随机过程在深度学习优化与模型泛化中的应用研究

张弛

2024 年 12 月 3 日

摘要

随着深度学习技术的迅猛发展，其在图像识别、自然语言处理和时间序列预测等领域取得了显著成果。然而，深度学习模型在训练过程中常面临过拟合、优化效率低下以及对新数据的泛化能力不足等挑战。随机过程作为一种有效的数学工具，可以为深度学习模型的优化与泛化提供新思路。本研究将探讨随机过程这一数学工具在深度学习中的应用，重点关注三点创新：首先，通过引入随机过程来优化随机梯度下降，这会使得我们能够动态调整学习率，提高模型的收敛速度和稳定性；其次，通过引入随机过程创新的 Dropout 方法可以有效防止过拟合，提升模型的泛化能力；最后，通过引人随机过程创新的数据特征提取方法来有效解决数据波动问题以提高预测精度。我们通过设计一系列实验，并对实验结果进行分析，最终验证基于随机过程的三项创新技术对深度学习模型具有良好的效果。这将为随机过程在深度学习中的应用提供实证支持，推动更高效、更准确的深度学习技术的发展。

关键词：随机过程、深度学习、梯度下降、Dropout、时间序列、模型泛化

Abstract

With the rapid development of deep learning technology, significant achievements have been made in fields such as image recognition, natural language processing, and time series prediction. However, deep learning models often face challenges such as overfitting, low optimization efficiency, and insufficient generalization ability to new data during the training process. Random processes, as an effective mathematical tool, can provide new ideas for optimizing and generalizing deep learning models. This study will explore the application of stochastic processes as a mathematical tool in deep learning, with a focus on three innovations: firstly, by introducing stochastic processes to optimize stochastic gradient descent, we can dynamically adjust the learning rate, improve the convergence speed and stability of the model; Secondly, the Dropout method, which introduces stochastic process innovation, can effectively prevent overfitting and improve the generalization ability of the model; Finally, by introducing innovative data feature extraction methods using stochastic processes, the problem of data fluctuations can be effectively addressed to improve prediction accuracy. We designed a series of experiments and analyzed the results to ultimately verify that the three innovative techniques based on stochastic processes have good effects on deep learning models. This will provide empirical support for the application of stochastic processes in deep learning, promoting the development of more efficient and accurate deep learning techniques.

Keywords: Random process, deep learning, gradient descent, dropout,time series，model generalization

1 引言

1.1 研究背景

深度学习已在计算机视觉、语音识别和自然语言处理等领域取得突破性成果，然而，随着数据规模和模型复杂度的增长，深度学习模型在训练效率和泛化能力方面的挑战逐渐突出。模型在大规模数据上训练通常需要大量计算资源，优化过程可能面临局部最优的困扰。同时，复杂模型容易过拟合训练数据，导致泛化能力不足，难以在新数据上表现稳定。如何提高模型的训练效率、避免过拟合、提升泛化能力，已成为深度学习研究中的重要课题。随机过程是数学中用于描述随时间或空间变化的随机现象的工具，广泛应用于统计学、金融学等领域。在深度学习中，随机过程可以为模型引入适当的随机性，帮助优化模型训练和提升泛化效果。例如，随机梯度下降（SGD）利用随机性在训练中抽取小批量数据来估计梯度，以减少计算量和跳出局部最优解。此外，Dropout 等正则化方法通过随机“丢弃”部分神经元，防止模型过拟合。同时，布朗运动能模拟市场中的波动性，进一步提升模型对复杂模式的捕捉能力。因此，随机过程在提升深度学习模型的效率与泛化能力方面具有广泛的应用潜力。

1.2 研究目的

本研究的目的是通过随机过程提升深度学习模型的优化效率和泛化能力，主要从以下三个创新点出发：

1. 优化随机梯度下降（SGD）：提出一种基于随机过程的自适应学习率调整方法，以提升 SGD 在训练效率和收敛速度上的表现。

2. Dropout 的创新方法：通过贝叶斯 Dropout 和动态丢弃概率，实现更高效的防过拟合策略，提升模型在新数据上的泛化能力。

3. 数据特征提取的创新方法：通过布朗运动来提高网络对数据特征波动的捕捉能力，进而提高模型这个整体的预测精度。通过对这些创新点的实验验证，我们希望证明随机过程能够有效提高深度学习模型的性能，使其在不同类型数据上具备更好的鲁棒性和适应性。

1.3 论文结构

本文结构如下：第二章简要介绍随机过程的理论基础及其在深度学习中的相关应用；第三章详细描述基于随机过程优化 SGD 的方法，包括具体实现细节与实验分析；第四章探讨基于随机过程的动态 Dropout 的方法，并设计实验验证其效果与分析结果；第五章介绍基于随机过程的数据特征提取创新方法，并设计实验验证其效果以及分析结果；第六章给出论文的结论，强调随机过程在深度学习模型优化与泛化中的关键作用。本研究通过三个实验，验证上述创新点在提升深度学习模型性能方面的有效性，为进一步探索随机过程在深度学习中的应用提供了参考。在本研究中，所有代码均已开源并托管于 GitHub：https://github.com/freastzc/drop-and-SGD-based-on-suijiguocheng。

2 随机过程在深度学习中的理论基础

2.1 随机过程概述

随机过程是描述系统随时间或空间随机变化的一种数学工具。数学上，随机过程通常定义为一个定义在时间集合上的随机变量序列 $\{X(t)\}_{t\in T}$，其中 T 表示时间集，可以是离散或连续的。根据不同的性质，随机过程可被分为多种类型，主要包括马尔可夫过程、泊松过程和布朗运动等，每种类型在不同应用场景中具备特定优势。

- 马尔可夫过程（Markov Process）[5]：马尔可夫过程是无记忆性质的随机过程，未来状态仅依赖于当前状态，与过去状态无关。其数学表达为 $\{X(t)\}_{t\in T} P\left(X_{t+1} \mid X_t, X_{t-1}, \ldots, X_0\right) = P\left(X_{t+1} \mid X_t\right)$。这种特性使得马尔可夫过程在状态转换和随机跳跃等领域广泛应用。在深度学习优化中，可以将梯度变化建模为马尔可夫过程，帮助动态调整学习率，从而提升模型的收敛速度。

- 贝叶斯推理（Bayesian Inference）[1] 贝叶斯推理是随机过程中一种强大的统计方法，用于结合先验知识和观测数据，在不确定性条件下更新对参数或系统状态的信念。其核心公式是：$P(\theta \mid D) = \frac{P(D|\theta)P(\theta)}{P(D)}$。我们可以利用贝叶斯推理的思想将其应用于机器学习领域。可以将 Dropout 中的丢弃概率视作随机变量，而非固定值。通过后验分布对丢弃率进行建模，使模型能够动态适应不同的数据分布。可以使模型在面对不确定性时具有更稳健的表现。

- 布朗运动 (Brownian Motion)[3]：布朗运动是一种连续的随机过程，其增量满足正态分布且相互独立，广泛用于金融模型中的资产价格预测。在深度学习的时间序列预测实验中，布朗运动可帮助对数据中的波动和趋势进行建模，为时间序列预测提供更稳定的基准。

2.2 深度学习中的随机性

深度学习模型中的随机性广泛存在于参数初始化、数据分割、优化算法等多个方面。合理地利用随机过程引入适当的随机性，不仅能提升模型的优化效率，还能帮助改善模型的泛化能力，适应复杂的数据分布。[7]

- 参数初始化中的随机性：在深度学习中，模型参数通常随机初始化，这种随机性帮助模型避免陷入局部最优解。在本研究中，随机初始化作为起始随机过程，结合后续的优化过程（如基于马尔可夫过程的学习率调整），可以更有效地在解空间中搜索更优解。

- 数据分割中的随机性：在小批量随机梯度下降（SGD）中，每次训练选择的数据集是随机的，避免了模型对单一数据分布的过度拟合。这种随机性使模型在训练过程中逐渐接近全局最优解。本研究将通过改进 SGD 方法，引入动态调整的随机过程，以提升模型的优化效果和训练效率。

