计算机科学与技术学院

本科生《机器学习》课程作业

第一次大作业 实验报告

学　　号 **22281052**

姓　　名 张鳃沣

班　　级 **AI2201**

日　　期 **2024 年 11 月 30 日**

《机器学习》实验报告

第一次大作业 实验报告

张鲴沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

实验目的

本实验的主要目的是通过实现 Cover/Stego 图像数据集的二分类任务，掌握机器学习分类方法在图像数据处理中的应用技能。实验旨在熟悉数据处理的完整流程，包括数据集的预处理、特征的选取和提取、分类模型的构建与优化，以及实验结果的分析和可视化展示。通过本实验，可以系统性地学习如何选择合适的机器学习算法应对实际问题，同时培养自主编程的能力，增强解决问题的思维和技巧。

实验要求

实验对 Cover/Stego 图像数据集进行分类，将每张图像归类为 Cover 或 Stego。实验数据集包含 20000 张图像，其中 Cover 和 Stego 图像各占一半。学生需要选择至少两种不同的分类模型，例如逻辑回归（Logistic Regression）、线性判别分析（FLD）、支持向量机（SVM）等，完成分类任务。为了简化工作，分类模型的实现方式不作硬性要求，既可以自己编程实现，也可以调用现成的机器学习工具库（例如 Python 的 Scikit-learn 库）。在实验中，需合理划分训练集和测试集，并根据图像数据特性提取合适的特征用于分类模型的训练。数据集可通过提供的链接下载，下载后对其进行预处理。为了确保实验结果的有效性，分类模型的分类准确率需高于 50%。

图1 Cover图像样例

图2 Stego图像样例

Cover/Stego 图像数据集是一个用于研究图像隐写分析的标准数据集。隐写分析（Steganalysis）是信息安全领域中的一个重要方向，旨在检测图像中是否隐藏了秘密信息。数据集中包含的图像分为两类：Cover 图像和 Stego 图像。其中，**Cover 图像是未经处理的普通图像，而Stego 图像则是通过隐写技术在 Cover 图像中嵌入了秘密信息的图像**。这些嵌入信息对于肉眼通常是不可见的，但可以通过特征提取和分类模型进行检测和区分。

实验流程

首先需要获取实验所需的 Cover/Stego 图像数据集。下载并解压数据后，需对数据进行整理和分析，划分为训练集和测试集，以保证分类模型可以在未见过的数据上进行合理的评估。数据预处理后，需要结合实验任务的特点，对图像数据提取适合的特征。例如，可以通过分析图像的纹理、颜色分布或其他隐写信息特征来构建特征向量。

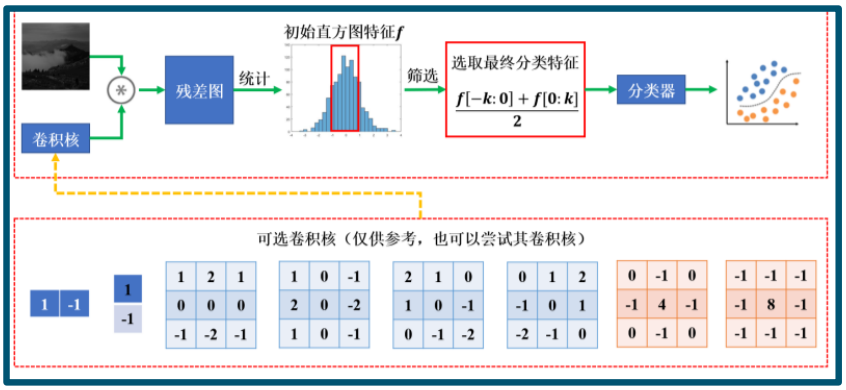


图3 基于直方图特征的隐写分析基本流程

在完成特征提取后，我们首先选择逻辑回归（Logistic Regression）作为基线模型进行分类实验。逻辑回归是一种经典的线性分类算法，其特点是易于实现、训练速度快，能够很好地处理特征与目标之间具有线性关系的数据。我们通过交叉验证优化正则化参数（如 C）。在完成逻辑回归模型的实验后，我们进一步引入支持向量机（Support Vector Machine, SVM）、LDR进行优化实验。SVM通过选择适当的核函数（如线性核和RBF核）可以灵活处理线性和非线性分类问题，并能通过调整正则化参数和核函数参数（如 C 和 gamma）进一步提升分类性能。此外，我们还实验性地加入卷积神经网络（CNN），构建一个简单的2层卷积模型，从图像数据中学习特征并进行分类。

在模型训练完成后，我们使用测试集数据评估模型性能，计算准确率（ACC）、AUC、F1-Score等评价指标，并绘制分类结果的性能对比图表。我们将分析不同模型的优缺点，包括逻辑回归在高维复杂数据上的局限性以及SVM在非线性分类中的表现等。同时，我们会关注模型在训练时间、资源消耗和预测准确性之间的权衡，总结其在分类效果上的差异，为后续的模型优化和选择提供依据。

张鲴沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第一部分　实验预处理

数据预处理流程

1. 设置工作目录

本部分通过设置当前工作目录确保代码的运行环境与数据集路径一致，否则由于工作目录导致程序无法读取实验数据集，造成报错。

2. 加载/分割数据集

加载和分割数据集的任务通过函数 load_images_from_folder 和 load_and_split_data 完成，实现从文件夹中加载图像的功能。并且对图像进行一系列处理，调整大小为 224×224 以统一输入格式，将像素值归一化到 [0, 1] 范围内，这有助于加速模型训练的收敛速度并增强数值稳定性。分割数据集时，将数据按 80%:20% 的比例划分为训练集和测试集。训练集用于模型的学习和参数调整，测试集用于验证模型的泛化能力。完成数据加载和分割后，通过统计训练集和测试集的标签分布，确保数据类别分布均衡。如果标签分布严重失衡，可能会导致模型对某些类别的预测能力显著下降。

3. 提取分类/残差特征

本部分通过多种核函数提取图像的残差特征，以找到最优的核函数提取特征。通过计算残差图的直方图特征，并将其归一化为概率密度（范围 [-0.5, 0.5] ），进一步提取图像的全局特征。并且绘制残差的直方图，能够直观展现特征提取的效果，帮助理解残差的分布规律。

