Exploring Kernel Point Convolution

Genglin Liu, Gary Wei

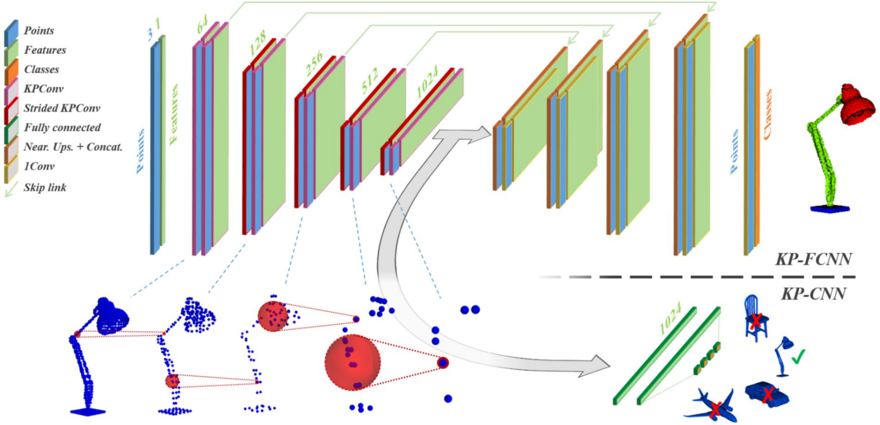
Introduction

We present a comprehensive study on kernel point convolution and the experiments we conducted using the KPConv segmentation network on two different datasets.

The KPConv has an encoder-decoder structure and it is the state-of-the-art on certain point cloud processing tasks.

It consists of 13 resnet blocks and the decoder uses nearest neighbor sampling to produced a segmentation map.

Our project is extended from the official implementation by Hugues at el. (CVPR 2019)



Task 1: Replication of Indoor Scene Segmentation

Trained on Nvidia 2060Ti and Titanx

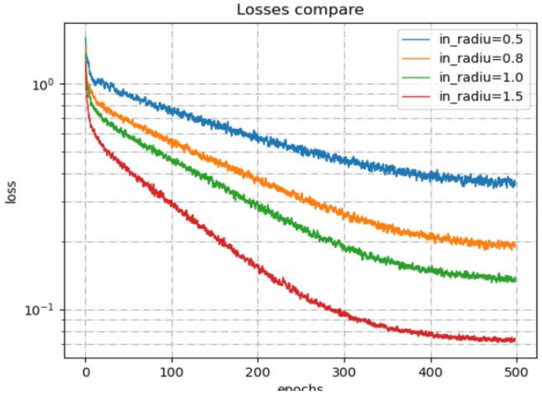
Task: replicating the S3DIS scene segmentation

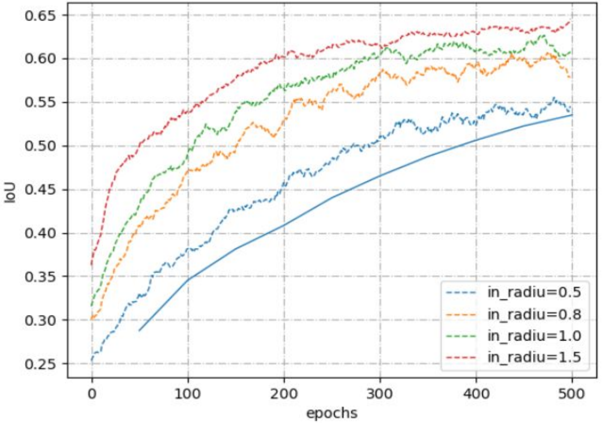
Tuning sphere radius

Memory problem with high radius

Evaluation metric: mean intersection over union (mIoU) - achieves 64%

Challenge: Time and Compute Resources

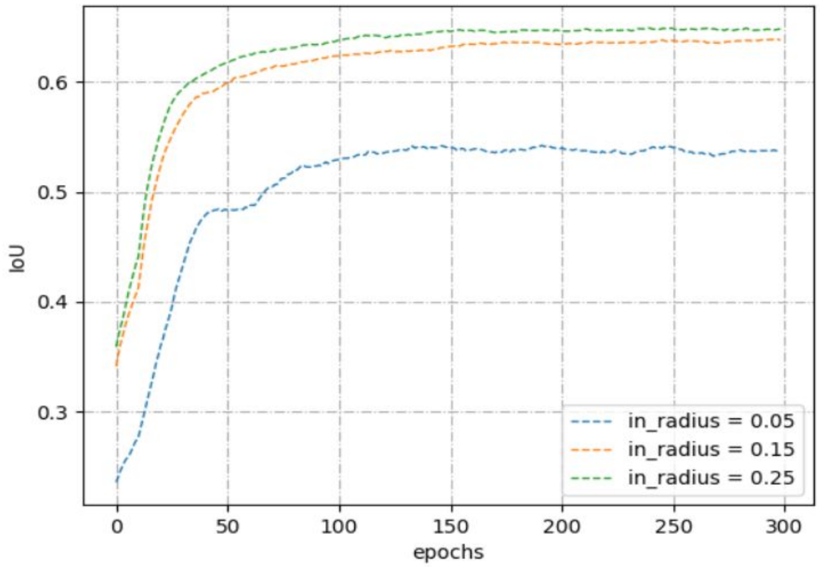




Task 2.1: Shapenet-Part input radius tuning

One of the biggest challenges was loading the new dataset and fit the batch input into the network. Tuning the relevant parameters after we adopted the new dataset to the KPConv pipeline

By tuning input sphere radius, we noticed that within a range that does not cause gradient problem, higher sphere radius gives better results and faster convergence.