- 模型优化中的随机性：SGD 和 Dropout 是深度学习中典型的优化方法，通过引入随机性来增强模型的训练效果。SGD 使用小批量样本估计梯度，减少计算量并跳出局部最优解；Dropout 则通过随机丢弃部分神经元，防止过拟合。本研究将通过实验验证基于随机过程的 Dropout 创新方法，展示随机性对模型泛化能力的正向作用。

在接下来的实验中，本文将从随机过程的角度出发，对深度学习中的随机性进行深入研究。具体而言，将通过三个实验验证随机过程在 SGD 优化、Dropout 创新、数据特征提取创新具体应用，并展示随机过程如何提升深度学习模型的优化效率和泛化能力。

3 优化随机梯度下降（SGD）

3.1 传统 SGD 及其局限性

随机梯度下降（SGD）是深度学习中广泛使用的优化方法，其基本原理是通过小批量样本对整体数据分布的梯度进行近似估计，以减少计算开销并逐步逼近全局最优解。SGD 的更新公式为 $\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta \mathcal{L}(\theta_t)$, 其中 θ 为模型参数，η[8] 为学习率，L 为损失函数。尽管 SGD 有效降低了计算成本，但它仍然面临几个局限性：

1. 固定学习率：
   SGD 的学习率通常是固定的，不适应不同训练阶段的需求。在初始阶段，较大的学习率有助于加速收敛，而在后期阶段，较小的学习率则更适合微调，但固定学习率难以满足这两种需求。

2. 易陷入局部最优：
   由于小批量梯度的随机性，SGD 容易受到局部最优点的影响，尤其是在非凸损失函数中。

3. 收敛速度慢：
   SGD 需要不断更新参数来接近全局最优解，因而训练过程较长，尤其在学习率设置不当的情况下，可能难以达到理想的收敛效果。为了克服这些局限性，我们引入随机过程方法来动态调整学习率，以提升模型的训练效率。

3.2 随机过程优化 SGD 的方法

在本研究中，我们提出了一种基于随机过程优化 SGD 的方法，是通过引入马尔可夫链动态调整学习率，以增强模型的适应性和训练效果。具体而言，我们将学习率视作一个马尔可夫过程，根据当前的损失值和梯度信息调整学习率，使模型能够在训练早期阶段使用较高学习率，以便快速探索参数空间，而在后期阶段逐渐减小学习率以精细调整。换言之：若损失下降速度缓慢，则增加学习率；若下降速度较快，则减小学习率。马尔可夫链模型定义为 [2]：$\eta_{t+1} = f\left(\eta_t, \mathcal{L}\left(\theta_t\right), \nabla_\theta \mathcal{L}\left(\theta_t\right)\right)$

其中，η_{t+1} 表示下一个训练步的学习率，函数 f 是基于马尔可夫过程的学习率更新策略。我们通过设置一系列状态（如高、低、适中）来模拟学习率变化，确保学习率能够动态响应模型的训练状态，以提高训练效率和收敛性能。该策略在模型训练过程中能够动态适应不同训练阶段的需求，提升训练效率并降低损失。下面一节是有关实验对上述理论的验证：

3.3 实验

3.3.1 实验设计

- 实验目的：本实验的主要目的是通过对比固定学习率和基于随机过程马克罗夫链模型动态学习率策略，来验证引入随机过程是否能提高训练效率和模型性能。

- 实验数据与模型结构：本实验使用的数据集 MNIST 数据集，是一个大型的手写数字数据库。包含 60,000 张手写数字训练图像和 10,000 张测试图像。每张图像都是由大小为 28x28 像素的灰度图像组成。具体示例可以见图1：

图 1: MNIST 数据集图像

在本实验中，我们使用一个自定义的卷积神经网络模型来对 MNIST 数据集进行分类。由于 MNIST 数据集较为简单，所以模型的结构定义也无需复杂，能足够满足实验需求就可以。同时设计成自定义的简单模型也有助于在之后实现更多有效的改进，也方便其他研究者的复现。以下是模型的结构与详细说明：卷积层 1：包含 32 个 3x3 的卷积核，步长为 1，带有 ReLU 激活函数。该层主要用于初步提取图像的特征。最大池化层 1：采用 2x2 池化，减少特征图的尺寸并保留重要特征。卷积层 2：包含 64 个 3x3 卷积核，同样带有 ReLU 激活函数，进一步提取图像的复杂特征。最大池化层 2：同样采用 2x2 池化，进一步压缩特征图的尺寸。全连接层 1：包含 128 个神经元，用于在高层特征空间中学习分类决策。输出层：包含 10 个神经元（对应 MNIST 数据集的 10 个类别），直接输出分类结果。具体模型实现代码可见下图2：

```
class SimpleCNN(nn.Module):  2 usages
    def __init__(self):
        super(SimpleCNN, self).__init__()
        self.conv1 = nn.Conv2d( in_channels: 1,  out_channels: 32, kernel_size=3, stride=1, padding=1)
        self.conv2 = nn.Conv2d( in_channels: 32,  out_channels: 64, kernel_size=3, stride=1, padding=1)
        self.fc1 = nn.Linear(64 * 7 * 7,  out_features: 128)
        self.fc2 = nn.Linear( in_features: 128,  out_features: 10)

    def forward(self, x):
        x = torch.relu(self.conv1(x))
        x = torch.max_pool2d(x,  kernel_size: 2)
        x = torch.relu(self.conv2(x))
        x = torch.max_pool2d(x,  kernel_size: 2)
        x = x.view(-1, 64 * 7 * 7)
        x = torch.relu(self.fc1(x))
        return self.fc2(x)
```

图 2: 自定义模型架构代码展示

- 评价指标：为了对比固定学习率和基于随机过程的动态学习率策略的效果，本实验定义了五个指标，并通过这五个指标来帮助我们判断哪种策略更为有效。

  1. 训练损失
     定义：在训练集上每个 epoch 的平均损失值。
     意义：训练损失衡量模型在训练集上的拟合情况。较低的训练损失通常表明模型能够更好地学习训练数据的模式。
     判断标准：在相同 epoch 数下，较低的训练损失意味着策略更有效。
  2. 验证损失
     定义：在验证集上每个 epoch 的平均损失值。
     意义：验证损失可以反映模型的泛化能力。如果验证损失远高于训练损失，则模型可能出现过拟合。理想情况下，验证损失应该逐步下降并保持在较低值。
     判断标准：验证损失更低或更早趋于稳定的策略，通常被认为更有效。
  3. 收敛速度
     定义：模型达到某个预定损失水平或目标精度的所需 epoch 数。
     意义：较快的收敛速度通常意味着模型可以在更短时间内达到合理性能，这动态学习率策略的主要优势。
     判断标准：如果动态学习率策略比固定学习率策略在更少的 epoch 内达到相似的训练或验证损失，说明动态学习率策略收敛速度更快。
  4. 最终精度
     定义：在测试集上训练结束后的准确率。
     意义：最终精度可以评估模型的泛化性能。更高的精度表明模型在测试数据上的预测能力更强。
     判断标准：最终精度越高，说明该学习率策略在训练过程中提升了模型的泛化能力。

  5. 学习率变化趋势
     定义：每个 epoch 的学习率变化趋势图。
     意义：通过观察动态学习率的变化趋势，可以直观了解策略调整是否合理。合适的调整应在训练初期提高学习率以快速收敛，后期降低学习率以细化模型参数。
     判断标准：如果动态学习率在训练中逐步下降并在适当的阶段收敛，说明其调整策略更合理。

- 实验结果展示：

  1. 首先是训练损失这一个评判标准的对比图3：
     可以很清晰地看出在 10 轮 epoch 中，基于随机过程的动态学习率策略与固定学习率策略相比，训练损失明显更低。由此可以表面表明基于随机过程的动态学习率策略能够更好地学习训练数据的模式。
  2. 其次是验证损失这一评判标准的对比图4：
     可以很清晰地看出在 10 轮 epoch 中，基于随机过程的动态学习率策略与固定学习率策略相比，验证训练损失明显更低。由此可以表面表明基于随机过程的动态学习率策略在模型的泛化能力上表现地更好。