4. PCA降维

为了降低数据维度并便于可视化，使用主成分分析（PCA）将高维特征降到二维空间。PCA 的目标是通过线性变换提取数据中方差最大的方向，并在此基础上生成较低维度的特征。降维后，能够在二维空间中直观地观察数据分布。分类任务中，若不同类别的点在二维空间中分布清晰分离，则说明特征提取是有效的；反之，则可能需要改进特征提取方法。并使用气泡图展示降维后的特征分布。若类别严重混叠，则提示需要优化特征提取或选择其他降维方法。

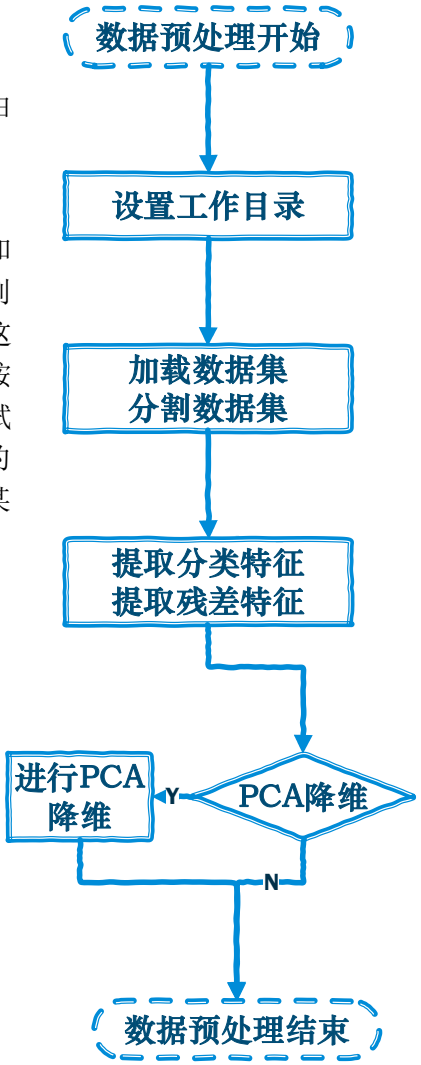


图4 数据预处理流程

实验选取核函数

拉普拉斯核：一种经典的高通滤波器，常用于图像的边缘检测和细节增强，能够准确捕捉图像中的局部变化，在隐写分析中被广泛应用，用于检测图像中的隐蔽特征。是一种方向无关的核。

0	-1	0
-1	4	-1
0	-1	0

拉普拉斯核

增强型边缘检测核：对垂直方向的边缘变化特别敏感，较大的权值会强化垂直方向的特征，使图像中的竖直边缘更加突出,常用于场景理解和文字识别任务中。

3	0	-3
10	0	10
3	0	-3

增强型边缘检测核

中心增强型拉普拉斯核：与经典的拉普拉斯核相比，这种核通过增加中心像素的权重，在边缘检测的同时，保留了更多的中心信息，有助于在边缘增强的同时，避免图像的过度模糊。

0	-1	0
-1	5	-1
0	-1	0

中心增强型拉普拉斯核

全方向性增强核：综合型的边缘增强滤波器，通过提高中心像素权重并减小周围像素影响，对全方向的边缘信息具有较强的检测能力，适用需要同时增强水平、垂直和对角线边缘的场景。

-1	-1	-1
-1	9	-1
-1	-1	-1

全方向性增强核

平滑与噪声增强核：结合平滑和增强的功能，通过对中心像素施加较大负权重，同时对周围像素赋予正权重，抑制背景平滑部分,增强局部突变信息，适用检测图像中的异常区域或噪声点。

1	1	1
1	-7	1
1	1	1

平滑与噪声增强核

图5 实验核函数选取

张鲖沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第一部分　实验预处理

提取分类特征解读

拉普拉斯核：残差分布在接近 0 的位置形成非常集中的高峰，表明该核提取的主要是低频平滑特性，大部分像素之间的差异非常小。这说明仅有少量边缘或剧烈变化的区域被捕捉到。

增强型边缘检测核：残差分布呈现较宽的对称形状，频率从中心向两侧逐渐递减。这表明增强型边缘检测核捕捉到了更多的边缘特征，尤其是与图像边缘相关的像素差异得到了较好的放大，同时保持了一定的对称性。

中心增强型拉普拉斯核：残差的分布呈现偏移特性，接近 0 的左侧值较少，而右侧残差频率显著增加。这表明中心增强型拉普拉斯核更加偏向于捕捉较大的正方向变化，能够突出较大的特征变化，但也可能对平滑区域有更弱的表现。

全方向性增强核：残差分布呈现显著的偏右趋势，右侧较高频率的残差值说明全方向性增强核在捕捉剧烈边缘变化方面有显著效果。这种不对称分布反映了该核倾向于增强正向变化的信息，可能对全局均衡性有一定影响。

平滑与噪声增强核：第五张图的残差分布与图4类似，但右侧的分布更加集中且高频。这表明平滑与噪声增强核进一步强调了图像中的局部突变区域，对剧烈变化的特征有更高的敏感性，但可能会对细节的噪声产生放大作用。

比较五个核函数的残差分布，可以看出每种核在捕捉图像特征上的侧重点不同。需要选择合适的核函数应结合实际任务需求，权衡边缘增强与噪声放大之间的平衡。

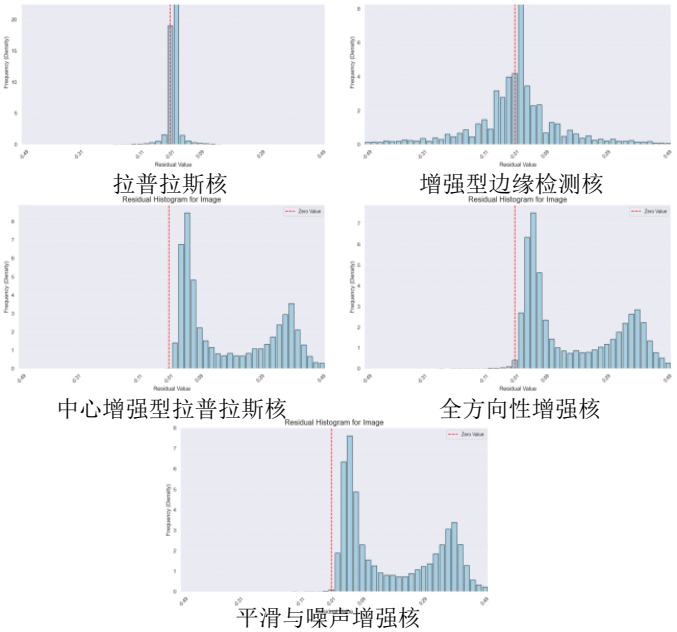


图6 5个核函数残差分布图

提取残差特征与PCA降维解读

图像中红色为标签1，蓝色为标签0

本部分进行了提取残差特征与PCA降维。图7是PCA降维前的气泡图，图8是PCA降维后的气泡图。通过图像对比，可以得出以下内容：

类别区分能力：未降维特征分布显示显著的类别重叠，而降维后部分核函数的特征在二维空间中形成了更清晰的类别分离。

特征集中性：未降维特征分布在多个方向上均有较大范围，而降维后的特征分布更集中。

降维后，特征分布得以简化，同时在低维空间中更好地突出类别间的差异性。对于分类任务，降维不仅有助于降低计算复杂度，还能通过去除冗余信息提升模型对特征的理解和利用效率。

