|  |  |
| --- | --- |
| 成绩 |  |

 

装

订

线

网络空间安全与计算机学院实验报告

实验课程名称**：神经网络与深度学习实验**

实验项目名称： 不同优化算法的比较分析

学生姓名： 刘新媛

专 业： 人工智能

学 号： 20221205037

实验地点： C1-320

实验时间： 2024.12.18第7-8节

指导教师： 魏勇刚

预习报告部分

1. **实验目的：**对不同优化算法在2D和3D图像上可视化进行比较分析
2. **实验原理：**

除了批大小对模型收敛速度的影响外，学习率和梯度估计也是影响神经网络优化的重要因素。

神经网络优化中常用的优化方法也主要是如下两方面的改进，包括：

* 学习率调整：通过自适应地调整学习率使得优化更稳定。AdaGrad、RMSprop、AdaDelta算法等。
* 梯度估计修正：通过修正每次迭代时估计的梯度方向来加快收敛速度。动量法、Nesterov加速梯度方法等。
* 综合学习率调整和梯度估计修正：Adam算法

1. **实验内容：**

1. 不同优化算法的比较分析-2D可视化实验

* 1. 优化算法的实验设定
  2. 学习率调整优化策略

（1）AdaGrad算法

（2）RMSprop算法

* 1. 梯度估计修正优化策略

（1）动量法

（2）Adam算法

2. 不同优化算法的比较分析-3D可视化实验

（1）函数的3D可视化

（2）加入不同优化算法，画出轨迹

（3）复现CS231经典动画

1. **实验所用设备：**pycharm、pytorch

实验报告部分

1. **实验步骤：**

**1. 不同优化算法的比较分析-2D可视化实验**

|  |
| --- |
|  |

* 1. **优化算法的实验设定**

|  |
| --- |
| **2D可视化实验（以Sphere函数为例）：** |
| * **被优化函数：**将被优化函数实现为**OptimizedFunction**算子，其forward方法是Sphere函数的前向计算，backward方法则计算被优化函数对x的偏导。 * **训练函数 ：**定义一个训练函数**train\_f**，记录梯度下降过程中每轮的参数x和损失。 * **可视化函数 ：**定义一个**Visualization**类，用于绘制x的更新轨迹。之后再定义**train\_and\_plot\_f**函数，调用**train\_f**和**Visualization**，训练模型并可视化参数更新轨迹。 * **模型训练与可视化：**指定Sphere函数中w的值，实例化被优化函数，通过相应优化算法更新参数，并可视化x的更新轨迹。 |

* 1. **学习率调整优化策略**

学习率是神经网络优化时的重要超参数。在梯度下降法中，学习率αα的取值非常关键，如果取值过大就不会收敛，如果过小则收敛速度太慢。

常用的学习率调整方法包括如下几种方法：

* 学习率衰减：如分段常数衰减（Piecewise Constant Decay）、余弦衰减（Cosine Decay）等；
* 学习率预热：如逐渐预热(Gradual Warmup) 等；
* 周期性学习率调整：如循环学习率等；
* 自适应调整学习率的方法：如AdaGrad、RMSprop、AdaDelta等。自适应学习率方法可以针对每个参数设置不同的学习率。

**（1）AdaGrad算法**

AdaGrad算法（Adaptive Gradient Algorithm，自适应梯度算法)是借鉴 ℓ2正则化的思想，每次迭代时自适应地调整每个参数的学习率。

在第t次迭代时，先计算每个参数梯度平方的累计值，如公式（1）所示，其参数更新差值如公式（2）所示。

**（1）**

**（2）**

其中 α 是初始的学习率，ϵ 是为了保持数值稳定性而设置的非常小的常数，一般取值e−7到e−10。此外，这里的开平方、除、加运算都是按元素进行的操作。

实验首先构建优化器 定义Adagrad类，继承Optimizer类。定义step函数调用adagrad进行参数更新，再使用被优化函数展示Adagrad算法的参数更新轨迹。

**（2）RMSprop算法**

RMSprop优化器是一种改进的Adagrad优化器，通过引入一个衰减系数，让每回合都衰减一定的比例。可以在有些情况下避免AdaGrad算法中学习率不断单调下降以至于过早衰减的缺点。RMSprop算法和AdaGrad算法的区别在于RMSprop算法中的计算由累积方式变成了**加权移动平均**。在迭代过程中，每个参数的学习率并不是呈衰减趋势，既可以变小也可以变大。

RMSprop算法先计算每次迭代梯度平方的加权移动平均，如公式（3）所示，其参数更新差值如公式（4）所示。

**（3）**

**（4）**

其中，β为衰减率，一般取值为0.9；α是初始的学习率。

实验首先构建优化器RMSprop类，继承Optimizer类。定义step函数调用rmsprop进行参数更新，再使用被优化函数展示RMSprop算法的参数更新轨迹。

