07｜Embedding进阶：如何利用图结构数据生成Graph Embedding？

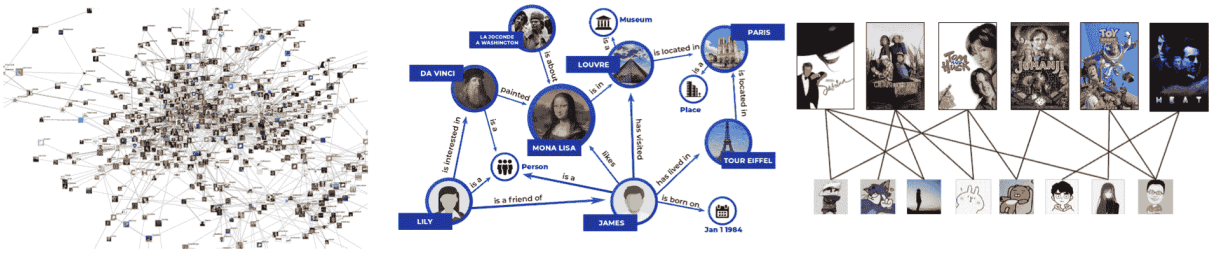
你好，我是王喆。

上一节课，我们一起学习了 Embedding 技术。我们知道，只要是能够被序列数据表示的物品，都可以通过 Item2vec 方法训练出 Embedding。但是，互联网的数据可不仅仅是序列数据那么简单，越来越多的数据被我们以图的形式展现出来。这个时候，基于序列数据的 Embedding 方法就显得“不够用”了。但在推荐系统中放弃图结构数据是非常可惜的，因为图数据中包含了大量非常有价值的结构信息。

那我们怎么样才能够基于图结构数据生成 Embedding 呢？这节课，我们就重点来讲讲基于图结构的 Embedding 方法，它也被称为 Graph Embedding。

互联网中有哪些图结构数据？

可能有的同学还不太清楚图结构中到底包含了哪些重要信息，为什么我们希望好好利用它们，并以它们为基础生成 Embedding？下面，我就先带你认识一下互联网中那些非常典型的图结构数据（如图 1）。



（a）社交关系图　　（b）知识图谱　　（c）行为关系图

图1 互联网图结构数据

事实上，图结构数据在互联网中几乎无处不在，最典型的就是我们每天都在使用的**社交网络**（如图 1-a）。从社交网络中，我们可以发现意见领袖，可以发现社区，再根据这些“社交”特性进行社交化的推荐，如果我们可以对社交网络中的节点进行 Embedding 编码，社交化推荐的过程将会非常方便。

知识图谱也是近来非常火热的研究和应用方向。像图 1-b 中描述的那样，知识图谱中包含了不同类型的知识主体（如人物、地点等），附着在知识主体上的属性（如人物描述，物品特点），以及主体和主体之间、主体和属性之间的关系。如果我们能够对知识图谱中的主体进行 Embedding 化，就可以发现主体之间的潜在关系，这对于基于内容和知识的推荐系统是非常有帮助的。

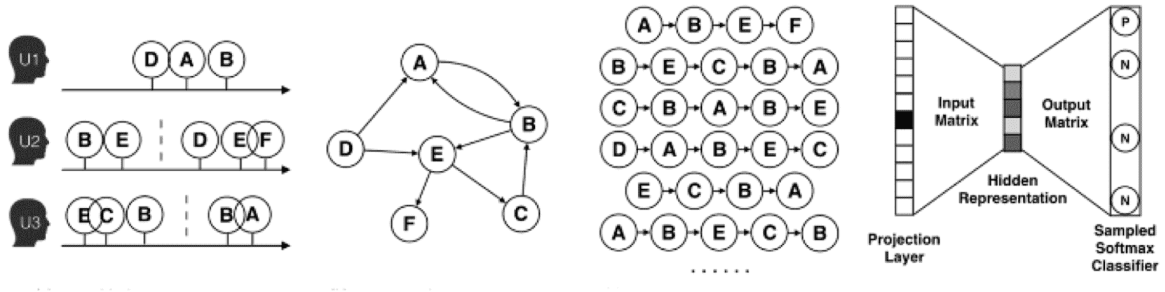
还有一类非常重要的图数据就是**行为关系类图数据**。这类数据几乎存在于所有互联网应用中，它事实上是由用户和物品组成的“二部

图”（也称二分图，如图 1c）。用户和物品之间的相互行为生成了行为关系图。借助这样的关系图，我们自然能够利用 Embedding 技术发掘出物品和物品之间、用户和用户之间，以及用户和物品之间的关系，从而应用于推荐系统的进一步推荐。

