arXiv:2303.09935v4 [cs.NE] 5 Nov 2024

用于分类和稳健回归的替代损失函数可提高人工神经网络的准确率

Mathew Mithra Noel[a]、Arindam Banerjee[b]、YugOswal[c]、Geraldine Bessie Amali D[d]、Venkataraman Muthiah-Nakarajan[e]

[a]韦洛尔理工学院电气工程学院，电子邮件：*mathew.m@vit.ac.in*，[b]安永*GDS*高级数据科学家，电子邮件：*arindam.banerjee3@gds.ey.com*，[c]韦洛尔理工学院计算机科学与工程学院，电子邮件：*yoswal071@gmail.com*，[d]韦洛尔理工学院计算机科学与工程学院，电子邮件：*geraldine.amali@vit.ac.in*，[e]韦洛尔理工学院电气工程学院，电子邮件：*mnvenkataraman@vit.ac.in*，

摘要

所有机器学习算法都会使用损失、代价、效用或奖励函数来编码学习目标并监督学习过程。这种用于监督学习的函数常常是一种未被充分认识的超参数，它决定了如何惩罚错误输出，并且可以通过调整来提升性能。本文表明，神经网络的训练速度和最终准确率可能显著取决于用于训练神经网络的损失函数。特别是，不同损失函数的导数值可能存在显著差异，导致基于梯度下降的反向传播（BP）训练后性能大不相同。本文探讨了使用新的损失函数对性能的影响；这些函数同样具有凸性，但与常用的交叉熵损失相比，对误差的惩罚方式有所不同。本文提出了两种新的分类损失函数，可在多种基准任务中显著提升性能。此外，本文还提出了一种名为平滑绝对误差的新损失函数，在含有大量离群值的数据集上，其表现优于平方误差、Huber和Log-Cosh损失。这种平滑绝对误差损失函数具有无穷阶可微性，与用于稳健回归的Huber和Log-Cosh损失相比，它对绝对误差损失的近似更为贴切。

关键词：
L损失函数、稳健回归、图像分类、文本分类 ，

提交至期刊的预印本 *2024*年*11*月*6*日

1. 引言

人工神经网络（ANN）是一类通用函数逼近器，它们通过基于梯度下降（GD）的优化过程，学习目标函数的参数化近似 [1]。因此，人工神经网络的学习可以归结为学习一组有限的实数参数，即人工神经网络模型中的权重和偏置。通过最小化损失（也称为代价）函数，可以计算出能够良好逼近目标函数的参数；该函数用于衡量人工神经网络的输出与目标值之间的差异。损失函数的一个重要特点是，它能将人工神经网络模型在整个数据集上的表现提炼为一个标量值。损失函数是关于人工神经网络模型所有参数的单个连续可微实值函数，通常具有凸性。梯度下降要正常运行，损失函数必须连续可微。此外，凸损失函数更可取，因为凸函数只有一个全局最小值。

选择一种受统计估计理论启发的损失函数，以根据数据集计算最可能的参数，是合理的。历史上，均方误差（MSE）和最大似然估计理论（MLE）中的交叉熵几乎被普遍用于训练人工神经网络。最大似然估计理论将未知参数设定为能使观测到实验数据的概率最大化的值。为方便起见，通常使用观测数据概率的对数，由此得到交叉熵损失。因此，最大似然估计通过最大化观测数据概率关于未知参数的对数来进行。出于历史原因，人们使用梯度下降（GD）最小化概率的负对数。

均方误差最常用于回归分析，此时需要预测连续实数值；交叉熵则用于分类问题，此时预测输出是离散的 [2]。如前所述，均方误差和交叉熵损失函数均源自最大似然估计理论。当模型为线性模型且噪声服从高斯分布时，均方误差是最大似然估计。然而，即使这些假设无法得到证实，实践中也经常使用均方误差。对于分类问题，均方误差和交叉熵损失都可以使用，但使用交叉熵通常能更快地学习，因为交叉熵损失中的对数函数会使梯度更大。交叉熵损失也称为二元交叉熵（BCE）损失，其灵感源自信息论，用于衡量两个概率分布之间的差异 [3]。

1.1. 数学形式化

考虑一个包含 N 个训练样本的数据集： $D = \{x^i, y^i : \ i = 1..N\}$。其中 $x \in R^m, y \in R^n$ 通常对于某些自然数 m 和 n 都成立。接下来，我们将阐述并讨论适用于分类问题的若干新损失函数。

对于二元分类问题 $y \in \{0, 1\}$。对于二元分类问题，输出层由一个 S 形神经元组成，该神经元输出概率$P(y = 1|x)$。此概率取决于人工神经网络中的所有权重和偏置，因此记为 $h_\theta(x)$。要使用最大似然估计理论计算未知参数，首先应计算给定参数时数据的概率。

假设数据集中的每个训练样本对彼此独立：

$$P(D|\theta) = \prod_{i=1}^{N} P(x^i, y^i; \theta)$$

.

