A Comprehensive Survey of Graph Embedding Problems, Techniques and Applications

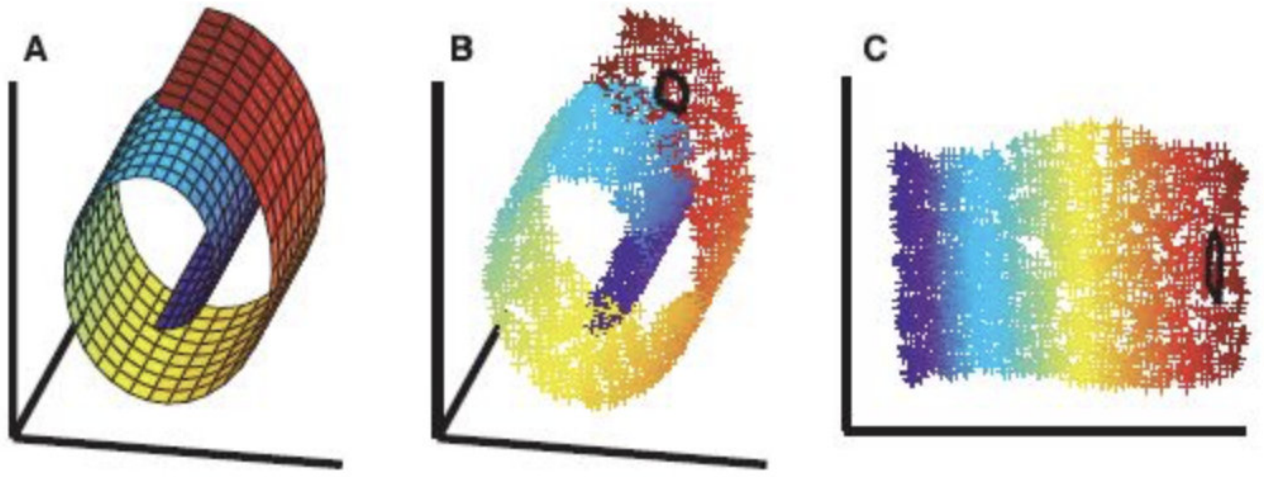
文章摘要

- 图分析的作用：有效的图数据分析可以为用户提供对于隐藏在数据背后的内容更深的理解，进而推动节点分类(node classification)，节点推荐(node recommendation)，链接预测(link prediction)等应用的发展。

  > 隐藏在数据背后的内容指的是节点之间的联系或者某种节点出现的频率所呈现的某种信息等，比如将用户在某个视频网站上看过的每一部电影或者电视剧都看成一个节点，每一个节点都具有一个或者多个标签，如果大部分的节点都具有某一个相同的标签，那么隐藏的内容便是“该用户喜欢看具有该标签的视频”

- 面临的瓶颈：图分析受阻于高计算成本和高空间成本。
- 解决瓶颈的方法：图嵌入(graph embedding)
- 图嵌入的效果/作用：将图数据转化为低维空间的数据并尽可能保留图的结构信息(graph structual information)和图的特征属性(graph properties)。

> 真实的图（网络）往往是高维、难以处理的，20世纪初，研究人员发明了图形嵌入算法，作为降维技术的一部分。他们首先根据实际问题构造一个D维空间中的图，然后将图的节点嵌入到d维向量空间中。嵌入的思想是在向量空间中保持连接的节点彼此靠近。拉普拉斯特征映射（Laplacian Eigenmaps）和局部线性嵌入（Locally Linear Embedding，LLE）是基于这一原理的算法的例子(来自知乎)



降维算法：拉普拉斯特征映射

- 这片文章的主要内容：
  - 对于现有的关于图嵌入的文献的一个综合概述。
  - 提出对于图嵌入的两个分类方法，这两个分类方法主要基于
    - 在不同图嵌入的问题设置(problem setting)中遇到的困难
    - 解决这些困难所采用的技术
  - 概述图嵌入可以起作用的应用领域(application)
  - 提出几个建议的未来研究方向：
    - 计算效率(computing efficiency)
    - 问题设置(problem setting)
    - 技术和应用场景(techniques and application scenarios)