* 1. **梯度估计修正优化策略**

 除了调整学习率之外，还可以进行梯度估计修正。在小批量梯度下降法中，由于每次迭代的样本具有一定的随机性，因此每次迭代的梯度估计和整个训练集上的最优梯度并不一致。如果每次选取样本数量比较小，损失会呈振荡的方式下降。

一种有效地缓解梯度估计随机性的方式是通过**使用最近一段时间内的平均梯度来代替当前时刻的随机梯度来作为参数更新的方向**，从而提高优化速度。

**（1）动量法**

动量法（Momentum Method）是用之前积累动量来替代真正的梯度。每次迭代的梯度可以看作加速度。

在第t次迭代时，计算负梯度的“加权移动平均”作为参数的更新方向，如公式（5）所示。

**（5）**

其中其中为动量因子，通常设为0.9，α 是初始的学习率。

实验首先构建优化器 Momentum类，继承Optimizer类。定义step函数调用momentum进行参数更新，再使用被优化函数展示Momentum算法的参数更新轨迹。

**（2）Adam算法**

Adam算法（Adaptive Moment Estimation Algorithm，自适应矩估计算法）可以看作**动量法和RMSprop算法**的结合，不但使用动量作为参数更新方向，而且可以自适应调整学习率。

Adam算法一方面计算梯度平方的加权移动平均，如公式（6）所示（和RMSprop算法类似），另一方面计算梯度gtgt​的加权移动平均，如公式（7）所示（和动量法类似）。

**（6）**

**（7）**

其中和分别为两个移动平均的衰减率，通常取值为=0. 9,=0.99，我们可以把和分别看作梯度的均值(一阶矩)和未减去均值的方差(二阶矩)。

假设，那么在迭代初期和的值会比真实的均值和方差要小。特别是当和都接近于1时，偏差会很大。因此，需要对偏差进行修正，如公式（8）和公式（9）所示。

**（8）**

**（9）**

Adam算法的参数更新差值如公式（10）所示。

**（10）**

|  |
| --- |
|  |
| · | |

实验首先构建优化器Adam类，继承Optimizer类。定义step函数调用adam函数更新参数。

**2. 不同优化算法的比较分析-3D可视化实验**

**（1）函数的3D可视化**

|  |
| --- |
|  |
|  |

|  |
| --- |
| **实现对两个不同函数进行三维可视化展示：** |
| * **函数定义：**定义两个函数，一个是，另一个是 ，以二维向量 x 作为输入来计算函数值，同时确定各自求导规则用于定义梯度。 * **生成坐标网格：**在 [-3, 3] 区间、间隔 0.1 生成 x1 和 x2 坐标取值范围，用 np.meshgrid 把它们转变成二维坐标网格，并转为 torch.Tensor 类型，作为后续函数计算输入。 * **计算函数值:**分别将坐标网格数据输入两个函数，算出各自对应的函数值。 * **绘制3D图：**创建一个整体图像，添加两个三维投影的子图。 |

**（2）加入不同优化算法，画出轨迹**

在 2D可视化实验的设定基础上添加3D绘图的要素，主要是构建 **Visualization3D** 类继承自 **animation.FuncAnimation**，初始化接收轨迹数据等并创建绘图对象，**init\_animation** 方法初始化绘图数据，**animate** 方法更新绘图数据实现动态绘制。

**（3）复现CS231经典动画**

绘制函数- x[0] \* x[0] / 2 + x[1] \* x[1] / 1的不同优化算法下的3D轨迹图。

**二、 实验数据及结果分析：**

[【实验17】不同优化算法的比较分析-CSDN博客](https://blog.csdn.net/qq_73704268/article/details/144564754?sharetype=blogdetail&sharerId=144564754&sharerefer=PC&sharesource=qq_73704268&spm=1011.2480.3001.8118)

1. **实验结论：**

在大多数情况下，带有动量的优化算法以及自适应学习率算法（如 RMSprop 和 Adam）能够提供更好的性能。动量法有助于加速收敛并减少震荡，而自适应学习率算法能够根据函数的特性自动调整学习率，适应不同参数的更新需求，从而更快地找到较优解。

