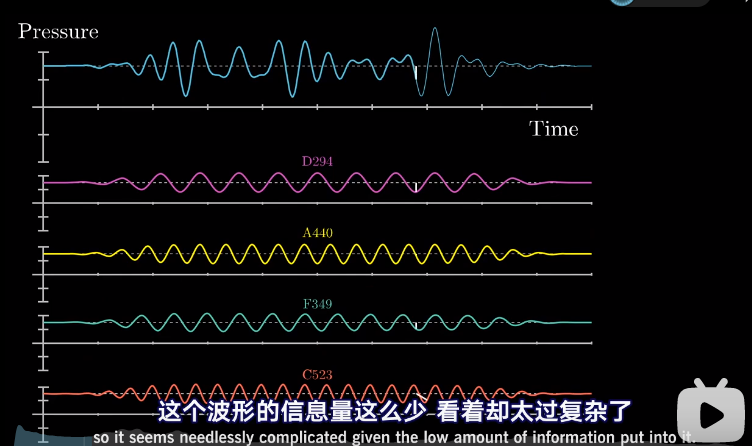
傅里叶变换

[【官方双语】形象展示傅里叶变换\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/BV1pW411J7s8?from=search&seid=7405866852400114612)

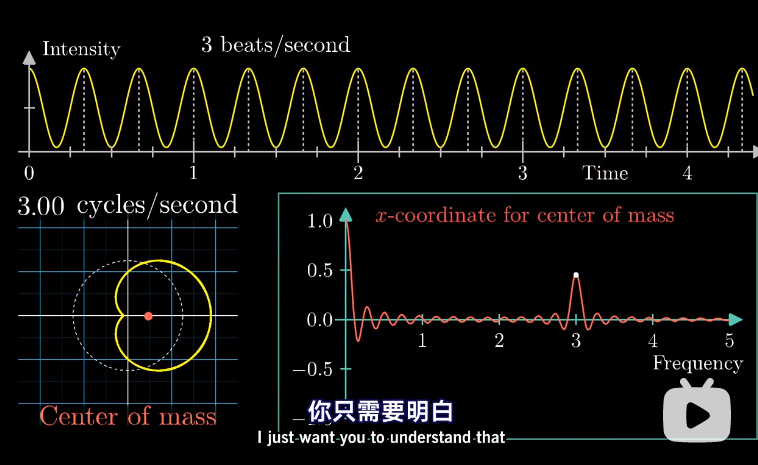
声波的周期与频率

四个纯音的波在空气中叠加，最后展示的是个复杂的波形



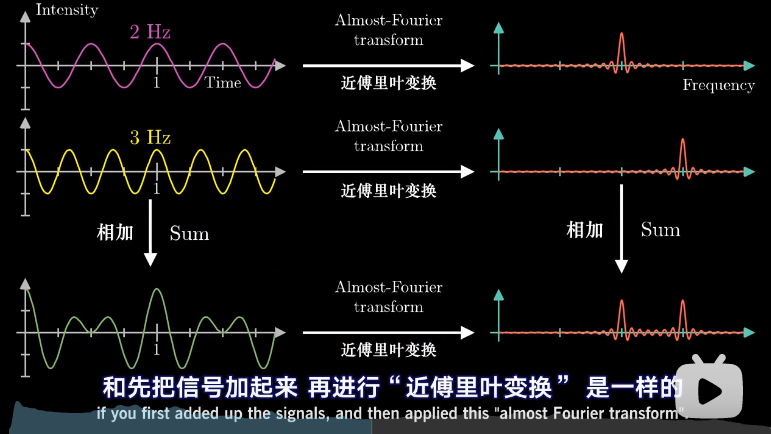
问题关键是，我们只能观察到最后的总和形式，需要通过它得到分别的原始信号

频率分离机



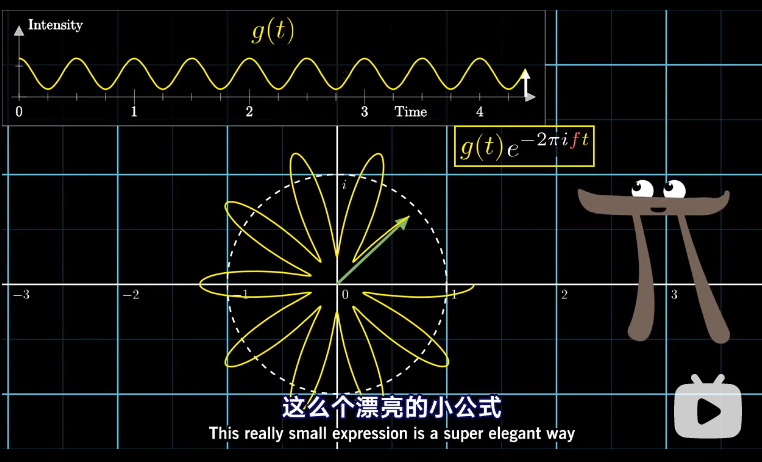
注意，右下图零点附近的突出值，只是由于原始波形的上下平移导致，并不会影响找出频率的点。

再考虑波的叠加



通常在描述在二维坐标平面的运动时，就会用到复平面（Complex plane）

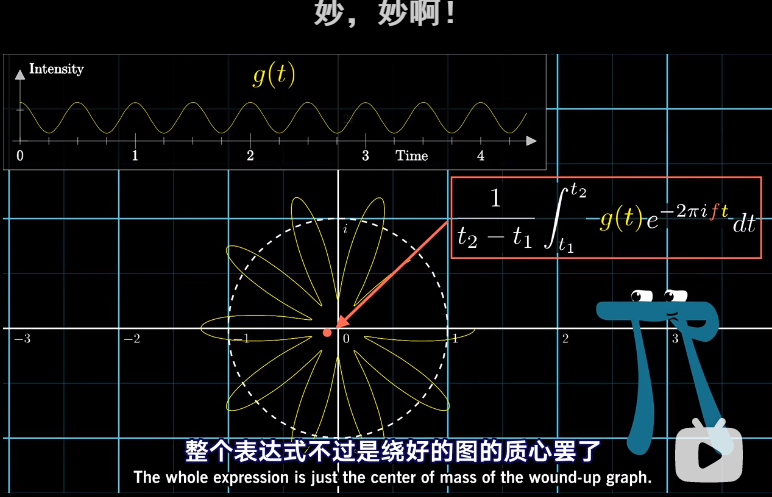
