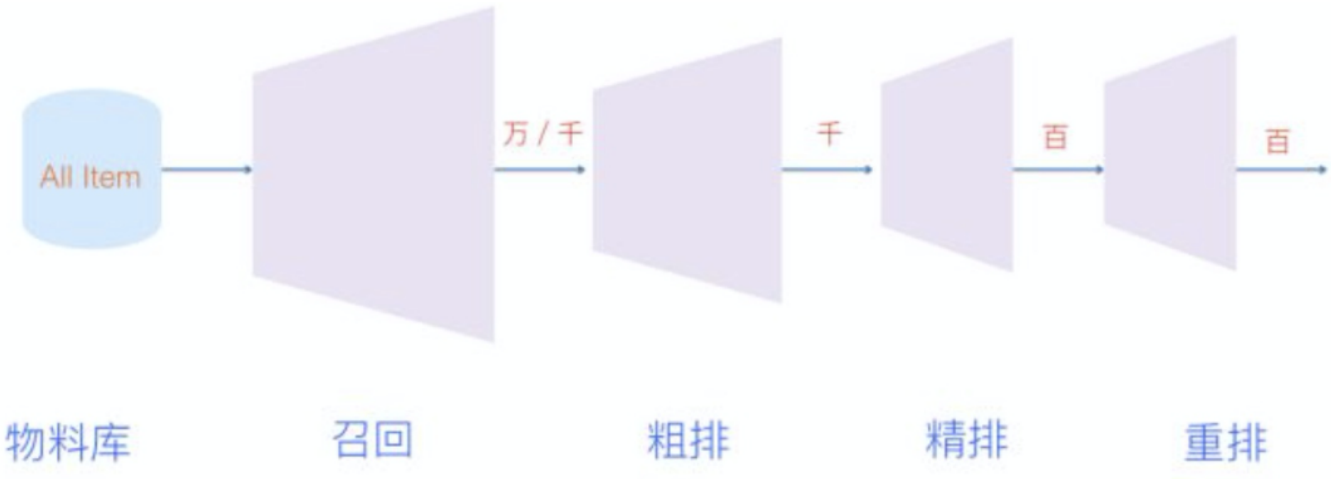
GraphSAGE在商品推荐中的应用

原创 长风 机器学习与数据挖掘实践 2020-10-16

收录于话题
#推荐系统学习

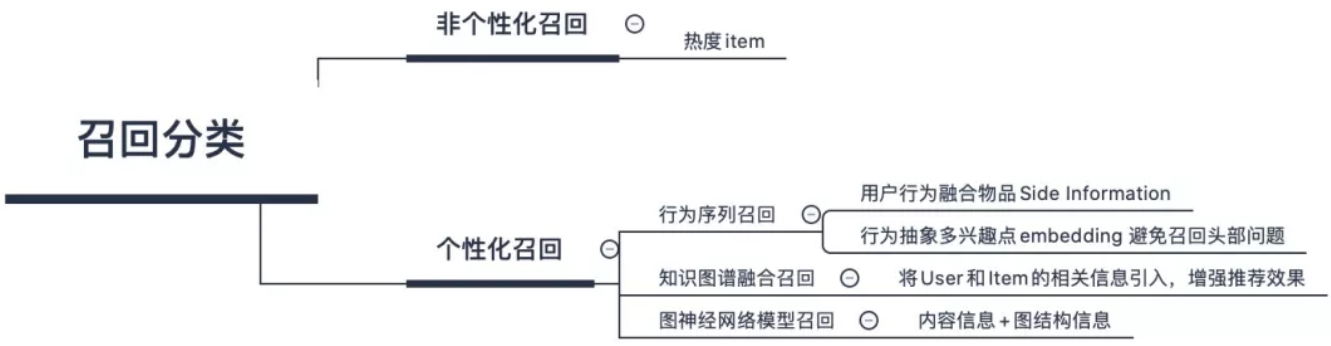
1个



如上图所示，推荐系统一般由召回、粗排，精排，重排这几部分组成，本文介绍的GraphSAGE，主要在召回环节应用，GraphSAGE则是一种能够利用顶点的属性信息高效产生未知顶点embedding的一种归纳式(inductive)学习的框架，其核心思想是通过学习一个对邻居顶点进行聚合表示的函数来产生目标顶点的embedding向量。

