分 类 号_____ 学号 M201872630

学校代码 10487 密级_____

華中科技大學

硕 士 学 位 论 文

神经网络训练过程中的
泛化误差二次下降研究

学位申请人： 张潇

学 科 专 业： 控制科学与工程

指 导 教 师： 伍冬睿 教授

答 辩 日 期： 2021 年 5 月 9 日

A Thesis Submitted in Partial Fulfillment of the Requirements
For the Degree of Master of Engineering

Delve into the Epoch-Wise Double Descent of Deep Neural Networks

Candidate : Xiao Zhang
Major : Control Science and Engineering
Supervisor : Prof. Dongrui Wu

Huazhong University of Science & Technology

Wuhan 430074, P. R. China

May, 2021

独创性声明

本人声明所呈交的学位论文是我个人在导师的指导下进行的研究工作及取得的研究成果。尽我所知，除文中已标明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本论文属于　保密□，在 ____ 年解密后适用本授权书。

　　不保密 □。

（请在以上方框内打“√”）

学位论文作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

摘 要

神经网络因其在非结构化数据上的优异表现而被广泛应用于图像、语音、自然语言处理等领域。然而神经网络理论基础目前相对薄弱，其诸多现象与问题并没有较好的理论支撑与解释，也因此成为了研究的热点与难点。神经网络理论研究的关键便是找到其独特泛化表现的理论依据，如为什么过参数化的神经网络模型依然保有较好的泛化能力。解释神经网络泛化误差二次下降现象，便是神经网络泛化性能研究中的一个重要组成部分。

传统机器学习理论中的偏差方差分析指出，随着模型的复杂度不断增加，虽然其偏差在不断下降，但是方差会不断上升，从而导致泛化误差呈现出先下降再上升的趋势。然而，神经网络的泛化误差有时会出现二次下降现象，即神经网络泛化误差随着模型复杂度的增加首先呈现经典的 U 型曲线，但后期却又会再次下降。最近研究人员发现，二次下降现象同样还出现在了训练过程中：随着训练回合数的增加，神经网络在测试集上的误差先下降，然后到达早停点后由于过拟合开始上升，最后在某个训练回合又会再次下降。神经网络这些现象都与传统的机器学习理论相违背，需要新的理论来解释其泛化能力的独特表现。

该论文研究了训练回合增加情况下的泛化误差二次下降现象。首先我们分析了分段线性神经网络片状输出地形的几何特性，论证了神经网络输出地形复杂度与其泛化能力之间的紧密联系。考虑到片状输出地形分析的局限性，我们提出了一种新的计算输出地形频谱的方法来解释神经网络泛化误差二次下降现象。过去的研究表明神经网络具有频谱偏好，即模型在训练过程中会从低频到高频地拟合目标输出地形。然而我们的研究表明，频谱偏好的单调性并不总是成立，而正是这种非单调性引起了模型泛化误差的第二次下降。为了进一步验证这种非单调性，我们对训练过程中神经网络的泛化误差进行了偏差方差分解。实验发现方差项并非如传统机器学习理论所说那样持续上升，而是会在训练后期由增加变为下降，进而使得模型泛化误差二次下降。基于该分析，我们提出了一个新的指标来度量方差项的引入程度。该指标能够仅在训练集上进行计算，但其变化趋势却能与泛化误差保持一致，也因此该指标可以在不使用校验集的情况下指示早停点。

该论文反驳了过去研究假定的学习偏好单调性，从实验上证明了正是非单调变化的学习偏好导致了神经网络泛化误差二次下降这个反常现象的出现。该研究对神经网络泛化性能的研究起到了一定的积极作用。

关键词： 深度学习　　神经网络　　泛化能力　　误差二次下降

Abstract

Due to the extraordinary performance of Deep Neural Networks (DNNs) on unstructured data, they have been widely applied in computer vision, speech recognition, natural language processing, etc. Despite of DNN's popularity in practice, its basic theory has not been completely understood yet, resulting in several unusual phenomena that cannot be explained by the traditional learning theory. The core step to understand DNNs is to figure out their unconventional generalization ability, e.g., why over-parameterized DNNs can still generalize well with extremely large model complexity. Exploring the double descent phenomenon of DNNs is a very essential part to achieve that.

The bias-variance analysis in statistic learning theory shows that, though the bias term keeps reducing with the growth of the model complexity, the variance term gradually increases, leading to a classical U-shaped curve of the generalization error. However, recent studies have discovered that DNNs demonstrate a double descent phenomenon, i.e., the generalization error sometimes decreases again after the U-shaped curve. More recently, it has been found that the double descent phenomenon also exists in the training process, i.e., it occurs when increasing training epochs. These unconventional behaviors of DNNs cannot be explained by traditional learning theory and hence require more exploration on their underlying causes.

This paper studies the epoch-wise double descent phenomenon of DNNs. We analyze the geometric properties of DNN's piecewise-linear prediction landscape and demonstrate the connection between its generalization performance and the complexity of its prediction landscape. Based on this finding, we propose an approach to calculating the spectrum of DNN's prediction landscape, which can be further utilized to explain the epoch-wise double descent phenomenon. The previous studies found that DNNs have a spectral bias to learn target functions monotonically from low to high frequencies during training. However, we show that the high-frequency components of DNNs diminish in the late stage of training, leading to the second descent of the test error. To further verify the nonmonotonicity of learning bias, we perform the bias-variance decomposition on the test error at every training epoch. We show that the variance term changes from increase to decrease in the training process, which causes the epoch-wise double descent. Based on our analysis, we design a new metric to measure the variance introduced by the optimization process. This metric is calculated on the training set alone but correlates well with the test error. As a result, early stopping point can be illustrated with no validation set.

This paper shows that the monotonicity of learning bias, which was assumed by the previous studies, does not always hold. We empirically show that it is the nonmonotonicity of learning bias that causes the epoch-wise double descent. Our study benefits the understanding of DNN's generalization ability.

Key words: Deep learning Neural network Generalization Double descent

目　录

1 绪论

1.1 研究背景与意义

机器学习（Machine Learning）旨在通过设计模型与算法来自动化地从数据中抽取表示形式、挖掘潜在关系、提炼内部信息，因此机器学习算法已经被广泛应用于科技、军事、金融以及医疗等领域[1]。近几十年来，随着互联网的不断发展及其用户群体的不断扩大，用户端每天的手机、电脑、可穿戴设备等电子产品会产生大量的数据等待分析。面对这样海量的数据，我们显然不可能使用人工的方法来分析与检测，因此机器学习提供的自动分析数据的模型与算法便受到了更多的关注与重视。

一般而言，机器学习中常见的数据分析流程大体上可以分为数据模块、模型算法、下游任务这三个模块，其具体的形式如图1-1所示。这里我们主要详细介绍数据模块以及模型算法。数据模块包含了数据采集、数据预处理以及特征提取。以生理信号数据为例，通常我们会通过可穿戴式设备上的传感器采集用户的生理信号，然后清洗数据并在其上提取可用特征以便后续分析。通常而言当用户端确定以后，数据采集的形式便大体确定了下来，因此我们必需先根据下游任务的具体要求来制定用户端。

当数据特征提取好以后，便需要使用模型算法模块来抽取数据与任务之间的关系。其中一个重要的步骤是根据数据集的分布特征、输出类型、训练样本量等信息来选择合适的模型。模型与特征之间的匹配度通常决定了这个模型泛化性能的上限，然后我们便需要使用训练集训练该模型。对于参数化模型而言，优化算法在训练过程中扮演着重要的角色。通过组合不同的优化策略以及超参数，同样的模型往往具有着差异较大的泛化性能。这时我们便需要建立一个筛选模型的评估指标，从中选出最合适的模型。模型评估指标是机器学习算法的核心，由下游任务的具体要求而定并且指导了整个模型的训练筛选过程。由于模型类型丰富、函数空间较大、优化策略多样等多种原因，我们往往需要根据先验知识来不断尝试这些组合，也因此模型的训练往往是一个迭代更新以及不断试错的过程。

在具有海量数据的情况下，模型训练的过程往往显得非常耗时，这大大增加了训练模型时试错的时间成本与硬件能源损耗。在尝试不同超参组合、模型类型时，如何有效降低试错成本便成为了一个极其重要的问题，其核心在于如何有效地去除掉理论上便不可能获得较好效果的组合形式。例如，对于线性不可分的数据集，我们便不再去考虑理论上不可行的线性模型；同样的，对于模型微调的情况我们也不会使用学习率极大的优化器来对模型进行优化。尽管如此，由于高维数据难以理解以及超参数难以精准设置，目前的现状是机器学习任务仍需要算法工程师大量的先

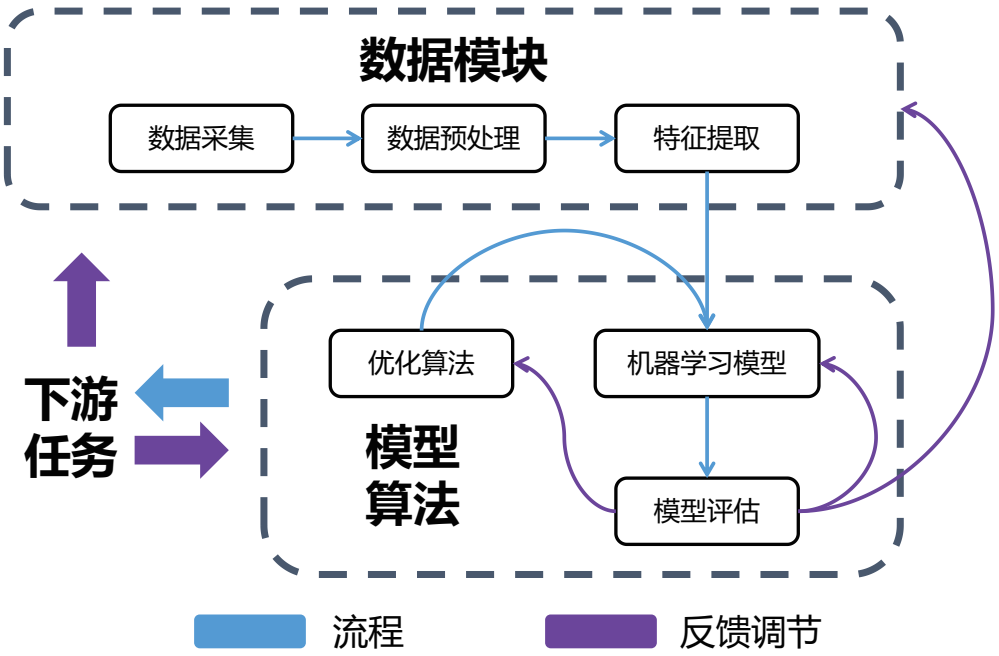


图 1-1 机器学习常见流程示意图

验知识与技巧来获得泛化性能较好的模型。

在数据与模型两者间众多需要适配的选项中，数据量与模型复杂度之间的匹配度显得尤为重要。通常而言，当训练数据样本较少时，使用模型复杂度较高的模型便极其容易发生过拟合；相反，当训练数据样本较多且分布相对复杂时，使用模型复杂度较低的模型又会出现不能拟合训练集而发生欠拟合的情形。为了在实际应用中选择模型更具有导向性而减少试错代价，一些研究深入地探讨了数据量与模型复杂度对模型泛化误差的影响，并为此提供了理论上的支撑[2,3]。更具体来说，这些研究结合了某个模型复杂度的度量以及数据量的大小，将模型的泛化误差通过不等式约束限制在某个误差上界以内，而这个误差上界便指示出了模型复杂度与数据量的“适配度”。该误差上界从理论上证明了模型可学习性的同时，也说明了如果想要模型泛化误差低于某个值，那么根据现有的数据量应该选取复杂度不高于多少的模型。这对于模型的应用与选取起到了重要的导向作用。

近年来，随着算力的提升以及数据量的不断增加，深度学习（Deep Learning）——通常特指深度神经网络①（Deep Neural Network）——在机器学习领域中开始扮演着越来越重要的角色[4–6]。对于图像、语音、文本这类非结构化数据而言，神经网络能够较好地去学习拟合其内部的规律与模式，这是传统机器学习算法很难具有的特性。更为有趣的是，神经网络往往具有端到端的特性，通常不需要复杂的特征

① 考虑到文章的简洁性，后文中的”神经网络”均指深度神经网络。

工程来人工提取数据特征，因此显得更加简洁与高效。正是因为神经网络在复杂的非结构化数据上所具有的优势，其几乎成为了机器学习系统中的一个必要组件，并且被广泛应用于各个任务中。然而人们很快发现，虽然神经网络在非结构化数据上具有优异的泛化能力，但其不能够被传统机器学习中模型复杂度与数据量之间的适配度所解释。更具体来讲，对于过参数化的神经网络而言，其模型复杂度往往远超对应的训练数据量却依然保有较好的泛化能力。如何更好地理解神经网络的泛化行为便成为了神经网络理论研究的基础以及难点。

与神经网络在现实任务中优异的效果形成鲜明对比的是其较为薄弱的理论基础，这也是神经网络诸多泛化行为不能被很好解释的根本原因。要想完全理解神经网络，需要对模型架构、数据流形以及优化过程都有深刻的理解，然而这在高维数据空间这是极其困难的。仅仅是完全弄清高维空间下的数据流形就已经是一件较为困难的事情，更不用说需要将三者之间的复杂耦合关系理清。目前的研究对于无限宽神经网络的优化过程已经有了较完善的理论解释：无限宽的神经网络模型与核学习等价，因此其优化过程也可以解析化地表示出来。无限宽神经网络所表示的核便被叫做神经正切核（Neural Tangent Kernel）[7]。然而，正常使用的有限宽神经网络与其依然存在着较大差别，有限宽神经网络所具有的一些反常泛化现象在无限宽时并不存在。综上，神经网络泛化性能的理解与研究依然任重而道远。

本文旨在对神经网络泛化误差回合二次下降这个反常的泛化现象进行探讨与研究，并希望以此为窗口更加深刻地理解神经网络的泛化能力。我们从输出地形复杂度以及偏差方差分解两个角度出发对该现象进行了解释，并以此为基础提出了不需要校验集便可以预测模型优化过程中泛化误差变化曲线的新方法。该指标减少了传统优化流程下多轮训练的时间代价与硬件损耗。

1.2 机器学习模型的泛化误差上界

机器学习的模型为什么能够较好地泛化？这个问题要求机器学习理论提供一个严格的泛化误差上界来证明其模型的可学习性。这个泛化误差上界既对机器学习模型算法的设计起到了指导作用，又可以用来解释模型的一些泛化行为，比如模型复杂度与数据量之间的匹配程度。

泛化误差（Generalization Error）刻画了机器学习模型在整个数据分布上的误差，是模型对分布中未知样本预测性能的评判标准。我们以二分类问题为例，假定我们存在某一模型函数 h，那么对于属于数据分布 D 的样本而言，模型的泛化误差可以表示为：

$$\varepsilon(h) = P_{(x,y)\sim D}(h(x) \neq y). \tag{1.1}$$

其中 x 为输入，y 为标签。显然，我们的目的是希望找到使得泛化误差最小的模型

函数 h_0，即：

$$h_0 = \arg\min_h \varepsilon(h). \tag{1.2}$$

然而遗憾的是，在真实情况下我们很难搜索所有的函数。因此我们通常只能够提前限定一个函数空间 H，然后从中搜索泛化误差最小的模型函数 h^*，即：

$$h^* = \arg\min_{h\in H} \varepsilon(h). \tag{1.3}$$

除此之外，我们通常也只能获得数据分布中的 N 个采样点 $S = \{(x_i, y_i) \mid (x_i, y_i) \sim D\}_{i=1}^N$ 而不能够知道数据完整的分布 D。在这样的情形下，我们一般使用经验风险最小化（Empirical Risk Minimization，ERM）——即令训练误差最小——来尽可能逼近函数空间 H 中的最优模型函数 h^*。具体而言，令训练误差为：

$$\hat{\varepsilon}(h) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(h(x_i) \neq y_i), \tag{1.4}$$

其中 $\mathbf{1}(\cdot)$ 为判断函数，当输入条件为真时为 1，输入为假时为 0。由此，经验风险最小化找到的函数 $\hat{h}$ 即为：

$$\hat{h} = \arg\min_{h\in H} \hat{\varepsilon}(h). \tag{1.5}$$

显然，当给定一个函数空间 H 时我们希望经验风险最小化原则找到的 $\hat{h}$ 与 h^* 之间的差异性较小，即 $\varepsilon(\hat{h}) - \varepsilon(h^*)$ 足够小。然而，通常而言通过经验风险最小化求得的 $\hat{h}$ 并不唯一，因此我们需要采取概率近似正确原则（Probably Approximately Correct，PAC）来略微放松这个要求。

所谓概率近似正确原则，其根本思想在于希望 $\varepsilon(h^*)$ 与 $\varepsilon(\hat{h})$ 差异较小的概率足够大。更加精确的来说，对于某一误差范围 $\epsilon > 0$ 以及容错概率 $\delta > 0$ 而言，我们希望有：

$$P(\varepsilon(\hat{h}) - \varepsilon(h^*) \leq \epsilon) \geq 1 - \delta. \tag{1.6}$$

为了方便后续说明，我们这里首先不加证明地给出 Hoeffding 不等式：

引理 1.1 (Hoeffding 不等式)： 令 $Z_1, Z_2, ..., Z_m$ 为从伯努利分布 $B(\phi)$ 中采样的 m 个独立同分布的变量，即对所有的 $i = 1, 2, ..., m$ 有 $P(Z_i = 1) = \phi$。令 $\hat{\phi} = (1/m)\sum_{i=1}^m Z_i$，则对于任意固定的 $\gamma > 0$，以下不等式成立：

$$P(|\phi - \hat{\phi}| > \gamma) \leq 2\exp(-2\gamma^2 m). \tag{1.7}$$

从 Hoeffding 不等式出发，我们可以知道对于任意 $h \in H$，我们有：

$$P(|\varepsilon(h) - \hat{\varepsilon}(h)| > \gamma) \leq 2\exp(-2\gamma^2 m). \tag{1.8}$$

1.2.1 有限函数空间

若函数空间 H 包含有限的 k 个函数，那么所有函数的泛化误差与训练误差均接近的概率为：

$$\begin{aligned} P(\forall h \in H, |\varepsilon(h) - \hat{\varepsilon}(h)| \leq \gamma) &= 1 - P(\exists h \in H, |\varepsilon(h) - \hat{\varepsilon}(h)| > \gamma) \\ &\geq 1 - \sum_{i=1}^{k} P(|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| > \gamma) \\ &\geq 1 - \sum_{i=1}^{k} 2\exp(-2\gamma^2 m) \\ &= 1 - 2k\exp(-2\gamma^2 m). \end{aligned} \tag{1.9}$$

显然，在满足 $\forall h \in H, |\varepsilon(h) - \hat{\varepsilon}(h)| \leq \gamma$ 的情形下我们有：

$$\varepsilon(\hat{h}) \leq \hat{\varepsilon}(\hat{h}) + \gamma \leq \hat{\varepsilon}(h^*) + \gamma \leq \varepsilon(h^*) + 2\gamma. \tag{1.10}$$

因此令 $\epsilon = 2\gamma$，则有：

$$P(\varepsilon(\hat{h}) - \varepsilon(h^*) \leq \epsilon) \geq 1 - 2k\exp\left(-\frac{\epsilon^2 m}{2}\right). \tag{1.11}$$

由上我们可知，容错率 δ 与误差范围 ϵ 之间的关系为 $\delta = 2k\exp(-\epsilon^2 m/2)$，即：

$$\epsilon = \sqrt{\frac{2}{m}\log\frac{2k}{\delta}}. \tag{1.12}$$

由此我们便可以知道，当 $\frac{1}{m}\log\frac{2k}{\delta}$ 足够小时，模型的可学习性便可以被保障，而这要求较多的训练样本以及较小的函数空间。

1.2.2 无限函数空间

虽然上一节介绍了有限函数空间下的模型泛化误差上界，但是在实际情况下函数空间 H 通常是无限的，因此针对该情况我们需要建立新的泛化误差上界。显然，我们需要类似 k 的某种针对无限函数空间的复杂度度量指标，而 VC 维（Vapnik-Chervonenkis Dimension）[2]便应运而生。

要想理解 VC 维，首先需要定义增长函数：

$$\Pi_H(m) = \max_{x_1,...,x_m} \left|\{(h(x_1), ..., h(x_m) \mid h \in H\}\right|, \tag{1.13}$$

其中 x_i 为从数据分布上采样的输入。我们可以看到，该函数定义了函数空间 H 在 m 个样本点上输出标签的最多可能性。而 VC 维，便是度量了该函数空间能拟合任意标签组合的能力。对于二分类问题来说，其函数空间的 VC 维可以定义为：

$$VC(H) = \max_m\{\Pi_H(m) = 2^m\}, \tag{1.14}$$

即函数空间能够对任意标签样本进行拟合的最大样本数。

通过 VC 维的概念，我们能够对无限函数空间的泛化误差进行限制，即有如下不等式成立：

$$P(\forall h \in H, |\hat{\varepsilon}(h) - \varepsilon(h)| \leq \gamma) \geq 1 - 4(2m)^{VC(H)} \exp\left(-\frac{\gamma^2 m}{8}\right). \tag{1.15}$$

这里我们不再详述其具体的证明，详细证明过程见下①。

与有限函数空间中的推导一致，我们令 $\epsilon = 2\gamma$，则有：

$$P(\varepsilon(\hat{h}) - \varepsilon(h^*) \leq \epsilon) \geq 1 - 4(2m)^{VC(H)} \exp\left(-\frac{\epsilon^2 m}{32}\right). \tag{1.16}$$

因此我们同样可以对无限函数空间进行泛化误差上界限定，由此函数空间 H 的可学习性便得到了证明。

1.3 神经网络泛化悖论

神经网络通常由多个隐层构成，而每个隐层又包含多个神经元。换句话说，神经网络所表示的函数可以看作是大量并行的简单操作符层级嵌套以后的运算。通常而言，神经网络的模型复杂度远超数据量，但其依然可以保有较好的泛化性能，这种反常的泛化能力与上一节中提出的机器学习泛化误差上界相违背。

之前的研究表明，过参数化神经网络的模型复杂度大到能够拟合具有随机标签的训练集，但是却依然能够在正常数据集上保有较好的泛化能力[8]。这个现象意味着从神经网络的 VC 维或者数据相关的 Rademacher 复杂度[3]出发，其对于训练集具有近乎无限的空间复杂度，然而其泛化误差却远低于上一节中给出的泛化误差上界。显然，前文中可学习性的证明对神经网络而言失效了。除此之外，人们发现在实际应用过参数化的神经网络时其泛化性能会随着模型复杂度的增加进一步提升，即增加神经网络的宽度与深度便能够使其具有更好的泛化性能[9–12]。还有研究表明，当神经网络宽度增加时，其误差平面变得更加光滑从而提高了可优化性[13]。这一切都指出，神经网络的泛化性能不再能被传统机器学习中的理论所解释。

针对神经网络反常的泛化表现，部分研究将其归因于优化过程中的学习偏好（Learning Bias）。所谓学习偏好，即指虽然神经网络拥有近乎无限大的函数空间，但其在优化过程中并不会等价地搜索整个函数空间，而是会对其部分子空间具有更强的偏好性。学习偏好意味着神经网络虽然有着极大的模型复杂度，但在实际应用中却并不会体现出来。Arpit 等在 2017 年指出，神经网络会优先学习数据集中的模式，然后才会记忆噪声数据[14]；Kalimeris 等通过互信息的方式证明了，优化过程中神经网络会优先学习简单的函数，然后才是复杂的函数[15]；还有系列研究表明，神

① https://nowak.ece.wisc.edu/SLT09/lecture19.pdf

经网络会优先拟合输出地形的低频分量，然后才是高频分量[16–18]。正是由于这些学习偏好的存在，神经网络的高复杂度在实际情况下得以隐藏，从而不会发生严重的过拟合。

1.4 神经网络的误差二次下降现象

学习偏好的存在似乎能够很好地解释神经网络反常的泛化行为，然而我们结合目前已有的关于神经网络误差二次下降现象的研究[19]，对学习偏好的单调性提出了质疑。以前的研究都假定了学习偏好是具有单调性的，即从学习到记忆、从简答到复杂、从低频到高频。近来研究发现，神经网络存在着误差二次下降的现象，即随着模型复杂度或者训练回合数的增加，模型泛化误差呈现出从下降到上升再到下降的复杂变化趋势。为方便起见，后文中将关于模型复杂度以及关于训练回合数的误差二次下降现象分别简称为**“宽度双降现象”**以及**“回合双降现象”**。图1-2详细展示了宽度双降现象以及回合双降现象。图中的结果为在包含 20% 标签噪声的 CIFAR10 上使用不同宽度的 ResNet18 训练后所对应的测试错误率变化趋势。我们可以看到，随着模型宽度的增加，神经网络的泛化误差呈现出二次下降的现象；同样，当我们增大训练回合数时也观察到了二次下降现象。由学习偏好的单调性我们可以知道，神经网络泛化误差为从欠拟合到过拟合的 U 型曲线，这便与泛化误差二次下降现象相违背。由此可知，理解神经网络的泛化误差二次下降现象对于研究神经网络的泛化能力有着重要意义。

宽度双降现象：宽度双降现象指出，模型测试错误率随着模型复杂度的增加会

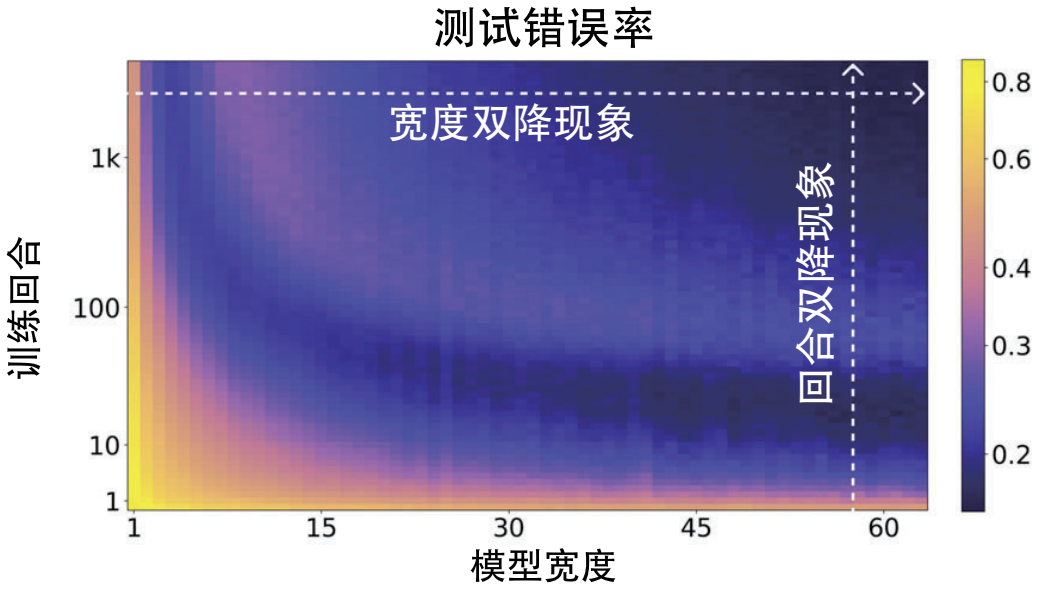


图 1-2 宽度双降现象以及回合双降现象示意图[19]

