几何计算前沿第四次作业报告
基于 MLP 的点云三维重建

沈千帆 2200013220

2024 年 6 月 11 日

1 简述

本作业主要在 `main.py` 和 `main_embed.py` 中实现了利用 MLP 对于点云进行三维重建的模型，整个算法可以分为以下几个部分：

- `def load_data(path)`: 这个函数实现了点云文件的加载。包括：pointcloud.npz

  - "points": 表面上的点云坐标

  - "normals": 表面上的法向量

  sdf.npz

  - "points": 空间中的采样点坐标（用做训练的空间点集合）

  - "grad": 采样点对应梯度（ground truth 梯度）

  - "sdf": 采样点对应 SDF 值（ground truth SDF 值）

  *.obj: 用于参考的 ground truth 三维形状

- `class MLP(nn.Module)`: 定义用来训练的 MLP 模型。

- `def sdf_loss(pred_sdf, true_sdf, gradient, true_normal)`: 定义 loss 函数。

- `def extract_mesh(model, device, resolution=128, level=0.05, batch_size=10000)`: 用已经训练好的模型在空间内均匀采样，提取 mesh。

- 基于 Fourier feature 的 position encoding

  调用 rff 库进行位置编码。

2 具体实现

2.1 MLP 模型的定义

```
self.fc = nn.Sequential(
            nn.Linear(input_dim, 256, dtype=torch.float64),
            activation_fn(),
            nn.Linear(256, 512, dtype=torch.float64),
            activation_fn(),
```

```
            nn.Linear(512, 512, dtype=torch.float64),
            activation_fn(),
            nn.Linear(512, 512, dtype=torch.float64),
            activation_fn(),
            nn.Linear(512, 256, dtype=torch.float64),
            activation_fn(),
            nn.Linear(256, output_dim, dtype=torch.float64)
        )
```

对于激活函数，我尝试了 LeakyRelu，Swish 和 ELU 函数，得到了不同的结果，在下文之中会进行具体的分析。

2.2 损失函数

根据要求定义损失函数 $L_{sdf} = \sum_{q_i \in Q} \|\tilde{F}(q_i) - F(q_i)\|^2 + \|\nabla\tilde{F}(q_i) - N(q_i)\|^2$ 由 sdf 损失和梯度损失相加而成

```
def sdf_loss(pred_sdf, true_sdf, gradient, true_normal):
    sdf_loss = torch.mean((pred_sdf - true_sdf) ** 2)
    grad_loss = torch.mean((gradient - true_normal) ** 2)
    return sdf_loss + grad_loss

# 计算损失
pred_sdf = model(sdf_points_batch)
pred_grad = torch.autograd.grad(
    outputs=pred_sdf,
    inputs=sdf_points_batchgrad_outputs=torch.ones(pred_sdf.size()).to(device),
    create_graph=True,
    retain_graph=True,
    only_inputs=True)[0]
loss = sdf_loss(pred_sdf, sdf_values_batch, pred_grad, sdf_grad_batch)
```

2.3 位置编码

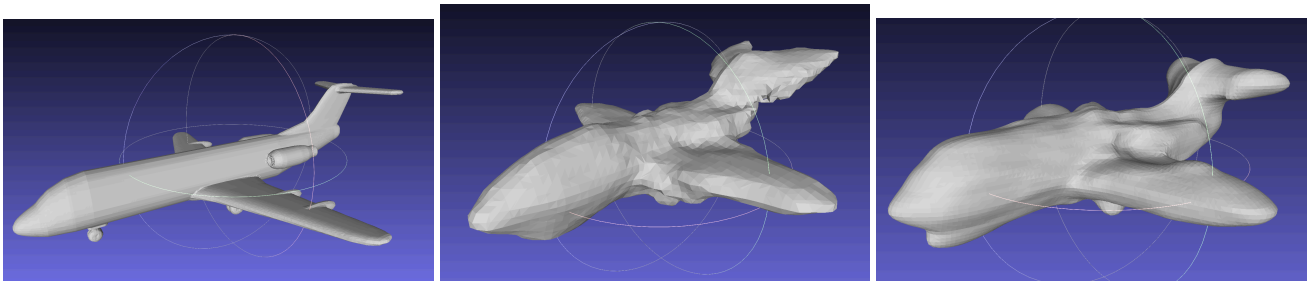
不管使用什么样的激活函数，如果只是把众多采样点和对应的 sdf 值塞进 MLP 中训练，最后得到的结果会趋于平滑。这是因为 MLP 往往只能捕捉低频特征，而物体的细节往往都在高频特征中。所以我们要引入 Fourier 位置编码，在作业中我们直接调用了 rff 的位置编码库，具体实现的是：

$$\sigma(v) = (..., \cos 2\pi\sigma^{(j/m)}v, \sin 2\pi\sigma^{(j/m)}v), j = 0, 1, ..., m-1)$$