毫无疑问，图数据是具备巨大价值的，如果能将图中的节点 Embedding 化，对于推荐系统来说将是非常有价值的特征。那下面，我们就进入正题，一起来学习基于图数据的 Graph Embedding 方法。

基于随机游走的 Graph Embedding 方法，Deep Walk

我们先来学习一种在业界影响力比较大，应用也很广泛的 Graph Embedding 方法，Deep Walk，它是 2014 年由美国石溪大学的研究者提出的。它的主要思想是在由物品组成的图结构上进行随机游走，产生大量物品序列，然后将这些物品序列作为训练样本输入 Word2vec 进行训练，最终得到物品的 Embedding。因此，DeepWalk 可以被看作连接序列 Embedding 和 Graph Embedding 的一种过渡方法。下图 2 展示了 DeepWalk 方法的执行过程。



（a）用户行为序列　（b）构建物品关系图　（c）随机游走　（d）使用Word2vec模型产生Embedding

图2 DeepWalk方法的过程

接下来，我就参照图 2 中 4 个示意图，来为你详细讲解一下 DeepWalk 的算法流程。

首先，我们基于原始的用户行为序列（图 2(a)），比如用户的购买物品序列、观看视频序列等等，来构建物品关系图（图 2(b)）。从中，我们可以看出，因为用户 U1 先后购买了物品 A 和物品 B，所以产生了一条由 A 到 B 的有向边。如果后续产生了多条相同的有向边，则有向边的权重被加强。在将所有用户行为序列都转换成物品相关图中的边之后，全局的物品相关图就建立起来了。

然后，我们采用随机游走的方式随机选择起始点，重新产生物品序列（图 2©）。其中，随机游走采样的次数、长度等都属于超参数，需要我们根据具体应用进行调整。

最后，我们将这些随机游走生成的物品序列输入图 2(d) 的 Word2vec 模型，生成最终的物品 Embedding 向量。

在上述 DeepWalk 的算法流程中，唯一需要形式化定义的就是随机游走的跳转概率，也就是到达节点 vi后，下一步遍历 vi的邻接点 vj 的概率。如果物品关系图是有向有权图，那么从节点 vi跳转到节点 vj的概率定义如下：

$$P(vj \mid vi) = \{ \sum j \in N+(Vi) Mij, mij vj \in N+(vi) \ 0, e\_ij \in / \varepsilon$$

其中，N+(vi) 是节点 vi所有的出边集合，Mij是节点 vi到节点 vj边的权重，即 DeepWalk 的跳转概率就是跳转边的权重占所有相关出边权重之和的比例。如果物品相关图是无向无权重图，那么跳转概率将是 (式 1) 的一个特例，即权重 Mij将为常数 1，且 N+(vi) 应是节点 vi所有“边”的集合，而不是所有“出边”的集合。

再通过随机游走得到新的物品序列，我们就可以通过经典的 word2vec 的方式生成物品 Embedding 了。当然，关于 word2vec 的细节你可以回顾上一节课的内容，这里就不再赘述了。

在同质性和结构性间权衡的方法，Node2vec

2016 年，斯坦福大学的研究人员在 DeepWalk 的基础上更进一步，他们提出了 Node2vec 模型。Node2vec 通过调整随机游走跳转概率的方法，让 Graph Embedding 的结果在网络的**同质性**（Homophily）和**结构性**（Structural Equivalence）中进行权衡。进一步可以把不同

的 Embedding 输入推荐模型，让推荐系统学习到不同的网络结构特点。

我这里所说的网络的**“同质性”指的是距离相近节点的 Embedding 应该尽量近似**，如图 3 所示，节点 u 与其相连的节点 s1、s2、s3、s4 的 Embedding 表达应该是接近的，这就是网络“同质性“的体现。在电商网站中，同质性的物品很可能是同品类、同属性，或者经常被一同购买的物品。

而**“结构性”指的是结构上相似的节点的 Embedding 应该尽量接近**，比如图 3 中节点 u 和节点 s6 都是各自局域网络的中心节点，它们在结构上相似，所以它们的 Embedding 表达也应该近似，这就是“结构性”的体现。在电商网站中，结构性相似的物品一般是各品类的爆款、最佳凑单商品等拥有类似趋势或者结构性属性的物品。