首先呈现出经典的 U 型曲线，然后紧接着却会再次下降。该现象较早被发现，且存在于机器学习的各个任务中[19–23]。许多研究在一些简单易处理的问题设置下对该现象提供了理论上的说明[24–29]。其中，Neal 等[30]与 Yang 等[27]通过对均方误差以及交叉熵误差进行偏差方差分解对宽度双降现象进行了解释，他们在真实任务中通过实验说明了方差项的钟型变换是宽度双降现象的主要原因；同时，他们在人造的极其简单的数据集与模型上，从理论上揭示了宽度双降现象发生的原因。Maddox 等于 2020 年度量了模型参数空间的有效维度，从而较好地表示了模型的实际复杂度并借此解释了宽度双降现象[23]。虽然这些研究都对宽度双降现象进行了解释与说明，但是由于正常神经网络模型以及真实数据集在优化过程中极为复杂的耦合，真实场景下的宽度双降现象依然欠缺理论上的证明。

回合双降现象： 除了宽度双降现象以外，研究人员于 2020 年观察到了一种新的误差双降现象，即模型泛化误差回合双降[19]。通常在具有一定的标签噪声下，模型的泛化错误率随着训练过程的不断进行会先呈现出下降趋势，然后到达早停点后由于过拟合开始上升，最后又会转变为下降趋势。与宽度双降现象相比，回合双降现象相对而言被研究的较少。Heckel 等在 2020 年指出，回合双降现象出现的根本原因在于神经网络不同部分在不同训练回合被学习，从而使得最终的泛化误差曲线类似于多个 U 型曲线的叠加组合[31]。他们通过调节学习率的大小来同步神经网络不同部分的学习程度，最后成功去除了回合双降现象的尖峰。但是，他们的研究依然没有能够解释清楚神经网络学习偏好与回合双降现象之间的矛盾。

1.5 研究内容与组织结构

本文旨在通过研究神经网络回合双降现象来对其泛化性能进行探究。由于神经网络在优化过程中通常只会搜索部分的函数空间，因此其模型复杂度在实际情况中远低于所能达到的理论上限。由此我们可以知道，如何在训练过程中度量神经网络的实际复杂度便成为了分析神经网络回合双降现象的关键。考虑到神经网络输出地形能够反映模型本身的复杂程度，我们在本文中将其与模型的泛化能力联系了起来，并以此为基础对回合双降现象进行分析。总体上，本文每章涉及的内容及其间的联系如图1-3所示。

在第二章节中，我们针对分段线性神经网络的片状输出地形给出了多种基于凸优化的分析工具，从而能够更加精确地刻画神经网络样本空间不同区域的输出地形的几何特性。通过对不同优化设置下神经网络的片状地形进行分析比较，我们发现使用 Batch Normalization（BN）或者 Dropout 等优化技巧会使得同样的模型架构具有更加细密的输出地形，从而显著增强了其拟合能力。这也解释了为什么 BN 与 Dropout 常常带来更好的泛化性能。我们的实验说明输出地形与模型泛化能力之间具有紧密的联系，从而证明了通过输出地形来分析回合双降现象的可行性。

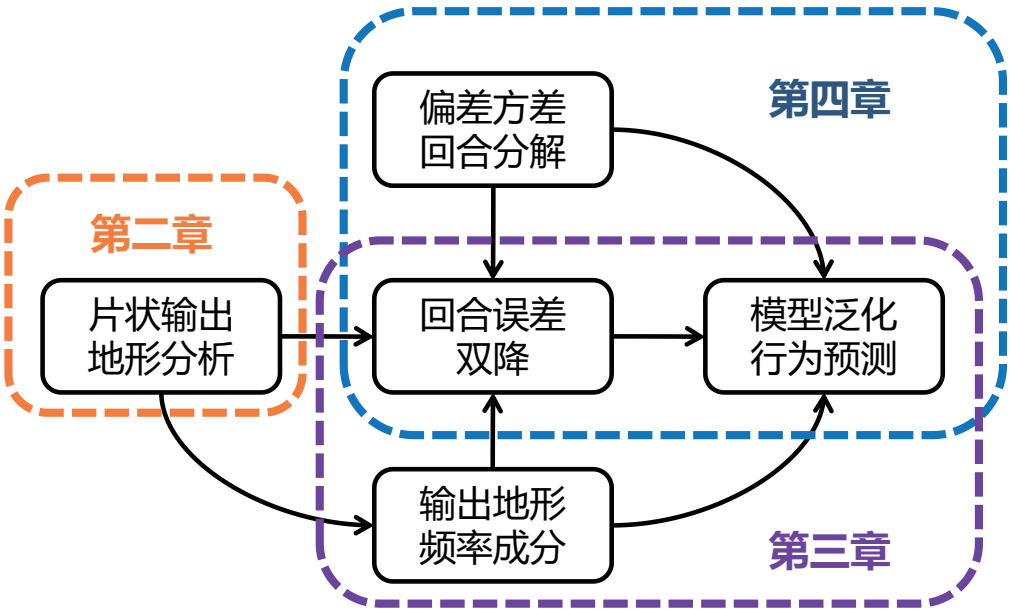


图 1-3 文章内容及其架构示意图

然而，简单地使用片状地形几何特性来分析回合双降现象存在着诸多局限与不足，因此第三章中我们在以往研究的基础上提出了一种新的输出地形频谱的计算方法，并用以揭示输出地形的复杂度。我们发现学习偏好在训练前期是成立的，即神经网络从低频分量到高频分量地拟合输出地形。但是随着进一步训练，神经网络输出地形的高频分量开始消失，从而引起了泛化误差的第二次下降。通过分解流形上的输出地形与非流形的输出地形，我们发现流形区域的高频分量为了拟合数据集中的噪声始终不断增长；然而非流形输出地形却不断变得平坦起来，从而使得高频分量下降，也因此导致了模型泛化性能的提升。除此之外，我们发现使用训练集上计算的频谱峰值可以有效确定模型泛化误差二次下降的起点，这也为不使用校验集便可以确定模型泛化行为提供了可能。

考虑到模型的输出地形会随着采样噪声的引入发生较大变化，在第四章节中我们对回合双降现象进行了偏差方差分析。我们采用了一个统一的偏差方差分析框架来对回合双降现象中的零一误差进行了分解。我们的实验结果证明，方差的变化主导了回合双降现象的发生。换句话说，噪声带来的输出地形差异使得模型方差增大，但在训练后期由于非流形区域输出地形的平整化方差又逐渐降低，从而使得泛化误差在训练过程中出现了二次下降的现象。基于该现象，我们提出了一个新的叫做优化方差的指标来度量采样噪声引入方差的大小。该指标能够仅在训练集上计算，但是却可以预测泛化误差的变化趋势。

最后，我们在第五章节对神经网络泛化性能的研究进行了总结，并且指出了进一步探究其反常泛化表现的可能研究方向。

总结来看，本文的主要贡献如下：

- 通过对片状地形几何特性的分析，我们证明了神经网络输出地形与其泛化能力的紧密联系，从而说明了通过输出地形复杂度来表示神经网络实际泛化能力的可行性。
- 我们通过实验证明了学习偏好的单调性并不总是成立，而正是训练过程中非单调变化的模型复杂度导致了泛化误差回合双降现象的发生。
- 我们提出了一个新的指标，该指标不需要使用校验集便可以对模型训练过程中的泛化误差变化趋势进行较为精准的预测。

2 神经网络的片状地形与泛化表现

本章从分段线性神经网络的片状地形切分粒度及其包含的决策边界出发，比较了各种优化模块对输出地形复杂度的影响，论证了输出地形与模型泛化行为之间的紧密联系。与此同时，我们也说明了使用片状地形来分析泛化行为中回合双降现象时存在的局限以及不足。该章为后续章节通过分解输出地形频谱成分来探究模型回合双降现象提供了线索与铺垫。

2.1 输出地形复杂度与泛化性能

模型的复杂度是衡量模型拟合能力的一个重要指标。通常而言，模型复杂度的上升会带来拟合能力的提高，从而模型能够更好地拟合训练集中的样本点。然而模型复杂度的提升往往是一把双刃剑：较高的模型复杂度通常会导致模型过拟合训练集中由于采样不充分或标注错误等问题引入的噪声，使得模型泛化能力反而逐渐退化[2]。因此，如何针对数据的特性适配恰当的模型便成了机器学习中的关键问题。

需要注意的是，模型复杂度一般分为两种，一种是模型理论上能够达到的最大模型复杂度，另一种是在实际应用中模型展现出来的模型复杂度。前者代表了模型本身的拟合潜力，一般根据模型类别和参数量发生改变。然而，模型在训练过程中往往并不会搜索整个函数空间，这也意味着理论上的模型复杂度上限不能够较好反映模型函数空间被搜索到的实际大小。更加准确地讲，模型对应的函数空间在实际训练过程中会体现出一定的偏好概率分布，即函数空间内的函数并不会被等概率的搜索到。换句话说，即使是具有相同函数空间的同一模型，当采用不同的优化器、学习率或者正则化手段时，其对应函数空间上的偏好概率分布也会有所不同，从而导致泛化性能存在差异。从这个角度来看，结合了数据、优化与模型的第二种复杂度，往往能够更好的体现出模型在实际任务中所具有的泛化能力。

模型实际展现出来的模型复杂度与其理论上限之间的差别在神经网络上体现得尤为明显。作为一种过参数化的模型，神经网络的函数空间十分庞大，甚至能够拟合随机标签的训练样本，然而在实际应用中神经网络极高的模型复杂度上限却并没有带来糟糕的泛化性能[8]。这个看似相悖的结论的破解点在于，纵使神经网络具有极高的模型复杂度上限，但是其在训练过程中展现出来的偏好函数概率分布使得仅仅较小一部分函数空间被搜索到。由此可见，度量模型在训练过程中的偏好函数分布是分析模型泛化能力的关键。

模型的输出地形复杂度便是一种模型实际复杂度的体现。从数学上讲，假使指标 C 定义了模型的输出地形复杂度，且经过训练后的模型输出地形复杂度为 C_0，那么模型泛化误差与经验误差之间的差异便可以被某个关于 $C \leq C_0$ 所定义的函数空

间大小的量限制住。图2-1给出了一个简单示意图来说明输出地形复杂度与模型泛化能力之间的关系，其中左图为模型复杂度较低的线性模型的输出地形图，右图为模型复杂度较高的非线性模型的输出地形图。显然我们可以发现，具有较高输出地形复杂度的模型能够更好的拟合训练集，但同时也存在着过拟合训练集中噪声从而降低泛化性能的风险。一般来说，我们期望的情形是模型具有较强的拟合潜力，即理论上较大的模型复杂度上限，但是在训练过程中却只搜索较小且合理的函数空间。为了满足这两个要求，实际应用中经常采取的策略是：1）使用如神经网络等具有较高复杂度的模型；2）选择合适的优化方案以及正则化方法来调整模型的函数搜索偏好。综上我们可以发现，模型输出地形的复杂度与其在训练过程中的泛化性能变化息息相关，也因此具有极为重要的研究意义。

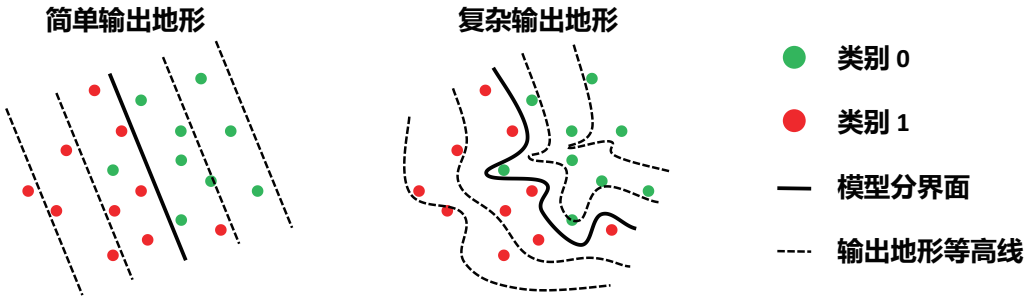


图 2-1 简单与复杂的输出地形示意图

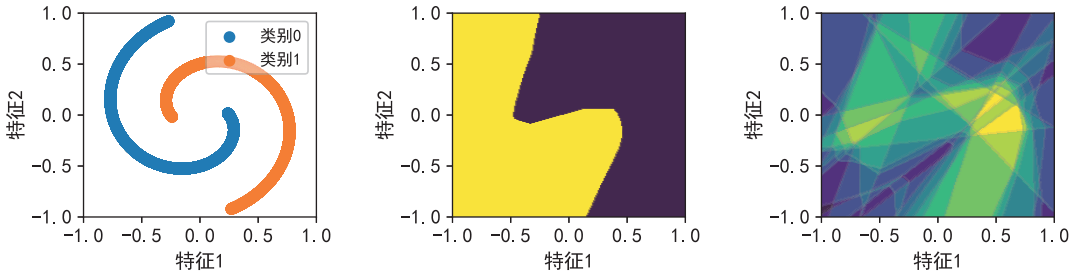
2.2 神经网络的片状地形

上一节内容介绍了模型输出地形复杂度与泛化能力之间的联系。然而，在高维空间下对模型的输出地形进行遍历分析十分困难——有时甚至很难对某一样本点的邻域进行完善的解析。因此，我们往往需要利用神经网络的某些特性来使分析更加便利，如利用神经网络使用 ReLU 等分段线性函数来激活时所具有的片状地形。

2.2.1 线性区域介绍

使用分段线性激活函数的前馈神经网络将样本空间切分为许多小的线性区域，而在每个线性区域中神经网络的表现就是一个完全的线性模型[32,33]。更加具体来讲，神经网络的激活状态与样本空间的线性区域一一对应，换句话说位于同一个线性区域的样本点会激活神经网络相同的神经元。因此，对于这些激活相同神经元的样本点来说，神经网络的隐层退化为一系列的线性变换，故神经网络在一个线性区域中退化为一个线性模型。图2-2展示了一个简单分类任务上神经网络模型划分线性区域的直观示意图。这里使用的神经网络包含三个隐层，其中每层具有10个激活函数

为 ReLU 的神经元。该模型使用 ADAM 优化器[34]进行训练，使其能够分开两条螺旋线。从图中我们可以看到，分段线性的神经网络根据激活状态的不同将样本空间切分为了许多个小的多边形，使其能够拟合复杂的分类边界。我们可以证明，对于使用分段线性激活函数的神经网络而言，其输出地形的构造为分段线性的片状结构。显然，当使用多条线段去拟合一条复杂曲线时，越多的线段会带来更小的拟合误差；同样的道理，线性区域的片数越多，也就意味着神经网络能够逼近更加复杂的输出地形[35]。从这个角度来讲，神经网络线性区域的切分粒度与其模型复杂度具有极强的相关性。



(a) 数据集分布　　(b) 类别区域　　(c) 线性区域

图 2-2 神经网络划分线性区域的简单示意图。a）二维数据集的分布；b）模型输出的分类边界与区域（不同颜色表示不同的类别）；c）模型划分的线性区域（灰色直线表示划分的超平面，颜色表示不同的激活程度）

研究表明，理论上神经网络划分的线性区域数量所能达到的上限会随着层数指数级上升，也因此有了“神经网络复杂度随着层数指数级上升”的说法[33,35,36]。同时，研究也表明了虽然深度的增加会带来梯度消失、梯度爆炸以及梯度弥散等优化问题[37–39]，但是神经网络深度提升模型复杂度的效率远高于宽度。我们需要强调的是，这里提到的模型复杂度都是理论上限值，而在训练过程中的实际模型复杂度取决于优化过程以及数据集本身。我们将在后面通过实验说明，即使同样的神经网络模型，当使用了 Batch Normalization（BN）[40]或 Dropout[41] 等优化技巧以后，其所对应的实际模型复杂度也会大大改变。

2.2.2 搜索某点所在的线性区域

一个线性区域本质上是由多个神经元激活状态决定的半空间构成的交集，因此搜索某个样本点所在的线性区域只需要找到该样本点对应的神经网络激活状态决定的半空间即可。

让我们考虑一个具有 L 层的 ReLU 全连接神经网络。从数学上来说，设 $\boldsymbol{x} \in \mathbb{R}^d$ 为 d 维的向量输入，$\boldsymbol{h}^l(\boldsymbol{x})$ 为具有 n_l 个神经元的第 l 层通过激活函数之前的输出 $(l = 1, 2, ..., L)$，$\boldsymbol{z}(\boldsymbol{x})$ 为输出层 logits 输出，M 为输出类别的数量。现在考虑某个输

入样本 $\boldsymbol{x}^*$，根据线性区域与激活状态的一一对应关系，这个样本点对应的线性区域是一个包含了所有跟它具有相同激活状态的样本的集合。

令 S_l 表示样本空间中使得神经网络前 l 层的激活状态与被搜索样本 $\boldsymbol{x}^*$ 相同的所有样本点的集合，显然 $S_{l+1} \subseteq S_l$。现在让我们考察第一层，因为 $\boldsymbol{h}^1 : \mathbb{R}^d \to \mathbb{R}^{n_1}$ 是 $\boldsymbol{x}$ 的线性变换，故根据 $\boldsymbol{x}^*$ 的激活状态，S_1 可以表示为:

$$S_1 = \left\{ \boldsymbol{x} \mid \boldsymbol{w}_i^T \boldsymbol{x} + b_i \geq 0, \quad \forall i \in \{1, ..., n_1\} \right\}, \tag{2.1}$$

其中

$$\boldsymbol{w}_i = \text{sgn}(\boldsymbol{h}_i^1(\boldsymbol{x}^*)) \nabla_{\boldsymbol{x}} \boldsymbol{h}_i^1(\boldsymbol{x}^*), \tag{2.2}$$

$$b_i = \text{sgn}(\boldsymbol{h}_i^1(\boldsymbol{x}^*)) \left[\boldsymbol{h}_i^1(\boldsymbol{x}^*) - (\nabla_{\boldsymbol{x}} \boldsymbol{h}_i^1(\boldsymbol{x}^*))^T \boldsymbol{x}^* \right]. \tag{2.3}$$

接着，我们从 S_1 中划分样本空间来构成 S_2，其中 S_2 包含了前两层神经元激活状态与 $\boldsymbol{x}^*$ 相同的所有样本点。我们已经知道 S_1 中的样本点在神经网络的第一层具有完全一样的激活状态，因此对于位于 S_1 中所有的样本点 $\boldsymbol{x}$ 来说，$\boldsymbol{h}^2 : S_1 \to \mathbb{R}^{n_2}$ 也是一个线性变换。因此，S_2 的构建可以按照公式2.1 中同样的方式在 S_1 中添加第二层神经网络激活状态所对应的线性不等式限制。以此类推，考虑到 $\boldsymbol{h}^l : S_{l-1} \to \mathbb{R}^{n_l}$（其中 $S_0 = \mathbb{R}^d$）在 l 所有的取值下都是线性映射函数，上述过程可以不断重复直到神经网络最后一层。图2-3展示了对 $\boldsymbol{x}^*$ 对应的线性区域进行逐层搜索的过程，其中每一层的每个节点都定义了一个超平面来切割前面所有层确定的多面体。

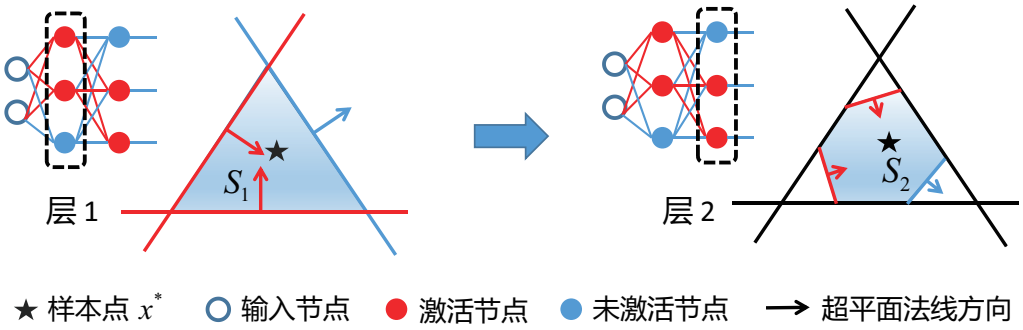


图 2-3 逐层切割搜索 $\boldsymbol{x}^*$ 所在的线性区域的示意图

令 $\mathcal{C}^* = \{(\boldsymbol{w}_i, b_i)\}_{i=1}^{\sum_{l=1}^{L} n_l}$ 表示线性区域所有线性不等式限制所对应的参数，那么 $\boldsymbol{x}^*$ 所在的线性区域 S_L 便可以表示为 $\sum_{l=1}^{L} n_l$ 个半空间的交集：

$$S_L = \left\{ \boldsymbol{x} \mid \boldsymbol{w}_i^T \boldsymbol{x} + b_i \geq 0, \quad \forall (\boldsymbol{w}_i, b_i) \in \mathcal{C}^* \right\}, \tag{2.4}$$

其中 $\boldsymbol{z} : S_L \to \mathbb{R}^M$ 为定义域内 $\boldsymbol{x}$ 的线性映射，换句话说神经网络在该区域为一个线性模型。

虽然我们主要讨论的是全连接神经网络，但是这种刻画线性区域的方式几乎可以不做修改地扩展到其他情况。比如，BN 只是一种权重参数的放缩与平移，卷积层可以被看做稀疏的全连接层，而最大池化层只是简单地增加了更多的线性不等式来指示局部的最大值。

我们后面的实验将会展示简单的卷积神经网络所对应的线性区域特性。卷积神经网络可以被视作一种高度稀疏化的全连接神经网络，这意味着我们可以用分析全连接神经网络线性区域的方式来分析卷积神经网络。但是，卷积层的特点会在线性区域上引入一些其他的性质。举例来说，由于卷积层的权重具有稀疏性以及共享性，故其每一个神经元只能切分样本空间的一个子空间，而不是像全连接神经网络的神经元一样切分整个空间。所以，其线性区域的切分相对全连接神经网络而言更加独立，因为大部分子空间并不具有交集。这种独立性导致的直接后果就是，卷积神经网络的线性区域比全连接神经网络更加鲁棒，这现象在后续的实验中也可以被看到。

2.3 线性区域的性质与特点

上一节内容我们给出了神经网络线性区域的解析表达式。这一节内容中，我们将详细介绍从线性区域的解析表达式中可以获取的地形几何特性及其计算方式。

2.3.1 线性区域与正多面体

一个线性区域可以表示为公式2.4定义的一组线性不等式的解集。这种表达方式本质上为凸多面体的 **H 表示**（H-representation），即多个半空间的交集。由此可见，线性区域的几何形态为高维空间下的凸多面体，因此我们可以使用分析凸多面体的方式来分析线性区域，从而能够对神经网络输出地形进行定量的分析。

除了 H 表示，凸多面体还可以表示为一系列点集的凸包，即凸多面体的 **V 表示**（V-representation）。针对相同的问题，使用凸多面体不同的表示方式进行处理会出现完全不同的处理复杂度，甚至这两种表示方式之间的相互转换本身就是困难的[42]。随着凸多面体所处的维度不断上升，我们很难通过 H 表示去计算顶点数量或通过 V 表示去计算面的数量[43]，甚至很多时候很难确定某个 V 表示与某个 H 表示是否等价为同一个凸多面体[44]。由于神经网络的线性区域为 H 表示，因此我们主要考察的为 H 表示下容易计算的几何特征。多面体计算的知识可参考该网站①。

在高维空间下分析神经网络哪怕是某一样本点邻域的输出地形都十分困难，但是神经网络的片状地形结构为其提供了可能。我们已经知道神经网络在样本点所处线性区域内的表现等价于线性模型，而线性区域本身为凸多面体且其解析表达可以通过上节提到的方法获得，由此神经网络在该样本点所处线性区域内的地形特点便

① https://inf.ethz.ch/personal/fukudak/polyfaq/polyfaq.html

完全可知的。除此之外，线性区域的凸多面体形式正好构成了一个凸可行域，这个特性为我们使用凸优化来分析线性区域带来了便利。

2.3.2 神经网络的切分粒度

样本空间中线性区域的数量与神经网络拟合能力高度相关。正如前文所说，拟合一个复杂的地形往往需要大量的线性区域，所以神经网络对样本空间的切分粒度便显得尤为关键。

衡量神经网络的切分粒度，其根本在于判断线性区域的“大小”。然而，线性区域作为高维度下的不规则凸多面体，很难找到一个完善的指标来度量其在各个维度上的大小。我们这里选择了线性区域的内切球半径来视作切分粒度的大小，该选择主要出于以下两点考虑：

- 内切球半径可以方便的在线性区域 H 表示下使用凸优化的方式求解出来。
- 即使遇到如线性区域极度狭长这样极端的情况，内切球半径也能够较好的反映至少某个维度的切分粒度。

所谓内切球，即其球心到凸多面体各面的最小距离最大时对应的球体。根据这个特性，我们可以知道线性区域所代表的凸多面体的内切球可以通过求解以下凸优化问题得到:

$$\begin{aligned} \max_{\overline{\boldsymbol{x}},r} \quad & r \\ \text{s. t.} \quad & \boldsymbol{w}_i^T\overline{\boldsymbol{x}} - r\|\boldsymbol{w}_i\| + b_i \geq 0, \quad \forall(\boldsymbol{w}_i, b_i) \in \mathcal{C}^*, \\ & \text{MIN}_x + r \leq \overline{\boldsymbol{x}} \leq \text{MAX}_x - r, \end{aligned} \tag{2.5}$$

其中 $\overline{\boldsymbol{x}}$ 为内切球的中心点，r 为内切球的半径，$\mathcal{C}^*$ 表示在公式2.4中定义的 S_L 所对应的参数