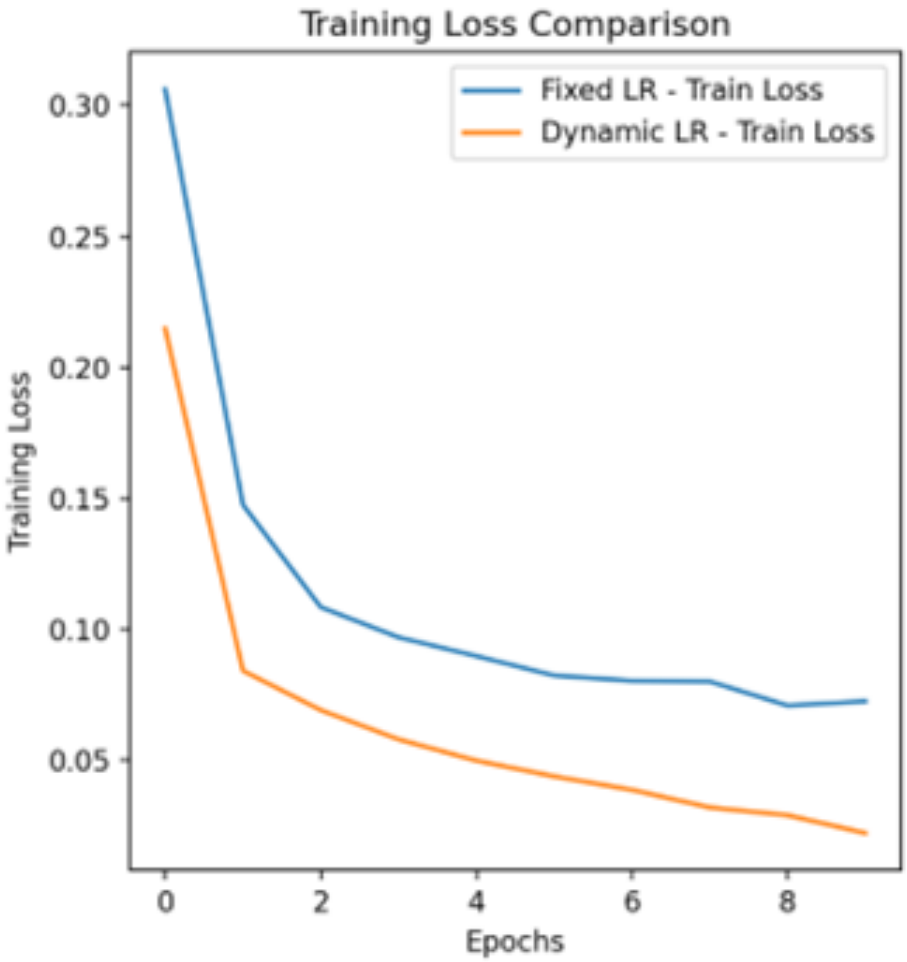
Training Loss Comparison
Fixed LR - Train Loss
Dynamic LR - Train Loss
Training Loss
0.30
0.25
0.20
0.15
0.10
0.05
0
2
4
6
8
Epochs

图 3: 训练损失值对比图

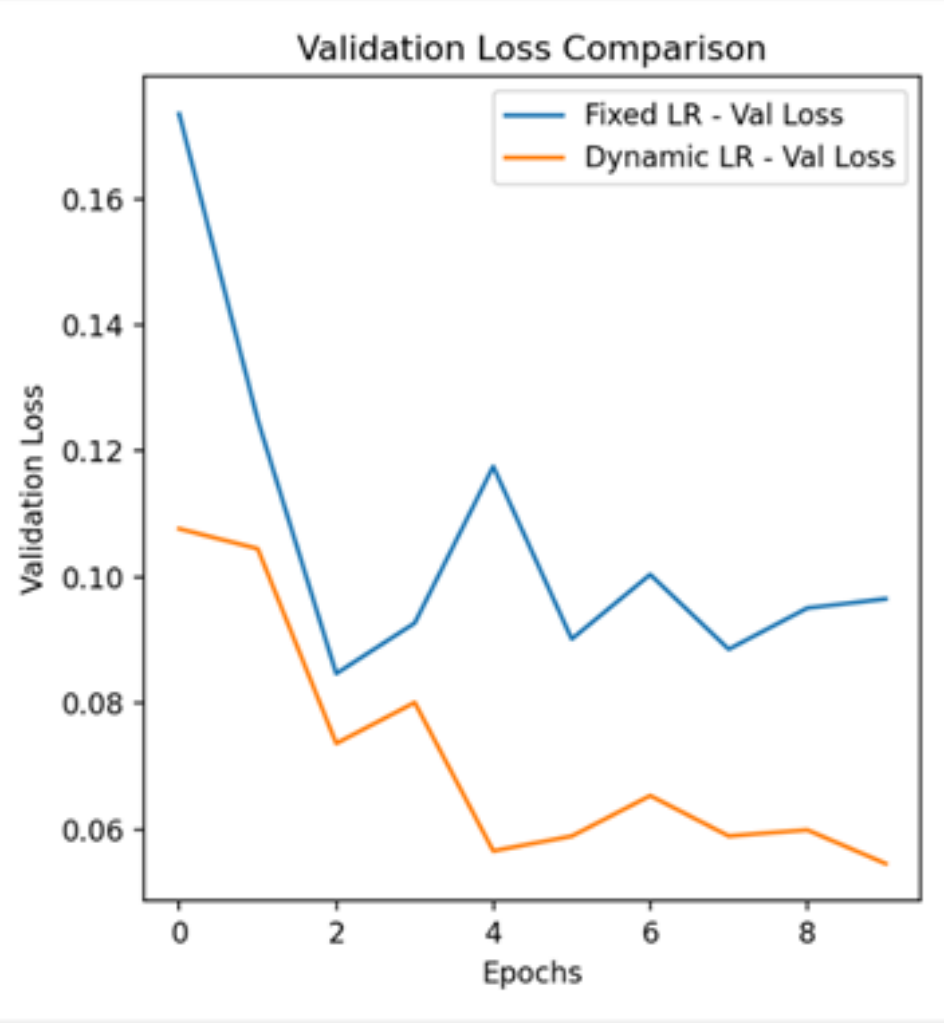
Validation Loss Comparison
Fixed LR - Val Loss
Dynamic LR - Val Loss
Validation Loss
0.16
0.14
0.12
0.10
0.08
0.06
0
2
4
6
8
Epochs

图 4: 验证损失值对比图

而且总结两种张对比图也很直观地看出在收敛速度这一评判标准上，基于随机过程的动态学习率策略也更胜一筹。

3. 之后是学习率变化趋势这一评判标准图5:

