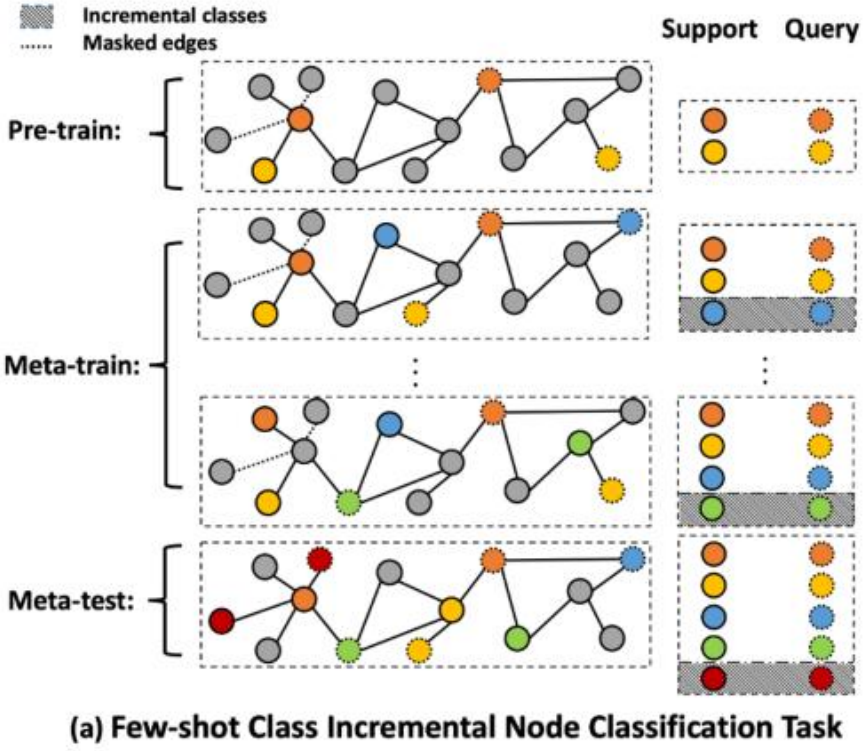
Graph Few-shot Class Incremental Learning

王嘉禾

问题背景

- Graph中每个结点有feature和label，通过自身feature和参考图中拓扑结构后选择的其他节点的feature进行分类
- Few-shot Class Incremental Learning：在graph中，是指在每个学习周期中会出现新的label，且样本数量较少（旧分类是否可参考的问题在graph上意义不大）
- 目标：提高分类精确度/记住旧模型在旧class上学习到的知识/防止在少样本上出现overfitting



(a) Few-shot Class Incremental Node Classification Task

论文调研：
Graph Few-shot Class-incremental Learning

- 问题设定：每个session的support set和query set均包含到目前为止所有出现过的分类，support set中关于novel class的部分为N way K shot
- Meta-learning，找到一组参数使得其在新的Few-shot Class上训练更具效率，直接套了MAML

Meta-training: To learn an initialization with more transferable meta-knowledge within the graph, here, we propose the **Graph Pseudo Incremental Learning (GPIL)** paradigm, where a model

训练过程

- 提前将数据集划分为base集和novel集(class不相交)，再分别分为train集和test集(base上两个集合随机分配，class可以重叠，novel集按照class划分，train集和test集不相交)
- 先通过base(train)上的监督学习初始化模型。
- 接下来的每个session，从novel(train)中找出N个novel class，每个class K个结点(N way K shot)，和上个session的support集并成新的support集，query集为到当前session为止所有的class各取K个节点(和support中的新的N*K个节点无关)
- (support set是不断固定增长的，query set是随机sample的)
- 每个session中，都是进行一次训练，然后根据训练后的loss来更新参数(meta learning)