由此，当给定某个样本点所处的线性区域后，我们便可以通过求取上述优化问题的解来获得其内切球的半径，进而可以将其看做神经网络在该区域的切分粒度。

2.3.3 线性区域内的分类区域

高维空间下神经网络的决策面往往极其复杂，也因此会出现一些“瑕点”，即对抗样本。对抗样本是一类在正常样本上添加微弱的恶意噪声后的样本，这种样本与原始样本几乎没有区别却会被模型错误分类[45,46]。从样本空间上来看，对抗样本本质上是将正常样本推向并跨过最近的决策边界从而使其分类错误[46–49]。因此，查看样本点邻域内的决策边界对神经网络的鲁棒性研究极其重要。然而正如前面所说，高维空间下完整地探究样本点的邻域并不容易。可以证明，神经网络模型在同一线

性区域内等效于一个线性模型，因此其决策面在线性区域内可以完全确定，从而使得在线性空间中探究决策边界变得简单且高效。需要注意的是，我们并不能保证样本所在线性区域内的决策边界离样本点最近，因此实际上这里的探究是对模型鲁棒性放宽了限制。

首先我们需要回答的一个重要问题是：**样本点所在的线性区域内存在决策边界吗?** 为了回答这个问题，我们搜索了 S_L 中以最大概率被划分为类别 $t \in \{1, 2, ..., M\}$ 的样本点。具体的，我们可以通过求解下面的凸优化问题来获得：

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{z}_t(\boldsymbol{x}) - \log\left(\sum_{j=1}^{M} \exp(\boldsymbol{z}_j(\boldsymbol{x}))\right) \\ \text{s. t.} \quad & \boldsymbol{w}_i^T \boldsymbol{x} + b_i \geq 0, \quad \forall (\boldsymbol{w}_i, b_i) \in \mathcal{C}^*, \\ & \text{MIN}_x \leq \boldsymbol{x} \leq \text{MAX}_x, \end{aligned} \tag{2.6}$$

其中 M 表示类别的数量，$\boldsymbol{z}_j(\boldsymbol{x})$ 表示模型 logits 输出 $\boldsymbol{z}(\boldsymbol{x})$ 的第 j 项。当 $\boldsymbol{x}$ 满足约束 $\boldsymbol{x} \in S_L$ 时，$\boldsymbol{z} : S_L \to \mathbb{R}^M$ 是 $\boldsymbol{x}$ 的线性映射，所以 $\boldsymbol{z}_j(\boldsymbol{x})$ 也可以表示为：

$$\boldsymbol{z}_j(\boldsymbol{x}) = (\nabla_{\boldsymbol{x}} \boldsymbol{z}_j(\boldsymbol{x}^*))^T (\boldsymbol{x} - \boldsymbol{x}^*) + \boldsymbol{z}_j(\boldsymbol{x}^*), \quad \forall \boldsymbol{x} \in S_L \tag{2.7}$$

这也意味着公式2.6中的优化目标函数为凹函数。那么当且仅当 $\boldsymbol{x}^t$ 被分类为类别 t 时，线性区域 S_L 里存在类别 t 的分类区域。我们可以证明，$\boldsymbol{x}^t$ 几乎总是位于线性区域这个凸多面体的面上：

证明： 令 M 表示分类类别的数量，$\boldsymbol{x}^* \in \mathbb{R}^d$为线性区域 S 内一个给定的样本点，∂S 为线性区域 S 的外边界。我们已经知道神经网络在线性区域内表现为一个线性模型，因此模型的 logits 输出 $\boldsymbol{z}(x)$ 的第 j 项可以写作：

$$\boldsymbol{z}_j(\boldsymbol{x}) = \boldsymbol{w}_j^T \boldsymbol{x} + b_j, \quad \forall j \in \{1, 2, ..., M\}. \tag{2.8}$$

根据公式2.6里的定义，令 $p_t(\boldsymbol{x}) = \frac{\exp(\boldsymbol{w}_t^T \boldsymbol{x} + b_t)}{\sum_j \exp(\boldsymbol{w}_j^T \boldsymbol{x} + b_j)}$，那么有 $\boldsymbol{x}^t = \arg\max_{\boldsymbol{x} \in S} \log p_t(\boldsymbol{x})$。如果 $\boldsymbol{x}^t \in S \setminus \partial S$，则 $\boldsymbol{x}^t$ 必定为极值点，这也意味着：

$$\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}^t) = \sum_{i=1}^{M} p_i(\boldsymbol{x}^t)(\boldsymbol{w}_t - \boldsymbol{w}_i) = 0. \tag{2.9}$$

所以有：

$$\boldsymbol{w}_t = \sum_{i=1}^{M} p_i(\boldsymbol{x}^t) \boldsymbol{w}_i. \tag{2.10}$$

由于 $\boldsymbol{w}_i \in \mathbb{R}^d$ 且 $d \gg M$，所以在几乎所有情况有 $\{\boldsymbol{w}_i\}_{i=1}^M$ 线性无关，这也意味着 $\boldsymbol{w}_t$ 不能被表示为 $\{\boldsymbol{w}_{i \neq t}\}_{i=1}^M$ 的线性组合。由于 $0 < p_t(\boldsymbol{x}^t) < 1$，则不存在满足公式2.10的 $\{p_i(\boldsymbol{x}^t)\}_{i=1}^M$。因此我们有 $\boldsymbol{x}^t \in \partial S$. □

虽然我们可以用线性区域内切球大小来大体估计线性区域的大小，但是全面多维度的刻画一个线性区域切分粒度是困难的，而根据多面体的 V 表示或者 H 表示来计算其体积也是一个 #P 难的问题[50]。因此既然 $\boldsymbol{x}^t$ 位于 S 的表面，我们可以将**畸变量** $\max_{t\in\{1,2,\ldots,M\}} \|\boldsymbol{x}^t - \boldsymbol{x}^*\|$ 作为另一个刻画线性区域切分粒度的量。

2.4 不同优化策略下的输出地形

在章节2.1中我们提到过，实际情况下模型泛化性能较好的策略通常为使用具有特定函数搜索偏好的优化策略来优化模型复杂度上限足够高的模型。BN 与 Dropout 就是神经网络中最为常用的两种优化策略。BN 层能够通过对模型的输出进行分布规整从而使得梯度更好地在模型间传递，进而加快模型的训练；除此之外，BN层本身也具有一定的正则化效果，常常带来更强的泛化能力。而 Dropout 层是通过人为地引入噪声带来隐式的集成效果，从而提高模型的泛化能力。这一小节我们将主要介绍使用 BN 或 Dropout 等优化策略时神经网络输出地形对应的偏好及其对泛化性能的影响。

2.4.1 实验设置

为方便分析，实验中我们使用的模型为在 MNIST 数据集[51] 上训练的全连接神经网络以及 CIFAR10 数据集[52]上训练的简单卷积神经网络。

2.4.1.1 全连接神经网络

我们使用了三隐层且每层具有1024个 ReLU 激活的神经元的全连接神经网络，并以此作为我们的基础模型。而其对应的 BN 模型则是在每个隐层输出激活之前加入 BN 层，而 Dropout 模型则是在每个隐层激活之后添加 Dropout 层。在训练这些模型之前，我们将 MNIST 数据集上每张图片的像素值归一化到了区间 $[-1, 1]$ 中。

在实验过程中，我们采用了学习率分别为 $1e-3$ 以及 $1e-4$ 的 ADAM 优化器来对模型进行训练，其中优化器参数 $\beta_1 = 0.9$、$\beta_2 = 0.999$，数据集每个批次的大小为 256。我们使用了 Xavier 均匀初始化[53]来对模型权重进行初始化，并且将偏置项的初始值设置为 0。同时我们使用了早停的技巧来确定训练停止的时间。在这样的优化设置下，各模型的测试准确率如表2.1 所示。我们后续的实验将会通过线性区域来分析这些模型的地形偏好。

在章节2.2.2中我们指出，公式2.4中的线性不等式数量与所有隐层的节点数相等，所以当神经网络规模特别大时很多处理会变得较为困难，这也是使用该方法分析模型的局限性之一。在我们的实验中，线性不等式的数量为 $1,024 \times 3 = 3,072$，同时我们需要加上对784个像素的值约束，即 $\text{MIN}_x \leq \boldsymbol{x} \leq \text{MAX}_x$。因此，每个线性区域会被 $3,072 + 784 \times 2 = 4,640$ 个线性不等式定义。

表 2.1 不同模型的测试准确率十次重复试验后的均值与方差（%）

学习率	基础模型	BN	Dropout
1e-3	97.45±0.24	97.90±0.12	97.81±0.18
1e-4	98.00±0.13	98.35±0.10	98.27±0.13

2.4.1.2 卷积神经网络

我们在 CIFAR10 数据集上使用的卷积神经网络基础框架如表2.2所示。为了简化分析过程，卷积层没有设置边缘补齐，并且我们使用带步长的卷积来代替最大池化层。而相应的 BN 模型以及 Dropout 模型中添加优化层的策略与全连接神经网络一致。

表 2.2 基础卷积模型的详细架构

层	具体参数	激活方式
输入层	输入尺寸=(32, 32)×3	-
卷积层	卷积核尺寸=(3, 3)×32; 步长=(2, 2)	ReLU
卷积层	卷积核尺寸=(3, 3)×64; 步长=(2, 2)	ReLU
卷积层	卷积核尺寸=(3, 3)×128; 步长=(2, 2)	ReLU
全连接层	神经元数量=1024	ReLU
全连接层	神经元数量=10	Softmax

我们使用 ADAM 优化器对模型进行训练，其中学习率[①]为$1e-3$、$\beta_1 = 0.9$，$\beta_2 = 0.999$。同样的，数据集的训练批次大小为 256，且我们使用早停来减少过拟合。对于数据而言，我们将像素值归一化到区间 $[-1, 1]$ 中，并且训练过程中没有采取数据增强等策略。训练过后，基础模型、BN 模型以及 Dropout 模型的测试准确率十次重复实验后的均值分别为 68.1%、70.3% 和 69.6%。

2.4.2 切分粒度实验分析

我们接下来分析上述各类模型的线性区域的切分粒度。为了更加直观清晰地看到不同优化技巧对神经网络片状输出地形的影响，我们在图2-4中直接给出了一个小网络在 MNIST 数据集上使用不同优化技巧后的线性区域及其类别区域的二维切片图，其中第一排图中的灰色线条代表线性区域的区分边界，而颜色代表了该线性区域的神经元激活度，第二排中不同颜色代表不同的类别。模型依然是三隐层的神经

① 这里我们只使用了该学习率，因为较大的学习率出现了训练不稳定的情况，而较小的学习率收敛十分缓慢。

网络，为了显示的清晰性每一层只包含了 20 个神经元。在图2-4中我们可以直观地发现：**相同的模型架构，BN 会将样本空间切分得更加细密；Dropout 也有同样的效果，但集中在决策边界处**。我们后续实验将会通过内切球半径来定量地显示不同优化技巧带来的差异。

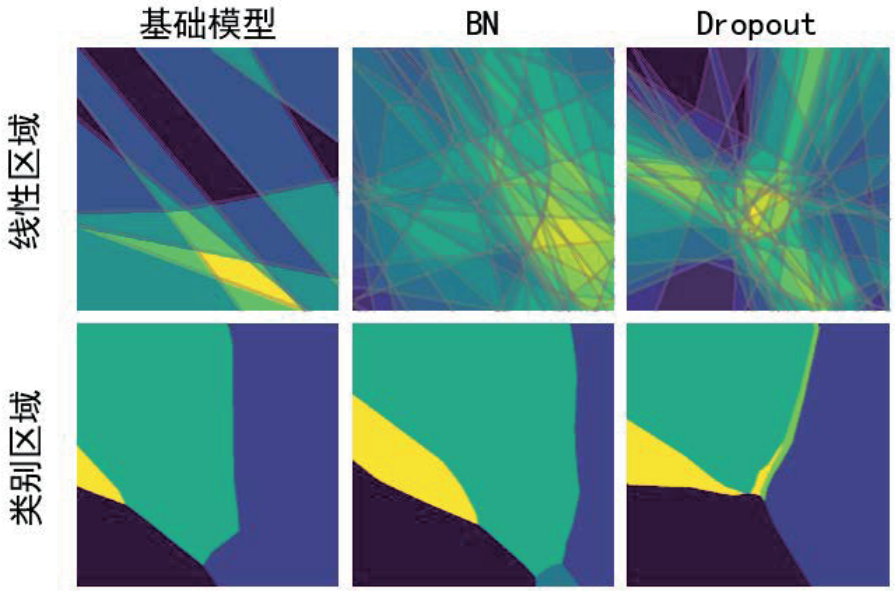


图 2-4 不同优化技巧下的线性区域以及分类区域

为了更好地反映线性区域在样本空间不同位置所带来的影响，我们主要分析了下列三种样本点所在的线性区域：

流形线性区域： 数据集样本点所在的线性区域。

决策线性区域： 包含正常的决策边界的线性区域。我们将正常决策边界定义为某一样本点与其他类别均值中心之间的分界面。

对抗线性区域： 包含对抗决策边界的线性区域。我们将对抗决策边界定义为样本点与其对抗样本之间的决策面。这里我们使用投影梯度下降[54]（Projected Gradient Descent，PGD）来构建对抗样本。

我们使用了线性插值的方法来找寻包含决策边界的线性区域。令 $\boldsymbol{x}$ 表示某一样本点，$\boldsymbol{x}_{target}$ 表示其对抗样本或者其他类别的中心点。我们寻找最大的 $\alpha \in [0,1]$ 使得 $\boldsymbol{x}_\alpha = \alpha \cdot \boldsymbol{x}_{target} + (1-\alpha) \cdot \boldsymbol{x}$ 与 $\boldsymbol{x}$ 位于同一分类区域中。这样，$\boldsymbol{x}_\alpha$ 所在的线性区域中便包含了决策边界。

全连接神经网络： 图2-5展示了不同优化设置下流形线性区域、决策线性区域以及对抗线性区域内径大小的分布。显然我们可以看到，比起基础模型，BN 会使得线性区域的切分粒度更加小。比较流形区域和决策区域的绿色分布我们可以看出 Dropout 也可以使得切分更加细致，但是该效果主要集中在决策边界区域。图2-5还清晰地说明了，对抗区域的切分粒度与流形区域相当，这证明了对抗决策边界与正常决策边界具有本质上的不同。比较不同学习率的切分粒度，我们还可以发现较大的学习率会增大内径分布的方差，但是 BN 对应的内径分布并不会随着学习率的变化发生较大的变化。该现象也证实了之前研究中的观点，即 BN 能够使得模型训练过程对学习率鲁棒，从而可以使用较大的学习率[55]。除此之外，比较图2-5的前两列我们可以发现，决策区域的切分粒度往往小于流形区域，因为决策边界的曲面需要更加复杂的近似。类似的现象也被 Novak 于2018年发现[56]。

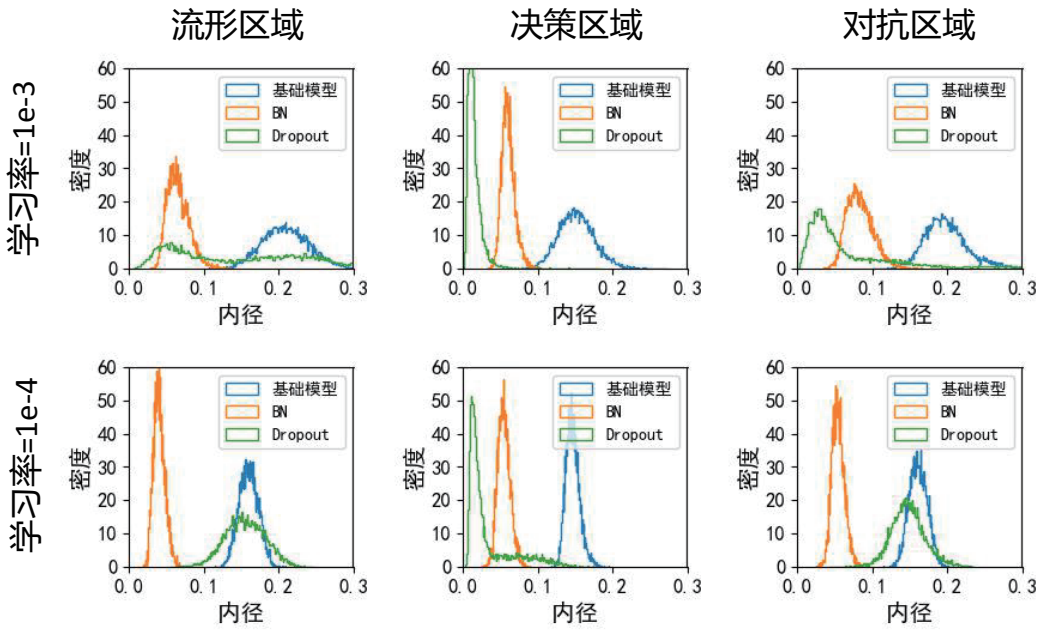


图 2-5 全连接神经网络在不同优化策略下其线性区域的内径分布

卷积神经网络： 我们从 CIFAR10 的测试集中随机抽取了 1000 个测试点来进行分析，其线性区域内切球的半径大小分布展示在了图2-6中。比起 MNIST 数据集，CIFAR10 数据集更加的复杂，其类别的中心点不再被分类为同一类别。为了更好地构建决策线性区域，我们从某一类别中随机采样一个样本点来代替前文中提到的寻找决策线性区域时需要的类别中心点。同全连接神经网络一样，实验的结果同样显示了 BN 与 Dropout 可以使得线性区域具有更小的切分粒度，而 Dropout 主要侧重于决策线性区域而不是流形线性区域。除此之外，在上节中观察到的现象依然出现在了卷积神经网络中，不过卷积神经网络的线性区域更加鲁棒。

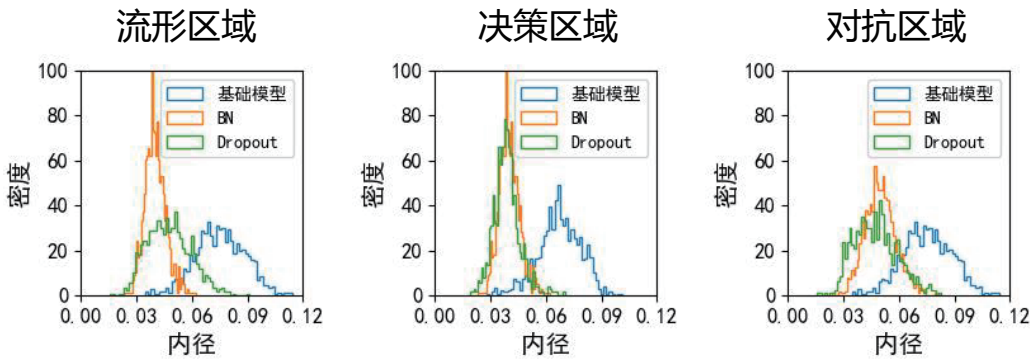


图 2-6 卷积神经网络在不同优化策略下其线性区域的内径分布

2.4.3 决策边界实验分析

之前的研究猜测当两个样本点位于同一线性区域内时有极大概率位于同一类别[57]。然而在这一小节中我们将通过实验说明，大部分线性区域会包含多个分类区域。使用 BN 等技巧虽然可以使得线性区域变小，却不会因此大幅减少线性区域内分类区域的数量，这也导致了对抗样本隐患的出现。

全连接神经网络：我们随机从 MNIST 测试集中采样了 1000 个样本点来搜索其所在线性区域内的分类区域。表2.3给出了每个线性区域内所包含的平均分类区域数量以及畸变量。我们可以发现，当使用较大学习率或使用 BN 以及 Dropout 等技巧可以使得流形区域包含较少的分类区域（需要注意的是，这里 BN 对全连接神经网络的影响与卷积神经网络不一致）。除此之外，BN 显著降低了畸变量，这也从另一个角度说明了 BN 会使得线性区域的切分粒度更小。图2-7 直观地对比了使用不同优化技巧训练模型后某一样本点其 $\boldsymbol{x}^t$ 的变化，其中从左往右 t 从 1 到 10 （数字 0 到数字 9），而绿色边框意味着成功分类到数字 9，而红色边框表明被分类到了类别 t。“L” 意味着使用较大学习率 $1e-3$ 训练，而“S”意味着小学习率 $1e-4$。可以看出，图中的结果也证实了我们的结论。

表 2.3 全连接神经网络流形线性区域包含分类区域的平均数量以及畸变量

指标	学习率	基础模型	BN	Dropout
分类区域数量	1e-3	1.116	1.083	1.012
	1e-4	8.766	2.622	1.048
畸变量	1e-3	26.35	14.90	25.28
	1e-4	23.81	07.61	25.41

从线性区域角度思考决策边界会带来很多方便。比如考虑到模型在线性区域的

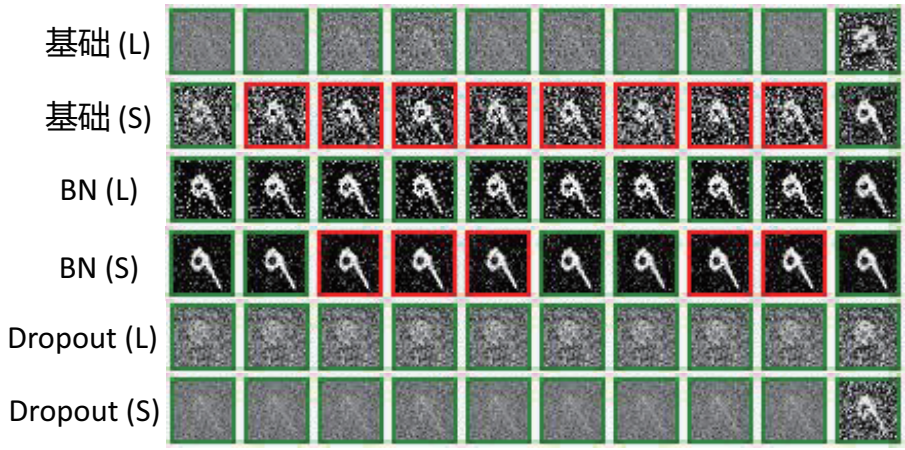


图 2-7 全连接神经网络在不同优化设置下样本点 $\boldsymbol{x}^*$ 所处线性区域内的 $\boldsymbol{x}^t$

线性，我们可以知道，在攻击具有分段线性激活函数的神经网络时，高阶导数的对抗样本产生方法并不会比一阶更好。除此之外，考虑到加入噪声有利于跳出当前的线性区域从而能够搜索更多的空间，这也不难解释为什么使用随机起点的 PGD 会具有更强的攻击性了[54]。

卷积神经网络：上一小节的模型与数据集都相对简单，这一小节我们展示了在 CIFAR10 上训练的卷积神经网络线性区域内决策边界的特性。表2.4给出了实验中卷积神经网络流形线性区域包含分类区域的平均数量以及畸变量。实验结果表明，线性区域内分类区域的多少与其线性区域的切分粒度并不存在着必然联系。比如，Dropout 虽然消除了大部分的分类区域，但是其并没有大幅度的降低流形线性区域的畸变量，而 BN 减小了切分粒度，却使得同一线性区域内的分类区域变得更多。由此可见，使用 BN 过后，模型的对抗鲁棒性会进一步降低，这也与之前文献中描述的现象一致[58,59]。我们在图2-8 展示了一个类别为“汽车”的样本所对应的 $\boldsymbol{x}^t$，其中从左往右 t 从 1 到 10（飞机，汽车，鸟，猫，鹿，狗，青蛙，马，船，卡车），绿色边框意味着成功分类为“汽车”，而红色边框表明被分类到了类别 t。我们同样可以发现 BN 使得 $\boldsymbol{x}^t$ 具有较小的畸变量以及较差的鲁棒性。

图 2-8 卷积神经网络在不同优化设置下样本点 $\boldsymbol{x}^*$ 所处线性区域内的 $\boldsymbol{x}^t$

表 2.4 卷积神经网络流形线性区域包含分类区域的平均数量以及畸变量

指标	基础模型	BN	Dropout
分类区域数量	7.811	9.984	2.901
畸变量	27.08	20.85	25.25

2.5 本章小结

本章对神经网络输出地形的复杂度与泛化性能之间的关系进行了分析，说明了具有较高模型复杂度的模型容易因为过拟合训练集而形成较为复杂的输出地形，因此分析模型的输出地形复杂度可以在一定程度上反映模型的泛化行为。本章以使用如 ReLU 等分段线性函数作为激活的神经网络为例，证明了这类广泛使用的模型的输出地形为片状结构（即线性区域），而神经网络在每个线性区域内表现为一个线性模型。通过逐层计算，我们获得了线性区域的 H 表示，并在此基础上分析了线性区域的切分粒度以及其内部的决策边界。我们发现：

- BN 和 Dropout 能够使同样的架构获得更加细小的切分粒度，从而能够具有更强的拟合能力。其中 BN 会使得整个空间的切分粒度都较小，而 Dropout 侧重于细化边界处的线性区域切分。

- 线性区域的大小与其内是否包含决策边界没有必然的联系。某些情况下甚至较小的线性区域可能包含更多的分类区域（如 CIFAR10 上使用 BN 过后的卷积神经网络），这大大降低了神经网络的对抗鲁棒性。

由上可知，线性区域的性质与模型的泛化行为紧密相关，这为通过考察每个训练回合的输出地形来对模型回合双下降现象进行探究提供了可能。然而，使用线性区域来讨论神经网络的输出地形存在着以下的局限与不足：

- 在实际情况中，神经网络包含有大量的神经元，这意味着线性区域的 H 表示极其庞大，为后续计算增添了难度。

- 较为简单的输出地形依然可能具有较小的切分粒度。例如，我们可以使用多个的线性区域去拟合一个平面，这样虽然输出地形复杂度并不高，但最后的切分粒度却较小。

- 很多模型会使用如 Sigmoid 等非分段函数来激活神经元，这阻碍了使用线性区域来分析输出地形。

综上，我们虽然在本章说明了神经网络片状地形与模型泛化性能之间的紧密联系，但是在实际情况中我们很难通过度量线性区域的切分粒度来分析模型的回合双

降现象。在下一章中，我们将详细地论述在实际应用中使用神经网络片状地形来分析回合双降现象的缺点，然后在其基础上提出一种更加简便且合理的输出地形复杂度的估计方式，进而解释回合双降现象中神经网络的独特表现。