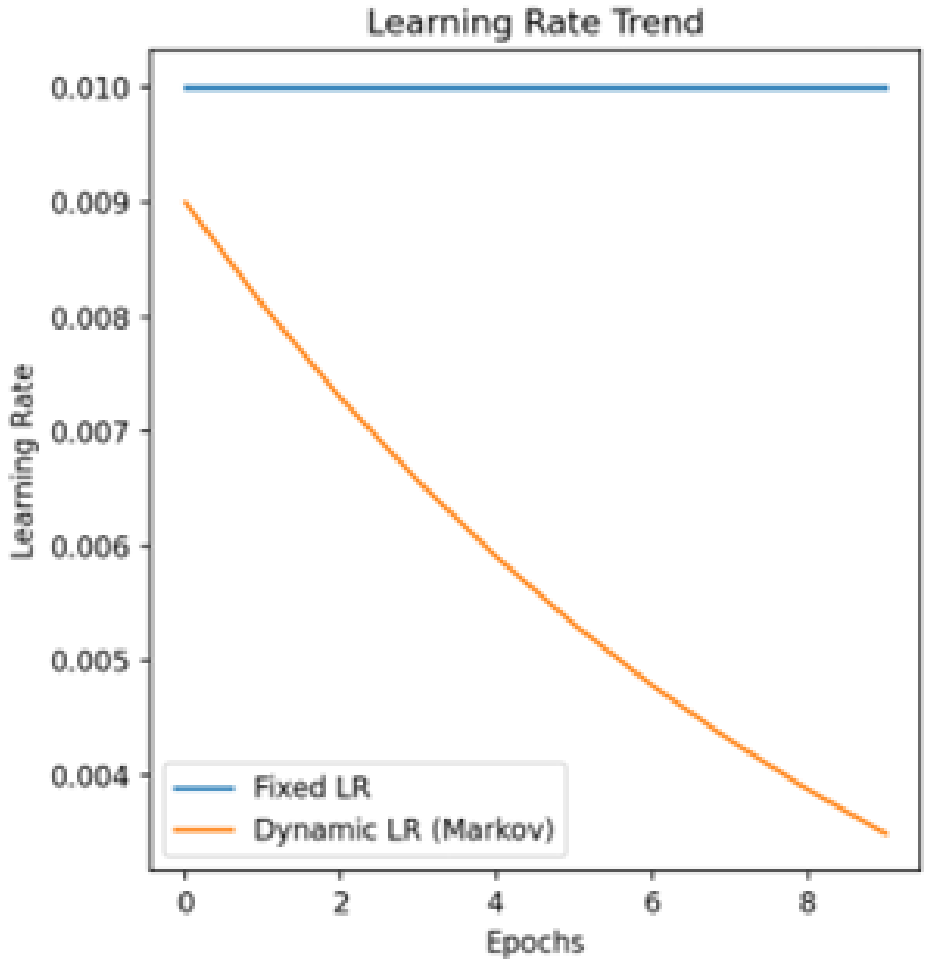

图 5: 学习率变化趋势图

在可视化图中可以看出训练初期为了快速收敛，学习率比较大，后期降低学习率以细化模型参数。所以基于随机过程的动态学习率策略可以正确地调整学习率来完成实验，达到目标。

4. 最终精度的对比数据如图6:

```
Final Accuracy - Fixed LR: 0.9716
Final Accuracy - Dynamic LR: 0.9855
```

图 6: 最终精度对比图

从图中的信息可以看出基于随机过程的动态学习率策略比固定学习率策略的最终精确度更高，由此说明该学习率策略在训练过程中提升了模型的泛化能力。

3.3.2 实验结果分析

实验结果显示，动态学习率策略在上述五个评判指标上都具有明显优势，表明基于随机过程的动态学习率策略具有助于训练过程优化，提升模型的泛化能力等优势。既然基于随机过程的动态学习率策略于具有良好的泛化能力，那在面对复杂性高、训练时间长的神经网络时，有可能也能发挥出良好的作用。在未来的研究中可以使用在其他更复杂的数据集中，用于提升训练的速度，测试的精度等。但也有不足之处，该策略在验证集训练时，产生了震荡，这就说明学习率调整不当反而影响模型收敛稳定性。未来改进方面可以就这方面调整算法，降低局限性，使其能够更好的应用于不同的任务和模型。

4 Dropout 的创新方法

4.1 Dropout 的传统应用

Dropout 是一种有效的正则化技术，旨在减少深度学习模型的过拟合 [6]。其基本思想是在每次训练迭代中随机“丢弃”一部分神经元（即将它们的输出设置为零），使得模型在每次更新中只能依赖于一部分特征，从而增强模型的鲁棒性。具体来说，Dropout 的工作机制如下：

- 随机丢弃
  在每次训练过程中，以一定的丢弃率随机选择神经元进行丢弃，通常在训练期间丢弃率设置为 0.2 到 0.5 之间。
- 特征共享
  通过随机丢弃，模型学习到的特征不再依赖于特定的神经元，使得模型在面对新数据时更具适应性和稳定性。
- 集成学习效果
  Dropout 可视为一种集成学习方法，因为每次训练都在训练一个不同的子网络，最终通过平均化不同子网络的预测结果来提升泛化能力。

然而，传统的 Dropout 方法在实际应用中存在一些局限性，主要是丢弃率的固定性。在训练过程中，固定的丢弃率可能无法适应不同的学习阶段和数据特性，从而限制了模型的进一步优化。

4.2 基于随机过程的创新方法

为了克服传统 Dropout 的局限性，本研究提出了一种基于随机过程的创新方法——贝叶斯 Dropout[1]。该方法通过动态调整丢弃概率来提升模型的泛化能力，具体包括以下几个方面：

1. 贝叶斯框架：
   贝叶斯 Dropout 利用贝叶斯推理的思想，将丢弃概率视作随机变量，通过后验分布对丢弃率进行建模。在训练过程中，模型不仅学习参数，还同时学习丢弃概率的分布。$P(\theta \mid D) = \frac{P(D|\theta)P(\theta)}{P(D)}$，其中 $P(\theta \mid D)$ 表示：后验分布，表示结合数据 θ 后对参数 D 的信念。$P(D \mid \theta)$ 表示：似然函数，描述在假定参数为 θ 时数据 D 出现的概率。$P(\theta)$ 表示：先验分布，表示在观测之前对参数　的初始假设。$P(D)$ 表示：归一化常数，也称为边际似然。

2. 动态调整丢弃概率：
   引入随机过程来根据当前模型性能和训练阶段动态调整丢弃概率。例如，在模型表现较差时，可以增加丢弃率以增强模型的鲁棒性；而在模型表现良好时，则降低丢弃率，以保留更多的特征信息。这种自适应的丢弃策略使得模型在不同阶段都能够最大化泛化能力。

3. 提升模型泛化能力：
   通过随机过程动态调整丢弃概率，贝叶斯 Dropout 可以有效减少模型的过拟合，增强在新数据上的表现。此外，模型在训练过程中能够更灵活地适应数据的变化，提升学习效果。

4.3 实验

4.3.1 实验设计

- 实验目的：
  本实验是将基于随机过程的贝叶斯 Dropout 与传统固定的丢弃率应用于同一个深度神经网络模型中，为了探究其中使用了随机过程的方法的是否能够有效地提高模型的泛化能力。

- 实验数据和模型架构：
  据集依旧是使用的 MNIST 数据集，这里就不多赘述。以下是模型的具体架构：该模型结构包含三层卷积网络，并在每层卷积后加入 Dropout 层。对于使用固定丢弃率策略和基于随机过程的贝叶斯 Dropout 策略的两个对比实验，它们的模型架构完全相同，只是在 Dropout 率处理上有所不同。
  输入层：接收 $1 \times 28 \times 28$ 的灰度图像，适用于 MNIST 数据集。
  卷积层 1：包含 32 个 3×3 卷积核，步长为 1，带有 ReLU 激活函数，用于初步提取图像特征。接着加入 Dropout 层，传统方法的丢弃率固定为 0.5，创新方法动态调整丢弃率。
  最大池化层 1：2×2 池化窗口，步长为 2，用于下采样特征图，降低特征图尺寸。
  卷积层 2：包含 64 个 3×3 卷积核，步长为 1，带有 ReLU 激活函数，进一步提取复杂特征。
  接入 Dropout 层，丢弃率同样为固定或动态调整。
  最大池化层 2：2×2 池化窗口，步长为 2，再次下采样特征图，减少模型计算量。
  全连接层 1：将展平的特征图输入到 128 个节点的全连接层，使用 ReLU 激活函数，并加入 Dropout 层（固定或动态丢弃率）。
  输出层：全连接层输出 10 个节点（对应数字 0-9 的分类任务），用于最终分类。
  模型的具体代码如图7：

- 评价指标:
  和第三章实验一样，其中第三章的学习率变化趋势变成了丢弃率变化趋势，所谓丢弃率变化趋势在本实验中具体如下：
  定义：在训练过程中随机丢弃神经元的概率，并在训练过程中随着每个 epoch 的损失值和准确率变化而改变。
  意义：丢弃率的改变可以反映模型在不同的阶段在以不同的方式学习。
  评判标准：动态丢弃率的调整应当在一定范围内进行，避免频繁变化导致训练不稳定。理想的情况是丢弃率在训练过程中表现出逐渐收敛的趋势，而非剧烈波动。

