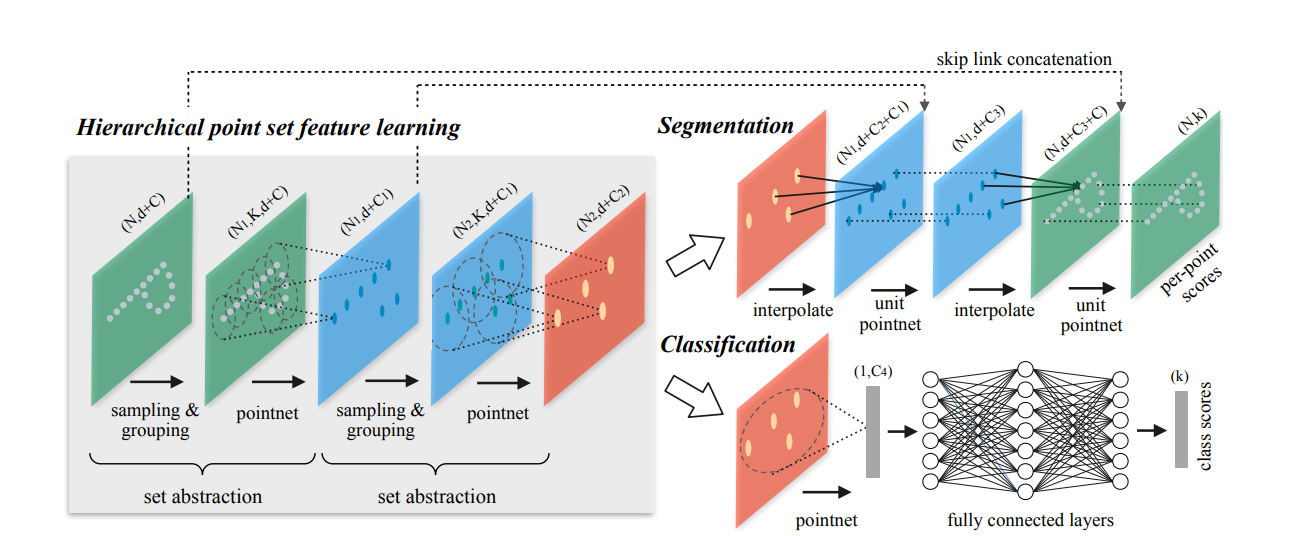
PointNet++

기존 pointnet에서, point가 존재하는 metric space의 정보를 local structure를 학습할 때 반영하지 못한 점을 보완하여 작은 pattern에 대한 인식 능력이나 complex scene에 대한 generalizability를 높이고자 함. Pointnet++에서 이를 input point set을 nested partitioning(NP, global optimization을 위한 방식)을 하고 이 구조에 pointnet을 recursive(순환)하게 적용하는 hierarchical neural network(계층적 신경망, 유향비순환 그래프 형태를 띠는 여러 네트워크의 구성)를 도입하여 해결함. Metric space의 distance 정보를 활용했기에 contextual scaling(문맥적 스케일링)이 향상된 local feature를 학습할 수 있다고 하며 density(밀도)에 invariant(불변)하도록 하기 위해 여러 scale의 feature들을 adaptive(적응)하게 결합하기 위해 새로운 set learning layer 또한 제안함.

여러 scale에서의 neighborhood를 이용해 robustness(튼튼함)와 detail한 학습을 이뤄냄

training에서 network가 random input dropout을 통해 adaptive하게 다양한 scale에서 얻은 pattern weight을 주고 결합하도록 학습시킴



Set abstraction은 pointnet++에서 point cloud의 local feature를 추출하기 위해 사용한 방법. Point cloud가 이 방법을 거치면 전반적인 semantic(의미론의) 정보를 포함한 압축된 point cloud로 변한다. Set abstraction layer는 세 가지 layer로 구성됨. 가장 처음 sampling layer에서는 input point cloud에서 중심이 되는 몇 개의 점들을 선택. 이 점들은 이전 point cloud의 대표성을 잃지 않아야 하며 local한 공간의 중심이 되어야 한다. 그 다음 grouping layer에서는 sampling layer에서 찾은 각 중심점에 대해 이웃한 점들을 찾고 이를 묶어 하나의 점묶음으로 구성. 이후 pointnet layer에서는 각 점묶음에 포함된 점들의 pattern을 파악하여 점묶음별로 feature vector를 추출한다.

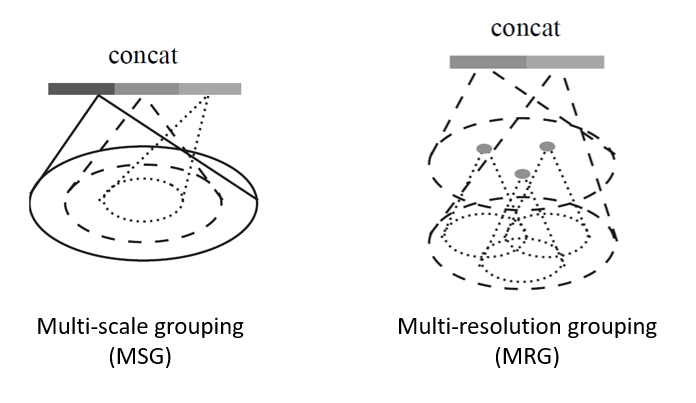
N개의 point로 구성된 point cloud에서 좌표가 d이고 C차원의 point feature를 가졌다면 N \* (d+C)로 표현할 수 있다. 여기서 set abstraction을 거치면 N’ \* (d+C’)가 되는데 N’은 subsample된 point들이고 C’는 feature vector다.