1. 召回回顾

标准召回结构一般是多路召回，根据召回路是否有用户个性化因素存在来划分，可以分成两大类：一类是无个性化因素的召回路，比如热门商品或者热门文章或者历史点击率高的物料的召回；另外一类是包含个性化因素的召回路，比如用户兴趣标签召回。



图神经网络的中图节点可以带有属性信息，比如物品的Content信息，所以明显这对于解决物品侧的冷启动问题有帮助；而因为它也允许知识在图中远距离进行传递，所以比如对于用户行为比较少的场景，可以形成知识传递和补充，这说明它也比较适合用于数据稀疏的推荐场景及冷启动；同时图还具备的一

个很好的优势是：它比较便于把协同信息、用户行为信息、内容属性信息等各种异质信息在一个统一的框架里进行融合，并统一表征为embedding的形式。最终获得图中节点的embedding，融合了各种异质信息。所以它是特别适合用来做召回的，比如拿到图网络中用户的embedding和物品embedding，可以直接用来做向量召回。

早期的图神经网络做推荐，因为需要全局信息，所以计算速度是个问题，往往图规模都非常小，不具备实战价值。而GraphSAGE则通过一些手段比如从临近节点进行采样等减少计算规模，加快计算速度，拓展了图计算的实用性。

2. GraphSAGE原理介绍

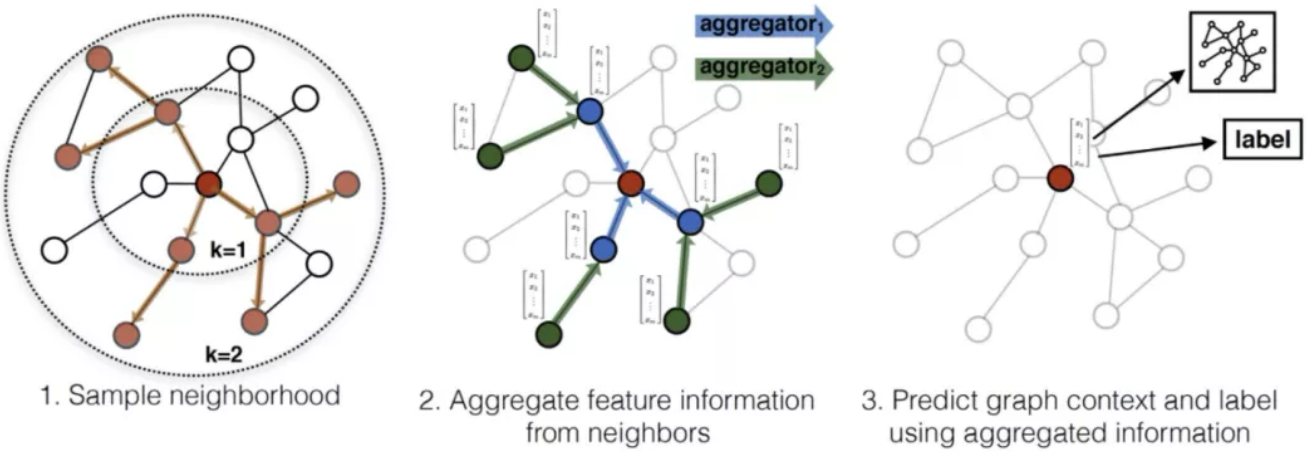


Figure 1: Visual illustration of the GraphSAGE sample and aggregate approach.