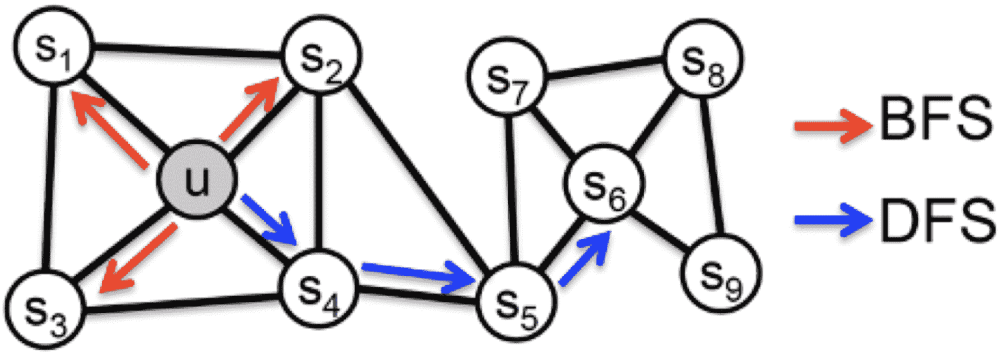


图3 网络的BFS和 DFS示意图

理解了这些基本概念之后，那么问题来了，Graph Embedding 的结果究竟是怎么表达结构性和同质性的呢？

首先，为了使 Graph Embedding 的结果能够表达网络的“**结构性**”，在随机游走的过程中，我们需要让游走的过程更倾向于 **BFS（Breadth First Search，宽度优先搜索）**，因为 BFS 会更多地在当前节点的邻域中进行游走遍历，相当于对当前节点周边的网络结构进行一次“微观扫描”。当前节点是“局部中心节点”，还是“边缘节点”，亦或是“连接性节点”，其生成的序列包含的节点数量和顺序必然是不同的，从而让最终的 Embedding 抓取到更多结构性信息。

而为了表达“**同质性**”，随机游走要更倾向于 **DFS（Depth First Search，深度优先搜索）**才行，因为 DFS 更有可能通过多次跳转，游走到远方的节点上。但无论怎样，DFS 的游走更大概率会在一个大的集团内部进行，这就使得一个集团或者社区内部节点的 Embedding 更为相似，从而更多地表达网络的“同质性”。

那在 Node2vec 算法中，究竟是怎样控制 BFS 和 DFS 的倾向性的呢？

其实，它主要是通过节点间的跳转概率来控制跳转的倾向性。图 4 所示为 Node2vec 算法从节点 t 跳转到节点 v 后，再从节点 v 跳转到周围各点的跳转概率。这里，你要注意这几个节点的特点。比如，节点 t 是随机游走上一步访问的节点，节点 v 是当前访问的节点，节点 x1、x2、x3是与 v 相连的非 t 节点，但节点 x1还与节点 t 相连，这些不同的特点决定了随机游走时下一次跳转的概率。

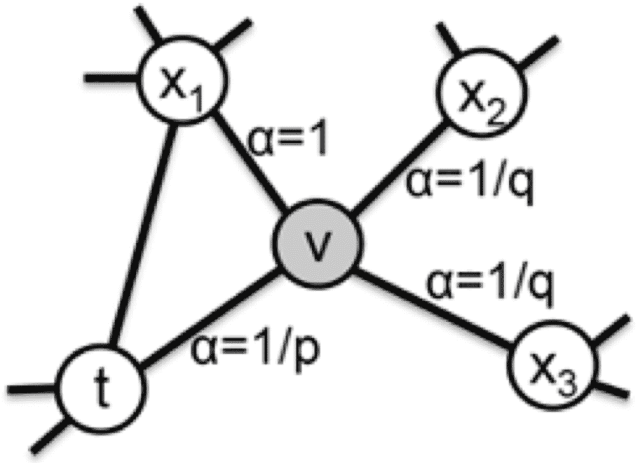


图4 Node2vec的跳转概率

这些概率我们还可以用具体的公式来表示，从当前节点 v 跳转到下一个节点 x 的概率 $\pi_{vx} = \alpha_{pq}(t,x) \cdot \omega_{vx}$ ，其中 w_{vx} 是边 vx 的原始权重，$\alpha_{pq}(t,x)$ 是 Node2vec 定义的一个跳转权重。到底是倾向于 DFS 还是 BFS，主要就与这个跳转权重的定义有关了。这里我们先了解一下它的精确定义，我再作进一步的解释：

$$\alpha_{pq}(t,x)=\begin{cases} p^{-1} & \text{如果 } d_{tx}=0 \\ 1 & \text{如果 } d_{tx}=1 \\ q^{-1} & \text{如果 } d_{tx}=2 \end{cases}$$

