分类号_____ 密级 _____
U D C_____ 编号 _____

硕 士 学 位 论 文

论文题目 鲁棒的高置信度样本选择方法在 Self-training 算法中的研究

研 究 生 姓 名：段会雨

指导教师姓名、职称：李 兵 教授

学 科、专 业 名 称：管理科学与工程

研 究 方 向：信息管理与信息系统

提 交 日 期：2024 年 5 月 31 日

独创性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：段含雨 签字日期：2024.5.31

导师签名：李乐 签字日期：2024.5.31

关于论文使用授权的说明

本人完全了解学校关于保留、使用学位论文的各项规定，同意（选择“同意”/“不同意”）以下事项：

1.学校有权保留本论文的复印件和磁盘，允许论文被查阅和借阅，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文；

2.学校有权将本人的学位论文提交至清华大学“中国学术期刊（光盘版）电子杂志社”用于出版和编入 CNKI《中国知识资源总库》或其他同类数据库，传播本学位论文的全部或部分内容。

学位论文作者签名：段含雨 签字日期：2024.5.31

导师签名：李乐 签字日期：2024.5.31

Robust High-Confidence Sample Selection Method to Self-training Algorithms

Candidate : Huiyu Duan

Supervisor: Bing Li

摘 要

Self-training 是半监督学习众所周知的一个的框架，其利用少量的有标签数据和大量的无标签数据训练分类器。如何从大量的无标签样本中筛选出高置信度样本加入训练集是 self-training 算法关键的一步，在迭代训练过程中如果使用误分类的高置信度样本，分类错误将被放大。因此，本文提出两种提升高置信度样本选取的 self-training 算法：

（1）基于对 self-training 算法的研究，本文发现许多现有的方法都是基于欧几里得距离计算样本间的距离，这不适用于分布复杂的数据。此外，很多现有算法不能充分利用数据的空间结构挖掘重要信息。因此，本文提出了鲁棒的近亲节点图编辑 self-training 算法（A Robust Self-training Algorithm based on Relative Node Graph，STRNG）。首先，STRNG 使用基于块的不相似性度量算每个样本的密度和峰值，以构建原型树揭示数据的潜在空间结构。然后，本文开始构建每个样本的近亲节点图（RNG）。最后，采用假设检验方法去除高置信度样本中可能标记错误的高置信度样本。本文在 14 个公开真实数据集上与 4 个已有 Self-training 算法进行对比，实验结果验证本文所提出的算法的有效性。

（2）针对现有的一些 self-training 基于局部信息来选择高置信度样本，以及参数依赖的问题。本文提出了一个基于全局信息过滤的 self-training 算法(A Self-training Algorithm for Adaptive Local Neighbor Filtering, STALN)。首先，STALN 结合数据的全局信息与数据的局部信息，自适应地选择无标签样本的局部邻居。其次，利用局部邻居的信息为无标签样本分配标签，并与基分类器预测的标签对比，如果标签一致则将其视为高置信度样本，此方法可以有效识别误标记样本。最后，将这些高置信度样本添加进训练集进行迭代训练。为了验证 STALN 的性能，在 18 个基准数据集与 4 个已有 Self-training 算法进行对比，实验结果验证了 STALN 的有效性。

关键词: Self-training 高置信度样本 局部邻居 全局信息 数据编辑

Abstract

Self-training is a well-known framework for semi-supervised learning, which utilizes a small amount of labeled data and a large amount of unlabeled data to train classifiers. How to select high-confidence samples from a large number of unlabeled samples and add them to the training set is a crucial step for self-training algorithms. In addition, if high-confidence samples with misclassification are used during the iterative training process, classification errors will be amplified. Therefore, we propose two self-training algorithms to improve the selection of high-confidence samples:

(1) Based on research on self training algorithms, many existing self training methods are based on Euclidean distance to measure the relationships between samples, which is not suitable for complex structured datasets. In addition, many existing algorithms cannot fully utilize the spatial structure of the dataset to mine important information. Therefore, a robust self training algorithm (A Robust Self training Algorithm based on Relative Node Graph, STRNG) was proposed. Firstly, STRNG uses block estimation to calculate the density and peak of each sample, in order to construct a prototype tree to reveal the potential spatial structure of the data. Then, we begin to construct a relative node graph (RNG) for each sample. Finally, based on hypothesis testing methods, samples with high confidence that may be labeled incorrectly are removed.

The effectiveness of the proposed algorithm was verified by comparing it with four existing algorithms on 14 publicly available datasets. (2) A self-training algorithm based on global information filtering (A Self training Algorithm for Adaptive Local Neighbor Filtering, STALN) is proposed. Firstly, STALN combines the global and local information of the data to adaptively select local neighbors for unlabeled samples. Secondly, using the information of local neighbors to assign labels to unlabeled samples, and comparing them with the labels predicted by the base classifier. If the labels are consistent, they are considered as high confidence samples. Finally, add these high- confidence samples to the training set for iterative training. The effectiveness of STALN was validated by comparing it with four existing algorithms on 18 publicly available datasets.

Keywords: Self-training; High-confidence samples; Local Neighbors; Global Information; Data Editing

目 录

1. 绪 论

1.1 研究背景及意义

近几十年来，随着互联网技术、计算机和智能手机的出现与普及，以及物联网、云计算、大数据技术等技术的进一步发展，人们收集数据、存储数据和利用海量数据的能力得到了极大的提高。海量的数据能为人们的决策提供数据支撑，但是海量的信息也带来了许多技术上的问题。比如，搜集到的数据中可能包含垃圾信息、冗余信息，另外数据的信息形式也有可能不一致，过多无用信息的干扰必然会造成一些有用知识的丢失等问题。因此，人们迫切希望能更好地利用海量数据，这就需要有效的手段对数据进行分析，发现并提取出数据中被隐藏的有价值的信息，如何从这些数据中发掘其中隐含的有价值信息，发现数据中隐藏的模式、关联和趋势，以帮助组织做出更明智的决策、预测未来趋势、发现新的商机，或者简单地了解数据中的特点是数据处理领域的难点。正是在这种背景下，数据挖掘 (Data mining)[1]技术应运而生。数据挖掘作为一种高效的数据处理技术，其从大量复杂数据中发现并提取出对决策、预测和分析有用信息和模式。通过数据挖掘，可以从海量数据来发现有价值的信息，从而提高业务决策的准确性和效率，获得竞争优势，数据挖掘技术的应用非常广泛，包括市场营销、金融风险管理、医疗保健、电子商务、社交网络分析等领域[4]。例如，超市或购物平台根据每天的交易信息判断客户的购物习惯，为平台提供更加科学进货、商品摆放等指导；为网约车司机提供各个区域各个时间段的打车情况；为大型商场管理人员提供客户画像以及各品类商品的销售情况等。

机器学习(Machine Learning)因其自动的探索数据中的有用信息进行学习从而获得规律，并利用规律对新数据进行预测的能力，而成为数据挖掘的有效途径之一[8]。机器学习分为监督学习[10] (Supervised Learning,SL)、无监督学习[11] (Unsupervised Learning,UL)和半监督学习[13] (Semi-supervised Learning,SSL)三大类。简单的归纳就是，是否有监督，就看输入数据是否有标签(label)。输入数据有标签，则为有监督学习；没标签则为无监督学习；有少量标签则为半监督学习。有监督目标是从有标签的训练数据中学习函数或映射关系，以便能够对新的未标

记数据进行预测或分类。在有监督学习中，模型通过将输入数据和对应的输出标签进行训练，从而学习输入和输出之间的映射关系。其基本特征是训练数据集中包含有标签的样本，也就是每个输入样本都有对应的输出标签。有监督学习通常包括分类[15]（Classification）和回归[16]（Regression）两种主要类型的任务。有监督学习作为机器学习领域常用的数据处理有效方法之一，但要想获得一个有效的监督学习模型需要大量的有标记数据，现实中有标记数据的获得通常是专家标记，因此获得充足的有标记的数据是昂贵的、耗时的，或者根本就不可能。无监督学习的目标是从未标记的数据中发现数据的结构、模式和关系，而无需提供标签或类别信息。在无监督学习中，模型试图从输入数据中学习隐藏的结构。无监督学习通常用于聚类[17](Clustering)、降维[18](Dimensionality Reduction)和关联规则挖掘[19](Association Rule Mining)。但无监督学习因不含任何的数据标注全靠机器自己进行数据规则的探索，因此存在结果不够准确，可能出现较大误差和缺陷；对参数设置和算法选择较为灵敏，需要更多的技巧和经验；很难对生成的结果进行解释，需要人工进行进一步分析的缺点。半监督学习作为一种结合了监督学习和无监督学习的方法，它使用少量已知标签和大量未知标签的数据进行学习数据的分布特性。半监督学习结合了有监督学习和无监督学习方法的优点，利用少量的带标签数据和大量的未带标签数据进行训练，从而在有限的标注数据下实现更好的性能。提高模型的泛化能力和性能，降低数据标准的成本提升学习效率，同时可以在有限的标注下实现更好的预测效果。半监督学习包括聚类[20]、分类[21](Classification)、降维[22]、回归[23]（Regression）四大类。半监督分类作为不需要任何假设通过对少量有标记数据的各个属性进的学习与分析，能够预测未知标签样本的方法，因其简单高效一直以来被广泛的关注。

常用的半监督分类模型有生成式方法[24] (Generative methods)、半监督支持向量机[25] (Semi-supervised Support Vector Machine)、基于图的方法[26]，协同训练[28](Co-training)和 Self-training[29]。而 Self-training 因其不需要特征划分或特定的假设条件而作为一种简单高效的半监督分类方法被广泛的应用在生物医疗、自然语言处理和图像分类等领域。Self-training 过程主要分为两步：1)利用原始有标签数据集训练参数分类器；2)选择无标记样本中的高置信度样本（High-confidence sample）使用初始分类器预测其标签并加入原始有标记数据集训练分类器。以上

两步迭代直到算法收敛。在标准的 Self-training 过程中，模型不断地从无标记样本集中选取一小组最自信的例子加入有标签数据集重新训练分类器。整个 Self-training 过程中既不需要估计和最大化一些后验概率，也不需要足够的条件独立属性，因此相对于其他的半监督模型和标准协同训练更容易使用。但是，由于初始有标记样本的数量较小，初始假设的泛化能力可能较差。因此，选择的高置信度样本集可能包含很多噪声，因为学习器可能会错误地为一些未标记的示例分配标签，并且在每次迭代的训练过程中，这些噪声的积累将损害最终假设的泛化能力。因此，很明显，如果在自我训练过程中特别是在早期的迭代中，能够识别出中错误标记的例子，那么学习到的假设就有望有更好的表现。因此，Self-training 方法获得优质分类器的关键很大程度上在与无标记的高置信度样本的选择，可靠度高的高置信度样本更有可能被分配到正确的标签，一旦选取的高置信度样本点中包含噪声点或异常点，它在扩充有标记样本时如果错误的选择高置信度样本加入训练集，这些错误将传播到下一次迭代会导致输出分类器性能下降。如何解决迭代过程中选取高置信度样本，从而更好的提高 Self-training 的泛化性能具有很重要的现实意义。

在近年来的研究中，Self-training 方法已经得到了广泛的应用和探讨。研究者们着重关注以下几个方面：①Self-training 算法的改进：针对 Self-training 算法中存在的问题，如标记传播错误、样本偏差等，研究者提出了改进的 Self-training 算法，旨在提高模型的性能和稳定性；②领域自适应：Self-training 方法在领域自适应问题中具有潜力，因为可以利用未标记数据来进行领域的适应和泛化；③实际应用：Self-training 在计算机视觉、自然语言处理、推荐系统等领域有着广泛的应用，研究者们试图探索 Self-training 方法在不同应用场景中的效果和特点。总的来说，Self-training 作为半监督学习的一种重要方法，受到了学术界和工业界的广泛关注。通过不断的研究和实践，Self-training 算法在提高模型性能、解决数据稀缺性等方面展现出了巨大的潜力。因此，本论文基于现有 Self-training 技术提出新的半监督分类模型，在提升高置信度样本的选取的基础上进一步的处理高置信样本中的噪声样本点，提高数据应用能力，并通过实验验证算法的有效性。希望本文提出的为算法思想能够为 Self-training 未来的研究提供理论和实践意义。

1.2 国内外研究现状

Agrawala[30]等人提出了使用所有可用信息计算适当概率为无标记样本分配给一个类，然后，在给定该类分配的情况下样本被用于学习分类器。这是最早的将无标签数据使用在有监督学习中进行 Self-training 的思想的提出。在 20 世纪 90年代，使用少量有标记样本和大量无标记样本的半监督学习得到了迅速发展，Merz 等人在 1992 首次提出了半监督学习的概念[31]。同一时期，Yarowsky 等人[32]通过使用少量无标记样本和大量有标记样本进行的自然语言数据分类是 Self-training 方法首次应用到现实的场景，Vapnik 等人[33]提出了转导支持向量机（Transductive Support Machine, TSVM），以及 Blum 等人[34]提出协同训练算法。随着科学技术的发展，人们对机器学习有了更为深入的研究和认识。基于半监督分类框架后来的国内外学者提出了很多优质的算法模型。R. Raina 等人[35]在 Self-taught learning: transfer learning from unlabeled data 一文中强调了利用未标记数据来增强无监督学习模型的能力的 Self-training 框架。

随着科学技术的发展学者们对 Self-training 有了更加深入的理解，进而提出了的更多性能更好的算法。如，李明和周志华提出的 SETRED[36](SETRED: Self-Training with Editing)算法是 Self-training 的一个经典算法。该方法首先使用原始有标记样本中训练分类器，再从无标记样本中抽取部分样本，根据最近邻规则(k-Nearest Neighbors Rule，KNN)[37]，从中选取由分类器标记后的置信度较高的样本，将高置信度样本和原始有标记样本作为潜在训练集构造相关邻近图(Relative Neighbor Graph,RNG)[38]。RNG 内拥有不同样本标记的两个顶点所构成的边为割边, RNG 内与样本点 x_i 有边相连的所有样本点的集合为其近邻集。若某一样本点与其近邻之间存在大量的割边，则将其视为存疑样本点[39]。同样利用割边选取高置信度样本，国外学者王宇[40]等提出了基于最近邻规则和割边的 Self-training 算法 SNNRCE(Self-training Nearest Neighbor Rule using Cut Edges)，SNNRCE 基于最近邻规则构建相对切边邻域图，并计算切边权重，选择没有切边的样本作为高置信度样本，使用切边权重限制每一类的样本数量以避免极端输出。SNNRCE 首先，为每个未标记样本构建了一个基于所有训练样本的邻接图，并对与其相连具有同一类别的的未标记样本进行标记。然后将这些新标记的样本添加到训练样

本并。然后，使用最近邻规则为其进行标签预测，最后，通过统计检验为其进行误标记标签进行修改。以上几步进行迭代直到算法收敛。该方法的主要优点是：（1）在半监督学习的初始阶段，利用相对邻域图进行分类，减少了误差增强；（2）它引入了一种标签修改机制，以获得更好的分类性能。

由于选取的高置信度样本很有可能存在噪声，尤其在多标签分类模型中选取的高置信度样本存在噪声的概率更大，基于去除高置信度样本中噪声样本的策略卫志华[41]等人提出了基于最近邻编辑的半监督多标签图像分类方法 MLSTE(Multi-Label Self-Training with Editing，MLSTE)。MLSTE 算法结合(Edited Nearest Neighbor , ENN)[42]数据编辑技术进行高置信度样本的选取。MLSTE 利用包括标记和未标记每个训练样本进行编辑，在编辑过程为每一个无标记样本保留一个以上的标签，并通过基于类标签分布的标签编辑实现在多标签数据分类。Jafar [43]等人提出了 SDTC(Semi-supervised self-training for decision tree classifiers, SDTC)算法，该算法使用决策树[44]作为基分类器将有标记样本数据集分为正类和负类，然后使用马氏距离[50]计算每个未标记样本与正类 p 和负类样本 q 的平均值之间的距离。以 $w=|p-q|$ 作为样本得分，选择得分最高的前 m 个样本作为高置信度样本加入有训练集训练分类器。吴迪[46]等人提出了基于密度峰值的 Self-training 半监督分类方法 STDP(Self-training semi-supervised classification based on density peaks of data,STDP)。STDP 首先利用 DPC[47](Clustering by fast search and find of density peaks, DPC)计算样本的密度和峰值，通过低密度样本跟随高密度样本的原则发现数据的潜在空间结构。选取有标签样本的无标签“前驱”样本和无标签“后继”样本添加进高置信度样本集并预测其标签。将稿子年度样本集加入训练集训练分类，进行迭代训练直到算法收敛。

甘海涛[48]等人提出了使用聚类分析改进半监督分类的 Self-training 算法 STSFCM(Using clustering analysis to improve semi-supervised classification，STSFCM)，STSFCM 算法将 SSFCM[49](semi-supervised fuzzy c-means,SFCM)算法和 SVM(Support Vector Machine,SVM)[50]进行结合，形成一种半监督分类框架，首先利用 SFCM 发现整个数据集的空间结构和分布，使用 SVM 对有标签样本集进行训练得到初始分类器，然后利用 SFCM 计算得出每个无标签样本对不同类