3 回合双降现象下的地形频谱

上一章我们已经验证了神经网络输出地形与其泛化行为之间的联系。本章详细分析了上一章提到的片状地形分析方法的局限与不足，并在此基础上提出了一种通过检测输出地形的频率成分来获取模型地形复杂度的方法。实验发现，神经网络输出地形的高频分量在训练后期会由增加变为减少，这意味着输出地形在训练后期重新被正则化，这也导致了模型泛化误差回合双降现象的出现。进一步地，我们发现通过检测训练集上计算的高频分量峰值便可以定位模型泛化误差二次下降的起点。除此之外，我们发现输出地形的高频分量出现衰减的原因主要是非流形区域的输出地形变得更加平坦，这也解释了为什么实验中训练样本中的噪声点依然被模型记忆但是其泛化误差却出现了二次下降。

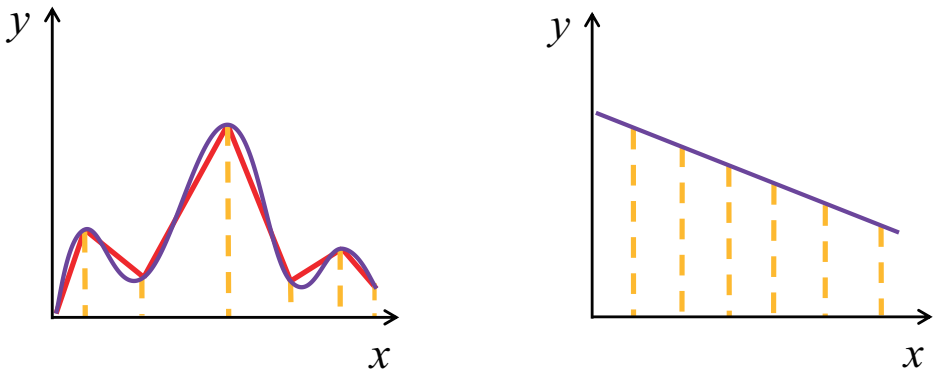
3.1 片状地形分析的局限性

在上一章的小结部分中我们已经大体说明了使用神经网络片状地形来度量模型地形复杂度的局限与不足，这一小节我们将会进行更加详细地说明。

首先我们可以知道，神经网络线性区域 H 表示中的不等式数量与模型神经元数量相等。这意味着当模型规模较大时，我们需要庞大的线性不等式组来刻画线性区域。举例来看，一个普通的 ResNet18 便具有大约三百万个神经元，即我们需要大约三百万个线性不等式来表示一个线性区域。这样庞大的不等式集合使得解决后续分析中可能出现的凸优化问题变得极为耗时且麻烦，因此很难将该方法应用于稍微大型的神经网络上。

其次，线性区域切分粒度越细并不一定意味着模型的输出地形更加复杂。诚然，复杂的输出地形必然意味着需要更加细致的线性区域来拟合，但反过来却并非一定成立。图3-1 给出了一个复杂地形与简单地形的拟合示意图，其中 x 为一维输入，y 为一维输出，紫线代表实际需要拟合的函数地形，橙色的虚线表示了每个线性区域的端点，而红线代表了每个线性区域拟合的地形（右图中红线与紫线重合）。在图3-1 中，一个复杂的地形使用了 6 段线段来拟合，而简单地形也使用了 6 段线段来拟合。从切分粒度出发的话，两者的输出地形复杂度相当，但显然简单地形中的各线性区域拟合的模型完全相同，因此其实际的输出地形复杂度相当低。虽然真实场景中上述情况很少出现，但是使用切分粒度度量模型地形输出复杂度的不严谨性可见一般。

最为重要的一点是，并非所有神经网络都具有片状地形。在上一章中的神经网络因其隐层的激活函数皆为类似 ReLU 的分段线性函数，从而使得拟合的函数也表现为分段线性，也即片状地形。然而虽然现实应用中分段线性激活函数因其计算简



(a) 拟合复杂地形　　(b) 拟合简单地形

图 3-1 神经网络划分线性区域的简单示意图

单且方便优化等优点被大量使用，但仍然有不少场景必须使用 Sigmoid 或 tanh 等平滑的激活函数。在这样的情况下，神经网络的输出地形也会呈现出平滑的特性，从而无法有效地构建线性区域。诚然，我们可以用分段线性激活函数来逼近 Sigmoid 或者 tanh 函数，但是当段数较少时其近似程度并不能得到保证，而段数较多时又会产生大量的线性区域而导致分析的困难，因此不具有实际的可操作性。

需要说明的是，这些局限与不足并不意味着神经网络输出地形的复杂度分析与模型泛化能力之间的联系本身存在着问题，而是对一种适用性与实用性更强且更加简洁的新方法提出了需求。针对上一章方法中的缺陷，新方法需要做到以下三点才能更好地被用于分析神经网络回合双降现象中的输出地形复杂度变化：

- 能够对更大型的神经网络进行分析。
- 与输出地形复杂度的联系更加紧密。
- 能够应用于非分段线性的神经网络。

3.2 神经网络输出地形的频谱

鉴于片状地形分析的种种弊端，本节我们根据上一章提到的模型输出地形复杂度与泛化性能之间的紧密关系，介绍了已有的关于神经网络学习偏好的研究，并且对神经网络输出地形的复杂度与其频谱图之间的关系进行了说明。更进一步地，我们提出了一种新的更加简洁的计算模型输出地形频谱的方法，该方法能被应用于实际的训练场景中，并且为下一节模型回合双降下输出地形的频谱研究提供了分析基础。

3.2.1 神经网络的频谱偏好

传统机器学习理论中，一个复杂度极高的模型虽然能够将经验误差降到几乎为零，却并不具有较好的泛化能力[2]。但是，这显然并不适用于现代的深度神经网络：Zhang 等通过实验证明，即使一个神经网络的模型复杂度大到能够记忆所有的随机样本，它依然能够在正常样本集上保有较好的泛化能力[8]。该现象并不能被传统机器学习中的 VC 维或 Rademacher 复杂度所解释。

一些研究人员将这个违反直觉的现象归因于神经网络训练过程的一种隐式学习偏好（Learning Bias）：尽管神经网络具有较大的假设空间，随机梯度下降算法（Stochastic Gradient Descent，SGD）会更加倾向于搜索具有较好泛化能力的那部分假设空间。章节2.1中我们也说明了类似的问题，即模型对应的函数空间在实际训练过程中会体现出一定的偏好概率分布，即函数空间内的函数并不会被等概率地搜索到，这也就起到了一个隐式的正则化作用。Arpit 等实验发现，深度神经网络首先会学习到训练集中的模式，然后开始暴力记忆训练集中不能够泛化的噪声[14]。Kalimeris 等更加细致地展现了这个过程，他们通过互信息的方法说明 SGD 优化器在神经网络训练过程中会不断增加拟合出来的模型的复杂度[15]。更进一步地，一些研究表明，神经网络在训练的过程中会先拟合目标输出地形的低频分量，然后再拟合更加高频的分量，这也被称作神经网络的“**频谱偏好**”或者“**F-准则**”[16–18]。这些研究也说明了，随着模型在训练过程的复杂度不断增加（高频分量不断被引入），神经网络会开始过拟合训练集中的噪声部分从而泛化能力变差。

本章内容将会从模型输出地形的频谱出发说明神经网络泛化误差回合双降现象出现的原因。由于神经网络输出地形的频谱为本章的关键之处，故此我们将仔细地说明其与神经网络复杂度之间的关系。首先我们要意识到，高频的分量意味着更加复杂的变化。在一个固定范围的样本空间中，模型输出地形所含的高频分量越大，便意味着其在该范围内变化得越频繁，进而说明输出地形越崎岖。章节2.1已经提到，崎岖的输出地形需要复杂的模型来拟合。由此，我们通过观察模型输出地形的频谱，便可以定量地测度模型实际复杂度的变化，进而能够更好地探究模型回合双降现象背后的原因。

3.2.2 频谱计算的方式

在使用输出地形频谱分析来探究模型泛化误差回合双降现象之前，我们需要知道对模型在高维样本空间上的输出地形进行完整的频谱分析是十分困难的，因为搜索模型输出的代价会随着样本空间维度的增加指数级上升。Rahaman 等利用了 ReLU 神经网络分段线性的特点给出了频谱的严格表达式，然而他们的方法不仅受限于分段线性的约束，并且依然很难用于实际的高维数据集上[16]。因此，虽然他们的实验说明了神经网络拟合输出地形具有从低频到高频的频谱偏好，但实验本身局

限在了简单的人造数据集而非真实场景下的数据集。Xu 等使用了非一致性的离散傅里叶变换（None-uniform Discrete Fourier Transform）来计算样本集在整个空间下的全局频谱，但是由于高维空间下样本天然的稀疏性，其频谱精细化不足且不能够很准确地抓住神经网络局部的性质[18]。为了解决上述问题，在本章中我们提出了一个启发式的但在实际应用中更加方便可行的方法来度量神经网络输出地形的频谱。考虑到数据集一般为高维空间下的低维流形分布，我们没有试图捕捉整个空间中输出地形的频率分量，而是侧重于考虑样本点周围局部的输出地形频谱特性。

具体而言，令 $\boldsymbol{x} \sim \mathcal{D}$ 为从数据集分布 $\mathcal{D}$ 中采样的样本点输入部分，$\boldsymbol{v}_x$ 为 $\boldsymbol{x}$ 上随机选取的一个归一化方向向量, 神经网络 logits 输出向量的第 c 项为 $f_c(\boldsymbol{x})$（其中 $c \in \{1, 2, ..., C\}$，C 为分类类别的数量）。接下来我们会从 $[\boldsymbol{x} - h\boldsymbol{v}_x, \boldsymbol{x} + h\boldsymbol{v}_x]$ 上均匀采样 N 个点来作为离散傅里叶变换（Discrete Fourier Transform，DFT）的输入，其中 h 限定了采样的范围。由此可知，$f_c(\boldsymbol{x})$ 对应的这些采样点计算出来的频谱可以写作：

$$\tilde{f}_{c,\boldsymbol{x}}(k) = \sum_{n=1}^{N} f_c\left(\boldsymbol{x} + \frac{2n - N - 1}{N - 1} h\boldsymbol{v}_x\right) e^{-i2\pi \frac{n}{N} k}. \tag{3.1}$$

其中 k 代表了频率分量标号，也即是频率的大小，后续论文中为简单起见将 k 称为“频率”。注意这里我们使用的模型 logits 输出而非概率输出，这样频谱就不会被神经网络的重参数化影响。比如，当我们将神经网络最后一层的模型参数乘上任意 $\alpha > 0$，此时模型的决策边界并不会发生任何变化，但是如果使用概率输出计算频谱，那么其频谱会随着 α 产生非线性的变化，这显然与我们的期望相违背。

为了获得样本点邻域输出地形更为准确且稳定的频谱成分，我们将多个样本点的各频率能量对应相加，即：

$$A_k = \frac{1}{C} \sum_{c=1}^{C} \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}} \left| \tilde{f}_{c,\boldsymbol{x}}(k) \right|^2. \tag{3.2}$$

为了能够更好地比较不同训练阶段下模型频谱的能量，我们观察 A_k 所占总能量的对数比例，即：

$$R_k = \log \frac{A_k}{\sum_j A_j}. \tag{3.3}$$

在后续实验中我们发现，实际情况下我们只需要很少一部分样本点就可以较为准确地估计公式3.2 中 $\left| \tilde{f}_{c,\boldsymbol{x}}(k) \right|^2$ 的期望（按照我们实验中所测试的，通常 500 个样本点就足够稳定）。另外需要注意的是，该方法中计算 A_k 并不需要使用数据集的任何标签信息，这也意味着该方法也能在无监督学习或半监督学习的场景下使用。

3.3 回合双降下的频谱分析

上一小节提出的计算频谱的方法简单易操作，计算过程与神经网络结构本身无关且不需要利用标签信息，因此能够被用来分析神经网络泛化误差回合双降现象的输出地形特点。这一小节中我们将根据该频谱计算方式来考察训练过程中模型的输出地形频谱特点。我们实验发现，神经网络泛化误差回合双降现象出现的本质是模型在后期训练的过程中被隐式地正则化，反映在频谱分析上则是模型输出地形的高频分量降低。更进一步地，我们发现通过计算高频分量的峰值，我们能够在不使用校验集的情况下预测神经网络泛化误差二次下降的起点。

3.3.1 频谱偏好与泛化误差回合双降之间的矛盾

前文提到，神经网络的训练过程具有隐式偏好，从而使其有方向性地搜索庞大函数空间中的某一部分子空间，比如从学习数据集模式到过拟合噪声，从低复杂度到高复杂度，或者说从拟合输出地形的低频分量到其高频分量。但是之前的这些发现都基于一个共同的假设：**神经网络训练过程中的学习偏好是单调变化的**。比如，在训练过程中神经网络只会逐渐地引入高频分量，且这个过程不会出现逆向的情况。从这个角度出发，我们可以较好解释传统学习理论中模型泛化误差所显示的“ 欠拟合-早停点-过拟合”的 U 型曲线。然而，**神经网络泛化误差回合双降现象却对神经网络的隐式学习偏好的单调性提出了挑战**。如果我们认可频谱偏好的单调性，那么可以预料随着高频分量的不断引入，模型会不断去过拟合训练集中的噪声，从而使得泛化误差单调地增加而不再具有第二次泛化误差下降的可能。

为了更好地理解神经网络泛化、记忆以及频谱偏好之间的联系，我们在出现回合双降现象的实验设置下（随机引入一定比例的标签噪声[①]，并且增加训练回合数）对学习到的函数的输出地形频率成分进行了分析。在实验中我们发现，之前研究中假定的神经网络训练过程的学习偏好单调性并不成立。图3-2中给出了 ResNet18 在 CIFAR10 上使用 ADAM 优化器（学习率 $1e-4$）训练 1000 个回合的结果，其中浅蓝色的竖直线标注出了模型在训练过程中三个阶段（*学习*、*记忆*、*误差二次下降*）的分割线，而横坐标为按照对数比例显示。我们可以看到，在学习阶段噪声集错误率较高，表明模型此时在学习数据集中的分类模式；到了记忆阶段，噪声集上的误差快速下降，此时模型开始记忆噪声，从而使得测试误差出现上升。从 R_k 对应的地形图中我们可以看出，在前两个过程中高频分量被不断地引入到神经网络的输出地形中。这个过程符合传统学习理论中所描述的由于频谱偏好单调性导致的泛化误差 U 型曲线。然而，随着训练过程进一步推进，高频分量的含量开始下降，使得模型的泛化误差出现第二次下降，该现象解释了模型回合二次下降现象并且反驳了神经网络频谱的单调性，在下一小节中我们将会更好地探究这个现象。除此之外，一

① 在后续我们将被扰动的部分称为噪声集。

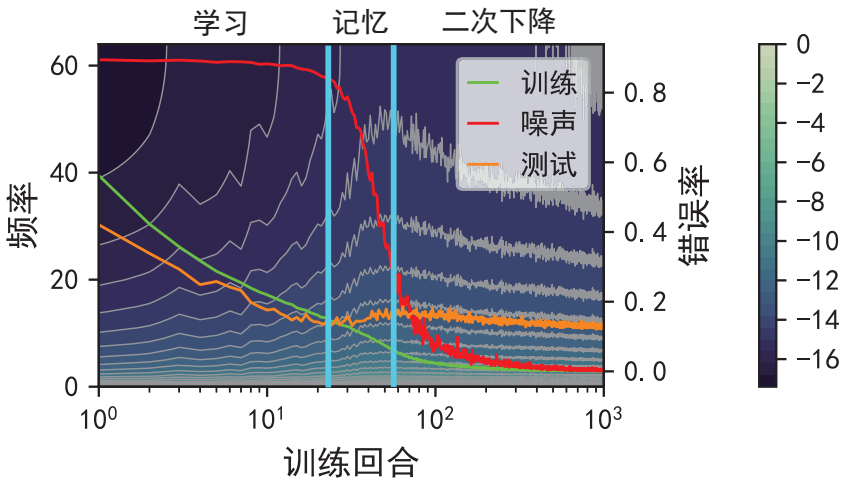


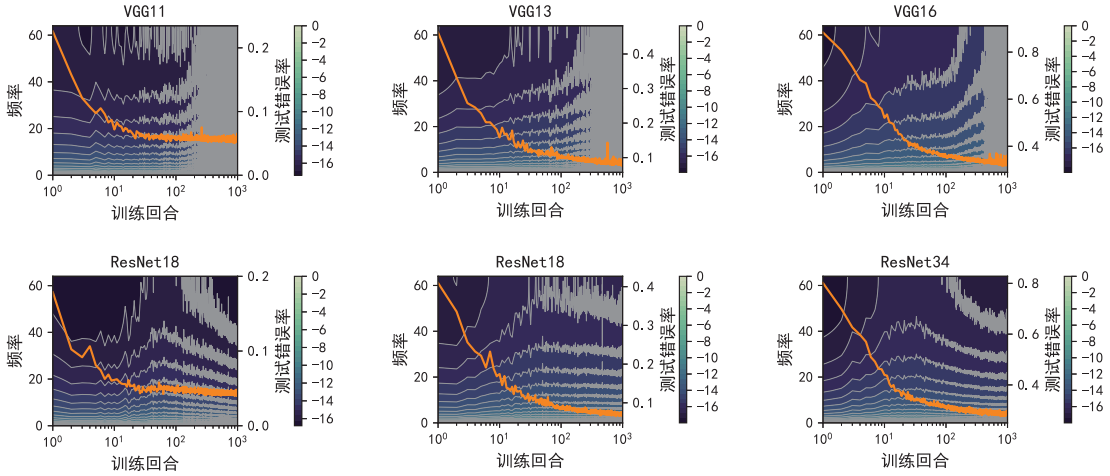
图 3-2 ResNet18 在 CIFAR10 上训练的频谱地形图 R_k 及对应的模型在训练集、噪声集、测试集上的误差

个令人疑惑的点在于，在泛化误差第二次下降时，噪声集依然被拟合而并没有被忘记。由此我们又发现了另一个反常的现象：当高频分量下降时，神经网络的噪声却依然被过拟合。该现象不寻常的原因在于，噪声集意味着数据点在训练流形上具有错误的标签，该情况近似于在输出地形上拟合一个狄拉克函数，考虑到狄拉克函数本身具有宽频谱，因此要求更加多的高频分量来拟合它。在章节3.4 中我们将详细探究其底层的原因。

3.3.2 频谱的非单调性变化

之前的研究假设了频谱偏好的单调性，即神经网络从低频到高频地拟合目标输出地形。然而，这一小节我们将通过在三个数据集上的多个模型上证明，这种单调性并不总是成立，且正是频谱偏好的非单调性导致了神经网络回合双降现象的发生。

我们的实验设置与 Nakkiran 等论文中观察到泛化误差回合双降现象时的实验设置基本保持一致[19]。我们这里考虑 VGG[10] 和 ResNet[11] 这两类架构在 SVHN[60]、CIFAR10 和 CIFAR100[52]三个数据集上的表现（其中 SVHN 数据集上我们使用了 VGG11 与 ResNet18，CIFAR10 上使用了 VGG13 以及 ResNet18，CIFAR100 上使用了 VGG16 以及 ResNet34。模型改写自https://github.com/kuangliu/pytorch-cifar。）。我们将图片像素点的值归一化到 $[0,1]$ 之间，并且随机打乱了一定比例训练集样本的标签（噪声集）。训练批次的样本数为 128，我们使用学习率为 $1e-4$ 的 ADAM 优化器[34]训练 1000 个回合。训练过程中，我们使用了数据增强来提高模型的泛化性能。在每个训练回合结束后，我们从训练集中随机采样 500 个样本点，然后对每个样本点随机采样一单位方向向量 $\boldsymbol{v}_x$，然后在其上根据公式3.3 计算各频率的对数能量占



(a) SVHN (b) CIFAR10 (c) CIFAR100

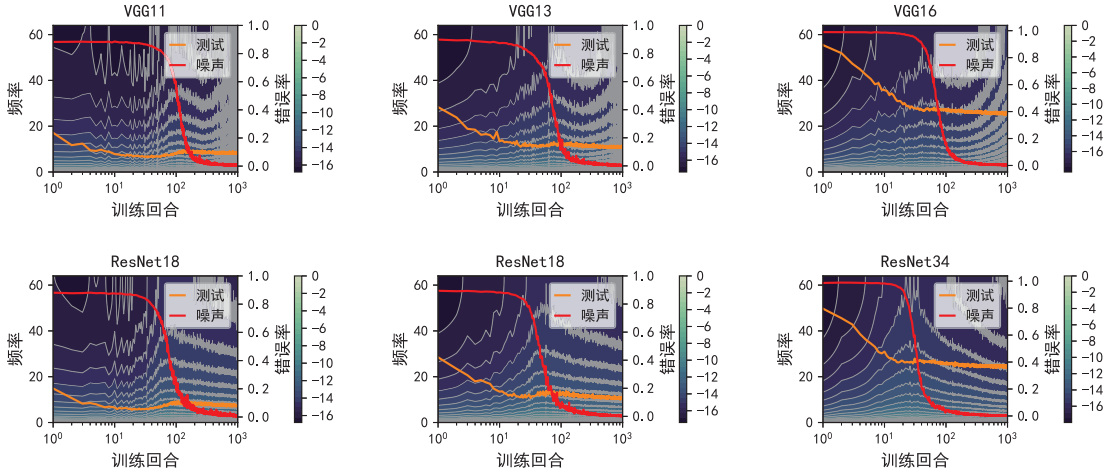
图 3-3 噪声集占比 0% 时模型的测试错误率与频谱 R_k

比 R_k。实验中，我们设置超参数 $h = 0.5$ 且 $N = 128$。

图3-3、3-4与3-5分别给出了扰动 0%、10% 以及 20% 训练集标签后的测试误差、噪声集误差以及每个训练回合对应的输出地形频谱变化。首先我们可以观察到当加入标签噪声时，模型的泛化误差回合双降现象出现：测试误差首先快速下降，然后在过拟合噪声集时略微上升，接着虽然噪声集上依然保持过拟合，但是模型的泛化误差却开始了第二次下降。而图中的 R_k 提供了另一个视角来观察这个现象。首先之前研究假设的频谱偏好单调性在学习和记忆阶段依然成立：模型首先引入低频分量然后才是高频分量，且高频分量的占比在模型试图记忆噪声集时快速上升。然而这种高频分量的不断引入并没有持续，而是在后期转变为消失，从而导致了模型泛化误差的二次下降。除此之外，我们可以发现随着噪声集的占比增加，模型的高频分量占比也会对应增加。这个现象出现主要因为流形空间上存在着更多的类似狄拉克函数的地形需要拟合，从而需要更多的高频分量的引入。更有趣的一点是，实验结果表明了模型结构会显著影响模型的频谱偏好。比较 ResNet 以及 VGG 两种架构在训练后期的表现，我们可以发现跳接（Skip Connection）这样的结构使得模型更加倾向于去探索低频分量占比较多的函数空间。这样的函数对应的输出地形具有较低复杂度，因而常常具有更好的泛化性能。该发现能为研究人员更改模型架构提供一定的启发作用。

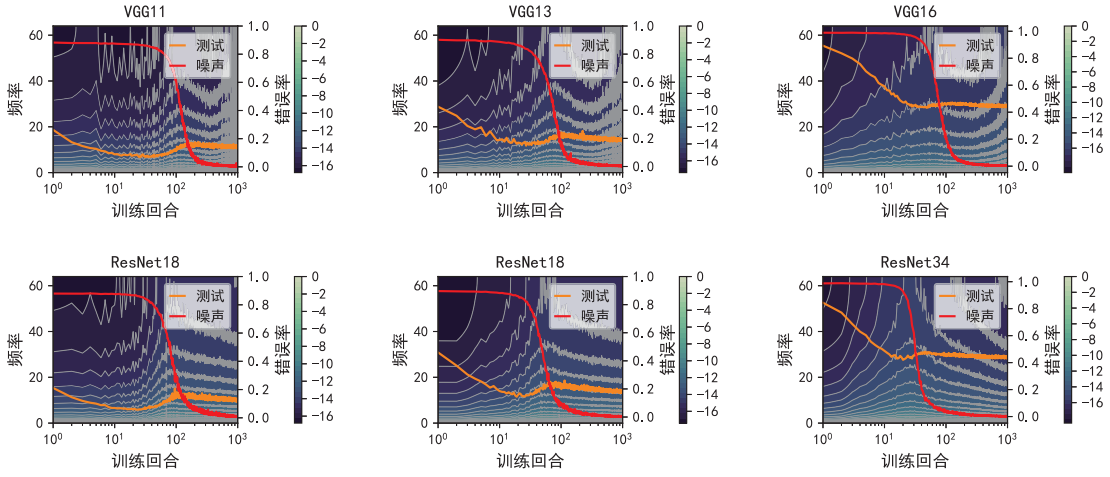
3.3.3 预测误差二次下降的起点

从上一节的实验结果中我们看到 R_k 的峰值与泛化误差二次下降的起点具有某种同步性。基于这个发现，我们在这一小节中提出一种只使用训练集便可以知道测试误差二次下降起点回合的方法。



(a) SVHN (b) CIFAR10 (c) CIFAR100

图 3-4 噪声集占比 10% 时模型在噪声集、测试集上的错误率以及频谱 R_k



(a) SVHN (b) CIFAR10 (c) CIFAR100

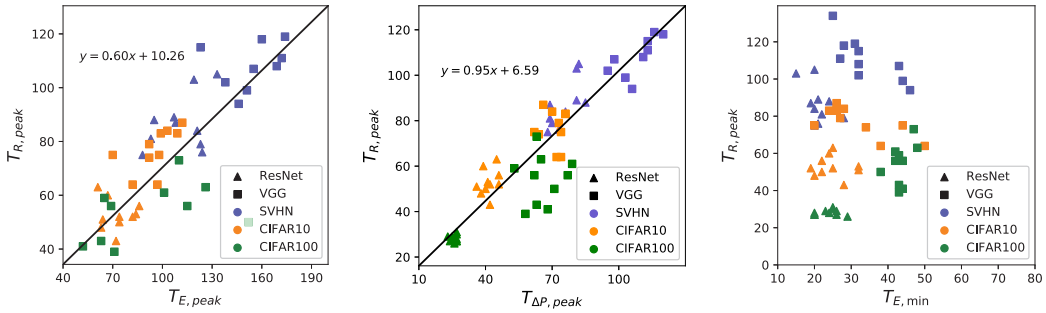
图 3-5 噪声集占比 20% 时模型在噪声集、测试集上的错误率以及频谱 R_k