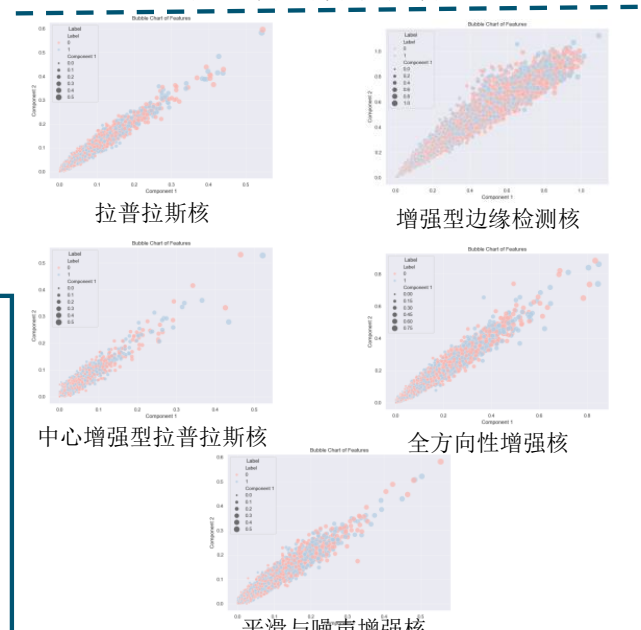


图7 5个核函数PCA降维前的气泡图

拉普拉斯核：数据点主要集中分布在一个密集扇形区域内，红色和蓝色的样本之间具有一定的重叠。这表明拉普拉斯核能捕捉部分边缘特征，但在分离两类样本时效果有限，类别间可分性较低。

增强型边缘检测核：数据点呈现出明显的线性分布趋势，红色和蓝色的样本之间仍然存在重叠，但类别分布的边界更清晰，表明该核对于提取方向性边缘信息效果较好。

中心增强型拉普拉斯核：数据点呈现出密集“U”型分布，红色和蓝色的样本在某些区域具有较好的分离效果，但在交叉区域仍有较大重叠，表明该核同时保留了一些全局信息。

全方向性增强核：数据点形成一个更宽的扇形分布，红蓝样本点分布更均匀，但类别间的交叉区域较多，表明该核由于提取特征的广泛性，其在分离细微特征上的表现有限。

平滑与噪声增强核：数据点分布与全方向性增强核相似，但红蓝样本点的边界更模糊，类别间的交叠更加明显。该核在抑制背景平滑部分的同时增强局部突变信息，但在类别间的可分性上表现稍弱。

图8 5个核函数PCA降维后的气泡图

张鳃沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第二部分　模型训练

Logistic Regression (LR) 模型

模型介绍

Logistic Regression（逻辑回归）是分类任务中一种经典的线性模型，通常用于二分类或多分类问题。它的基本思想是通过一个线性函数计算输入特征的线性组合，然后利用 sigmoid 等函数将结果映射到 [0, 1] 的概率空间。逻辑回归的核心目标是找到最佳的参数，使得模型能够准确预测样本属于某个类别的概率。其损失函数通常采用对数似然损失（Log Loss），优化方法可以使用梯度下降、随机梯度下降（SGD）等。逻辑回归因其简单性、可解释性以及较快的计算速度而在许多场景下被广泛应用。

基于库函数的实现

代码中的 train_lr 函数利用库函数直接调用逻辑回归模型，并通过参数 C 控制正则化强度，learning_rate 和 epochs 作为优化的超参数。每轮训练时调用 fit 方法更新模型的参数，并计算训练集上的对数损失值，记录到 loss_list 中。性能评估：在训练完成后，通过模型在测试集上的预测结果计算分类性能指标，包括准确率、精确率、召回率和 F1 分数。

通过网格搜索的方式尝试不同的超参数组合（如 C 和 learning_rate），记录每组参数的性能并找到最佳组合。

表1 LR模型训练参数

C	0.0001	0.001	0.01	0.1	1
learning_rate	0.0001	0.001	0.01	0.1	1

Code1 自行实现LR模型代码

```
class LogisticRegressionModel(nn.Module):
    """
    Logistic Regression 模型定义
    """
    def __init__(self, input_dim):
        super(LogisticRegressionModel, self).__init__()
        self.linear = nn.Linear(input_dim, 1)  # 定义线性层，输出维度为 1
    def forward(self, x):
        # 使用 Sigmoid 激活函数，将输出映射到 [0, 1]
        return torch.sigmoid(self.linear(x))
```

自行实现 LR模型

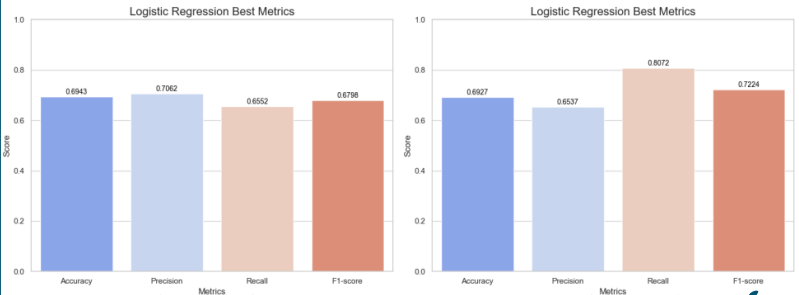
相比直接调用库函数，自行实现逻辑回归（使用 PyTorch）提供了更大的灵活性，尤其在训练流程和模型设计上。