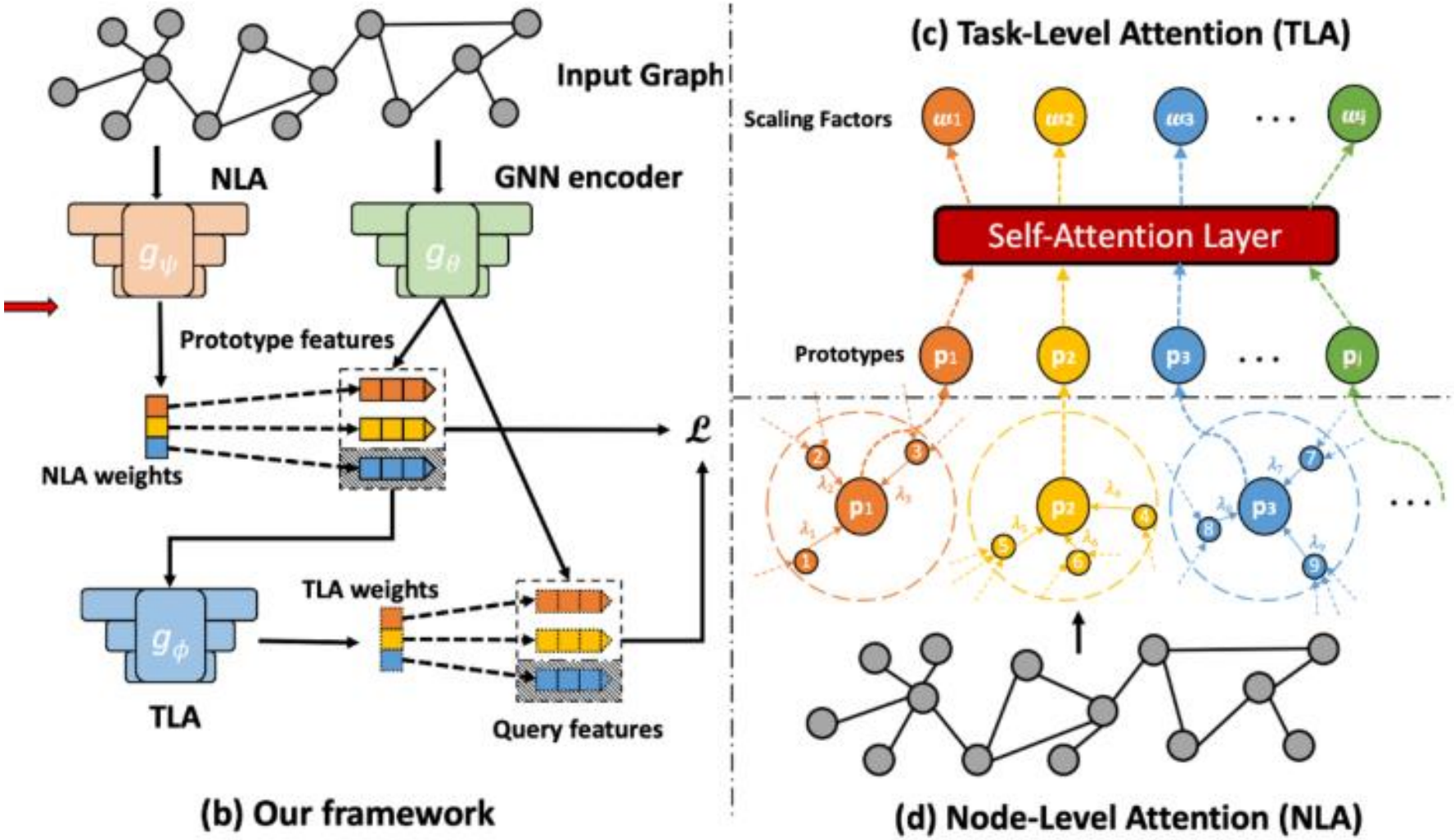
测试过程

- 从base(test)和novel(train)中提取support set
- evaluation session：完全模拟训练过程，每个session，从novel(test)中提取N way K shot和过去的support集组成新support集，query集为当前所有class中抽取，进行一次训练，计算loss，并将当前的N way K shot并入support集

模型

- 分类机制：对每个class求一个prototype representation(NLA)，对于每个node，按照其embedding和所有prototype的距离分类
- Task level attention：(通过所有的prototype)计算在每次分类任务(每个session)中对query集中每个类别分类的重要程度，算loss时进行加权平均(理解：例如正常情况下新的class可能重要程度较高，最初在base中的class已经被query了很多次了，已经不重要/或是当前task的训练容易造成遗忘，则加强对旧class的重视程度等等)
- Node level attention：(通过所有的node)计算同一个class中每个node的重要程度，计算prototype representation时采用加权平均(理解：同一个class中一些node可能非常典型，而另一些可能特征较弱，处在特征空间的边缘，在样本很少时比较重要)
- Loss：比较简单，就是对query set中节点分类的cross entropy

模型


Input Graph
NLA
GNN encoder
Prototype features
NLA weights
TLA weights
TLA
Query features
(b) Our framework
(c) Task-Level Attention (TLA)
Scaling Factors
Self-Attention Layer
Prototypes
(d) Node-Level Attention (NLA)

