解k和$_m$□□对模型的准确性和运行时间的影响。图4显示了GRAND+（S）在不同的k和□□$_{ax}$对MAG-Scholar-S的相应结果。C.我们可以看到，当$_m$□□变小时，精度和运行时间都会增加，这与定理1的结论相吻合。而k则有相反的效果--随着□的增加，精度和运行时间都会增加。有趣的是，当k从128下降到32时，运行时间减少了一半，而准确度只下降了2%。这证明了顶层稀疏化策略的有效性，它可以在几乎不牺牲精度的情况下实现大幅加速。

传播顺序 □。我们研究了在使用不同传播矩阵时，传播顺序□对GRAND+的影响。图5显示了三种GRAND+变体在MAG-Scholar-C上的分类性能和运行时间，其中□的数值不同。我们可以看到，当□=2时，GRAND+ (S)取得了更好的成绩。



(a) 分类精度。　(b) 运行时间。

图4：GRAND+与□和□□□□在MAG-Scholar-C上。



(a) 分类精度。　(b) 运行时间。

图5：传播顺序□对MAG-Scholar-C的影响。

比GRAND+(P)和GRAND+(P)的精度更高，运行时间更快。(A).然而，随着□的增加，由于过度平滑问题，GRAND+（S）的精度急剧下降，而GRAND+(P)和GRAND+(A)不存在这个问题，并受益于较大的传播顺序。另一方面，增加N会增加模型的运行时间。在实际应用中，我们可以灵活地调整传播矩阵和N的值，以取得理想的效率和效果。

5 结论

我们提出了GRAND+，一个可扩展和高性能的GNN框架，用于基于图的半监督学习。GRAND+的优势包括可扩展性和泛化能力，而现有的最先进的解决方案通常只具有这两者中的一个。为此，我们在实现泛化性能时遵循了GRAND的一致性调节原则，同时对其进行了大幅扩展，以实现可扩展性，并保留甚至超越了GRAND的灵活性和泛化能力。为了实现这些，GRAND+利用广义混序矩阵进行特征传播，并使用我们的计算方法广义前推（GFPush）对其进行有效计算。此外，GRAND+还采用了一种新的置信度感知的一致性损失来实现更好的一致性训练。大量的实验表明，GRAND+不仅在基准数据集上获得了最好的表现，而且在具有数百万节点的数据集上，也取得了比现有可扩展GNN更高的性能和效率。在未来，我们希望探索更精确的近似方法来加速GNNs。

鸣谢

该工作得到了国家自然科学基金委杰出青年学者（61825602）和清华-博世联合ML中心的支持。

参考文献

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Hrayr Harutyunyan, Nazanin Alipourfard, Kristina Lerman, Greg Ver Steeg, and Aram Galstyan.2019.Mix-hop：通过稀疏化邻域 混合的高阶图卷积架构。*icml'19*（2019）。

[2] Reid Andersen, Fan Chung, and Kevin Lang.2006.使用 pagerank向量进行局部图分区。In *FOCS'06*.IEEE, 475-486.

[3] Siddhartha Banerjee和Peter Lofgren。2015.马尔科夫模型中的快速双向概率 估计。*NeurIPS*（2015）。

[4] David Berthelot, Nicholas Carlini, J. Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel.2019.MixMatch：一个半监督的整体方法 学习。*NeurIPS*（2019），5050-5060。

[5] Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann.2020.用近似的pagerank来扩展图神经网络。In *KDD'20*.2464- 2473.

[6] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien.2006.*半监督学习*.麻省理工学院出版社。https://doi.org/10.7551/mitpress/9780262033589.001。0001

[7] 陈杰，马腾飞，和曹晓。2018.FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling.*ICLR*（2018）。

[8] 陈明，魏哲伟，丁波林，李亚良，袁烨，杜晓勇，温继荣。2020.通过双向传播的可扩展图神经网络.*NeurIPS* (2020).