- 实验结果展示

```python
class SimpleCNNWithDropout(nn.Module):  3 usages
    def __init__(self, dropout_prob=0.5):
        super(SimpleCNNWithDropout, self).__init__()
        self.conv1 = nn.Conv2d( in_channels: 1,  out_channels: 32, kernel_size=3, stride=1, padding=1)
        self.dropout1 = nn.Dropout(dropout_prob)
        self.conv2 = nn.Conv2d( in_channels: 32,  out_channels: 64, kernel_size=3, stride=1, padding=1)
        self.dropout2 = nn.Dropout(dropout_prob)
        self.fc1 = nn.Linear(64 * 7 * 7,  out_features: 128)
        self.dropout3 = nn.Dropout(dropout_prob)
        self.fc2 = nn.Linear( in_features: 128,  out_features: 10)

    def forward(self, x):
        x = torch.relu(self.conv1(x))
        x = self.dropout1(x)
        x = torch.max_pool2d(x,  kernel_size: 2)
        x = torch.relu(self.conv2(x))
        x = self.dropout2(x)
        x = torch.max_pool2d(x,  kernel_size: 2)
        x = x.view(-1, 64 * 7 * 7)
        x = torch.relu(self.fc1(x))
        x = self.dropout3(x)
        return self.fc2(x)
```

图 7: 自定义模型架构图

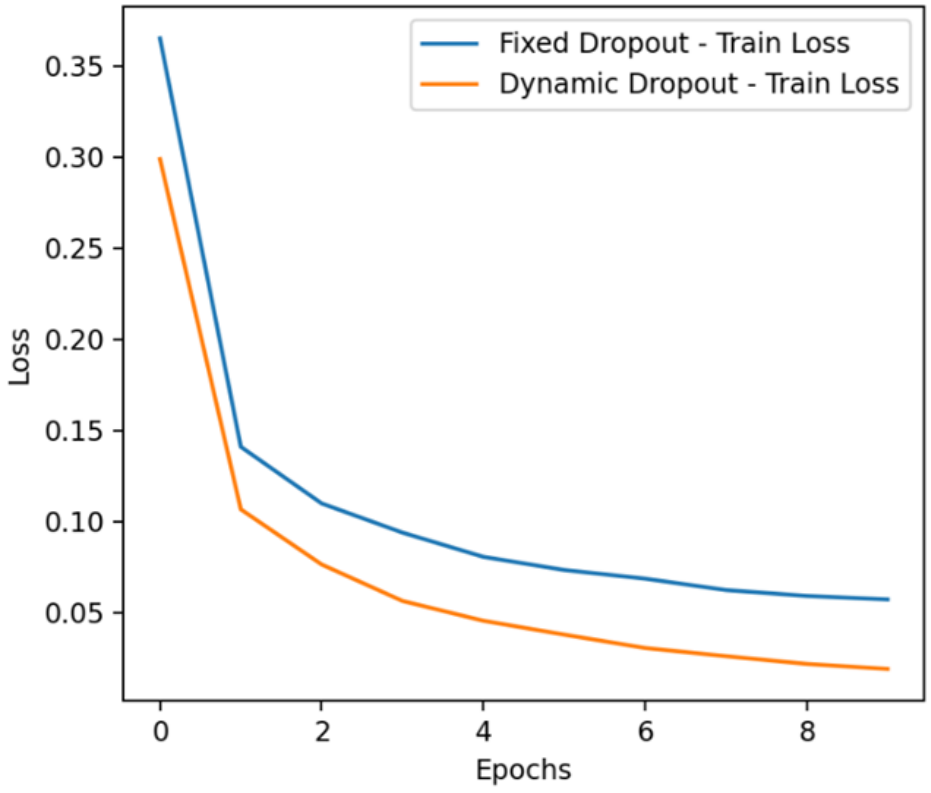

图 8: 训练损失对比图

1. 首先是训练损失这一个评判标准的对比图8：
   可以清晰基于随机过程的动态丢弃率策略的损失值低于固定丢弃率策略，说明基于随机过程的动态丢弃率策略能够更好地学习训练数据的模式。
2. 验证损失这一个评判标准的对比图9：

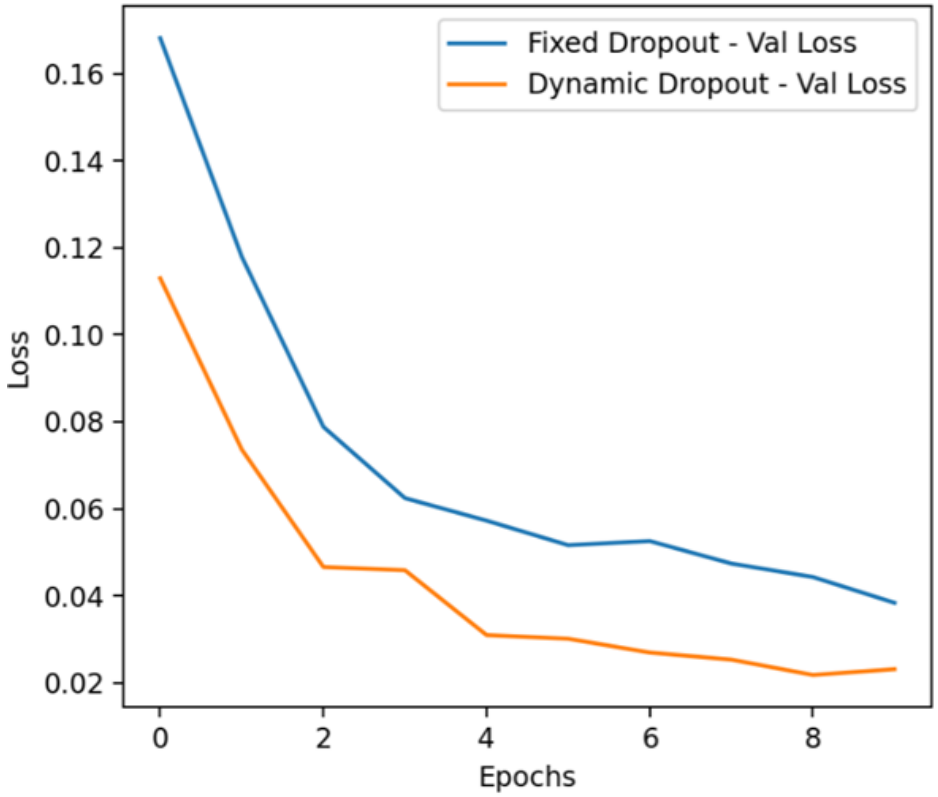

图 9: 验证损失对比图

   可以清晰看出基于随机过程的动态丢弃率策略的验证损失值低于固定丢弃率策略。由此可以表面表明基于随机过程的动态丢弃率策略在模型的泛化能力上表现地更好。而且总结两种张对比图也很直观地看出在收敛速度这一评判标准上，基于随机过程的动态丢弃率策略也更胜一筹。
3. 丢弃率变化趋势对比图10：
   可以看出基于随机过程的动态丢弃率策略的丢弃率随 epoch 逐渐降低，在损失值高的 epoch，丢弃率高，在损失值低的 epoch，丢弃率低，同时没有出现震荡的情况，而是逐渐收敛。
4. 最后是准确率的对比图11和12：
   可以清晰看出无论是在测试集还是在验证集上，基于随机过程的动态丢弃率策略的准确率都更高，表明该策略在训练过程中提升了模型的泛化能力。

4.3.2 实验结果分析

实验结果显示，基于随机过程的动态丢弃率策略在训练过程中表现出明显优势，特别是在五个评判指标上均取得了优异的成绩。这一结果表明，动态丢弃率策略不仅有效优化了训练过程，还