再根据欧拉公式，就可以给出圆周周期运动的数学表达式。



e指数部分的负号：顺时针运动；f：频率，如果是1/10就表示10秒钟转一圈。

g(t)：表示了波形的变化。

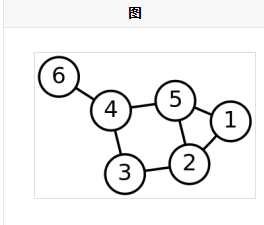
这样，和前面的联系起来，可以根据新的到的这个漂亮小公式进行积分求和，来计算图像的质心。



真正的傅里叶变换是如下公式，没有除以时间，表示随着时间增加，会被放大。



图（Graph）的小知识



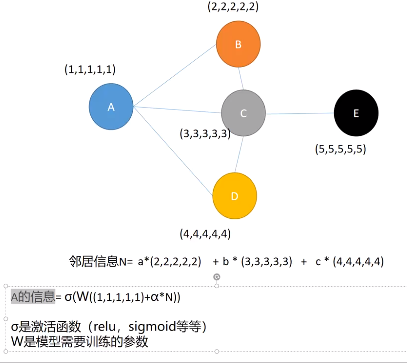
GNN

聚合：

聚合邻接点的信息，可以根据不同邻接点的对主节点的贡献不同，来设置不同的权重。

更新：

将邻居的信息聚合后贴到它本身的信息上，作为一个补足。

、

↑是一个结点经过一次GNN的操作大致流程。α是超参数，可以人为设置。

多层：

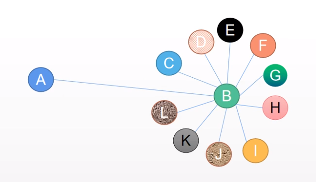
第一次聚合更新，只能使一个结点聚合到其直接邻接点的信息。多次聚合更新，就能收到更远结点（多接邻接点）的信息了。