[9] 陈明，魏哲伟，黄增丰，丁波林，和李亚良。2020.简单和深入的图卷积网络。在*ICML中*。pmlr, 1725-1735.

[10] 蒋伟林，刘玄清，思思，李阳，Samy Bengio，和谢祖瑞。2019.Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks.In *KDD'19*.

[11] Ming Ding, Jie Tang, and Jie Zhang.2018.图上的半监督学习 与生成式对抗网。*cikm'18* (2018).

[12] 冯文正，张杰，董玉晓，韩瑜，栾焕波，徐倩，杨强，叶夫根尼-哈拉莫夫，和唐杰。2020.图随机神经网络 ，用于图上的半监督学习。*NeurIPS* (2020).

[13] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He.2017.准确的、大型的minibatch sgd：在1小时内训练imagenet。*arXiv预印本arXiv:1706.02677*（2017）。

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec.2017.归纳表征 大图上的学习。*NeurIPS*（2017），1025-1035。

[15] Sergey Ioffe和Christian Szegedy。2015.批量归一化：通过减少内部协变量偏移来加速深度 网络训练。*ICML*（2015）。

[16] Shirish Nitish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Tak Peter Ping Tang.2017.On Large-Batch Training for Deep Learning：Generalization Gap and Sharp Minima.*ICLR* (2017).

[17] Sohn Kihyuk, Berthelot David, Li Chun-Liang, Zhang Zizhao, Carlini Nicholas, Ekin Cubuk D., Kurakin Alex, Zhang Han, and Raffel Colin.2020.FixMatch：简化具有一致性和信心的半监督学习。*NeurIPS* (2020).

[18] P.Diederik Kingma 和 Lei Jimmy Ba.2015.Adam: A Method for Stochastic Optimization.*ICLR*（2015）。

[19] N.Thomas Kipf和Max Welling。2017.用 图卷积网络进行半监督分类。*ICLR*（2017）。

[20] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann.2019.Predict then Propagate：图神经网络满足个性化的PageRank。*ICLR* (2019).

[21] Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann.2019.Diffusion改善了图的学习。*arXiv预印本arXiv:1911.05485*（2019）。

[22] Pan Li, Eli Chien, and Olgica Milenkovic.2019.优化用于种子扩展社区检测的广义寻根方法。*arXiv预印本 arXiv:1905.10881*（2019）。

[23] Qimai Li, Zhichao Han, and Xiao-Ming Wu.2018.深入了解图 卷积网络的半监督学习。在*AAAI'18*。

[24] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio.2013.论 训练递归神经网络的难度。*icml* (2013), 1310-1318.

[25] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang.2018.网络嵌入作为矩阵分解：统一deepwalk、line、pte和 node2vec。In *WSDM'18*.459-467.

[26] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Paul Bo-Jun Hsu, and Kuansan Wang.2015.微软学术服务（MAS）概述 和应用。*WWW（配套卷）*（2015），243-246。

[27] 唐杰，张静，姚利民，李娟子，张丽，和苏中。2008.Arnet- miner：学术社会网络的提取和挖掘。在*KDD'08*中。

[28] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio.2018.图注意网络。*ICLR*（2018）。

[29] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger.2019.Simplifying graph convolutional networks.*ICML*（2019）、6861-6871.

[30] 杨志林，William W Cohen，和Ruslan Salakhutdinov。2016.重新审视 与图嵌入的半监督学习。*ICML*（2016）。

[31] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna.2020.GraphSAINT: 基于图形抽样的归纳学习方法.*ICLR* (2020).

[32] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf.2004.具有局部和全局一致性的学习。*NeurIPS*（2004），321-328。

[33] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty.2003.使用高斯场和谐波函数的半监督性 学习.*ICML* (2003).