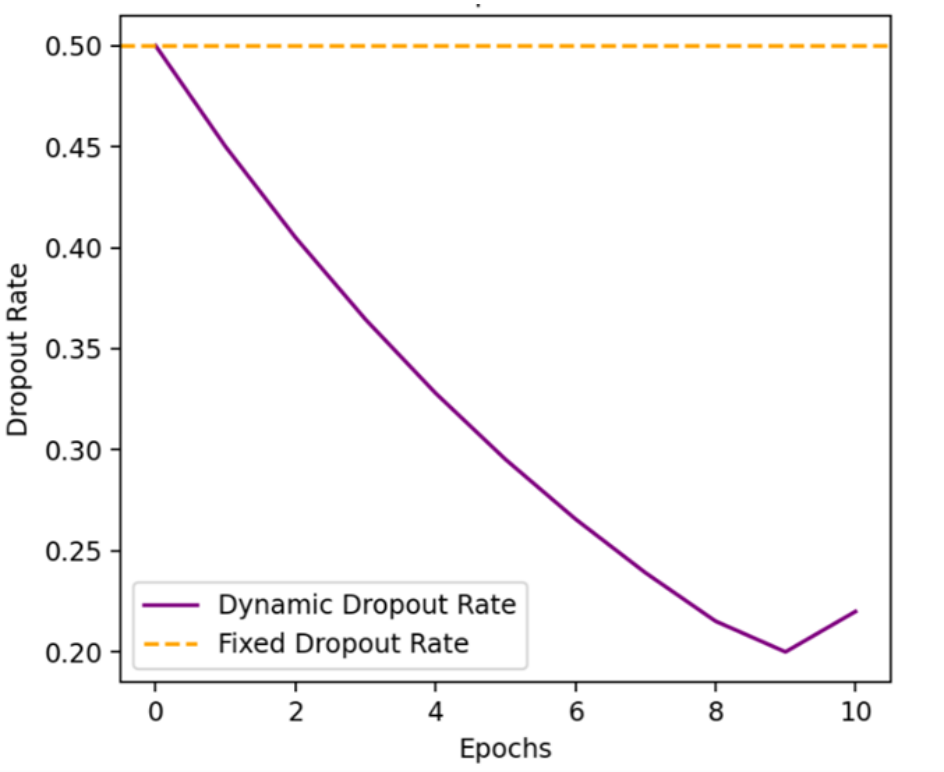

图 10: 丢弃率变化趋势对比图

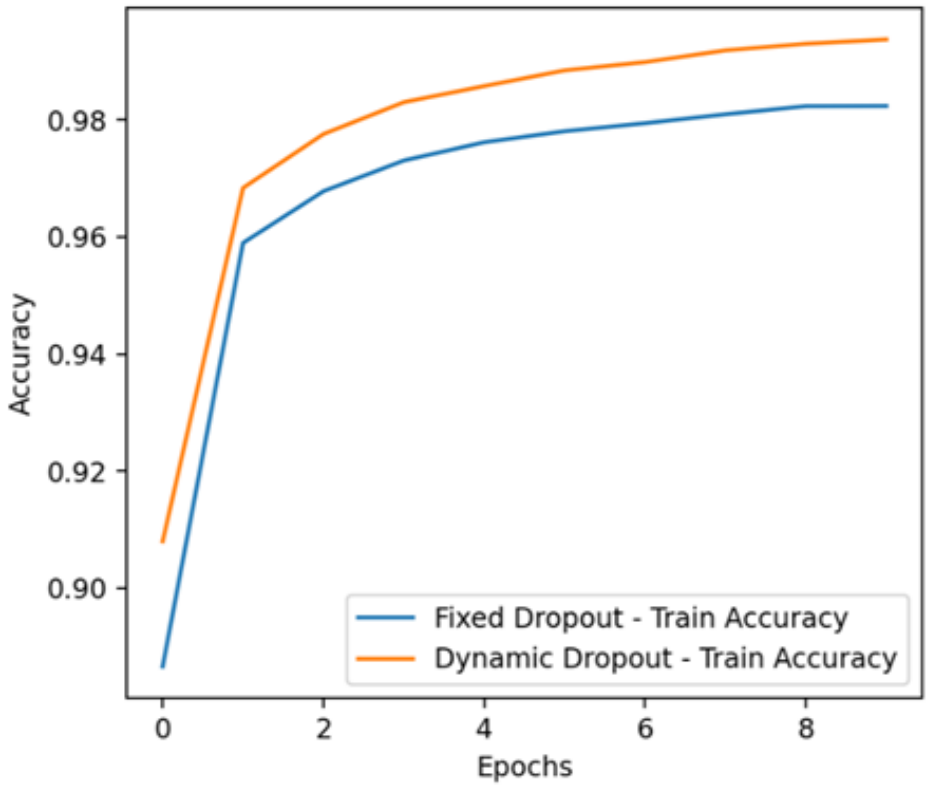

图 11: 训练集准确率对比图

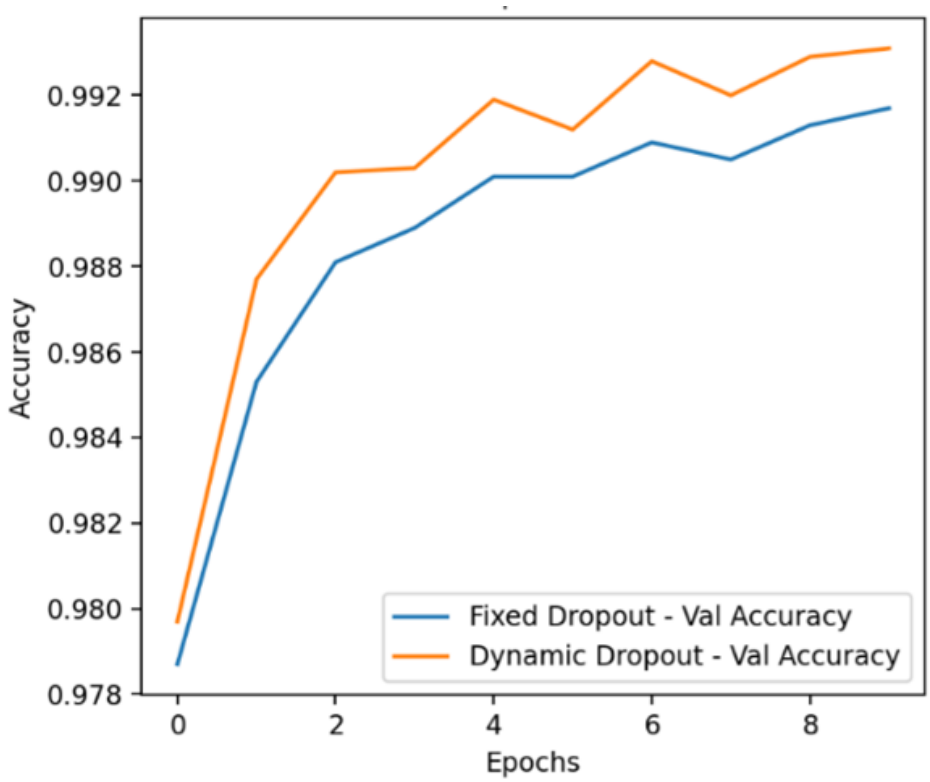

图 12: 验证集准确率对比图

显著提升了模型的泛化能力。这一优势使得模型在面对新数据时更加鲁棒，减少了过拟合的风险。未来，该策略有着广阔的应用前景。首先，可以将动态丢弃率策略推广到其他深度学习模型和任务中，以进一步验证其泛化能力的有效性。例如，在图像分类、自然语言处理和时间序列预测等领域，应用这一策略可能会带来显著的性能提升。此外，研究更为智能的算法，如通过在线学习或进化算法来动态优化丢弃率，将有助于进一步提升模型的自适应能力，使模型能够根据训练进度和数据特性进行实时调整，达到更优的效果。然而，值得注意的是，该策略在实施过程中可能面临一些局限性。由于需要实时动态计算丢弃率，这可能导致额外的计算资源消耗，从而在大规模数据集上增加训练时间。这一问题在处理非常复杂的模型或海量数据时尤为明显，可能影响训练效率。因此，如何在保持模型性能的同时，降低计算开销，将是未来研究的重要方向。解决这一局限性的方法可能包括算法优化、计算资源的合理配置或使用更高效的硬件，以确保在大规模应用中的可行性。综上所述，动态丢弃率策略展示了其在优化训练过程和提升模型泛化能力方面的潜力，同时也为后续研究提供了丰富的探索空间和挑战。

