Pointnet paper review

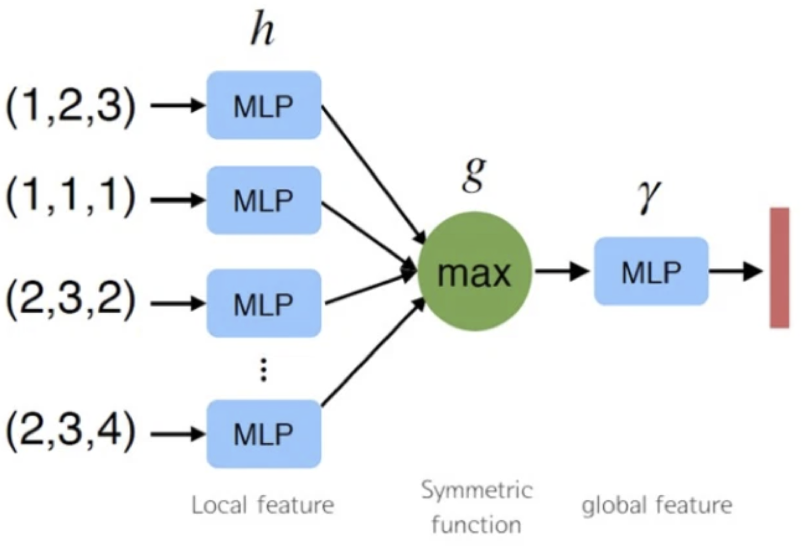
Abstract

Point cloud(점 구름)는 기하학 데이터 구조의 중요 요소로 지금까지 우리는 이러한 데이터를image collection 또는 3d voxel grid로 표현해왔다. 이는 여러 문제를 야기함. Pointnet은 3d 데이터를 다루는 데에 큰 효율과 효능을 발휘한다고 함. 기존 방식과 달리 3d 데이터의 본 형태를 잘 유지시켜준다고 함.

Introduction

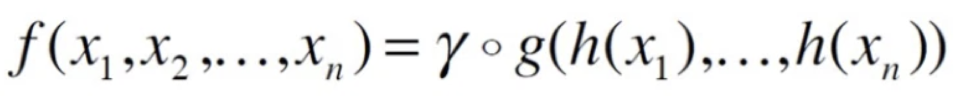
이 논문에서는 3d 기하학 데이터 이를 테면 point cloud 또는 mesh(계층구조)를 다루는 딥러닝 구조를 탐구할 것이다. 이들은 규칙적인 행렬로 이루어진 이미지와 달리 불규칙하기에 우리는 앞서 말했 듯 image collection 또는 3d voxel grid로 변환했다. 이는 볼륨 문제, 데이터의 natural invariances(자연적 불변성)을 모호하게 하는 문제가 발생함. 그래서 우리는 단순 point cloud를 사용한 3d geometry에 변환 없이 바로 접근한다. 이는 PointNets이며 point cloud는 단순하고 통합된 구조로 combinatorial irregularities(조합상의 불규칙) 또는 complexities of meshes(메시의 복잡성)을 피한다. 그래서 형태를 학습하기 쉽다.

Permutation invariant(순열 불변)은 input이 어떤 순서로 오든 결과는 동일해야 한다. 논문에서는 이러한 성질을 만족하기 위해 symmetric function(변수들의 위치가 바뀌어도 결과가 같은 함수)을 제시함. 그 function으로 max pooling 함수를 제시했다. Max pooling은 f(x, y, z) = max(x, y, z)를 의미하기에 symmetric function의 성질에 적합하다. Feature extraction(특징 추출)을 위한 network도 당연히 존재하는데 각 point마다 mlp(multi-layer perceptron: 여러 레이어로 이루어진 다수 입력으로부터 하나의 결과를 내보내는 알고리즘)를 통해 feature를 생성함.