1.介绍(Introduction)

- 图这个数据结构的作用场景广大：社交图(social graph)、扩散图(diffusion graph)、引用图(citation graph)、用户兴趣图(user interest graph)。
- 图分析的作用：推动以下领域的发展：
  - 节点分类(node classification)
  - 节点推荐(node recommendation)
  - 链接预测(link prediction)
- 图分析的瓶颈：极高的计算成本和空间消耗，对于此，已经有许多研究工作着力于减少昂贵的图分析成本：
  - 分布式图数据处理框架(distributed graph data processing framework)
  - 新的可以有效提高I/O计算成本的高效空间图数据存储数据

    直接在图上进行机器学习是很困难的。图由边和节点组成， 数学、统计学和机器学习方法只能部分处理图数据，而在向量空间可以更充分的利用图数据。
- 图嵌入可以解决以上问题，原理是：在尽可能保留图信息的情况下将图转换为低维空间（比如表示为低维向量）。之后一些对于图的算法便可以更加有效率地进行计算

  关于为什么将图转换为低维可以提高效率，这里类比立体问题和平面问题，我们在求解立体问题的时候总要比求解平面问题要复杂，毕竟多了一个纬度。同样，在对图进行分析的时候，处理位于高维空间的数据要比处理位于低维空间的数据复杂得多
- 图的分类：
  - 同构图（齐次图）(homogeneous graph)
  - 异构图(heterogeneous graph)

    - 属性图(attribute graph)
    - etc

- 图嵌入的输入：根据图的种类和情景的不同而不同
- 图嵌入的输出：一个（组）低维向量，表示图的一部分或者全部
- 图嵌入的方式：
    - 节点嵌入(node embedding)
    - 边嵌入(edge embedding)
    - 子结构嵌入(substructure embedding)
    - 全图嵌入(whole-graph embedding)
- 图嵌入和两个传统问题有关：
    - 图分析(graph analytics)：图分析致力于从图中挖掘有用的信息（Q: more information）
    - 表征学习(representation learning)：表征学习获得的数据表示(data representation)将使得构建分类器或其他其他预测变量时更加容易提取

      关于表征学习
- 图嵌入会在这两个问题上有重合，但区别是图嵌入致力于从低维数据表示中学习。同时在这篇综述中区分图表征学习(graph representation learning)和图嵌入。主要区别是表征学习不需要数据是低维的。
- 图嵌入面临的主要挑战：
    - 问题设置(problem setting)：
        - 嵌入输入(embedding input):输入图(input graph)的种类有：
            - 同构图（齐次图）(homogeneous graph)
            - 异构图(heterogeneous graph)
            - 带有辅助信息的图(graph with auxiliary information)
            - 由无关联数据构成的图(graph constructed from non-relational data)

      不同的输入图需要保留的信息不同，给图嵌入带来了不同的挑战。比如当对带有结构信息的图进行嵌入时，需要保留的信息时节点之间的链接；当对带有辅助信息的图进行嵌入时，一些其他结构所带有的辅助信息是否要被保留也需要被考虑。

        - 嵌入输出(embedding output)：嵌入输出时任务驱动的，嵌入输出可分为4类:
            - 节点嵌入(node embedding)
            - 边嵌入(edge embedding)
            - 混合嵌入(hybrid embedding)
            - 全图嵌入(whole-graph embedding)

      不同的嵌入输出粒度（间隔尺寸）对于“好”的嵌入有着不同的评价尺度，同时也面临着不同挑战。

例如：一个好的节点嵌入应该保证相邻的节点具有比较高的相似性；一个好的全图嵌入应该保证在图级(graph-level)上具有较高的相似性。

- 对于图嵌入工作(graph embedding work)的两个分类方法，分别基于：
    - 问题设置
    - 嵌入技术（方法）

这两个分类方法对应了图嵌入面临的挑战和现有研究如何解决这些挑战。

- 现有研究工作的局限：
    - 他们通常只提出了一种基于图嵌入技术的分类方法，并且极少人从问题设置出发来分析图嵌入工作，也没有人对于图嵌入面临的挑战又一个较好的综述
    - 目前关于图嵌入所做的研究还是很少。他们主要是介绍了12种代表性的图嵌入算法，或者只是专注于图嵌入本身的知识。没有对于每种图嵌入技术背后的见解(insight)的分析