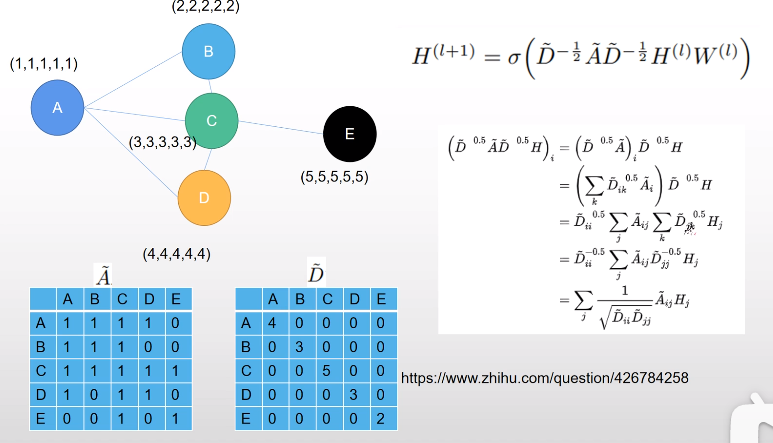
GCN：

[简单粗暴带你理解GCN图卷积神经网络\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/BV1Xy4y1i7sq?from=search&seid=12407523432725020180)

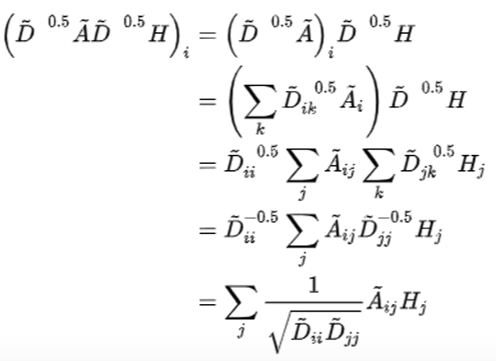
在聚合的步骤中，与GNN操作不同。

因为一个结点A的邻接结点B与其本身可能差距很大（极端情况，A只有B一个邻接结点，B有很多其他邻接点，B的特征加到A上就会使A的特征偏离很多）

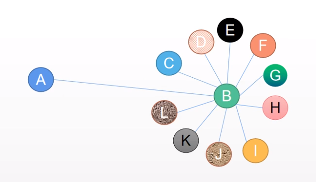




D为度矩阵，可见，就是对角线上是邻接矩阵每一行的和。

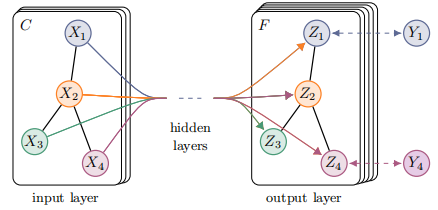


从结果可以看到，聚合A的信息时，也会考虑到与其邻接的结点B的度矩阵信息（B结点的邻接点的多少）。



通俗理解就是，B结点的邻接结点这么多，B对它们每一个的影响也会变少。

Semi-supervised GCN论文阅读



实际上就是特征提取的方法，通过对邻接矩阵和度矩阵操作；

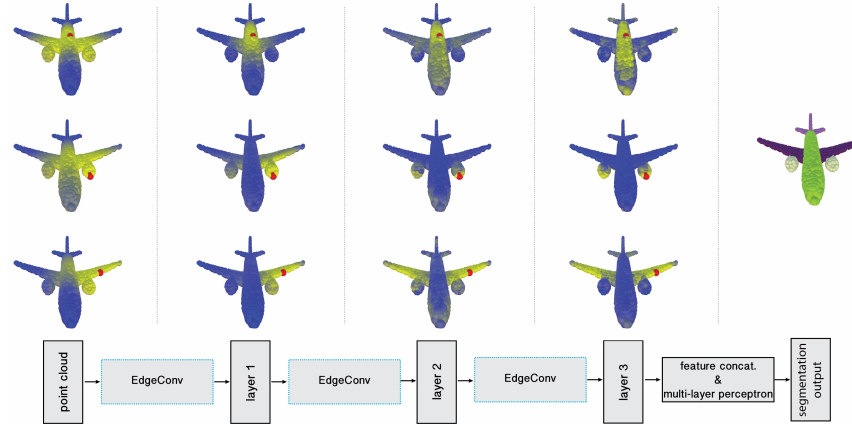
卷积体现在了图傅里叶变换，就是前面的那个几个矩阵相乘。

[图傅里叶变换 - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/147687999)；

DGCNN