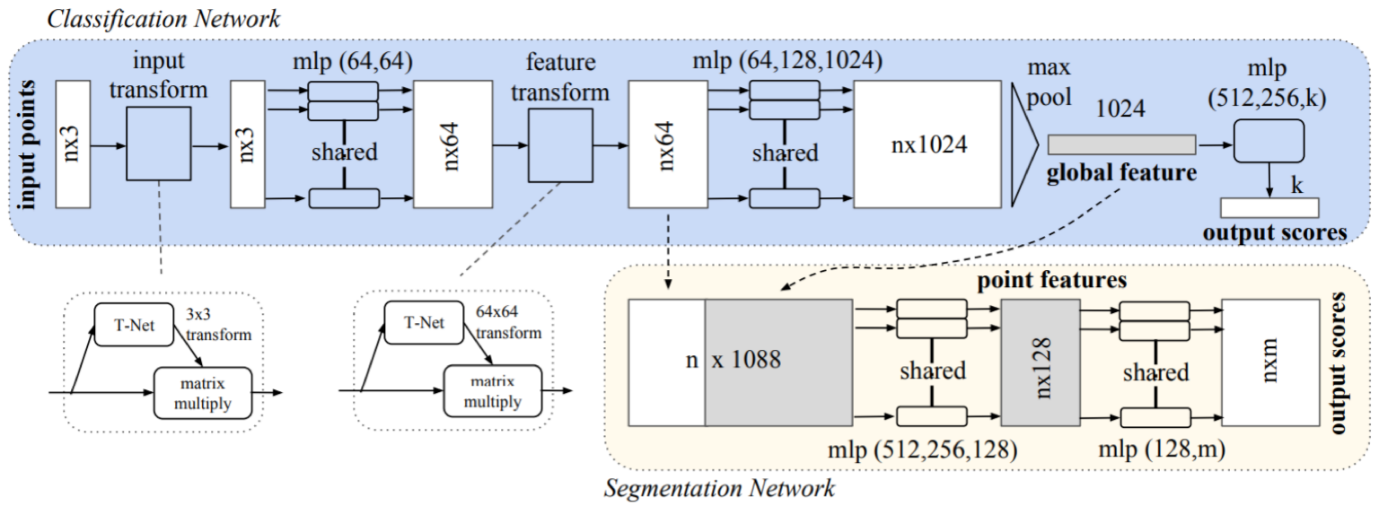


(1,2,3), (1,1,1) 등 각 point(x1~n)들이 mlp에서 h(x)를 거친 후 max pooling으로 가서 g(x)를 거친 후 다음 mlp에서 r(x)를 거치게 된다.

각 point에서 1024개의 feature 추출 -> max pooling으로 각 point에서 1024개의 feature(4, 4) 중 max값을 뽑아서 global feature(4, 1) 1024개 추출.

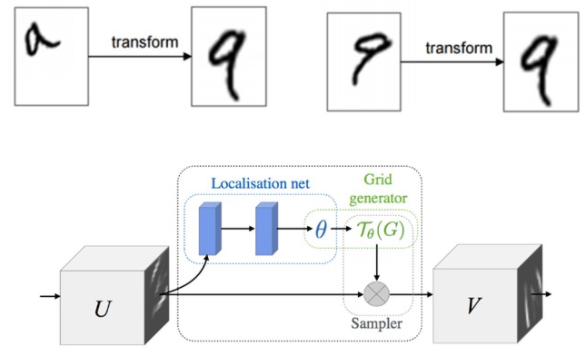


위를 수식화. Point가 되는 x1~n이 h(x)함수를 가진 mlp를 거친 후 feature를 추출하고 그 feature들이 max에 가서 max pooling g(x)함수를 연산하고 다음 mlp에서 r(x)을 거침.