我们的贡献(Our Contributions)

我们的主要贡献如下：

- 提出一个关于图嵌入的基于问题设置的分类方法，同时对每种问题设置中面临的挑战做了一个综述。
- 提供了对于图嵌入技术的更深入详细的分析。与现有工作相比，我们：
    - 调查了更全面的图嵌入工作
    - 总结了每种图嵌入技术背后的见解
    - 对于见解的总结可以回答为什么可以以某种方式解决图嵌入的问题

- 系统地将图嵌入可以发挥作用的应用领域进行分类并将这些应用分为：
    - 节点相关(node related)
    - 边相关(edge related)
    - 图相关(graph related)

- 提出四种建议的未来研究方向：
    - 计算效率(computing efficiency)
    - 问题设置(problem setting)
    - 解决方案技术(solution techniques)
    - 应用(applications)

这篇综述的结构(Organization of The Survey)

- Sec2：基本概念和符号的介绍/图嵌入的正式问题定义/两个关于图嵌入的分类方法
- Sec3:基于问题设置比较相关的工作/对与问题设置中面临的挑战的一个概述
- Sec4:基于嵌入技术对文献（资料）进行分类/每种技术背后的见解(insight)/对于每种技术详细的比较
- Sec5:介绍图嵌入技术能应用到的领域(applications)
- Sec6:讨论四种我们建议的未来研究方向

- Sec7:对于这篇文章的一个总结

问题的形式化(problem formalization)

符号(notation)和定义(definition)

- 图的概念
- 齐次图：节点和边都是同一类比的图
- 异构图：节点集和边集的元素个数均大于一
- 知识图(knowledge graph)：其中节点表示实体，边表示连接的实体之间的关系，表示为，r表示h和t的关系。例如表示爱丽丝是勃德的朋友。因为这些实体可以是不同的种类（比如学生，工作者，无业者等），实体之间的关系也可以是不同的种类（比如血缘关系，工作关系等），所以知识图可以看成是关系图的一种特例（实例）。
- 节点之间的一阶相似度表示为连接节点的边的权重(A_{ij})。如果节点之间没有边直接相连，它们的相似度为0
- 连接两个节点的边的权重越大，它们的相似度越高。我们用$S_{ij}^{(1)}$来表示节点v_i和v_j之间的相似度，其实它的值等于边e_{ij}的权重。我们用$s_i^{(1)} = [s_{i1}^{(1)}, s_{i2}^{(1)}, s_{i3}^{(1)}, s_{i4}^{(1)}, \ldots, s_{i|V|}^{(1)}]$来表示节点$v_i$与其他节点的相似度。
- 节点之间的二阶相似度表示为两个节点的邻域节点之间的相似度。邻域节点之间的相似度越高，二阶相似度越大。(The **second-order proximity** between node v_i and v_j is a similarity between v_i's neighbourhood $s_i^{(1)}$ and v_j's neighbourhood $s_j^{(1)}$)

  关于余弦相似性简单来说便是通过测量两个向量的夹角的余弦值来度量它们之间的相似性

- 更高阶的相似度也可以按照这些规律推导下去，节点v_i和v_j的k阶相似度(the **k-th-order proximity**)可以表示为$s_i^{(k-1)}$和$s_j^{(k-1)}$的相似度。
- 图嵌入(graph embedding)。给定一个图G和一个预先给定的维度d(d << |V|)。目标是将图G转换到d维空间中，并尽量保留图的属性。图的属性可以用一阶相似度或者更高的相似度来衡量。嵌入的结果是每一个图都表示为一个向量，或者表示为一个向量集，其中每一个向量都表示嵌入之后图的一部分（比如节点、边、子图）