由于模型会过拟合训练集上的噪声，故使用训练错误率来监视测试错误率是一种错误的行为。归根结底，这个错误的本质是训练误差与测试误差在训练过程中的不一致变化，特别是在训练回合数较大时。这种不一致性出现的原因在于优化目标包含了训练误差而没有包含测试误差。所以，如果我们想要找到一个在训练集上计算却能够反映测试特性的指标，那么最重要的一步在于切断这个指标与优化目标的直接联系。显然，R_k 符合这个性质，其既没有出现在优化目标中，也没有使用标签的信息。在图3-4与3-5中，我们可以清晰的观察到训练集上计算的 R_k 与测试错误率二次下降起点之间的某种同步性。接下来我们将这种预测性定量地显示出来。

令 $R_{k,t}$ 表示第 t 个训练回合时的 R_k （为了结果的稳定性，$R_{k,t}$ 使用窗口为 10 个回合的均值滤波来平滑），E_t 为对应回合的测试错误率。与早停法类似，我们在忍耐回合数（Patience）为 30 的超参下搜索 E_t 以及 $R_t = \sum_k \alpha_k R_{k,t}$ 的峰值，其中 α_k 为 $R_{k,t}$ 对应的加权权重（在我们的实验中全部设置为1）。更具体地来说，我们使用了两次早停法：第一次我们试图找到最小的 R_t 与 E_t 对应的回合，分别用 $T_{R,\min}$ 与 $T_{E,\min}$ 来表示；然后，我们使用 $T_{R,\min}$ 与 $T_{E,\min}$ 作为起点来搜索最大的 R_t 与 E_t 对应的回合，分别用 $T_{R,\text{peak}}$ 和 $T_{E,\text{peak}}$ 来表示。

我们在多个数据集上训练了多种模型架构（SVHN：ResNet18 与 VGG11；CIFAR10：ResNet18 和 VGG13；CIFAR100：ResNet34 和 VGG16）。每组实验在不同占比的噪声集（10% 和 20%）进行且各自重复了五次，然后我们考察频谱峰值与各种泛化行为之间的联系，结果详见图3-6，其中不同的颜色代表不同的数据集，不同的形状表示不同的模型架构，黑色直线通过线性回归拟合（（图3-6(a)与**??**对应的皮尔逊相关系数分别为0.88和0.92。）。首先，从图3-6(a)上我们可以清晰观察到 $T_{R,\text{peak}}$ 与 $T_{E,\text{peak}}$ 的正相关的线性关系，这意味着我们可以只使用训练集便可以检测到泛化误差二次下降的起点。除此之外，我们还可以看出 $T_{R,\text{peak}}$ 与模型在噪声集上错误率下降的速度息息相关。令 P_t 表示第 t 回合的噪声错误率，而 $\Delta P_t = -(P_t - P_{t-1})$ 为其下降的速率。由于 ΔP_t 具有较大的波动，我们搜索平滑过后 $\overline{\Delta P}_t = \frac{1}{2\Delta T+1}\sum_{i=-\Delta T}^{\Delta T} \Delta P_{t+i}$ 的峰值（我们设置 $\Delta T = 5$），并且将搜索到的回合数用 $T_{\Delta P,\text{peak}}$ 表示。图3-6(b)的实验结果表明，当高频分量占比到达其峰值时，模型在噪声集上的错误率下降得最快，这进一步说明了模型频谱可以用来指示模型的泛化行为。遗憾的是，如图3-6(c) 所示，我们并没有观察到早停点与 R_k 峰值之间的关系，我们将在章节4中解决这个问题。

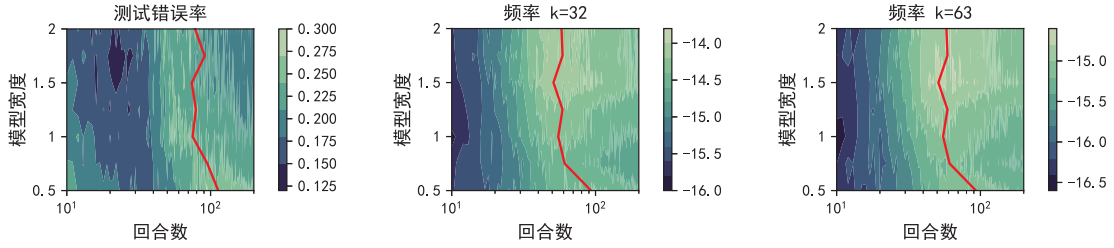
为了更好的展示频谱峰值与模型泛化误差二次下降起点之间的联系，我们通过变换模型的宽度来观察峰值与泛化误差二次下降起点的移动情况。我们使用不同宽度的 ResNet18 在 CIFAR10 上使用学习率为 $1e-4$ 的 ADAM 优化器训练了 200 个回合。对每层卷积层，我们将滤波器的数量设置为原始的 0.5 倍到 2.0 倍，然后我们检测频谱峰值与泛化误差二次下降起点之间的同步性。图3-7展示了实验的结果，其中红色线条表示了测试错误率以及 R_k 峰值的位置。。首先我们可以观察到，当模型变



(a) (b) (c)

图 3-6 (a) $T_{R,\text{peak}}$ 与 $T_{E,\text{peak}}$；(b) $T_{R,\text{peak}}$ 与 $T_{\Delta P,\text{peak}}$；(c) $T_{R,\text{peak}}$ 与 $T_{E,\text{min}}$

宽时，变大的模型复杂度允许拟合输出地形时引入更多的高频分量；其次，如图中红色线条所示，频谱峰值与测试错误率的峰值同步移动，这进一步证明了输出地形频谱与神经网络泛化表现之间的紧密联系。



(a) Test errors (b) R_{32} (c) R_{63}

图 3-7 不同模型宽度下测试错误率以及 R_k

3.4 频谱分析的流形解释

标签噪声对输出地形的高频分量提出了潜在的需求，因为扰动的点类似于输出地形上具有宽频谱的狄拉克函数。但是，章节3.3.2中的实验结果显示，在训练后期虽然输出地形的高频分量开始下降，但是噪声集依然被神经网络所记忆。于是我们需要回答一个新的问题：*为什么高频分量的下降并没有影响噪声集的拟合?*

首先我们注意到，噪声集在泛化误差第二次下降时依然被记忆，那么模型泛化性能在这个阶段的提升并不来源于拟合训练流形或者忘记训练流形上的噪声。这意味着模型泛化性能的提升应该从训练流形外的区域寻找原因，比如训练流形以外输出地形的正则化。根据这个想法，我们将输出地形的频谱分开考虑：训练流形的频谱以及非训练流形的频谱。在后续实验中我们将说明，章节3.3.2中展示的频谱复杂

变化是两个过程的组合：**训练流形上的输出地形持续地引入高频分量来拟合噪声点，而非训练流形的输出地形频谱逐渐偏好低频分量**。

为了证明该猜想，我们设计了一个简单任务来追踪两种频谱的变化。该任务要求分开两条三维空间中互相垂直且不相交的直线（为了引入高频分量的拟合需求，其中一条直线的某点标签被扰动）。接下来在这个易分析的框架下，我们可以考察训练流形以及非训练流形两种频谱情况及其泛化表现。图3-8给出了这个任务的简单示意图，其中蓝色的平面代表理想情况下划分直线 l_0 与 l_1 的最优分类边界，被扰动的点打标为类别 1，非训练流形的直线 $l_\perp$ 与训练流形 l_0 垂直相交。这个分类的任务便是在 l_0 与 l_1 上训练模型，并且观察在 l_0 以及 $l_\perp$ 上的频谱。

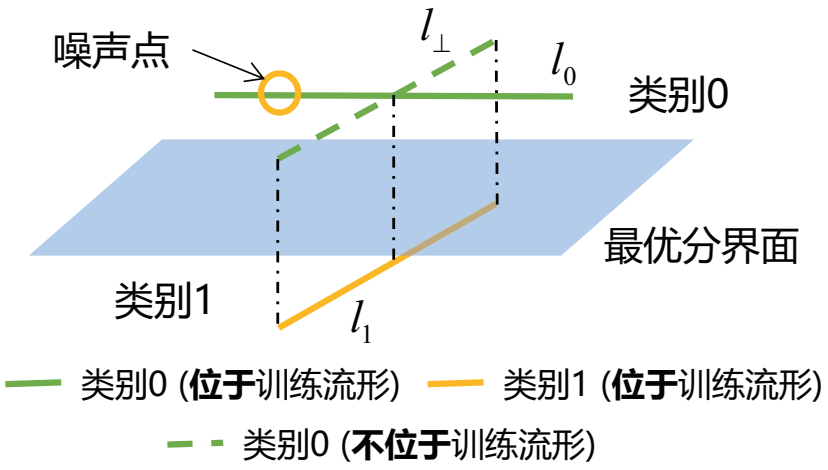


图 3-8 分类任务的简单示意图

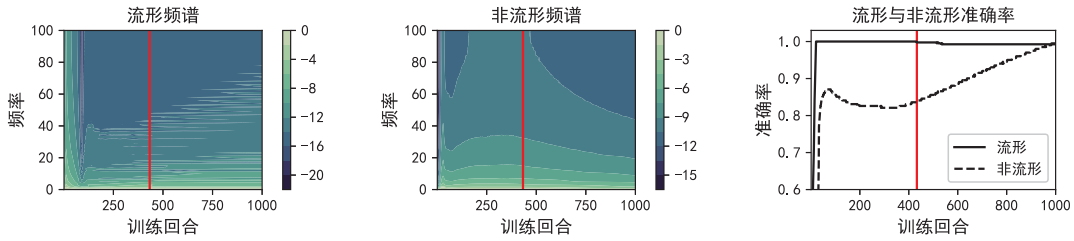
具体而言，我们定义 l_0 以及 l_1 为：

$$l_i = \{\boldsymbol{r}_i + k \cdot \boldsymbol{v}_i | -1 \leq k \leq 1\}, i \in \{0, 1\}, \tag{3.4}$$

其中 $\boldsymbol{r}_0 = [0.1, 0.1, 0.1]$, $\boldsymbol{r}_1 = -\boldsymbol{r}_0$，$\boldsymbol{v}_0 = \frac{1}{\sqrt{2}}[1, -1, 0]$，$\boldsymbol{v}_1 = \frac{1}{\sqrt{6}}[1, 1, -2]$。对于训练集而言，我们在 l_0 和 l_1上分别均匀采样 51 个点，并且将 l_i 上采样的点标记为类别 i（l_0 上随机选取的被扰动的点除外）。对于测试集而言，我们在 l_0 和 l_1上分别均匀采样 201 个点，且不做任何扰动。我们使用具有两层各包含 100 ReLU 神经元的神经网络来拟合训练集。学习率为 $5e-4$ 的 ADAM 优化器被用于训练。在训练过程中，我们在训练流形 l_0 以及非训练流形 $l_\perp = \{\boldsymbol{r}_0 + k \cdot \boldsymbol{v}_1 | -1 \leq k \leq 1\}$ 上计算输出地形频谱。需要注意的是，由于任务的简单性，我们这里可以直接在对应的直线上进行离散傅里叶变换来获取精准的频谱。

图3-9展示了训练流形以及非训练流形的频谱以及对应的准确率，其中红色直线标示了噪声点开始被记忆的回合。如图3-9(a) 所示，训练流形上的频谱持续引入高频分量来记忆噪声点。显然，这种情况并没有出现在非训练流形的频谱上。

图3-9(b)中显示，当噪声点被拟合记忆以后，非训练流形的频谱会更加偏向于低频分量，从而使得非训练流形的输出地形更加平坦。这种被正则化过后的输出地形具有较低的复杂度，从而提升了模型在那些没有被训练集覆盖的测试样本点上的泛化性能，进而导致了模型泛化误差第二次下降。在图3-9(c)中，我们可以清晰看到训练流形上的准确率在记忆噪声点后开始下降，但是非训练流形的准确率却在相同阶段持续提升。



(a) (b) (c)

图 3-9 训练流形以及非训练流形的频谱以及准确率

3.5 本章小结

过参数化的神经网络具有极大的函数空间，但是在实际训练中却并没有发生严重的过拟合现象。目前普遍认为神经网络在使用随机梯度下降算法优化更新时具有一定的学习偏好，即优先搜索某部分具有特定性质的函数空间。正是这样的学习偏好使得模型不会发生严重的过拟合，因为实际搜索的函数空间并不大。频谱偏好便是描述了这样一种学习偏好，即模型会优先拟合输出地形中的低频分量，然后再慢慢去拟合高频分量。部分研究探讨了不同频率分量的收敛速率，但是这部分探讨却仅限于特殊情形，如神经网络无限宽或者使用人造的数据集[61,62]。更重要的是，这些研究都有一个假定，那就是频谱偏好具有单调性。而我们的研究表明，学习偏好的单调性并不总是成立。

我们在这一章节中提出了一种在实际任务中切实可行的分析输出地形频谱的方法，然后在误差回合双降的实验设置下使用该方法对神经网络的频谱进行了分析。我们发现在训练初期频谱的单调性确实成立，即高频分量不断被引入到输出地形中；然而到训练后期时这种单调性被打破，高频分量开始衰退。这种非单调变化的频谱特性导致了模型泛化误差的二次下降。基于这个发现，我们提出了一种使用训练集便可以获得神经网络泛化行为的方法：通过监控神经网络在训练集上的频谱峰值，我们便可以知道神经网络泛化误差第二次下降的起点。遗憾的是，该方法并不能有效指示模型的早停点，该问题我们将在下一章解决。

通过分开计算训练流形以及非训练流形的频谱，我们回答了另一个回合双降中的问题：为什么在噪声点依然被记忆的情况下神经网络的泛化性能会再次提升？我们发现，在泛化误差第二次下降时，其泛化性能的提升主要来自于模型非训练流形上输出地形的正则化效应。换句话说，在训练流形上噪声点依然被不断引入的高频分量所记忆从而泛化性能不再提高，但是在其周围的非训练流形上输出地形会变得越来越平坦，从而起到一种隐式的正则化作用，最终使得模型在那些并没有被训练流形覆盖的测试点上取得了更好的泛化能力。

总结来看，这一章再次验证了输出地形复杂度与模型泛化能力之间的紧密联系，并且从输出地形的频谱入手分析了神经网络泛化误差的回合双降现象。在下一章节中，我们将会从偏差方差分解的角度更加详细地探讨该现象。

4 回合双降现象下的偏差与方差

本章节我们通过对零一误差（即错误率）进行偏差方差分解从而对神经网络泛化误差回合双降现象进行了分析。通过实验，我们发现神经网络泛化误差的回合双降现象主要来源于方差的二次下降，这也与上一章节中提到的输出地形复杂度非单调性变化相吻合。另外，不同于之前的研究在可导误差函数上进行分析，我们在零一误差上的分解结果表明，即使在模型误差回合双降这样复杂的情况下，误差的方差项也足以反映神经网络的泛化误差的变化趋势。基于此，我们提出了一个叫做**“优化方差”**的新指标来度量模型训练过程中引入方差的大小，该指标仅在训练集上进行计算却具有和模型泛化误差几乎一致的变化趋势，从而我们可以在不使用校验集的情况下获得模型训练过程中泛化误差的变化曲线。

4.1 偏差方差分解的意义与框架

偏差方差分解（Bias-Variance Decomposition）是一种广泛用于分析机器学习模型泛化能力的一种工具，对模型与算法的设计与理解具有着极其重要的意义[63,64]。在传统的统计学习理论中，正是偏差方差分解揭示了训练过程中模型过拟合训练集使得模型泛化能力下降的原因。随着训练回合数的增加，我们通常认为模型的偏差会不断降低，但是其方差会由于逐渐过拟合训练集中非密集采样带来的噪声而上升，从而综合起来使得模型的泛化误差呈现出 U 型曲线。这个现象也符合上一章提到的频谱偏好理论。我们知道，训练集的噪声来源于数据分布的非密集采样，因此模型在不同采样的训练集上训练时所需要记忆的噪声点是很难具有共性的，这也导致了模型过拟合各自训练集以后输出地形往往会具有较大差异。在训练前期，模型由于高频分量尚未引入而不能记忆噪声点，因此模型此时均只能去拟合普适的模式，这便导致了开始阶段模型具有较低的方差项。然而随着训练的不断进行，高频分量逐渐被引入输出地形，使得不具有共性的噪声开始被记忆，此时数据采样带来的差异使得输出地形的共性部分减弱，进而使得模型的方差增大。但是我们已经知道，回合误差双降下的频谱偏好单调性并不成立，因此我们也需要重新分析这种情况下的模型偏差与方差。

原始的偏差方差分解技术主要针对的是均方误差，后来又被逐渐推广到了交叉熵误差以及零一误差[65–68]。然而，只有均方误差以及交叉熵误差这类可导的误差具有唯一的分解形式。而零一误差这类不可导的误差函数，其分解方式具有多种且各自具有不同的特性。由于模型泛化误差回合双降现象主要发生在测试错误率（即零一误差）上，因此我们需要确定一个更加合理且规范的分解框架。在该章节中我们

使用的是 Domingos 在 2000 年提出的一种统一的偏差方差分解框架[69]。该分解框架既能够对零一误差进行分解，同时能够容纳均方误差以及交叉熵误差的情况，因此具有较强的说服力。

令 $(\boldsymbol{x}, \boldsymbol{t})$ 为从数据分布 $\mathcal{D}$ 中采样的样本点，其中 $\boldsymbol{x} \in \mathbb{R}^d$ 为 d 维的输入而 $\boldsymbol{t} \in \mathbb{R}^c$ 为 c 个分类类别中某一类别的独热编码形式（One-Hot Encoding）。接下来采样的训练集 $\mathcal{T} = \{(\boldsymbol{x}_i, \boldsymbol{t}_i)\}_{i=1}^n \sim \mathcal{D}^n$ 被用来训练模型 $f : \mathbb{R}^d \to \mathbb{R}^c$。令 $\boldsymbol{y} = f(\boldsymbol{x}; \mathcal{T}) \in \mathbb{R}^c$ 为在 $\mathcal{T}$ 上训练的模型 f 上的概率输出，而 $\mathcal{L}(\boldsymbol{t}, \boldsymbol{y})$ 为其误差函数。那么期望误差 $\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{t}, \boldsymbol{y})]$ 需要足够小才能保证模型准确地学习到训练集中的模式，并且能够较好地泛化到训练集以外的样本点上。

根据 Domingos 的论文[69]，对 $\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{y}, \boldsymbol{t})]$ 的通用分解框架写作：

$$\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{t}, \boldsymbol{y})] = \underbrace{\mathcal{L}(\boldsymbol{t}, \bar{\boldsymbol{y}})}_{\text{偏差}} + \beta \underbrace{\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\bar{\boldsymbol{y}}, \boldsymbol{y})]}_{\text{方差}}, \tag{4.1}$$

其中 β 在不同的误差函数形式下表示不同的值，而 $\bar{\boldsymbol{y}}$ 通过以下式子获得：

$$\bar{\boldsymbol{y}} = \underset{\boldsymbol{y}^* \in \mathbb{R}^c \,\middle|\, \sum_{k=1}^c \boldsymbol{y}_k^* = 1, \boldsymbol{y}_k^* \geq 0}{\arg\min} \mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{y}^*, \boldsymbol{y})]. \tag{4.2}$$

显然，$\bar{\boldsymbol{y}}$ 最小化了公式4.1中的方差项，其能够被看作为不同采样的训练集 $\mathcal{T}$ 下各个模型输出 $\boldsymbol{y}$ 的“ 中心点”或者“集成输出”。需要注意的是，在真实情况下除了偏差与方差项以外还存在着噪声项。在这里我们将 $\boldsymbol{t}$ 看做真值标签而忽略了噪声项。

为方便查看，在表格4.1中我们总结了不同误差函数 $\mathcal{L}$ 所对应的 $\bar{\boldsymbol{y}}$ 以及 β 的具体形式（详细推导过程见下一节内容）。其中，交叉熵误差为常用形式的 Kullback-Leibler 散度（KL 散度）完整形态，$Z = \sum_{k=1}^c \exp\{\mathbb{E}_{\mathcal{T}}[\log y_k]\}$ 为与 k 无关的归一化常数，$\mathrm{H}(\cdot)$ 为将向量中的最大元素变为 1 且其余变为 0 的函数，$\mathbf{1}_{\mathrm{con}}\{\cdot\}$ 为输入为真时等于 1 而输入为假时输出 0 的指示函数，$\log$ 与 $\exp$ 为对向量各元素分别进行操作的运算符。这一章中我们主要关注零一误差的偏差方差分解，因为神经网络泛化误差回合双降现象主要出现在零一误差上（具体对比详见下一节内容）。为了获得总体的偏差项与方差项，我们分析了 $\mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{t}, \boldsymbol{y})]$，即 $\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\boldsymbol{t}, \boldsymbol{y})]$ 在数据分布 $\mathcal{D}$ 上的期望值。

表 4.1 不同误差函数下的偏差方差分解

误差函数	$\mathcal{L}(\boldsymbol{t},\boldsymbol{y})$	$\bar{\boldsymbol{y}}$	β
均方根误差	$\|\boldsymbol{t}-\boldsymbol{y}\|_2^2$	$\mathbb{E}_{\mathcal{T}}\boldsymbol{y}$	1
交叉熵误差	$\sum_{k=1}^{c} t_k \log \frac{t_k}{y_k}$	$\frac{1}{Z}\exp\{\mathbb{E}_{\mathcal{T}}[\log \boldsymbol{y}]\}$	1
零一误差	$\mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\boldsymbol{y})\}$	$\mathrm{H}(\mathbb{E}_{\mathcal{T}}[\mathrm{H}(\boldsymbol{y})])$	当 $\bar{\boldsymbol{y}}=\boldsymbol{t}$ 时为 1，其余情况为 $-P_{\mathcal{T}}(\mathrm{H}(\boldsymbol{y})=\boldsymbol{t}\big\vert\bar{\boldsymbol{y}}\neq\mathrm{H}(\boldsymbol{y}))$

4.2 不同误差函数的偏差方差分解

在这一小节中我们将对均方误差、交叉熵误差以及零一误差这三种误差函数在通用的偏差方差分解框架下进行分解。与此同时，我们将会详细说明为什么实验中使用零一误差来进行偏差方差分解。

4.2.1 均方误差的分解

对于均方误差我们有 $\mathcal{L}(\boldsymbol{t},\boldsymbol{y}) = \|\boldsymbol{t}-\boldsymbol{y}\|_2^2$，现在我们首先需要根据公式4.2计算 $\bar{\boldsymbol{y}}$。我们先忽略可行域的限制从而直接考虑解决下面的优化问题：

$$\widetilde{\boldsymbol{y}} = \arg\min_{\boldsymbol{y}^*} \mathbb{E}_{\mathcal{T}}[\|\boldsymbol{y}^*-\boldsymbol{y}\|_2^2], \tag{4.3}$$

显然其解为 $\widetilde{\boldsymbol{y}} = \mathbb{E}_{\mathcal{T}}\boldsymbol{y}$。很容易验证解 $\widetilde{\boldsymbol{y}}$ 天然满足公式4.2可行域的限制，故 $\bar{\boldsymbol{y}} = \widetilde{\boldsymbol{y}} = \mathbb{E}_{\mathcal{T}}\boldsymbol{y}$.

