I. 论文阅读

一、定义

1. 一些基本的符号定义。
把 G = (V, E)记作一个图，其中 V 是节点的集合，E ⊆ V x V 是边的集合。假定每个节点 v_i 对应一个 k 维实值特征向量 $w_i \in R^k$，和一个标签集合 $y_i \in \{0,..., M-1\}$。节点有 M 个。如果节点 v_i 的标签 y_i 未知，我们就说节点 v_i 是一个无标记的节点。把有标记的节点集合记作 V^L，无标记的节点集合记作 $V^U = V \backslash V^L$。通常$|V^L| << |V^U|$。称图 G 为部分标记图。

2. 定义图上的半监督学习
已知部分标记图 $G = (V^L \cup V^U, E)$，使用与每个节点和图结构相关的特征矩阵 W 学习到一个函数 f，以此预测图中无标记样本的标签。
注意：在半监督学习中，训练和预测通常是同时执行的。在这个例子中，学习同时考虑有标记和无标记的样本，以及整个图的结构。

二、生成对抗模型

生成对抗模型 GAN 是一个通过对抗过程猜测生成模型的新的框架。生成模型 G 被训练以最大限度地 fit 原始的训练数据。判别模型 D 被训练以最大限度地区分由模型 G 生成的样本和真实的样本。这个过程可以被公式化为一个 G 和 D 之间的最小-最大博弈，损失函数如下：

$$\min_G \max_D V(G, D) = \mathbb{E}_{\mathbf{x} \sim p_d(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} \log[1 - D(G(\mathbf{z}))] \tag{1}$$

p_d 是训练数据中的数据分布。$p_z(z)$是输入噪声变量的先验。

三、GraphSGAN 模型框架

1.动机：

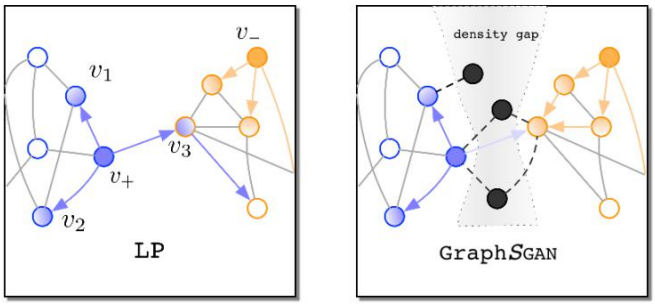

Figure 1: A qualitative illustration of working principles in GraphSGAN. The two labeled nodes are in solid blue and solid orange respectively and all the other nodes are unlabeled nodes. The left figure is a bad case of vanilla Label Propagation algorithm. In this case, node v_3 is assigned a wrong label due to its direct link to node v_+. The right figure illustrates how GraphSGAN works. It generates fake nodes (in black) in the density gap thus reduces the influence of nodes across the density gap.

将 GAN 直接应用于图学习是不可行的，因为它不考虑图的结构。图 1 中左边的图是基于图

的半监督学习的一个典型例子。传统的方法如标签传播法没有考虑图的拓扑，因此无法区分节点 v_+到节点 v_1、v_2 和 v_3 的传播。仔细看看图结构，我们可以看到有两个子图。我们把两个子图之间的面积称为密度间隙 density gap。

文章的想法是利用 GAN 来估计密度子图，然后在密度间隙区域生成样本。然后要求分类器首先对假样本进行识别，然后再对其进行分类。这样,从真实样本中分类出假样本将导致学习到的分类函数在 density gap 附近的曲率会更高，这会导致穿过 density gap 的 propagation 会被削弱(如图 1 中的右图所示)。因为 supervised loss decreasing 和 general smoothing techniques（例如随机层）的存在，导致正确样本的信心将会逐渐提高。

2.架构：

GraphSGAN 首先使用网络嵌入方法（DeepWalk, LINE, NetMF），学习每个节点的潜在分布表示 q_i ，然后将潜在分布 q_i 与原始特征向量 w_i 连接，例如, $x_i = (w_i, q_i)$. 最终 x_i 被用作输入。