关于一阶相似度和二阶相似度一阶相似度通常代表着现实世界中两个节点的相似性，例如在社交网络中相互交友的人往往有着相似的兴趣；二阶相似度通常意味着一对节点的邻近网络结构之间的相似性，例如在社交网络中，分享相似朋友或者兴趣的人倾向于有相似的兴趣

图嵌入的问题设置(Problem Settings Of Graph Embedding)

图嵌入输入(graph embedding input)

这里用节点嵌入(embedding output)作为例子说明图嵌入输入，理由是目前大部分的图输入输出都是节点嵌

入。图嵌入输入是一个图，在这里分为四类：同构图，异构图，带有辅助信息的图，由无关数据构成的图(在这里不知道为什么是constructed graph?难道是为了省字数？)。下面介绍每一种图并总结目前面临的挑战。

同构图(Homogeneous Graph)

- 定义：节点和边都只有一种分类的图，进一步还可以有两种分类方法：
  - 有权图(weighted graph)和无权图(unweighted graph)
  - 连接图(directed graph)和非连接图(undirected graph)

- 无权图是最基本（简单）的图嵌入输入设置(graph embedding input setting)，对于此已经有很多的研究工作，他们将所有的节点和边都看成是等同的，他们只考虑图的基本结构信息
- 一方面，在有权图中，被具有较大权重的边连接的节点在嵌入的时候会靠近彼此；另一方面，一些工作在嵌入的过程中将边的方向(direction)加以区分，并将边的信息保存在嵌入空间中。

  这里的“将边的方向加以区分”应该是将连接图分为有向图和无向图

- 一个对于连接图的很好例子是社交网络图(social network graph)，每一个用户与其他的用户都拥有追随(followership)或者被追随(followeeship)的关系，但在这种情况下，边的权重变的不可用了。
- 一个新的算法被提出，区别是同时考虑边的权重和方向。换句话说，这个算法可以覆盖有权图无权图连接图和非连接图。
- **面对的挑战**：***如何捕获在图中观察到的多种连接方式？*** (How to capture the diversity of connectivity patterns observed in graphs?)因为在同构图中，只有结构信息是可用的（感觉跟前面有点矛盾？），目前所面临的挑战便是如何在图嵌入时保留在输入图中观察到的连接方式

  这里的“连接方式”指的是连接/非连接/加权/无权，根据需求不同来构建不同的输入图

异构图(Heterogeneous Graph)

异构图主要存在于一下三种情景中

- 基于社区的问答网站(Community-based Question Answering (cQA) sites)。一个cQA图中有多种节点：问题节点，回答节点，用户节点。目前对于cQA图的算法之间的区分点是他们利用的链接(exploit links)：
  - user-user, user-question
  - user-user, user-question question-answer
  - user-user, question-answer, user-answer
  - user's asymmetric following links, a ordered tuple(i, j, k, o, p)(表示对于问题i，第k个用户提供的答案i比第p个用户提出的答案o获得更多的投票（点赞）)

- 多媒体网络(Multimedia Networks)。多媒体网络会包含多种数据，例如图像，文本等。
- 知识图(Knowledge Graphs)。在知识图中，实体（节点）和实体之间的关系（边）通常都是不同类型的。关于知识图的嵌入算法在后面介绍。
- **面对的挑战**：

- **怎么探索图中不同对象(object)之间的一致性(consistency)**
- **怎么解决不同对象(object)之间的不平衡性(imbalance)**

> 对于“一致性”和“不平衡性”，文章中并没有过多的解释。

带有辅助信息的图

这里的辅助信息主要指三种：

- 标签(Label)
- 属性(attribute)
- 节点特征(node feature)
- 信息传播(information propagation)
- 知识库(knowledge base)

标签(Label)

