技术报告

2022 级计科 3 班 陈秋澄 3022244290

摘要

本研究旨在比较和分析几种常见的优化算法在 LeNet-5 卷积神经网络训练中的表现，评估其在 MNIST 和 CIFAR-10 数据集上的收敛行为和泛化能力。具体而言，我比较了动量 SGD、AdaDelta、Adam 和 RMSprop 四种优化方法，分析了它们在不同数据集上的训练过程、收敛速度及最终性能。实验结果表明，Adam 优化算法在收敛速度和训练稳定性上均表现出色，尤其在 MNIST 数据集上取得了快速且准确的结果。动量 SGD 虽然在 MNIST 数据集上表现良好，但在 CIFAR-10 上收敛较慢且泛化能力较弱。AdaDelta 和 RMSprop 则在不同的数据集上表现出较为稳定的收敛特性，但在泛化能力上略逊于 Adam。总体而言，Adam 优化算法在大多数情况下展现了最佳的性能，尤其适用于深度神经网络训练。然而，针对不同任务和数据集，选择适合的优化方法仍然是提升模型性能的关键。通过本研究，我为 LeNet-5 及其他卷积神经网络的优化提供了有价值的参考，并为实际应用中的优化方法选择提供了依据。

关键词— LeNet-5，优化算法，动量 SGD，AdaDelta，Adam，RMSprop，收敛行为，泛化能力

1. 引言

随着深度学习技术的迅猛发展，卷积神经网络（CNN）已成为解决图像分类、目标识别等计算机视觉任务的核心技术之一。LeNet-5 作为最早的深度卷积神经网络之一，因其在手写数字识别（MNIST 数据集）上的出色表现，成为了深度学习研究中的经典模型之一。然而，随着模型复杂性的增加和数据集规模的扩大，如何优化神经网络模型的训练过程，提升其收敛速度和泛化能力，成为了深度学习研究中的重要课题。

近年来，深度学习领域在神经网络训练优化方面取得了显著进展。最初，传统的随机梯度下降（SGD）方法由于其较慢的收敛速度和对学习率的敏感性，导致在处理大规模数据集时面临许多困难。为了克服这些问题，各种自适应优化方法应运而生，如 AdaGrad、RMSprop、Adam 等，这些方法通过动态调整每个参数的学习率，显著提高了训练效率和收敛速度与效果，成为现代深度学习中的主流优化算法。

尽管现有优化方法在许多深度学习任务中取得了显著成果，但仍然存在一些挑战。首先，尽管现代优化算法能够在许多任务中提高收敛速度，但其在不同数据集和任务上的表现差异较大，优化方法的选择和超参数的调节仍然依赖于经验，缺乏普适性的理论指导。尤其是当面对数据较少或者具有较强噪声的情况时，现有优化方法的泛化性能可能受到影响。其次，不同优化算法在训练过程中的收敛行为存在较大差异，可能导致在相同的网络架构下，收敛速度和最终精度的显著不同。如何平衡模型的收敛速度与最终的泛化性能，仍然是深度学习训练中的难题之一。

另外，虽然诸如 Adam 等优化算法在收敛速度和鲁棒性上表现优越，但在某些任务中，它们可能会出现过拟合的风险，尤其是在数据量较少或任务较为简单时，优化算法可能会依赖于噪声数据，从而影响模型的泛化能力。因此，研究如何在保证收敛速度的同时，提升优化算法的泛化性能，仍然是深度学习研究中的一个亟待解决的问题。

本研究通过比较分析四种常见优化方法（动量 SGD、AdaDelta 、RMSprop 和 Adam）对 LeNet-5 模型在 MNIST 数据集上的表现。通过深入探讨这些优化方法的收敛行为、泛化能力及其优缺点，我们期望为 LeNet-5 模型的训练提供更有针对性的优化策略，同时为深度学习领域中的优化方法选择提供一定的理论支持和实践指导。