$\prod_{i=1}^{N} P(x^i, \ y^i; \ \theta) \ = \prod_{i=1}^{N} P(y^i|x^i; \ \theta)P(x^i)$，根据条件概率的定义。

由于 $P(x^i)$ 与 θ 独立，最大化 $P(D|\theta)$ 等价于最大化 $\log(\prod_{i=1}^{N} P(y^i|x^i; \ \theta))$。

因此，要最大化的对数似然函数为

$$\log(\prod_{i=1}^{N} P(y^i|x^i; \theta)) = \sum_{i=1}^{N} P(y^i|x^i; \theta)$$

但是 $P(y^i|x^i; \ \theta) \ = h_\theta(x^i)^{y^i}(1 - h_\theta(x^i))^{1-y^i}$。

无需最大化对数似然，而是可以最小化负对数似然。

化简后的负对数似然就是损失

$$\mathcal{L}(\theta) = -\sum_{i=1}^{N}(y^i \log(h_\theta(x^i)) + (1 - y^i) \log(1 - h_\theta(x^i)))$$

.

乘以任意正数都不会改变最小值，因此为便于数值计算，我们可以除以 N，得到

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}(y^i \log(h_\theta(x^i)) + (1-y^i)\log(1-h_\theta(x^i))))$$

$$l(\theta) = -\frac{1}{N}(\sum_{i=1}^{N} y^i \log(\hat{y}^i) + (1-y^i)\log(1-\hat{y}^i)) \tag{1}$$

为方便记号，将作为目标值 y 近似值的 $h_\theta(x)$ 记为 $\hat{y}$。对于单个训练样本对 (x, y)，上述二元交叉熵损失为

$$l(y, \hat{y}) = -[y\log(\hat{y}) + (1-y)\log(1-\hat{y})] \tag{2}$$

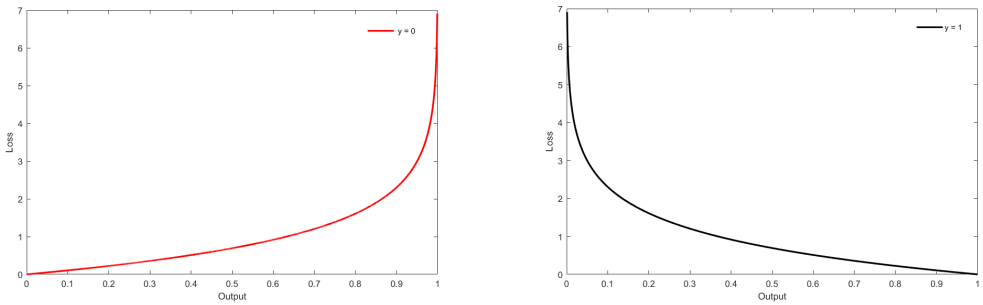

图1：目标值为 $y = 0$ （左）和 $y = 1$ （右）时二元交叉熵损失的曲线图。

目标值恰好取两个可能值之一，即 0 或 1，因此公式2中只有一项非零。当 $y = 1$, $l(y, \hat{y}) = -\log(\hat{y})$，而当 $y = 0$,$l(y, \hat{y}) = -\log(1-\hat{y})$。图1.1展示了目标值 y 分别为 0 和 1 时，二元分类器输出 $\hat{y}$ 对二元交叉熵损失的影响。可以看出，目标值等于输出时，损失恰好为零。此外，损失是可微且具有凸性的，并且当输出趋近于错误的目标值时，损失趋于无穷大。凸性有助于避免引入不必要的局部极小值，而可微性则是基于梯度的反向传播学习所必需的。接下来，我们将提出具备以下理想属性的新损失函数：