别的隶属度，选择其中具有较高隶属度的样本点使用初始分类器进行分类，将具有高置信度的无标签数据及其标签添加进有标签样本集中，直到所有无标签样本都被标记，训练所有样本及其标签得到优化后的分类器。在 STDP 的基础上，吴迪[50]等人提出了一种基于数据密度峰值和差分进化的 Self-training 半监督分类算法 STDPDE(A Self-Training Semi-Supervised Classification Algorithm Based on Density Peaks of Data and Differential Evolution, STDPDE), STDPDE 利用差分进化编辑优化 Self-training 过程中有标签数据的“上一个”和“下一个”无标签数据的属性值，将编辑后的数据作为高置信度样本。同样在在 STDP 的基础上，Yang Liu 等人[51]提出 STDPCEW 算法(A self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor, STDPCEW), 文献[52]提出了一种基于密度峰值和自然邻居的局部噪声过滤算法(Self-training method based on density peaks and an extended parameter-free local noise filter, STDPNF)，文献[53]提出了基于密度峰值和自然邻居的 self-training 算法(Self-training method based on density peaks and natural neighbors, STDPNaN)。STDPCEW 和 SETRED 一样利用割边权值统计和零假设检验的方法识别噪声，过滤高置信度样本中可能误分类的问题。同样，SETRED 和 STDPCEW 都需要构建 RNG，但是构造 RNG 的复杂度较高导致 STDPCEW 不适用于大型数据集。STDPNF 引入自然邻居[54]设计局部噪声过滤器。然而，同 STDP 一样，STDPNF 与 STDPCEW 均未考虑数据的全局信息且难以确定适当的截断距离。STDPNaN 将 STDP 与自然邻居相结合的 self-training 算法，并使用集成分类器提升算法的性能。首先，STDPNaN 使用自然邻居来重新定义并无参求得每个样本的局部密度。其次，同 STDP 一样，STDPNaN 使用 DPC 来发现数据的潜在结构进一步探索样本的空间结构，然后选取有标签样本的无标签“前驱”样本和无标签“后继”样本添加进高置信度样本集。虽然 STDPNaN 克服了 STDP 需要指定截断距离的缺陷，然而 STDPNaN 仍然缺少过滤噪声样本的步骤。李俊南[55]等人提出了基于最优路径森林的 Self-training 算法 STOPF (self-training method based on an optimum path forest，STOPF)，STOPF 在整个数据集上构建 OPF(Optimum-path forest，OPF)[56]来发现特征空间的结构和分布，然后利用特征空间的结构和分布选择有标签数据的高置信度的无标签数据赋予标签，将标记后的数据添加进有标签数据集中，迭代训练直到得到优化后的

分类器。文献[49]提出了密度峰值隶属度优化的半监督 self-training 算法（Semi-supervised self-training algorithm for density peak membership optimization, STDPM）。STDPM 利用样本间的距离和密度的关系定义了无标签近亲结点集和密度峰值隶属度矩阵，通过设置密度峰值隶属度阈值来选择高置信度样本。Self-training 是一种常见的半监督学习算法框架。如何选择高阶样本是基于 Self-training 框架的算法的关键步骤。为了减轻噪声数据的影响，研究人员提出了许多方法来提高高阶样本的选择质量。然而现算法时间复杂度基本上不小于$O(n^2)$，其中n表示样本的数量。受到 Ball-k-means[58]算法的启发，文章[59]在保证训练质量的同时提高训练速度，提出了一种基于数据编辑的快速半监督 Self-training 算法(Fast semi-supervised self-training algorithm based on data editing，EBSA)，它具有线性的时间复杂度。EBSA 首先使用 Ball-k-means 算法进行聚类并选择的球簇中心，计算每个类簇的类心与类内样本点的欧式距离，以类簇为圆心类内最大距离作为半径画圆。根据圆与圆之间的距离关系将每个类簇包含的样本区域将划分为稳定区域和活动区域。在每次迭代过程中，只有中心之间的距离和距离需要计算活性区域中的中心和样本之间的距离，这减少了计算量并降低了。提出了改进的球簇划分和数据编辑方法结合，通过评价在稳定区域中高置信度样本是否标记错误，然后编辑找到的标记错误采样点过滤高置信度样本中的噪声。

在不断地探索 self-training 高置信度样本的选取的同时，研究者们在深度学习方向对半监督算法也进行了一系列研究。文献[61]提出了一种对抗性学习框架 MetaGAN，MetaGAN 通过引入以任务为条件的对抗性生成器增强分类模型区分真实数据和伪数据的能力，MetaGAN 可以在只有少量标签的分类器学习中学习到清晰的决策边界，适用性广泛。文献[61]提出了一种基于图的半监督模型框架，该框架结合 k 个最近邻图来发掘数据空间特征，然后使用稀疏表示来评估成对的像素是否属于同一类并构造概率矩阵，最后将概率矩阵进行归一化处理，该框架可以解决样本标签稀少的问题。

传统统计学习方法更专注于算法实现背后的数理原理以及可解释性，而随着大数据、云计算等技术的迅速发展，机器学习方法的优势逐渐显现，机器学习方法具有高准确率、高自动化、高效率、高自定义和高规模化等优势。因此，未来将会有越来越多的研究者采用机器学习方法进行半监督分类。

1.3 主要研究内容

本文主要研究的是鲁棒的高置信度样本选择方法在 Self-training 算法研究。提出了近亲节点图编辑 Self-training 学习模型和自适应局部邻居过滤的 self-training 算法。

（1）基本思想是通过基于块的不相似性进行平行分隔建立树，从新构建样本之间的相似性矩阵。在一定条件下，鲁棒的近亲节点图编辑 self-training 算法(A Robust Self-training Algorithm based on Relative Node Graph,STRNG)所提出的新的度量方式可以更好的度量不规则数据集样本之间的关系。然后，利用新的相似性矩阵求解样本点的密度和峰值,并利用有向图原理构建原型树揭示数据集潜在空间结构。并在此基础上通过假设检验进一步过滤可能得噪声点

（2）本文提出自适应局部邻居过滤的 Self-training 算法(A Self-training Algorithm for Adaptive Local Neighbor Filtering, STALN)。STALN 主要包含三步：(1)使用初始的有标记样本训练分类器；(2)STALN 基于数据的全局信息，考虑每个无标记样本与全部有标记样本的距离关系，自适应地找到每个无标记样本的局部邻居；(3)基于局部邻居的信息为无标签样本分配一个标签，并与分类器分配的标签对比，将两者标签一致的无标签样本加入有标签集迭代训练。

最后，将所提出的学习模型应用于公开数据集进行实验。本文将提出算法与在多个数据集上与现有优质算法进行对比实验。实验结果表明，本文所提出的模型在数值指标上都要优于已有模型。

全文主要内容分为五个章节，每个章节的具体内容如下：

第一章为绪论。首先简要介绍了该课题的研究背景及意义，其次概述了 self-training 国内外研究现状，最后叙述了本文的研究安排与本文的创新点。

第二章为基相关工作论述。论述了半监督学习理论，包括图理论、self-training 理论及几种经典的 self-training 学习方法介绍与优缺点总结，以及本文涉及的基分类器、评价指标和数据集的介绍。

第三章为鲁棒的近亲节点图编辑 self-training 算法模型及实验。本章对提的新 self-training 模型思想进行详细的介绍。通过在 14 个真实数据的实验验证了本文所提鲁棒的近亲节点图编辑 self-training 算法模型的有效性。在 14 个数据集上

将所提出模型与 STDP、STDPDE、STDPCEW 及 SETRED 算法进行比较通 10%的有标记样本，不同比例标记样本、噪声以及运行时间等实验结果的分析评估本文所提算法的有效性。实验结果表明，本文所提出的鲁棒的近亲节点图编辑 self-training 算法模型具有较好的分类能力。

第四章为自适应局部邻居过滤的 self-training。本章对所提的模型思想进行详细的介绍在 18 个数据集上将所提出模型与 STDP、STOPF、STDPNaN 及 SETRED 算法进行比较通 10%的有标记样本，不同比例标记样本、噪声以及运行时间等实验结果的分析评估本文所提算法的有效性。实验结果表明，本文所提出的适应局部邻居过滤的 self-training 算法模型具有较好的分类能力。

第五章为总结与展望。主要是对研究的内容以及贡献进行总结，并提出了本文仍有的不足之处和展望未来的研究方向。

1.4 创新点

（1）本文发现许多现有的基于自我训练的框架方法大多不能利用数据集的底层结构信息以选择高置信度样本，即使有一些算法利用数据集的底层结构但使用的方法大都适用于常规数据集。此外，几乎所有的算法都使用欧几里得距离计算样本之间的关系，这不适用于复杂的结构化数据集。对于具有复杂结构的数据集，例如流形数据，现有算法不能充分利用数据集的空间结构进行挖掘重要信息。受图结构和自训练的启发，基于以上问题本文提出了一种鲁棒的近亲节点图编辑 Self-training 算法（STRNG）来解决这些问题。STRNG 将基于块的不相似性度量与原型树结构相结合，运用图结构更好的选取高置信度样本。首先，STRNG 使用基于块的相似性而不是欧几里得距离来度量两本之间关系，并使用原型树来揭示底层数据的空间结构。其次，STRNG 使用假设检验来去除噪声来自高置信度样本集。

STRNG 的主要贡献如下：

- 采用基于块的不相似性度量作为样本间距离度量方式，使得 STRNG 能适用于结构复杂的数据集。

- 在基于块的不相似性度量的基础上本文定义了这一种新颖的数据结构原型树，以揭示数据集的潜在的空间结构，使得利用数据的分布特征选取高置信度样本。
- 此外，本文在原型树的基础上提出了一种新的数据编辑方法（RNGE）来识别和编辑标签错误 RNG 上的数据样本，以过滤标记错误的高置信度样本。

（2）现有的一些 self-training 算法大多存在两个问题：(1)都是基于有标签样本的局部无标签邻居来选择高置信度样本，缺乏对数据全局信息的考虑；(2)针对不同的数据集，算法很困难确定适合的参数。针对传统的 self-training 算法仅仅考虑无标签样本与其邻近有标签样本的关系而忽略了全部有标签样本的信息这一问题，提出了一种考虑全局信息无并且无参的自适应局部邻居过滤的 self-training 算法(A Self-training Algorithm for Adaptive Local Neighbor Filtering, STALN)。STALN 挖掘全部有标签样本的信息，自适应地选择无标签样本的局部有标签邻居。一方面自适应地选择邻居避免不同参数对算法性能的影响，另一方面通过结合数据的全局信息与数据的局部信息，能准确地预测样本的标签。STALN 主要包含三步：(1)使用初始的有标记样本训练分类器；(2)STALN 基于数据的全局信息，考虑每个无标记样本与全部有标记样本的距离关系，自适应地找到每个无标记样本的局部邻居；(3)基于局部邻居的信息为无标签样本分配一个标签，并与分类器分配的标签对比，将两者标签一致的无标签样本加入有标签集迭代训练。STALN 的主要贡献如下：

STALN 的主要贡献如下：

- 本文提出了一种结合数据的全局信息与数据的局部信息自适应地确定无标签样本的局部邻居的方法。
- 本文提出了一种新颖的利用局部邻居进行高置信度样本选择的方法。
- 所提算法能有效地识别并过滤噪声样本，从而提升选择高置信度样本的质量。

2. 理论基础及相关知识

本章主要对对半监督学习、self-training 和图结构的基本原理进行介绍。同时对本文涉及的主要符号、数据集以及涉及到的对比算法进行详细的介绍。为接下来理解提出 self-training 分类模型奠定基础。

2.1 本文涉及符号及说明

本节对本文涉及的重要符号进行简单的介绍，符号描述如表 2.1。

表 2.1 本文涉及的重要符号和说明

符号	说明
$X=\{x_1,\cdots,x_n\}\in\mathbb{R}^{n\times d}$	X 表示包含 n 个样本的 d 维数据集
$Y=\{y_1,\cdots,y_n\}\in\mathbb{R}^{n\times 1}$	Y 表示包含 n 个样本的标签集合
$L=\{(x_1,y_1),\cdots,(x_l,y_l)\}$	L 表示包含 l 个有标签样本的数据集
$U=\{x_{l+1},x_{l+2},\cdots,x_n\}$	U 表示包含 $n-l$ 个无标签样本的数据集
S	高置信度样本集
$LNN(x_i)$	表示样本 x_i 的局部有标签样本邻居
$Mode(\bullet)$	计算集合的众数

2.2 半监督学习理论

传统的机器学习技术分为无监督学习和监督学习，无监督学习只利用未标记的样本集，而监督学习则只利用标记的样本集进行学习。但在很多实际问题中，只有少量的带有标记的数据，因为对数据进行标记的代价有时很高，而大量的未标记的数据却很容易得到，为了更好的利用无标记样本人们尝试将其与有标价样本一起训练来学习规则，期望能改进分类的性能，因此半监督学习应运而生。无类标记样本在学习建模中能发挥作用的根本原因在于它们和有类标记样本都是独

立同分布地采样于相同的数据源。半监督学习避免了数据和资源的浪费,同时解决了有监督学习的模型泛化能力不强和无监督学习的模型不精确等问题。半监督学习有两个样本集,一个有标记样本集,一个没有标记样本集.分别记作 $Lable=\{(x_i,y_i\},Unlabled=\{(x_i)\}$，并且数量上 $L<<U$ 。一般而言,半监督学习侧重于在有监督的分类算法中加入无标记样本来实现半监督分类，也就是在有标记样本集中加入无标记样本,增强分类效果。半监督学习模型如图 2.1。

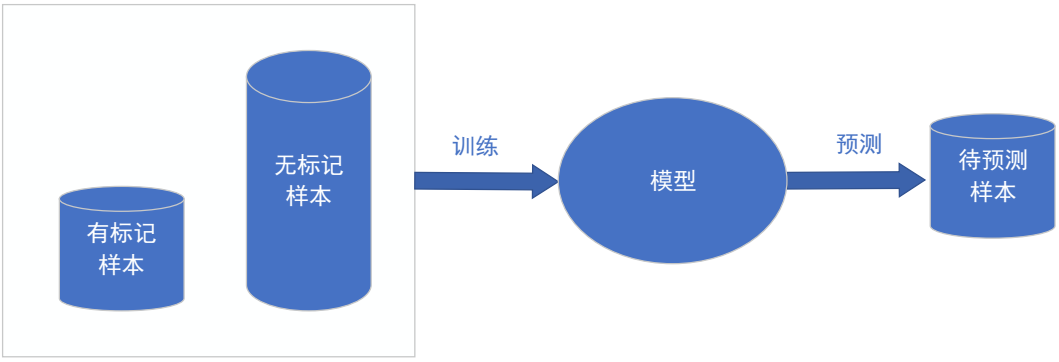


图 2.1 半监督学习模型

半监督学习利用大量的未标记数据，这减少对大量数据进行手动标记的成本。通过利用未标记数据，半监督学习可以提供更广泛和更全面的数据利用，从而改善模型的泛化性能。同时，由于未标记数据包含了更丰富的信息，半监督学习可以帮助模型更好地捕捉数据分布和特征之间的关系，从而提高模型的鲁棒性和泛化能力。此外，半监督学习解决数据稀疏性问题。在某些领域，标记数据可能非常稀缺，而未标记数据往往更为丰富。半监督学习可以利用这些未标记数据来解决标记数据稀疏性问题，帮助模型更好地适应数据的特点。

2.3 Self-training 理论及相关算法

2.3.1 Self-training 理论

半监督学习通过不断给未标记的例子贴赋予标签，并添加大量未标记的样本到训练放大的带标签的训练集，进行提高最终分类器的有效性。self-training 相比其它的半监督学习方法的因其简单以及不需要任何假设而被广泛的应用。Self-training 是最简单的半监督方法之一，其主要思想是找到一种方法，用未标记的

数据集来扩充已标记的数据集，流程如下：首先，利用已标记的数据来训练一个好的模型；然后使用这个模型对未标记的数据进行标记生成伪标签的；最后，将生成的伪标签与原始的标记数据相结合，并在合并后数据上进行联合训练；整个过程重复直到达到收敛。Self-training 伪代码如算法 1.

算法 1: Self-training 算法

输入: L，U

输出: 分类器 H

1. 初始化高置信度样本集 $S=\varnothing$
2. **While** $U\neq\varnothing$ **Dos**
3. 在 L 上训练一个分类器 H
4. 利用 H 为 U 分配标签
5. 选择高置信度无标签样本加入 S
6. $L=L\bigcup S;U=U-S$
7. **End while**
8. **Return** H

2.3.2 Self-training 相关算法

由以上对 Self-training 的介绍可知，使用有标记样本训练好的模型对未标记数据进行伪标签的预测不可能都是准确的，因此对于 Self-training 最大的问题在就在于伪标签可能存在噪声。因此，Self-training 算法的关键是尽可能选择不含噪声的无标记样本进行伪标签的预测。基于这个特点学者们提出了很多通过提升高置度样本选取的算法，通常两种方法是通过一定的规则进行高置信度样本选取和通过对选取的高置信度样本中可能得噪声进行过滤以选择一个子集加入训练集。基于以上高置信度样本的选取方法，接下来本文对一些现有的优秀的 Self-training 算法进行简单的介绍。

1.SETRED

SETRED 提出了利用一种特定的数据编辑方法，从高置信度样本中识别和去除错误标记的样例。具体而言，在每次迭代的 Self-training 过程中，使用局部切边权统计量来帮助估计新标记的样本是否可靠，并且只使用可靠的自标记样本来扩大标记的训练集。SETRED

具体来说，SETRE 首先通过从标记集 L 中学习假设进行自我训练过程。在每次自我训练迭代中，训练的分类器选取未标记高置信度样本，并利用训练的分类器预测其标签后并加入原始有标签数据集 L 。SERTED 要求加入高置度样本不改变原始有标签样本集的数据类别分布。对于一个二分类数据集，假设原始有标记样本第一类和第二类的样本数量为 1:2，则加入高置信度样本后两类的比例依然保持 1:2。然后，SETRE 使用更新后的有标签集 L 构造邻近图进行错误标记示例的识别。图中的每个样本都是一个顶点，如果 x_i 和 x_j 之间的距离满足等式 $Dist(x_i,x_j)\le \max(Dist(x_i,x_z),Dist(x_i,x_z))$，则连接具有不同标签的两个顶点的边称为割边。然后，根据图中的邻域来识别错误标记的例子。一个样本的邻域是与其构建连接的样本组成。直观地说，如果一个样本位于切割边过多的邻域中，则该示例应被视为异常点。为了利用样本的切割边缘的信息，SETRED 统计每个有标记样本中割边权重进行假设检验识别噪声。

SETRED 主要包含三步：（1）根据最近邻规则从无标签样本集中选择高置信度样本并预测标签，结合原始有标签样本集构造邻近图；（2）利用 CEWS 识别预测错误的样本（可疑样本），将这些可疑样本从高置信度样本集中删除。（3）将剩余的高置信度样本加入到有标签集中进行迭代训练。

