GNNExplainer论文分享

1. Overview

- 一句话概括："We formulate GNNEXPLAINER as an optimization task that maximizes the mutual information between a GNN’s prediction and distribution of possible subgraph structures."

2. 背景

2.1 graph embedding问题定义

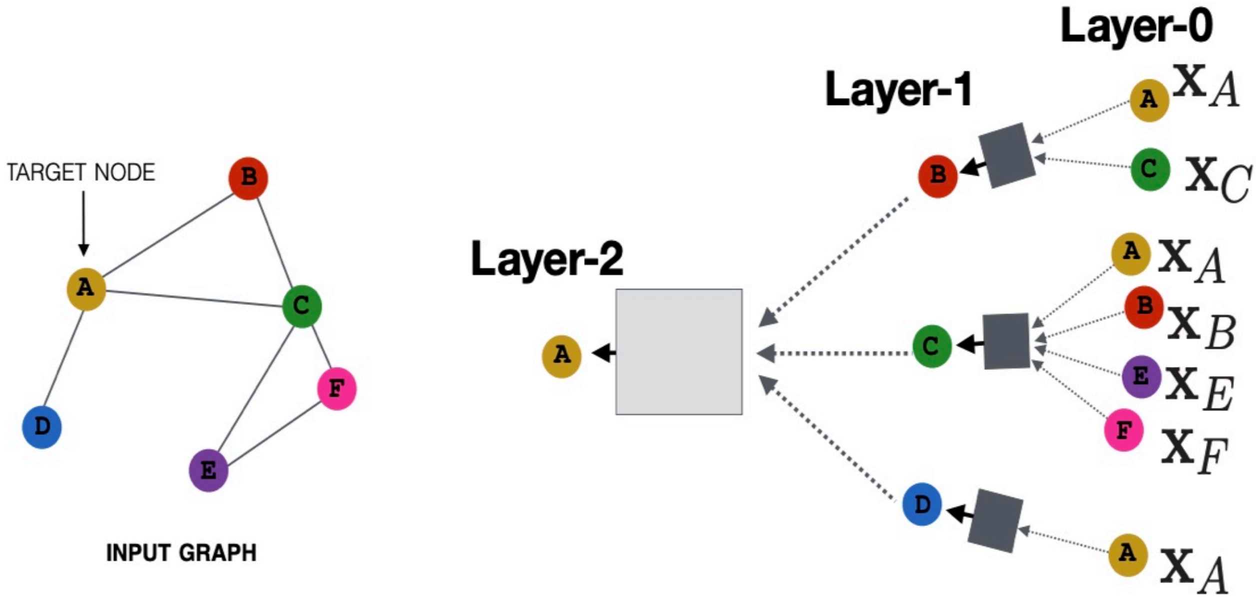
Definition 4. *(Graph embedding) Given a graph* $G = (V, E)$*, a graph embedding is a mapping* $f : v_i \rightarrow \boldsymbol{y_i} \in \mathbb{R}^d \ \forall i \in [n]$ *such that* $d \ll |V|$ *and the function* f *preserves some proximity measure defined on graph* G*.*

- 定义：求图上节点的映射，使得映射后的节点满足图结构上的”相似“
- 怎么定义相似？基于图的应用场景定义
- 以社交网络为例：
    - 节点间有边相连（first-order proximity），人以类聚物以群分
    - 节点的邻居重叠（second-order proximity），共同好友多
    - 节点的特征（node feature）相似，爱好相同
    - ……