在本节的最后一段中，我将对本技术报告的主要贡献总结如下：

1) 通过对比不同优化方法在训练过程中收敛速度的差异，研究它们在不同复杂程度数据集上收敛的稳定性和效率。

2) 通过验证集和测试集的性能对比，分析不同优化方法对模型泛化能力的影响，探讨其对过拟合的抑制作用。

3) 分析每种优化方法在训练过程中是否容易产生震荡，是否容易出现梯度爆炸或消失的问题。

2. 研究方法

2.1 研究思路

本研究的核心目标是分析和比较三种优化方法（动量 SGD、AdaDelta 和 Adam）在训练 LeNet-5 模型时的

收敛行为、稳定性和泛化能力。为了实现这一目标，我们从以下几个方面展开实验分析：

1. 优化算法选择与应用：选择三种常见且具有代表性的优化方法：传统的动量 SGD、自适应学习率方法 AdaDelta 和 Adam（结合动量和自适应学习率的优化算法）。对比这三种优化算法的训练效果，并分析它们在不同阶段的表现差异。
2. 数据集选择：本研究使用两种标准数据集：MNIST 和 CIFAR-10。MNIST 数据集包含手写数字图像，适合基础模型的训练与验证；CIFAR-10 数据集包含 10 类物体图像，数据复杂度更高，适合测试优化算法在更复杂任务上的效果。
3. 模型架构：使用 LeNet-5 模型作为基础网络架构，LeNet-5 是卷积神经网络（CNN）的经典代表，具有两个卷积层、两个池化层和两个全连接层。我们在此架构基础上进行优化算法比较。
4. 评估指标：使用训练集和验证集的损失值、准确率以及最终测试集的泛化性能来评估不同优化方法的效果。通过比较各优化方法的收敛速度、最终模型准确性以及过拟合现象，分析其在不同数据集上的适应性。

2.2 LeNet-5 网络架构

LeNet-5 网络架构的池化层原为平均池化，Gholamalinezhad[1]等研究证明平均池化使整体信息的平滑表示，而最大池化则突出目标的关键特征。现在，人们几乎都使用最大池化而很少用平均池化，因此我们将原 LeNet-5 网络架构的两个平均池化层修改为最大池化。首先，我们使用 MNIST 手写数字数据集来进行实验：

1. 输入层：28*28 像素的灰度图像。
2. 卷积层 1（C1）：6 个 5*5 卷积核，输出尺寸为 28*28*6，填充为 2。
3. 池化层 1（S2）：采用 2*2 的最大池化（步幅 stride=2），输出尺寸为 14*14*6。
4. 卷积层 2（C3）：16 个 5*5 卷积核，输出尺寸为 10*10*16，填充为 0。
5. 池化层 2（S4）：采用 2*2 的最大池化，输出尺寸为 5*5*16，步幅为 2。
6. 全连接层 1（F5）：将前一层的输出展平成一维，大小为 120，连接到 120 个神经元。
7. 全连接层 2（F6）：连接到 84 个神经元。
8. 输出层（F7）：10 个神经元，分别表示图像集的 10 个类别。

此结构在手写数字识别任务（如 MNIST 数据集）中具有良好的表现，但在处理复杂数据集（如 CIFAR-10）时，其深度和特征提取能力较为有限。

2.3 优化算法及原理

2.3.1 动量 SGD

动量 SGD[4]是传统的梯度下降优化方法的一种变种，其基本思想是在每次更新时引入上一次梯度更新的“惯性”，从而加速收敛并减少震荡。动量 SGD 的更新公式如下：

$$v_t = \beta_{t-1} + (1-\beta)\nabla_\theta J(\theta)$$

$$\theta_t = \theta_{t-1} - \eta v_t$$

其中，β是动量因子（通常取值 0.9），η 是学习率，$\nabla_\theta J(\theta)$是损失函数对参数的梯度。动量方法通过平滑梯度的更新，避免了梯度下降中的震荡和局部最小值问题。