2. STDP

基于高置信度样本的选取方法，STDP 介绍了一种基于密度和峰值空间发现的数据空间 Self-training 半监督分类的框架，帮助训练出更好的分类器。STDP 是一个将 DPC 算法和 self-training 算法相结合的半监督算法。首先，STDP 利用少量有标记样本训练初始分类器。然后，DPC 基于聚类中心的密度高于其邻居，并且与密度较高的点之间的距离相对较大的思想揭示数据集的空间结构，换句话说，如果两个数据点的距离越近，它们就越相似的。DPC 让每个数据点指向它的密度比它大距离它最近的样本对探索整个数据空间的真实结构。之后，STDP 找到每个有标记样本点的“前置”和“后记”样本作为高置信度样本，并利于训练的分类器预测其标签加入有标签样本集训练分类器。以上几步迭代直到算法收敛。

3. STDPNaN

STDPNaN 是一个将 STDP 与自然邻居相结合的 self-training 算法。受自然社

会社交网络的启发，STDPNaN 认为互为邻居的样本为自然邻居。即如果 x_i 认为 x_j 是它的邻居，同时 x_j 认为 x_i 也是它的邻居，则 x_i 和 x_j 为自然邻居。其次，每个样本依据自然邻居和 DPC 算法重新定义并计算样本密度和峰值。然后，每个样本寻找密度比它大距离最近的样本为其“前置样本”探索整个数据空间的真实结构。之后，找到每个有标记样本点的“前置”和“后记”样本作为高置信度样本，并利于训练的分类器预测其标签加入有标签样本集训练分类器。以上几步迭代直到算法收敛。

4. STDPCEW

STDPCEW[24]结合 STDP 和 CEW[25] 来训练分类器。与 STDP 相同 STDPCEW 将每个已标注样本的“前置”和“后继”未标注样本作为候选高置信度样本。此外，STDPCEW 还使用 CEW 选取高置信度样本中具有更高置信度的未标记样本，并将其添加到 L 中以重新训练分类器。上述过程反复进行，直到所有未标注样本都被完全标注为止。

5. STDPDE

STDPDE 将 DE 集成到 STDP 框架中以训练分类器。在 STDP 的基础上，通过 DPC 揭示数据集的潜在空间结构。然后，STDPDE 利用 DE 来优化 Self-training 过程分类器预测的未标记数据的定位在此基础上，STDPDE 通过 DPC 揭示数据集的底层空间结构。

6. STOPF

STOPF 是一个将最优路径森林(OPF)与 self-training 融合的半监督算法。STOPF 主要包含 2 步：(1)将全部的有标签样本作为 OPF 的根，寻找有标签样本到无标签样本的最佳路径，从而完成 OPF 的构造；(2)利用 OPF 筛选出高置信度样本，使用分类器对这些样本进行预测并添加进训练集。

2.3.3 Self-training 相关算法优缺点总结

在本节中，本文总结了相关算法的优点和缺点，如表 2.2 所示。

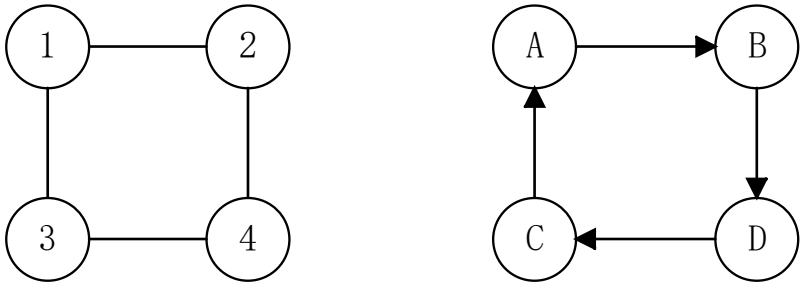
表 2.2　相关算法的优缺点总结

算法	优点	缺点
STDP	使用 DPC 揭示了数据的潜在空间结构，提升了高置信度样本的选取质量。	样本点密度的计算依赖与截断距离，不适用像流形数据非规则数据集;使用局部信息选取高置信度样本
STDPNaN	无参数	不适用非规则数据集；使用局部信息选取高置信度样本
STDPCEW	通过数据编辑技术对高置信度样本中的噪声进行过滤，提升了分类其的性能。	不适用于不规则数据集，且时间复杂度高
SETRED	通过数据编辑技术对高置信度样本中的噪声进行过滤，提升了分类其的性能。	构架邻近图的时间复杂度高且基于局部信息，此外基于欧式距离度量样本之间的关系不适用于非规则数据集
STDPOF	无参数	时间复杂度高
STDPDE	使用差分进化过滤高置信度样本中的噪声点	时间复杂度高；不适用于不规则数据集

2.4 图理论

在线性表中，数据元素之间是被串起来的，仅有线性关系，每个数据元素只有一个直接前驱和一个直接后继。在树形结构中，数据元素之间有着明显的层次关系，并且每一层上的数据元素可能和下一层中多个元素相关，但只能和上一层中一个元素相关。图是一种较线性表和树更加复杂的数据结构。

定义：图(graph)是由一些点(vertex)和这些点之间的连线(edge)所组成的；其中，点通常被成为"顶点(vertex)"，而点与点之间的连线则被成为"边或弧"(edege)根据边是否有方向，将图可以划分为无向图和有向图，如图 2.2 所示。



无向图 有向图

图 2.2 图结构

顾名思义无向图就是图上的边没有方向，图 2.1 左图展示了一个无向图。该图的顶点集为$V=\{1,2,3,4\}$，边集$E=\{(1,2),(1,3),(2,4),(3,4)\}$在无向图中，边$(1,2)$和边$(2,1)$是一样的，也就是说和方向无关。有向图就是图上的边有方向图 2 右图展示了一个有向图。该图的顶点集为$V=\{A,B,C,D\}$，边集$E=\{(A,B),(B,D),(D,C),(C,A)\}$在有向图中，边$(A,B)$和边$(B,A)$是不一样的，也就是说和方向有关。

本文中本文使用有向图结构生成原型树，并用无向图进行近亲节点图编辑过滤噪声。

2.5 基分类器

为了验证所提算法的有效性，本文以KNN作为基分类器。因为STDP和STDPNaN使用3NN(3 nearest neighbors)作为基分类器，为了保持实验的一致性，在以KNN为基分类器时，所有算法都采用3NN作为基分类器。

(1) K-近邻(KNN)：它是最简单的分类方法之一.K-近邻的基本思路如下：对于给 定的训练数据集以及待分类的样本 x，在训练集中找出 k 个样本，这几个样本是与 x 距离最近的前 k 个样本，找到 k 个近邻样本中最多的类标签，即为 x 的类标签.k 的选 择和距离度度量会对 K-近邻算法的分类性能产生重大影响.此外，K-

近邻方法便于理 解和操作，但计算量复杂度较高，每个待分类的样本的 k 个最近邻样本都要通过计算 该样本到所有已知类标签样本的距离才能求得.

(2) 支持向量机(SVM)：SVM 只能对有两种类标签的数据进行分类，是一个二分 类的模型，有线性和非线性的，主要过程如下：给定的训练数据集被映射到另一个空 间，在这个新的空间中寻找一个间隔最大化的超平面，也就是决策边界，属于不同类 别的样本被分到超平面的两侧，以便更好地预测新数据的标签，这里的另一个空间是比原空间更高维的空间.支持向量的数量是决定支持向量机算法复杂度的关键，而与空间的维数无关，这在一定程度上避免了因维度过高造成计算复杂度高的缺点，但是支持向量机对大规模数据样本难以实施，主要是由于该方法借助二次规划求解支持向 量.

(3) 决策树：决策树是一种树形结构的模型，可以解决多分类的问题，该分类模 型中的根节点表示全部属性，非叶子节点表示对样本属性的测试，叶节点表示对样本 进行分类后得到的类标签，分支表示判断结果的输出.对已知标签的样本学习训练之 后，每个样本都有一组属性和对应的分类结果，通过节点的分裂和阈值的确定这两步的学习可以得到一个决策树模型.ID3、C4.5、C5.0 和 CART 等是常见的几种决策树生成算法.

2.6 评价指标

（1）准确率 *Accuracy* 和 F_1-*score*

本文选取准确率 *Accuracy* [62]和 F_1-*score* [63] 百分值作的为评价指标。*Accuracy* 和 F_1-*score* 可以根据混淆矩阵来计算。本文使用二元分类模型来简单地介绍这两个评估度量。

Accuracy 是预测准确样本的数量比样本总量，计算公式如下：

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

F_1-*score* 是一个基于准确度和召回率的综合评估指标，计算公式如下：

$$F_1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{2}$$

其中，$\mathrm{Pr}ecision=\frac{TP}{TP+FP},\mathrm{Re}call=\frac{TP}{TP+FN}$ 。

True Positive（TP）：真正类。样本的真实类别是正类，并且模型识别的结果也是正类。

False Negative（FN）：假负类。样本的真实类别是正类，但是模型将其识别为负类。

False Positive（FP）：假正类。样本的真实类别是负类，但是模型将其识别为正类。

True Negative（TN）：真负类。样本的真实类别是负类，并且模型将其识别为负类

（2）威尔科克森符号秩检验(Wilcoxon signed-rank test, Wilcoxon)

Wilcoxon 符号秩检验是一种非参数检验[64]。Wilcoxon Signed-Rank 检验是在在不假设数据服从正态分布的前提下，判断出相应的数据总体分布是否有差异性。在 95%的置信水平下进行 ，以验实验结果的有效性。符号“+”，“-”分别表示本文所提算法显著优于和低于对比算法，符号“~”表示与对比算法没有显著差异。

2.7 数据集介绍

为了验证本文所提算法的有效性，本文在一些公开进行实验来验证本文提出算法的有效性。表 2.3 对本文所涉及的数据集基本特征进行简要的描述。数据集 Glass、SPECTFheart、Heart、Ilpd、Australian、Breast、Transfusion、Pima、Diabetes、German、Cmc、Cleve、Ecoli、Haberman、Planning-relax、Iris、wdbc、Wholesale-customers、Wilocalization、tic-tac-toe、Waveform、Hepatitis 均来自 UCI[1]公开数据集。ORL[2]、Palm[3]、AR[65]、FERET32x32[66]、UPS[67]、YaleB[68]、umist[4]、Mpeg7uni[69]是图像数据集。接下来本文对数据集的来源和用途进行简单的介绍：

[1] https://archive.ics.uci.edu/ml/datasets.php.

[2] http://www.uk.research.att.com/facedatabase.html.

[3] https://www.gwern.net/Crops.

[4] http://images.ee.umist.ac.uk/danny/database.html.

表 2.3 数据集描述

ID	数据集	样本量	维度	类别	缩写
1	Glass	214	9	6	GLA
2	SPECTFheart	267	44	2	SPH
3	Heart	270	13	2	HEA
4	ORL	400	1024	40	ORL
5	umist	574	1030	20	UMI
6	Ilpd	583	10	2	ILPD
7	Australian	690	14	2	AUS
8	Breast	699	10	2	BRE
9	Transfusion	748	4	2	TRA
10	Pima	768	8	2	PIM
11	Diabetes	768	8	2	DIA
12	German	1000	20	2	GER
13	FERET32x32	1400	1024	200	FER
14	Mpeg7uni	1400	6000	70	MPE
15	Cmc	1473	9	3	CMC
16	AR	1680	1024	120	AR
17	Palm	2000	256	100	PAL
18	YaleB	2414	1024	38	YALEB
19	Cleve	303	13	4	CLE
20	Ecoli	336	7	8	ECI
21	Haberman	306	3	2	HAN
22	Planning-relax	182	12	2	PRX
23	Iris	150	4	3	IRS
24	wdbc	569	30	2	WDC
25	Wholesale-	440	7	2	WCS
26	Wilocalization	2000	7	4	WIN
27	tic-tac-toe	671	9	2	TCE
28	Waveform	2746	21	3	WFM
29	UPS	2007	256	10	UPS
30	Hepatitis	142	19	2	HPS

3. 鲁棒的近亲节点图编辑 self-training 算法

为了提高置信度样本的选取质量，本文提出了一种新的数据编辑技术——相对节点图编辑（Relative Node Graph Editing，RNGE）。具体来说，基于块的不相似性度量被用于计算每个样本的密度和峰值，以构建原型树来揭示数据的潜在空间结构。然后，本文为每个样本定义相对节点图（RNG）。最后，通过基于的假设检验来识别候选高置信度样本集中的错误标记样本 RNG。结合以上内容，本文提出了一种基于相对节点图的鲁棒 Self-training 算法（A Robust Self-training Algorithm based on Relative Node Graph，STRNG），该算法使用 RNGE 来识别错误标记的样本并对其进行编辑。本文在 14 个真是数据集与已有 STDP、STDPCEW、SETRED 和 STDPDE 算法进行实验结果对比，实验结果表明本文所提出的算法可以提高 Self-training 算法的性能。

本章将详细介绍本文提出的基于近亲节点图鲁棒的 self-training 算法思想，实验的设置和实验结果等。实验部分本文展示 10%有标记样本下各个算法 Accuracy 和 F_1-score 值不同比例有标记样本下各个算法的 Accuracy 值折线图；各个算法的噪声实验折线图；各个算法时间复杂度及运行时间分析。

3.1 算法描述

STRNG 的伪代码如算法 3 所示：

算法 3：STRNG

输入: L ,U , k ,左拒绝域 τ ， 基分类器

输出：分类器

1.随机将 X 分成 t 个子集 D

2.FOR $D \subset X$,$|D| \geq 2$

3. 根据算法 2 生成树建立森林

4. 计算不相似性矩阵 M

5.END FOR

6.根据公式 5 计算样本密度 ρ_i

7.根据公式 6 计算样本峰值 δ_i

8.根据公式 7 计算样本原型 P_i，并建立原型树

9.计算全局分布 $\pi g = \left|L_{Yg}\right| \Big/ \left|L_Y\right|$

10.WHILE $S \neq \varnothing$ DO

11. 选取高置信度样本集 S

12. 为高置信度样本分配标签

13. 计算 q_{ij} 和 I_{ij}

14. 计算 J_i

15. 计算 $E(J_i \mid H_0)$， $D(J_i \mid H_0)$

16. if $J_i > \tau$

17. $S = S - x_i$

18. end if

19. $L = L \cup S$

20 训练分类器

21.end while

22.输出分类器

3.2 算法思想

3.1.1 基于块的不相似性度量

欧式距离以样本各个维度之间的距离来衡量样本之间的相似性，这也导致了欧式距离不适用像流形数据等结构不规则的数据集。为了克服欧氏距离不考虑数据分布的缺点，文章[70]提出了基于块的不相似性来度量样本间的关系。简单来讲，基于块的不相似性以数据分布为考量定义为覆盖两个样本的最小概率块。接下来，本文将详细介绍基于块的不相似性原理及在 STRNG 中的应用。

基于块的不相似性即包含两个样本的最小质量块[72]，计算如公式 3。

定义 1. 设 $R(x_i,x_j \mid H;D)$ 表示相对于 H 和 D 的覆盖 x_i 和 x_j 的最小块。定义为：

$$R\left(x_i,x_j \mid H;D\right)=\underset{r\subset H,s.t.\{x_i,x_j\}\in r}{\arg\min}\sum_{z\in D} I\left(z\in r\right) \tag{3}$$

其中，$I(\bullet)$ 表示指数函数。假设 D 表示样本集包含 ψ 个样本，从 X 中随机选取数据集 D。如果 $|X|<256$，$|D|=\psi$，否则 $\psi=256$。$H\in \mathrm{h}(D)$ 表示将空间划分为非重叠区域。x_i 和 x_j 代表 D 中的两个样本。

覆盖两个样本的最小概率块是通过平行分隔获得树来计算，树的分类过程如算法 2。本文以数据集 $\{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9,x_{10}\}$ 为例，使用算法 2 通过平行分隔建立一棵树，如图 3.1 所示。每次分裂随机选择一个属性，在该属性的最大值和最小值之间随机选择的一个值作为分裂点，属性值小于分裂点的样本划入左子集，大于分类点进入右子集。以第一次分裂为例，随机选择 $\{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9,x_{10}\}$ 的一个属性进行分裂，由图可知样本 x_2，x_4，x_6 和 x_{10} 属性小于分裂点作为左节点，反之右节点为 $\{x_1,x_3,x_5,x_7,x_8,x_9\}$，树以此进行分裂直到一个样本为一个节点或者达到到树的最大高度，本文中本文设置树的最大高度为 $V=\log 2$。由公式 3 可知包含采样点 x_1 和 x_8 的最小质量块是 $\{x_1,x_3,x_8,x_9\}$

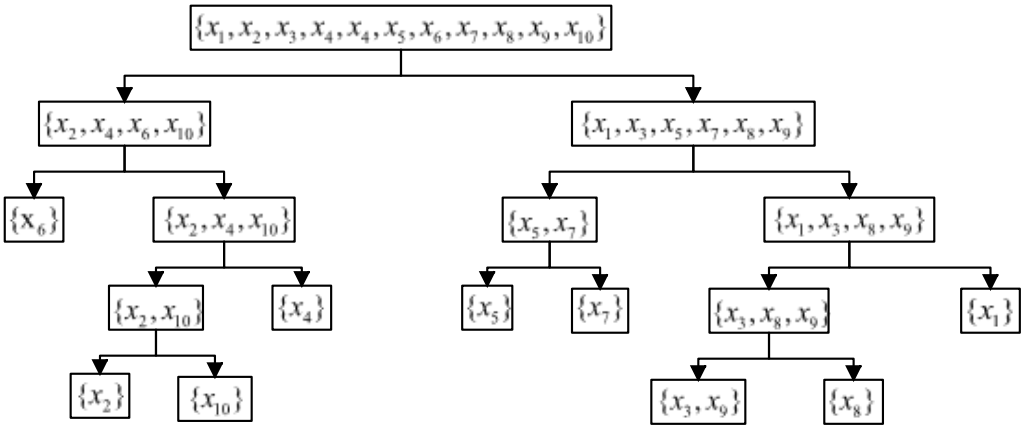


图 3.1 树结构

算法 2：构建树

输入：数据 X ,树的限制高度 V ，树的分类高度 e

输出：一棵树

1.**IF** $|X| \le 1$ **OR** $e \ge V$ **Then**

2. **RETURN** 节点大小

3.**ELSE**

4. 随机选取分类属性

5. 随机选取分裂点 //分裂点事属性最大值和最小值之间的值

6 属性值小于分裂点的样本归入左节点

7. 属性值大于分裂点的样本归入左节点

8.**RETURN** 分裂点、分裂属性、左节点、右节点

9.**END IF**

在本文中，本文通过平行分隔分裂方法构建 t 棵树形成森林[73]（iForest），每棵树都由独立子集构建。样本间块的不相似性通过遍历森林中每一棵树最小概率块来计算包含样本 x_i 和 x_j 的最小区域块总和 $\frac{1}{t}\left|R\left(x_i, x_j \mid H_v, D\right)\right|$，如定义 2：