<https://blog.csdn.net/hongbin_xu/article/details/85258278>

<https://zhuanlan.zhihu.com/p/267895014?utm_source=qq>



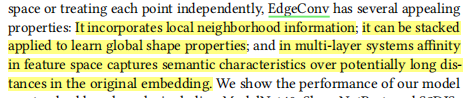
Schematic neural network architecture ↑



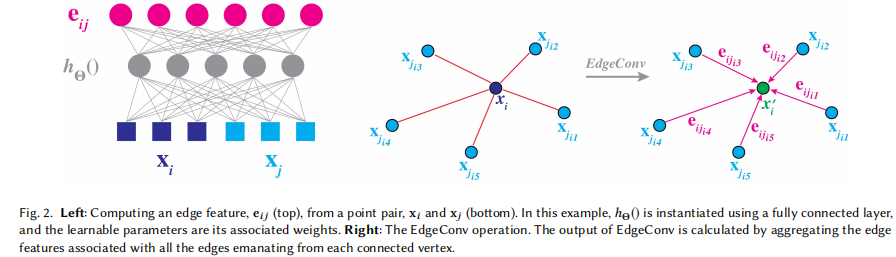


=>点云缺少拓扑信息（也就是单个点与点之间的关联并没有显式建立，但是他们之间应该是有实际意义的）

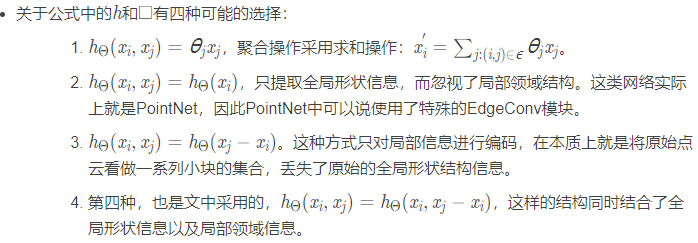
=>使用EdgeConv

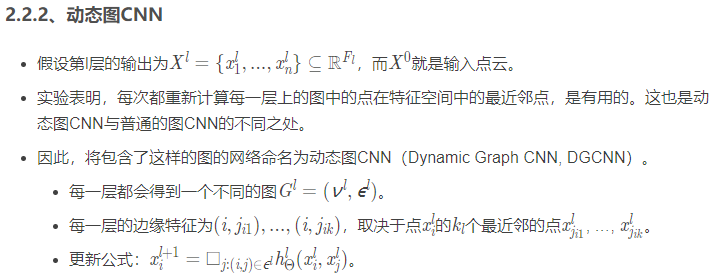


似乎是寻找edge的特征？



Right Figure: 将相连接点的信息集中在中心结点上，是对中心结点特特征的扩充。





**类似于GCN，不过GCN在选取的邻接点上是一直固定的，而DGCNN是动态的，在每一层的更新特征后，会重新计算相邻结点。（是这样的吗）**

**ConvEdge layer**：

Input tensor n \* F. F = 3 stands for coordinate information(x,y,z)；

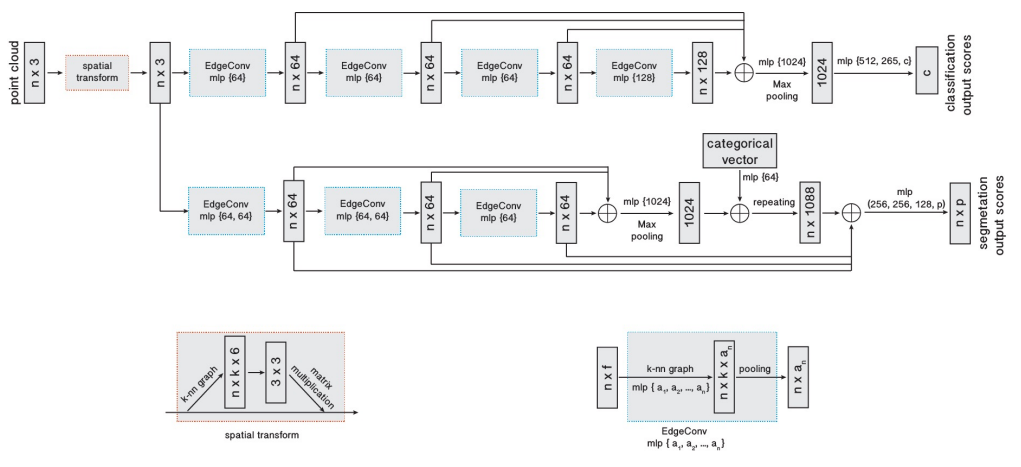
计算出Graph G = (V,E);G表示局部信息，通过KNN方法建立邻接矩阵，每个节点与其最近的k个结点相邻。