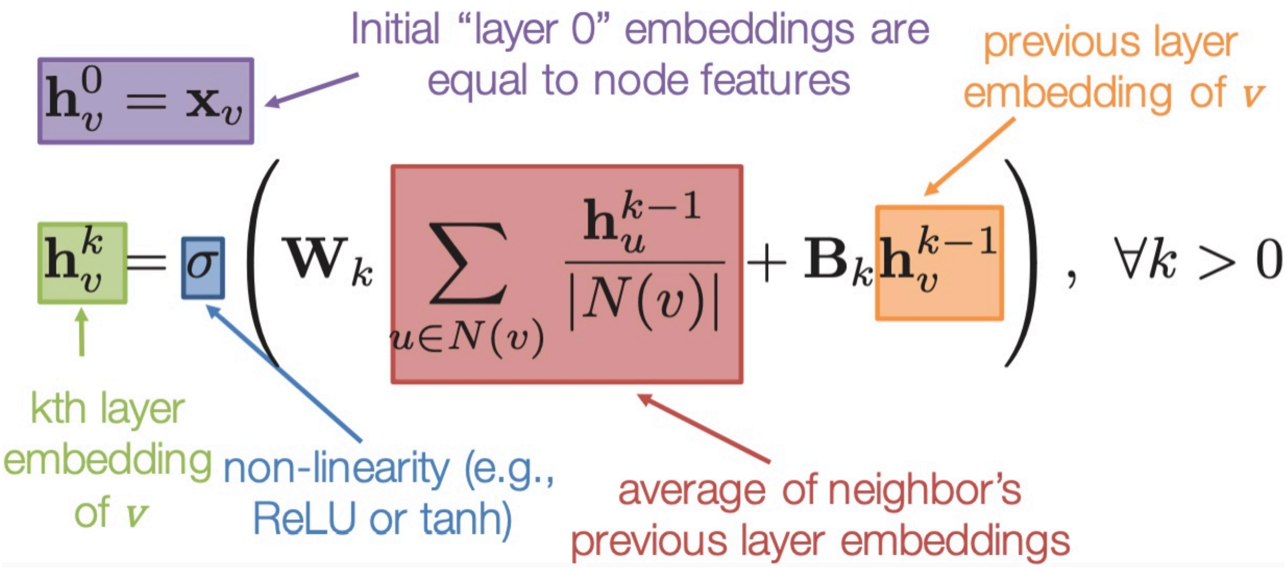
2.2 Graph Neural Network(GNN)

- 迭代地延边传播节点间的神经网络信号，巧妙融合了节点特征和图上信号传播的思想
- inductive learning：参数共享，训练好的网络参数可直接用于新增节点的预测

直观上



数学上



2.3 GNN计算流程

GCN:

$$H^{(l+1)} = \sigma(H^{(l)} W_0^{(l)} + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} H^{(l)} W_1^{(l)})$$

GraphSage:

$$h_v^K = \sigma([W_k AGG(\{h_u^{k-1}, \forall u \in N(v)\}), B_k h_v^{k-1}])$$

1. MSG: 节点对(v_i, v_j)在第l层的传递信号可以用$m_{ij}^l = MSG(h_i^{l-1}, h_j^{l-1}, r_{ij})$其中$h_i^{l-1}$表示$l-1$层的节点编码（信号），$r_{ij}$表示节点间关联。

2. AGG: 节点v_i聚合邻居传递的信号，$M_i^l = AGG(\{m_{ij}^l, \forall v_j \in N(v_i)\})$。
3. $UPDATE$: 节点v_i结合AGG和自身在上一层的编码，非线性变化得l层编码 $h_i^l = UPDATE(M_i^l, h_i^{l-1})$。

本文意在建模解释上述流程描述的任何GNN预测。

2.4 理解GNN预测的意义

- 风控场景模型结果的可解释性极为重要，提高业务对技术的信任
- 提高模型的透明度，赋能业务理解黑产行为/社区生态行为
- 提高开发者对GNN网络结构的理解，降低调试成本，便于debug