[34] Difan Zou, Ziniu Hu, Yewen Wang, Song Jiang, Yizhou Sun, and Quanquan Gu.2019.用于训练深度和大图卷积网络的层依赖重要性采样。*NeurIPS*（2019）。

A 附录

A.1 实施说明

*A.1.1 运行环境。*实验在一台Linux服务器上进行，该服务器配备英特尔（R）至强（R）CPU Gold 6420 @ 2.60GHz，376G内存和10个NVIDIA GeForce RTX 3090TI-24GB。Python的版本是3.8.5。

*A.1.2 实施细节。*我们用C++实现GFPush，并使用OpenMP来执行并行化。我们使用Pytorch来实现GRAND+的训练过程，并使用pybind[5]来创建近似模块的Python绑定。在GRAND+和其他基线中，我们使用BatchNorm[15]和梯度剪裁[24]来稳定模型的训练，并采用Adam[18]进行优化。

*A.1.3 数据集详情。*本论文共使用了7个数据集，即Cora、Citeseer、Pubmed、AMiner-CS、Reddit、Amazon2M和MAG-Scholar-C。我们对Cora、Citeseer和Pubmed的预处理脚本是参照Pane-toid[6]的代码实现的。按照[19, 28, 30]中使用的实验设置，我们在第4.2节中报告的Cora、Citeseer和Pubmed的结果中使用随机种子进行了100次试验。AMiner-CS是由Feng等人[12]基于AMiner引文网络[27]构建的。在AMiner-CS中，每个节点代表一篇论文，边是引文关系，标签是论文的研究主题。Reddit是由Hamilton等人[14]发表的，其中每个节点代表Reddit社区中的一个帖子，一个图链代表两个帖子被同一个用户评论过。任务是预测每个帖子的类别。Amazon2M是由Chiang等人发表的[10]，其中每个节点是一个产品，每个边表示两个产品一起购买，标签代表产品的类别。MAG-Scholar-C由Bojchevski[5]基于微软学术图谱（MAG）[26]构建，其中节点指的是论文，边表示论文之间的引用关系，特征是论文摘要的词包。

对于AMiner-CS、Reddit、Amazon2M和MAG-Scholar-C，我们用20×#类用于训练，30×#类节点用于验证，其余节点用于测试。对于Aminer、Reddit和MAG-Scholar-C，我们对每个类随机抽取相同数量的节点，训练时每类20个节点，验证时每类30个节点。对于Amazon2M，我们从整个数据集中统一抽取所有的训练和验证节点，因为有些类的节点数少于20。对于这些数据集，我们报告了随机分割的10个追踪的平均结果。

*A.1.4 超参数选择。*对于表2的结果，我们在验证集上调整了GRAND+的超参数，并使用最佳配置进行预测，其他基线方法的结果取自以前的工作[9, 12, 28]。对于表3-5的结果，我们对GRAND+和其他GNN基线（即FastGCN、GraphSAINT、SGC、GBP和PPRGo）都进行了详细的超参数搜索。对于每一次搜索，我们用随机种子进行3次实验，并选择在验证集上获得最佳平均精度的超参数配置。然后，我们用选定的配置训练模型。