* Sampling 단계에서 N개 중 N’게 만큼 중심점을 선택. 중심점은 각 point마다 uclidian distance가 가장 먼 점임. 이 과정을 Farthest point sampling(FPS)라고 부른다. 이렇게 점들로부터 가장 먼 점을 중심점으로 택하면 일반적이고 대표성을 띨 수 있다.
* Grouping 단계에서는 sampling 단계에서 추출한 각 중심점들과 그 주변 점들을 점묶음으로 묶는다. 점묶음을 구성하는 점들의 개수는 저마다 다름. 바로 다음인 pointnet layer에서 점묶음마다 점들의 개수에 상관없이 고정된 크기의 feature vector를 하나씩 추출해주기 때문. 점묶음을 구성하는 방식은 ball query, KNN가 있다. Ball query는 반지름 r을 정하고 중심점을 기준으로 반지름 내의 점들을 이웃점으로 정의함. KNN은 중심점에 대해 가장 가까운 K개의 점을 이웃점으로 정의. 여기선 ball query방식으로 이웃점을 정의함. 고정된 크기 내에서 점묶음을 형성할 수 있기 때문이다.
* Pointnet 단계에선 각 점묶음들의 feature vector를 pointnet에 통과시켜 하나의 feature vector를 추출. Point cloud의 모든 점묶음들은 N’ \* K \* (3+C’) 크기의 행렬로 표현할 수 있는데 pointnet 단계를 거치면 N’ \* (3+C’) 크기의 행렬로 변환된다(K개의 feature vector 중 하나만 추출하므로). 이 때 절대적인 xyz 좌표가 아닌 중심점과의 상대적인 좌표값을 input으로 사용했는데 점들 간의 위치적 관계에 대한 정보를 담을 수 있기 때문이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Point cloud는 대부분 point들의 밀도가 일정하지 않다. 이는 point cloud의 일반적인 학습을 어렵게 한다. paper에서는 이를 해결하기 위해 point cloud를 다양한 밀도로 sampling하여 학습함. 일종의 data augmentation. 또한 다양한 scale의 point cloud에서 feature vector를 추출하는 네트워크 구조를 활용했는데 이를 density adaptive layer라고 부른다.



Grouping 방식 두 가지다.

* MSG는 grouping 단계를 다양한 scale로 여러 번 적용하여 하나의 중심점에 대해 여러 scale의 point group을 얻는 방식이다. 각 point group에서 추출한 feature vector를 concatenate(합치다)하면 multi-scale feature vector를 얻는다. 각 point group은 임의의 dropout ratio를 선택하여 그 비율에 맞게 random하게 down sampling한다. 그렇게 각 point group마다 서로 다른 scale로 균일하지 않은 density를 갖게끔 변환시켰다. 이를 통해 다양한 sparsity와 서로 다른 uniformity를 가지는 점들을 얻을 수 있다. 그러나 이는 각 중심점들과 이웃점들이 모두 pointnet을 거쳐야 하므로 연산이 많아 비효율적. 그래서 제안한 것이 MRG이다.
* MRG는 MSG의 단점을 보완했다. 서로 다른 scale로 얻은 두 feature vector를 합쳐 multi-scale feature vector를 얻는다. 여기서 첫 vector는 local group에 해당하는 점들 전체에 대해 pointnet 단계를 거쳐서 얻고, 두 번째 vector는 local group보다 한 단계 아래의 sub-region에서 얻은 feature를 종합하여 얻는다. 만약 input으로 들어오는 point cloud의 density가 낮다면 처음 vector에 의해 전반적인 feature를 추출하고, density가 높다면 sub-region에 대한 feature가 더 디테일한 feature를 제공한다고 한다. 그러므로 다양한 density에 대응할 수 있고 계산량도 효율적이다.

Set abstraction layer를 거치면 sampling 단계 때문에 point cloud 크기가 줄어듦. 얻어낸 feature vector를 segmentation task에 쓰려면 원래 크기로 복원해야 한다. 복원하지 않고 set abstraction을 하기 위해 모든 점들을 중심점으로 지정해서 feature aggregation(집합)을 해주는 방법도 있지만 이는 계산량이 커서 paper에서는 point cloud를 down-sampling하고 다시 interpolation(보간, 새로운 점을 만들기 위해 점들을 평균화시키는 것) 기반의 방법을 통해 up-sampling하는 방식을 제안함. Feature vector로부터 (1/거리값)으로 weight를 줘서 interpolation했다. Down-sampling 되기 전 feature vector를 skip-connection(하나의 layer의 output을 몇 개의 layer를 건너뛴 다음 layer의 input에 추기하는 방식)을 통해 concatenate해서 부족할 수 있는 정보량을 보충해줌. Interpolation 고정은 원래 point 개수로 복원될 때까지 반복하고, 얻어낸 feature vector로 segmentation task를 수행함.