- $l(y, \hat{y}) = 0$ 和 $\hat{y} = y$
- $l(0,\hat{y})$ 和 $l(1,\hat{y})$ 是 $\hat{y}$ 的凸性可微函数

- $l(0,0) = l(1,1) = 0$
- $\lim_{\hat{y}\to 1} l(0,\hat{y}) = \infty$ 和 $\lim_{\hat{y}\to 0} l(1,\hat{y}) = \infty$

在上述 $0 < \hat{y} < 1$ 中，因为 $\hat{y}$ 是 $P(y = 1|x)$ 的概率。对于多分类问题，$\hat{y}$ 将是一个维度与类别数相同的向量。对于 C 个类别，输出层将是一个具有 C 个神经元的 softmax 层，损失函数是每个输出对应的损失函数之和：$-\frac{1}{N}\sum_{j=1}^{C}\sum_{i=1}^{N}[y_j^i \log(\hat{y_j}^i) + (1-y_j^i)\log(1-\hat{y_j}^i)]$。除了使用更好的损失函数外，还可以使用已知性能优于 ReLU 等常用激活函数的先进激活函数 [4], [5] 来提升性能。

训练期间，人工神经网络模型中的每个参数都使用 $\frac{\partial \mathcal{L}}{\partial \Theta}$ 进行更新。每个参数只通过网络的输出间接影响损失。使用链式法则：

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial \mathcal{L}}{\partial \hat{y}} . \frac{\partial \hat{y}}{\partial \Theta} \tag{3}$$

因此，损失对输出的导数比实际损失值更重要。为了比较不同的损失函数，我们引入以下定义：

定义：考虑两个损失函数 $\mathcal{L}_1(\hat{y}, y)$ 和 $\mathcal{L}_2(\hat{y}, y)$。对于 $\hat{y}$ 的某个取值集合，如果 $\frac{\partial \mathcal{L}_1}{\partial \hat{y}} \geq \frac{\partial \mathcal{L}_2}{\partial \hat{y}}$，我们就称 $\mathcal{L}_1$ 比 $\mathcal{L}_2$ 更严格。—— ——

如果 $\mathcal{L}_1$ 比 $\mathcal{L}_2$ 更严格，我们就称 $\mathcal{L}_2$ 比 $\mathcal{L}_1$ 更宽松。下面我们将介绍一些与二元交叉熵相比对网络误差惩罚方式不同的替代损失函数。

本研究的主要贡献如下：

- 提出了两种在基准分类任务上性能优于交叉熵的新损失函数
- 提出了一种新的稳健回归损失函数，其性能优于 Huber 损失和 Log-Cosh 损失
- 本文对多种架构和基准测试中的不同损失函数进行了广泛比较。

2. 用于分类的新损失函数

接下来，我们将探讨满足 1.1 节所列性质的其他损失函数。针对二分类问题，并假设 $0 < \hat{y} < 1$ and$y \in \{0, 1\}$，本文提出以下新的损失函数：

- **M 损失：** $M(y, \hat{y}) = y\left(\frac{1-\hat{y}}{y} - 1\right)$
- **L损失：** $L(y, \hat{y}) = \frac{y}{\sqrt{1-(1-\hat{y})^2}} - 1$

在上述情况下，当 $y = 0$ 时，损失为零；只有当 $y = 1$时，才会对人工神经网络的错误进行惩罚。下面给出了上述损失的修改版本，这些版本即使在 $y = 0$ 时也会对人工神经网络的错误进行惩罚，称为完整损失：

完整损失函数：

- **M 损失：** $M(y, \hat{y}) = \frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}} - 1$
- **L损失：** $L(y, \hat{y}) = \frac{\sqrt{1-(1-\hat{y})^2}}{y} + \frac{1-y}{\sqrt{1-\hat{y}^2}} - 1$