[5] https://github.com/pybind/pybind11

[6] https://github.com/kimiyoung/planetoid

表4：GRAND+的超参数配置。

		$\bar{A}$	wr	L	$d_{mh\,rm}ax$		k	N	λ	$m\,a\,x$
Cora	GRAND+ (P)	10^{-2}	10^{-3}	2	64	10^{-7}	32	20		1.5
	GRAND+ (A)	10^{-2}	10^{-3}	2	64	10^{-7}	32	4		1.5
	GRAND+ (S)	10^{-2}	10^{-3}	2	64	10^{-7}	32	2		1.5
馨香坊	盛大+ (P)	10^{-3}	10^{-3}	2	256	10^{-7}	32	10	0	.8
	盛大+ (a)	10^{-3}	10^{-3}	2	256	10^{-7}	32	2		0.8
	grand+ (s)	10^{-3}	10^{-3}	2	256	10^{-7}	32	2		0.8
公共医学	盛大+ (P)	10^{-2}	10^{-2}	1	-	10^{-5}	16	6		1.0
	大+ (a)	10^{-2}	10^{-2}	1	-	10^{-5}	16	4		1.0
	盛大+ (s)	10^{-2}	10^{-2}	1	-	10^{-5}	16	2		1.0
AMiner-CS	盛大+ (P)	10^{-2}	10^{-2}	1	-	10^{-5}	64	6		1.5
	盛大+ (a)	10^{-2}	10^{-2}	1	-	10^{-5}	64	4		1.5
	盛大+ (s)	10^{-2}	10^{-2}	1	-	10^{-5}	64	2		1.5
瑞德	grand+ (p)	10^{-4}	0	2	512	10^{-5}	64	6		1.5
	grand+ (a)	10^{-4}	0	2	512	10^{-5}	64	6		1.5
	盛大+ (s)	10^{-4}	0	2	512	10^{-7}	64	2		1.5
Amazon2M	大+ (p)	10^{-3}	10^{-5}	2	1024	10^{-6}	64	6		0.8
	盛大+ (a)	10^{-3}	10^{-5}	2	1024	10^{-6}	64	4		0.8
	大+ (s)	10^{-3}	10^{-5}	2	1024	10^{-6}	32	2		0.8
MAG-Scholar-C	Grand+ (p)	10^{-2}	0	2	32	10^{-5}	32	10	1	.0
	GRAND+ (A)	10^{-2}	0	2	32	10^{-5}	32	10	1	.0
	盛大+ (s)	10^{-2}	0	2	32	10^{-5}	32	2		1.0

GRAND+的超参数选择包括两个阶段：我们首先对神经网络的基本超参数进行搜索。具体来说，我们搜索学习率□□，从{10^{-2}, 10^{-3}, 10^{-4}}，重量衰减率wr来自{0, 10^{-5}, 10^{-3}, 10^{-2}}、隐层的数量$_{Lm}$来自{1,2}，隐层的尺寸$_{dh}$来自{32, 64, 128, 256, 512, 1024}。

在第二阶段，我们将这些基本超参数固定为最佳配置，并搜索以下具体的超参数：DropNode rate □, augmentation times per batch □, threshold $_{rmax}$, max neighborhood size □, propagation order N、信度阈值γ，最大一致性损失权重$_{\lambda max}$，未标记子集的大小|□'|和一致性损失函数D。为了降低搜索成本，我们将一些超参数固定下来。具体来说，我们在所有的数据集上固定δ=0.5，M=2和γ=2。对于Cora、Pubmed和Citeseer，我们设定|□'|= |□|，而对于其他数据集，我们设定|□'|= 10000。我们还提供了一个分析，以了解以下因素的影响
|U'|,见附录A.3。我们采用KL发散作为一致性。
对AMiner-CS、Reddit和Amazon2M使用损失函数，而对其他数据集使用□$_2$ 距离。这是因为□$_2$ 距离在处理有大量类的数据集时容易出现梯度消失的问题。然后，我们对□□□□、□、□和□□□□进行超参数选择。具体而言，我们从以下方面搜索□□□□{10^{-5}, 10^{-6}, 10^{-7}}，k来自{16, 32, 64, 128}，N来自{2, 4, 6, 8, 10, 20}。和□□□□{0.5，0.8，1.0，1.2，1.5}。表4列出了GRAND+的最佳超参数配置。

A.2 定理证明

为了证明定理1，我们首先介绍以下定理：

判断方法 1. *对于任何储备向量q(□)，残留向量r(□)和随机行走过渡向量* $P_s^{(n)} = (D^{-1}A)^n$ $(0 \le n \le N)$*，我们有：*

$$P_s^{(n)} = q^{(n)} + \sum_{i=1}^{n} (P^i)^T \cdot r^{(n-i)}$$

证明。我们通过归纳法来证明该定理。为简洁起见，我们使用 RHS(□)，表示方程15的右侧。在算法1中、

$q^{(n)}$和$r^{(n)}$在$1 \le n \le N$的情况下初始化为0→，$r^{(0)}$和$q^{(0)}$初始化为$e^{(s)}$。因此，方程15在算法开始时成立，基于以下事实：