GraphSAGE 是Graph SAmple and aggreGatE的缩写，其运行流程如上图所示，可以分为三个步骤

1. 对图中每个顶点邻居顶点进行采样
2. 根据聚合函数聚合邻居顶点蕴含的信息
3. 得到图中各顶点的向量表示供下游任务使用

a. 采样邻居顶点

出于对计算效率的考虑，对每个顶点采样一定数量的邻居顶点作为待聚合信息的顶点。设采样数量为k，若顶点邻居数少于k,则采用有放回的抽样方法，直到采样出k个顶点。若顶点邻居数大于k，则采用无放回的抽样。

当然，若不考虑计算效率，我们完全可以对每个顶点利用其所有的邻居顶点进行信息聚合，这样是信息无损的。

b. 生成向量的伪代码

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$

Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

1 $\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
2 **for** $k = 1...K$ **do**
3 **for** $v \in \mathcal{V}$ **do**
4 $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
5 $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
6 **end**
7 $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
8 **end**
9 $\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

这里K是网络的层数，也代表着每个顶点能够聚合的邻接点的跳数，如K=2的时候每个顶点可以最多根据其2跳邻接点的信息学习其自身的embedding表示。

在每一层的循环k中，对每个顶点v，首先使用v的邻接点的k-1层的embedding表示 h_u^{k-1} 来产生其邻居顶点的第k层聚合表示 $h_{N(v)}^k$,之后将 $h_{N(v)}^k$ 和顶点v的第k-1层表示 h_v^{k-1} 进行拼接，经过一个非线性变换产生顶点v的第k层embedding表示 h_v^k 。

c. 聚合函数的选取

由于在图中顶点的邻居是天然无序的，所以我们希望构造出的聚合函数是对称的（即改变输入的顺序，函数的输出结果不变），同时具有较高的表达能力。

- MEAN aggregator

$$h_v^k \leftarrow \sigma(\boldsymbol{W} \cdot \mathbf{MEAN}(\{h_v^{k-1}\} \cup \{h_u^{k-1}, \forall u \in N(v)\})$$

上式对应于伪代码中的第4-5行，直接产生顶点的向量表示，而不是邻居顶点的向量表示。mean aggregator将目标顶点和邻居顶点的第k-1层向量拼接起来，然后对向量的每个维度进行求均值的操作，将得到的结果做一次非线性变换产生目标顶点的第k层表示向量。

- Pooling aggregator

$$AGGREGATE^{pool}k = \max(\{\sigma(\boldsymbol{W}pool h_{u_i}^k + b), \forall u_i \in N(v)\})$$

Pooling aggregator 先对目标顶点的邻接点表示向量进行一次非线性变换，之后进行一次pooling操作(maxpooling or meanpooling)，将得到结果与目标顶点的表示向量拼接，最后再经过一次非线性变换得到目标顶点的第k层表示向量。

- LSTM aggregator

LSTM相比简单的求平均操作具有更强的表达能力，然而由于LSTM函数不是关于输入对称的，所以在使用时需要对顶点的邻居进行一次乱序操作。

d. 参数的学习

在定义好聚合函数之后，接下来就是对函数中的参数进行学习。文章分别介绍了无监督学习和监督学习两种方式。

- 无监督学习形式

基于图的损失函数希望临近的顶点具有相似的向量表示，同时让分离的顶点的表示尽可能区分。目标函数如下

$$J_{\mathcal{G}}(\mathbf{z}_u) = -\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right),$$