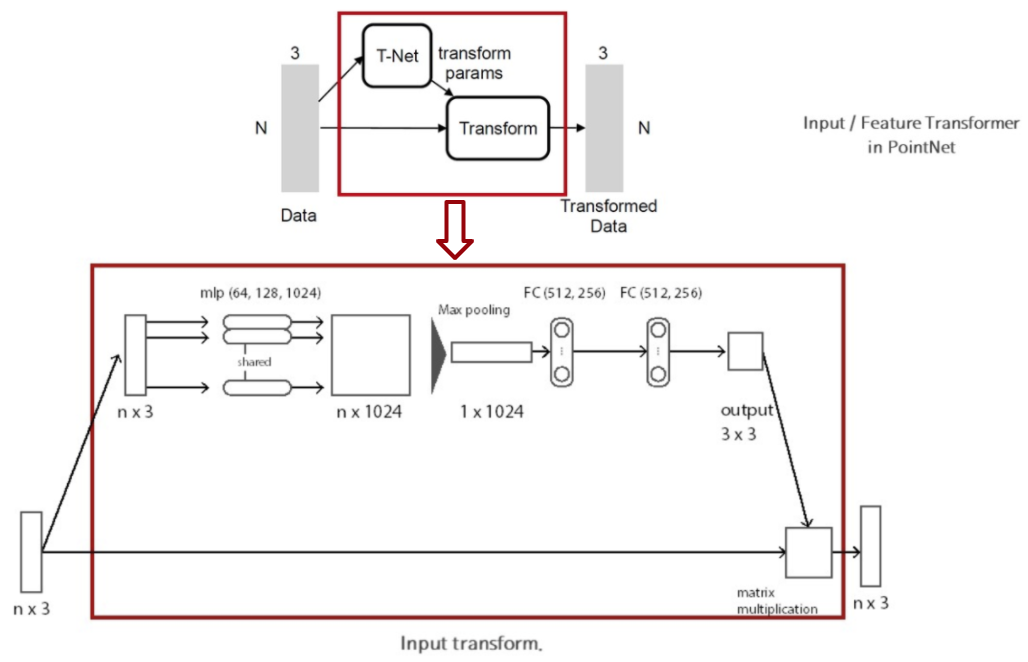


위는 classification network, 아래는 segmentation network이다. 아까 말한 1024개 feature 얘기는 classification 얘기임. Segmentation은 global feature 뿐 아니라 local feature도 함께 이용되어야 함. Classification은 input에 대해 하나의 output이 나오는데 segmentation은 각 pixel 별로 output을 계산해야 하므로 둘 다 필요하다. 이 논문에서는 64개의 feature가 생성된 intermediate layer(중간층)에 max pooling을 거쳐 생성된 1024개의 feature를 concatenate 시키는 것이다. 즉 64개의 local에 max pooling 거친 1024개의 global을 합친다는 것. 이후 mlp를 또 거쳐 point 별 class에 대한 score가 m개가 나와 nXm의 shape이 output으로 나온다고 함.

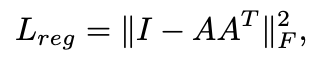
3d data를 다루는 데에 여러 접근들이 있는데 CNN의 다양한 종류들(Spectral CNNs, Multiview CNNs, Volumetric CNNs 등)을 씀. Input에 어떤 transformation이 되더라도 output에는 영향이 가선 안 된다. Point cloud에서는 이를 위해 spatial transformer network(STN)을 이용함. STN는 image에 rigid motion invariant(정적 운동 불변성)를 만족시키기 위해 image를 orthogonal(수직)하게 만든다.



9를 수직하게 만들기 위해 transform을 적용하여 canonical space(규범적인 공간)로 보낸다. orthogonal하게 하려면 어떤 transformation을 적용할 지 계산한 후 input image에 곱하여 변형이 일어나지 않은 모습의 output image를 만든다.



Point cloud에서는 STN의 아이디어를 차용해 T-net을 소개함. 위에 PointNet architecture 아래 쪽에 나오는 부분임. Mini pointnet이라고 하고 T-net에서 point data들을 canonical space(표준 공간)로 보내기 위해 적용될 transformation matrix를 계산하고 input data에 matrix를 곱함. 위에 architecture 보면 input 말고도 feature도 transform을 함. Feature transform은 64X64 transformation matrix를 predict해야해서 차원이 커져 optimizer시키기 어려움. 그래서 정규화 식을 추가함.



A는 transformation matrix이고 A를 orthogonal matrix가 되도록 하는 식이다. Orthogonal matrix를 x라 하면 xx^T가 I임. 즉 transformation matrix가 orthogonal matrix가 되면 input matrix에 곱해도 도형의 원래 고유 모양이 바뀌지 않는 rigid motion이 됨. STN이 rigid motion invariant를 만족시키기 위해 image를 orthogonal하게 하는 것을 차용.

Experiment

-3D object classification:

12311 CAD model들을 사용하고 75% 정도 train, 25% test set으로 나눴다. point cloud는 mesh faces에서 1024개 point들을 sampling 함. 그러고 unit sphere로 normalize 했다. data augmentation으로는 up-axis 축을 따라 랜덤 하게 회전시키거나, 값에 gaussian noise를 넣는 것을 사용했다.

-3D object part segmentation:

part segmentation 문제를 per-point classification problem으로 고려한다. metric으로는 shape의 mIOU 사용했다. 겹치는 point 수들을 IOU에 포함시켜서 측정한다.

-3D part segmentation

각 point는 X, Y, Z 좌표 정보 외에 R, G, B, + normalized location 정보를 사용한다. 즉, point 하나당 9-dim vector다. 여기에 추가로 hand-craft point features로서 local features, local curvature, normal를 추가해서 12-dim vector를 사용했다.

<https://ganghee-lee.tistory.com/50>

<https://arxiv.org/pdf/1612.00593.pdf>