5 优化数据特征提取

5.1 传统特征提取策略

传统特征提取策略主要基于卷积神经网络设计，其核心思想是通过卷积层提取时间序列中的局部特征，如趋势、周期性波动等，用这些特征预测未来的资产价格。这种方法已在许多时间序列预测任务中取得显著效果，尤其是在具有明显周期性或结构化特征的数据中。然而，对于金融市场这种高度非线性和随机性的环境，传统策略存在一定局限性。

1. 忽略波动性特征：
   金融市场的一个重要特征是价格的随机波动性，这种波动往往被视为噪声，而未被纳入建模范围。传统 CNN 模型难以处理这些高频、不规则的波动特性。

2. 非平稳性挑战：
   金融时间序列通常表现为非平稳性，即统计特性（如均值、方差）会随着时间变化。传统卷积网络的固定滤波器难以动态适应数据的非平稳性。

3. 对随机性的适应性不足：
   在高度不确定性的数据中，传统模型可能过度拟合历史模式，而无法对未来的随机波动进行准确预测。

5.2 基于随机过程的创新策略

为克服传统策略在处理金融市场随机性和波动性方面的不足，本文提出了一种基于随机过程的创新策略，将布朗运动（Brownian Motion）层引入到 CNN 中，以增强模型对波动性的建模能力。这一创新策略不仅可以提取了时间序列的局部特征，还显式地模拟了市场的随机波动性和非平稳性，从而大幅提升了模型的预测能力。

传统 CNN 模型主要提取局部特征，其数学表示为：[4] $h_i = f\left(W \cdot x_i + b\right)$, 其中 h_i 是卷积后的特征,x_i 是输入数据的局部窗口,W 是卷积核权重，$f\left(\right)$ 是激活函数。卷积操作只能捕捉输入的局部模式，难以对高频的价格波动建模。例如, 短期随机波动部分 η_t 会被视为噪声，被忽略或平滑掉：$x_t = \text{Trend}(t) + \eta_t, \quad \eta_t \sim \mathcal{N}\left(0, \sigma^2\right)$ 这里的随机波动项 η_t 未被有效建模，因此传统策略难以捕捉价格的波动性特征。而布朗运动层将随机波动显式建模为金融市场的重要组成部分。布朗运动的数学模型为：$S_t = S_{t-1} + \mu\Delta t + \sigma\Delta W_t$, 其中 S_t 为资产价格。μ：漂移项，代表价格的长期趋势。$\sigma\Delta W_t$：波动项。$\Delta W_t \sim \mathcal{N}\left(0, \sigma^2\right)$: 用于描述短期随机波动。通过在 CNN 模型中加入布朗运动层，输入特征不再仅由原始数据提供，而是通过布朗运动模拟增强了波动性建模：$\tilde{x}_t = x_t + \sigma\Delta W_t$

5.3 实验

5.3.1 实验设计

- 实验目的：
  本实验旨在通过将布朗运动引入卷积神经网络，提高模型在金融时间序列数据中的预测性能，特别是在模拟和捕捉金融市场的波动性方面。我们将布朗运动作为一种模拟金融资产价格波动的工具，嵌入到传统的 CNN 模型中，以期提升模型的适应能力和预测精度。我们通过与传统 CNN 模型的对比，探索布朗运动层对资产价格预测的贡献，并验证其在金融预测任务中的有效性。

- 实验数据和模型结构：
  实验数据来自于 Apple Inc.（AAPL）的历史股价，时间范围从 2015 年 1 月 1 日到 2023 年 1 月 1 日。数据主要包含每日的收盘价，这些股价数据经过 Min-Max 归一化处理，将股价数据缩放到 [0, 1] 范围内，确保模型能够高效地学习。为了生成时间序列数据，每个数据点使用了 20 天的历史股价数据（即序列长度为 20），每个序列的目标值是该序列之后的下一天的股价。对于两个模型的对比实验，两者只有在卷积层有所不同，其他结构保持一致形成对照：

1. 传统 CNN 模型结构
   输入层：形状为 (20, 1)，表示过去 20 天的股价。
   卷积层：1D 卷积，32 个滤波器，卷积核大小为 3，激活函数为 ReLU。
   平展层：将卷积输出展平成一维。
   全连接层：50 个神经元，激活函数为 ReLU。
   输出层：线性层，输出未来一天的股价。
2. 带布朗运动层的 CNN 模型结构
   输入层：与传统 CNN 相同，输入为过去 20 天的股价。
   布朗运动层：模拟随机波动，通过累加噪声增强输入数据的波动性。
   卷积层：与传统 CNN 相同，32 个滤波器，卷积核大小为 3，激活函数为 ReLU。
   平展层：将卷积输出展平成一维。
   全连接层：50 个神经元，激活函数为 ReLU。
   输出层：线性层，输出未来一天的股价。

模型的具体代码如下图所示：

1. 传统模型结构：
2. 带布朗运动层的 CNN 模型结构：

- 评估标准: 为了评价模型的性能，我们使用了以下几种常见的评估指标：
  - 均方误差 (MSE)
    MSE 用于衡量预测值与真实值之间的差异。MSE 越小，说明模型的预测精度越高。
  - 平均绝对误差 (MAE)
    MAE 是另一个常用的回归任务评估指标，表示预测值与真实值之间的平均绝对误差。它能直观地反映模型的预测偏差。
  - 预测误差分布
    通过分析两种模型的预测误差分布，我们可以直观地看到两种模型在预测精度上的差异。如果带有布朗运动层的模型预测误差更集中或更小，说明布朗运动层对模型的优化效果显著。
  - 训练损失曲线
    通过可视化模型训练过程中的损失曲线，我们可以比较模型训练的收敛速度和稳定性。更低的训练损失通常表示模型在学习过程中收敛得更快，且训练更加稳定。
- 实验结果展示
  1. 首先是 MSE 和 MAE 这两个评估标准的对比图13：

```
带有布朗运动的 CNN 模型 - 测试集 MSE: 0.0002, MAE: 0.0084
传统 CNN 模型 - 测试集 MSE: 0.0012, MAE: 0.0249
```

图 13: MSE 和 MAE 对比图

可以从实验结果数据中清晰地看出带有布朗运动的 CNN 模型对比于传统的 CNN 模型

无论是 MSE 还是 MAE，都是前者在数值上更小，说明其的预测值误差更小，对模型的优化效果显著。

2. 其次是预测误差分布这一评估标准的对比图14：

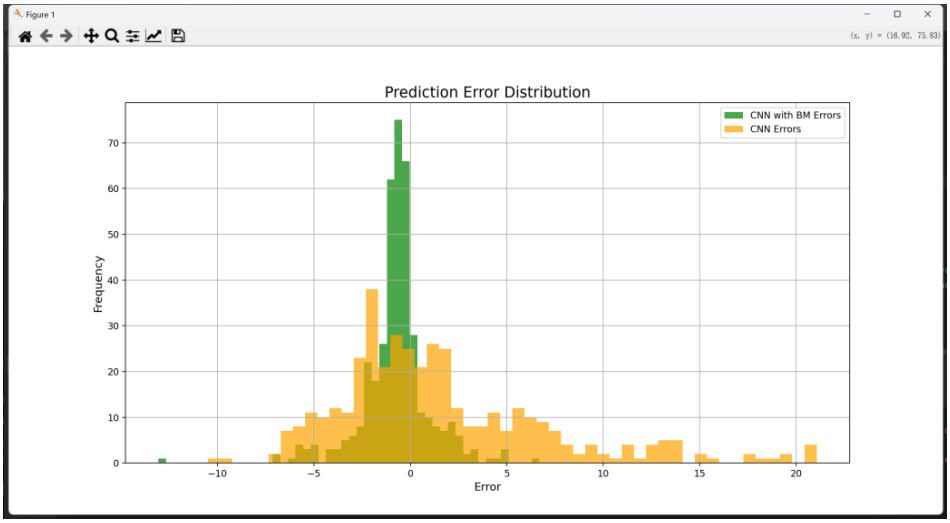

图 14: 预测误差对比图

可以很清晰的看出带有布朗运动的 CNN 模型对比于传统的 CNN 模型，前者的预测错误频率十分的低，预测正确或十分接近的频率高达 75%，而后者在正确预测上表现的相对糟糕，且预测失败的频率相对高了相当多。