$\alpha_{pq}(t,x)$ 里的 dtx是指节点 t 到节点 x 的距离，比如节点 x1其实是与节点 t 直接相连的，所以这个距离 dtx就是 1，节点 t 到节点 t 自己的距离 dtt就是 0，而 x2、x3这些不与 t 相连的节点，dtx就是 2。

此外，$\alpha_{pq}(t,x)$ 中的参数 p 和 q 共同控制着随机游走的倾向性。参数 p 被称为返回参数（return parameter），p 越小，随机游走回节点 t 的可能性越大，Node2vec 就更注重表达网络的结构性。参数 q 被称为进出参数（in-out parameter），q 越小，随机游走到远方节点的可能性越大，Node2vec 更注重表达网络的同质性。反之，当前节点更可能在附近节点游走。你可以自己尝试给 p 和 q 设置不同大小的值，算一算从 v 跳转到 t、x1、x2和 x3的跳转概率。这样一来，应该就不难理解我刚才所说的随机游走倾向性的问题啦。

Node2vec 这种灵活表达同质性和结构性的特点也得到了实验的证实，我们可以通过调整 p 和 q 参数让它产生不同的 Embedding 结果。图 5 上就是 Node2vec 更注重同质性的体现，从中我们可以看到，距离相近的节点颜色更为接近，图 5 下则是更注重结构性的体现，其中结构特点相近的节点的颜色更为接近。

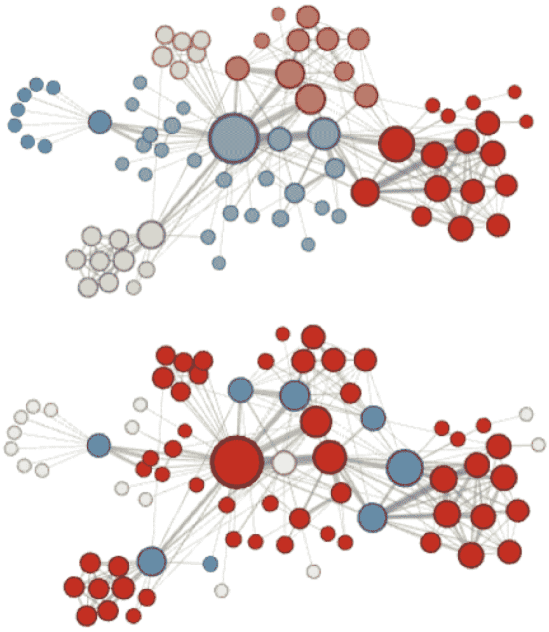


图5 Node2vec实验结果

毫无疑问，Node2vec 所体现的网络的同质性和结构性，在推荐系统中都是非常重要的特征表达。由于 Node2vec 的这种灵活性，以及发掘不同图特征的能力，我们甚至可以把不同 Node2vec 生成的偏向“结构性”的 Embedding 结果，以及偏向“同质性”的 Embedding 结果共同输入后续深度学习网络，以保留物品的不同图特征信息。

Embedding 是如何应用在推荐系统的特征工程中的？

到这里，我们已经学习了好几种主流的 Embedding 方法，包括序列数据的 Embedding 方法，Word2vec 和 Item2vec，以及图数据的 Embedding 方法，Deep Walk 和 Node2vec。那你有没有想过，我为什么要在特征工程这一模块里介绍 Embedding 呢？Embedding 又是是怎么应用到推荐系统中的呢？这里，我就来做一个统一的解答。

第一个问题不难回答，由于 Embedding 的产出就是一个数值型特征向量，所以 Embedding 技术本身就可以视作特征处理方式的一种。只不过与简单的 One-hot 编码等方式不同，Embedding 是一种更高阶的特征处理方法，它具备了把序列结构、网络结构、甚至其他特征融合到一个特征向量中的能力。

而第二个问题的答案有三个。Embedding 在推荐系统中的应用方式大致有三种，分别是“直接应用”、“预训练应用”和“End2End 应用”。

其中，“**直接应用**”最简单，就是在我们得到 Embedding 向量之后，直接利用 Embedding 向量的相似性实现某些推荐系统的功能。典型的功能有，利用物品 Embedding 间的相似性实现相似物品推荐，利用物品 Embedding 和用户 Embedding 的相似性实现“猜你喜欢”等经典推荐功能，还可以利用物品 Embedding 实现推荐系统中的召回层

等。当然，如果你还不熟悉这些应用细节，也完全不用担心，我们在之后的课程中都会讲到。

“**预训练应用**”指的是在我们预先训练好物品和用户的 Embedding 之后，不直接应用，而是把这些 Embedding 向量作为特征向量的一部分，跟其余的特征向量拼接起来，作为推荐模型的输入参与训练。这样做能够更好的把其他特征引入进来，让推荐模型作出更为全面且准确的预测。