1. **总结及心得体会：**
2. 优化算法对深度学习模型性能和训练速度影响重大，但经过实验我发现无通用最优算法，需依情况选择。选择算法时要考虑模型、数据集大小特性与训练效率等因素，真正实践中常需多试几种。下表是我对这几种优化算法的总结：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **类别** | **算法名称** | **特点与说明** | | **优点** | **缺点** | **适用场景** |
| **梯**  **度**  **下**  **降**  **类**  **方**  **法** | **梯度下降法 (GD)** | 通过计算所有样本的梯度更新参数（批量模式） | 理论简单  收敛稳定 | | 耗时 | 小数据集  凸优化问题 |
| **随机梯度下降 (SGD)** | 每次使用一个样本计算梯度更新参数 | 更新频繁  计算效率高 | | 可能导致  局部震荡 | 大规模数据集 |
| **小批量梯度下降 (Mini-batch GD)** | 每次使用一个小批量样本计算梯度并更新参数。 | 平衡计算效率 与稳定性 | | 更新频率依赖批量大小  仍可能震荡 | 大规模数据集 |
| **动量法 (Momentum)** | 引入动量项，利用历史梯度加速下降，减少震荡。 | 加快收敛速 度，减少震荡 | | 动量参数需调节，初始设置不当收敛不良。 | 非凸优化中减少局部震荡 |
| **Nesterov 动量 (NAG)** | 动量法的改进，提前调整梯度方向，减少计算冗余。 | 更快收敛，比普通动量法平滑 | | 计算复杂度高 | 非凸优化中减少局部震荡 |
| **自**  **适**  **应**  **学**  **习**  **率**  **方**  **法** | **Adagrad** | 每个参数有独立的学习率，自适应调整更新步长。 | 对稀疏数据友好，学习率随时间自动调整。 | | 学习率会持续变小，可能导致模型早停 | NLP、稀疏特征任务 |
| **Adadelta** | 不直接累积梯度平方，动态调整学习率范围。 | 无需手动调整学习率，解决 Adagrad 的学习率过小问题。 | | 对于某些复杂任务，学习率仍需微调。 | 动态数据流任务，如时间序列预测 |
| **RMSprop** | Adagrad 的改进版，用滑动平均代替累积梯度平方。 | 稳定收敛 | | 容易过拟合小批量数据 | RNN、深层网络的训练，非凸优化问题 |
| **Adam** | 结合动量法和 RMSprop，自适应更新学习率和动量。 | 学习率调节自动化 | | 对超参数敏感 | 图像分类、生成模型、语言模型等多任务 |

2、在查资料时看到一句话“机器学习界有一群炼丹师，他们每天的日常是：拿来药材（数据），架起八卦炉（模型），点着六味真火（优化算法），就摇着蒲扇等着丹药出炉了。”所以模型优化算法的选择直接关系到最终模型的性能。有时候效果不好，未必是特征的问题或者模型设计的问题，很可能就是优化算法的问题。

基于前人的经验，①优先考虑SGD+Nesterov Momentum或者Adam②模型是非常稀疏的，那么优先考虑自适应学习率的算法③在模型设计实验过程中，要快速验证新模型的效果，可以先用Adam进行快速实验优化；在模型上线或者结果发布前，可以用精调的SGD进行模型的极致优化④制定一个合适的学习率衰减策略。可以使用定期衰减策略，比如每过多少个epoch就衰减一次⑤考虑不同算法的组合。先用Adam进行快速下降，而后再换到SGD进行充分的调优。

3、这学期的深度学习实验课已经接近尾声，起初空空的“深度学习实验”文件夹也一点点变成了1.5G大小，csdn的粉丝也从0慢慢增长到了400，感谢耐心解答疑问的学长学姐，感谢认真负责的老师，感谢认真写好每一份实验的自己，虽然课程已结，但是对深度学习的探索之路永远不会终止，希望自己在未来的学习和实践中能收获更多宝贵的知识！

参考：

[NNDL实验 优化算法3D轨迹 复现cs231经典动画\_优化算法寻优过程3d-CSDN博客](https://blog.csdn.net/qq_38975453/article/details/128192767?ops_request_misc=&request_id=&biz_id=102&utm_term=CS231%E7%BB%8F%E5%85%B8%E5%8A%A8%E7%94%BB&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-0-128192767.142^v100^pc_search_result_base4&spm=1018.2226.3001.4187)

[【NNDL 作业】优化算法比较 增加 RMSprop、Nesterov\_rmsprop+nesterov的python实现-CSDN博客](https://blog.csdn.net/qq_38975453/article/details/128206475?spm=1001.2014.3001.5501)

[NNDL 作业11：优化算法比较\_optimizers = ordereddict() optimizers["sgd"] = sgd-CSDN博客](https://blog.csdn.net/qq_38975453/article/details/128025693)

[深度学习优化算法比较\_深度学习 常用优化算法比较-CSDN博客](https://blog.csdn.net/weixin_42132035/article/details/139288141?ops_request_misc=%7B%22request%5Fid%22%3A%22bdb9ab14f4affa07817215f4857d81bb%22%2C%22scm%22%3A%2220140713.130102334..%22%7D&request_id=bdb9ab14f4affa07817215f4857d81bb&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-1-139288141-null-null.142^v100^pc_search_result_base4&utm_term=%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0%E4%BC%98%E5%8C%96%E7%AE%97%E6%B3%95%E6%AF%94%E8%BE%83&spm=1018.2226.3001.4187)

[Adam那么棒，为什么还对SGD念念不忘 (3)—— 优化算法的选择与使用策略 - 知乎](https://zhuanlan.zhihu.com/p/32338983)

[深度学习中的优化算法总结 - ZingpLiu - 博客园](https://www.cnblogs.com/zingp/p/11352012.html)