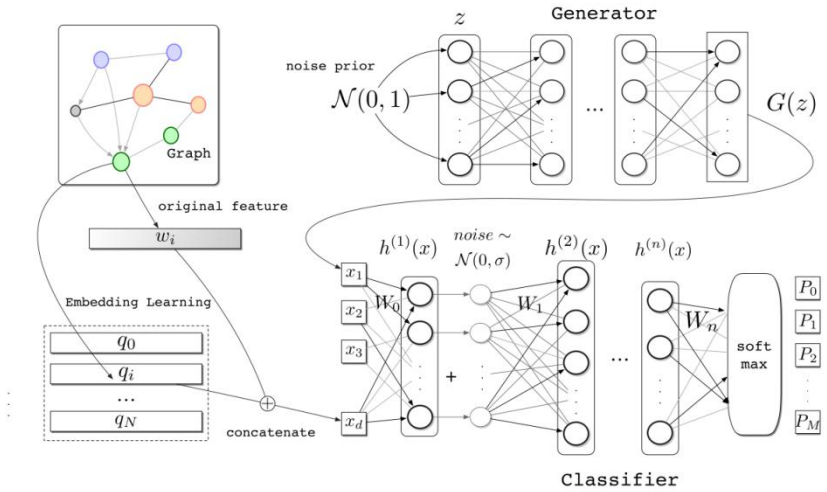

Figure 2: An overview of our model. Fake inputs are generated by generator and real inputs are acquired by concatenating original feature w_i and learned embedding q_i. Both real inputs and fake samples generated by generator are fed into the classifier.

图 2 显示了 GraphSGAN 的架构。GraphSGAN 中的分类器 D 和生成器 G 都是多层感知器。具体来说，生成器以高斯噪声 z 作为输入，输出有着与 x_i 形状相似的假样本。在生成器中，使用批量归一化。生成器的输出层由权值归一化技巧与一个可训练的权值尺度约束。

GAN 中的判别器被一个分类器替代，在输入层和全连接层之后加入了随机层(加性高斯噪声)进行平滑处理。在预测模式下，噪声被移除了。全连接层中的参数通过权值归一化进行约束。分类器 $h^{(n)}(x)$中的最后一个隐含层的输出是通过对输入 x 进行非线性变换来提取到的特征，这对于训练 generator 时进行的特征匹配至关重要。分类器以(M + 1)单元输出层和 softmax 激活结束。单位 0 到单位 M - 1 的输出可以解释为不同类别的概率，单位 M 的输出表示为假的概率。在实践中，我们只考虑前 M 个单位，并假设分类为假的类 P_M 在 softmax 之前的输出总是 0，因为在 softmax 之前的所有单位减去一个相同的数字不会改变 softmax 结果。

3.学习算法：

A. 博弈和均衡

GANs 试图生成与训练数据相似的样本，但我们想在密度间隙中生成假样本。因此，所提出的 GraphSGAN 模型中的优化目标必须与原始 GANs 有所不同。为了更好的解释，首先从博弈论中一个更普遍的角度来回顾 GANs。

在一个普通的两方博弈中，G 和 D 有自己的损失函数，并试图最小化它们。G 和 D 的损失

是相互依存的。损失函数分别记为 $L_G(G, D)$和 $L_D(G, D)$，效用函数 $V_G(G, D)$和 $V_D(G, D)$为负损失函数。

GANs 定义了一个零和博弈，其中 $L_G(G, D) = -L_D(G, D)$，在这种情况下，唯一的纳什均衡可以通过极大极小策略达到。找到平衡解等价于求解优化问题:

$$\min_G \max_D V_D(G, D)$$

Goodfellow 等证明，如果 $V_D(G, D)$定义为式 1，则 G 会生成在平衡时服从于数据分布的样本。生成样本 $p_G(x)$的分布是真实数据 $p_D(x)$的分布的近似。但此时我们希望在密度间隙中生成样本，而不是仅仅模拟真实数据。所以，原来的 GANs 无法解决这个任务。在提出的 GraphSGAN 中，LD(G, D)和 LG(G, D)得到了修改，一种新的博弈被设计，即在均衡状态下，G 会在密度间隙中生成样本。更准确地说，期望真样本和假样本在其最终代表层 $h^{(n)}(x)$中能像图 3 那样映射。由于“密度间隙”的概念在表示层中比在图中更直观，所以作者们定义一个节点落在密度间隙当且仅当它落在 $h^{(n)}(x)$层的密度间隙之中。

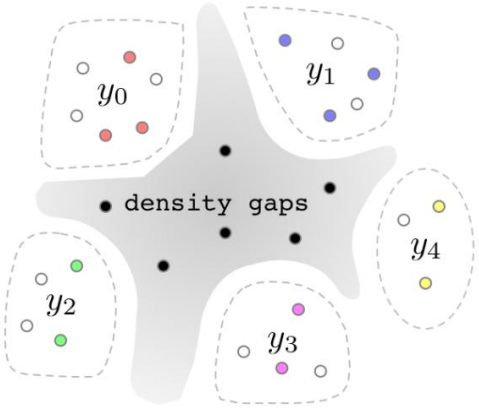

Figure 3: An illustration of the expected equilibrium in $h^{(n)}(x)$. The dotted areas are clusters of training data with different labels(colorful points). Unlabeled samples(white points) are mapped into a particular clusters. Distinct density gaps appear in the center, in which lie the generated samples(black points).