2.3.2 AdaDelta

AdaDelta[5]是一种自适应学习率优化方法，它在训练过程中根据参数更新的历史变化自适应地调整学习率。与 AdaGrad 类似，AdaDelta 通过累计梯度的平方来调整每个参数的学习率，但与 AdaGrad 不同的是，AdaDelta 使用了一个衰减因子来限制历史梯度的影响，使得学习率的更新不会随着时间推移不断减少。AdaDelta 的更新公式如下：

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho)g_t^2$$

$$\Delta\theta_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

其中，ρ是衰减因子，通常设置为 0.95，ϵ 是防止除零错误的小常数（如 10^{-6}）。AdaDelta 通过自适应调整学习率，能够有效应对稀疏梯度和不同参数的尺度差异。

2.3.3 Adam

Adam[6]结合了动量法和 RMSprop 的优点，通过计算一阶矩（梯度的均值）和二阶矩（梯度的平方的均值）来动态调整每个参数的学习率。Adam 的更新规则如下：

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2$$

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t}, \quad \widehat{v_t} = \frac{v_t}{1-\beta_2^t}$$

$$\theta_t = \theta_{t-1} - \eta \frac{\widehat{m_t}}{\sqrt{\widehat{v_t} + \epsilon}}$$

其中，β_1和β_2 是动量和均方根传播的衰减系数（通常设置为 0.9 和 0.999），η 是学习率，ϵ 是一个防止除零错误的小常数。Adam 通过计算梯度的一阶和二阶

矩，动态调整每个参数的学习率，从而加速收敛并提高训练的稳定性。

2.3.4 RMSprop

RMSprop[7]优化算法通过自适应调整每个参数的学习率来克服梯度下降算法中学习率选择不当的问题。与 AdaDelta 相似，RMSprop 保持了过去梯度的均方根（RMS）值，并根据这个均值调整每次参数更新的步长。更新规则如下：

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta)\left(\nabla L(\theta_t)\right)^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \nabla L(\theta_t)$$

β为指数加权移动平均的衰减因子（通常取值为 0.9）。$\nabla L(\theta_t)$为损失函数L对参数θ_t的梯度。η为全局学习率（通常需调参）。ϵ为一个小值，用于防止除零错误（例如 $\epsilon = 10^{-8}$）。

3. 实验与结果分析

3.1 实验数据集

1. MNIST 数据集是一个通常用于训练各种数字图像处理系统的大型数据集。该数据库通过对来自 NIST 原始数据库的样本进行修改创建，涵盖手写数字的图像，共包含 60,000 张训练图像和 10,000 张测试图像，尺寸为 28×28 像素。该数据集相对简单，背景干净，噪声较少，广泛运用于机器学习领域的训练与测试当中。

2. CIFAR10 数据集共有 60000 个样本，包含 10 种物体的图像，分别是飞机 airplane、汽车 automobile、鸟 bird、猫 cat、鹿 deer、狗 dog 、青蛙 frog、马 horse、船 ship 和卡车 truck。每个样本都是一张 32*32 像素的 RGB 图像（彩色图像）。这 60000 个样本被分成了 50000 个训练样本和 10000 个测试样本。数据集的图像类别之间的差异相对较小，并且图像本身包含较多的噪声和复杂的背景，因此相较于 MNIST，CIFAR-10 数据集更加具有挑战性。

3.2 实验平台

硬件：本实验使用了 NVIDIA RTX 3050 GPU，确保了高效的训练过程，减少了训练时间。

软件：使用 PyTorch 作为深度学习框架，能够高效地执行反向传播和梯度更新操作。使用 CUDA 加速计算，提高了 GPU 性能。

3.3 实验结果分析

本实验的主要内容是通过比较四种经典优化方法：Momentum SGD、AdaDelta、Adam 和 RMSprop，分析它们在 MNIST 和 CIFAR-10 数据集上的收敛行为和泛化能力。