具有不同标签的节点在嵌入的时候应该远离彼此。为了实现这个目标，不同的研究运用了不同的方法：

- [47]和[48]用一个分类器函数(classifier function)来对嵌入目标函数(embedding objective function)进行优化。
- [49]对剧具有不同标签的节点之间的相似度设置了一个惩罚项(penalty)
- [50]在计算不同图的内核的时候，将节点标签和边标签考虑进去。
- [51]和[52]嵌入一个知识图，图中实体（节点）具有一个语义类别(semantic category)
- [53]嵌入了一个更复杂的知识图，其中图节点基于层次结构而被分类。例如分类书(book)会有两个子分类作者(author)和作品(written_book)

属性(Attribute)

相比于一个标签，属性值可以是离散(discrete)的也可以是连续(continuous)的。

> 例如在一个分子的原子序数图中，将每一个原子看成一个节点，则原子序数作为原子的属性是离散的；在将时间作为节点属性的图中，则节点的属性值是连续的

节点特征(node feature)

大部分的节点特征是文本(text)，不是表现为特征向量(feature vector)就是被表现为文档(document)。对于后者（文档）,使用以下技术，对文档进行进一步处理以提取特征向量。

- 单词包(bag-of-words)
- 主题建模(topic modelling)
- 将“字”（word）作为一种节点类型

节点特征提高了图嵌入的效果，因为它提供了丰富和无条理(unstructured)的信息，这些在很多现实世界的图中都可以利用。更进一步节点特征使归纳图嵌入(inductive graph embedding)成为可 能。

信息传播(information propagation)

对于信息传播的一个例子是在推特(Twitter)上的“转推”(retweet), 给定一个数据图$G = (V, E)$，一个级联图$G^c = (V^c, E^c)$，其中V^c是采用(adopt)c的节点，E^c是V^c两端的边。之后可以通过嵌入这个级联图来预测级联大小的增量。

关于级联

知识库(knowledge base)

最流行的知识库有：

- Wikipedia
- Freebase
- YAGO
- DBpedia

拿Wikipedia作例子，“概念”(concepts)是用户提出的实体，“文本”(text)则是与该实体（概念）有关的文章。通过将每一个社交网络用户(social network user)链接到给定的一组知识概念(knowledge concepts)，便可以使用知识库从社交网络中学习社交知识图。

由无关联数据构成的图(Graph Constructed from Non-relational Data)

这种图是通过不同的策略根据非关系输入数据构建的，通常在假设输入数据位于低维流(low dimensional manifold)中时发生。

关于什么是低维流形：想象一个画着图形的白纸平铺在桌面上，我们可以很容易地辨认出该图形，但是如果将这张白纸揉成一团，我们辨认出该图形的难度便会大大增大。数据便是这个图形，当它处于高维空间的时候我们很难对它有一个直观的认识，提取特征的难度也大大增大

大多数情况下进行嵌入的时候输入的是一个特征矩阵(feature matrix)，其中每行X_i是第i个训练实例的N维特征向量。还有一个概念是相似度矩阵S，通过使用(X_i, X_j)之间的相似度计算S_{ij}来构造。有两种从S构造图的方法：

- 将S直接视为不可见图的邻接矩阵。但这种方法节点之间的距离是基于欧几里得距离的，并且在计算S_{ij}时不考虑相邻节点。因此如果X位于弯曲的数据流或者附近，则数据流上的X_i和X_j之间的距离远大于它们的欧几里得距离。
- 首先从S中构造一个K最邻近(KNN)图，并基于KNN图估计邻接矩阵A。这种方法可以解决上一种方法存在的问题。

构造图的另一种方法是基于节点的共现(co-occurrence)建立节点之间的边。比如在图像处理中，研究者通过将像素视为节点，像素之间的空间关系视为边来构造图。

面对的挑战

- **如何构造一个对实例之间的成对关系进行编码的图**
- **如何将生成的节点邻接矩阵保存在嵌入式空间中**

> 嵌入一个由无关联数据构成的图的第一步还是需要构建节点之间的关系，将问题转换为以上三种输入图的问题

图嵌入输出(Graph Embedding Output)