这个设计背后的直觉是基于著名的“维数诅咒”现象。在像 $h^{(n)}(x)$这样的高维空间中，中心区域远比外部区域狭窄。处在中心区域的训练数据很容易成为集散点。集散点经常出现在其他类样本的最近邻上，这可能会严重影响半监督学习，成为半监督学习的主要难点。所以，作者们希望中心区域成为一个密度间隙，而不是一个集群。

为保证期望达到的平衡，$L_G(G, D)$和 $L_D(G, D)$定义如下：

$$\begin{aligned} \mathcal{L}_D &= loss_{sup} + \lambda_0 loss_{un} + \lambda_1 loss_{ent} + loss_{pt} \\ \mathcal{L}_G &= loss_{fm} + \lambda_2 loss_{pt} \end{aligned} \tag{2}$$

B.帮助判别的损失函数：

在均衡状态下，没有玩家可以单方面改变策略来减少自己的损失。假设 G 在平衡时在中心区域生成样本，文章作者对 D 提出了四个条件以保证在 $h^{(n)}(x)$中达到期待的平衡：

(1)将来自不同类的节点映射到不同的簇中。

(2)标记节点和无标记节点都不映射到中心区域，使中心区域成为密度间隙。

(3)将每个未标记节点映射到一个表示特定标签的簇中。

(4)不同的簇应该离得足够远。

满足条件 1 的最自然的方式是一个监督损失 $loss_{sup}$.$loss_{sup}$被定义为 M 个类的预测分布和真实

标签的 one-hot representation 之间的交叉熵。

$$loss_{sup} = -\mathbb{E}_{\mathbf{x}_i \in X^L} \log P(y_i|\mathbf{x}_i, y_i < M) \quad (3)$$

X^L 对有标记节点 V^L 的输入集合。

假设 G 在中心密度间隙中产生假样本,则条件 2 等价于在原始 GAN 中 D 的目标。因此我们仍然使用等式 1 中的 loss，并称其为 $loss_{un}$。当真误分类和假误分类发生时，分类器会产生损失。

$$\begin{aligned} loss_{un} = &- \mathbb{E}_{\mathbf{x}_i \in X^U} \log[1 - P(M|\mathbf{x}_i)] \\ &- \mathbb{E}_{\mathbf{x}_i \sim G(\mathbf{z})} \log P(M|\mathbf{x}_i) \end{aligned} \quad (4)$$

X^U 是对无标记节点 V^L 的预处理输入集合。G(Z)是生成样本的分布，$P(M|x_i)$表示预测 x_i 为假的概率。

条件 3 要求 D 为每个未标记的节点分配一个明确的标签。添加一个熵正则项可以解决这个问题，即 M 个标签上的分布熵。熵是对概率分布的不确定性的度量。很长一段时间以来，它在半监督学习中已经成为一个正则化术语，并首次在参考文献[28]中与 GANs 首次结合。减少熵可以促使分类器为每个节点确定一个明确的标签。

$$loss_{ent} = -\mathbb{E}_{\mathbf{x}_i \in X^U} \sum_{y=0}^{M-1} P(y|\mathbf{x}_i, y_i < M) \log P(y|\mathbf{x}_i, y_i < M) \quad (5)$$

条件 4 增大密度间隙以帮助分类。我们利用拉远项 $loss_{pt}$ 来满足它。$loss_{pt}$ 最初在普通 GANs 中设计用于生成不同的样本。它是一批向量之间的平均余弦距离。它使得在 $h^{(n)}(x)$层的表示互相之间尽可能远离彼此。因此，它还鼓励簇和簇之间互相远离。

$$loss_{pt} = \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i} \frac{h^{(n)}(\mathbf{x}_i)^\top h^{(n)}(\mathbf{x}_j)}{||h^{(n)}(\mathbf{x}_i)||||h^{(n)}(\mathbf{x}_j)||}^2 \quad (6)$$

x_i, x_j 在同一批里面，m 是批次的尺寸。

C.帮助生成的损失函数：

假设 D 已经满足了四个条件，另有两个针对 G 的条件以促使我们达到所期待的平衡。

(1)G 生成那些映射到了中心区域的样本。

(2)生成的样本不能过拟合于唯一的中心点。

对于条件 1，使用特征匹配 loss 训练 G，它使生成的样本与真实样本的中心点 $\mathbb{E}_{\mathbf{x}_i \in X^U \cup X^L} h^{(n)}(\mathbf{x}_i)$ 之间的距离最小化。实际上，在训练过程中，中心点被真实批次的样本中心替换为 $\mathbb{E}_{\mathbf{x}_i \in X_{batch}} h^{(n)}(\mathbf{x}_i)$，这有助于满足条件 2.距离最初是用 L2 范数测量的。(但是，在实践中，我们发现 L1 范数也可以很好地工作，性能甚至稍微好一点。)

$$loss_{fm} = ||\mathbb{E}_{\mathbf{x}_i \in X_{batch}} h^{(n)}(\mathbf{x}_i) - \mathbb{E}_{\mathbf{x}_j \sim G(\mathbf{z})} h^{(n)}(\mathbf{x}_j)||_2^2 \quad (7)$$