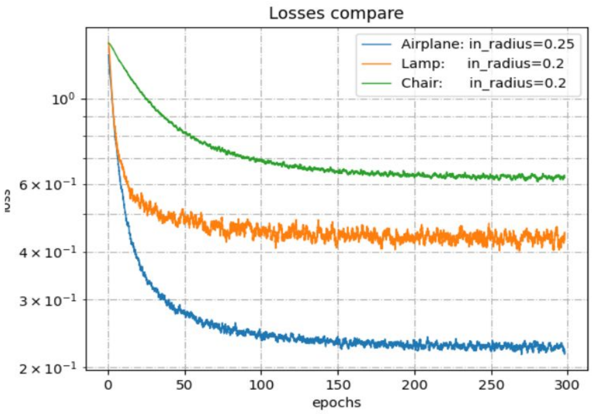
Task 2.2: Part Segmentation on Shapenet-Part

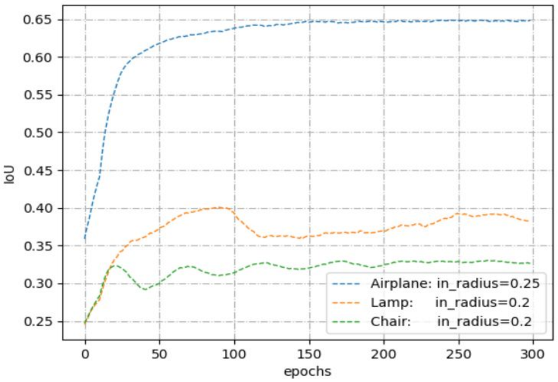
Object part segmentation on three different categories: airplane, lamp, chair

Different classes gave very different mean IoU results

We plotted loss curves with different input radius, and observed the highest IoU for airplane, which is above 65%

Problem: scales of objects and that of indoor scenes are different therefore it’s hard to find the most fitting hyperparameters





Task 2.3: Varying Feature Dimension

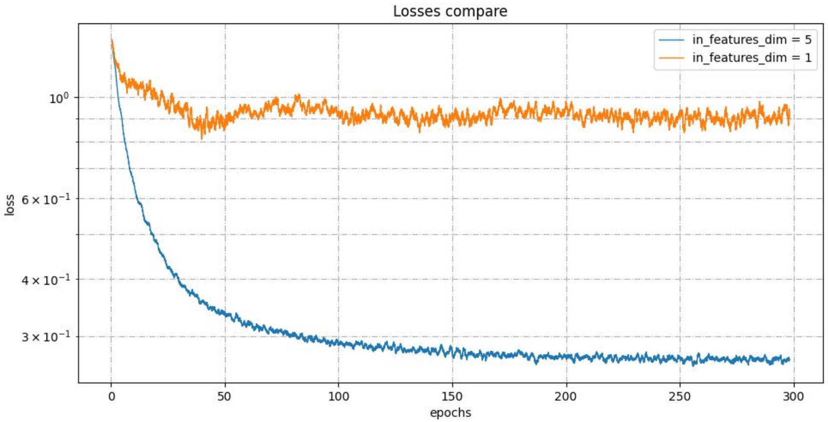
Input feature dimension: 1, 4, 5

1: all 1’s

4: all 1’s + RGB 3 channels

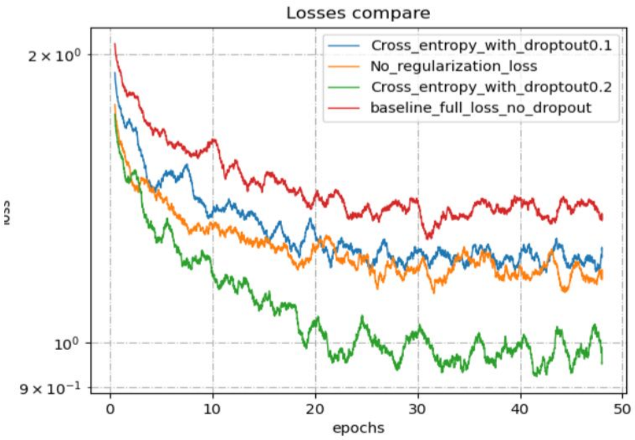
5: all 4 previous dimensions + z coordinate of the point

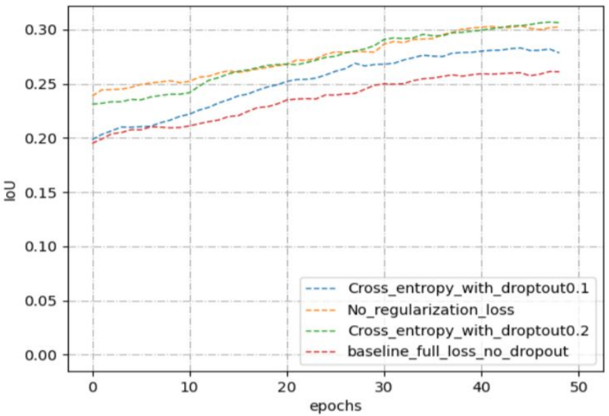
Higher feature dimension - higher model capacity



Task 3: Network Modification

- Different loss functions
    - KL divergence, multiclass soft margin
- Removing regularization loss
- Loss converges just as fast as the combined and IoU seems even better in our experiment
- Challenges the idea of combined loss
- Adding dropout layer to unary blocks
- p=0.2 brings better result





Conclusion

- Replicated one of the core experiments
- Modified network layers and saw potential improvements
- Observed the combined loss vs single cross-entropy loss
- Part Segmentation on three categories of Shapenet-part and parameter tuning

Thanks for watching!