Point cloud generation with

1. With raw GAN
2. With latent GAN (i.e., AE + GAN)

1. Raw GAN

a) Architecture

Generator (Decoder)

Input 128 ---> Linear 64 + ReLU ---> Linear 128 + ReLU ---> Linear 512 + ReLU
---> Linear 1024 + ReLU ---> Linear 2048*3

Discriminator (AE)
Input 2048 x 3 --->

ENCODER

Conv1D (2048, 64, kernel_size=1) + LeakyReLU --->
Conv1D (64, 128, kernel_size=1) + LeakyReLU --->
Conv1D (128, 256, kernel_size=1) + LeakyReLU --->
Conv1D (256, 256, kernel_size=1) + LeakyReLU --->
Conv1D (256, 512, kernel_size=1) + LeakyReLU --->
(output.shape = (512, 3))
--- > MaxPool(dim=-1) --- > (512,)

DECODER

---> Linear(512, 128) + ReLU ---> Linear(128, 64) + ReLU --->
---> Linear(64, 1) + Sigmoid

(Fake or Real)

b) Loss

J = - [t log(y) + (1 - t) log(1-y)]

For Discriminator

J^D = - [sum_x (log(D(x)) + sum_z (log(1 - D(G(z))))],

where x -- real sample, z ~ p(z)

For Generator $J^G = -J^D = [\text{sum}_x (\log(D(x)) + \text{sum}_z (\log(1 - D(G(z))))]$

$d\ \text{sum}_x (\log(D(x)) / d\ \theta_G = 0$

A problem:

If D is very good at distinguishing fake examples from real ones, D(G(z)) will be 0.
Then $\text{sum}_z (\log(1 - D(G(z)))) = 0$.
Then the gradient $d J^G / d\ \theta_G = 0$.
Then the parameters will not be updated!

That is, when Discriminator is too good, Generator has no chance to improve.

Solution:
The heuristic

Let's try to minimize

$J^G = - \text{sum}_z (\log(D(G(z)))),$ -- "non-saturating heuristics"

i.e., Generator wants to make D(G(z)) = 1,