由上，我们便可以将均方误差分解为：

$$\begin{aligned}\mathbb{E}_{\mathcal{T}}[\|\boldsymbol{t}-\boldsymbol{y}\|_2^2] &= \mathbb{E}_{\mathcal{T}}[\|\boldsymbol{t}-\bar{\boldsymbol{y}}+\bar{\boldsymbol{y}}-\boldsymbol{y}\|_2^2] \\ &= \mathbb{E}_{\mathcal{T}}[\|\boldsymbol{t}-\bar{\boldsymbol{y}}\|_2^2 + \|\bar{\boldsymbol{y}}-\boldsymbol{y}\|_2^2 + 2(\boldsymbol{t}-\bar{\boldsymbol{y}})^T(\bar{\boldsymbol{y}}-\boldsymbol{y})] \\ &= \|\boldsymbol{t}-\bar{\boldsymbol{y}}\|_2^2 + \mathbb{E}_{\mathcal{T}}[\|\bar{\boldsymbol{y}}-\boldsymbol{y}\|_2^2] + 0,\end{aligned} \tag{4.4}$$

其中第一项表示误差的偏差项，第二项表示方差项，同时根据该分解我们可以很容易知道 $\beta = 1$。

4.2.2 交叉熵误差的分解

对于交叉熵误差 $\mathcal{L}(\boldsymbol{t},\boldsymbol{y}) = \sum_{k=1}^{c} t_k \log \frac{t_k}{y_k}$ 而言，其 $\bar{\boldsymbol{y}}$ 可以通过使用拉格朗日乘子法[70] 来计算公式4.2：

$$l(\boldsymbol{y}^*,\lambda) = \mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c} y_k^* \log \frac{y_k^*}{y_k}\right] + \lambda\cdot\left(1-\sum_{k=1}^{c} y_k^*\right). \tag{4.5}$$

为了最小化 $l(\boldsymbol{y}^*, \lambda)$，我们需要计算它关于 $\boldsymbol{y}^*$ 和 λ 的偏导数：

$$\frac{\partial l}{\partial y_k^*} = \mathbb{E}_{\mathcal{T}}\left[\log\frac{y_k^*}{y_k} + 1\right] - \lambda, \quad k = 1, 2, ..., c$$
$$\frac{\partial l}{\partial \lambda} = 1 - \sum_{k=1}^{c} y_k^*.$$

通过令偏导数等于零，我们有：

$$\bar{y}_k = \frac{1}{Z}\exp\{\mathbb{E}_{\mathcal{T}}[\log y_k]\}, \quad k = 1, 2, ..., c \tag{4.6}$$

其中 $Z = \sum_{k=1}^{c}\exp\{\mathbb{E}_{\mathcal{T}}[\log y_k]\}$ 为与 k 无关的归一化常数。

由于

$$\mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c}\alpha_k\log\frac{\bar{y}_k}{y_k}\right] = -\log Z, \quad \forall_{\alpha_k}\sum_{k=1}^{c}\alpha_k = 1, \tag{4.7}$$

故我们有：

$$\begin{aligned}\mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c}t_k\log\frac{t_k}{y_k}\right] &= \mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c}t_k\left(\log\frac{t_k}{\bar{y}_k} + \log\frac{\bar{y}_k}{y_k}\right)\right] \\ &= \sum_{k=1}^{c}t_k\log\frac{t_k}{\bar{y}_k} + \mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c}t_k\log\frac{\bar{y}_k}{y_k}\right] \\ &= \sum_{k=1}^{c}t_k\log\frac{t_k}{\bar{y}_k} - \log Z \\ &= \sum_{k=1}^{c}t_k\log\frac{t_k}{\bar{y}_k} + \mathbb{E}_{\mathcal{T}}\left[\sum_{k=1}^{c}\bar{y}_k\log\frac{\bar{y}_k}{y_k}\right],\end{aligned} \tag{4.8}$$

从中我们可以知道 $\beta = 1$。

4.2.3 零一误差的分解

对于零一误差函数 $\mathcal{L}(\boldsymbol{t}, \boldsymbol{y}) = \mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\boldsymbol{y})\}$，$\bar{\boldsymbol{y}}$ 为多个模型的投票结果，即 $\mathrm{H}(\mathbb{E}_{\mathcal{T}}[\mathrm{H}(\boldsymbol{y})])$，这样才能使得方差最小。但是，$\beta$ 的值取决于 $\bar{\boldsymbol{y}}$ 与 $\boldsymbol{t}$ 之间的关系。

当满足 $\bar{\boldsymbol{y}} = \boldsymbol{t}$ 时，我们有：

$$\begin{aligned}\mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\boldsymbol{y})\}\right] &= 0 + \mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\bar{\boldsymbol{y}}) \neq \mathrm{H}(\boldsymbol{y})\}\right] \\ &= \mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\bar{\boldsymbol{y}})\} + \mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\bar{\boldsymbol{y}}) \neq \mathrm{H}(\boldsymbol{y})\}\right],\end{aligned} \tag{4.9}$$

显然此时 $\beta = 1$。

当满足 $\bar{\boldsymbol{y}} \neq \boldsymbol{t}$ 时，我们有：

$$\mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\boldsymbol{y})\}\right] = P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) \neq \boldsymbol{t}\right) = 1 - P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t}\right)$$

$$
\begin{aligned}
&= \mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\bar{\boldsymbol{y}})\} \\
&- P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) = \bar{\boldsymbol{y}}\right) P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \bar{\boldsymbol{y}}\right) \\
&- P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right) P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right). \qquad (4.10)
\end{aligned}
$$

由于 $\bar{\boldsymbol{y}} \neq \mathrm{H}(\boldsymbol{t})$，我们可以知道：

$$
P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) = \bar{\boldsymbol{y}}\right) = 0. \qquad (4.11)
$$

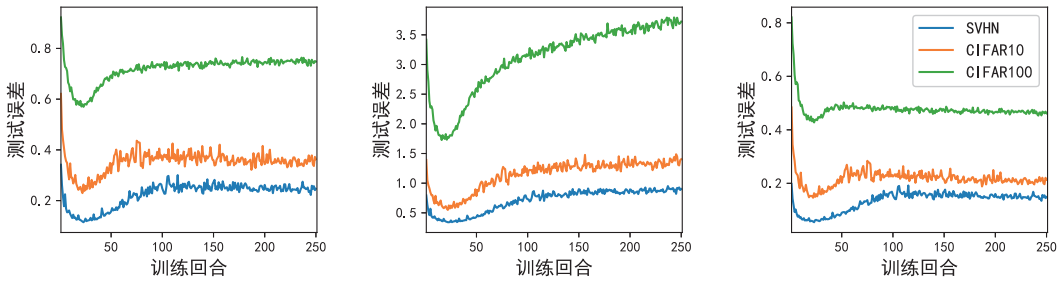
此时公式4.10变为：

$$
\begin{aligned}
\mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\boldsymbol{y})\}\right] &= \mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\bar{\boldsymbol{y}})\} - P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right) P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right) \\
&= \mathbf{1}_{\text{con}}\{\mathrm{H}(\boldsymbol{t}) \neq \mathrm{H}(\bar{\boldsymbol{y}})\} \\
&- P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right) \mathbb{E}_{\mathcal{T}}\left[\mathbf{1}_{\text{con}}\{\mathrm{H}(\bar{\boldsymbol{y}}) \neq \mathrm{H}(\boldsymbol{y})\}\right], \qquad (4.12)
\end{aligned}
$$

故我们有 $\beta = -P_{\mathcal{T}}\left(\mathrm{H}(\boldsymbol{y}) = \boldsymbol{t} \middle| \mathrm{H}(\boldsymbol{y}) \neq \bar{\boldsymbol{y}}\right)$。

4.2.4 不同误差函数对应的测试误差

多种误差函数可以用于对训练过程中的测试模型进行泛化性能度量。其中最为常用的便是上一节内容中提到的均方误差、交叉熵误差以及零一误差。但是我们需要知道的是，即使是对同样的一个模型使用不同误差函数来测试，其训练过程中的泛化误差变化趋势也并不一致。为了证明这一点，我们分别在 SVHN、CIFAR10、CIFAR100 三个数据集上训练了 ResNet18 模型，在训练过程中我们引入了 20% 的标签噪声来促成回合双降现象。学习率为 $1e-4$ 的 ADAM 优化器被用于训练模型。在每一个数据集上，我们对同样的模型使用了不同的误差函数来度量泛化性能，其结果我们展示在了图4-1 上。通过比较不同误差函数的情形我们可以发现，在测试零一误差——也就是测试错误率——的情况下神经网络泛化误差回合双降现象最为明显，而在交叉熵误差或者均方误差上则几乎没有二次下降的现象。该现象的主要原因在于，在训练后期模型的输出极化为 0 或者 1，从而使得在错误分类的样本上交叉熵误差与均方误差较大。但是需要说明的是，即使交叉熵误差与均方误差发生了变化，其决策边界本身没有发生改变。譬如，当我们将神经网络最后一层的权重乘上某一正数，此时决策边界并没有发生变化，因此对应的零一误差不发生改变，但交叉熵误差与均方误差却会随着该正数的改变而改变。综上，我们主要在零一误差上进行偏差方差分解，因为其对泛化性能的分析更为准确。



(a) 均方误差　　(b) 交叉熵误差　　(c) 零一误差

图 4-1 训练过程中的不同误差函数值

4.3 回合双降现象下的偏差与方差

上一节内容我们已经介绍了偏差方差分解的方法以及分析的误差函数，这一节内容我们将在回合双降现象出现的设置下考察训练过程中模型对应的偏差项 $\mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}[\mathcal{L}(\boldsymbol{t},\bar{\boldsymbol{y}})]$ 以及方差项 $\mathbb{E}_{\boldsymbol{x},\boldsymbol{t}}\mathbb{E}_{\mathcal{T}}[\mathcal{L}(\bar{\boldsymbol{y}},\boldsymbol{y})]$。显然，我们需要在数据分布中采样多个训练集并分别在其上训练模型，这样才能够从中估计出偏差与方差的变化。

具体而言，令 $\mathcal{T}^*$ 表示测试集，$f(\boldsymbol{x};\mathcal{T}_j,q)$ 为在训练集 $\mathcal{T}_j \sim \mathcal{D}^n$ $(j=1,2,...,K)$ 上训练 q 个回合的模型 f。那么根据上一节的计算方式我们可以知道，第 q 个回合的偏差 $B(q)$ 与方差 $V(q)$ 可以分别表示为：

$$B(q) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\in\mathcal{T}^*}\left[\mathcal{L}\left(\boldsymbol{t},\bar{f}(\boldsymbol{x};q)\right)\right], \tag{4.13}$$

$$V(q) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\in\mathcal{T}^*}\left[\frac{1}{K}\sum_{j=1}^{K}\mathcal{L}(\bar{f}(\boldsymbol{x};q), f(\boldsymbol{x};\mathcal{T}_j,q))\right], \tag{4.14}$$

其中

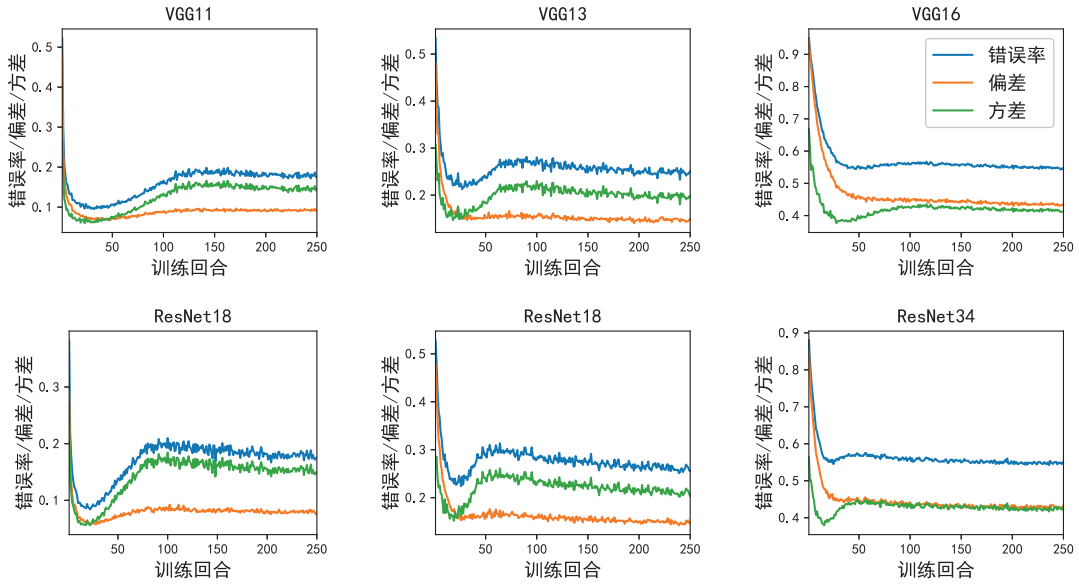
$$\bar{f}(\boldsymbol{x};q) = \mathrm{H}\left(\sum_{j=1}^{K}\mathrm{H}(f(\boldsymbol{x};\mathcal{T}_j,q))\right), \tag{4.15}$$

为 $\{f(\boldsymbol{x};\mathcal{T}_j,q)\}_{j=1}^{K}$ 的投票结果。

我们需要强调的是，在实际情形下 $\mathcal{D}$ 是未知的，故我们实验中的 $\mathcal{T}_j$ 为原始完整训练集中的部分采样，其中每个 $\mathcal{T}_j$ 为原始训练集 50% 的采样。因此，虽然我们的实验结果展示了泛化误差回合双降现象的主要因素，但是具体计算的偏差与方差并不与实际训练整个训练集时相一致。

4.3.1 方差项的二次下降

实验中我们考虑 ResNet 与 VGG 两种架构类型，数据集为 SVHN，CIFAR10 以及 CIFAR100。我们对不同学习率的 SGD 优化器与 ADAM 优化器的训练过程均进行



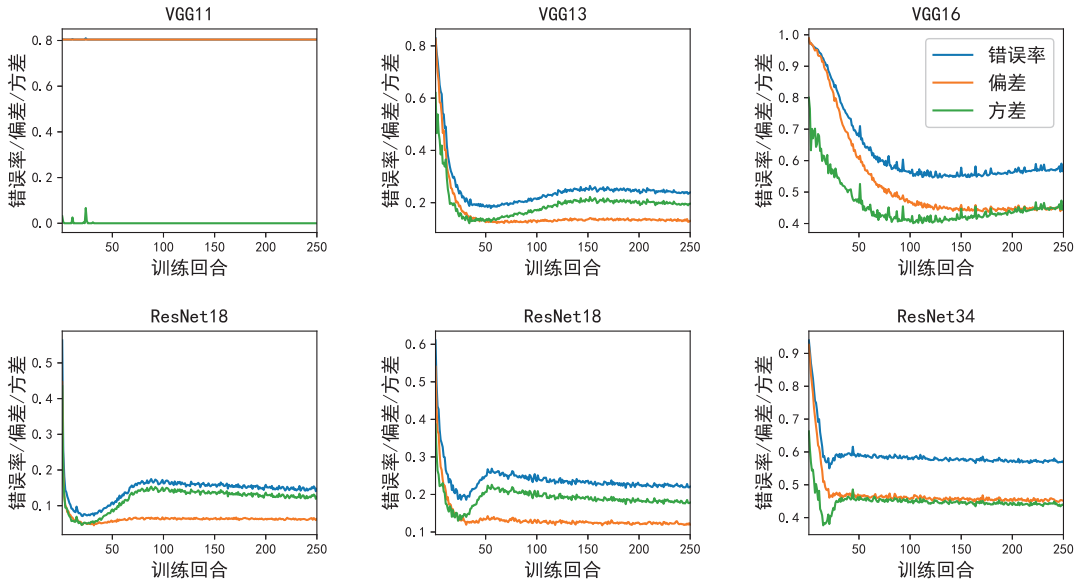
(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-2 使用学习率为 **1e-4** 的 **ADAM** 优化器训练时的零一方差及其对应的偏差与方差

了实验。训练集批次大小设置为 128，并且所有对模型都在数据增强的情况下训练了 250 个训练回合。在从完整的训练集中采集 $\{\mathcal{T}_j\}_{j=1}^{K}$ （$K=5$）之前，我们随机扰动了训练集中 20% 样本点的标签从而引入泛化误差回合双降现象。

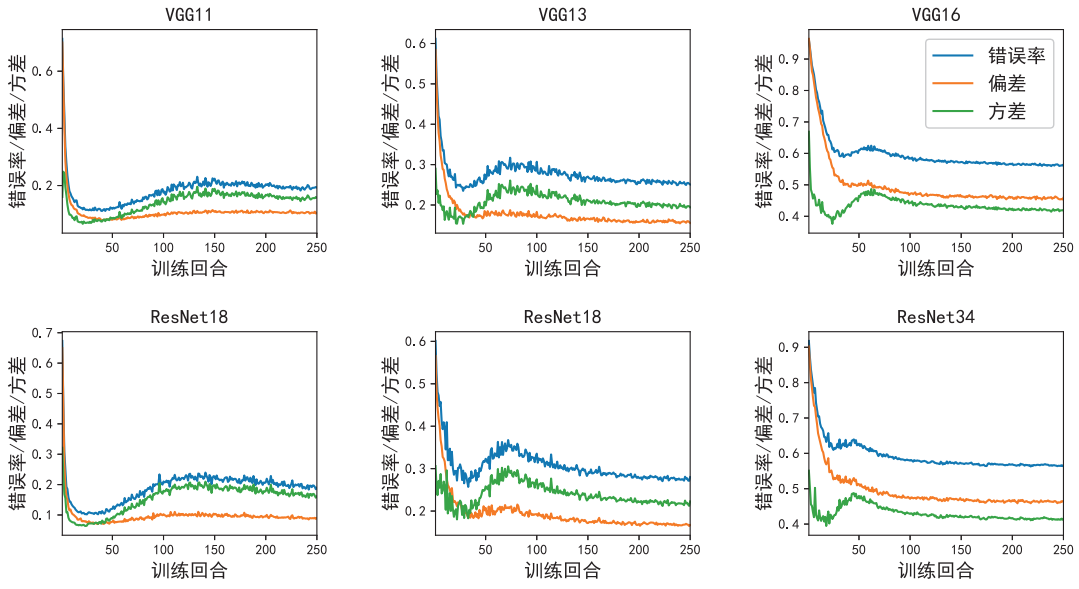
图4-2、4-3、4-4 以及4-5分别展示了在不同优化设置下的模型在测试集上的零一误差以及对应的偏差项与方差项。几乎在所有的图中我们都可以看到，在训练过程中偏差快速下降然后收敛到一个较低的值，但是方差却表现得大不相同。从图中我们可以看到，方差项在训练过程中的变化趋势与测试误差几乎一模一样，甚至连一些局部的波动都基本保持一致。这些实验都说明了，在零一误差的情况下，方差项为影响模型泛化误差变化的主体。

从实验结果中我们也可以发现，方差项不再如传统统计学习上所认为的那样随着训练不断增加，而是具有着更加复杂的变化。零一误差的方差项从一个较大的值开始快速衰减，紧接着开始上升，随后却再次开始下降。初始阶段与传统统计学习中不符合的原因主要在于分析误差函数的不同。统计学习中常常分析交叉熵误差与均方误差这类能够反映输出概率不同程度的误差函数，而零一误差只能反映标签的不同。在训练的初始阶段，输出的概率全部接近于随机，故此时输出概率上很小的不同就可能导致完全不同的标签，从而使其具有与其他误差函数所不同的初始情况。在训练后期，由于输出地形高频分量的逐渐消失，模型之间由于过拟合噪声所导致的输出差异减小，从而使得方差下降。由于方差项的降低，模型的泛化误差也随着减少，从而出现了神经网络泛化误差回合双降的现象。



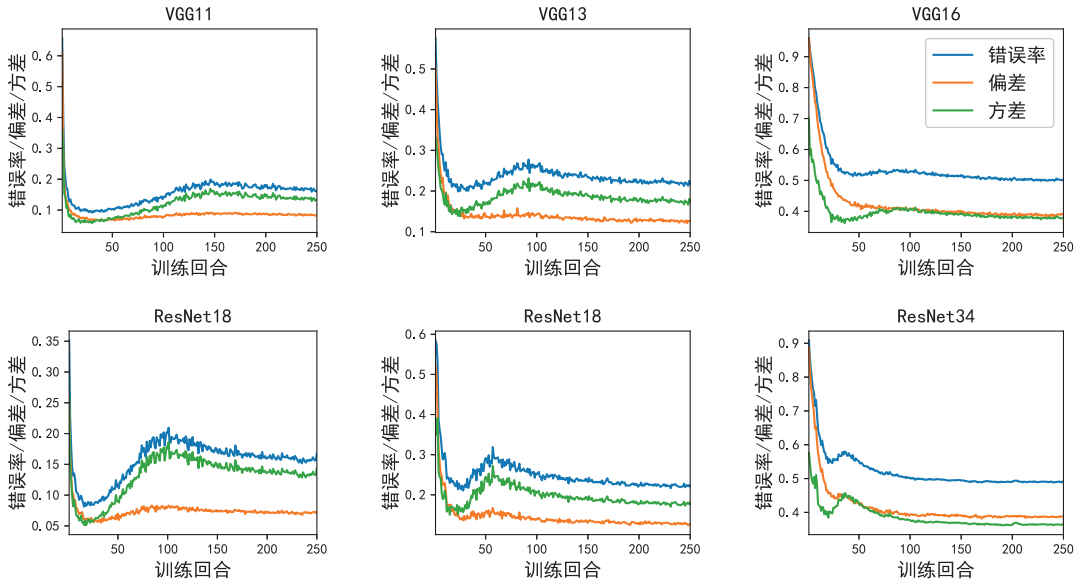
(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-3 使用使用学习率为 **1e-3** 的 **ADAM** 优化器训练时的零一方差及其对应的偏差与方差



(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-4 使用学习率为 **1e-3** 的 **SGD** 优化器训练的零一方差及其对应的偏差与方差



(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-5 使用学习率为 **1e-2** 的 **SGD** 优化器训练时的零一方差及其对应的偏差与方差

4.3.2 不同噪声水平下的方差与偏差

上一节的实验已经说明，零一误差与其方差项高度一致，且正是由于其方差项在后期的衰减导致了泛化误差的二次下降。已知较高水平的标签噪声可以使得模型泛化误差回合双降现象更加明显与清晰[19]，接下来我们将验证在不同噪声水平下泛化误差与方差的同步性。

该实验在 CIFAR10 上的 ResNet18 上进行，且使用学习率为 $1e-4$ 的 ADAM 优化器进行训练。我们调节标签噪声的占比从 0% 到 40% 变化，然后观察在测试集上对应的零一误差、方差项与偏差项。图4-6 展示了训练过程中这三者在不同噪声水平下的情况。我们可以看到，虽然标签噪声同时影响了偏差项与方差项，但是后者显然更加敏感且显示出与误差更好的同步性。例如，当我们随机扰动一小部分比例——如 10%——标签时，方差项在 20 回合到 50 回合之间出现了一个清晰的低谷，但是偏差项上却没有非常明显。另外一个有趣的发现是，标签噪声的占比似乎并不影响误差到达其极小值时的回合数。这是一个出乎意料的现象，因为通常而言噪声被认为与数据集的复杂度高度相关。我们将在未来的研究中对如何评测数据集复杂度这个问题进行更加深入的探讨。

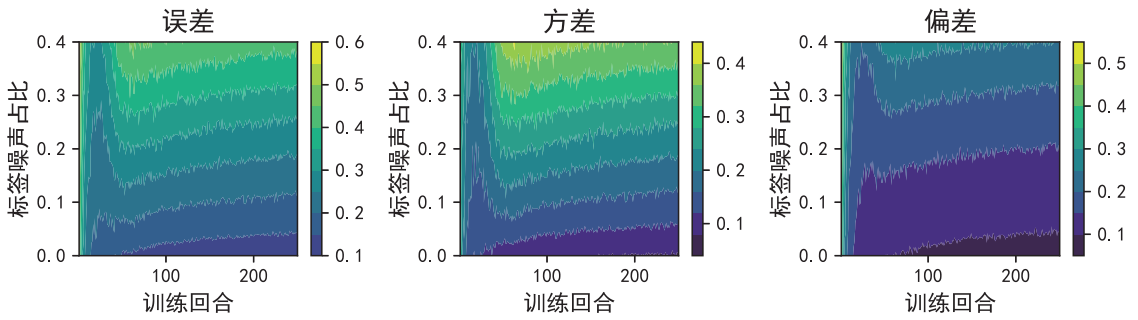


图 4-6 训练过程中不同噪声水平下的误差、方差与偏差热力图

4.4 基于非校验集信息的泛化误差曲线预测

上一节内容我们已经验证了测试错误率与其方差之间的同步性，但是该方法除了分析模型泛化性能变化以外其应用场景极其有限。在估计方差的过程中，我们需要用到测试集，同时需要在同一个数据分布上采样多个训练集来训练多个模型。然而现实情况下我们没有办法获得测试集，更没有办法获得数据集的完整分布。故此，我们希望能够在单一训练集上估计出类似方差的量，这样就可以在不使用校验集的情况下直接预测获取模型的泛化性能变化趋势。这便是这一节内容中我们提出的**“优化方差”**。

事实上，目前已有部分论文提出了一些复杂度度量来预测神经网络的泛化能力，比如解在参数空间的平坦度[71] 以及一些基于参数大小的指标[72]。但是，这些指标都非常依赖于模型的参数与结构，使得跨模型的比较非常困难。Dinh 等在 2017 年指出，当我们重参数化模型时，神经网络解的平坦度可以任意变化但是却不会影响神经网络所表示的函数[73]；Neyshabur 等在 2018 年指出，这些指标不能够解释神经网络不断变宽情况下泛化性能的变化[74]。Chatterji 于 2020 年提出了一种叫做模型临界度（Model Criticality）的指标，该指标能够反映出跨模型的泛化性能的变化[75]，但遗憾的是该指标并不能在训练过程中反映模型的泛化性能变化，特别是回合双降这种复杂变化的情况。

我们提出的优化方差能够只在训练集上计算便可以反映模型训练过程中的泛化误差变化趋势。即使是在模型泛化误差回合双降的复杂变化下，我们的指标依然能够保有较好的一致性。同时，其计算只需要使用神经网络的 logits 输出，因此与其架构或参数依赖较小，从而能够解释更多的泛化现象，比如模型泛化能力随着其宽度增加的变化。我们将在本章后续部分详细阐述。

4.4.1 度量模型泛化能力的新指标：优化方差

根据公式4.1中的定义我们可以知道，方差项度量了同一数据分布中训练集的不同采样所带来的模型多样性。换句话说，其度量了模型对某一样本点的输出由于训练集的随机性而发生变化的程度。我们知道，梯度是训练集在优化过程中传递给模型的唯一信息，因此我们可以去度量在同一回合不同训练批次梯度所引入的方差情况。更具体来讲，我们可以使用某个指标来反映神经网络对采样噪声的鲁棒性。如果神经网络所表示的模型会随着不同训练批次而发生剧烈的变化，那么它的泛化误差也极有可能因为优化过程引入的较大方差而变得较差。一个与之相似的指标为解的平坦度[71]，但是由于该指标只能在局部极小点计算故不能够在整个优化过程中使用。

数学上来讲，对于某一样本点 $(\boldsymbol{x}, \boldsymbol{t}) \sim \mathcal{D}$，令 $f(\boldsymbol{x}; \boldsymbol{\theta})$ 表示神经网络在参数 $\boldsymbol{\theta}$ 下的 logits 输出。令 $\mathcal{T}_B \sim \mathcal{D}^m$ 表示样本数为 m 的训练批次，$g : \mathcal{T}_B \to \mathbb{R}^{|\boldsymbol{\theta}|}$ 为优化器根据 $\mathcal{T}_B$ 计算出来的 $\boldsymbol{\theta}$ 的更新量。那么我们可以获得一个以 $\mathcal{T}_B$ 为自变量的函数分布 $F_{\boldsymbol{x}}(\mathcal{T}_B)$，即 $f(\boldsymbol{x}; \boldsymbol{\theta} + g(\mathcal{T}_B)) \sim F_{\boldsymbol{x}}(\mathcal{T}_B)$。从这个角度来看，$F_{\boldsymbol{x}}(\mathcal{T}_B)$ 分布的方差反映了训练批次改变所带来的模型多样性。基于该角度，我们在下列给出优化方差的具体定义式。

定义 4.1 (**优化方差**（Optimization Variance，OV）)： 给定一个输入 $\boldsymbol{x}$ 以及模型在第 q 训练回合的参数 $\boldsymbol{\theta}_q$，那么在该回合针对 $\boldsymbol{x}$ 的优化方差定义为：

$$OV_q(\boldsymbol{x}) \triangleq \frac{\mathbb{E}_{\mathcal{T}_B}\left[\left\|f(\boldsymbol{x}; \boldsymbol{\theta}_q + g(\mathcal{T}_B)) - \mathbb{E}_{\mathcal{T}_B} f(\boldsymbol{x}; \boldsymbol{\theta}_q + g(\mathcal{T}_B))\right\|_2^2\right]}{\mathbb{E}_{\mathcal{T}_B}\left[\left\|f(\boldsymbol{x}; \boldsymbol{\theta}_q + g(\mathcal{T}_B))\right\|_2^2\right]}. \tag{4.16}$$

需要注意的是，在公式4.16的分母消除了优化过程中 logits 值大小本身带来的影响下，$OV_q(\boldsymbol{x})$ 度量的是方差的相对变化值。这样一来，优化的不同阶段下 $OV_q(\boldsymbol{x})$ 就具有了可比性。这里的动机来自于统计理论中的变异系数①，也被叫做相对标准差。变异系数被定义为标准差与均值的比值，故与模型的量纲无关，因此变异系数能够比较具有不同量纲的两个量的分散程度。