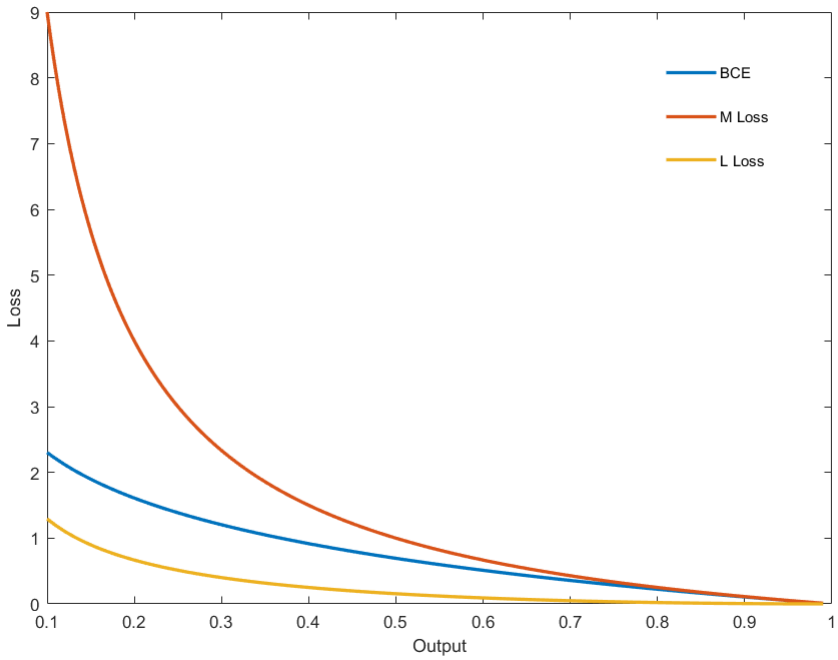

图 2：目标值 y $= 1$ 时不同损失函数的图示。

图 2 展示了本文提出的交叉熵与其他损失函数之间的差异。从图 2 可以看出，M 损失比二元交叉熵更严格，而对于大多数输出值，L损失比二元交叉熵更宽松。要将上述损失函数应用于多分类问题，需要将每个输出对应的损失相加。特别地，如果 $l(y_j^i,\ \hat{}y_j^i)$ 是 j^{th} 输出和 i^{th} 训练样本对对应的损失，则总损失为 $\mathcal{L} = \frac{1}{N}\sum_{j=1}^{C}\sum_{i=1}^{N} l(y_j^i,\ \hat{}y_j^i)$，其中 N 是小批量梯度下降中小批量的大小。

3. 稳健回归的新损失函数

对于回归问题，最常用的是均方误差（MSE）。然而，MSE 对离群值非常敏感，因此人们希望找到一种可微的平均绝对误差（MAE）近似形式。尤其是当回归训练数据来自具有“重尾”的分布时，如果训练期间使用 MSE，许多离群值的存在会导致较大的预测误差。MAE 对所有数据一视同仁，且对离群值具有鲁棒性，但其应用受到限制，因为它在原点处不可微。

MAE 损失的另一个问题是，它在原点附近的导数较大（± 1），会导致梯度下降过程中在最小值附近振荡。人工神经网络的训练损失函数必须可微，因为反向传播算法要求损失函数可微。因此，过去人们探索过一些可微且计算成本低的均方误差损失替代方案，例如 Huber 损失 [6] 和对数双曲余弦损失 [7]。本文提出以下一种计算成本低且处处可微的 MAE 损失平滑近似形式，称为平滑平均绝对误差（SMAE）：

$$SMAE(e) = e \tanh(\frac{e}{2})$$

本文提出的 SMAE 损失具有以下优点：

对于较小的误差值（$|e| << 1$），SMAE 近似于均方误差，因为 $SMAE(e) = e \tanh(\frac{e}{2}) \approx \frac{e^2}{2}$。

对于较大的误差值（$e \to \infty$），SMAE 近似于平均绝对误差，因为 $SMAE(e) = e \tanh(\frac{e}{2}) \approx |e|$。

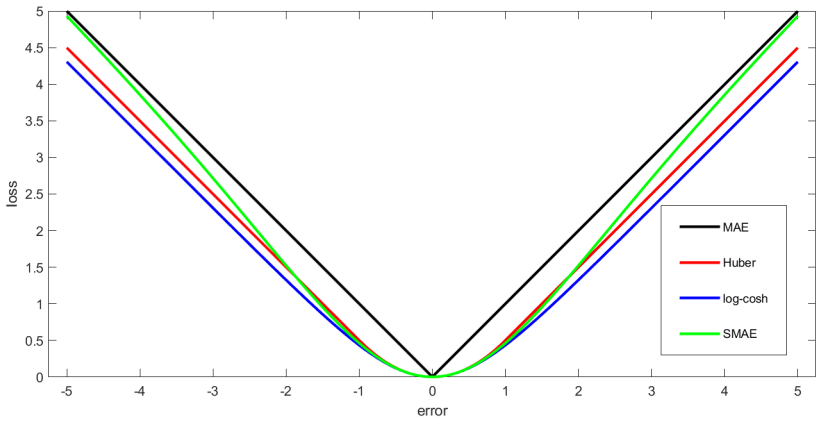

图 3：用于稳健回归的不同损失函数图示。

图 3 对比了本文提出的 SMAE 损失与用于回归的不同损失函数。由图 3 可清楚看出，与其他用于稳健回归的损失函数相比，SMAE 对理想的 MAE 损失的逼近更为准确。此外，从图 3 还可以看出，对于较大的