LogisticRegressionModel 定义了一个简单的线性模型，使用单层线性变换和 sigmoid 激活函数完成逻辑回归的核心部分。标签数据需要调整为列向量（view(-1, 1)），以匹配模型的输出形状。损失函数采用二分类交叉熵损失 nn.BCELoss，能够很好地处理逻辑回归任务的概率输出。优化器使用随机梯度下降（SGD）并根据学习率参数 learning_rate 更新模型权重。每轮训练包括以下步骤：

1. 前向传播：通过模型计算预测概率。
2. 损失计算：将预测概率和真实标签传入损失函数，得到当前轮次的损失值。
3. 反向传播：计算损失对模型参数的梯度。
4. 参数更新：利用优化器根据梯度更新模型参数。
5. 评估过程：在训练结束后，模型进入评估模式，使用测试数据计算预测概率，并将其二值化（概率大于 0.5 预测为 1，否则为 0）。

表2 增强型边缘检测核PCA降维前 评价指标表

Library Implementation					Custom Implementation				
C	Learning Rate	Accuracy	Precision	Recall	C	Learning Rate	Accuracy	Precision	Recall
0.0001	0.0001	0.6415	0.6944	0.4932	0.0001	0.0001	0.4893	0.4844	0.4856
0.0001	0.001	0.6415	0.6944	0.4932	0.0001	0.001	0.6532	0.6049	0.8647
0.0001	0.01	0.6415	0.6944	0.4932	0.0001	0.01	0.6885	0.6498	0.8046
0.0001	0.1	0.6415	0.6944	0.4932	0.0001	0.1	0.6538	0.7463	0.4558
0.0001	1	0.6415	0.6944	0.4932	0.0001	1	0.5690	0.5349	0.9934
0.001	**0.0001**	**0.6943**	**0.7062**	**0.6552**	0.001	0.0001	0.5485	0.5365	0.6487
0.001	0.001	0.6943	0.7062	0.6552	0.001	0.001	0.6607	0.6230	0.7976
0.001	0.01	0.6943	0.7062	0.6552	0.001	0.01	0.6895	0.6503	0.8072
0.001	0.1	0.6943	0.7062	0.6552	0.001	0.1	0.6530	0.7452	0.4548
0.001	1	0.6943	0.7062	0.6552	0.001	1	0.5595	0.7758	0.1555
0.01	0.0001	0.6930	0.7019	0.6608	0.01	0.0001	0.5272	0.5176	0.6683
0.01	0.001	0.6930	0.7019	0.6608	0.01	0.001	0.6428	0.6026	0.8183
0.01	0.01	0.6930	0.7019	0.6608	**0.01**	**0.01**	**0.6927**	**0.6537**	**0.8072**
0.01	0.1	0.6930	0.7019	0.6608	0.01	0.1	0.6623	0.6014	0.9430
0.01	1	0.6930	0.7019	0.6608	0.01	1	0.5727	0.7857	0.1888
0.1	0.0001	0.6935	0.6990	0.6694	0.1	0.0001	0.5433	0.5992	0.2347
0.1	0.001	0.6935	0.6990	0.6694	0.1	0.001	0.6595	0.6188	0.8137
0.1	0.01	0.6935	0.6990	0.6694	0.1	0.01	0.6887	0.6501	0.8046
0.1	0.1	0.6935	0.6990	0.6694	0.1	0.1	0.6520	0.7448	0.4523
0.1	1	0.6935	0.6990	0.6694	0.1	1	0.4953	0.4952	1.0000
1	0.0001	0.6840	0.6866	0.6658	1	0.0001	0.5895	0.6258	0.4255
1	0.001	0.6840	0.6866	0.6658	1	0.001	0.6558	0.6049	0.8794
1	0.01	0.6840	0.6866	0.6658	1	0.01	0.6890	0.6511	0.8016
1	0.1	0.6840	0.6866	0.6658	1	0.1	0.6617	0.6008	0.9445
1	1	0.6840	0.6866	0.6658	1	1	0.4953	0.4952	1.0000



a) 库函数的实现　　b) 自行实现 LR模型

图9 增强型边缘检测核 部分评价指标图

张鲴沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第二部分　模型训练

支持向量机 (SVM) 模型

模型介绍

支持向量机（Support Vector Machine, SVM）是一种经典的监督学习算法，主要用于分类和回归任务。SVM 的核心思想是通过寻找一个最佳超平面，将不同类别的样本分隔开，并最大化分类间的间隔（margin）。对于线性不可分的数据，SVM 通过核技巧（Kernel Trick）将数据映射到更高维的特征空间，在该空间中实现线性分隔。SVM 具有强大的分类能力，特别适用于高维数据和小样本问题。SVM 的优化目标是同时最小化分类误差和模型复杂度，受正则化参数 C 的控制：C 值较大时，更注重降低误分类率，可能导致过拟合；C 值较小时，允许一定程度的误分类，提升模型的泛化能力。

基于库函数的实现

实现方法代码中实现了基于 Scikit-learn 的 SVM 模型训练与实验调优。

模型训练函数 train_svmtrain_svm 函数接受正则化参数 C 和核函数类型 kernel 作为输入，通过调用 Scikit-learn 的 SVC实现模型训练与预测，使用 predict 方法预测测试集标签，调用 evaluate_model 计算并返回分类评估指标（如准确率、精确率、召回率和 F1 分数）。

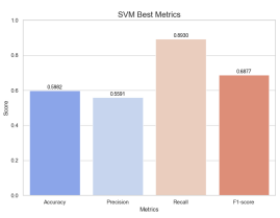
run_svm_experiments 函数通过遍历参数网格 param_grid 中的 C 和 kernel 组合，依次调用 train_svm 进行实验：通过日志文件记录每次实验的参数组合及其对应的评估指标；对比实验结果，选出最佳参数组合（基于最高准确率）；使用 plot_best_metrics 绘制最佳模型的指标柱状图。

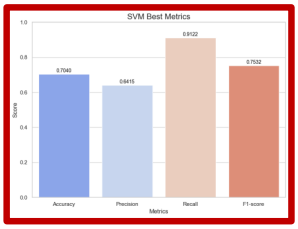
表3 SVM模型训练参数

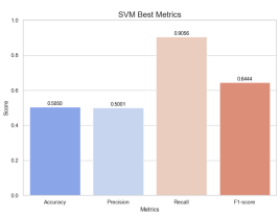
C	0.001	0.01	0.1	1
kernel	linear	poly	rbf	sigmoid