每种优化方法的超参数设置保持一致：

学习率：0.001。这个学习率值是经过初步试验后选择的，通常适用于大部分优化方法。

批次大小：64。较小的批次大小可以帮助模型更频繁地更新参数，增加训练的稳定性。

训练周期数：20 个 epoch。这一周期数在较为简单的数据集上（如 MNIST）已足够，且在 CIFAR-10 上也能达到较为稳定的训练效果。

采用两类**对比实验**：保持学习率、训练轮次等相同的情况下，横向对比 4 种优化方法在 MNIST 数据集和 CIFAR10 数据集上的表现：

3.3.1 动量 SGD

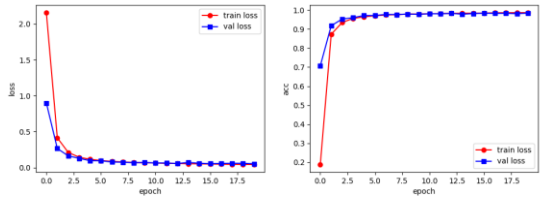

图 3-1 MNIST 数据集上动量 SGD 方法的损失值和准确率随训练轮次变化曲线

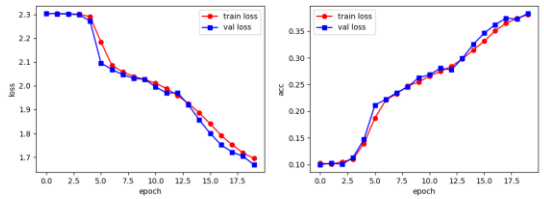

图 3-2 CIFAR10 数据集上动量 SGD 方法的损失值和准确率随训练轮次变化曲线

由图 3-1 分析可知：

在训练初期，损失值迅速下降，这表明模型快速学习到了数据的基本特征。随着训练的进行，损失值趋于平稳，说明模型逐渐收敛。

训练损失和验证损失曲线非常接近，这表明模型没有过拟合，泛化能力较好。

准确率在训练初期迅速上升，随后逐渐趋于平稳，最终达 98.34%，这表明模型在 MNIST 数据集上表现非常好。训练准确率和验证准确率曲线非常接近，进一步证实了模型的泛化能力。

由图 3-2 分析可知：

损失值的下降趋势与 MNIST 数据集相似，但下降速度较慢，这可能是由于 CIFAR-10 数据集的复杂性更高。训练损失和验证损失曲线在初期有较大差距，但随着训练的进行，差距逐渐缩小，表明模型在逐渐学习并改善泛化能力。损失值在训练后期趋于平稳，但下降幅度不如 MNIST 数据集显著。