$$RHS^{(0)} = e^{(s)} = P_s^{(0)}$$

$$RHS^{(n)} = (P)^{nT} \cdot e^{(s)} = P_s^{n},\ 1 \le n \le N。$$

那么我们假设方程15在n'-次迭代开始时成立，我们将证明该方程在节点□的推送操作后仍然正确。我们有三种情况，即不同的n值：

1) 当$n < n'$，推操作不会改变 $q^{(n)}$ 和 $r^{(n-i)}$，$1 \le i \le n$。因此，对于$n < n'$，方程15成立。

2) 当$n = n'$，推操作将$r^{(n-1)}$递减为$r^{(n-1)} \cdot e^{(v)}$和增量 $q^{(n)}$ 由 $\sum_{u \in N_v} r_v^{(n-1)}/d_v \cdot e^{(u)}$。因此，我们有

$$RHS^{(n)} = P^n + \sum_{u \in N_v} r^{(n-1)}/d_v \cdot e^{(u)} - r^{(n-1)} \cdot P_v = P^n + 0^{\rightarrow} = P^n。$$

因此，对于$n = n' + 1$，方程 15 成立。

3) 当$n > n'$，推操作将减少$r^{(n')}$，即$r^{(n')} \cdot e^{(v)}$并增加 $r^{(n'+1)}$ 由 $\sum_{u \in N_v} r^{(n')}/d_v$。因此，我们有

$$RHS^{(n)} = P^n + (P^{n-n'-1})^T \sum_{u \in N_v} r^{(n')}/d_v \cdot e^{(u)} - (P^{n-n'})^T (r^{(n')} \cdot e^{(v)})$$

$$= P^n + (P^{n-n'-1})^T \left(\sum_{u \in N_v} r^{(n')}/d_v \cdot e^{(u)} - (r^{(n')} \cdot P_v) \right)$$

$$= p^n + (p^{n-n'-1})^T \cdot 0^{\rightarrow} = p^n。$$

因此，对于$n > n' + 1$，方程15成立。

因此，归纳法是成立的，而本结论也得到了证明。 □

那么，我们可以按以下方式证明定理1。定理1. *算法1具有*$O(N/r_{max})$*的时间复杂度和*$O(N/r_{max})$*的内存复杂度，并返回*Π_s*，作为*Π_s*的近似值，其误差界限为：*$\|\tilde{\Pi}_s - \Pi_s\|_1 \le N \cdot (2|E| + |V|) \cdot r_{max}$。

证明。让 V_n 为步骤□中要推送的节点集合。当对$v \in V_n$ 进行推送操作，$r(\square\text{-}1)$的值将 $\|r^{(n-1)}\|_1$将至少减少$r_{max} \cdot d_v$。由于$\|r^{(n-1)}\|_1 \le 1$，我们必须有 $\sum_{v \in V_n} d_v \cdot r_{max} \le 1$，因

$$\sum_{v \in V_n} d_v \le 1/r_{max}。 \tag{16}$$

时间复杂度。对于步骤□中对□的推操作，我们需要对$r^{(n)}$执行d_v次的上推操作。所以推送操作的总时间在步骤□中被约束为 $\sum_{v \in V_n} d_v$。因此，根据公式16，每一步的时间复杂度以$O(1/r_{max})$为界，算法1的总时间复杂度以 O（N/r_{max}）为界。

内存复杂度。当第□个step迭代完成后，在第□个step迭代中的计数$r^{(n)}$的非零元素不多于 $\sum_{v \in V_n} d_v$，因为推操作在□上只对$r^{(n)}$进行了d_v次更新。因此，非零的计数的元素也小于$q^{(n)}$的 $\sum_{v \in V_n} d_v$。根据公式16，我们可以得出结论：r_n最多有N/r_{max}非零元素。在实现在算法中，所有的向量都被存储为稀疏结构。因此，算法1有记忆复杂性为O（N/r_{max}）。