图嵌入的输出是一个（或者一组）表示（一部分）图的低维特征向量。图嵌入输出的分类有：

- 节点嵌入
- 边嵌入
- 混合嵌入
- 全图嵌入

图嵌入输出是任务驱动的，比如，如果一个图分析任务与各种节点紧密相关（例如节点分类和节点聚类），那么节点嵌入是一个合适的选择。

节点嵌入(node embedding)

节点嵌入将每个节点表示为低维空间的向量，并且图中“接近”的节点会用相似的向量表示。我们用一阶相似度和二阶相似度来衡量节点之间的相似度。

面临的挑战：

- **如何在各种类型的输入图中定义成对节点的相似度**
- **以及如何在嵌入中对相似度进行编码**

边嵌入(edge embedding)

跟节点嵌入类似，只不过边嵌入是将每条边表示为低维空间的向量。边嵌入在两个应用场景中用处颇大：

- 知识图嵌入(knowledge graph embedding)。在知识图中每一条边都是一个三元组$< h, r, t >$其中r是h和t的关系。用边嵌入来学习如何保留r，使得一个缺失的实体（关系）可以在知道h和t的基础上正确预测r。
- 一些工作嵌入一个节点对来作为向量特征，以使该节点对其他节点具有可比性，或者预测两个节点之间是否存在链接。比如将一个用户对和内容嵌入同一个空间，来预测给定该内容的条件下这对用户交互的可能性。

面临的挑战：

- **如何定义边的相似度**
- **如何对边的不对称性进行建模**

> 图嵌入需要保证相似的边在嵌入时彼此靠近，因此需要定义边的相似度；边的不对称性指的是边的两端连接的节点不一致，这可能会导致某种问题。

混合嵌入(hybrid embedding)

混合嵌入是嵌入不同类型图组件的组合，比如节点+边，节点+社区(comminity)

- 子结构嵌入(substructure embedding)。
- 社区嵌入(community embedding)

面临的挑战：

- **如何生成目标子结构**
- **如何在一个公共空间中嵌入不同类型的图形组件**

> 混合嵌入实质上是嵌入图的部分结构，因此如何决定嵌入图的哪个部分，以及如何用低维向量来表示这些图的子结构是需要解决的难题

全图嵌入(whole-graph embedding)

全图嵌入是将图表示为一个向量，并且两个相似的图在嵌入的时候会彼此靠近。应用场景是将整个图嵌入蛋白质，分子等小图的情况。

面临的挑战：

- **如何捕获整个图形的属性**
- **如何在表达性和效率之间进行权衡**

> 与其他嵌入类似，嵌入图的时候我们需要得到图的属性，而且这个相比于其他嵌入更耗时，成本更高，因此在表达性和效率上难以兼顾，如何获得图的属性，如何在表达性和效率之间进行权衡成为新的难题。

图嵌入技术(graph embedding techniques)

这一节主要介绍图嵌入的各种技术和每种技术背后的见解。

矩阵分解(Matrix Factorization)

用矩阵的形式来表示图属性，然后对这个矩阵进行分解以实现节点嵌入。

矩阵分解的两个方法：

- 分解拉普拉斯特征图(graph Laplacian eigenmaps)
- 直接分解节点邻接矩阵(the node proximity matrix)

分解拉普拉斯特征图(Graph Laplacian Eigenmaps)