准确率的上升趋势与 MNIST 数据集相似，但最终准确率较低，这同样可能是由于 CIFAR-10 数据集的复杂性。训练准确率和验证准确率曲线在初期有较大差距，

但随着训练的进行，差距逐渐缩小，表明模型在逐渐学习并改善泛化能力。

综合两个数据集的表现，发现：

CIFAR-10 数据集的图像比 MNIST 数据集的图像更复杂，这可能导致模型需要更多的训练轮次来学习特征，从而影响收敛速度和最终准确率。

LeNet-5 模型可能在 CIFAR-10 数据集上容量不足，无法捕捉到所有特征，这可能导致准确率提升有限。

动量 SGD 在处理复杂数据集时可能不如其他优化方法（Adam 和 RMSprop）有效，Adam 和 RMSprop 可能在调整学习率和梯度更新方面更灵活。

3.3.2 AdaDelta

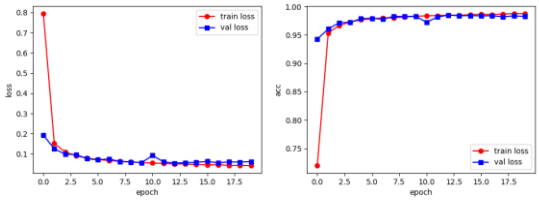

图 3-3 MNIST 数据集上 AdaDelta 方法的损失值和准确率随训练轮次变化曲线

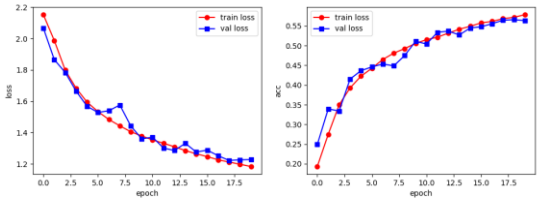

图 3-4 CIFAR10 数据集上 AdaDelta 方法的损失值和准确率随训练轮次变化曲线

由图 3-3 分析可知：

训练损失和验证损失都在前 3 个 Epoch 内快速下降，从 0.8 左右迅速降低到 0.1 以下，表明 AdaDelta 方法在 MNIST 数据集上表现出了很快的收敛速度。随着 Epoch 的增加，训练损失和验证损失趋于平稳，表明模型在后期的优化幅度变小。训练损失和验证损失几乎完全重合，表明模型在 MNIST 数据集上没有出现明显的过拟合现象，泛化性能良好。

训练准确率和验证准确率在前 3 轮快速上升，从接近 80%迅速达到接近 99%。准确率曲线在第 5 轮后几乎完全趋于饱和，表明模型能够很好地拟合 MNIST 数据集。最终训练准确率和验证准确率达 98.28%，说明 LeNet-5 结合 AdaDelta 优化方法在 MNIST 数据集上的表现极为优秀。

由图 3-4 分析可知：

训练损失和验证损失在前几个 Epoch 内迅速下降，但下降速度明显慢于 MNIST 数据集。随着训练轮次增加，损失值持续下降但始终未完全收敛，在第 10 轮后下降变缓。损失值曲线存在较大的波动现象，尤其是验证损失在某些 Epoch 出现了短暂的上升，说明模型在训练过程中泛化性能不稳定。

验证损失始终高于训练损失，表明模型存在一定的过拟合现象。

准确率在前几轮快速上升，但整体上升幅度小于 MNIST 数据集。验证准确率的增长在第 10 轮后趋于平缓，最终停留在 50%~60%之间。在 20 轮训练后，验证准确率远低于训练准确率，表明模型在 CIFAR-10 数据集上的泛化性能较差。

综合两个数据集的表现，发现：

在 MNIST 上，AdaDelta 表现出快速收敛和良好泛化能力，而在 CIFAR-10 上，收敛速度显著降低且泛化性能受限。

AdaDelta 在简单任务上具有较高的收敛效率，但在更复杂的任务中难以保证较强的优化能力。

AdaDelta 是一种自适应学习率优化方法，能够动态调整参数的学习率，避免了学习率过大或过小的问题，适合处理简单数据集。AdaDelta 依赖历史梯度和平方梯度的动态调整机制，在复杂的高维数据中可能难以找到最优解，导致模型在复杂数据集上收敛速度较慢。由于缺乏长远的优化路径探索能力，AdaDelta 容易陷入局部最优。