其中v是通过固定长度的随机游走出现在u附近的顶点，p_n 是负采样的概率分布，Q 是负样本的数量。

与DeepWalk不同的是，这里的顶点表示向量是通过聚合顶点的邻接点特征产生的，而不是简单的进行一个embedding lookup操作得到。

- 监督学习形式

监督学习形式根据任务的不同直接设置目标函数即可，如最常用的节点分类任务使用交叉熵损失函数。

3. GraphSAGE实践

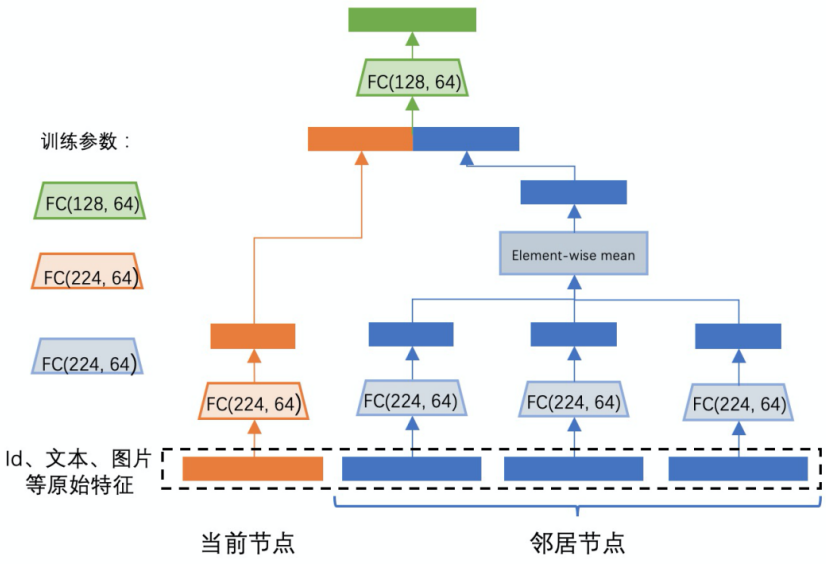
a. 图构建

图主要是利用用户的历史行为日志来构建。商品item作为图节点，同一个session内的连续行为建立边，同时会考虑边的类型，并将点击次数作为边权重。节点信息包括以下三类特征，id类特征、统计类特征和稠密(Dense)特征。id类特征包括商品id、类目id、商家id等，id类特征都会进行hash。统计类特征包括30天和90天内商品的ctr、cvr等，稠密特征包括商品的图片向量和文本向量等。

注意：

1. 在构建过程中考虑对超级节点进行过滤，只保留top N最近的邻居节点。如果噪声比较多，可以限制在同一级类目下才建立边。
2. 稠密特征需进行归一化（x/||x||）

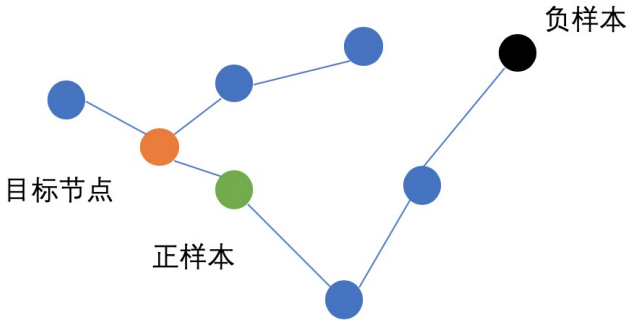
b. 模型



这里我们使用两阶聚合的方式，两阶比一阶有比较大的增益，三阶运算开销较大并且收益不大。聚合模式试过mean，mean-pool，max-pool，以及attention的方式（其实也就相当于邻居采样版本的

GAT），其中除了mean以外，后面几种聚合方式在最后的离线效果上没有看到特别大的差异。在邻居采样上，我们根据边权重来做采样，这样行为次数多和行为类型权重大的邻居将会被大概率采样到。

C. 无监督学习样本



无监督的样本是选取一些目标节点，正样本一般为1个且选取目标节点的一阶邻居，负样本一般为随机选取N个。

在无监督学习中，负样本的选取对最终模型的效果有非常重要的影响。如果负样本的个数过少，对于模型有亿级别的商品集来说，随机选取的这几十个负样本与正样本有可能差别非常大，模型不用充分学习就可分辨正负样本。但是如果单纯增大负样本数量的话，又会增加运算开销，可以通过增加难学习样本缓解该问题。比较好的方法是融合“容易”和“困难”样本一起训练。

4. 思考

1. 用户的兴趣随着时间变化会发生迁移，如何在最后的embedding不仅仅是把item的side information融合进来，同时把时序的特征融入是一个。
2. 整个模型构造和训练迭代的时间较场，如何在链路上缩短时间
3. mrr无法直接作为最后效果好坏的评价指标，因为与负样本的选取直接相关。
4. 选择负样本时进行全局随机样本进行取代，无需对每个目标节点进行选择，从而降低计算量实现加速训练。

推荐阅读

从面试官视角看算法面试如何考察

Focal Loss | 解决样本不平衡利器

推荐系统 | RecSys2019 新浪微博 FiBiNET

推荐系统 | KDD2019 阿里Res-embedding

你点的每个好看，我都认真当成了喜欢 🌹