误差值，SMAE 损失会收敛到 MAE 损失，而 Huber 和 Log-Cosh 损失则不同，它们相对于理想的 MAE 损失始终存在偏差。此外，Huber 损失包含一个需要调优的超参数，因此无法将其与不含超参数的其他损失函数直接比较。

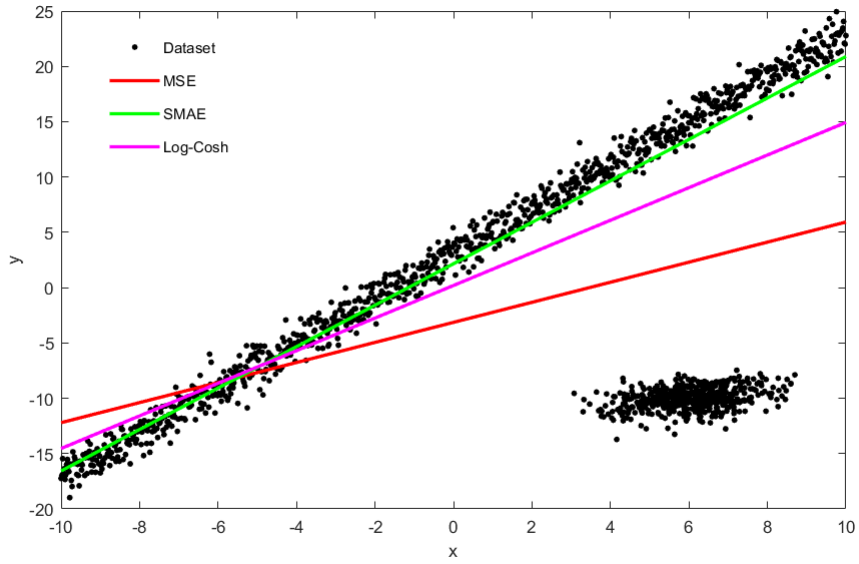

图 4：当数据集中含有大量离群值时，使用不同损失函数进行线性回归的效果。训练数据集以黑点表示，由 1500 个样本组成，其中包含 500 个离群值。

图 4 显示，与其他损失函数相比，本文提出的 SMAE 损失受离群值的影响较小。与其他损失函数相比，SMAE 损失更接近理想的 MAE 损失，因此具有更强的鲁棒性。表 1 展示了使用不同损失函数估计线性模型参数时的参数估计误差百分比。训练数据由 500 个离群值和 1000 个点（图 4）组成，这些点是通过向线性模型 $y = 2 * x + 3$ 添加零均值、单位方差的高斯噪声生成的。下面所示的 500 个离群值来自均值向量为 μ、协方差矩阵为 $\mathbf{\Sigma}$ 的二维高斯分布：

$$\mu = \begin{bmatrix} 6 \\ -10 \end{bmatrix}$$

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix}$$

表 1：在含有离群值的数据集上，使用不同回归损失函数时的参数估计准确率。

Loss 函数	**参数估计误差（%）**	
	斜率	Y轴截距
SMAE	8.73	22.6
MSE	68.2	205.82
Log-Cosh	22.76	82.3

4. 基准视觉数据集上的性能比较

下文将在各种基准架构和问题上，将 M 损失函数和 L损失函数的两个不同版本与标准交叉熵损失进行比较。不同的损失函数仅在训练期间用于计算模型参数。训练完成后，将不同模型在独立测试数据集上的准确率作为指标，对其性能进行比较。