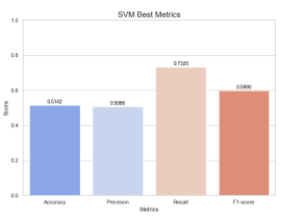
表4 增强型边缘检测核 SVM评价指标表

PCA Not Applied					PCA Applied				
C	Kernel	Accuracy	Precision	Recall	C	Kernel	Accuracy	Precision	Recall
0.001	linear	0.6365	0.5850	0.9152	0.001	linear	0.6135	0.5794	0.8011
0.001	poly	0.5767	0.5407	0.9662	0.001	poly	0.5333	0.5151	0.9793
0.001	rbf	0.5393	0.5182	0.9909	0.001	rbf	0.5610	0.5312	0.9672
0.001	sigmoid	0.4953	0.4952	1.0000	0.001	sigmoid	0.5513	0.5342	0.7340
0.01	linear	0.6560	0.6014	0.9056	0.01	linear	0.6180	0.5877	0.7663
0.01	poly	0.6165	0.5687	0.9334	0.01	poly	0.5420	0.5199	0.9803
0.01	rbf	0.6175	0.5700	0.9273	0.01	rbf	0.5992	0.5595	0.8975
0.01	sigmoid	0.5675	0.5345	0.9823	0.01	sigmoid	0.5405	0.5389	0.4997
0.1	linear	0.6697	0.6129	0.9046	0.1	linear	0.6172	0.5875	0.7627
0.1	poly	0.6617	0.6017	0.9374	0.1	poly	0.5480	0.5233	0.9803
0.1	rbf	0.6635	0.6076	0.9051	0.1	rbf	0.6075	0.5657	0.8935
0.1	sigmoid	0.5252	0.5205	0.5250	0.1	sigmoid	0.5340	0.5289	0.5396
1	linear	0.6820	0.6281	0.8773	1	linear	0.6172	0.5875	0.7627
1	poly	0.6993	0.6353	0.9223	1	poly	0.5495	0.5242	0.9798
1	**rbf**	**0.7040**	**0.6415**	**0.9122**	1	rbf	0.6085	0.5659	0.8995
1	sigmoid	0.5235	0.5191	0.5144	1	sigmoid	0.5333	0.5282	0.5396


拉普拉斯核


增强型边缘检测核（**最优**）


中心增强型拉普拉斯核


全方向性增强核

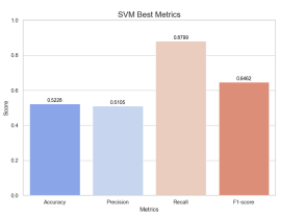
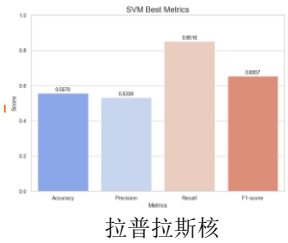
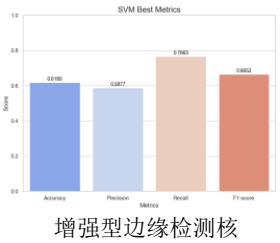
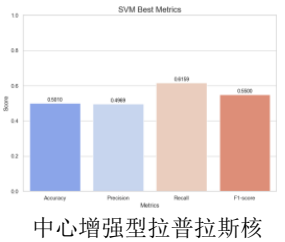
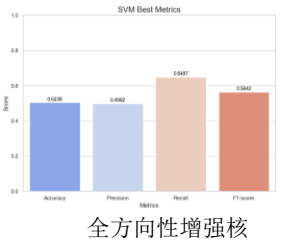

平滑与噪声增强核

图10 5个核函数PCA降维前SVM评价指标


拉普拉斯核


增强型边缘检测核


中心增强型拉普拉斯核


全方向性增强核


平滑与噪声增强核

图11 5个核函数PCA降维后SVM评价指标

张鲖沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第二部分　模型训练

Linear Discriminant Analysis (LDA) 模型

模型介绍

Linear Discriminant Analysis（LDA，线性判别分析）是一种监督学习算法，主要用于分类任务。其目标是通过寻找一个线性投影，将数据投影到低维空间，同时最大化类间的分离度和最小化类内的散布。LDA 的核心思想是找到一个变换矩阵，使得不同类别的样本在投影后的空间中尽可能分离，同时类内样本尽可能紧凑。该算法特别适合用于降维和分类任务，尤其是在数据分布呈高斯分布且类别均匀分布的情况下效果最佳。LDA 同时考虑了类内方差和类间方差，因此能够在保留数据结构的同时提高分类性能，并且在样本量有限或高维数据场景下，LDA 能有效避免维度灾难并提高模型的泛化能力。

基于库函数的实现

实现方法代码中实现了基于 Scikit-learn 的 LDA 模型训练与实验调优。

train_lda 接受两个关键参数：solver LDA 求解方法；shrinkage：正则化参数。函数会初始化 LDA 模型，根据传入的 solver 和 shrinkage 参数配置，使用 fit 方法在训练集上训练模型，使用 predict 方法在测试集上进行预测，调用 evaluate_model 函数计算分类评估指标（如准确率、精确率、召回率和 F1 分数）。

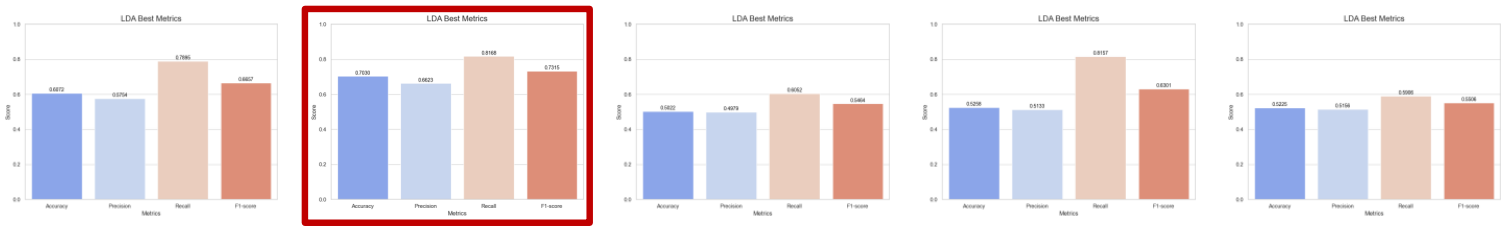
run_lda_experiments 该函数通过遍历参数网格 param_grid，对 LDA 的超参数进行调优，遍历所有可能的参数组合，对每组参数调用 train_lda 训练模型并记录评估结果，将每次实验的参数和对应评估指标保存到文件，比较实验结果，选出分类性能最佳的参数组合（基于准确率）并可视化最佳模型的评估指标。