定义 2 样本 x_i 和 x_j 基于块的相似度计算公式如下：

$$m_e\left(x_i, x_j\right) = \frac{1}{t}\sum_{v=1}^{t} R\left(x_i, x_j \mid H_v, D\right) \tag{4}$$

其中，$t = 100$。

3.1.2 原型树

原型树基于块的不相似性重新定义样本的密度和峰值，并找到数据空间的底层结构来评估点之间的相似程度选取高置信度样本。树的结构克服了因数据空间结构导致的误分类问题。接下来，本文将提供详细的原型树原理介绍及在本文中的应用。

定义 3 样本点 x_i 基于块的局部密度函数，如下：

$$\rho_i = \sum_{x_j \in knn(i)} \exp\left(-m_e\left(x_i, x_j\right)\right) \tag{5}$$

其中，$knn(i)$ 表示距离样本点 x_i 最近的 i 个样本，样本点的距离计算依据公式（2）得到的相对度。

定义 4 样本 x_i 的密度峰值 δ_i，如下：

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} \left(m_e\left(x_i, x_j\right)\right), if\ \exists j, s.t. \rho_i > \rho_j \\ \\ \max\limits_{j} \left(m_e\left(x_i, x_j\right)\right), otherwise \end{cases} \tag{6}$$

在计算块的不相似性度量计算样本的密度 ρ_i 和峰值 δ_i 的后，本文利用 ρ_i 和值 δ_i 构建新的数据结构原型树[39]，样本原型的定义下：

定义 5　P_i 表示样本 x_i 的原型，公式如下

$$\begin{aligned} &P_i = x_i \\ &s.t. \exists j, \delta_i = \min_{j:\rho_i < \rho_j} me\left(x_i, x_j\right) \end{aligned} \tag{7}$$

构建原型树的步骤如下：

（1）首先，对样本的密度进行排序；

（2）密度最大的样本被作为根节点。即，根节点 x_i 必须满足 $\forall j, \rho_i \geq \rho_j$；

（3）据公式 (5) 计算每个样本的原型点，根节点没有原型；

（4）样本及其原型具有相同类标记由实线边连接。反正则用则用虚线连接。

在 Iris 数据集构建原型树的过程中，本文随机选择包括三个类别的 10 个样本，以及它们的密度 ρ_i 和峰值 δ_i。本文以这 10 个样本作为示例展示构建原型树的构建过程。表 3.1 显示了 10 个样本的密度和类别，其中 sort 是对样本密度 ρ_i 从大到小排序。表 3.2 展示了样本与密度 ρ_i 比自己大的样本之间的 $m_e(x_{i,}x_j)$，根据定义 5 本文对最小的 $m_e(x_{i,}x_j)$ 加粗。根据原型树的构建过程以及表 3.1 和表 3.2，建立一棵树如图 3.2 所示。

表 3.1 样本密度

Sort	Sample	ρ_i	Class
1	x_{30}	4.9545	Class 1
2	x_{123}	4.8288	Class 3
3	x_{143}	4.7762	Class 3
4	x_{47}	4.7757	Class 1
5	x_{14}	4.7345	Class 1
6	x_{75}	4.6593	Class 2
7	x_{51}	4.6152	Class 2
8	x_{15}	4.5533	Class 1
9	x_{88}	4.3379	Class 2
10	x_{143}	4.1972	Class 3

表 3.2 样本原型

	x_{14}	x_{15}	x_{30}	x_{47}	x_{51}	x_{75}	x_{88}	x_{120}	x_{123}	x_{143}
x_{14}	\	\	**0.3441**	\	\	\	\	\	0.9707	0.9045
x_{15}	0.5083	\	0.5115	0.3673	0.8321	0.8237	\	\	0.9759	0.8729
x_{30}	\	\	\	\	\	\	\	\	\	\
x_{47}	\	\	**0.3379**	\	\	\	\	\	0.9838	0.9151
x_{51}	0.8952	\	0.8478	0.8656	\	**0.3807**	\	\	0.6495	0.7069
x_{75}	0.8646	\	0.8589	0.8683	\	\	\	\	0.7258	**0.6225**
x_{88}	0.8973	0.8602	0.8903	0.8961	0.4911	**0.3147**	\	0.7371	0.6209	\
x_{120}	0.9013	0.8752	0.9024	0.9069	0.6197	0.5556	**0.3813**	\	0.7351	0.4564
x_{123}	\	\	**0.9772**	\	\	\	\	\	\	\
x_{143}	\	\	0.9106	\	\	\	\	\	**0.6439**	\

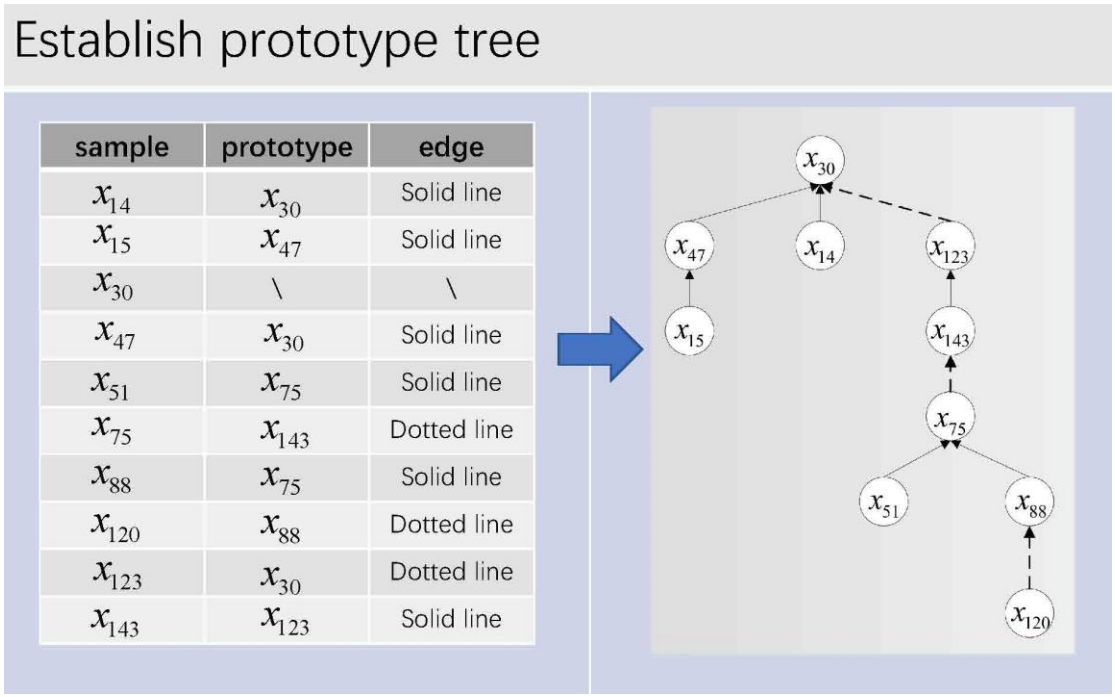


图 3.2 原型树

3.1.3 近亲节点图编辑

为了利用样本之间的原型关系来选择高置信度样本，本文在定义 (6) 中定义了相对节点集。

定义 6 样本 x_i 的近亲节点集 B_i

$$B_i = \left\{ x_j \mid x_j = P_i \vee x_i = P_j, i \neq j \right\} \tag{8}$$

如图 1 所示，样本 x_{75} 的近期节点集 $B_{75} = \{x_{51}, x_{88}, x_{143}\}$。

确定 x_i 的相对节点集后，原型树中包含相对节点集的子图即为 x_i 的 RNG。图 3.3 是显示了原型树中样本点 x_{75} 的 RNG。图 3.3 中实线称为有向边，虚线称为切边。

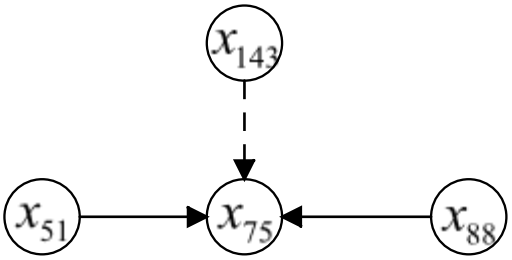


图 3.3 近亲节点集

显然，如果一个无标记样本有大量的切边，那么很明显它被许多贴有不同标签的样本所包围。，更可能会被错误地标记。因此，本文提出了相对节点图编辑（RNGE）方法，通过对统计信息的节点权重进行假设检验，来识别高置信度样本。接下来本文将详细介绍 RNGE。

定义 7　样本 x_i 与其近邻样本 x_j 之间的相对边权重 q_{ij} 计算如下：

$$q_{ij} = \rho_i \left(1 + m_e\left(x_i, x_j\right)\right)^{-1} \tag{9}$$

显然，相对边缘权重是不对称的，即在大多数情况下 $q_{ij} \neq q_{ji}$，样本密度 ρ_i 越大切边权重 q_{ij} 越大。

定义 8　J_i 为样本 x_i 的切边权重之和，其定义如下

$$J_i = \sum_{x_j \in B_i} q_{ij} J_{ij} \tag{10}$$

如果样本 x_i 和 x_j 属于同一类别，则 $I_{ij} = 0$，否则 $I_{ij} = 1$。

零假设 H_0：样本标记过程是独立的，且服从概率分布 $\pi_g = \left|L_{Y_g}\right| / \left|L_Y\right|, g \in [1, m]$. L_{Y_g} 表示数据集 L 中带有标签 g 的样本数量，L_Y 是所有标签样本的数量。在 H_0 设下，I_{ij} 是一个独立的随机变量服从布尔参数 $1-\pi_{g_i}$。其中 g_i 是样本 x_i 属于 g 类别的先验概率。假设 J_i 服从正态分布，其的数学期望 E 和方差 D 可通过以下公式计算:

$$E(J_i \mid H_0)=(1-\pi_{g_i})\sum_{x_j \in B_i} q_{ij}$$
$$D(J_i \mid H_0)=\pi_{g_i}(1-\pi_{g_i})\sum_{xj \in Bi} q_{ij}^2 \tag{11}$$

显然，高置信度的样本中具有更小比例的不同标签的近亲节点。在 H_0 下，高高置信度性样本的 J_i 应显著小于预期。因此，设置为显著性水平 θ，并且属于左拒绝域的样本将作为高置信度样本添加到训练集中，而属于右拒绝域除域的样本将被丢弃。

3.3 时间复杂度

n 表示样本数，t 表示 iTree 的数量，并代表每个 iTree 的大小。ψ 代表每个 iTree 的大小。RNG 的时间复杂度分为以下几个部分：

（1） 计算块的时间复杂度为 $O(t\psi\log\psi+n^2t\log\psi)$。

（2） 计算密度 ρ 的时间复杂度为 $O(n^2)$；

（3） 计算每个样本的聚类峰值 δ 和原型 P 需要 $O(n^2)$；

（4） 还需要 $O(n^2)$ 来构建近亲节点图；

（5） J 的计算时间复杂度为 $O(n^2)$

上述分析表明，STRNG 算法的总体时间复杂度为 $O(t\psi\log\psi+n^2t\log\psi)$。

3.4 实验设置

3.4.1 实验环境

所有实验均使用英特尔 (R) 酷睿 (TM) i9-11900k CPU@3.50GHz 64.0GB 内存、Wine10 64 位操作系统进行

3.4.2 数据集和实验设置

为验证 STRNG 的性能、STDP、STDPCEW、SETRED 和 STDPDE 作为比较算法，并根据原文章复制代码。在 GLA、CLE、ECI、HAN、PRX、IRS、WDC、WCS、WIN、TCE、WFM、HPS、UPS 和 ORL 数据进行实验。

在 95%的置信水平下进行威尔科克森符号秩检验(Wilcoxon signed-rank test, Wilcoxon)，以验 STALN 实验结果的有效性。符号“+”，“-”分别表示 STALN 显著优于和低于对比算法，符号“~”表示 STALN 与对比算法没有显著差异。

对比算法涉及参数均取自原文，STRNG 涉及的所有参数值见表 3.3。

表 3.3　参数设置

算法	参数
KNN	K=3
STDP[46]	$P_\alpha = 2$
STDPCEW[52]	$\alpha = 2, \theta = 0.1$
SETRED[36]	阈值$\theta = 0.1$
STDPDE[51]	$\alpha = 2, \theta = 3$
STRNG	$k = 7, \theta = 0.1$

3.5 分类性能分析

以 3NN 为基分类器在 14 个基准数据集进行实验。每次实验随机选择 10%的样本构建有标签样本集，其余样本构建无标签样本集。本文选择准确率(Accuracy)和 F_1 分数(F_1-score)作为算法的评价指标。记录所有算法在 14 个数据集上运行 50 次的平均值和标准差，实验结果见表 3.4 和表 3.5，在每个数据集上将分类性能最好的结果用粗体字标记，括号中的数字是每个算法的排名。AVE.ACC、AVE.Fscore 和 AVE.STD 分别表示平均 Accuracy、平均 F_1-score 和平均标准差。

表 3.4　每个算法在不同数据集中的分类 Accuracy（平均值±标准差（排名））

	STDP	STDPCEW	STDPDE	SETRED	STRNG
GLS	76.48±3.88(3)	76.85±2.01(2)	**77.32±5.21(1)**	76.03±4.11(4)	76.03±4.11(5)
CLE	71.56±6.71(4)	71.00±6.35(5)	76.87±5.16(2)	71.89±4.68(3)	**80.44±1.75(1)**
ECI	87.14±3.08(5)	88.69±2.38(3)	88.79±2.77(2)	87.65±3.31(4)	**92.83±1.09(1)**
HAN	60.70±3.56(2)	59.84±1.96(5)	60.01±7.80(4)	60.67±5.34(3)	**66.77±4.83(1)**
ORL	67.63±1.23(3)	88.98±1.05(2)	57.43±1.29(5)	67.39±1.09(4)	**89.03±2.07(1)**
PRX	54.07±2.22(3)	53.63±3.48 (4)	50.48±4.03(5)	54.86±3.51(2)	**60.63±9.53(1)**
WDC	90.39±1.56(2)	89.54±1.81(4)	89.58±2.60(3)	83.31±3.22(5)	**92.93±0.91(1)**
IRS	92.51±5.35(4)	93.01±5.56(3)	76.79±3.93(5)	93.52±3.43(2)	**96.52±1.38(1)**
WCS	81.59±2.41(5)	81.72±4.38(4)	85.43±1.15(2)	83.30±2.81(3)	**86.94±1.01(1)**
WIN	98.27±0.23(3)	97.85±0.48(4)	97.57±0.57(5)	98.28±0.24(2)	**98.65±0.17(1)**
TCE	67.63±3.23(2)	65.73±5.33(4)	57.43±7.01(5)	67.39±3.15(3)	**71.94±5.06(1)**
WFM	82.96±0.97(3)	80.67±1.14(5)	81.74±0.96(4)	83.14±0.61(2)	**85.80±0.67(1)**
UPS	92.71±0.28(4)	93.49±0.77(2)	87.33±1.42(5)	92.79±0.71(3)	**94.45±0.41(1)**
HPS	58.01±8.50(4)	56.08±7.36(5)	59.82±11.32(2)	59.31±5.05(3)	**74.20±13.92(1)**
WSR-test	+	+	+	+	N/A
Ave.ACC	77.30	78.40	74.97	77.25	83.86
Ave.std	3.01	3.09	3.84	2.88	3.31

表 3.5　每个算法在不同数据集中的分类 F_1-score（平均值±标准差（排名））

	STDP	STDPCEW	STDPDE	SRTRED	STRNG
GLA	64.45±3.92(4)	71.38±2.51(1)	55.46±10.97(5)	68.83±4.43(2)	**68.20±6.33(3)**
CLE	55.71±5.99(4)	58.16±6.93(3)	59.32±5.73(2)	53.71±4.95(5)	**62.85±5.43(1)**
ECI	80.33±3.07(3)	8165±4.74(2)	76.02±7.87(5)	79.50±5.85(4)	**87.04±3.19(1)**
HAN	60.71±1.83(2)	58.92±3.63(4)	57.64±7.87(5)	60.67±5.34(3)	**66.77±4.83(1)**
ORL	75.41±3.44(3)	81.36±2.61(2)	34.42±8.73(5)	75.05±3.08(4)	**88.93±5.57(1)**
PRX	54.07±2.22(3)	53.63±3.48(4)	50.48±4.03(5)	54.86±3.51(2)	**60.63±9.54(1)**
WDC	91.03±1.55(4)	89.76±2.62(5)	91.24±1.16(3)	92.08±0.91(2)	**92.69±0.97(1)**
IRS	95.51±5.35(2)	93.01±5.68(4)	76.79±3.93(5)	93.52±3.43(3)	**96.52±1.38(1)**
WCS	81.58±2.41(5)	81.72±4.38(4)	85.43±2.82(2)	83.30±2.81(3)	**86.94±1.02(1)**
WIN	98.25±0.23(3)	97.82±0.35(4)	97.54±0.58(5)	98.29±0.25(2)	**98.63±0.18(1)**
TCE	67.63±3.23(2)	65.72±5.33(4)	57.43±7.01(5)	67.39±3.15(3)	**71.00±8.44(1)**
WFM	82.15±0.70(2)	79.71±0.85(5)	80.54±1.09(4)	82.12±0.69(3)	**84.94±0.65(1)**
UPS	92.25±0.32(4)	93.13±0.86(2)	85.88±1.79(5)	92.50±0.86(3)	**94.19±0.44(1)**
HPS	58.01±8.50(3)	56.08±7.36(4)	**59.82±11.31(1)**	59.31±5.05(2)	55.80±8.43(5)
Wilcoxon	+	+	+	+	N/A
Ave.Fscore	75.46	75.84	69.02	75.64	79.67
Ave.std	3.06	3.61	5.45	3.20	3.66