总结

- Meta learning体现在两个attention模型的训练，相当于用这个两个attention模型直接去训练新任务效果会较高（感觉本质上是套了MAML的壳）
- 模型算是比较典型的prototype network，两个attention比较有亮点，算是从两个角度克服few-shot的问题
- 问题设定有点模糊

论文调研：

Geometer

- 问题设定：和上一篇完全相同，每个session的support set和query set均包含到目前为止所有出现过的分类，support set中关于novel class的部分为N way K shot

训练过程

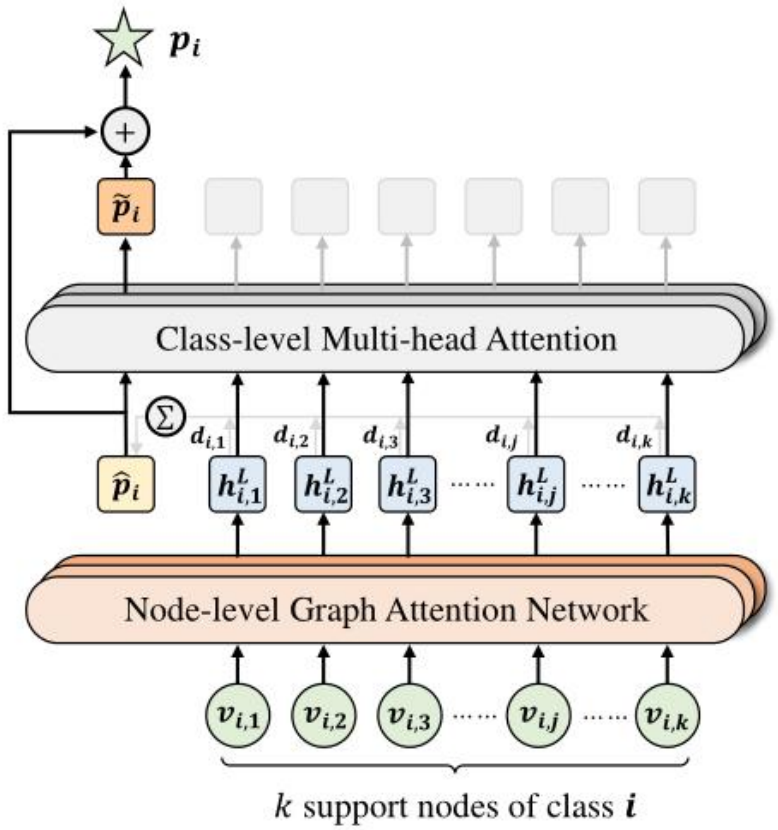
- 和上一篇类似，但是没有那么复杂的数据集划分
- 在base set中，support set和query set都为均匀分布
- 每个session，query set和support set均包含目前为止所有class，其中support set中对新的class是N way K shot(没有说清楚？)，query set倾向于多选择旧class，防止忘记旧知识

PROBLEM 1. ***Graph Few-Shot Class-Incremental Learning***

In t-th streaming session, we denote ΔC^t novel classes with K labeled nodes as the ΔC^t-way K-shot GFSCIL problem. The labeled training samples are denoted as support sets $\mathcal{S}$. Another batch of nodes to

模型

- Node level attention：目的本质上还是算出同一个class中每个node的embedding对prototype presentation的不同贡献，在样本少的情况下是比较重要的(和上一篇没什么本质区别)。
- 先在node之间作attention，通过度数加权求得一个prototype presentation，再用prototype和所有的node作attention，额外加上每个node的贡献



模型

- Intra-Class Proximity：目的在提高同一个class中node embedding的聚合程度，比较关注分布平均度，对个别距离较远的embedding反应较敏感(存疑？大多数embedding聚合程度较高？)

$$\mathcal{L}_P = \sum_{k=1}^{\|C^k\|} \frac{\alpha_k}{n_k} \sum_{i=1}^{n_k} -\log \frac{\exp\left(-d\left(f_{\mathcal{G}}\left(\boldsymbol{x}_i\right), \boldsymbol{p}_k\right)\right)}{\sum_{k' \in C^k} \exp\left(-d\left(f_{\mathcal{G}}\left(\boldsymbol{x}_i\right), \boldsymbol{p}_{k'}\right)\right)}, \quad (7)$$