表5 LDA模型训练参数

solver	svd	lsqr	eigen	
shrinkage	None	auto	0.1	0.5

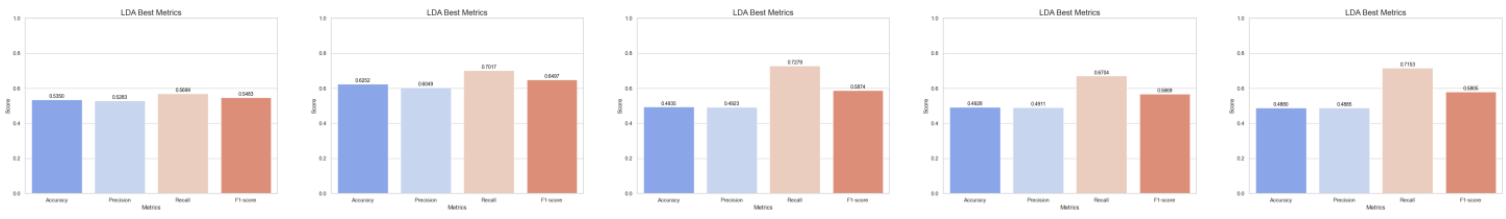
表6 增强型边缘检测核 LDA评价指标表

PCA Not Applied					PCA Applied				
Solver	Shrinkage	Accuracy	Precision	Recall	Solver	Shrinkage	Accuracy	Precision	Recall
svd	None	0.6850	0.6600	0.7506	svd	None	0.6250	0.6047	0.7012
svd	auto	*Failed*	*Failed*	*Failed*	svd	auto	*Failed*	*Failed*	*Failed*
svd	0.1	*Failed*	*Failed*	*Failed*	svd	0.1	*Failed*	*Failed*	*Failed*
svd	0.5	*Failed*	*Failed*	*Failed*	svd	0.5	*Failed*	*Failed*	*Failed*
lsqr	None	0.6723	0.7050	0.5815	lsqr	None	0.6250	0.6047	0.7012
lsqr	auto	0.6855	0.6602	0.7521	lsqr	auto	0.6252	0.6049	0.7017
lsqr	**0.1**	**0.7030**	**0.6623**	**0.8168**	lsqr	0.1	0.6125	0.5898	0.7148
lsqr	0.5	0.6795	0.6308	0.8506	lsqr	0.5	0.5753	0.5522	0.7527
eigen	None	0.4953	0.4952	1.0000	eigen	None	0.6250	0.6047	0.7012
eigen	auto	0.6855	0.6602	0.7521	eigen	auto	0.6252	0.6049	0.7017
eigen	**0.1**	**0.7030**	**0.6623**	**0.8168**	eigen	0.1	0.6125	0.5898	0.7148
eigen	0.5	0.6795	0.6308	0.8506	eigen	0.5	0.5753	0.5522	0.7527



拉普拉斯核　增强型边缘检测核（**最优**）　中心增强型拉普拉斯核　全方向性增强核　平滑与噪声增强核

图12 5个核函数PCA降维前LDA评价指标



拉普拉斯核　增强型边缘检测核　中心增强型拉普拉斯核　全方向性增强核　平滑与噪声增强核

图13 5个核函数PCA降维后LDA评价指标

张鳃沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第二部分　模型训练

卷积神经网络 (CNN) 模型

模型介绍

卷积神经网络（Convolutional Neural Network, CNN）是一种专为处理图像数据设计的深度学习模型。CNN 的核心特点是通过卷积操作提取图像的局部特征，同时通过池化操作降低数据维度，从而捕捉图像中的重要模式。CNN 具有较强的特征提取能力，尤其在处理二维图像数据时具有显著优势。其参数共享机制减少了模型参数量，同时增强了模型对平移等变换的鲁棒性。

CNN 通常由多个卷积层、池化层和全连接层组成：

- 卷积层：通过卷积核（过滤器）提取图像的局部特征，如边缘、纹理等。
- 池化层：通过下采样（如最大池化）减少特征图的大小，降低计算复杂度，同时保留主要特征。
- 全连接层：将卷积层提取的特征映射到分类任务中的输出类别。

基于库函数的实现

实现方法代码中实现了基于 Scikit-learn 的 LDA 模型训练与实验调优。

train_lda 是基于 Scikit-learn 的 LDA 实现函数。train_lda 函数会初始化 LDA 模型，根据传入的 solver 和 shrinkage 参数配置，使用 fit 方法在训练集上训练模型。使用 predict 方法在测试集上进行预测。调用 evaluate_model 函数计算分类评估指标。

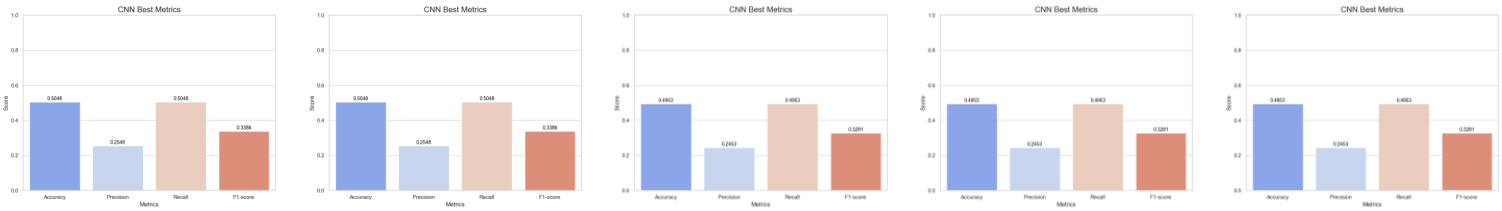
run_lda_experiments 函数通过遍历参数网格 param_grid，对 LDA 的超参数（solver 和 shrinkage）进行调优：遍历所有可能的参数组合（如不同求解方法和收缩参数），对每组参数调用 train_lda 训练模型并记录评估结果，记录日志，将每次实验的参数和对应评估指标保存到文件，比较实验结果，选出分类性能最佳的参数组合（基于准确率）并可视化最佳模型的评估指标。