3. 其次是训练损失曲线这一评估标准图15：

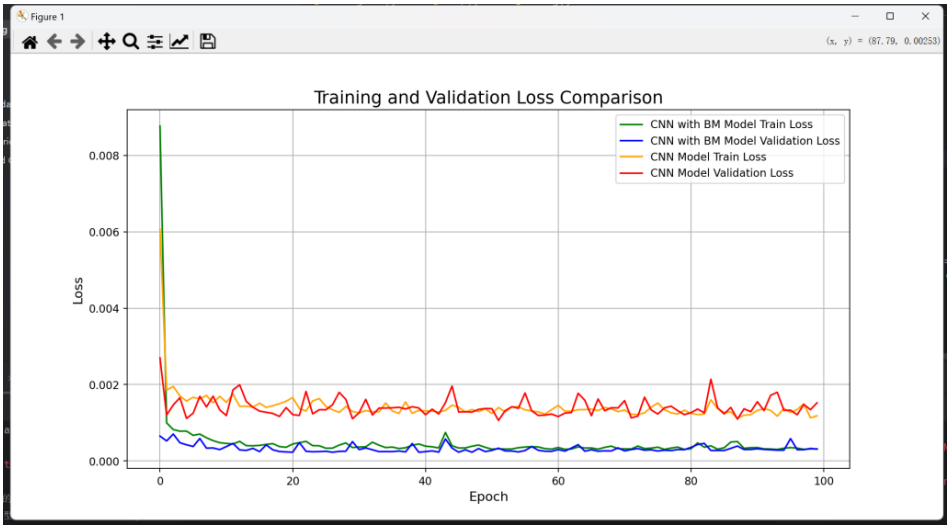

图 15: 损失值曲线对比图

可以很清晰地看出带有布朗运动的 CNN 模型对比于传统的 CNN 模型，前者在 100 个轮次中无论是训练损失还是验证损失，收敛的速度都更快，最终的损失值也更低。

4. 实际价格与预测价格对比图16。
注意：不作为评估标准来对比，仅供阅读。

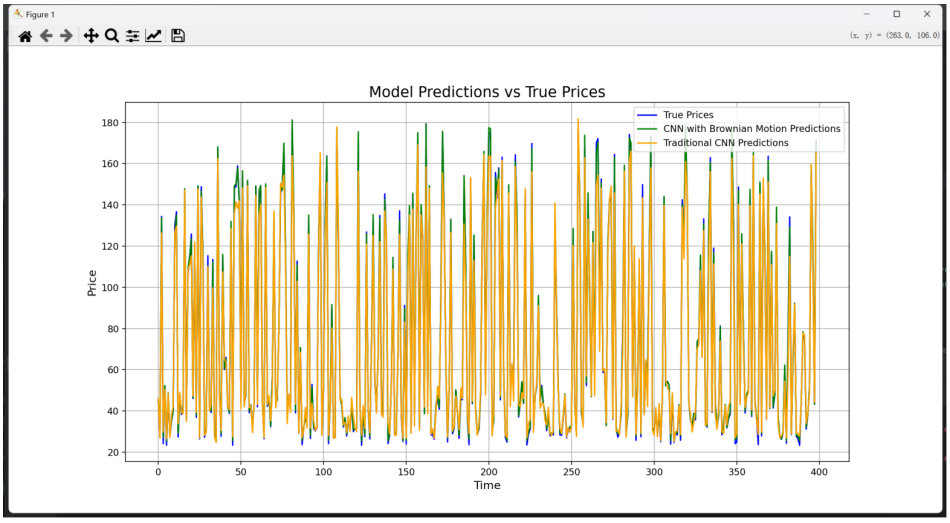

图 16: 预测与实际对比图

5.3.2 实验结果分析

根据上述实验数据，我们可以从多个方面清晰地看到带有布朗运动的 CNN 模型相对于传统 CNN 模型的优势。

1. 带布朗运动的 CNN 模型的 MSE 和 MAE 都明显更小，这意味着模型预测值与真实值之间的误差较小，从而表明布朗运动层在模拟市场波动性方面的有效性。传统 CNN 模型的 MSE 和 MAE 数值相对较大，预测误差较高，未能很好地捕捉到股市的复杂波动模式。
2. 从预测误差的分布来看，带布朗运动的 CNN 模型的误差频率明显较低。而传统 CNN 模型在正确预测方面的表现则相对较差，且预测失败的频率较高。这表明，布朗运动层的引入使得模型能够更好地拟合股市的波动规律，说明能够更准确地预测未来价格。
3. 带布朗运动的 CNN 模型在训练损失和验证损失的收敛速度上明显优于传统 CNN 模型。这表明，带布朗运动的 CNN 模型不仅收敛速度更快，而且最终的损失值更低，表明它在训练过程中对数据的适应性更强，能够更快速地达到较优的性能。
4. 总的来说，带布朗运动的 CNN 模型在资产价格预测任务中展现出了强大的性能提升，尤其是在捕捉市场波动方面，其在预测精度和模型优化方面的表现均优于传统的 CNN 模型。

6 总结和展望

本研究深入探讨了随机过程在深度学习模型优化与泛化中的关键应用，特别是通过动态调整学习率和丢弃率来提升模型性能以及优化数据的特征提取。实验结果表明，基于随机过程的动态策略能够有效改善训练过程，使模型在面对新数据时表现出更好的适应性和鲁棒性。随机过程为深度学习模型的优化提供了一种创新的思路。通过引入随机过程，可以动态调整关键超参数（如学习率和丢弃率），使得模型在不同训练阶段能够灵活适应数据特性。这种灵活性显著提高了模型的训练效率，减少了过拟合现象，从而提升了模型的泛化能力。而且在

数据波动方面，通过采用布朗运动可以更有效地捕捉其波动，提高预测精度。此外，动态策略还为传统的固定超参数设置提供了有效的替代方案，增强了深度学习技术的实用价值。未来，在这一领域的研究潜力仍然巨大。首先，可以进一步探索随机过程与其他优化方法的结合，例如与元学习、强化学习等先进技术的结合，来提升模型在多种任务上的泛化能力。其次，针对计算资源的限制，研究更高效的随机过程模型和算法，以降低动态调整超参数所需的计算开销，将是一个重要的研究方向。综上所述，随机过程在深度学习优化与泛化中的关键作用不仅为理论研究提供了新的视角，也为实际应用带来了显著的性能提升。未来的研究将有助于推动深度学习领域的发展，为解决更复杂的问题提供新的解决方案。

参考文献

[1] 彭志昊. 基于贝叶斯神经网络的鲁棒基数估计方法研究, 2024. DOI:10.27005/d.cnki.gdzku.2024.002837.

[2] 朱东升, 董旭欣, 成松鹤, and 等. 基于马尔科夫链的软件系统接口预警技术研究. 长江信息通信, 37(09):127–130+146, 2024.

[3] 汪义汉 and 张雪康. 第二类分数布朗运动驱动的随机过程中趋势函数的非参数估计（英文）. 应用数学, 37(04):885–892, 2024.

[4] 滕斌. 深度学习在倒向随机微分方程数值计算及金融资产定价中的应用, 2022. DOI:10.27272/d.cnki.gshdu.2022.000476.

[5] 满新鹏. 基于深度学习与隐藏马尔科夫模型的遥感图像分类与分割方法研究, 2023. DOI:10.27171/d.cnki.ghdcc.2023.001029.

[6] 解天舒. 基于卷积神经网络的 dropout 方法研究, 2021. DOI:10.27005/d.cnki.gdzku.2021.003500.

[7] 陈东昱, 陈华, 范丽敏, and 等. 基于深度学习的随机性检验策略研究. 通信学报, 44(06):23–33, 2023.

[8] 黄建勇. 基于深度学习的随机梯度下降优化算法研究, 2024.