MNIST 数据集的损失值和准确率曲线非常平稳，而 CIFAR-10 数据集的损失值和准确率曲线存在较大波动。

3.3.3 Adam

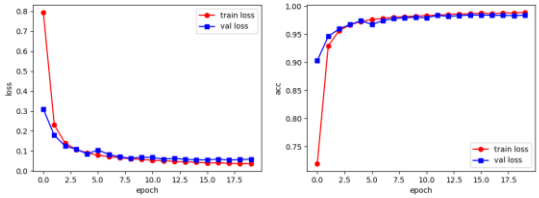

图 3-5 MNIST 数据集上 Adam 方法的损失值和准确率随训练轮次变化曲线

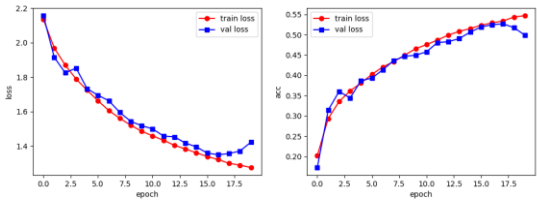

图 3-6 CIFAR10 数据集上 Adam 方法的损失值和准确率随训练轮次变化曲线

由图 3-5 分析可知：

Adam 优化器在 MNIST 数据集上的表现非常出色，损失值在最初的几个 epoch 迅速下降，表明模型快速学习到了数据的特征。

训练损失和验证损失曲线非常接近，这表明模型在训练过程中没有出现过拟合现象，泛化能力很好。

准确率在训练初期迅速上升，最终达 98.38%，这表明模型在 MNIST 数据集上的表现非常好。训练准确率和验证准确率曲线非常接近，进一步证实了模型的泛化能力

由图 3-6 分析可知：

在 CIFAR-10 数据集上，Adam 优化器同样表现出色，损失值在训练初期迅速下降，但下降速度相比 MNIST 数据集慢，这可能是由于 CIFAR-10 数据集的复杂性更高。训练损失和验证损失曲线在初期有轻微的差距，但随着训练的进行，差距逐渐缩小，表明模型在逐渐学习并改善泛化能力。

准确率的上升趋势与 MNIST 数据集相似，但最终准确率较低，这可能是由于 CIFAR-10 数据集的复杂性。训练准确率和验证准确率曲线在初期有轻微的差距，但随着训练的进行，差距逐渐缩小，表明模型在逐渐学习并改善泛化能力。

3.3.4 RMSprop

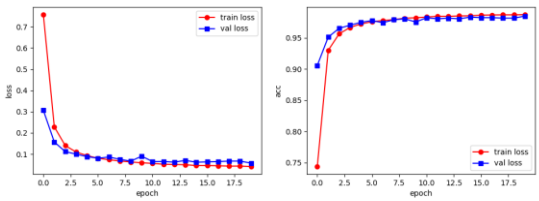

图 3-7 MNIST 数据集上 RMSprop 方法的损失值和准确率随训练轮次变化曲线

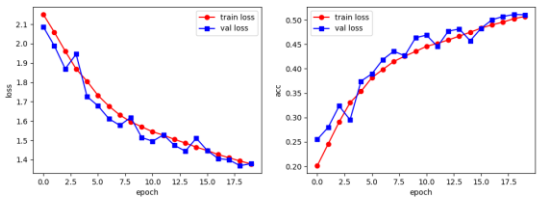

图 3-8 CIFAR10 数据集上 RMSprop 方法的损失值和准确率随训练轮次变化曲线

由图 3-7 分析可知：

训练损失和验证损失在前几轮快速下降，在大约第 5 轮时趋于平稳，并且二者非常接近，表明模型几乎没有过拟合现象，泛化能力较好。

损失值的下降趋势说明 RMSprop 方法在 MNIST 数据集上的收敛速度非常快，模型能快速找到较优解。

训练准确率和验证准确率在初始几轮快速上升，并在大约第 5 轮后趋于饱和。最终的训练和验证准确率都超过 98%（分别为 98.73%，98.50%），说明 LeNet-5 模型在 MNIST 数据集上训练效果非常好，RMSprop 优化方法在这个任务上性能优异。

由图 3-8 分析可知：

训练损失和验证损失整体呈下降趋势，但下降速度较 MNIST 数据集慢，且在整个训练过程中都未完全收敛。训练损失和验证损失存在明显的波动现象，验证损失的波动幅度较大，可能表明模型在某些阶段的泛化能力不稳定。

验证损失明显高于训练损失，表明模型在训练数据上的拟合效果优于测试数据。

训练准确率和验证准确率均呈上升趋势，但增长较为缓慢，验证准确率的最终值低于训练准确率。训练损失低于验证损失，且训练准确率高于验证准确率，这种差距表明模型可能存在一定程度的过拟合现象。