Code2 CNN模型代码

```
class CNNModel(nn.Module):
    def __init__(self, input_channels=1, num_classes=2):
        super(CNNModel, self).__init__()
        self.conv1 = nn.Conv2d(input_channels, 16, kernel_size=3, stride=1, padding=1) # 卷积层 1
        self.conv2 = nn.Conv2d(16, 32, kernel_size=3, stride=1, padding=1) # 卷积层 2
        self.fc1 = nn.Linear(32 * 8 * 8, 128) # 全连接层 1
        self.fc2 = nn.Linear(128, num_classes) # 全连接层 2
    def forward(self, x):
        x = nn.ReLU()(self.conv1(x)) # 卷积层 1 + ReLU
        x = nn.MaxPool2d(kernel_size=2, stride=2)(x) # 最大池化层 1
        x = nn.ReLU()(self.conv2(x)) # 卷积层 2 + ReLU
        x = nn.MaxPool2d(kernel_size=2, stride=2)(x) # 最大池化层 2
        x = x.view(x.size(0), -1) # 展平
        x = nn.ReLU()(self.fc1(x)) # 全连接层 1 + ReLU
        x = self.fc2(x) # 全连接层 2
        return x
```

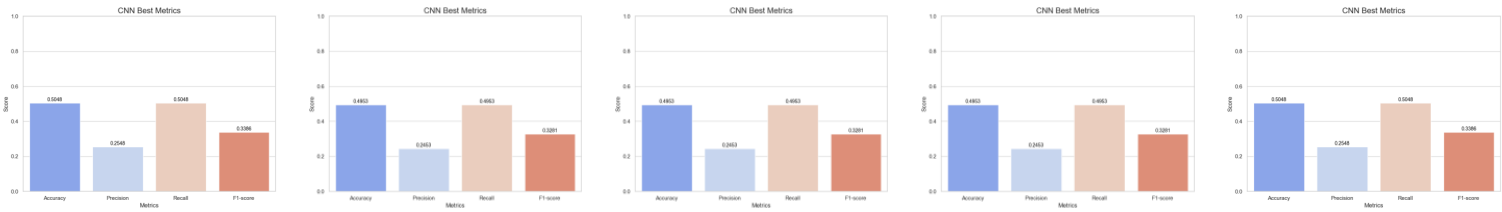
表7 5种核函数 CNN评价指标表

PCA Not Applied					PCA Applied				
Kernel	Accuracy	Precision	Recall	F1-score	Kernel	Accuracy	Precision	Recall	F1-score
Kernel 1	0.5048	0.2548	0.5048	0.3386	Kernel 1	0.5048	0.2548	0.5048	0.3386
Kernel 2	0.5048	0.2548	0.5048	0.3386	Kernel 2	0.4953	0.2453	0.4953	0.3281
Kernel 3	0.4953	0.2453	0.4953	0.3281	Kernel 3	0.4953	0.2453	0.4953	0.3281
Kernel 4	0.4953	0.2453	0.4953	0.3281	Kernel 4	0.4953	0.2453	0.4953	0.3281
Kernel 5	0.4953	0.2453	0.4953	0.3281	Kernel 5	0.5048	0.2548	0.5048	0.3386



拉普拉斯核　增强型边缘检测核　中心增强型拉普拉斯核　全方向性增强核　平滑与噪声增强核

图14 5个核函数PCA降维前CNN评价指标



拉普拉斯核　增强型边缘检测核　中心增强型拉普拉斯核　全方向性增强核　平滑与噪声增强核

图15 5个核函数PCA降维后CNN评价指标

张鲴沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第三部分　结果分析

实验结果

表8 5种模型最优参数对比

模型	最佳核函数	最佳参数	Accuracy	Precision	Recall	F1-score
逻辑回归（库函数）	增强型边缘检测核	*C=0.001, learning_rate=0.0001*	0.6943	0.7062	0.6552	0.6798
逻辑回归（自实现）	增强型边缘检测核	*C=0.01, learning_rate=0.01*	0.6927	0.6537	0.8072	0.7224
支持向量机（SVM）	**增强型边缘检测核**	**C=1, kernel=rbf**	**0.7040**	**0.6415**	**0.9122**	**0.7532**
线性判别分析（LDA）	增强型边缘检测核	*solver=lsqr, shrinkage=0.1*	0.7030	0.6623	0.8168	0.7315
卷积神经网络（CNN）	拉普拉斯核/增强型边缘检测核	*2-layer network*	0.5048	0.2548	0.5048	0.3386

注：其中LR\SVM\LDA方法不进行降维结果最好，CNN对于是否降维在数据结果上无关联

从实验结果可以看出，**增强型边缘检测核在五种模型中表现最优**，尤其在支持向量机（SVM）和线性判别分析（LDA）模型中提升明显。**SVM 模型整体表现最佳**，其最高准确率达到了 0.7040，F1-score 为 0.7530，这表明 SVM 在捕捉数据特征和分类能力上最为出色。相比之下，**CNN 模型表现最差**，其所有指标均远低于其他方法。

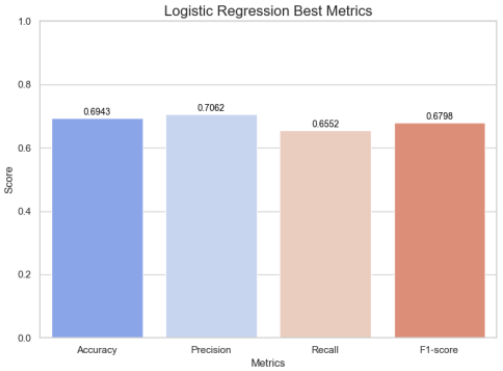


图16 LR模型库实现最优结果评价

结果分析

1. 为什么增强型边缘检测核效果整体比其他好?

