我今天分享一篇NeurIPS 2021上一篇和超图有关的论文。首先文章在背景部分提到了，目前大多数的GNN更关注于节点的表示，而边的表示与节点表示相比不那么重要，比如说我们在考虑一张图的表示的时候，比较普遍的方法是用节点去做pooling，而很少会把边去做pooling，虽然在信息传递的阶段考虑了边的信息，但作者认为边的表示最终还是服务于节点的表示。但在一些情况下，边的表示也很重要，比如说在重构一张图或者生成一张图的时候，不仅要还原节点还需要还原边，所以作者的motivation就是希望克服现有的GNN方法对于边表示的不足之处，所以它采用了将图与超图之间互相转化的方法来学到更为丰富的边的表示。不仅如此，他们认为单纯地将边表示地更好是不够的，所以他们还提出了两种新的pooling方式，一种是cluster方式，在pooling的时候将相似的边聚合为1个，另一种是drop方式，在pooling的时候将不相关的边丢掉，这两种pooling方式分别适用于不同的任务场景，在之后的任务里会讲到。

然后首先看一下它把图与超图之间进行转化的方式，首先的话，不管是超图还是图都是可以用三个矩阵来表示的，分别是节点矩阵、边矩阵、以及节点和边之间的关系矩阵，那么在把图转化为超图的过程中就相当于把原来图中的节点变成超图中的边，把原来图中的边变成超图中的节点，就有点像把原始图里面节点和边的角色互换了，这样的话我们在超图上使用一些经典的GNN方法的话，就可以更好地学习到边的表示了。在文章里他们对于超图上的信息传递是没有做什么手脚的，直接使用了GCN、GAT、GraphSAGE这些经典的GNN方法来做信息传递，在这篇文章里它使用的是GCN来做超图上的信息传递。

然后除了对边进行更好地表示之外，它们还提出了两种pooling方法。首先第一种pooling方法是对于相似的边表示做cluster，但是具体怎么衡量边之间的相似性在正文部分其实没有明确说，在附录里也写得比较简单，在附录里面他提到了他们的方法参考了GMPool的方法，看起来有点像transformer，用自注意力去算的，但具体怎么做的在这篇文章里都没有提到，可能需要去看一下它所使用的方法的原文。但他最后做的事情就是把超图里面相似的边聚合到一起，作用到原图的话相当于把原始的相似的节点给压缩到一起了，然后作者认为这种pooling方法在图的重构任务中会比一般的pooling方法更好。

然后还有一种pooling方法是drop方法，他做的事情就是对hypergraph中的所有节点去算一个分数，相当于对原始图里的所有边去算一个分数，然后选取出topk个，将这topk个作为这张图的表示，然后score的计算方法的话也是用自注意力方法去算的。这么做的话，对于原始的图来说，节点的数量是不会变的，但是边会变得更稀疏，作者认为它可以抛弃一些原始图中不重要的边，然后作者认为这样更适用于图分类或者节点分类的任务中，并且也可以解决over-smooth的问题。

然后是他们的实验部分，然后他们首先做的是图的重构任务，然后重构任务也被分为两类，一类是边的重构，另一类是整张图的重构。首先在边的重构任务里，节点是被固定的，模型需要还原边的类型，首先在训练的时候会先把原始的图转化为超图，然后在超图上使用GCN，然后使用cluster方法进行pooling，相当于是一个encoder，把超图上的边进行聚合压缩，然后使用一个decoder进行还原，训练的时候需要minimize原来的边特征和模型还原的边特征之间的误差。然后fig3是在zinc数据集上做得，就是和不使用超图的方法相比的话，使用超图训练得到的准确率会高得多，fig4是可视化的结果，超图得到的结果和原来的会更像。

除了边的重建任务之外，他们还做了一整张图的重构任务，既需要还原边也需要还原节点，结果和普通的方法相比也是比较好的。

然后还有图的生成任务，然后这个任务的话它是基于另外一个模型做的，叫做MolGAN，这个任务他本来是用R-GCN做的，但是他把R-GCN换成了他们的超图，最后的话效果也是比原来的更好。

后面又做了节点分类和图分类的任务，效果的话个别数据集上有比较大的提升，其他的效果和之前的方法比起来是相当的。