回到优化方差，我们知道 logits 输出的方差——即优化方差定义式的分子——由于 logits 大小的变化而不能在优化过程的不同阶段进行比较。事实上，即使 logits 的方差在整个优化过程中保持相同，在 logits 本身值较大时其对决策边界的影响也会极其有限。因此，通过将 logits 大小视作量纲，根据变异系数的定义我们将优化方差定义为 $\sum_i \sigma_i^2 / \sum_i \mu_i^2$ 的形式，其中 σ_i 与 μ_i 分别表示了 logits 第 i 项的标准差与均值。如果我们去掉了分母项，那么优化方差的值便不再能够指示泛化误差的变化趋势，尤其是在优化过程的早期阶段。

① https://en.wikipedia.org/wiki/Coefficient_of_variation

直观上来讲，优化方差代表了梯度对模型影响的非一致性。如果 $OV_q(\boldsymbol{x})$ 非常大，那么不同的采样批次 $\mathcal{T}_B$ 便会使得模型对相同输入具有完全不同的的输出，从而导致较高的模型多样性与方差。注意这里我们强调的是梯度对模型影响的非一致性而不是梯度本身的非一致性。后者可以被梯度方差度量，但其与模型函数的方差存在着不同，比如某些情况下不同的梯度会导致模型函数变化相同。但与此同时，这两者之间也具有着密切的联系，下面我们将详细阐述。

简洁起见，我们省略 $OV_q(\boldsymbol{x})$ 中的脚标 q 并且用 $\tilde{g}(\mathcal{T}_B)$ 来表示 $g(\mathcal{T}_B)-\mathbb{E}_{\mathcal{T}_B}g(\mathcal{T}_B)$。那么梯度的方差 V_g 可以写作：

$$V_g = \mathbb{E}_{\mathcal{T}_B}\left[\|g(\mathcal{T}_B)-\mathbb{E}_{\mathcal{T}_B}g(\mathcal{T}_B)\|_2^2\right] = \mathbb{E}_{\mathcal{T}_B}\left[\tilde{g}(\mathcal{T}_B)^T\tilde{g}(\mathcal{T}_B)\right]. \tag{4.17}$$

令 $\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})$ 表示模型的 logits 输出 $f(\boldsymbol{x};\boldsymbol{\theta})$ 关于参数 $\boldsymbol{\theta}$ 的雅可比矩阵（Jacobian Matrix）：

$$\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \left[\nabla_{\boldsymbol{\theta}}f_1(\boldsymbol{x};\boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}}f_2(\boldsymbol{x};\boldsymbol{\theta}), ..., \nabla_{\boldsymbol{\theta}}f_c(\boldsymbol{x};\boldsymbol{\theta})\right], \tag{4.18}$$

其中 $f_j(\boldsymbol{x};\boldsymbol{\theta})$ 表示 $f(\boldsymbol{x};\boldsymbol{\theta})$ 的第 j 项，而 c 为类别数量。

使用一阶近似我们有：

$$f(\boldsymbol{x};\boldsymbol{\theta}+g(\mathcal{T}_B)) \approx f(\boldsymbol{x};\boldsymbol{\theta}) + \boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})^T g(\mathcal{T}_B), \tag{4.19}$$

而 $OV(\boldsymbol{x})$ 可以写作：

$$OV(\boldsymbol{x}) \approx \frac{\mathbb{E}_{\mathcal{T}_B}\left[\tilde{g}(\mathcal{T}_B)^T\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})^T\tilde{g}(\mathcal{T}_B)\right]}{f(\boldsymbol{x};\boldsymbol{\theta})^T f(\boldsymbol{x};\boldsymbol{\theta}) + \mathbb{E}_{\mathcal{T}_B}\left[O\left(\|g(\mathcal{T}_B)\|_2\right)\right]} \approx \frac{\mathbb{E}_{\mathcal{T}_B}\left[\tilde{g}(\mathcal{T}_B)^T\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})^T\tilde{g}(\mathcal{T}_B)\right]}{f(\boldsymbol{x};\boldsymbol{\theta})^T f(\boldsymbol{x};\boldsymbol{\theta})}. \tag{4.20}$$

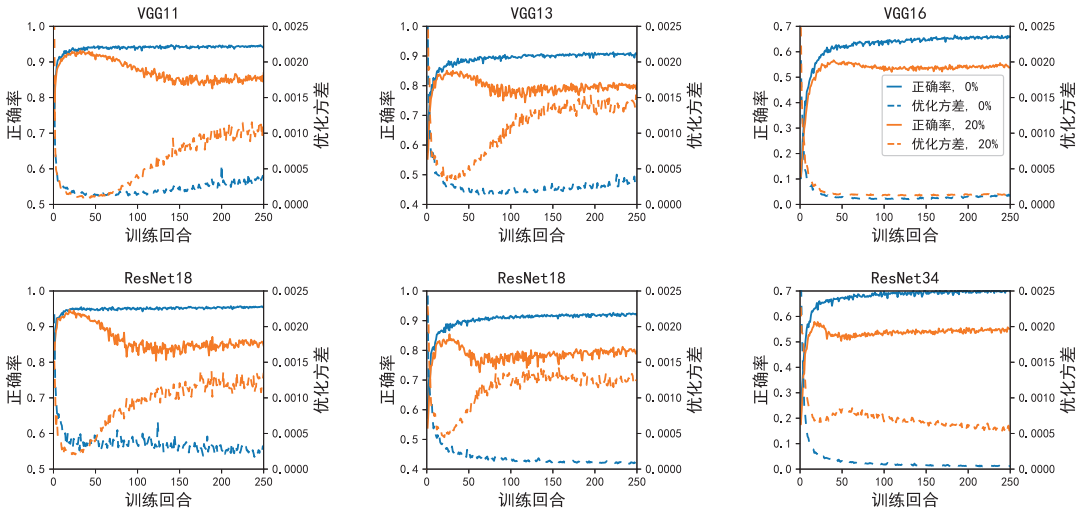
我们可以发现 $\mathbb{E}_{\boldsymbol{x}}[OV(x)]$ 与 V_g 之间唯一的区别在于中间的权重矩阵 $\mathbb{E}_{\boldsymbol{x}}\left[\frac{\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x})^T}{f(\boldsymbol{x};\boldsymbol{\theta})^T f(\boldsymbol{x};\boldsymbol{\theta})}\right]$。这意味着惩罚梯度方差也能够降低优化方差。

4.4.2 基于优化方差的泛化误差曲线预测

上一节内容中我们提出了一个新的叫做优化方差的指标，这一节我们将根据其定义检验在实际实验中其与泛化误差变化的一致性。

4.4.2.1 训练过程中的优化方差与测试准确率

我们遵循本章前面所述的实验设置，在 SVHN、CIFAR10、CIFAR100 三个数据集上检验了 VGG 与 ResNet 两类架构。我们在训练过程中的每一个回合计算了优化方差在训练样本 $\boldsymbol{x}$ 上的期望，即 $\mathbb{E}_{\boldsymbol{x}}[OV_q(\boldsymbol{x})]$，其值使用训练集中随机采样的 1000 个样本点来估计。需要注意的是，其计算的每一个步骤都完全没有使用测试集。



(a) SVHN (b) CIFAR10 (c) CIFAR100

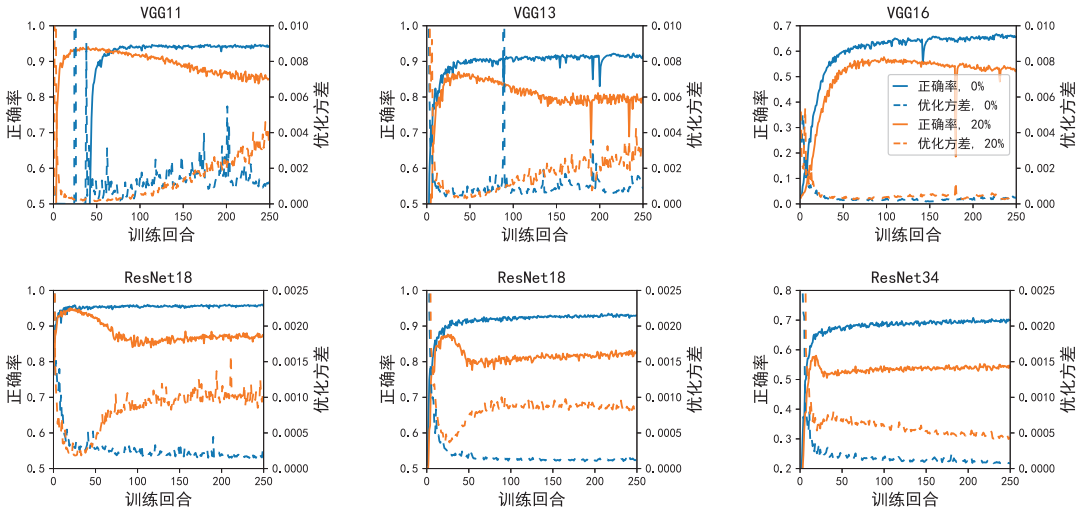
图 4-7 使用学习率为 **1e-4** 的 **ADAM** 优化器训练时的测试正确率以及优化方差

图4-7、4-8、4-9 以及4-10展示了不同学习率的不同优化器在不同标签噪声下的优化方差与测试准确率，其中图例中的数字表示标签噪声的占比。其中实线展示的为测试准确率，虚线展示的为 $\mathbb{E}_{\boldsymbol{x}}[OV_q(\boldsymbol{x})]$，不同颜色表示不同的标签噪声占比。虽然某些情况下优化方差并没有呈现出严格的回合双降现象，例如图4-8 中 CIFAR100 上的 VGG16 模型，但是我们可以普遍观测到实现与曲线之间的对称性，这意味着优化方差虽然能够从训练集中直接计算，却能够较好地预测测试准确率的变化趋势。另外我们可以发现，当模型添加噪声后泛化误差上升，而此时优化方差本身也会随之增加，这同样验证了优化方差较好的泛化能力指示能力。

这里需要注意的是模型回合双降现象并不是使用优化方差的必要条件，而是为了验证在训练过程中即使模型泛化能力变化较为复杂，优化方差也依然能够捕捉到其变化趋势。正如上面图片中所示，我们比较了没有噪声情况下的优化方差与测试准确率，当泛化误差回合双降现象没有出现时，我们提出的优化方差也能够较好地预测。

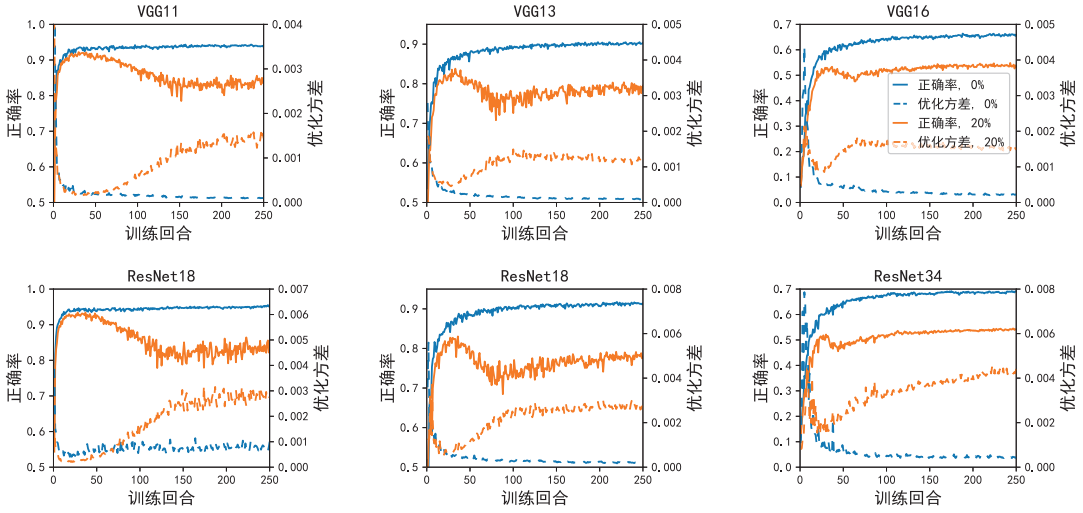
上述实验中的 $OV_q(\boldsymbol{x})$ 使用了所有的训练批次来进行计算，但是这可能并不是必要的。事实上，即使只使用一小部分训练批次，我们通常也能够较为准确的估计优化方差。为了验证这一点，我们使用了不同数量的训练批次来估计 $OV_q(\boldsymbol{x})$，其结果显示在图4-11 中（训练集的标签噪声占比为 20%）。我们可以发现，某些情况即使只使用 10 个训练批次的数据也可以较好地估计优化方差的值。

另一个有趣的发现是即使是测试准确率上一些不规则的波动，优化方差也能够较好地反映出。这种联系在一些简单的数据集上最为明显，比如 MNIST[51] 与 FashionMNIST[76] 数据集。我们在这两个数据集上使用了学习率为 $1e-4$ 的 ADAM



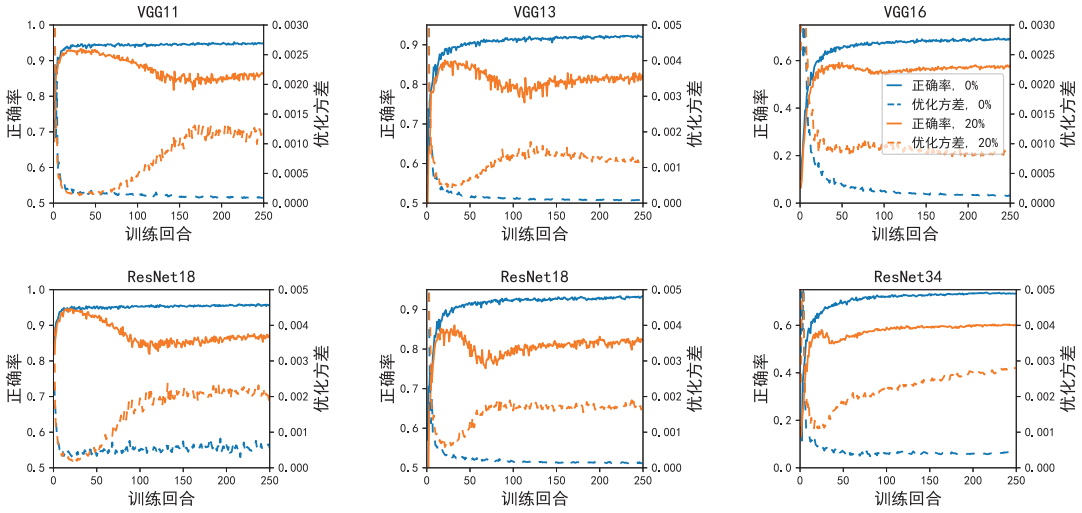
(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-8 使用学习率为 **1e-3** 的 **ADAM** 优化器训练时的测试正确率以及优化方差



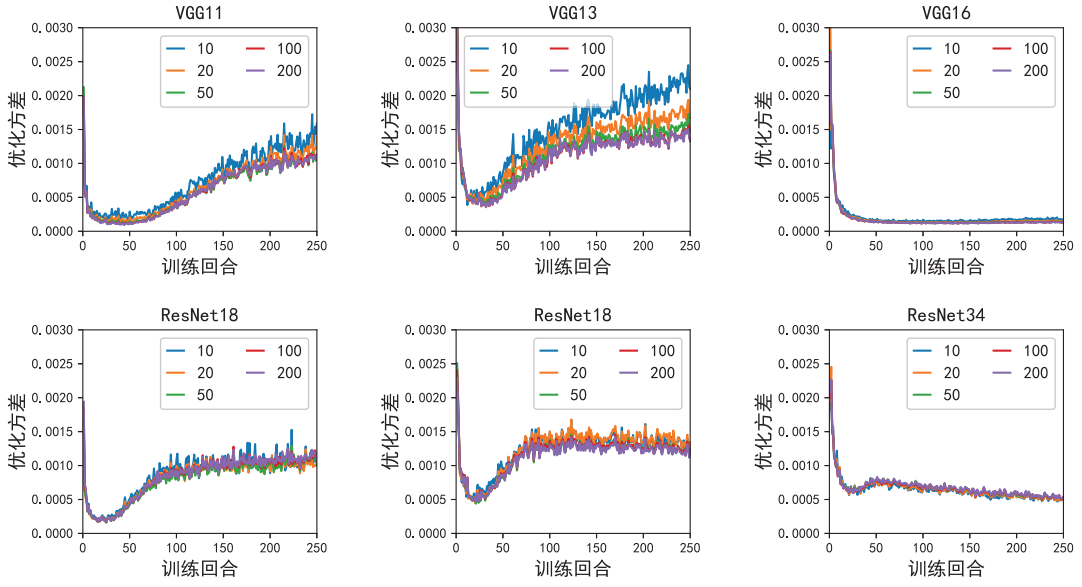
(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-9 使用学习率为 **1e-3** 的 **SGD** 优化器训练时的测试正确率以及优化方差



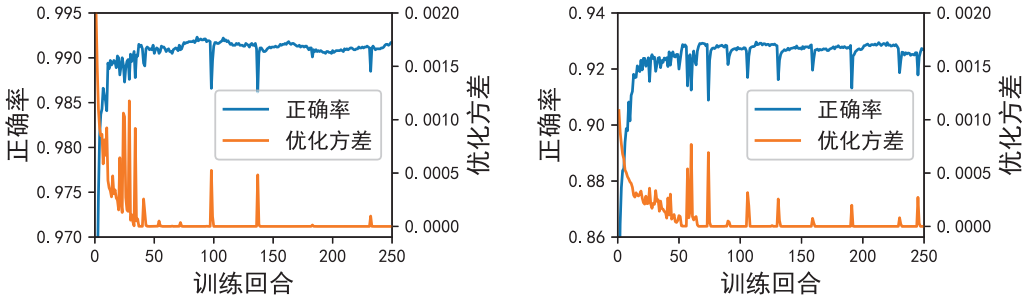
(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-10 使用学习率为 **1e-2** 的 **SGD** 优化器训练时的测试正确率以及优化方差



(a) SVHN (b) CIFAR10 (c) CIFAR100

图 4-11 使用学习率为 $1e-4$ 的 ADAM 优化器训练时，不同数量训练批次估计出的优化方差



(a) MNIST (b) FashionMNIST

图 4-12 MNIST 与 FashionMNIST 数据集上 LeNet-5 的测试准确率与优化方差

优化器训练了模型 LeNet-5[51]，图4-12 展示了其结果。我们可以看到，优化方差的尖峰与测试准确率的一些尖峰发生在同一个回合，因此具有极好的同步性。

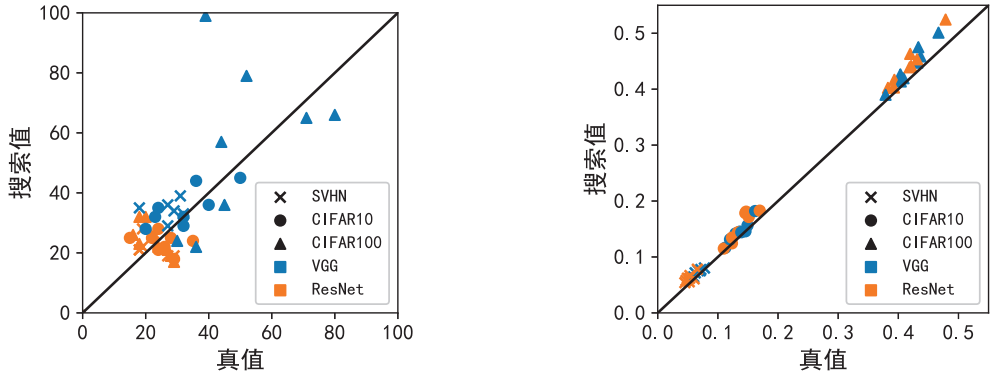
综上，我们的实验结果证明了模型在训练过程中的泛化能力可以被我们提出的优化方差预测。这样我们就可以在不使用校验集的情况下预测早停点以及其他一些泛化现象。

4.4.2.2 不使用校验集的早停技巧

训练神经网络的步骤一般包含三步：1）将大训练集划分为小训练集与校验集；2）使用小训练集来训练模型，并且在校验集上确定并记录早停点；3）在大训练集上训练同样的回合数获得更好的泛化性能。然而，这样并不能保证在大训练集上的早停点与小训练集上的相同，因此一个有趣的问题是：我们能否直接在大训练集上训练并且不使用校验集便找到早停点？根据上一节的实验我们可以知道，优化方差由于其与泛化误差的一致性可以用作该用途。

这里为了更加鲁棒的结果，我们使用优化方差在训练过程中的平滑值而非优化方差本身。具体而言，我们使用一个窗口为 10 个回合的窗口来对优化方差进行滑动均值滤波，然后在平滑后的优化方差上使用忍耐回合数（Patience）为 10 的早停方法找到其极小值。作为参照，我们也使用相同的早停方法直接在测试错误率上找到最小值并以此为理想真值情况。但是需要注意的是真值情况在实际情况中并不能获得，这里只是出于一种验证目的来作为比较。

同样地，我们在不同标签噪声水平下（10% 和 20%）的多个数据集上训练了多个模型（SVHN：VGG11 和 ResNet18；CIFAR10：VGG13 和 ResNet18；CIFAR100：VGG16 和 ResNet34），同时我们使用了不同学习率下的多种优化器来验证情况（学习率为 $1e-3$ 与 $1e-4$ 的 ADAM 优化器以及学习率为 $1e-2$ 与 $1e-3$、动量项为 0.9 的 SGD 优化器）。接下来，我们比较了真值早停点、真值测试错误率与使用优化



(a) 最优早停回合　　(b) 最优测试错误率

图 4-13 基于测试误差真值以及优化方差搜索的早停方法的对比

方差搜索到的早停点、测试错误率之间的大小①。实验结果我们展示在了图4-13，其中不同的形状表示不同的数据集，不同颜色表示不同的模型。黑色直线指示了搜索值与真值相等的情况。我们可以发现真值早停点与搜索到的早停点相对较近，但是也存在着一些例外，比如图4-13(a) 位于（40，100）坐标的点。在实际情况中，模型泛化误差较低时会存在这波动，导致早停点也会在其上前后有较大的波动，但是这种波动引起的泛化误差的波动却不会太大，而泛化误差才是我们真正关心的值。从图4-13(b) 中我们可以看到，测试错误率与真值错误率总是非常相近，这证明了使用优化方差来作为早停指标的可行性。

4.4.2.3 其余情况下的泛化性能预测

训练集较小情况下的预测：对于较大训练集而言，校验集能够很好的划分出来且一般不容易伤害模型的泛化性能，但是在较小数据下校验集却并不容易合理的划分出来。优化方差因其能够直接在训练集上获得且能较好指示出模型的泛化能力，故此对较难划分出校验集的小训练集而言有较重要的意义。从这个角度出发，我们验证了优化方差在较小数据量情况下的指示效果。我们从 CIFAR10 上分别随机抽取了 2000、4000 与 6000 个训练样本来验证优化方差在小训练集下的有效性。考虑到只有较少的训练样本，这里我们使用较小的一个卷积神经网络，并使用学习率为 $1e-4$ 的 ADAM 优化器来训练模型。该卷积神经网络的具体架构细节详见表4.2。

实验的结果我们展示在图4-14中，其中图例中的数字代表模型使用的训练样本数量。我们可以观察到两个现象：1）当训练样本数量较小时，优化方差与模型的泛化能力依然在训练过程中有着较好的一致性，这验证了我们提出的指标在小数据集

① 在 SVHN 上训练的 VGG11 在学习率为 $1e-3$ 的 ADAM 优化器情形下无法正常训练，所以我们没有将其纳入实验结果。

表 4.2 较小卷积神经网络的架构细节

网络层	参数	BN	激活函数	最大池化
输入	输入尺寸=(32, 32)×3	-	-	-
卷积层	滤波器=(3, 3)×32	✓	ReLU	(2, 2)
卷积层	滤波器=(3, 3)×64	✓	ReLU	(2, 2)
全连接层	神经元数量=1024	-	ReLU	-
全连接层	神经元数量=10	-	Softmax	-

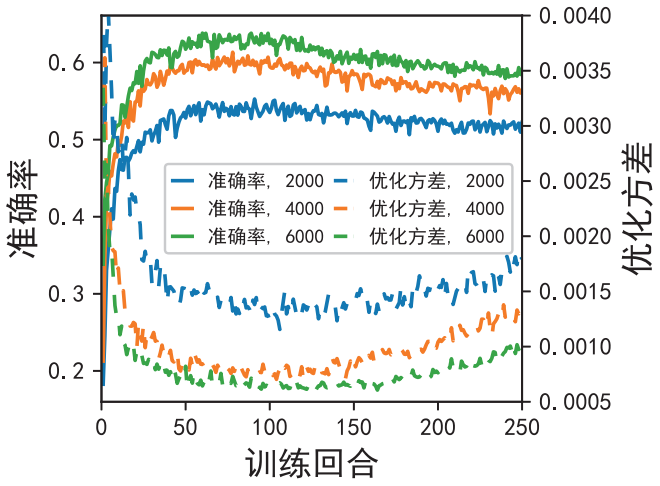


图 4-14 使用较少训练样本时模型的测试准确率与优化方差

上的效果；2）另一个符合预期的现象是，当更多的训练样本被使用时，模型的泛化性能有较大的提升，在这时优化方差也体现出了同样的特性。该现象很好地说明了优化方差对模型泛化性能的指示能力。

模型宽度变化下的预测：除了指示模型在训练过程中的泛化性能变化，优化方差也能够解释一些其他的跨模型泛化现象，比如模型宽度对泛化能力的影响。为了验证这一点，我们使用不同宽度的 ResNet18 在 CIFAR10 上通过学习率为 $1e-4$ 的 ADAM 优化器训练 100 个回合，然后比较其测试准确率与优化方差的变化。对于每层卷积层，我们将其滤波器数量设置为 $k/4$ 倍于原始模型的滤波器数量，其中 $k=1,2,...,8$。然后我们验证优化方差与测试准确率之间的相关度。注意为了减少不同模型训练过程的影响，在计算优化方差时我们统一使用学习率为 $1e-3$ 且不包含动量的 SGD 优化器。这样一来，不同模型之间的比较更加公平。