增强型边缘检测核**通过强调图像的边缘和结构化特征**来提取更多显著的特征，这种特性使其在**捕捉数据的高频信息和边界信息方面具有优势**。这些信息对分类任务（尤其是图像分类）至关重要，因为边缘和结构通常是区分类别的关键特征。相比其他核函数（如拉普拉斯核或平滑与噪声增强核），**增强型边缘检测核对图像中特征的敏感性更强，同时避免了对噪声信息的过度放大。**

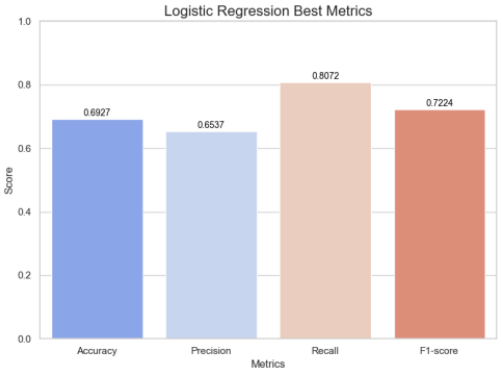


图17 LR自行实现最优结果评价

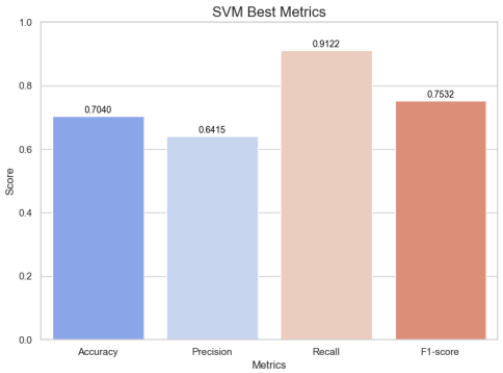


图18 SVM模型最优结果评价

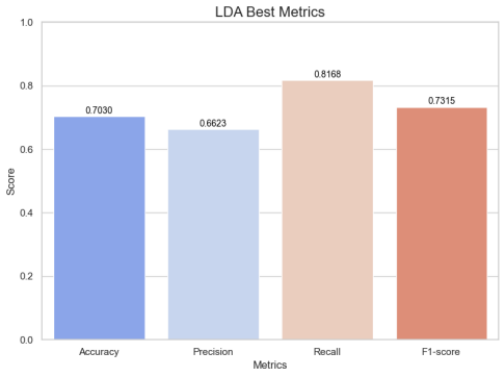


图19 LDA模型实现最优结果评价

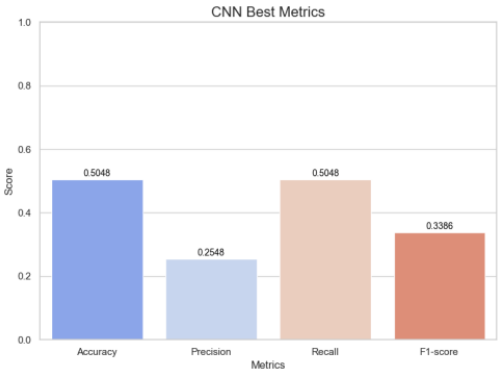


图20 CNN模型实现最优结果评价

张鲴沣 北京交通大学计算机科学与技术学院 AI2201 班 22281052@bjtu.edu.cn

第一次大作业 实验报告

第四部分　结论感想

结果分析

2. 为什么 SVM 效果最优?

SVM 在处理小样本和高维数据方面具有优势，其通过寻找最优分类超平面最大化类间间隔，从而提升分类性能。在实验中，SVM 结合增强型边缘检测核提取的高质量特征，并使用 RBF 核捕捉非线性关系，进一步提高了分类能力。此外，SVM 的正则化参数 C 和核函数灵活性使其能够适应不同的数据分布，这些特性使得 SVM 成为最佳模型。

3. 为什么 CNN 效果最差?

- **数据规模不足**：CNN 需要大量数据进行训练，而实验数据量有限，可能导致模型未能充分学习特征。
- **模型复杂性**：CNN 的参数较多（如卷积核、全连接层权重），在小样本情况下容易过拟合。
- **特征提取局限**：实验中使用了简单的 2 层卷积结构，对于复杂的图像模式可能不足以提取有效特征。相比其他模型（如 SVM 和 LDA），CNN 的效果受限于数据规模和模型设计。

4. 为什么 SVM 中 RBF 核效果最优?

RBF 核（径向基核函数）**能够很好地捕捉数据的非线性关系**，其通过计算样本间的高斯相似性，将数据映射到高维空间，使得原本线性不可分的样本变得线性可分。在实验中，图像特征往往具有复杂的分布模式，RBF 核可以有效捕捉这些非线性分布关系，从而提升分类性能。相比其他核（如线性核或多项式核），RBF 核在适应复杂数据模式方面表现更为稳定。

5.为什么 LR、SVM、LDA 方法不进行降维结果最好，而 CNN 对降维无明显影响?

LR、SVM、LDA这些模型本质上对原始特征具有较强的利用能力，降维后可能会丢失部分关键信息，尤其在 PCA 降维中，一些非线性特征可能被压缩，从而影响分类性能。

CNN 本身具有强大的自动特征提取能力，通过卷积操作能够从原始输入中提取特征。因此降维对 CNN 的影响较小。降维后的输入数据仍然能通过卷积操作重新提取有效特征。

6. 为什么自己实现的逻辑回归比库函数实现精度低?

主要因为优化算法的差异。**库函数使用高级优化器（如 lbfgs），而自行实现可能采用简单的梯度下降。**此外，库函数在**数值稳定性**和**正则化处理**（如默认使用 L2 正则化）方面表现更优，同时提供更多默认优化参数。而自己实现可能存在超参数调优不足或代码细节不完善的问题，导致模型性能下降。

实验感想

本次实验通过对LR、SVM、LDA和CNN等模型的实现与优化，探讨了不同模型在分类任务中的表现差异。实验结果表明，**增强型边缘检测核在提取特征和提升分类性能方面具有显著优势**，尤其在传统机器学习模型中（如 SVM 和 LDA）表现突出。**SVM 的非线性建模能力和 RBF 核的高效特征映射，成为此次实验的最佳模型。**通过调整 SVM 的正则化参数 C 和核函数类型，不仅能够捕捉复杂的特征关系，还能实现优良的分类性能。这一过程让我更加深刻理解了模型与数据特征的匹配性对于实验结果的重要性，同时也认识到参数调优和核函数选择在提升模型性能中的关键作用。相比之下，CNN 模型的表现相对较差，主要原因在于数据规模的限制，而实验数据较少，导致模型未能充分训练。

此外，实验还认识到降维处理对不同模型的影响。传统机器学习模型（如 LR 和 SVM）在高维数据上表现更优，而 CNN 对降维的敏感性较低。通过对比库函数与自行实现的逻辑回归模型，也让我体会到库函数在优化算法、数值稳定性和正则化处理方面的优势。整体来看，这次实验不仅强化了我对各模型原理的理解，也让我体会到数据预处理、参数调优和模型选择在机器学习任务中的重要性。