模型

- Inter-Class Uniformity：不同class的prototype之间的关于向量中心分散平均程度

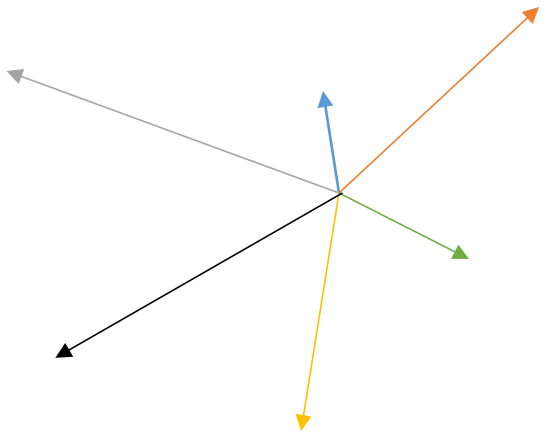
$$\mathcal{L}_U = \frac{1}{\|C^k\|} \sum_{i=1}^{\|C^k\|} \left\{ 1 + \max_{j \in \{C^k\} \setminus i} \left[\frac{(\boldsymbol{p}_i - \boldsymbol{p}_c)}{\|\boldsymbol{p}_i - \boldsymbol{p}_c\|} \cdot \frac{(\boldsymbol{p}_j - \boldsymbol{p}_c)}{\|\boldsymbol{p}_j - \boldsymbol{p}_c\|} \right] \right\}, \quad (9)$$

- nter-Class Separability：新class和老class的可区分度

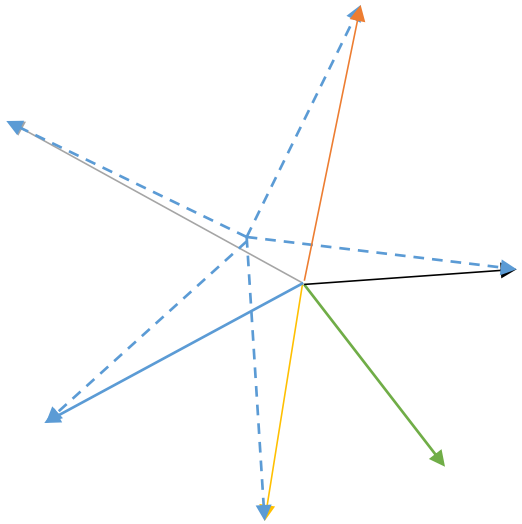
$$\mathcal{L}_S = \frac{1}{\Delta C^k} \sum_{i \in \Delta C^k} \min_{j \in C^{k-1}} \exp\left(-d\left(\boldsymbol{p}_i, \boldsymbol{p}_j\right)\right),$$

模型

ω两个loss相互制衡，因为无法控制class出现的顺序

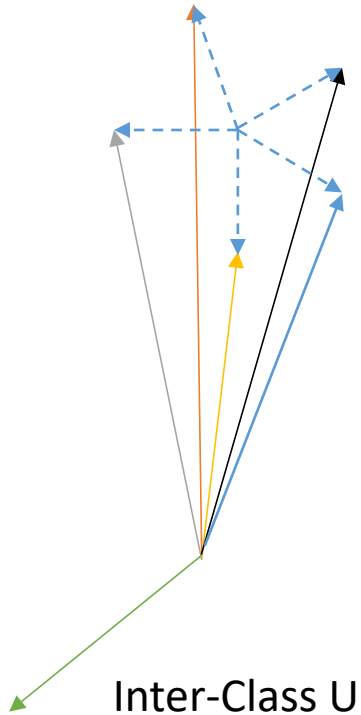


理想状况



Inter-Class Uniformity
较高的情况(1)

New class



Inter-Class Uniformity
较高的情况(2)

- Teacher-Student Knowledge Distillation：知识蒸馏用于在每个session用于保存上个session学习后获得的知识
- 在base set：

$$\mathcal{L}_{\text{train}} = \lambda_P \mathcal{L}_P + \lambda_U \mathcal{L}_U .$$

- 在后续的session：

$$\mathcal{L}_{\text{finetune}} = \lambda_P \mathcal{L}_P + \lambda_U \mathcal{L}_U + \lambda_S \mathcal{L}_S + \lambda_{KD} \mathcal{L}_{KD},$$

总结

- Meta learning：体现在哪？
- 分类的loss体现在关于prototype的loss中
- Intra-Class Proximity值得更多讨论，论文中倾向于将所有node embedding全部聚拢，但实际上的分布可能倾向于部分聚拢和个别的边缘特征？
- 关于prototype分布的Geometric Metric Learning可以更多尝试，例如label的联合分布
- Support set中已知的label怎么利用？（？）
- 问题设定方面：新的class在后续是否持续增加？