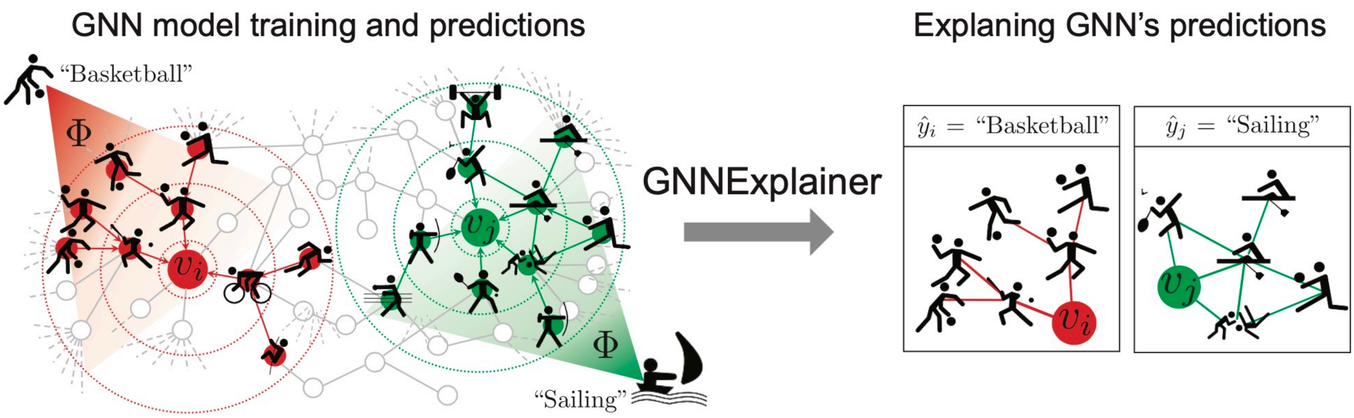


Figure 1: GNNEXPLAINER provides interpretable explanations for predictions made by any GNN model on any graph-based machine learning task. Shown is a hypothetical node classification task where a GNN model Φ is trained on a social interaction graph to predict future sport activities. Given a trained GNN Φ and a prediction $\hat{y}_i$ = "Basketball" for person v_i, GNNEXPLAINER generates an explanation by identifying a small subgraph of the input graph together with a small subset of node features (shown on the right) that are most influential for $\hat{y}_i$. Examining explanation for $\hat{y}_i$, we see that many friends in one part of v_i's social circle enjoy ball games, and so the GNN predicts that v_i will like basketball. Similarly, examining explanation for $\hat{y}_j$, we see that v_j's friends and friends of his friends enjoy water and beach sports, and so the GNN predicts $\hat{y}_j$ = "Sailing."

2.5 问题的数学表述

对给定节点v，GNN模型Φ的预测可表示为

$$\hat{y} = \Phi(G_c(v), X_c(v)).$$

其中G_c表示v的计算图，X_c表示图上的节点特征。即GNN模型Φ可表示为条件分布 $P_\Phi(Y|G_c, X_c)$。

那么解释预测$\hat{y}$的问题可以通过求解下式表述：

$$\arg \max_{G_S \subset G_c(v), X_S^F \subset X_c(v)} MI(Y, (G_S, X_S^F))$$

其中MI表示互信息，X_S表示子图G_S的节点特征集合，F为节点特征集合上的掩码。我们希望找到最主要影响预测结果$\hat{y}$的子图和对应节点特征的子集，输出作为解释预测的论据。这种影响通过互信息（mutual information）量化描述，即找到对随机变量Y分布影响最大的因子。