根据表 3.4 和表 3.5 所示的结果，本文得出如下结论：

（1） 表 3.4 中的结果表明，在 14 个数据集上，STRNG 的准确率高于 SETRED、STDP、STDPDE 和 STDPCEW。这是因为基于块的的不相似性使 STRNG 更适合于在高维和密度非均匀数据集中度量样本之间各个算法在的距离。同时，原型树揭示了样本的潜在空间结构，使高置信度样本的选择更加高效。此外，STRNG 还通过假设检验对选取的高置信度样本进一步的过滤。这使得 STRNG 具有更好的分类性能。

（2） 从表 3.5 中可以看出，在 14 个数据集上，STRNG 的 F1-score 值均高于 SETRED 和 STDPDE，在 13 个数据集上，STRNG 的 F1-score 值高于 STDP 和 STDPCEW。在 GLS 数据集上，STRNG 比 STDPCEW 低 3.18%，在 HPS 上比 STDPDE 低 4.02%。在 HPS 上比 STDPDE 低 4.02%，各个算法的分类 F_1-score 值实在一定范围的误差上上下浮动，结合算法的方差可得出在 GLS 和 HPS 可得出本文的算法与对比比算法的分类

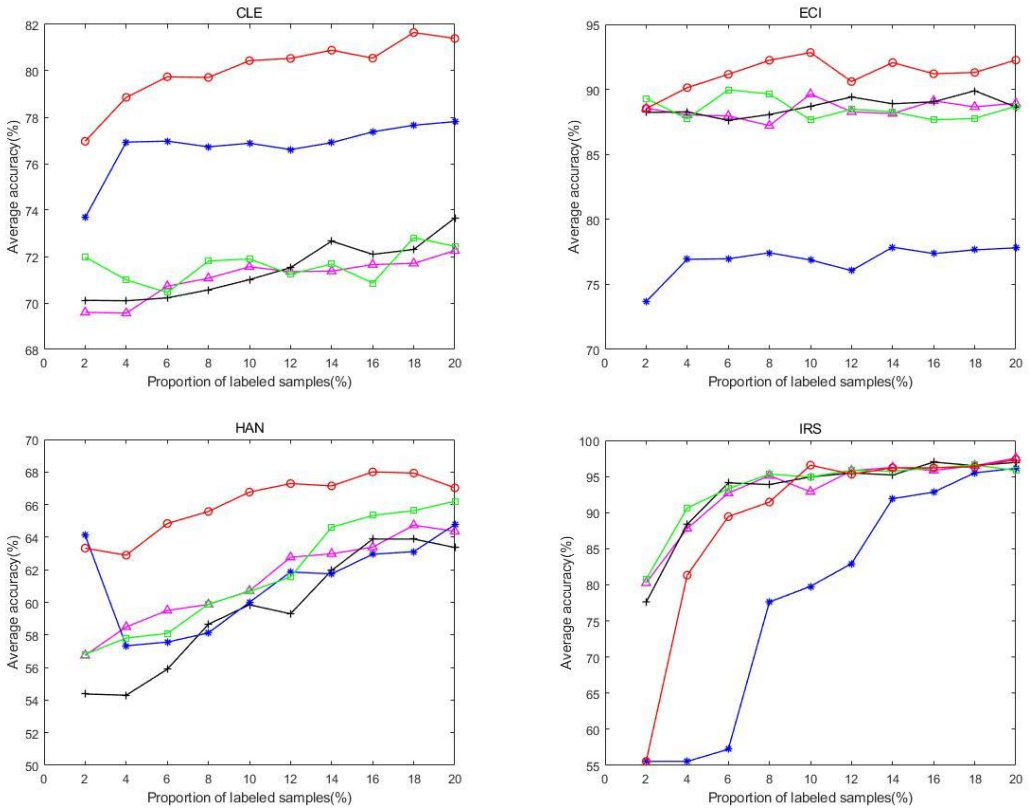
性能在同一水平上。此外，结合各个算法 Accuracy 上的结果也证明这一结论。因此实验结果表明，STRNG 的分类性能优于对比算法。

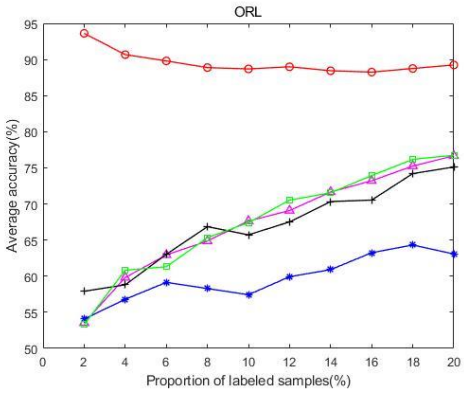
（3） 就表 3.4 和表 3.5 显示在 14 个数据集上的 Accuracy 和 F_1-score 均值，STRNG 均排名第一。同时，统计假设检验测试结果表明，STRNG 的分类性能明显优于 STDPDE、STDP、SETRED 和 STDPDE。

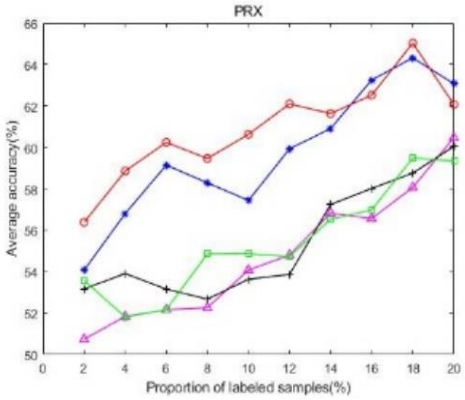
上述实验结果验证了本文提出的鲁棒的近亲节点点图编辑算法可以有效地选择高序列样本，从而提高分类性能。

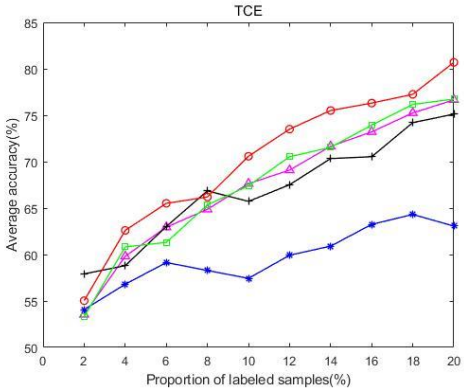
3.6 有标签样本比例对分类性能的影响

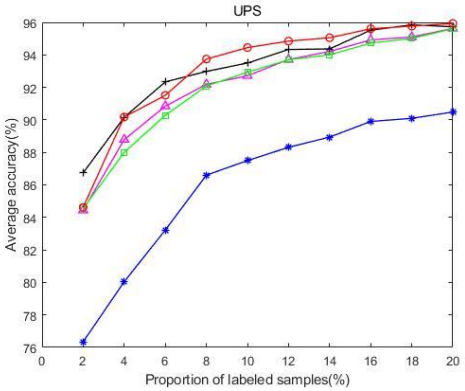
为了验证不同比例的标注样本对分类准确性的影响。本文从 14 个数据集中随机选取标注样本，比例从 2% 到 20% 步长为 2%作为有标签数据集比例，每个比例下进行 50 次实验并记录平均 Accuracy。图 3.4 显示了实验结果。

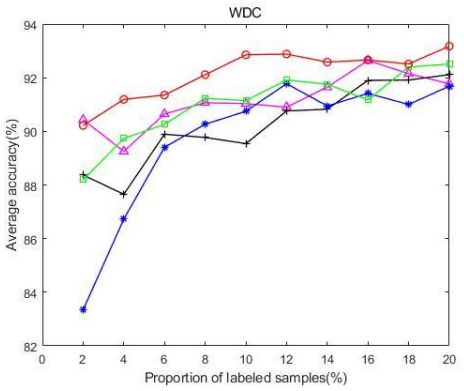


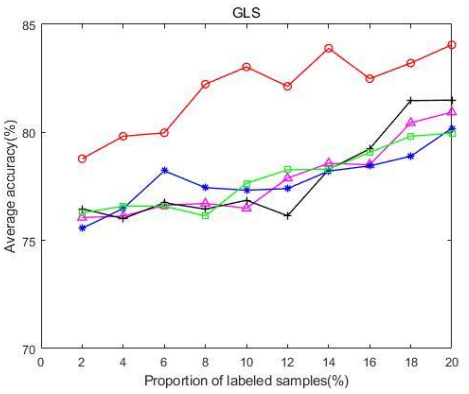

ORL
Average accuracy(%)
Proportion of labeled samples(%)

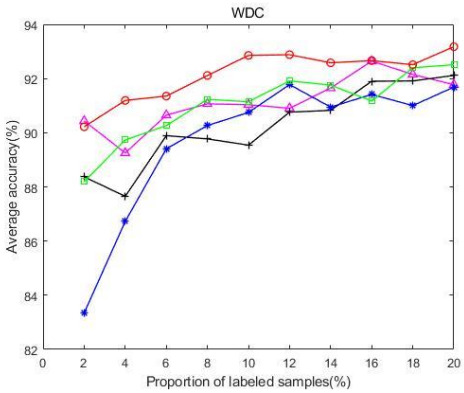

PRX
Average accuracy(%)
Proportion of labeled samples(%)

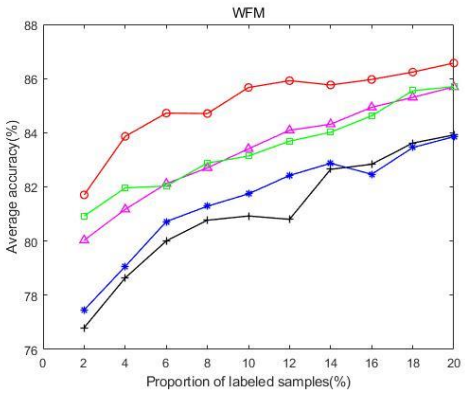

TCE
Average accuracy(%)
Proportion of labeled samples(%)


UPS
Average accuracy(%)
Proportion of labeled samples(%)


WDC
Average accuracy(%)
Proportion of labeled samples(%)


GLS
Average accuracy(%)
Proportion of labeled samples(%)


WDC
Average accuracy(%)
Proportion of labeled samples(%)


WFM
Average accuracy(%)
Proportion of labeled samples(%)

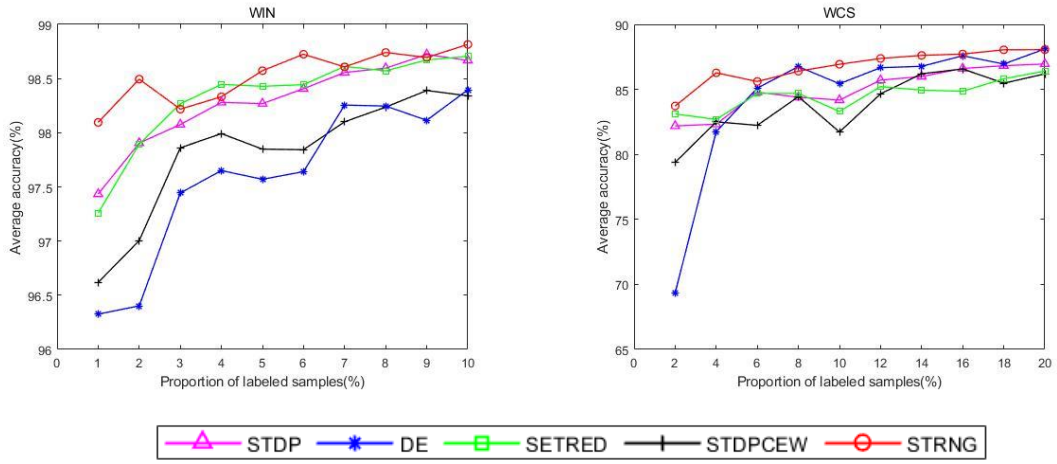


图 3.4 不同比例有标签样本下各分类器分类性能

从图 3.4 中本文可以得出以下结论：

（1） 在 14 个数据集上，五个算法的分类准确率均通常随着标注样本的增加而提高。

（2） 总体而言，STRNG 算法在 ECI、GLS、CLE、ORL、HAN、HPS、WFM、WDC 和其他四种算法上的分类性能优于其他四种算法。在 WIN 和 WCS 数据集。在 WIN 和 TCE 数据集上，随着标注样本比例的增加，分类准确率也随之增加。增加，STRNG 的分类准确率在某些比例上低于对比算法。但总体而言，STRNG 的分类准确率优于对比算法。

（3） 在 IRS 数据集中，STRNG 的分类准确率最初低于 STDP、STDPCEW 和 SETRED。但随着标注样本比例的增加 随着标记样本比例的增加，STRNG 的分类准确率显著提高，最终与 STDP、STDPCEW 和 SETRED 的分类准确率持平。在 TCE 数据集中，STRNG 的分类准确率略低于 STDPCEW。当标注样本比例增加 8%时，STRNG 的分类准确率开始高于 STDPCEW。

上述分析表明，STRNG 具有更好的分类性能。

3.7 参数设置分析

k 和 θ 是 STRNG 中涉及的两个参数。参数 k 是计算密度时的邻居数量。参数 θ 是显著性水平，这两个参数影响了高置信度样本的选择。基于以上 14 个数据集，在 10%的有标记样本比例下对 k 和 θ 运行 STRNG 50 次并记录平均 Accuracy 的百分值。图 5 显示了在不同参数下 STRNG 在 14 个数据上的分类准