4.1. CIFAR-10 基准上的比较

使用不同的损失函数，基于 CIFAR-10 数据集训练 VGG-19 模型。CIFAR-10 数据集由 60000 张 32x32 彩色图像组成，分为 10 个类别，每个类别有 6000 张图像。其中有 50000 张训练图像和 10000 张测试图像。使用学习率为 1e-5 的 Adam优化器。在 Google Colab 环境中对模型进行了 50 个训练轮次的训练，每轮步数为 100。研究了五种损失函数：i) 交叉熵损失，ii) L损失，iii) M损失，iv) 完整 L 损失，以及 v) 完整M损失。这五种损失函数对应的训练准确率和测试准确率如表 2 所示。

表 2：P　在 CIFAR-10 基准上使用不同损失函数训练的 VGG-19 的性能　rk

损失函数	**训练准确率**	**测试准确率**
交叉熵损失	0.5806	0.5668
L损失	0.5791	0.5684
M 损失	0.5745	0.5666
完整 L 损失	0.5827	0.571
完整 M 损失	0.5762	0.5668

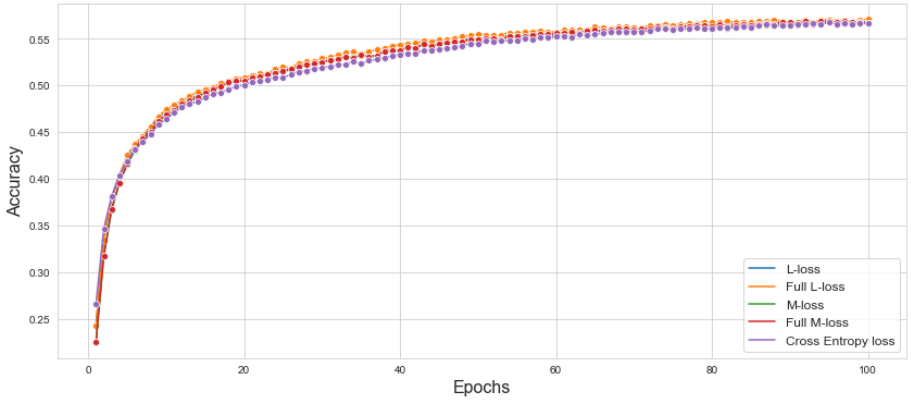

图 5：CIFAR-10 基准上准确率随训练轮次的变化。

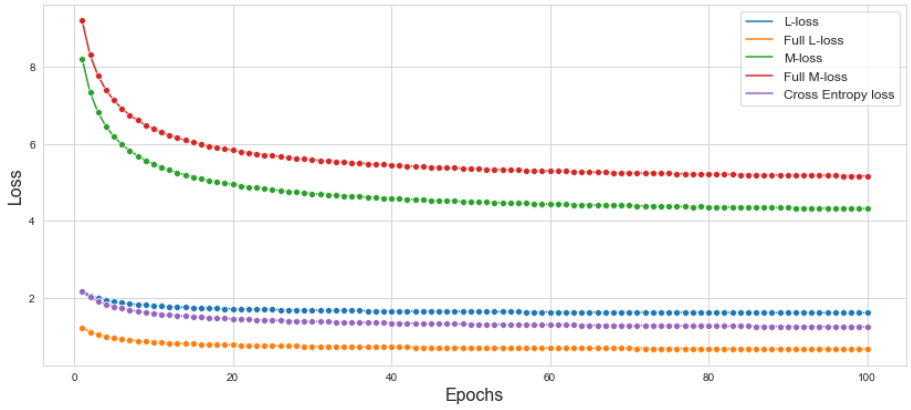

图 6：CIFAR-10 基准上损失随训练轮次的变化。

图 5 和图 6 展示了使用不同损失函数进行训练的过程。可以看出，完整 L 损失优于其他所有损失函数。

4.2. Imagenette 数据集上的比较

使用不同的损失函数和 Imagenette 数据集 [10] 训练 VGG-19 模型。Imagenette 是 ImageNet 数据集的一个子集，包含 10 个类别。Imagenette 的 160px-v2 版本包含 9,469 张训练图像和 3,925 张测试图像。进行迁移学习时，使用了 TensorFlow 中预训练的 VGG-19 模型，其权重在“imagenet”数据集上预训练。未采用顶部的全连接层，而是明确设定这些层使用给定数据集进行训练。

训练模型时使用了学习率为 1e-5 的 Adam优化器。模型在 Colab 环境中训练了 50 个训练轮次，每轮步数为 100。表 3 展示了 5 种损失函数在 Imagenette 基准上的表现。