图4-15中我们展示了实验的结果。当 k 增加时，优化方差也会逐渐下降，换句话说采样噪声引入的模型多样性随着模型宽度的增加而降低，这意味着增加模型宽

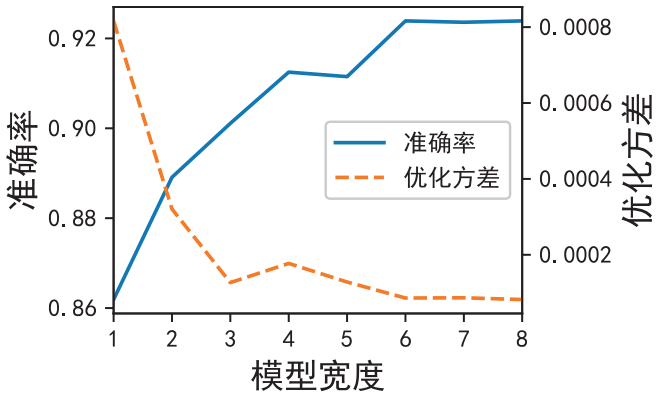


图 4-15 不同模型宽度下的测试准确率与优化方差

度能够较好地提升模型对采样噪声的鲁棒性从而带来更好的泛化性能。图中优化方差与测试准确率之间的皮尔逊相关系数（Pearson Correlation Coefficient）为 -0.94，且 p 值为 0.0006。该实验说明优化方差对跨模型的泛化性能变化同样具有一定的指示能力。

4.5 本章小结

神经网络过拟合训练集噪声以后所具有的复杂输出地形会带来模型方差的上升，因此研究神经网络泛化误差回合双降现象下的方差与偏差极为重要。传统的统计学习理论指出，随着训练的不断进行，模型的偏差会不断下降，而方差会逐渐上升，从而使得最后的泛化误差出现先下降后上升的 U 型曲线。然而，这种情况显然与神经网络的回合双降现象相违背，因此我们在该现象下重新分析了模型的偏差与方差。

与之前研究不同的是，由于误差回合双降现象一般出现在使用错误率作为泛化度量的情形下，因此我们偏差方差分解的对象不再是统计学习中常分解的均方误差或者交叉熵误差这类可导的误差函数，而是通过一个通用的分解框架对零一误差——即错误率——进行了分解。通过在多个数据集、多个模型框架、不同占比的噪声以及多种优化策略下进行实验，我们发现零一误差的方差项主导了模型泛化误差的变化。换句话说，正是由于方差的回合双降导致了零一误差的回合双降。这个发现既说明了神经网络泛化误差回合双降的原因，又证明了传统统计学习中方差不断增加的假设并不总是成立。

由于方差与模型测试误差的高度一致性，我们期望能够在训练集上计算一个类似的指标来度量优化过程中逐渐引入的方差项，这样我们就可以在不使用校验集的情况下获取模型泛化误差的变化趋势。基于此，我们提出了优化方差这个新的指标，

该指标度量了优化过程中随机采样的不同训练批次对模型表示的函数带来的影响。通过大量的实验，我们发现该指标虽然是在训练集上进行计算，却能够在优化过程中与模型泛化误差具有相同的变化趋势。即使是在回合双降现象这样复杂的泛化误差变化情况下，优化方差依然保有较好的预测泛化误差曲线的能力。在这个发现的基础上，我们进一步提出了不需要校验集的早停方法，从而使得模型能够直接在完整训练集上进行训练而不用担心过拟合。从这一点出发，优化方差尤其适合在训练集较小这种难以划分校验集的情况下使用。除此之外我们还发现，优化方差不仅能够在同一个模型的优化过程中指示模型的泛化性能，其还具有一定的跨模型能力。当模型宽度增加时，优化方差也能够较好地反映模型泛化性能的变化。

最后我们需要指出的是，虽然优化方差在模型宽度变化时能够作为跨模型的泛化性能指示指标，但是我们并没有在模型架构发生非常大改变时观察到优化方差与测试准确率之间的显著联系。比如，优化方差的大小并不能用来有效地比较 VGG 与 ResNet 两种架构之间的泛化性能大小。我们未来的研究会寻找具有更强跨模型通用性的指标来指示模型的泛化性能。

5 总结与展望

5.1 总结

为什么过参数化的神经网络模型复杂度大到能够拟合随机噪声，但是在正常训练集上却依然保有较好的泛化能力? 神经网络的泛化性能难以被传统机器学习理论所解释，也因此成为了研究的难点与热点。一些研究认为这是因为神经网络的优化过程具有学习偏好，即其会由简单到复杂地搜索模型所在的函数空间。学习偏好使得神经网络优化过程中实际的模型复杂度较小，从而具有较好的泛化性能。然而，这些研究的基本假设都建立在神经网络的学习偏好单调性的基础上，而这与近来发现的神经网络泛化误差二次下降现象相违背。

本文旨在对神经网络地形复杂度以及误差分解的偏差方差进行分析，从而能够对泛化误差回合双降现象进行解释与说明，进而促进神经网络泛化性能的研究。我们首先利用分段线性神经网络片状输出地形的几何特性，使用凸优化的方式对其切分粒度以及泛化边界进行了分析。我们实验表明，不同优化技巧能够显著改变输出地形的切分粒度从而对神经网络泛化能力进行影响。在这一部分，我们论证了神经网络输出地形与其泛化能力之间的紧密联系，从而为后续部分使用输出地形频谱来解释泛化误差回合双降现象垫定了基础。

基于本文提出的一种新的计算输出地形频谱的方法，我们通过实验反驳了之前研究假定的神经网络在训练过程中所具有的频谱偏好这种学习偏好的单调性。通过分解流形区域上的输出地形以及非流形区域的输出地形，我们发现学习偏好的单调性只存在于流形区域输出地形拟合噪声的过程中，而这一步并不会带来泛化性能的进一步提升；与之相反的是，非流形区域的输出地形高频分量不断下降，使得其上的输出地形逐渐变得平坦，从而令模型在部分没有被训练流形覆盖的测试点上的泛化误差显著下降，进而导致了模型泛化误差回合双降的现象。从总体上来看，我们也可以看到频谱的高频成分呈现出先上升后下降的趋势，且峰值与泛化误差第二次下降的起点具有同步性。

考虑到样本采样的随机性以及噪声样本点的独特性，神经网络拟合噪声点时复杂的输出地形会带来较大的方差。通过对零一误差进行偏差方差分析，我们发现方差项非单调性的变化主导了泛化误差回合双降现象的发生。基于该结论，我们提出了一种叫作优化方差的新指标来度量神经网络在优化过程中所表示的函数对采样噪声的稳定性。该指标通过指示模型在优化过程中引入方差的大小，能够仅在训练集上计算便可以与模型泛化误差保持一致的变化趋势。即使是在模型回合双降这样的泛化误差复杂变化的场景下，优化方差依然具有较好的预测能力。

5.2 展望

虽然本文一定程度上对神经网络回合双降现象做出了解释并对神经网络泛化性能的研究起到了促进作用，但是其背后的理论基础依然尚不清晰。由于在优化过程中模型、数据以及优化算法之间的复杂耦合，真实场景下神经网络的泛化性能很难被完整定义，也因此该研究方向尚存在着诸多难题与挑战：

- 目前的研究对于无限宽神经网络的优化过程已经有了较好的理解。研究发现无限宽神经网络等效于核学习，而其所表示的核函数被命名为神经正切核。因此通过核学习的理论，我们便可以搞清楚无限宽神经网络优化的具体过程。然而事实表明，实际使用的神经网络与无限宽神经网络之间依然存在着较大差异，如何将无限宽神经网络上成立的理论迁移到实际使用的神经网络上是一个难题。

- 本文通过实验证明了神经网络非单调性变化的频谱偏好来源于非流形区域输出地形在训练中的平整化，但是其背后的原因尚不清晰。虽然目前已有部分研究认为其主要原因是神经网络优化过程具有的隐式正则作用，但其作用的机理依然很难解释清楚。即使优化过程的隐式正则作用确实存在，为什么其能够对神经网络训练流形以外的输出地形起作用也依然是一个难题。

- 在本文中，我们提出了一种新的指标进行非校验集信息的神经网络泛化性能曲线预测。虽然该指标在训练回合数不断增加的情况下对神经网络的泛化误差具有较好的预测性，甚至能够对模型架构发生一定程度改变时具有跨模型的比较能力，但是在模型架构差异较大时该指标便不再具有可比性。如何获得一个更加统一的跨训练阶段、跨模型的指标来度量模型的泛化性能也是一个十分有意义的研究方向。

总体而言，我们认为研究神经网络类似误差双降这样的异常现象是理解其泛化能力的窗口。透过这些窗口，我们有望能够一睹神经网络的全貌。虽然这个研究方向任重而道远，但是我相信随着研究人员的不断努力，神经网络的泛化之谜有一天能被完全解开。

致　　谢

一叶而知秋，一叶而障目，所见不过一叶；一言而警世，一言而惑众，所闻不过一言。改变世界的麦克斯韦方程组不过区区四行，装着宇宙规律的头脑也不过分米见方。用微小撬动伟大，这或许便是科研的魅力。

从这个角度来看，这篇硕士毕业论文是毫无魅力可言的，因为它在篇幅上称不了微小，在意义上与伟大更是相去甚远。然而，虽然它在茫茫如烟海的硕士论文中显得平平无奇，但是其下却铺垫着我三年来的思考与实验、成功与失败、快乐与迷茫，正是这些赋予了它非凡的意义。当然，这篇论文的完成也离不开这三年来不断促使我进步的人，在这里我对他们表达自己由衷的谢意。我首先要感谢我的导师伍冬睿教授，感谢他愿意接受一个跨专业的学生来到实验室进行科研，也感谢他提供的宽松的研究氛围以及严谨的学术指导。正是他在我每一篇论文中的反复斟酌与修改，让我学会了科研所必须的严谨与认真。同时，我还要感谢实验室的每一位同学，他们既在科研中给予了我灵感，也在生活中让我收获了欢乐。除此之外我还要感谢我的家人，他们理解并支持我的每一个决定，即使这个决定看上去风险是那么的大。最后，我还要感谢每一个帮助过我的老师、朋友、同学，正是他们让我在三年间不断成长并且有所收获。由此观之，这篇硕士论文承载的是一个硕士生谈不上精彩但是至少丰富的三年，它既是一段生涯的结束，也是另一端生涯的开始。因此，它是沉甸的。

研究神经网络的泛化性能是困难的，因为它需要扎实的数学基础、广泛的文献阅读以及敏锐的洞察能力。仅仅是学习已有的理论便常常让人陷入晦涩的公式而无法建立直观的物理图像。然而，正是这个问题所具有的挑战性吸引了我，让我在这个并非实验室主干的方向上慢慢探索。确定这个研究方向时，我完全没有去想能否有研究成果或者能否完成毕业要求，凭借着的只是挑战难题的习惯以及对这个问题的兴趣罢了。现在想来，或许正是这种纯粹的热情让我能够在其上有所收获，所以我也要感谢这样一个自己。

三年的研究生涯让我明白，相对于科研本身我更喜欢的是思考问题这个过程。哪怕这个问题毫无意义，只要有趣，我便愿意花时间去慢慢思考，这无关乎论文与专利，也无关乎名利与待遇。我想，这或许也是我这三年中最大的收获——更加清晰地认识了自己。这样的想法可能是对的，也可能是错的；可能是有利的，也可能是有害的；可能是有可实现性的，也可能是太过理想的。但那又怎样呢？人太有限了，而我坦然接受这种局限性。与其瞻前顾后，不如自己走走看。

于是我决定走走看。

记于二〇二一年三月二十一日晚。

参考文献

[1] 王珏, 周志华. 机器学习及其应用[M]. 清华大学出版社, 2009.

[2] Vapnik V N. An overview of statistical learning theory. IEEE Trans. on Neural Networks, 1999, 10(5):988–999.

[3] Shalev-Shwartz S, Ben-David S. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, 2014.

[4] 孙志军, 薛磊, 许阳明, 等. 深度学习研究综述[J]. 计算机应用研究, 2012, 29(8):2806–2810.

[5] 刘建伟, 刘媛, 罗雄麟. 深度学习研究进展[J]. 计算机应用研究, 2014, 31(7):1921–1930.

[6] 孙志远, 鲁成祥, 史忠植, 等. 深度学习研究与进展[J]. 计算机科学, 2016, 43(2):1–8.

[7] Jacot A, Gabriel F, Hongler C. Neural tangent kernel: Convergence and generalization in neural networks. in: Proc. Advances in Neural Information Processing Systems, Montreal, Canada, December, 2018, 8571–8580.

[8] Zhang C, Bengio S, Hardt M, et al. Understanding deep learning requires rethinking generalization. in: Proc. Int'l Conf. on Learning Representations, Toulon, France, April, 2017.

[9] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. in: Proc. Advances in Neural Information Processing Systems, Lake Tahoe, NE, December, 2012, 1097–1105.

[10] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. in: Proc. Int'l Conf. on Learning Representations, San Diego, CA, May, 2015.

[11] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, NV, June, 2016, 770–778.

[12] Zagoruyko S, Komodakis N. Wide Residual Networks. in: Richard C. Wilson E R H, Smith W A P, editors, Proceedings of Proc. of the British Machine Vision Conf., York, UK, September, 2016, 87.1-87.12.

[13] Wang H, Keskar N S, Xiong C, et al. Identifying generalization properties in neural networks. arXiv, 2018, abs/1809.07402.

[14] Arpit D, Jastrzebski S, Ballas N, et al. A closer look at memorization in deep networks. in: Proc. 34th Int'l Conf. on Machine Learning, Sydney, Australia, August, 2017, 233–242.

[15] Kalimeris D, Kaplun G, Nakkiran P, et al. SGD on neural networks learns functions of increasing complexity. in: Proc. Advances in Neural Information Processing Systems, Vancouver, Canada, December, 2019, 3491–3501.

[16] Rahaman N, Baratin A, Arpit D, et al. On the spectral bias of neural networks. in: Proc. 36th Int'l Conf. on Machine Learning, Long Beach, CA, May, 2019, 5301–5310.

[17] Xu Z Q J, Zhang Y, Xiao Y. Training behavior of deep neural network in frequency domain. in: Proc. Int'l Conf. on Neural Information Processing, Sydney, Australia, December, 2019, 264–274.

[18] Xu Z Q J, Zhang Y, Luo T, et al. Frequency principle: Fourier analysis sheds light on deep neural networks. Communications in Computational Physics, 2020, 28(5):1746–1767.

[19] Nakkiran P, Kaplun G, Bansal Y, et al. Deep double descent: Where bigger models and more data hurt. in: Proc. Int'l Conf. on Learning Representations, Addis Ababa, Ethiopia, April, 2020.

[20] Advani M S, Saxe A M, Sompolinsky H. High-dimensional dynamics of generalization error in neural networks. Neural Networks, 2020, 132:428–446.

[21] Belkin M, Hsu D, Ma S, et al. Reconciling modern machine learning practice and the classical bias-variance trade-off. Proceedings of the National Academy of Sciences, 2019, 116(32):15849–15854.

[22] Geiger M, Jacot A, Spigler S, et al. Scaling description of generalization with number of parameters in deep learning. Journal of Statistical Mechanics: Theory and Experiment, 2020, 2020(2):023401.

[23] Maddox W J, Benton G, Wilson A G. Rethinking parameter counting in deep models: Effective dimensionality revisited. arXiv, 2020, abs/2003.02139.

[24] Mitra P P. Understanding overfitting peaks in generalization error: Analytical risk curves for l_2 and l_1 penalized interpolation. arXiv, 2019, abs/1906.03667.

[25] Hastie T, Montanari A, Rosset S, et al. Surprises in high-dimensional ridgeless least squares interpolation. arXiv, 2019, abs/1903.08560.

[26] Belkin M, Hsu D, Xu J. Two models of double descent for weak features. SIAM Journal on Mathematics of Data Science, 2020, 2(4):1167–1180.

[27] Yang Z, Yu Y, You C, et al. Rethinking bias-variance trade-off for generalization of neural networks. in: Proc. 37th Int'l Conf. on Machine Learning, Vienna, Austria, July, 2020, 10767–10777.

[28] Bartlett P L, Long P M, Lugosi G, et al. Benign overfitting in linear regression. Proceedings of the National Academy of Sciences, 2020, 117(48):30063–30070.

[29] Muthukumar V, Vodrahalli K, Subramanian V, et al. Harmless interpolation of noisy data in regression. IEEE Journal on Selected Areas in Information Theory, 2020, 1(1):67–83.

[30] Neal B, Mittal S, Baratin A, et al. A modern take on the bias-variance tradeoff in neural networks. arXiv, 2018, abs/1810.08591.

[31] Heckel R, Yilmaz F F. Early stopping in deep networks: Double descent and how to eliminate it. arXiv, 2020, abs/2007.10099.

[32] Pascanu R, Montufar G, Bengio Y. On the number of response regions of deep feed forward networks with piece-wise linear activations. arXiv, 2014, abs/1312.6098.

[33] Montufar G F, Pascanu R, Cho K, et al. On the number of linear regions of deep neural networks. in: Proc. Advances in Neural Information Processing Systems, Montreal, Canada, December, 2014, 2924–2932.

[34] Kingma D P, Ba J. Adam: A method for stochastic optimization. in: Proc. Int'l Conf. on Learning Representations, Banff, Canada, April, 2014.

[35] Poole B, Lahiri S, Raghu M, et al. Exponential expressivity in deep neural networks through transient chaos. in: Proc. Advances in Neural Information Processing Systems, Barcelona, Spain, December, 2016, 3360–3368.

[36] Arora R, Basu A, Mianjy P, et al. Understanding deep neural networks with rectified linear units. in: Proc. Int'l Conf. on Learning Representations, Vancouver, Canada, May, 2018.

[37] Bengio Y, Simard P, Frasconi P, et al. Learning long-term dependencies with gradient descent is difficult. IEEE Trans. on Neural Networks, 1994, 5(2):157–166.

[38] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int'l Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 1998, 6(02):107–116.

[39] Balduzzi D, Frean M, Leary L, et al. The shattered gradients problem: If resnets are the answer, then what is the question? in: Proc. 34th Int'l Conf. on Machine Learning, Sydney, Australia, August, 2017, 342–350.

[40] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. in: Proc. 32nd Int'l Conf. on Machine Learning, Lile, France, July, 2015, 448–456.

[41] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 2014, 15(1):1929–1958.

[42] Toth C D, O'Rourke J, Goodman J E. Handbook of discrete and computational geometry, Third ed. Chapman and Hall/CRC, 2017.

[43] Linial N. Hard enumeration problems in geometry and combinatorics. SIAM Journal on Algebraic Discrete Methods, 1986, 7(2):331–335.

[44] Freund R M, Orlin J B. On the complexity of four polyhedral set containment problems. Mathematical programming, 1985, 33(2):139–145.

[45] Biggio B, Corona I, Maiorca D, et al. Evasion attacks against machine learning at test time. in: Proc. Joint European Conf. on Machine Learning and Knowledge Discovery in Databases, Berlin, Germany: Springer, September, 2013, 387–402.

[46] Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks. in: Proc. Int'l Conf. on Learning Representations, Banff, Canada, April, 2014.

[47] Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. in: Proc. Int'l Conf. on Learning Representations, San Diego, CA, May, 2015.

[48] Moosavi-Dezfooli S M, Fawzi A, Frossard P. Deepfool: A simple and accurate method to fool deep neural networks. in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, NV, June, 2016, 2574–2582.

[49] Carlini N, Wagner D. Towards evaluating the robustness of neural networks. in: Proc. IEEE Symposium on Security and Privacy, San Jose, CA, May, 2017, 39–57.

[50] Dyer M E, Frieze A M. On the complexity of computing the volume of a polyhedron. SIAM Journal on Computing, 1988, 17(5):967–974.

[51] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, 86(11):2278–2324.

[52] Krizhevsky A. Learning multiple layers of features from tiny images. 2009.

[53] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. in: Proc. of the 13th Int'l Conf. on Artificial Intelligence and Statistics, Sardinia, Italy, May, 2010, 249–256.

[54] Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks. in: Proc. Int'l Conf. on Learning Representations, Vancouver, Canada, May, 2018.

[55] Bjorck N, Gomes C P, Selman B, et al. Understanding batch normalization. in: Proc. Advances in Neural Information Processing Systems, Montreal, Canada, December, 2018, 7694–7705.

[56] Novak R, Bahri Y, Abolafia D A, et al. Sensitivity and generalization in neural networks: An empirical study. in: Proc. Int'l Conf. on Learning Representations, Vancouver, Canada, May, 2018.

[57] Hanin B, Rolnick D. Complexity of linear regions in deep networks. in: Proc. 36th Int'l Conf. on Machine Learning, Long Beach, CA, May, 2019, 2596–2604.

[58] Galloway A, Golubeva A, Tanay T, et al. Batch normalization is a cause of adversarial vulnerability. arXiv, 2019, abs/1905.02161.

[59] Benz P, Zhang C, Kweon I S. Batch normalization increases adversarial vulnerability: Disentangling usefulness and robustness of model features. arXiv, 2020, abs/2010.03316.

[60] Netzer Y, Wang T, Coates A, et al. Reading digits in natural images with unsupervised feature learning. in: Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011, Granada, Spain, December, 2011.

[61] Ronen B, Jacobs D, Kasten Y, et al. The convergence rate of neural networks for learned functions of different frequencies. in: Proc. Advances in Neural Information Processing Systems, Vancouver, Canada, December, 2019, 4763–4772.

[62] Cao Y, Fang Z, Wu Y, et al. Towards understanding the spectral bias of deep learning. arXiv, 2019, abs/1912.01198.

[63] Geman S, Bienenstock E, Doursat R. Neural networks and the bias/variance dilemma. Neural Computation, 1992, 4(1):1–58.

[64] Friedman J, Hastie T, Tibshirani R. The elements of statistical learning, Second ed., volume 1. Springer series in statistics New York, 2001.

[65] Kong E B, Dietterich T G. Error-correcting output coding corrects bias and variance. in: Proc. 12th Int'l Conf. on Machine Learning, Tahoe City, CA, July, 1995, 313–321.

[66] Tibshirani R. Bias, variance and prediction error for classification rules. Technical report, Department of Preventive Medicine and Biostatistics and Department of Statistics, University of Toronto, Toronto, Canada, 1996.

[67] Kohavi R, Wolpert D H, et al. Bias plus variance decomposition for zero-one loss functions. in: Proc. 13th Int'l Conf. on Machine Learning, Bari, Italy, July, 1996, 275–283.

[68] Heskes T. Bias/variance decompositions for likelihood-based estimators. Neural Computation, 1998, 10(6):1425–1433.

[69] Domingos P. A unified bias-variance decomposition for zero-one and squared loss. in: Proc. of the 17th National Conf. on Artificial Intelligence, Austin, TX, July, 2000, 564–569.

[70] Boyd S, Vandenberghe L. Convex optimization. Cambridge University Press, 2004.

[71] Keskar N S, Mudigere D, Nocedal J, et al. On large-batch training for deep learning: Generalization gap and sharp minima. in: Proc. Int'l Conf. on Learning Representations, Toulon, France, April, 2017.

[72] Neyshabur B, Tomioka R, Srebro N. Norm-based capacity control in neural networks. in: Proc. of the 28th Conf. on Learning Theory, Paris, France, July, 2015, 1376–1401.

[73] Dinh L, Pascanu R, Bengio S, et al. Sharp minima can generalize for deep nets. in: Proc. 34th Int'l Conf. on Machine Learning, Sydney, Australia, August, 2017, 1019–1028.

[74] Neyshabur B, Bhojanapalli S, McAllester D, et al. Exploring generalization in deep learning. in: Proc. Advances in Neural Information Processing Systems, Long Beach, CA, January, 2018, 5947–5956.

[75] Chatterji N, Neyshabur B, Sedghi H. The intriguing role of module criticality in the generalization of deep networks. in: Proc. Int'l Conf. on Learning Representations, Addis Ababa, Ethiopia, April, 2020.

[76] Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. arXiv, 2017, abs/1708.07747.

附录 1　攻读硕士学位期间发表论文目录

[1] **Xiao Zhang**, Dongrui Wu, Lieyun Ding, Hanbin Luo, Chin-Teng Lin, Tzyy-Ping Jung and Ricardo Chavarriaga. "Tiny Noise, Big Mistakes: Adversarial Perturbations Induce Errors in Brain-Computer Interface Spellers", National Science Review, vol. 8, no. 4, 2021.（第一作者，顶级期刊，IF=16.69）

[2] **Xiao Zhang** and Dongrui Wu. "On the Vulnerability of CNN Classifiers in EEG-Based BCIs", IEEE Trans. on Neural Systems and Rehabilitation Engineering, vol. 27, no. 5, pp. 814-825, 2019.（第一作者，SCI 1区期刊，IF=3.34）

[3] **Xiao Zhang** and Dongrui Wu. "Empirical Studies on the Properties of Linear Regions in Deep Neural Networks", Proc. Int'l Conf. on Learning Representations (ICLR), Addis Ababa, Ethiopia, April 2020.（第一作者，顶级会议）

[4] **Xiao Zhang**, Haoyi Xiong and Dongrui Wu. "Rethink the Connections among Generalization, Memorization and the Spectral Bias of DNNs", Proc. Int'l Joint Conf. on Artificial Intelligence (IJCAI), Montreal, Canada, August 2021.（第一作者，顶级会议）

[5] **Xiao Zhang**, Dongrui Wu, Haoyi Xiong and Bo Dai. "Optimization Variance: Exploring Generalization Properties of DNNs", 2021, *work in progress*.（第一作者）

附录 2　攻读硕士学位期间的其他研究成果

[1] 伍冬睿，**张潇**，一种针对以卷积神经网络为基础的 EEG 脑机接口的攻击方法，ZL201811543220.3，CN109376556B。（第二作者，发明专利）

[2] **张潇**，何赫，郭陈凤，彭睿旻，伍冬睿，2019世界机器人大赛第三届中国脑机接口比赛技术赛全国一等奖，颁奖单位：国家自然科学基金委，2019 年。(排名第一，竞赛获奖)

[3] **张潇**，刘子涵，崔雨琦，伍冬睿，IEEE WCCI Open Source Intelligence Discovery for Cybersecurity Threat 第一名，颁奖单位：IEEE WCCI 比赛组委会，2018 年。(排名第一，竞赛获奖)

[4] 刘子涵，王阳，崔雨琦，徐祎璠，何赫，赵昶铭，**张潇**，张稳，谭显烽，伍冬睿，第一届深圳医疗健康大数据创新应用国际大赛标准组三等奖，颁奖单位：深圳市发改委、卫计委，2018 年。(排名第七，竞赛获奖)

[5] 2020 年华中科技大学人工智能与自动化学院硕士国家奖学金，颁奖单位：华中科技大学，2020 年。

[6] 2019 年华中科技大学人工智能与自动化学院硕士国家奖学金，颁奖单位：华中科技大学，2019 年。

[7] 2020 年汇顶科技一等奖学金，颁奖单位：汇顶科技股份有限公司，2020 年。