性能表现：在 20 轮训练后，验证准确率大约在 60% 左右，说明模型对更为复杂的 CIFAR-10 数据集的泛化能力有限。

综合两个数据集的表现，发现：

RMSprop 利用平方梯度的指数衰减平均值调整学习率，能够在初始阶段快速找到合适的学习步长，导致模型在相对简单的数据集上迅速收敛 RMSprop 方法在处理高噪声梯度时表现较好。但在更复杂的数据集上可能存在一定局限性，例如难以找到全局最优解或容易陷入局部最优。

3.3.5 综合分析

1. SGD
   优点：动量 SGD 通过历史梯度的加权平均，有效克服了 SGD 的震荡现象，提升了收敛速度，特别是在有较长平坦区域的情况下。
   缺点：虽然动量方法能加速收敛，但其学习率需要手动调整，且在某些情况下可能出现梯度更新过度的现象。
2. AdaDelta
   优点：AdaDelta 自动调节每个参数的学习率，无需手动调整初始学习率。它可以在训练过程中自适应地调整学习率，使得模型训练更加灵活。
   缺点：由于仅使用梯度的平方信息，AdaDelta 可能会在某些复杂问题中表现不如其他优化方法，且它对不同任务的适应性不如 Adam。
3. Adam
   优点：Adam 具有极好的收敛速度和稳定性，尤其适用于具有稀疏梯度的深度神经网络。其自适应的学习率使得在大多数任务中，Adam 表现出色。
   缺点：尽管 Adam 通常能较好地收敛，但在某些情况下，过度依赖于自适应的学习率可能导致模型在泛化能力上有所下降，特别是在训练数据较少时。
4. RMSprop
   优点：RMSprop 能自适应调整学习率，并且在训练过程中较为稳定，适合非凸优化问题。它尤其适用于处理稀疏数据和不稳定的梯度更新。

缺点：RMSprop 可能在训练过程中出现较大的波动，尤其在较复杂的任务中，它对超参数选择非常敏感。

4. 结论

4.1 研究意义

本课题主要聚焦于分析和比较不同优化方法（动量 SGD、AdaDelta、RMSprop 和 Adam）在 LeNet-5 模型训练中的性能表现，特别是在收敛速度、稳定性和泛化能力等方面。通过对比这些优化方法的优缺点，旨在为深度学习模型训练提供有力的指导，并探索如何通过选择合适的优化算法提高模型的训练效果和性能。

4.2 课题特点与方法

本研究的主要特点是：

优化算法的对比分析：我们选择了三种常用的优化方法——动量 SGD、AdaDelta、RMSprop 和 Adam，并在标准数据集（MNIST 和 CIFAR-10）上对它们进行广泛的实验分析。这些优化算法代表了不同的思想：动量 SGD 依赖于历史梯度的加权平均来加速收敛，AdaDelta 通过自适应调整学习率来避免梯度爆炸或消失，而 Adam 结合了动量和自适应学习率的优势，成为当前最为广泛使用的优化算法；RMSprop 会根据每个参数梯度的变化动态调整其学习率，并且能够很好地适应深度学习中梯度分布可能剧烈变化的情况，例如稀疏特征或高噪声环境。通过对梯度的平方值进行指数加权平均，RMSprop 在计算上高效，适合大规模神经网络。。

数据集的选择：使用 MNIST 和 CIFAR-10 两个典型数据集进行实验，分别对应较简单和较复杂的任务。这样，能够全面评估不同优化方法在不同难度任务中的适应性。

对比实验设计：通过对比实验，深入分析了不同优化算法在不同复杂程度的数据集上的训练过程、收敛性、泛化性及其超参数对结果的影响。

4.3 实验结果与分析

Adam 是综合表现最佳的优化算法，适合需要快速收敛和稳定训练的任务，特别是在深度神经网络和复杂数据集上表现优异。

RMSprop 表现较为平衡，适合对收敛速度有一定要求且目标函数梯度变化剧烈的任务。

AdaDelta 是稳定性最强的算法，适合稀疏高维数据环境，但在复杂任务上的性能可能不如 Adam。

动量 SGD 适合对泛化能力要求较高的任务，但其收敛速度和稳定性不如其他自适应优化方法。

尽管动量 SGD 和 AdaDelta 也在一定程度上表现出了不错的性能，特别是在 MNIST 数据集上，然而它们在较复杂任务上的优势并不如 Adam 明显。此外，动量 SGD 在某些情况下出现了震荡，影响了训练过程的稳定性。