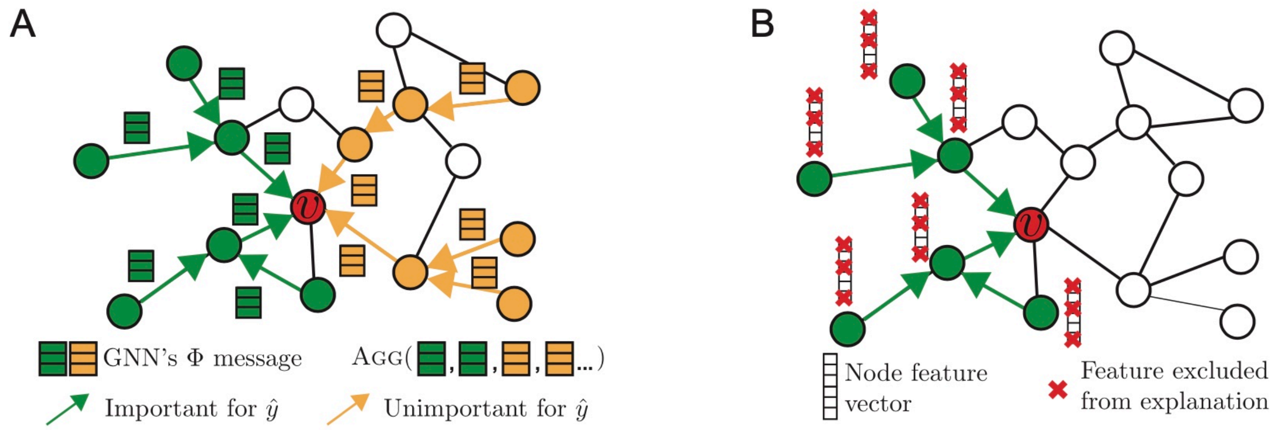


Figure 2: **A.** GNN computation graph G_c (green and orange) for making prediction $\hat{y}$ at node v. Some edges in G_c form important neural message-passing pathways (green), which allow useful node information to be propagated across G_c and aggregated at v for prediction, while other edges do not (orange). However, GNN needs to aggregate important as well as unimportant messages to form a prediction at node v, which can dilute the signal accumulated from v's neighborhood. The goal of GNNEXPLAINER is to identify a small set of important features and pathways (green) that are crucial for prediction. **B.** In addition to G_S (green), GNNEXPLAINER identifies what feature dimensions of G_S's nodes are important for prediction by learning a node feature mask.

注意到$MI(Y,(G_S,X_S^F))=H(Y)-H(Y|G=G_S,X=X_S^F)$，对给定的GNN模型$H(Y)$固定，那么原问题转化为最小化条件熵

$$H(Y|G=G_S,X=X_S^F)=-\mathbf{E}_{Y|G_S,X_S^F}[\log P(Y|G=G_S,X=X_S^F)].$$

2.6 求解

2.6.1 不筛选节点特征

即$X_S^F=X_S=X_S(G_S)$时，记G_S服从随机图分布$\mathcal{G}$，那么优化目标可表示为

$$\min_{\mathcal{G}} \mathbf{E}_{G_S\sim\mathcal{G}}[H(Y|G=G_S,X=X_S)]$$

在凸性假设下由Jensen's Inequality有上界：

$$\min_{\mathcal{G}} H(Y|G=\mathbf{E}_{G_S\sim\mathcal{G}}[G_S],X=X_S).$$

不过NN模型不满足凸性，这里作者表示实验表明加上*正则项*（参考2.6.3）后，由目标函数得到的局部极小值也很好用。

这里直接用exact inference求分布$\mathcal{G}$使目标函数最小的计算复杂度高，因G_c有指数量级的子图G_S。假设隐变量e_{ij}相互独立，采用mean-field variational family近似分布$\mathcal{G}$有

$$P_{\mathcal{G}}(G_S) = \prod_{(j,k)\in G_c} A_S[j,k] \;\; s.t. \;\; A_S \in [0,1]^{n\times n}, \;\; \forall j,k \; A_S[j,k] \leq A_c[j,k].$$

这里A_S是fractional adjacency matrix，$P(e_{jk} \in G_S) = A_S[j,k]$。因此在近似分布$P_{\mathcal{G}}$下，有$\mathbf{E}_{G_S\sim\mathcal{G}}[G_S] \approx A_c \odot \sigma(M)$，$M \in \mathbf{R}^{n\times n}$就是我们需要学习的参数矩阵（mask）。通过sigmoid函数映射到$[0,1]$上的概率。

在一些应用中，我们往往只关心为什么给节点打上一类标记（如只关心”坏人“的预测）。此时可以把优化目标改为交叉熵，用梯度下降优化：

$$\min_M -\sum_{c=1}^{C} \mathbb{I}[y=c] \log P_\Phi(Y=y|G=A_c \odot \sigma(M), X=X_c).$$

最后输出解释子图G_S可通过对M设定阈值后移除低于阈值的边实现。

2.6.2 筛选节点特征

$F \in \{0,1\}^d$表示节点特征集合上的掩码，即

$$X_S^F = \{x_j^F | v_j \in G_S\}, \quad x_j^F = [x_{j,t_1}, \ldots, x_{j,t_k}] \; for \; F_{t_i} = 1.$$

此时最大化互信息目标函数表示为：

$$MI(Y,(G_S,F)) = H(Y) - H(Y|G=G_S, X=X_S^F).$$

同上节，记$X_S^F = X_S \odot F$，用reparametrization tricks有$X = Z + (X_S - Z) \odot F$，$Z$为基于经验分布的采样，使梯度可以传到参数矩阵$F$。