表 3：性 能：使用不同损失函数在 Imagenette 基准上训练的 VGG-19 rk

损失函数	训练准确率	测试准确率
交叉熵损失	0.73	0.74
L损失	0.75	0.77
M 损失	0.74	0.78
完整 L 损失	0.72	0.78
完整 M 损失	0.74	0.76

表 3 显示，使用 M 损失和 L 损失（两个版本）时的测试准确率显著高于使用交叉熵损失时的测试准确率。图 7 和图 8 展示了使用不同损失函数进行训练的过程。可以观察到，本文提出的完整 L 损失在 Imagenette 基准上的表现优于所有其他损失函数。

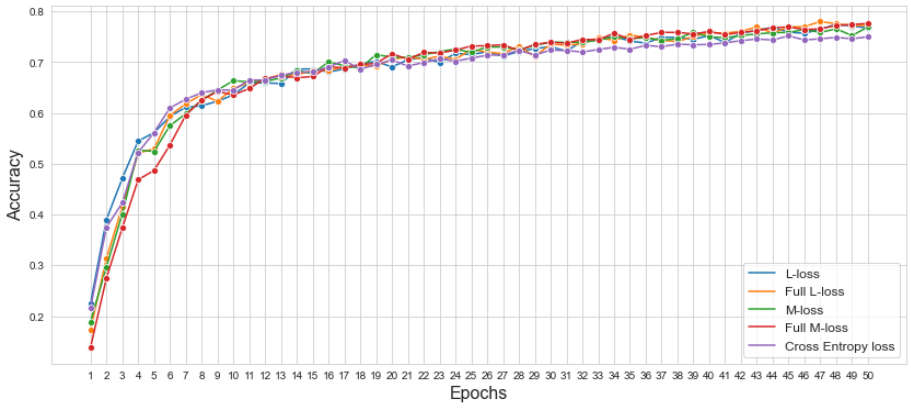

图 7：Imagenette 基准的准确率随训练轮次的变化。

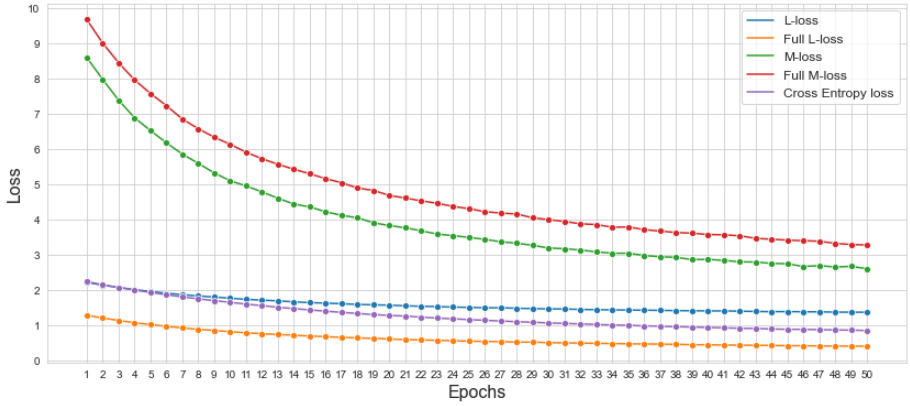

图8：Imagenette 基准上损失随训练轮次的变化。

5. 文本数据集上的比较

为了测试所提出的损失函数在计算机视觉任务和 VGG-19 模型之外的普适性，本文还考察了基于文本的基准测试和先进架构。

5.1. 美国消费者金融保护局（*CFPB*）数据集上的比较

首先，使用基于长短期记忆网络的模型对美国消费者金融保护局（CFPB）数据集 [11] 中的文本进行分类。CFPB 每周都会将数千名消费者对金融产品和服务的投诉转交给相关公司，以便公司作出回应。公司回应后，或自投诉发布之日起 15 天后（以先发生者为准），这些投诉便会公开。在 CFPB 数据集中，目标变量有 10 个可能的类别。该模型包含一个嵌入层，后接一个空间随机失活层、一个包含 100 个单元的长短期记忆网络层，以及一个用于输出 10 个可能类别的全连接层。