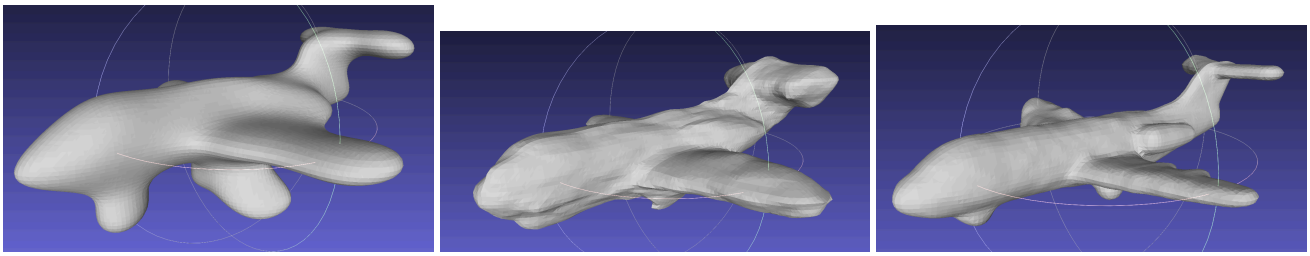
```
encoding = rff.layers.PositionalEncoding(sigma=0.5, m=32)
Xp = encoding(sdf_points_batch)
```

3 实现结果与分析

在实验的过程中，我发现最后呈现出的结果不直接和计算出来的 loss 完全相关。这是因为我们计算的 loss 只是基于 sdf 值和法向。这也不能完全代表整个空间内所有点都能很好符合这个模型，并且由于到处的是显示 mesh，这和基于隐式表达的 MLP 也有一定程度上的出入。在实验中也发现相邻两个 epoch 之间经常会有明显的区别，是重建过度或者是重建不足（已经排除学习率问题）。所以我们接下来会呈现一些实验结果，结合 loss 和具体的几何效果进行分析。



(a) Ground Truth (b) No Embedding+LeakyRelu (c) No Embedding+Elu



(d) Embedding+Swish (e) Embedding+LeakyRelu (f) Embedding+Elu

图 1: Airplane 4(1a32f10b20170883663e90eaf6b4ca52)

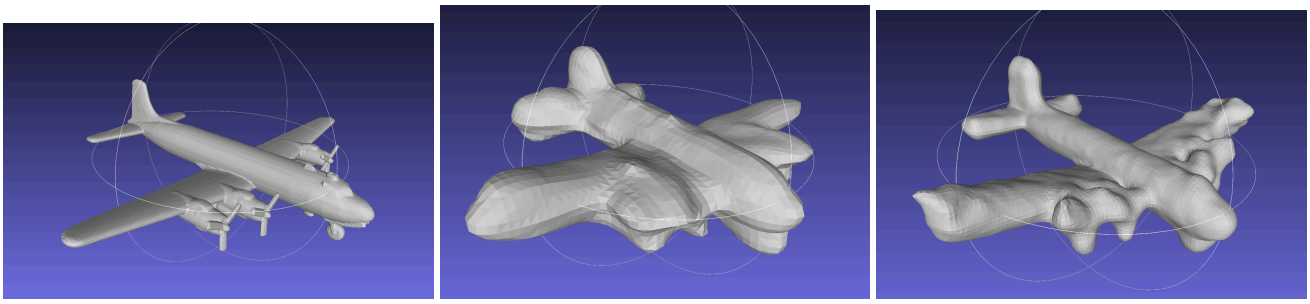
Method	Epochs	Loss
No Embedding + LeakyReLu	25	0.10664553261420563
No Embedding + ELU	35	0.05159308785206694
Embedding + Swish	12	0.12305697416088163
Embedding + LeakyReLu	27	0.1054999612688706
Embedding + ELU	31	0.044168727447405284

表 1: 对应的量化指标 1

首先对 data 文件夹中第四个模型进行了测试。在没有进行位置编码的情况下，如果采用 LeakyReLu 作为激活函数，模型将很快收敛，效果比较一般，对于细节之处比如轮子，发动机基本上会和其他部分耦合。如果使用了 embedding，效果会明显提高，一些细节的特征比如轮子可以重建出来。这样的对比在使用 ELU 激活函数的情况下更加明显，当我们使用了 ELU 函数之后，两种情况下重建效果都更佳（已高亮），这可能是因为 ELU 函数在负半轴上有更强的非线性表达能力。