2.6.3 正则项

实际应用中为了使梯度下降收敛到较好的局部最优解，可对参数矩阵$\sigma(M)$和F取element-wise entropy加入loss使参数$\sigma(M)$和F稀疏化，即$L_{entropy} = \frac{1}{n^2}\sum_{j,k} H(\sigma(M)_{j,k})$。

```
# entropy
mask_ent = -mask * torch.log(mask) - (1 - mask) * torch.log(1 - mask)
mask_ent_loss = self.coeffs["ent"] * torch.mean(mask_ent)
```

```python
feat_mask_ent = - feat_mask             \
                * torch.log(feat_mask)  \
                - (1 - feat_mask)       \
                * torch.log(1 - feat_mask)

feat_mask_ent_loss = self.coeffs["feat_ent"] * torch.mean(feat_mask_ent)
```

2.6.4 multi-instance explanations through graph prototypes

3. 拓展

3.1 weighted graph

Algorithm 1 Optimize mask for weighted graph

Input: 1. $G_c(u)$, computation graph of node u; 2. Pre-trained GNN model Φ; 3. y_u, node u's real label; 4. $\mathcal{M}$, learn-able mask; 5. K, number of optimization iterations; 6. L, number of layers of GNN.

$\mathcal{M} \leftarrow$ randomize parameters ▷ initialize, $\mathcal{M} \in [0,1]^Q$
$\mathbf{h}_v^{(0)} \leftarrow \mathbf{x}_v$, for $v \in G_c(u)$
for $k = 1$ to K **do**
  $\mathcal{M}_{vw} \leftarrow \frac{exp(\mathcal{M}_{vw} e_{vw})}{\sum_v exp(\mathcal{M}_{vw} e_{vw})}$ ▷ renormalize mask
  for $l = 1$ to L **do**
    $\mathbf{m}_{vu}^{(l)} \leftarrow W_1^{(l-1)} \mathbf{h}_v^{(l-1)}$ ▷ message
    $M_u^{(l)} \leftarrow \sum_v g(\mathcal{M}_{vu} \mathbf{m}_{vu}^{(l)}, \mathbf{h}_u^{(l-1)})$ ▷ aggregate
    $\mathbf{h}_u^{(l)} \leftarrow \sigma(W_0 \mathbf{h}_u^{(l-1)} + M_u^{(l)})$ ▷ update
  end for
  $\hat{\mathbf{y}}_u \leftarrow softmax(\mathbf{h}_u^{(L)})$ ▷ predict on masked graph
  $loss \leftarrow crossentropy(\mathbf{y}_u, \hat{\mathbf{y}}_u) + regularizations$
  $\mathcal{M} \leftarrow optimizer(loss, \mathcal{M})$ ▷ update mask
end for
Return: $\mathcal{M}$

4. 实验

见论文

5. 代码分析

https://github.com/RexYing/gnn-model-explainer
待填坑

6. 总结

7. 参考文献

1. Variational Inference: A Review for Statisticians David M. Blei, Alp Kucukelbir, Jon D. McAuliffe
2. GNNExplainer: Generating Explanations for Graph Neural Networks Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, Jure Leskovec
3. Semi-Supervised Classification with Graph Convolutional Networks Thomas N. Kipf, Max Welling
4. Inductive Representation Learning on Large Graphs William L. Hamilton, Rex Ying, Jure Leskovec
5. Graph Embedding Techniques, Applications, and Performance: A Survey Palash Goyal, Emilio Ferrara
6. https://github.com/RexYing/gnn-model-explainer
7. Auto-Encoding Variational Bayes Diederik P Kingma, Max Welling
8. Visualizing Deep Neural Network Decisions: Prediction Difference Analysis Luisa M Zintgraf, Taco S Cohen, Tameem Adel, Max Welling
9. Classifying and Understanding Financial Data Using Graph Neural Network Xiaoxiao Li1∗ Joao Saude 2 Prashant Reddy 2 Manuela Veloso2
10. Hierarchical Graph Representation Learning with Differentiable Pooling