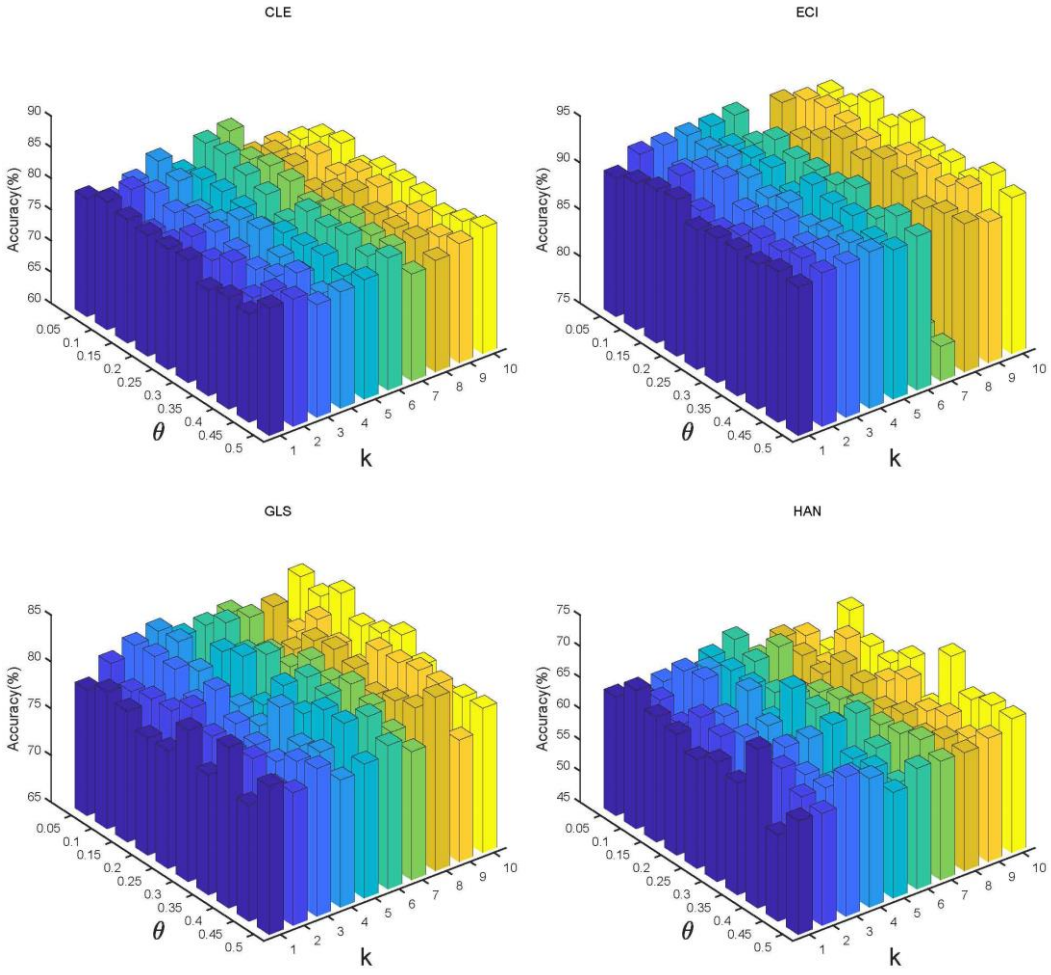
确率。

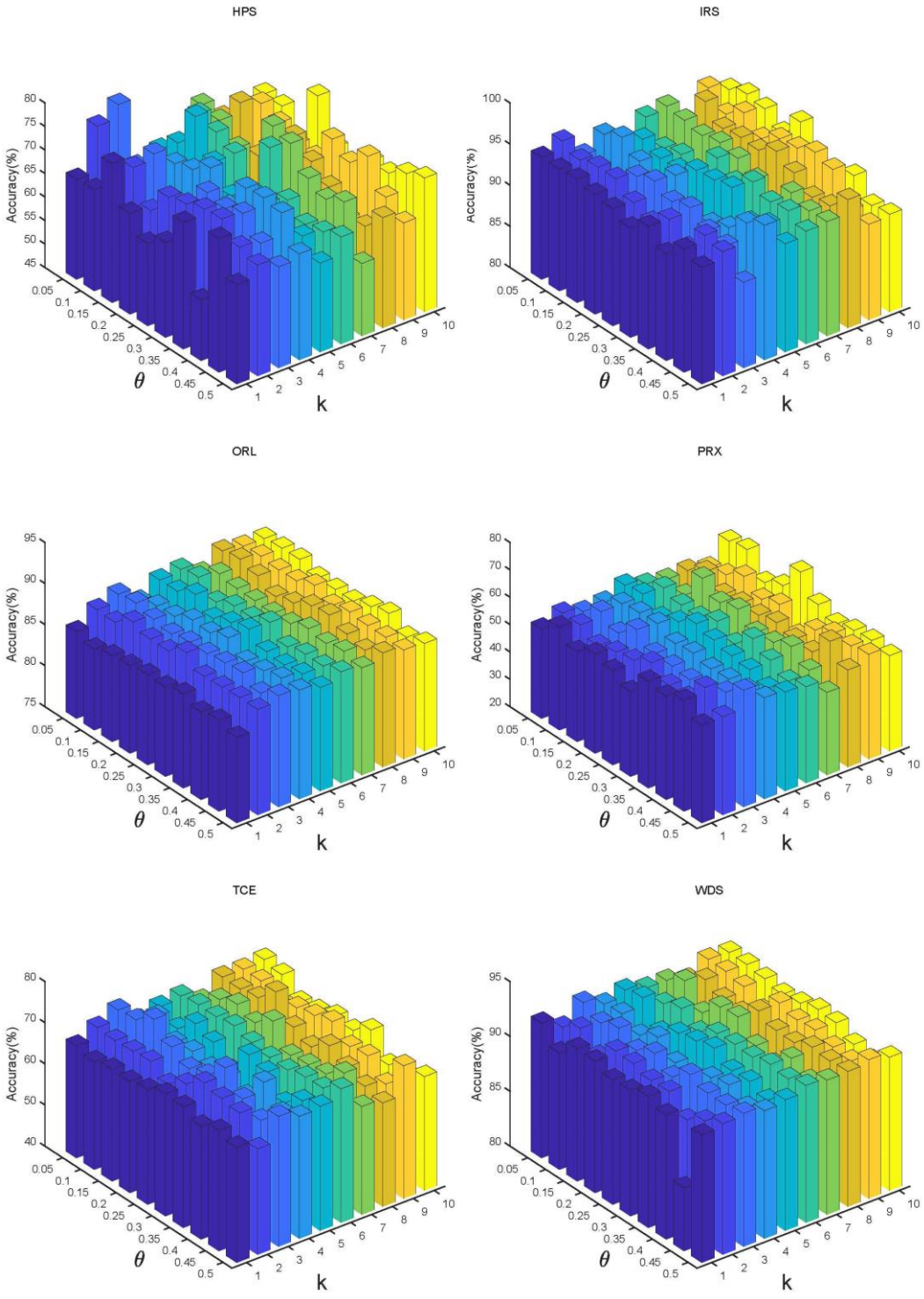
（1）θ 取值对分类准确率的影响

θ 作为一个显著性水平，如果的值太小，则拒绝中的样本会太少域，并且在每次迭代过程中将获得太少的高置信度样本。这将导致过多的迭代，并且无法充分利用潜力每次迭代的数据结构；如果 θ 的值太大，则拒绝的范围域将被扩展、噪声样本可以被添加到高置信度样本，这将影响了分类器的性能。因此本文在[0.05、0.5]的范围内并且步长为 0.05 求取 θ 的最优值。

（1）k 取值对分类准确率的影响

如果 k 的值太小，则不足以显示两个样本之间的密度差异。如果 k 的值太大，也很难区分高密度样和低密度样本，并且计算成本太大。因此本文在[1，10]的范围内设置步长为 1 求取 k 的最优值。




HPS
IRS
ORL
PRX
TCE
WDS
Accuracy(%)
θ
k

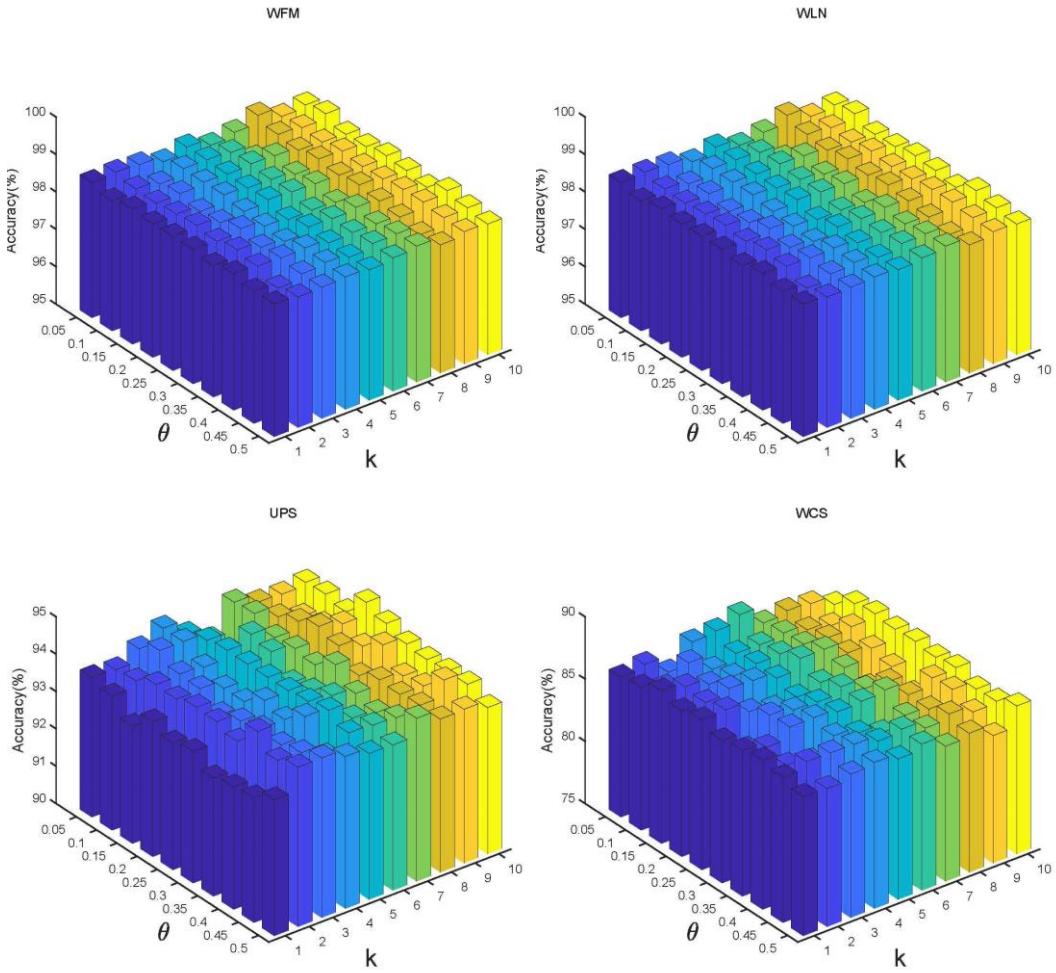


图 3.5 参数分析图

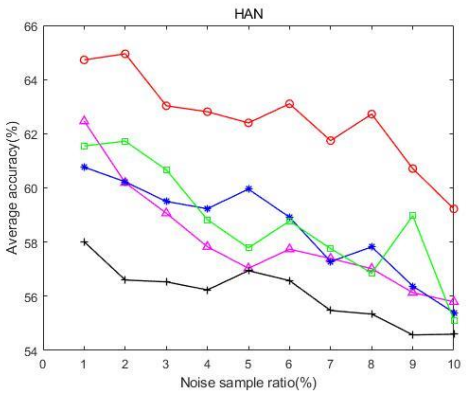
有图 3.5 可看出不同的 k 和 θ 值具有不同的分类准确率，对 STRNG 在 14 个数据集上的分类准确率进行了综合考量，最终确定 $k=7$， $\theta=0.1$。

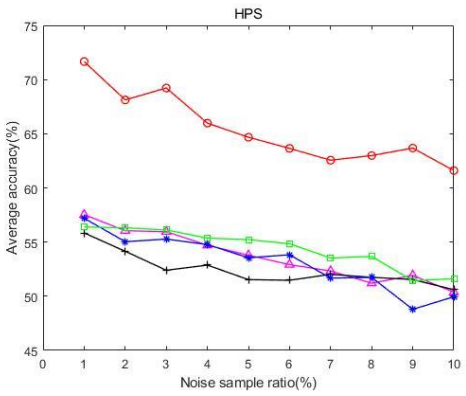
3.8 噪声实验分析

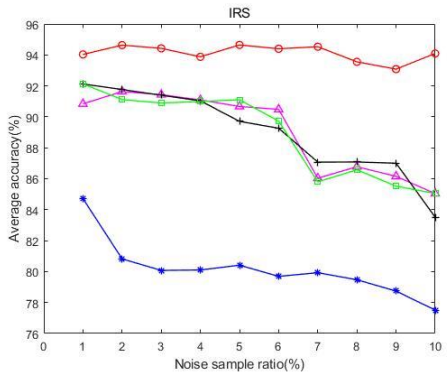
在 Self-training 算法的迭代过程中，一旦错误标记的样本被添加到训练过程中，误差就会在迭代过程中不断扩大。本文所提出的 STRNG 能够从候选高置信度样本中筛选出误标记样本。在噪声实验中，选取 10% 的标注样本作为训练集，并随机分配其中 1% 到 10% 的错误标签。Accuracy 作为性能评估指标。表 3.6 显示了每个数据集在噪声比率 1% 到 10%下的 Accuracy 平均值。图 3.6 显示了不同噪声比例下的各个算法的分类 Accuracy 值。

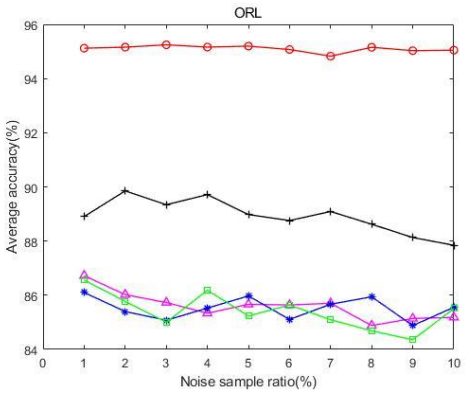
表 3.6　不同比例下个算法噪声分类 Accuracy（平均值±标准差（排名））

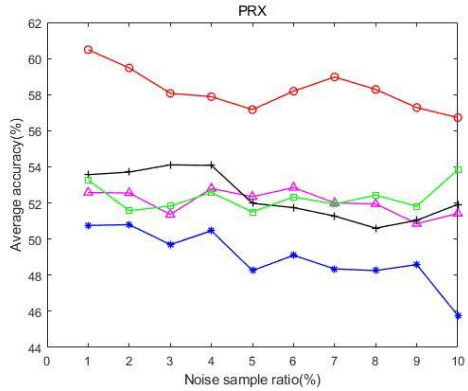
	STDP	STDPCEW	STDPDE	SETRED	STRNG
GLS	74.35±3.75(4)	74.99±3.95(3)	75.52±4.04(2)	72.47±2.76(5)	**80.92±4.25(1)**
CLE	68.30±5.90(5)	69.86±5.30(3)	75.23±6.06(2)	69.50±6.23(4)	**79.22±3.56(1)**
ECI	84.26±3.35(4)	84.80±3.09(3)	87.85±3.23(2)	84.08±2.19(5)	**91.29±1.87(1)**
HAN	58.06±3.68(4)	56.09±3.47(5)	58.55±6.9(3)7	58.80±3.86(2)	**62.27±6.75(1)**
ORL	85.60±1.08(3)	88.92±1.03(2)	85.52±1.60(4)	85.40±0.35(5)	**95.10±0.35(1)**
PRX	52.07±2.69(4)	52.40±2.89(2)	49.00±3.73(5)	52.3±2.21(3)	**58.25±8.43(1)**
WDC	85.31±2.71(3)	85.15±3.22(5)	90.53±2.00(2)	85.24±2.65(4)	**91.38±1.96(1)**
IRS	89.01±3.82(2)	88.99±4.20(3)	80.15±6.59(5)	88.89±4.19(4)	**94.13±2.67(1)**
WCS	79.00±3.72(4)	79.27±3.17(3)	**86.32±2.73(1)**	78.42±3.51(5)	85.39±2.39 (2)
WIN	94.81±0.65(4)	94.63±0.98(5)	97.97±0.44(2)	95.13±0.08(3)	**98.46±0.25(1)**
TCE	64.16±2.00(2)	61.91±3.98(4)	58.81±4.76(5)	63.88±2.22(3)	**66.38±5.11(1)**
WFM	80.57±1.18(4)	78.73±1.38(5)	81.82±0.86(2)	80.68 1.42(3))	**84.11±1.26(1)**
UPS	80.57±1.1(4)	78.73±1.38(5)	81.82±0.86(2)	80.68±1.42(3)	**84.11±1.26(1)**
HPS	53.68±5.23(4)	54.42±3.57(3)	53.18±9.78(5)	54.45±3.44(2)	**65.41±14.07(1)**
WSR-test	~	+	+	~	N/A
Ave.ACC	76.65	75.80	64.24	76.63	81.84
Ave.std	2.89	2.95	3.88	2.58	3.83

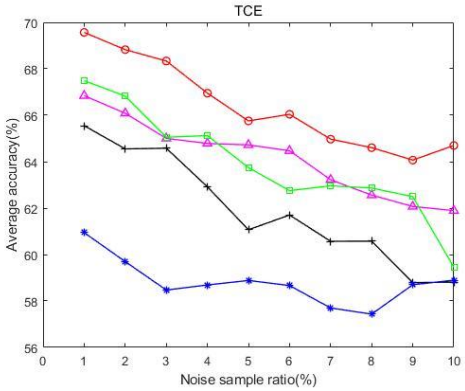


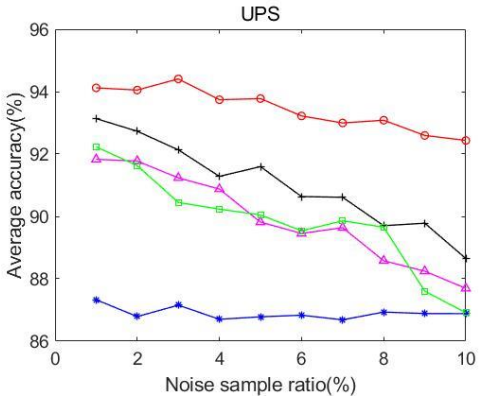


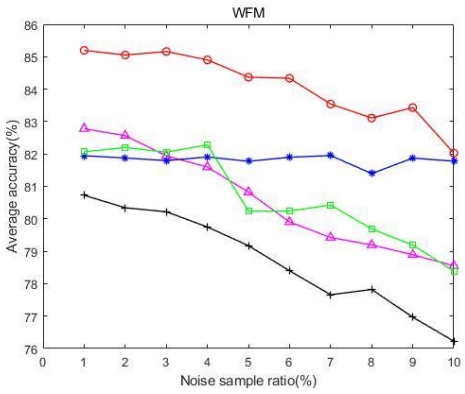

IRS
Average accuracy(%)
Noise sample ratio(%)

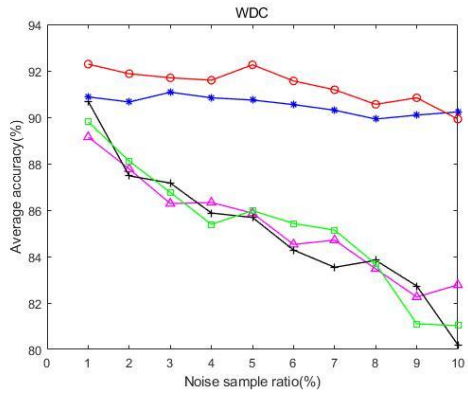

ORL
Average accuracy(%)
Noise sample ratio(%)

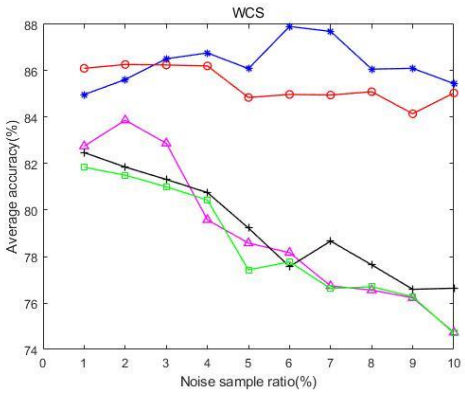

PRX
Average accuracy(%)
Noise sample ratio(%)


TCE
Average accuracy(%)
Noise sample ratio(%)


UPS
Average accuracy(%)
Noise sample ratio(%)


WFM
Average accuracy(%)
Noise sample ratio(%)


WDC
Average accuracy(%)
Noise sample ratio(%)


WCS
Average accuracy(%)
Noise sample ratio(%)

