第一次实验报告

胡延伸 PB22050983

2024 年 10 月 7 日

1 预测 GPU 的运行时间

1.1 数据预处理

(a) data_preprocessing_regression() 函数实现

根据官方数据集文档，这一部分只需对 Run_time 列做取对数处理，使得这一列值更小。这一点是重要的，因为取对数处理往往能增强数据的线性性.

(b) data_split_regression() 函数实现

这里只需要将数据集划分为 8:1:1 的训练集、验证集、测试集，使用 dataset 库中的 train_test_split 方法即可方便完成.

1.2 定义模型

(a) LinearRegression 模型实现

包含两个部分:

1. **参数的初始化:** 由于是线性模型，只需定义权重 weights, 以及偏置 bias, 在实验中我们注意到此处的初始值需要设置的非常小.

2. **predict:** 按照公式 $y = XW + b$ 完成该方法的定义即可.

1.3 定义 MSELoss

(a) MSELoss 的 ___call___ 方法

计算对于 N 个样本的均方误差. 公式为

$$L = \frac{1}{2}\sum(\hat{y} - y)^2$$

(b) MSELoss 的 backward 方法

使用链式法则，我们可以推导出 L 对 W 和 b 的梯度:

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \hat{W}} = (\hat{y} - y)x$$

同理,

$$\frac{\partial L}{\partial b} = (\hat{y} - y)$$

按照这两个公式完成对方法的定义即可.

1.4 TrainerR

TrainerR 的 train 方法

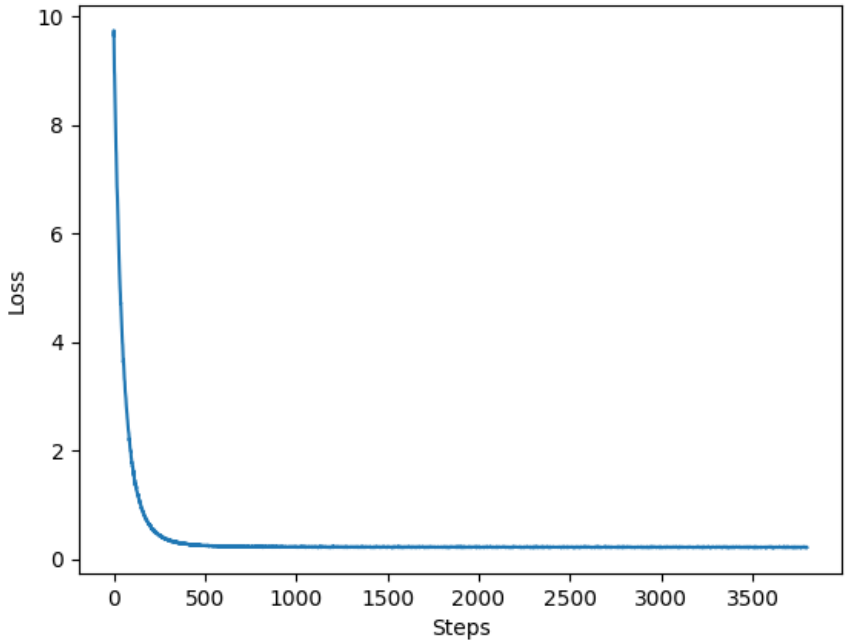
实现 training loop, 即按照顺序完成 data loading, forward propagate, backward propagate 以及 parameter update.

1.5 训练

按照如下的超参数，得到较为不错的拟合效果:

```
decay_every: 20
epochs: 100
lr: 0.01
lr_decay: 0.95
```

其中 loss 值降为 0.2136，训练过程中 loss 曲线展现为极速下降后平缓，如下图:



1.6 评估模型性能

eval_LinearRegression() 函数实现

此处计算均方误差、相对误差和平均预测值，其中平均预测值用于下一个模型的数据处理。在最好的模型上运行得到的结果如下:

```
Average prediction: 4.273265278208095
Relative error: -0.019055604273037097
```