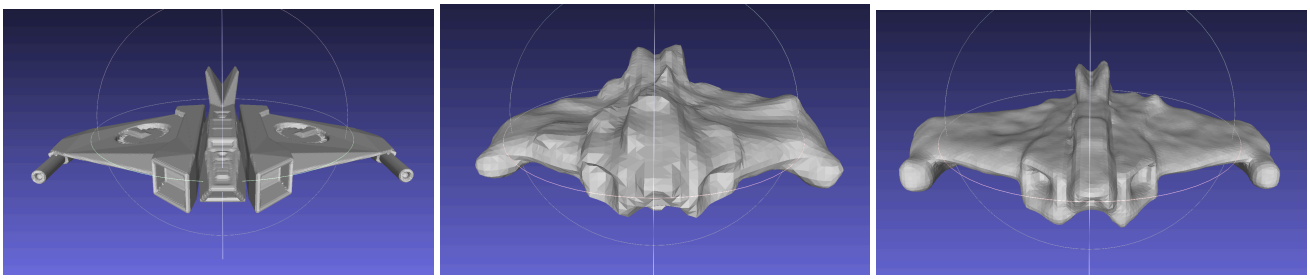
$$\mathrm{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{if } x \leq 0 \end{cases}$$

而用了 Embedding+ELU 之后，轮子，机翼上的扰流板，发动机，尾翼等都已经和 GT 非常相似。在表 1 中我们也提供了具体的量化指标，但是我们发现在 loss 差别不大的时候，loss 的高低并不能直接体现重建效果的好坏，因此展示的图片是效果较好而不一定是训练论述很大的。不过当 loss 有明显降低的时候，效果也会显著变好。所以接下来我们将使用 ELU 作为激活函数对其他几组数据进行 embedding 与否的比对。



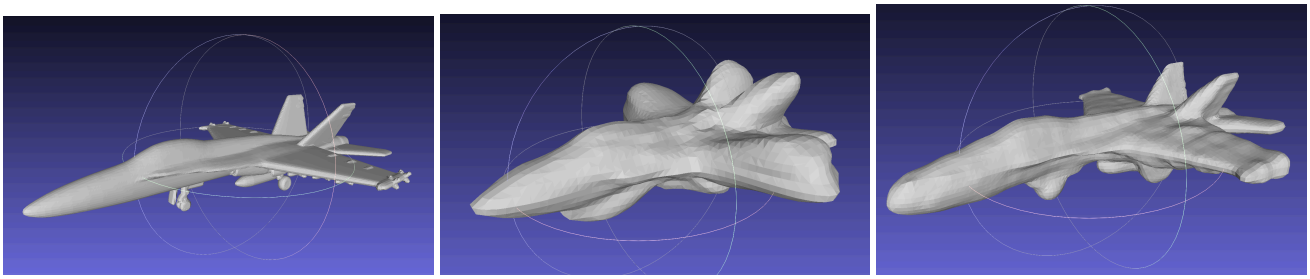
(a) Ground Truth (b) No Embedding (c) Embedding

图 2: Airplane 5(1a9b552befd6306cc8f2d5fe7449af61)



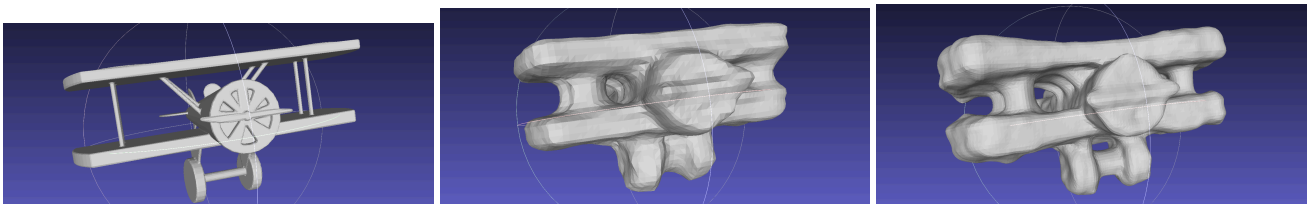
(a) Ground Truth (b) No Embedding (c) Embedding

图 3: Airplane 3(1a9b552befd6306cc8f2d5fe7449af61)



(a) Ground Truth (b) No Embedding (c) Embedding

图 4: Airplane 2(1a6ad7a24bb89733f412783097373bdc)



(a) Ground Truth (b) No Embedding (c) Embedding

图 5: Airplane 1(1a04e3eab45ca15dd86060f189eb133)

Data	Is_Embedding	Epochs	Loss
Airplane 5	No	50	0.060976005779852925
Airplane 5	Yes	44	0.05004158918404567
Airplane 3	No	37	0.06983799904480972
Airplane 3	Yes	12	0.06602005417361578
Airplane 2	No	28	0.09224873765291458
Airplane 2	Yes	28	0.07689028013434064
Airplane 1	No	43	0.034745519746039406
Airplane 1	Yes	36	0.02747466454012587

表 2: 对应的量化指标 2

总的来说，加入 Fourier 的位置编码之后，对于高频细节的处理会得到显著的提高。但如果要更进一步提高模型的表达能力和重建效果，就要使用其他的算法和更完善的损失函数进行处理。

备注：所有的实验结果 obj 文件在作业的文件夹中。