误差的约束。根据定理1，我们可以得出以下等式一。训练：

$$\left\| P_s^{(n)} - q^{(n)} \right\|_1 = \left\| \sum_{i=1}^{n} (P^i)^T \cdot r^{(n-i)} \right\|_1$$

$$= \sum_{i=1}^{n} \left\| r^{(n-i)} \right\|_1 \tag{17}$$

$$\le \sum_{i=1}^{n} \left\| r^{(n-i)} \right\|_1$$

算法终止后，我们有 $0 \le r^{(n)} \le d_v \cdot r_{max}$，对于所有$v \in V$。因此、

$$\left\| R^{(n)} \right\|_1 = \sum_{v \in V} r_v^{(n)} \le \sum_{v \in V} d_v \cdot r_{max} = (2|E| + |V|) \cdot r_{max}. \tag{18}$$

根据公式17，我们可以得出结论：$\|P^{(n)} - q^{(n)}\|_1 \le n \cdot (2|E| + |V|) \cdot r_{max}$。更进一步，我们有：

$$\left\| \tilde{\Pi}_s - \Pi_s \right\|_1 \le \sum_{n=0}^{N} w_n \left\| P_s^{(n)} - q^{(n)} \right\|_1 \le \sum_{n=0}^{N} w_n \cdot n \cdot (2|E| + |V|) \cdot r_{max} \tag{19}$$

至于$0 \le w_n$ 和 $\sum_{n=0}^{N} w_n = 1$，因此：

$$\sum_{n=0}^{N} w_n \cdot n \cdot (2|E| + |V|) \cdot r_{max} \le N \cdot (2|E| + |V|) \cdot r_{max}, \tag{20}$$

这表明$\|\tilde{\Pi}_s - \Pi_s\|_1 \le N \cdot (2|E| + |V|) \cdot r_{max}$. □

A.3 额外的实验

表5：无标签子集大小的影响（$|U'|$）。

|U'|	Aminer 累积	Aminer RT (s)	Aminer AT (ms)	Reddit 累积 (%)	Reddit RT (s)	Reddit AT (ms)	Amazon2M Acc (%)	Amazon2M RT (s)	Amazon2M AT (ms)
0	51.1 ± 1.4	10	149	92.3 ± 0.2	53	717	75.0 ± 0.7	63	2356
103	53.6 ± 1.6	9	153	92.6 ± 1.2	58	882	75.2 ± 0.5	62	2630
104	54.2 ± 1.6	10	250	92.8 ± 0.2	62	2407	76.1 ± 0.6	69	3649
105	54.4 ± 1.2	13	1121	92.9 ± 0.2	78	17670	76.3 ± 0.7	86	14250

对U'的大小进行分析。在GRAND+中，一个无标签的子集结点U'是从U中取样的，用于一致性正则化。对为此，我们需要预先计算稀疏化的近似值$\tilde{\Pi}_v$的行向量Π_v为每个节点$v \in U'$。在此，我们根据经验分析了U'的大小如何影响分类精度（Acc），运行GRAND+的计算时间（RT）和近似时间（AT）。表5介绍了当我们改变GRAND+(S)的结果时$|U'|$从0到10^5在AMiner-CS、Reddit和Amazon2M上。我们有两个ob-服务：首先，当$|U'|$从0（意味着一致性损失退化为0）变为10^3时，分类性能得到明显改善，而运行时间变化不大。表明一致性正则化是一种经济的方法，可以提高GRAND+在半------的泛化性能。监督的设置。第二，当$|U'|$超过10^4时，增加率的准确度会减慢，而运行时间和近似计算时间增加得更快。这一观察结果表明，对无标签节点的抽样程序对保证模型的效率，这也使我们能够明确地控制通过以下方式在GRAND+的有效性和效率之间进行权衡采样规模$|U'|$。