4.4 不足与原因分析

尽管本研究提供了对不同优化方法的深入分析，但仍存在一些不足之处：

网络架构的局限性：LeNet-5 是一个较为简单的卷积神经网络架构，可能没有充分展示优化方法在更复杂网络中的效果。在未来的研究中，可以考虑在更深层次的网络（如 ResNet[2]、DenseNet[3]等）上进行优化方法的比较，进一步验证不同优化方法的适用性。

超参数的选择：由于设备的局限性，本研究中使用了固定的超参数（学习率、批量大小等），可能未能充分挖掘每种优化算法在不同超参数设置下的潜力。在未来的工作中，可以通过网格搜索或贝叶斯优化等方法来自动调整超参数，从而获得更好的训练效果。

复杂数据集的多样性：尽管我们使用了 CIFAR-10 作为复杂数据集，但它仍然是相对简单的图像分类任务。更复杂的图像数据集（如 ImageNet）或具有时序性质的数据（如视频分析、自然语言处理任务）可能会揭示不同优化方法的优势和局限性。

4.5 对未来工作的展望

未来的工作可以从以下几个方向进行深入探索：

在更复杂网络架构上的应用：可以将不同优化算法应用到更复杂的深度学习模型（如 ResNet、Transformer 等）中，研究它们在深度神经网络中的适应性与表现。

更多任务的扩展：除了图像分类任务，可以将优化算法应用到其他任务，如目标检测、语义分割、自然语言处理等，评估其在不同任务上的性能。

自适应优化算法的改进与创新：虽然 Adam 是当前最流行的优化算法，但在某些情况下，它的表现可能不如预期。因此，探索新的优化方法或改进现有方法（如基于元学习的自适应优化算法）可能会进一步提高深度学习模型的训练效率和泛化能力。

自动化超参数调优：通过自动化的超参数调优方法（如贝叶斯优化、遗传算法等）寻找最佳的学习率、动量因子和其他超参数，从而提升模型的训练效果和稳定性。

4.6 总结

本研究选择任务四，通过系统地对比和分析了动量 SGD、AdaDelta、RMSprop 和 Adam 三种优化方法在 LeNet-5 模型上的表现，得出了 Adam 优化算法在大多数情况下表现最优的结论。Adam 在收敛速度、训练稳定性和泛化能力方面的优势，使其成为深度学习任务中最常用的优化方法。此外，研究还揭示了几种方法的局限性，尤其在面对复杂任务时的性能不足。尽管本研究在某些方面存在一定局限，但其为选择合适的优化算法提供了宝贵的回顾与总结，并为未来的研究工作奠定了基础。

5. 参考文献

[1] Hossein Gholamalinezhad, Hossein Khosravi, “Pooling Methods in Deep Neural Networks, a Review”, *arXiv preprint arXiv:2009.07485*.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, *arXiv:1512.03385*

[3] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, “Densely Connected Convolutional Networks”, *arXiv:1608.06993*

[4] Sebastian Ruder, “An overview of gradient descent optimization algorithms”, a*rXiv:1609.04747*

[5] Matthew D. Zeiler, “ADADELTA: An Adaptive Learning Rate Method”, *arXiv:1212.5701*

[6] Diederik P. Kingma, Jimmy Ba, “Adam: A Method for Stochastic Optimization”, *arXiv:1412.6980*

[7] Dongpo Xu, Shengdong Zhang, Huisheng Zhang, Danilo P. Mandic, “Convergence of the RMSProp deep learning method with penalty for nonconvex optimization”, Neural Networks, Volume 139, 2021