表4：在 CFPB 基准上使用不同损失函数训练的长短期记忆网络模型的性能

损失函数	测试准确率
二元交叉熵损失	0.822
L损失	0.843
M 损失	0.827
完整 L 损失	0.839
完整 M 损失	0.825

使用不同损失函数训练的长短期记忆网络模型的测试准确率见表4。这些实验结果表明，在大多数情况下，所提出的 L 损失和 M 损失的两种变体均优于交叉熵损失。

*5.2. IMDB*电影评论基准上的比较

使用全部 5 种损失函数训练 BERT 情感分析模型，根据评论文本将电影评论分类为正面或负面。使用来自互联网电影数据库（IMDB）的 50,000 条电影评论文本的大型电影评论数据集 [12] 训练 BERT 模型。研究采用了 TensorFlow Hub 中较小的 BERT 模型，该模型有 4 个隐藏层（Transformer块）、512 的隐藏维度和 8 个注意力头。该 BERT 模型使用 Adam优化器进行训练。表5 展示了这五种损失函数对应的测试准确率。表5 表明，L损失和完整 L 损失的表现略优于其他所有损失函数。

表5：P　使用不同损失函数训练的 BERT 模型在 IMDB 基准上的表现　k.

损失函数	**测试准确率**
二元交叉熵损失	0.8551
L损失	0.8557
M 损失	0.8539
完整 L 损失	0.8577
完整 M 损失	0.8555

6. 结论

本文探讨了在分类和稳健回归中使用常见损失函数替代方案的潜在优势。本文提出了四种新的分类损失函数，并证明它们在多种架构和基准测试中优于常见的二元交叉熵损失。研究表明，本文提出的分类损失在所有考察的基准测试和架构上均优于常见的二元交叉熵损失。在某些任务中，所提出的损失函数显著提高了分类准确率。与二元交叉熵损失相比，M 损失和完整 M 损失对错误的惩罚更为严格，而 L 损失和完整 L 损失则较为宽松。L 损失和 M 损失表现更优，这表明受最大似然估计理论启发的二元交叉熵损失并非最优，还可以设计出更好的损失函数。

许多真实世界数据集受到大量较大离群值的污染，这些离群值会干扰回归算法，导致较大的预测

误差。本文提出了一种与平均绝对误差（MAE）损失非常接近、无限可微且计算成本低廉的近似方法，称为 SMAE 损失。SMAE 损失无需像稳健回归中使用的 Huber 损失那样选择额外的超参数，而且比 Log-Cosh 损失对离群值显著更加稳健。

参考文献

[1] I. Goodfellow、Y. Bengio、A. Courville，《深度学习》，麻省理工学院出版社，2016。[2]T. Hastie、R. Tibshirani、J. Friedman，《统计学习要素》。施普林格统计学丛书，2001。[3] J. Brownlee，《机器学习概率论：了解如何利用 Python 驾驭不确定性》，Machine Learning Mastery，2019。[4] M. M. Noel、A. Trivedi、P. Dutta 等，《增长余弦单元：一种新颖的振荡激活函数，可加快训练并减少卷积神经网络中的参数》，arXiv 预印本 arXiv:2108.12943（2021）。[5] M. M. Noel、S. Bharadwaj、V. Muthiah-Nakarajan、P. Dutta、G. B. Amali，《受生物学启发的振荡激活函数可弥合生物神经元与人工神经元之间的性能差距》（2023）。 arXiv:2111.04020.[6] P. J. Huber，《位置参数的稳健估计》，《数学统计年刊》35 (1)（1964）73–101。 doi:10.1214/aoms/1177703732。URL https://doi.org/10.1214/aoms/1177703732[7] Q. Wang、Y. Ma、K. Zhao、Y. Tian，《机器学习中损失函数的综合调查》，《数据科学年刊》（2020）1–26。[8] K. Simonyan、A. Zisserman，《用于大规模图像识别的超深卷积网络》（2015）。 arXiv:1409.1556.[9] A. Krizhevsky、G. Hinton 等，《从微小图像中学习多层特征》（2009）。[10] J. Howard 等，Imagenette，URL https://github. com/fastai/imagenette 5（2020）11。[11]美国消费者金融保护局，https://www.consumerfinance.gov/data-research/consumer-complaints/。

[12] A. L. Maas、R. E. Daly、P. T. Pham、D. Huang、A. Y. Ng、C. Potts，《用于情感分析的词向量学习》，载于：《第49届计算语言学协会年会论文集：人类语言技术》，计算语言学协会，波特兰，俄勒冈州，美国，2011年，第142–150页。URL http://www.aclweb.org/anthology/P11-1015