2 对 GPU 的表现进行分类

2.1 数据预处理

(a) data_preprocessing_classification 函数实现

按照第一个模型预测结果的 Run_time 均值构造 label 列，并删除 Run_time 列。

(b) 划分数据集

和第一个模型逻辑上相同，不同之处仅仅在于并非返回 Dataloader 这个类，而直接返回 numpy 数组即可

2.2 定义模型

(a) LogisticRegression 模型实现

此处要求将两个参数（权重和偏置）合成为一个参数，这只需要另外让输入变量额外增加一个特征，这个特征的所有值置为 1 即可。

(b) 模型的预测功能

利用公式 $y = x\beta$ 完成定义即可。

2.3 定义 BCELoss

(a) BCELoss 的 ___call___ 方法

按照 CrossEntropy 公式 $L = -E[y\ln(\hat{y}) + (1-y)\ln(1-\hat{y})]$ 完成函数的定义即可。

(b) BCELoss 的 backward 方法

按照链式法则，得到梯度公式为

$$\frac{\partial L}{\partial \beta} = X^T(\hat{y} - y)/N$$

2.4 TrainerC

(a) TrainerC 的 train 方法

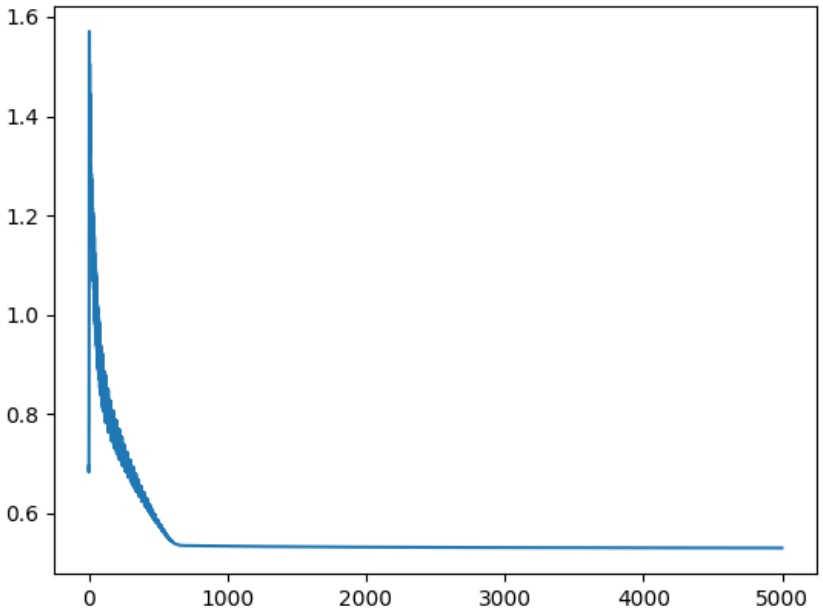
同模型一类似，只不过不使用分批次梯度下降，构建 training loop.

2.5 训练

训练过程相当迅速，在 1 分钟内即可完成，采用的超参数为:

```
decay_every: 20
lr: 0.001
lr_decay: 0.99
mean: 4.273265278208095
steps: 5000
```

得到 loss:0.5546, 及其下降曲线为:



2.6 评估模型性能

计算训练模型的准确率，在最好的模型上，得到的结果为：

```
Accuracy: 0.8265321955003879
```

3 [Optional] 可选项

4 回答问题

- 在对数据进行预处理时，你做了什么额外的操作，为什么?

  z-score Normalization: 对于训练而言，当训练数据的分布保持一致时，训练效率将会大大提高，而对所有输入变量做标准化处理将会大大提高分布的一致性，实际效果上确实使得 loss 下降了一半（由 0.6 降至 0.2）.

- 在处理分类问题时，使用交叉熵损失的优势是什么?

  交叉熵损失能够直接衡量模型输出的概率分布与真实类别标签之间的差异，并且对错误分类惩罚较大，即：当模型的预测接近正确时，损失变化较小；而当模型预测与真实标签相差较大时，损失急剧增大。

- 本次实验中的回归问题参数并不好调，在实验过程中你总结了哪些调参经验?

  先确定大致的合理范围，当 loss 明显过大或者明显不下降时迅速停止训练；影响训练主要因素是 learning rate，先调节它，其次调节 decay every.

- 你是否做了正则化, 效果如何？为什么?

  尝试做过 L2 regularization，但实际使用过程中并没造成训练效率的提升，因此最后的模型的训练并未采用。

5 反馈

- 你在本次作业花费的时间大概是?

  包括作业一和实验一大概在 15 小时左右.

- 你可以随心吐槽课程不足的方面，或者给出合理的建议。若没有想法，直接忽略本小题即可。

  仅对作业一提出一点建议。作业一的题目条件描述并不十分严谨。第一题的各种名词（如响应变量，规范响应函数）作为大多数人第一次遇到，而且各种参考资料上的名称并不统一的情况下，应该指明这些名词所对应的变量、函数到底指的是什么；第一题，第二题中的各个变量如 η, θ，作为线性代数的证明题，应该明确指明它的形状，否则就会造成证明中对象的认知困难，容易导致一种“我不知道要证明什么”的尴尬局面；第四题中题目很有歧义，题目中多次出现“观测”这个词，并且同时用在真实值和预测上，造成理解上的困难，个人认为应该用更精准的词加以区分。总而言之，个人希望以后这种证明作业题条件描述应该越精准越好，而不仅仅是让我们去找各种参考资料然后去猜题目意思。