ConvEdge表示为公式:

Xi 为中心结点，Xj 为中心结点附近的结点们，

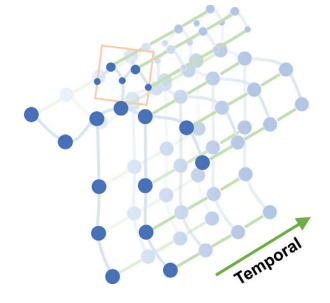
函数h实际上就是包含了Xi本身的信息（全局信息），和Xi与Xj的相邻信息（局部信息）的函数。（输出的特征维度F‘由函数h决定）

正方形函数是一个通道级的对称聚集函数操作，就是为了整合各个相邻点通过函数h后得到的信息。



相比PointNet。前面特征提取的方法不同。后面分类和分割部分一致。

**STG-CNN**

和用2d或3d栅格的方法不同，这是用图来表示。

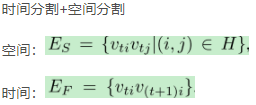
单独的joints不再分离，而是通过edge相连接，空间上和时间上。



由于等级性质，减少了手动标记和人工设定运动模式的需要。

**Skeleton Graph Construction**

edge set E:

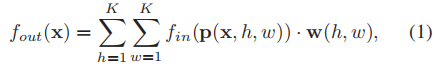


**Spatial Graph Convolutional Neural Network**

<https://www.cnblogs.com/wangxiaocvpr/p/8371660.html>

用一个时间点的Frame的2d卷积做例子

对于一个K x K 的kernel：



x是选择的一个点，p函数是采样函数，表示了x附近的点。

W（）是可训练的权重函数，由p采样出来的所有附近点共享。

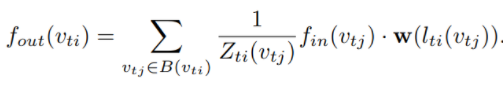
**图卷积**中类似

定义采样函数p是与xi直接相邻的结点（距离为1）。

邻近结点定义为：

B(vti) ={vtj |d(vtj , vti) ≤ D}

定义权重函数w，是先将xi 的邻近结点分类为k个子集，然后对k个子集进行对应。

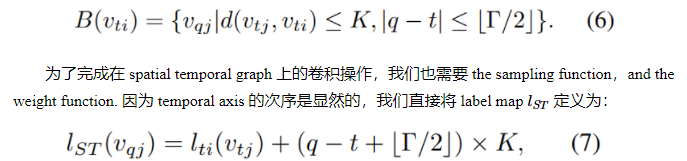


l是一个索引函数，将vi的每一个邻近结点对应k个子集中的一个。

the normalizing term Z 等于对应子集的基数，也就是一共有多少个邻近结点与vj属于同一个子集。

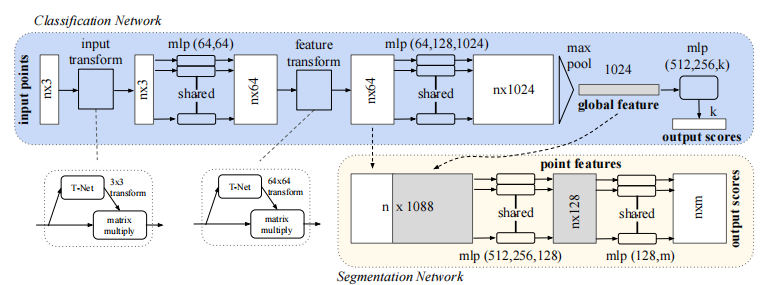
**Spatial Temporal Modeling**

将时间引入采样函数和分子集的函数。



821点云小总结（分类分割）

Pointnet





1. γ和h都是MLP层。
2. h可以理解为，对一个point从低维空间到高维的映射（spatial encoding），实现了特征扩充。由于MLP特点，所有points将会按照统一规则进行映射，所以不会改变点与点之间的关系。