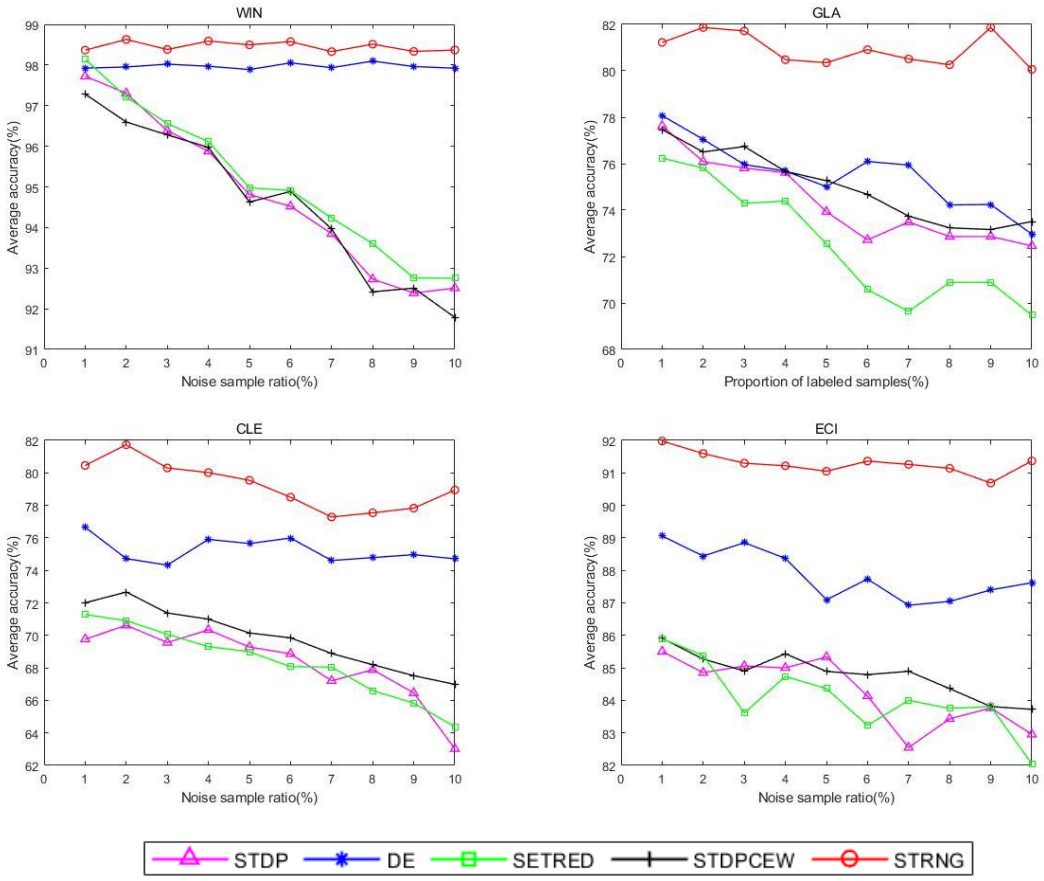


图 3.6 不同比例噪声下各算法分类性能

从不同比率的噪音实验结果中，本文可以得出以下结论：

（1） STRNG 的分类准确率高于 STDPCEW、 SETRED、STDP 和 STDPDE 在 13 个数据集上。STRNG 在 WCS 数据集上的分类准确率仅略低于 STDPDE。STRNG 在 14 个数据集上的平均分类准确率为 81.84%，分别比 STDP、STDCPEW、STDPDE 和 SETRED 高 5.19%、6.04%、5.60%和 5.21%。

（2） 表 5 显示，STRNG 算法在 ECI、ORL、IRS、WDC、WIN 和 UPS 数据集上的分类准确率超过 90%。此外，在这 6 个数据集上，STRNG 的分类准确率分别比排名第二的算法高 7.21%、9.7%， 13.98%、6.23%、3.83%和 6.54%。

3.9 运行时间分析

SETRED 主要是构建 RNG 起总的时间复杂度为$O\left(n^3\right)$。STDCPEW 算法主要需要时间来计算 CEW，其总体时间复杂度为$O\left(n^3\right)$。STDP 的时间复杂度主要

是计算和的时间，其时间复杂度为$O\left(n^2\right)$。STDPDE 是$O\left(n^2\right)$以计算未标记样本的最近邻居。每个算法运行 50 次以计算平均时间消耗。3.7 显示每个算法的运行时间(秒)。

表 3.7 运行时间(排名)

	STDP	**STDPCEW**	**STDPDE**	**SETRED**	**STRNG**
GLS	0.12(2)	0.39(3)	0.02(1)	1.74(5)	0.85(4)
CLE	0.07(2)	0.37(3)	0.04(1)	1.26(5)	0.71(4)
ECI	0.08(2)	0.44(3)	0.04(1)	1.44(5)	0.80(4)
HAN	0.08(2)	0.29(3)	0.02(1)	1.18(5)	0.88(4)
ORL	0.25(1)	1.99(3)	0.55(2)	8.01(5)	2.72(4)
PRX	0.04(2)	0.14(3)	0.02(1)	1.11(4))	1.51(5)
IRS	0.04(1)	0.18(3)	0.09(2)	2.37(5)	0.66(4)
WDC	0.33(2)	2.23(4)	0.19(1)	2.95(5)	1.27(3)
WCS	0.25(1)	1.19(4)	0.10(2)	1.32(5)	1.11(3)
WIN	0.72(2)	105.32(5)	0.62(1)	22.05(4)	5.46(3)
TCE	0.13(2)	2.17(5)	0.06(1)	1.99(4)	1.71(3)
WFM	1.58(2)	302.67(5)	1.28(1)	2.09(3)	3.54(4)
UPS	1.60(1)	157.53(5)	1.75(2)	38.95(4)	7.83(3)
HPS	0.12(2)	0.25(3)	0.03(1)	1.04(5)	0.62(4)
Ave.time	0.39	41.08	0.34	6.25	2.12

从表 3.7 中，本文可以得出以下结论：STRNG 在 self-training 算法迭代的过程中需要通过平行分割构建树，要遍历包含两个样本间最小块，此外还需要构建原型树这需要消耗一定的计算时间。总的来说，STALN 在 18 个数据集上的平均时间排名第三，实验的结果与理论分析一致。

3.10 本章小结

在节中，本文提出了一种基于相对节点图的鲁棒 Self-training 算法 STRNG。STRNG 使用基于质量的相异性来计算样本之间的距离，解决了欧几里得距离不

适合复杂数据的缺点。此外，本文构建了一个原型树来探索数据的底层结构。接下来，本文计算每个样本的相对节点图，并使用它来提高高置信度样本的质量。实验表明，所提出的 STRNG 算法比现有算法具有更好的分类性能。90%置信水平下 Wilcoxon 的测试结果表明，STRNG 的分类有效性得到了显著提高。

与比较算法相比，STRNG 显著提高了 Self-training 算法的性能。然而，STRNG 也存在以下缺点：（1）在不平衡的小型数据集中，初始标记数据的数量会影响分类器的性能。这是因为比例很小的一类有标记样本在近亲节点图的编辑阶段周围易有大量的切切边，导致这些样本易于被认为噪声样本。（2） STRNG 平行分隔生成树并遍历每棵树以找到最小的块，这需要大量的时间消耗。（3） STRNG 有两个参数。当面对不同的数据集时，很难确定的适当值。

4. 自适应局部邻居过滤的 self-training 算法

4.1 算法描述

为了充分挖掘无标签样本与有标签样本的关系，设计了自适应局部邻居过滤的 self-training 算法（STALN）。首先，基于有标签样本的全局信息 自适应得到每个无标签样本的局部邻居，并利用无标签样本局部邻居的信息分配该样本一个标签，通过与基分类器分配的标签对比，筛选出具有一致标签的无标签样本添加进高置信度样本集。STALN 算法描述如算法 4。

算法 4：STALN 算法

输入： 有标签样本集 L ，无标签样本集 U

输出： 分类器 H

While $U \neq \varnothing$ **Do**

  初始化高置信度样本集 $S = \varnothing$

  利用训练一个分类器 H

  For x_i in U

    利用 H 为 x_i 分配标签 T_i

    使用公式（1）计算样本 x_i 的自适应 距离 $AD(x_i)$；

    If $T_j = l_{LNN(x_i)}$

      $S = S \cup x_i$

    End If

  End For

  $L = L \cup S; U = U - S$

End While

Return H

4.2 算法思想介绍

现实生活中有标签样本的数量很少，少量的有标签样本难以训练有效的分类器。为了提高分类器 的性能，需要充分利用全部有标签样本的信息以及 挖掘无

标签样本潜在的信息。STALN 在 self-training 迭代的过程中，利用有标签样本的全局信 息，自适应地获得无标签样本的局部邻居。为了清晰地描述 STALN 算法，进行了如下定义：

定义 1 自适应距离

令 $AD(x_i)$ 为无标签样本 x_i 的自适应距离，样本 x_j 为有标签样本，自适应距离的定义如下：

$$AD(xi)=\frac{1}{|L|}\sum_{j=1}^{|L|}D\left(x_i,x_j\right) \tag{12}$$

其中 $|L|$ 为有标签样本的数量，表示无标 $D\left(x_i,x_j\right)$ 签样本 x_i 与有标签样本 x_j 之间的欧式距离。

定义 2 自适应局部邻居

令 $LNN\left(x_i\right)$ 为样本 x_i 的自适应局部邻居，自适应局部邻居定义如下：

$$LNN\left(x_i\right)=\left\{x_j \mid D\left(x_i,x_j\right)\le AD\left(x_i\right),x_j\in L,x_i\in U\right\} \tag{13}$$

由公式（2）看出，$LNN\left(x_i\right)$ 是一个有标签样本集合。无标签样本 x_i 考虑全部有标签样本的信息，自适应地获取局部邻居。

为了清晰地描述 STALN 算法自适应选择局部邻居的计算过程，随机生成两类高斯样本，图 4.1 展示了无标签样本 x_8 的自适应局部近邻 $LNN\left(x_8\right)$ 计算过程。

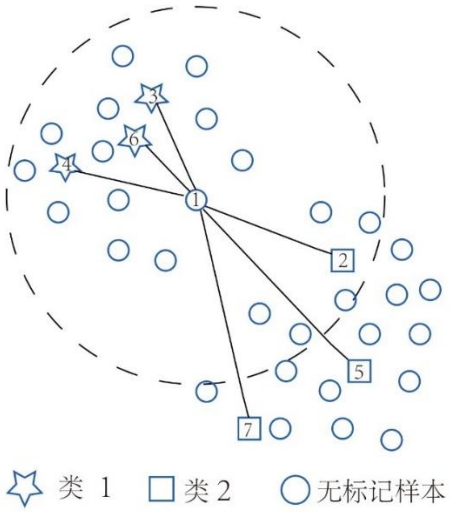


图 4.1 无标签样本的局部邻居计算示意图

图 4.1 中圆圈代表无标签样本，菱形和方块分别代表两个类的有标签样本。自适应局部邻居的计算一般分为两步，以无标签样本 x_1 为例，计算无标签样本 x_1 与所有的有标签样本的自适应距离为 $AD(x_1)$，以样本 x_8 为圆心和自适应距离 $AD(x_1)$ 为半径画一个圆。统计圆内有标签样本为 x_8 的自适应 局部邻居，即 $LNN(x_1)=\{x_2,x_6,x_3,x_4\}$。可以看出 $LNN(x_1)$ 内所有样本都是有标签样本，与传统的 K 近邻算法选取邻居的方式相比，所提算法能够自 适应地选取无标签样本 x_1 的局部邻居 $LNN(x_1)$。

定义 3 局部邻居标签

令 $l_{LNN(x_i)}$ 为 $LNN(x_i)$ 的标签，计算如下：

$$y_{LNN(x_i)}=\left\{y_j \mid x_j \in LNN(x_i)\right\} \tag{14}$$

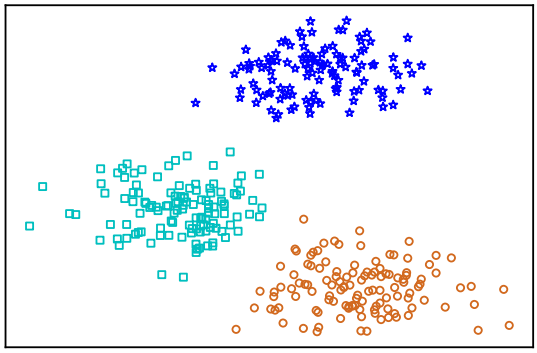
$$l_{LNN(x_i)}=Mode\left(y_{KNN(x_i)}\right) \tag{15}$$

其中 $y_{LNN(x_i)}$ 为 $LNN(x_i)$ 的标签集合，$Mode(\bullet)$ 是用来计算集合众数的函数。通过（4）可以看出 $l_{LNN(x_i)}$ 示集合 $y_{LNN(x_i)}$ 中的众数标签。

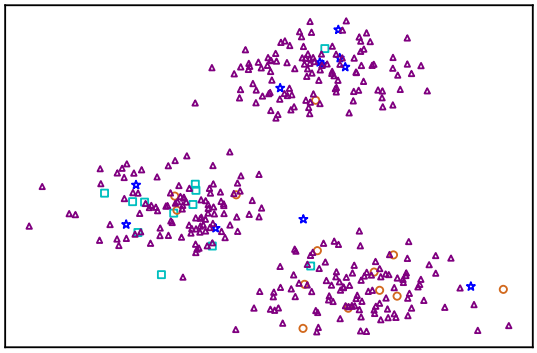
4.3 迭代过程展示

为了更加清晰地展示 STALN 算法的迭代过程，随机生成样本数量为 300 的 3

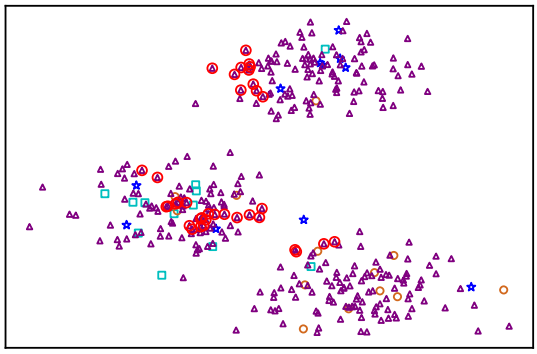
类别高斯分布样本， 原始样本分布如图 4.2(a)。随机选取 10%的样本作为 有标签样本并且分配 4%的噪声样本，其余为无标 签样本，如图 4.2(b)。记录并展示了 STALN 的迭代过程。



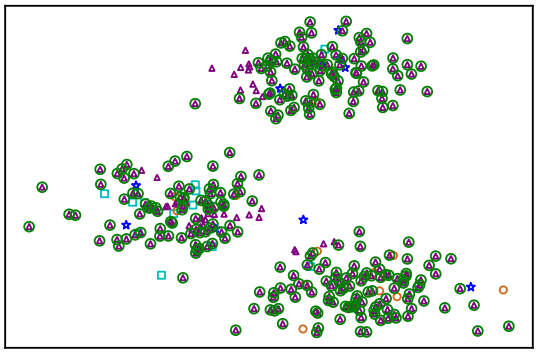
(a) 原始样本分布



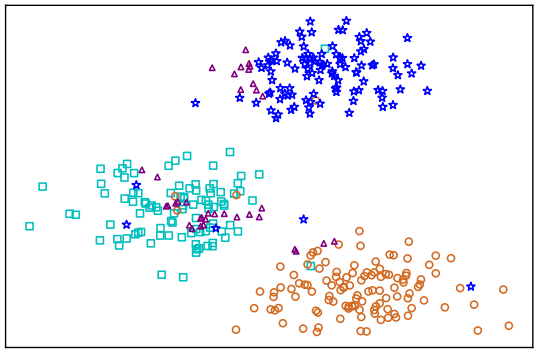
(b) 初始 90%的无标签样本并分配 4%的噪声样本



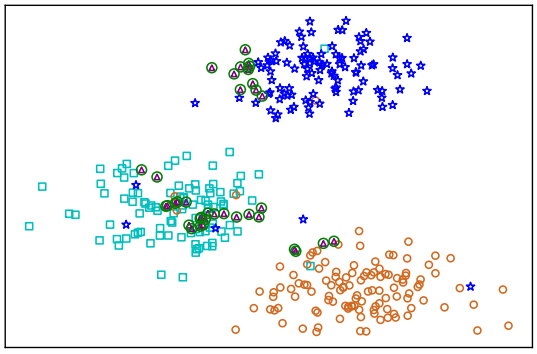
(c) 第 1 次迭代误分类样本



(d) 第 1 次迭代高置信样本



(e) 第 1 次迭代完成后样本分布



(f) 第 2 次迭代高置信样本

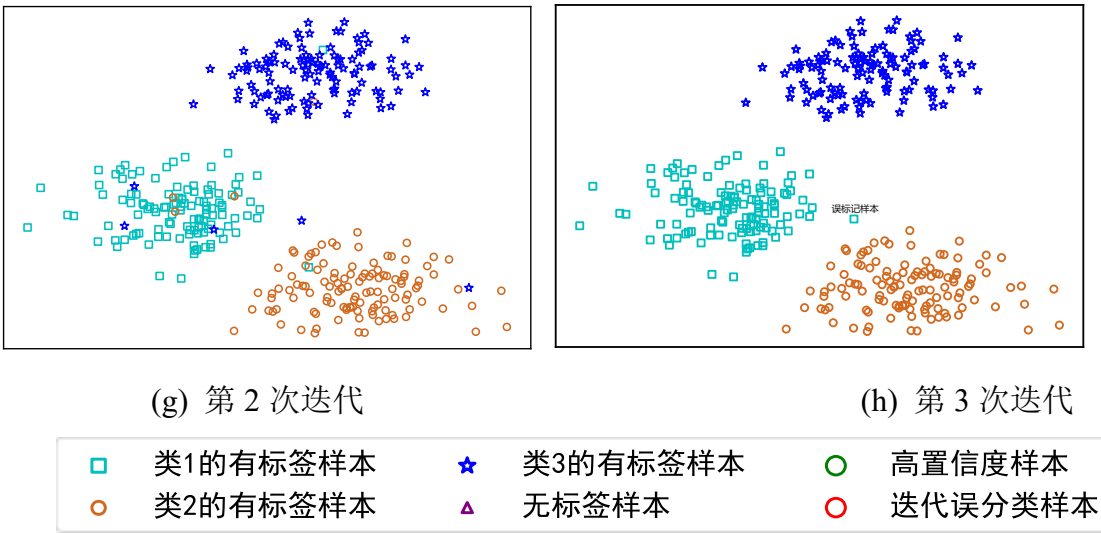


图 4.2　STALN 迭代过程

从图 4.2 中，本文可以看到 STALN 进行了两轮迭代。在第一次迭代中，选择 252 个未标记的样本作为高置信度样本，其分布如图 4.2(c)所示。从图 4.2(c)可以看出，自适应局部邻居和分类器分配的标签不一致的 72 个未标记样本被认为是有争议的样本。从图 4.2(d)中有争议样本的分布可以看出，STALN 选择的有争议样本通常位于每个类别的边界或 z 在噪声样本周围。这表明 STALN 具有极好的识别噪声的能力。如图 4.2(f)所示，来自第一次迭代的 72 个有争议的样本被选为第二次迭代中的高置信度样本。图 4.2(h)显示，最终输出分类器预测的样本标签只有一个异常值预测误差，并且通过比较图 4.2(a)和 4.(b)对噪声样本标签进行了校正。STANN 考虑样本的全局信息，利用未标记样本和所有标记样本之间的关系自适应地获得局部邻居样本。它不仅迭代速度快，而且具有良好的数据编辑能力。

