图神经网络：

社交网络中，人是顶点，人和人之间的关系是边，人/顶点的属性比如年龄、性别、职业、爱好等构成了一个向量，类似的，边也可用向量来表示。微信好友是双向的，微博关注的单向的。深度学习最大的优点恐怕是“端到端”(End-to-End)。所谓端到端，是指将数据从模型的输入端灌进去，预测结果从模型的输出端输出来，中间无需任何人工介入

节点特征的表达学习：受word2vec启发，我们希望训练一个f，输入一个图，输出该图所有节点的嵌入向量。同时要求：在图中比较邻近的节点，所得到的向量之间的距离也比较近。

对于从图的节点映射到向量表达的映射时有一定要求：在图中比较相似的两个节点 u，v ，函数ENC将节点 u和v 映射到(低维的)d维嵌入空间的向量 zu，zv: zu=ENC(u),zv=ENC(v)，这两者也应比较接近。两个向量之间的相似性可用余弦(cosine)相似度 zvT \* zu来表示(已归一化为单位向量)，问题在于：如何定义节点间的相似度呢？边提供了一种度量相似性的思路：存在边连接的两个节点比较相似，如专业社交网络(LinkedIn)中，你和你的同事相似度较高。

图的要求是置换不变性，不是图像的平移不变性和旋转不变性

消息传递：多轮消息传递的本质是多层汇聚，是更新节点的重要手段，类似于多层卷积。对于一个网络，每传递一次消息，消息就传播到离初始节点更远一步范围，GNN就增加一层，源于不同节点的消息经过汇聚(相当于CNN的卷积)，再进行多层的组合(相当于CNN多个卷积层的叠加)，对节点进行更新，本质上与CNN的思想是一致的。但是CNN不具有置换不变性，像素交换可能导致不同的输出结果。

CNN可以视作为一类特殊的GNN，相邻节点大小和顺序固定的GNN

下图中第0层(Layer-0)节点的嵌入向量是特征向量自身( XA,XB,XC)；第1层节点(Layer-1)的嵌入向量来源于其直接邻居的融合(Layer-0节点的节点嵌入)，也就是距离为1跳(1-hop)的节点嵌入向量融合的结果，比如：节点B的嵌入向量是节点A和C嵌入向量融合的结果；以此类推，第 k层节点的嵌入向量源于k跳(k-hop)之外的节点。

图表, 图示

描述已自动生成

第0层节点 v 的(隐 hidden)嵌入向量 hv0 即节点的特征向量 xv 。

第k+1层网络的节点 v的(隐)嵌入向量 hv(k+1)是两个部分的线性组合后再进行[非线性变换](https://www.zhihu.com/search?q=%E9%9D%9E%E7%BA%BF%E6%80%A7%E5%8F%98%E6%8D%A2&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A%22463666907%22%7D" \t "_blank) σ(如relu)的结果: 第1个部分是第 k层节点 v所有相邻节点 u嵌入向量的均值( |N(v)|指 v的相邻节点的数量)乘以汇聚权重 Wk，第2个部分是第 k层节点 v自己的嵌入 hv(k)乘以系数 Bk。即: hv(k+1)可理解为其相邻节点均值的汇聚(上图中灰色箭头)和自身的变形(Transformation，上图中棕色的指向自身的箭头)的线性组合再进行非线性变换 σ的结果。

汇聚权重 Wk和 Bk可通过训练得到——将这些嵌入输入损失函数，通过SGD得到。

第3个式子表示，经过 K层汇聚，最终得到我们所要求的节点 v的嵌入 zvzv

图示

中度可信度描述已自动生成