两部分：分类和分割。

解决unorderd input无序性：Symmerty function；最大池化。

解决几何空间spatial transform变化：使用T-net。

分割网络中：

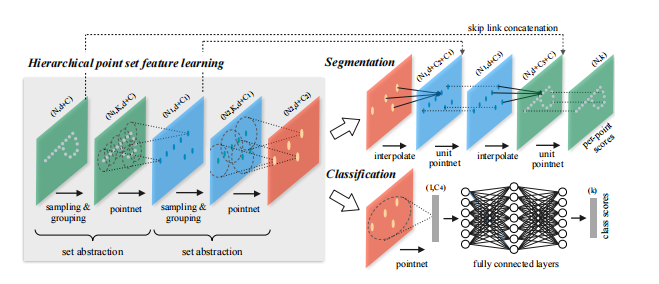
考虑局部信息和全局信息：通过将global feature与前面的local feature结合。

Pointnet缺点：没有局部信息的上下文联系no local context，limited translation invariance,

Pointnet++

总括：采用hierarchical structure利用mini-pointnet作为特征提取器来实现分类分割。

细节：

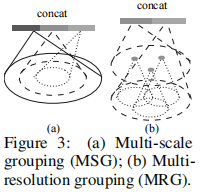


Sampling layer：FPS，最远点采样法；选出关键结点。

Grouping layer：Ball query ,K临近算法。以关键结点为中心，选取邻接k个点作为neighbour。

PointNet layer：对每一个关键结点的neighbour(包括了关键节点本身)进行mini-pointnet，提取出特征。

hierarchical structure：以上结构，反复提取。该方法被称为Multiresolution grouping (MRG)相比于 Multi-scale grouping (MSG)，可以减少很多计算量。



GNN图神经网络

聚合：

聚合邻接点的信息，可以根据不同邻接点的对主节点的贡献不同，来设置不同的权重。

更新：

将邻居的信息聚合后贴到它本身的信息上，作为一个补足。

多层：

第一次聚合更新，只能使一个结点聚合到其直接邻接点的信息。多次聚合更新，就能收到更远结点（多接邻接点）的信息了。

GCN图卷积（论文：SEMI-SUPERVISED GCN半监督图卷积分类）

在**聚合**的步骤中，与GNN操作不同。

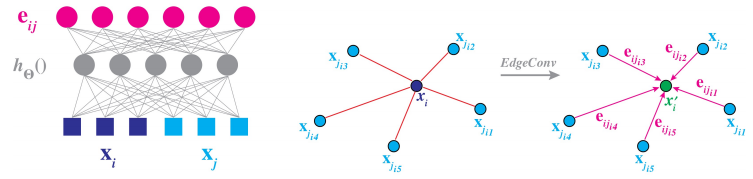
因为一个结点A的邻接结点B与其本身可能差距很大（极端情况，A只有B一个邻接结点，B有很多其他邻接点，B的特征加到A上就会使A的特征偏离很多）；

解决方法：通过考虑A、B结点度矩阵的信息，来计算出“权重”，来聚合邻接点的信息。

DGCNN(: Dynamic Graph CNN)

引入了EdgeConv block。相比于GCN，该网络的改进是，会更新对于邻近点的选择。

EdgeConv如此描绘edge的形成。



EdgeConv的输入输出：

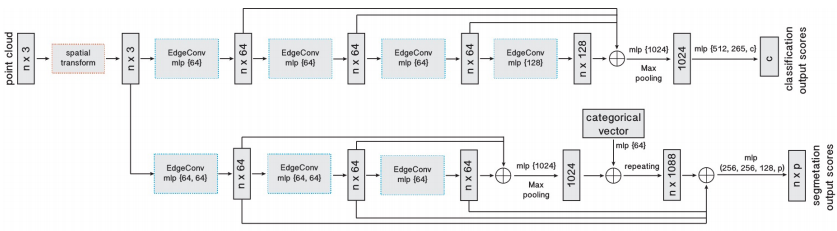


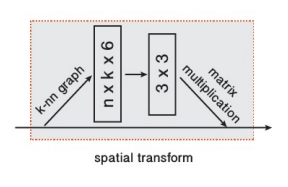
Xi 为中心结点，Xj 为中心结点附近的结点们，

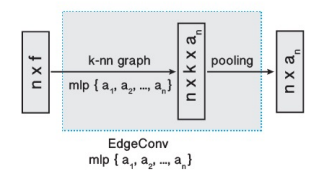
函数h实际上就是包含了Xi本身的信息（全局信息），和Xi与Xj的相邻信息（局部信息）的函数。（输出的特征维度F‘由函数h决定）