条件 2 要求生成的样本尽可能多地覆盖住中心的区域。我们也需要一个拉远项（等式 6）来保证这一条件的满足，因为它鼓励 G 生成分散的样本。一个中心性和分散性的 trade-off 是需要的，因此我们使用了一个超参数 λ_2 来平衡 $loss_{fm}$ 和 $loss_{pt}$。D 中的随机层给假的输入添加了噪声，这不仅仅提升了鲁棒性，还阻止假样本过拟合。

D.训练：GAN 通过迭代地最小化 D 和 G 的 loss 来训练 D 和 G。在博弈论中，它被称为短

视最佳响应，是一种寻找平衡点的有效启发式方法。GraphSGAN 也是这样训练的。训练的第一部分是将图中的节点转化为特征空间中的向量。文章作者使用 LINE[29]对 **q** 进行预处理。为了加快收敛速度，采用 neighbor fusion technique 重新计算节点的特征。设 $Ne(v_i)$为 v_i 的邻居集，节点 v_i 的权值重新计算为：

$$\mathbf{w}_i' = \alpha \mathbf{w}_i + \frac{1-\alpha}{|Ne(v_i)|} \sum_{v_j \in Ne(v_i)} \mathbf{w}_j. \qquad (8)$$

邻居融合的想法类似于使用注意力机制的预处理技巧。

在主训练循环中，迭代地训练 D 和 G。为了计算 LD，需要三批样本，分别是有标记的、无标记的和生成的。$loss_{sup}$ 需要有标记的数据。$loss_{un}$ 是基于无标记的和生成的样本计算的。理论上讲，$loss_{un}$ 也应该考虑标记数据，以确保它们被归类为真实数据。但 $loss_{sup}$ 已经将标签数据正确归类为其真实标签，因此在 $loss_{un}$ 中无需考虑标签数据。$loss_{ent}$ 仅仅考虑无标记数据，$loss_{pt}$ 应该将有标记和无标记的数据拉远。通常三个超参数来平衡四个 loss 的 scale.在 GraphSGAN 中只考虑两个超参数因为 $loss_{sup}$ 将会被很快地优化到 0 因为在半监督设定下有标记的样本很少。

训练 G.需要真实批次和生成批次的数据。$loss_{fm}$ 比较真实数据和 $h^{(n)}(x)$中生成数据的批次中心，$loss_{pt}$ 度量生成数据的分散性。总是希望 G 在中心区域生成样本。因此，每次训练 D 之后，我们都要训练 G 几个步骤来收敛。具体过程在算法 1 中说明。

Algorithm 1: Minibatch stochastic gradient descent training of Graph*S*GAN

Input: Node features $\{\mathbf{w}_i\}$, Labels y^L, Graph $G = (V, E)$, Embedding Algorithm $\mathcal{A}$, batch size m.

Calculate $\{\mathbf{w}_i'\}$ according to Eq. (8)

Calculate $\{\mathbf{q}_i\}$ via $\mathcal{A}$

Concatenate $\{\mathbf{w}_i'\}$ with $\{\mathbf{q}_i\}$ for $X^L \cup X^U$

repeat

  Sample m labeled samples $\{\mathbf{x}_1^L, ..., \mathbf{x}_m^L\}$ from X^L

  Sample m unlabeled samples $\{\mathbf{x}_1^U, ..., \mathbf{x}_m^U\}$ from X^U

  Sample m noise samples $\{\mathbf{z}_1, ..., \mathbf{z}_m\}$ from $p_{\mathbf{z}}(\mathbf{z})$

  Update the classifier by descending gradients of losses:

$$\nabla_{\theta_D} \frac{1}{m} \sum loss_{sup} + \lambda_0 loss_{un} + \lambda_1 loss_{ent} + loss_{pt}$$

  for *t steps* **do**

    Sample m unlabeled samples $\{\mathbf{x}_1^U, ..., \mathbf{x}_m^U\}$ from X^U

    Sample m noise samples $\{\mathbf{z}_1, ..., \mathbf{z}_m\}$ from $p_{\mathbf{z}}(\mathbf{z})$

    Update the generator by descending gradients of losses:

$$\nabla_{\theta_G} \frac{1}{m} \sum loss_{fm} + \lambda_2 loss_{pt}$$

  end

until *Convergence*;

II. 程序实现

仅仅完成了文献阅读，代码实现及结果比较尚未完成。