见解(insight)：要被保留的图属性可以被解释为成对节点的相似性，因此如果相距较远的两个节点的相似度较大，则会施加更大的惩罚

基于这个见解，可以提出一个最优的嵌入y模型：
$y^* = argmin \sum_{i \neq j}(y_i - y_i^2 W_{ij}) = argmin y^T L y$（1），其中$W_{ij}$用来定义节点i和j之间的相似度；L = D - W是图拉普拉斯算子；D是对角矩阵：$D_{ii} = \sum_{j \neq i} W_{ij}$，$D_{ii}$越大，$y_i$越重要。另外，通常对方程(1)施加约束$y^T D y = 1$来删除嵌入中的任意缩放比例(arbitrary scaling factor)。因此方程(1)可以等价为：
$y^* = argmin_{y^T D y = 1} y^T L y = argmin \frac{y^T L y}{y^T D y} = argmax \frac{y^T W y}{y^T D y} (2)$
代表最优的y是对应于特征问题$W_y = \lambda D_y$的最大特征值的特征向量。
上述的图嵌入是可转导(transductive)的，因为它只能嵌入训练集中的节点。在实践中可能还需要嵌入在训练中从未见过的新节点，因此提出了一个解决办法：
设计一个线性函数$y = X^T a$以便只要提供节点特征就可以导出嵌入。然后方程(1)就变成了在下面目标函数中找到一个最优的a：
$a^* = argmin_{y^T D y = 1} y^T L y = argmin \frac{y^T L y}{y^D L y} = argmax \frac{y^W L y}{y^D L y} (3)$
跟方程(2)类似，加一个约束条件$a^T X D X^T a = 1$，则方程(3)等价于：
$a^* = argmin \frac{a^T X L X^T a}{a^T X D X^T a} = argmax \frac{a^T X W X^T a}{a^T X D X^T a}$
最优a是求解$X W X^T a = \lambda X D X^T a$时具有最大特征值的特征向量

节点邻接矩阵分解(Node Proximity Matrix Factorization)

见解(insight)：可以通过矩阵分解在低维空间中近似节点相似度，保留节点临近性的目的是使近似损失最小化。
给定一个节点相似矩阵W，则目标是：
$min||W - YY^{cT}||(5)$
其中Y是节点嵌入，Y^c是相邻节点的嵌入。方程(5)的目的是找到邻接矩阵W的最优秩-d近似值（d是嵌入的维数）。一种解决方案是使用SVD（奇异值分解(Singular Value Decomposition)）：
$S = \sum_{i=1}^{|V|} \sigma_i u_i u_i^{cT} \approx \sum_{i=1}^{d} \sigma_i u_i u_i^{cT}$
其中$\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_|V|$是以降序排列的奇异值，$u_i$和$u_i^c$是$\sigma_i$的奇异矩阵。使用最大的d奇异值和对应的奇异向量（如下所示）可以得到最优的嵌入：
$Y = [\sqrt{\sigma_1} u_1, \ldots, \sqrt{\sigma_d} u_d]$
$Y^c = [\sqrt{\sigma_1} u_1^c, \ldots, \sqrt{\sigma_d} u_d^c]$

根据是否保留不对称属性，节点i的嵌入取$y_i = Y_i$或者$y_i = Y_i^c$。

总结(summary)：矩阵分解(MF)通常用于嵌入由无关联数据构成的图，这也是拉普拉斯图问题的经典设置。另外，MF通常也用于嵌入齐次图。

深度学习(deep learning)

嵌入的输入可以是从图中采样得到的路径，也可以是图本身。根据游动(random walk)是否被用于从图中采样路径，将DL分为两类：

- 基于DL的随机游动图嵌入(DL based Graph Embedding with Random Walk)
- 基于DL的无需游动的图嵌入(DL based Graph Embedding without Random Walk)

基于DL的随机游动图嵌入(DL based Graph Embedding with Random Walk)

见解(insight)：通过最大化观察到以其嵌入为条件的概率，可以将图的二阶接近度保留在嵌入空间中

基于DL的无需游动的图嵌入(DL based Graph Embedding without Random Walk)

见解(insight): 多层学习架构是将图编码到低维空间的强大且有效的解决方案

应用(Application)

图嵌入的应用主要分为三大类：

- 节点相关
- 边相关
- 图相关

节点相关(Node Related)

节点相关的应用可分为：

- 节点分类
- 节点聚类
- 节点推荐/检索/排名

节点分类(Node Classification)

节点分类是基于从被标记节点中学到的规则，为图中的每一个节点分配一个类别标签。
在节点分类中，“相似”的节点具有相同的标签。节点分类是通过在标记的节点嵌入集集中应用分类器进行训练来进行的。分类器主要有：