第三种应用叫做“**End2End 应用**”。看上去这是个新的名词，它的全称叫做“end to end training”，也就是端到端训练。不过，它其实并不神秘，指的是我们不预先训练 Embedding，而是把 Embedding 的训练与深度学习推荐模型结合起来，采用统一的、端到端的方式一起训练，直接得到包含 Embedding 层的推荐模型。这种方式非常流行，比如图 6 就展示了三个包含 Embedding 层的经典模型，分别是微软的 Deep Crossing，UCL 提出的 FNN 和 Google 的 Wide&Deep。它们的实现细节我们也会在后续课程里面介绍，你这里只需要了解这个概念就可以了。

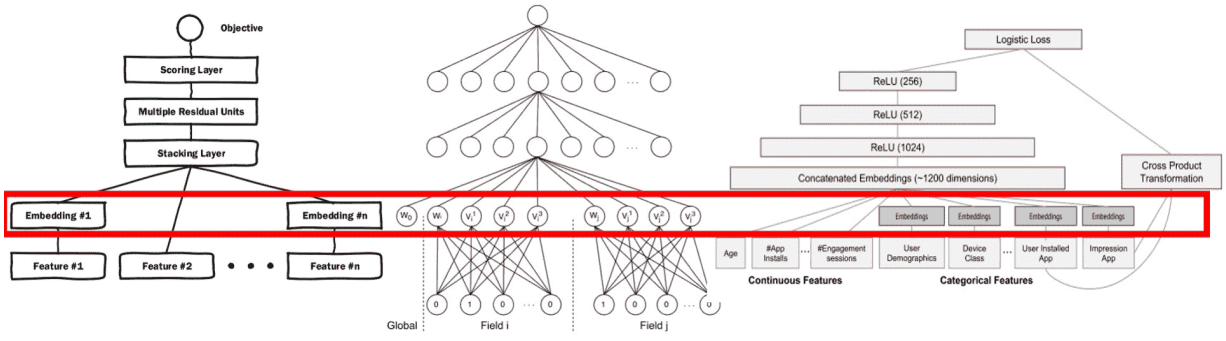


图6 带有Embedding层的深度学习模型

小结

这节课我们一起学习了 Graph Embedding 的两种主要方法，分别是 Deep Walk 和 Node2vec，并且我们还总结了 Embedding 技术在深度学习推荐系统中的应用方法。

学习 Deep Walk 方法关键在于理解它的算法流程，首先，我们基于原始的用户行为序列来构建物品关系图，然后采用随机游走的方式随机选择起始点，重新产生物品序列，最后将这些随机游走生成的物品序列输入 Word2vec 模型，生成最终的物品 Embedding 向量。

而 Node2vec 相比于 Deep Walk，增加了随机游走过程中跳转概率的倾向性。如果倾向于宽度优先搜索，则 Embedding 结果更加体现“结构性”。如果倾向于深度优先搜索，则更加体现“同质性”。

最后，我们介绍了 Embedding 技术在深度学习推荐系统中的三种应用方法，“直接应用”“预训练”和“End2End 训练”。这些方法各有特点，它们都是业界主流的应用方法，随着课程的不断深入，我会带你一步一步揭开它们的面纱。

老规矩，在课程的最后，我还是用表格的方式总结了这次课的关键知识点，你可以利用它来复习巩固。

知识点	关键描述
互联网中的图数据	社交网络、知识图谱、行为关系图
Deep Walk的概念	利用随机游走的方式产生物品序列， 再利用Word2vec生成物品Embedding
Node2vec的概念	在Deep Walk的基础上，调整随机游走时的跳转概率， 从而控制生成Embeding对结构性和同质性的倾向性
Embedding在 深度学习推荐系统中的应用	直接应用、预训练应用、End2End训练应用

至此，我们就完成了所有 Embedding 理论部分的学习。下节课，我们再一起进入 Embedding 和 Graph Embedding 的实践部分，利用 Sparrow Recsys 的数据，使用 Spark 实现 Embedding 的训练，希望你到时能跟我一起动起手来！

课后思考

对于 Embedding 预训练和 Embedding End2End 训练两种应用方法，你能说出它们之间的优劣点吗？

你能尝试对比一下 Embedding 预训练和 Embedding End2End 训练这两种应用方法，说出它们之间的优缺点吗？

欢迎在留言区分享你的思考和答案，如果这节 Graph Embedding 的课程让你有所收获，那不妨也把这节课分享给你的朋友们，我们下节课见！