正方形函数是一个通道级的对称聚集函数操作，就是为了整合各个相邻点通过函数h后得到的信息。

Dynamic的体现：将会根据结点新的特征，重新选择相应的neighbour，也就是选择相应新的相邻结点。(也就是多个EdgeConv的话，就是迭代选取neighbour)

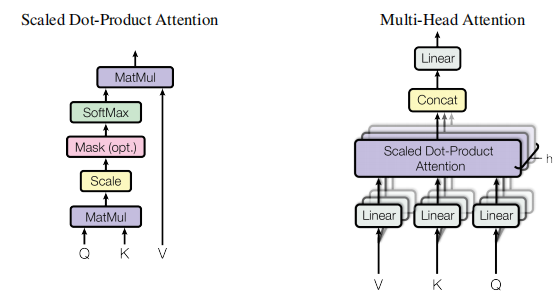


通过knn得到邻接点信息来寻找（学习）一种变化模式（矩阵），来对输入点进行空间变化。

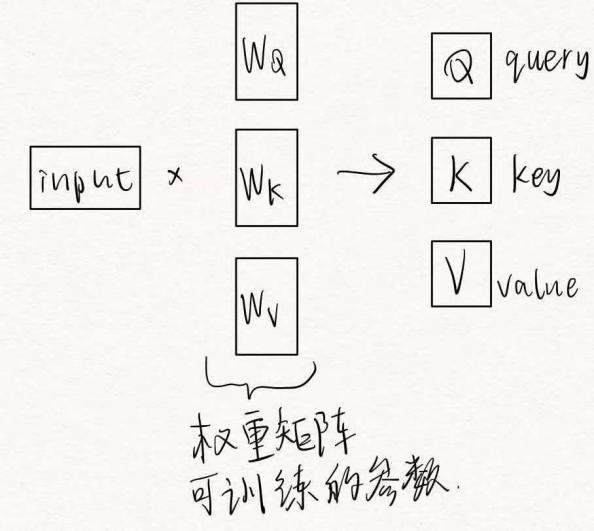
由于h函数的性质，可以控制特征通道的维度。再通过pooling，来缩减参数数量，聚合为一个结点的新特征。

Attention Is All You Need论文：注意力机制引入

Attention部分：



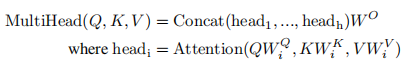
这样来描绘QKV的生成过程：



Attention的公式描述：

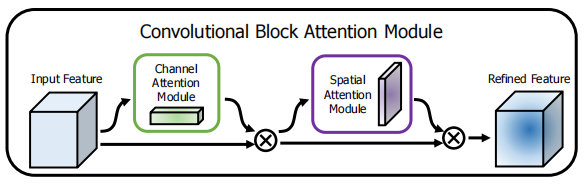
，其中V前面的矩阵是权重矩阵（注意力机制的体现）。

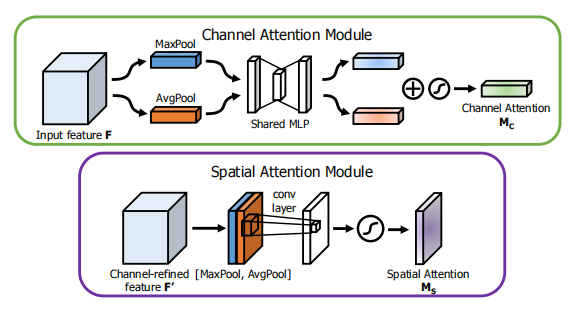
Multi head：

，并行计算，新增了为QKV新增了可训练权重参数W（我是这样理解的）。

CBAM: Convolutional Block Attention Module

总结：结合了通道注意力和空间注意力。





点云的卷积通过GCN来完成。