- 逻辑回归(logistic regression)
- k最近邻分类(KNN)

在嵌入未标记的节点时候，分类器可以预测其类别并进行标记。

> 将图嵌入到低维平面之后，便可以使用KNN等机器学习算法来进行节点分类，因为此时数据处于低维流形中，所以计算成本得到了极大的降低

节点聚类(Node Clustering)

节点聚类的目的是将相似的节点分组到一起，以便使同一组的节点比不同组的节点更加相似。目前大部分工作都是使用K均值(K-means)作为聚类算法。

节点推荐/检索/排名(Node Recommendation/Retrieval/Ranking)

节点推荐的任务是根据某些条件（例如相似性）向给定的节点推荐感兴趣的前K个节点。例如：

- 研究人员的研究兴趣
- 顾客用品
- 社交网络用户的朋友

边相关应用(Edge Related Applications)

这里主要介绍下面两种应用：

链接预测(Link Prediction)

在现实情况中，图通常是不完整的。例如在社交网络图中，彼此认识的两个用户之间的友情链接可能会丢失。图嵌入旨在用低维向量表示图，并且输出向量(output vector)也可以帮助推断图结构。这些向量可以用来预测不完整图中的丢失链接。

> 同样，将图嵌入到低维平面中，训练模型的计算成本大大降低，进行预测的效率也得到了提高

三元体分类(Triple Classification)

三元体分类是知识图的特定应用，旨在对看不见的三元组是否正确地进行分类，即h和t的关系是否是r。

图相关应用(Graph Related Applications)

这里也主要介绍下面两种应用：

图分类(Graph Classification)

图分类与节点分类类似，只是图分类是给整个图设置标签。

可视化(Visualization)

在低维空间生成图的可视化。将所有节点嵌入为2D向量，然后将其绘制在2D空间中，并用不同的颜色指示节点的类别。

其他应用

- 知识图相关(Knowledge graph related)
- 多媒体网络相关(Multimedia network related)
- 信息传播相关(Information propagation related)
- 社交网络相关(Social networks related)
- 图片相关(Image related)

未来方向(Future Direction)

总结了图嵌入领域的四个未来方向：

计算(computaion)

采用几何输入（比如图形）的深度架构的效率非常低下。传统的深度学习模型需要利用现代GPU通过假设数据位于1D或者2D网格上来优化其效率。但是图不一定具有这种网格结构，因此设计用于图嵌入的深层体系结构需要寻求替代解决方案以提高模型效率。

问题设置(problem setting)

动态图(dynamic graph)是图嵌入中的一个有前途的设置(setting)。在现实生活中图并不总是静态的，例如推特上的社交图和DBLP上的引文图。图是动态的可能会导致以下问题：

- 图结构可能会随着时间而改变（发展）：出现新的节点或者旧的节点会消失
- 某些边和节点是通过时变信息来描述的

与静态图不同，用于动态图的技术需要具备伸缩性，并且最好具备增量性，才能有效地处理动态变化。这使得现有的图嵌入算法不再适用。因此如何在动态域中设计有效的图嵌入算法仍然是一个悬而未决的问题

> 关于伸缩性：动态图中点和边的信息都随时间在不断变化，因此系统也需要具备对此做出调整的能力，这便是伸缩性。例如如果图中节点或边的数量增加，系统也需要相应增加吞吐量

技术(techniques)

结构感知(Structure awareness)对于基于边重构的图嵌入非常重要。当前基于边重构的图嵌入算法主要也仅仅基于边，缺少基于图全局结构的算法。并且从直观上，一个子结构包含的信息要比单个边提供的信息更加丰富。为提高效率，需要一种有效的结构感知图嵌入优化解决方案以及子结构采样策略。

应用(application)

图嵌入已应用于不同的应用程序中。它可以将来自不同源/平台/视图的数据实例转换为一个公共空间，以便它们可以直接比较。图嵌入为很多常规问题提供了有效的解决方案。