4.4 时间复杂度分析

STALN 算法充分考虑了有标签样本的全局信息，因此在每次迭代过程中需要计算每个无标签样 本与全部有标签样本之间的距离。假设 n 是样本的数量，l 是有标签样本的数量，则 $n-l$ 是无标签样本的数量。STALN 的时间复杂度计算主要包含三部分：(1)使用欧式距离构建相似度矩阵的时间复杂度为 $O\left(n^2\right)$；(2)在迭代的过程中，计算无标签样本的自适应局部邻居的时间复杂度为 $O\left(l\left(n-l\right)\right)$；

(3) 计算局部邻居标签并选择高置信度样本需要$O(n-l)$。

因此，STALN 算法总的时间复杂度为$O\left(n^2+(n-l)+l(n-l)\right)$。

4.5 实验设置

4.5.1 实验环境

所有实验基于 64 位的 Windows 10 系统 Matlab 2019b，处理器为 IntelCore i5，内存为 8G。

4.5.2 数据集和对实验设置

选择 STDP、STOPF、SETRED 和 STDPNaN，5 个相关的 self-training 作为对比算法并根据算法的原文复现代码，对比算法的参数根据原文设置。本文在 18 个公开数据集上进行实验来验证本文提出算法的有效性。分别是 GLA、SPH、HEA、ILPD、AUS、BRE、TRA、Pima、DIA、GER、CMC、ORL、PAL、AR、FER、YALEB、UMI、MPE 数据集。

在 95%的置信水平下进行威尔科克森符号秩检验(Wilcoxon signed-rank test, Wilcoxon)，以验 STALN 实验结果的有效性。符号“+”，“-”分别表 示 STALN 显著优于和低于对比算法，符号“~”表示 STALN 与对比算法没有显著差异。

表 4.1　实验参数设置

算法	参数
STDP[46]	$P_\alpha = 2$
STDPNAN[54]	无参数
SETRED[36]	阈值 $\theta = 0.1$
STOPF[56]	无参数
STALN	无参数

4.6 分类性能分析

本节选择以 3NN、SVM 和 itree 为基分类器在 16 个基准数据集进行实验。每次实验随机选择 10%的样本构建有标签样本集，其余样本构建无标签样本集。本文选择准确率(Accuracy)和 F 分数(F1-score)作为 算法的评价指标。在不同的基分类器下，记录所有 算法在 16 个数据集上运行 50 次的平均值和标准差，实验结果见下表，在每个数据集上将分类性 能最好的结果用粗体字标记，括号中的数字是每个算法的排名。AVE.ACC、AVE. F_1-score 和 AVE.STD 分别表示平均 Accuracy、平均 F_1-score 和平均标准差。

表 4.2 3NN 基分类器下每个算法的 Accuracy（平均值±标准差（排名））

	3NN	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	60.99±1.46(3)	60.52±1.27(5)	61.57±1.04(2)	60.45±1.43(6)	60.99±1.01(4)	**63.77±3.12(1)**
BRE	55.88±2.34(2)	54.3±3.55(6)	55.65±4.71(3)	55.47±3.0(5)	55.49±2.21(4)	**59.42±5.15(1)**
UMI	77.97±3.26(6)	78.33±2.34(5)	79.11±3.59(4)	85.91±2.4(2)	83.5±2.97(3)	**90.72±0.88(1)**
ILPD	60.03±2.94(5)	59.72±1.88(6)	61.3±1.8(2)	60.17±3.44(4)	60.66±3.96(3)	**73.60±9.10(1)**
AUS	63.57±3.45(4)	63.21±3.29(6)	63.33±4.13(5)	64.48±3.83(3)	64.51±2.88(2)	**72.04±3.13(1)**
YALEB	69.24±1.41(5)	68.78±0.26(6)	73.84±0.71(4)	79.33±1.65(2)	75.28±0.52(3)	**89.86±2.11(1)**
ORL	83.27±1.82(5)	83.04±2.3(6)	84.63±1.83(4)	89.11±2.55(2)	86.9±2.23(3)	**94.69±1.33(1)**
FER	83.88±0.82(6)	84.48±1.2(5)	86.69±0.49(4)	89.31±0.44(2)	88.06±1.25(3)	**97.42±0.69(1)**
PAL	85.51±0.88(5)	85.36±1.27(6)	87.89±0.99(4)	91.18±0.92(2)	89.09±0.59(3)	**97.62±0.36(1)**
AR	77.94±1.26(5)	77.72±0.75(6)	80.89±1.47(4)	86.18±0.66(2)	85.02±0.8(3)	**96.85±0.75(1)**
GER	54.93±2.61(2)	54.56±2.18(5)	54.93±2.44(3)	54.77±2.47(4)	54.46±2.35(6)	**65.91±3.99(1)**
HEA	63.08±3.25(3)	62.99±3.79(4)	62.4±4.11(5)	61.79±5.08(6)	64.64±4.18(2)	**66.96±7.51(1)**
SPH	63.53±4.57(4)	61.35±2.71(5)	61.26±5.59(6)	64.74±0.59(3)	68.42±8.55(2)	**72.49±13.66(1)**
PIM	66.73±3.18(5)	67.49±2.31(4)	67.88±2.67(2)	64.95±3.51(6)	67.55±2.43(3)	**72.26±8.04(1)**
GLA	77.79±5.26(6)	78.51±5.1(5)	79.89±4.81(4)	81.21±3.42(3)	81.73±4.73(2)	**82.59±3.8(1)**
DIA	66.4±3.31(4)	66.05±2.95(5)	67.09±2.86(3)	65.03±3.23(6)	67.12±2.85(2)	**70.57±6.7(1)**
BTSC	58.42±3.97(4)	59.27±4.66(2)	57.82±3.8(5)	58.59±4.63(3)	57.73±3.66(6)	**62.93±10.13(1)**
MPE	80.15±2.05(5)	79.94±2.56(6)	83.2±1.46(4)	86.31±1.38(3)	86.52±1.34(2)	94.39±1.97(1)
Wilcoxon	+	+	+	+	+	N/A
平均值	69.41	69.20	70.52	72.17	72.09	79.12

表 4.3 3NN 基分类器下每个算法的 F_1-score（平均值±标准差（排名））

	3NN	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	55.1±2.0(4)	54.49±1.75(5)	55.92±1.36(2)	54.38±1.91(6)	55.14±1.4(3)	**58.62±4.68(1)**
BRE	55.88±2.34(2)	54.3±3.55(6)	55.65±4.71(3)	55.47±3.0(5)	55.49±2.21(4)	**59.42±5.15(1)**
UMI	69.6±4.93(5)	67.64±4.82(6)	69.94±7.52(4)	79.03±4.29(2)	75.33±4.86(3)	**84.69±1.59(1)**
ILPD	60.03±2.94(5)	59.72±1.88(6)	61.3±1.8(2)	60.17±3.44(4)	60.66±3.96(3)	**73.6±9.1(1)**
AUS	63.57±3.45(4)	63.21±3.29(6)	63.33±4.13(5)	64.48±3.83(3)	64.51±2.88(2)	**72.04±3.13(1)**
YALEB	55.75±2.39(5)	53.89±1.56(6)	64.27±1.45(4)	73.06±2.19(2)	65.98±1.3(3)	**85.06±2.62(1)**
ORL	52.74±5.14(6)	53.92±5.82(5)	57.66±3.74(4)	68.64±6.72(2)	63.49±7.25(3)	**74.99±9.65(1)**
FER	30.4±3.04(6)	34.73±3.94(5)	37.74±1.93(4)	44.32±9.16(3)	45.5±5.37(2)	**66.81±5.67(1)**
PAL	77.63±1.76(6)	78.05±2.05(5)	81.29±2.25(4)	85.89±2.17(2)	82.66±1.09(3)	**95.38±0.9(1)**
AR	51.06±2.68(5)	47.96±0.91(6)	59.36±2.66(4)	67.44±3.51(2)	64.14±3.1(3)	**88.36±2.83(1)**
GER	54.93±2.61(2)	54.56±2.18(5)	54.93±2.44(3)	54.77±2.47(4)	54.46±2.35(6)	**65.91±3.99(1)**
HEA	63.08±3.25(3)	62.99±3.79(4)	62.4±4.11(5)	61.79±5.08(6)	64.64±4.18(2)	**66.96±7.51(1)**
SPH	63.53±4.57(4)	61.35±2.71(5)	61.26±5.59(6)	64.74±0.59(3)	68.42±8.55(2)	**72.49±13.66(1)**
PIM	66.73±3.18(5)	67.49±2.31(4)	67.88±2.67(2)	64.95±3.51(6)	67.55±2.43(3)	**72.26±8.04(1)**
GLA	62.65±9.54(6)	64.33±5.98(5)	64.94±9.1(4)	65.73±10.32(3)	66.39±9.45(2)	**66.98±9.95(1)**
DIA	66.4±3.31(4)	66.05±2.95(5)	67.09±2.86(3)	65.03±3.23(6)	67.12±2.85(2)	**70.57±6.7(1)**
BTSC	58.42±3.97(4)	59.27±4.66(2)	57.82±3.8(5)	58.59±4.63(3)	57.73±3.66(6)	**62.93±10.13(1)**
MPE	63.86±3.63(5)	62.72±5.09(6)	68.72±2.32(4)	72.46±2.69(3)	73.23±2.66(2)	**85.7±2.96(1)**
Wilcoxon	+	+	+	+	+	N/A
平均值	59.52	59.26	61.75	64.49	64.02	73.49

表 4.4　SVM 基分类器下每个算法的 Accuracy（平均值±标准差（排名））

	SVM	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	**64.54±2.64(1)**	63.56±2.64(4)	63.27±4.41(5)	62.50±1.45(6)	63.92±2.13(3)	64.29±3.71(2)
BRE	47.69±23.47(6)	59.54±17.15(5)	66.71±4.45(2)	60.83±15.09(4)	61.43±13.74(3)	71.74±8.79(1)
UMI	82.43±3.33(6)	84.07±0.96(5)	91.49±1.17(1)	86.55±0.36(4)	88.81±0.68(3)	89.14±2.78(2)
ILPD	37.33±9.13(6)	72.89±4.61(3)	71.86±2.73(4)	74.48±6.25(2)	59.04±7.87(5)	75.62±6.04(1)
AUS	30.11±3.17(6)	56.93±8.36(4)	57.04±5.72(3)	50.72±5.95(5)	60.51±7.66(2)	**63.37±11.12(1)**
YALEB	93.64±0.30(6)	94.87±0.01(4)	94.88±0.30(3)	94.45±0.60(5)	**94.90±0.06(1)**	94.88±0.01(2)
ORL	93.90±1.02(6)	95.13±0.01(5)	95.13±0.21(2)	95.13±0.07(3)	95.13±0.11(4)	**95.13±0.01(1)**
FER	98.73±0.14(6)	99.01±0.12(5)	99.01±0.20(2)	99.01±0.05(3)	99.01±0.01(4)	**99.01±0.01(1)**
PAL	97.62±0.17(6)	98.01±0.02(4)	97.92±0.20(5)	98.02±0.01(3)	98.02±0.05(2)	**98.02±0.02(1)**
AR	95.55±1.90(6)	97.58±0.83(5)	**98.35±0.09(1)**	97.94±0.33(2)	97.76±0.17(3)	97.68±0.18(4)
GER	34.07±0.37(6)	66.54±18.17(2)	66.0±9.49(3)	64.37±16.89(4)	60.08±9.71(5)	**73.19±6.37(1)**
HEA	25.45±3.03(6)	31.93±13.79(4)	47.13±9.33(2)	29.17±13.3(5)	**47.40±6.07(1)**	44.09±16.22(3)
SPH	40.16±7.29(6)	65.64±21.27(3)	50.56±3.65(4)	70.12±14.48(2)	48.12±14.61(5)	**79.40±9.79(1)**
PIM	32.51±6.06(5)	22.24±16.65(6)	**52.07±15.96(1)**	33.04±24.58(3)	46.86±15.29(2)	32.65±22.95(4)
GLA	76.32±7.06(6)	79.32±5.11(5)	80.16±3.95(4)	81.51±5.53(2)	80.83±3.73(3)	**84.74±1.46(1)**
DIA	31.36±0.23(5)	25.61±19.6(6)	**56.55±14.95(1)**	36.12±23.68(3)	47.67±16.84(2)	32.42±23.11(4)
BTSC	73.08±8.93(6)	74.04±9.49(4)	76.2±2.62(2)	**76.68±1.42(1)**	73.09±10.37(5)	76.05±6.80(3)
MPE	95.41±1.20(5)	95.29±0.57(6)	**97.42±0.34(1)**	96.6±0.83(3)	95.77±0.38(4)	97.29±0.07(2)
Wilcoxon	+	+	~	+	~	**N/A**
平均值	63.88	71.23	75.65	72.62	73.24	76.04

表 4.5　SVM 基分类器下每个算法的 F_1-score（平均值±标准差（排名））

	SVM	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	56.93±5.42(3)	**57.75±5.00(1)**	45.57±8.60(6)	55.86±4.30(4)	57.49±4.47(2)	46.97±8.40(5)
BRE	47.69±23.47(6)	59.54±17.15(5)	66.71±4.45(2)	60.83±15.09(4)	61.43±13.74(3)	71.74±8.79(1)
UMI	26.02±15.77(5)	27.07±11.03(4)	22.22±22.06(6)	**38.51±6.95(1)**	34.65±14.12(3)	34.80±5.32(2)
ILPD	37.33±9.13(6)	72.89±4.61(3)	71.86±2.73(4)	74.48±6.25(2)	59.04±7.87(5)	**75.62±6.04(1)**
AUS	30.11±3.17(6)	56.93±8.36(4)	57.04±5.72(3)	50.72±5.95(5)	60.51±7.66(2)	**63.37±11.12(1)**
YALEB	1.10±0.08(6)	2.58±0.10(4)	2.65±0.01(2)	2.20±0.61(5)	4.18±2.25(1)	2.65±0.01(3)
ORL	0.85±0.18(6)	2.50±0.01(5)	2.50±0.21(2)	2.50±0.43(3)	2.50±0.38(4)	**2.50±0.01(1)**
FER	0.13±0.02(6)	0.50±0.05(4)	0.44±0.08(5)	0.50±0.10(2)	0.50±0.01(3)	**0.50±0.01(1)**
PAL	0.38±0.06(6)	0.61±0.29(5)	0.72±0.08(4)	**1.07±0.33(1)**	0.81±0.10(2)	0.76±0.01(3)
AR	3.78±5.12(3)	4.95±2.63(2)	0.83±0.01(6)	1.38±0.62(5)	**9.62±1.85(1)**	2.48±1.72(4)
GER	34.07±0.37(6)	66.54±18.17(2)	66.00±9.49(3)	64.37±16.89(4)	60.08±9.71(5)	**73.19±6.37(1)**
HEA	25.45±3.03(6)	31.93±13.79(4)	47.13±9.33(2)	29.17±13.30(5)	**47.40±6.07(1)**	44.09±16.22(3)
SPH	40.16±7.29(6)	65.64±21.27(3)	50.56±3.65(4)	70.12±14.48(2)	48.12±14.61(5)	**79.40±9.79(1)**
PIM	32.51±6.06(5)	22.24±16.65(6)	**52.07±15.96(1)**	33.04±24.58(3)	46.86±15.29(2)	32.65±22.95(4)
GLA	51.23±17.7(4)	50.44±14.82(5)	42.84±15.48(6)	59.94±12.49(2)	54.06±14.08(3)	**68.24±2.43(1)**
DIA	31.36±0.23(5)	25.61±19.60(6)	**56.55±14.95(1)**	36.12±23.68(3)	47.67±16.84(2)	32.42±23.11(4)
BTSC	73.08±8.93(6)	74.04±9.49(4)	76.20±2.62(2)	**76.68±1.42(1)**	73.09±10.37(5)	76.05±6.80(3)
MPE	**25.54±16.58(1)**	25.17±10.42(2)	17.88±23.27(5)	6.52±6.61(6)	19.69±12.34(4)	20.99±18.41(3)
Wilcoxon	+	+	~	+	~	**N/A**
平均值	28.76	35.94	37.77	36.89		

表 4.6 CART 基分类器下每个算法的 Accuracy（平均值±标准差（排名））

	CART	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	61.02±1.55(6)	61.24±1.83(5)	61.75±1.89(3)	61.34±1.61(4)	62.15±1.72(2)	**64.64±1.28(1)**
BRE	92.40±1.59(3)	92.37±1.74(4)	92.24±1.64(5)	92.12±1.60(6)	92.43±2.12(2)	**92.56±1.48(1)**
UMI	**79.37**±1.45(1)	79.21±2.14(2)	77.98±1.98(4)	73.72±2.05(6)	75.50±1.82(5)	**78.06±2.01(3)**
ILPD	59.40±3.21(6)	61.10±2.88(3)	60.74±3.33(5)	60.88±2.23(4)	61.16±2.97(2)	**69.90±10.63(1)**
AUS	82.0±3.18(5)	82.06±3.39(3)	82.0±4.04(4)	81.19±3.64(6)	82.14±3.15(2)	**82.20±2.66(1)**
YALEB	72.07±1.49(3)	72.63±1.39(2)	71.78±0.01(4)	67.04±0.87(6)	70.11±1.65(5)	**77.77±6.03(1)**
ORL	83.57±1.23(2)	83.43±0.84(3)	83.01±0.92(4)	80.19±1.35(6)	82.23±1.75(5)	**83.63±1.28(1)**
FER	90.10±0.76(3)	**90.54±0.66(1)**	89.17±0.80(4)	86.21±0.77(6)	87.34±0.51(5)	90.20±1.74(2)
PAL	78.87±1.08(4)	79.74±0.90(2)	79.40±0.36(3)	74.60±1.17(6)	77.99±1.08(5)	**80.85±4.24(1)**
AR	**82.76±1.40(1)**	82.58±0.61(3)	81.12±0.49(4)	78.15±1.20(6)	80.99±1.16(5)	82.68±3.57(2)
GER	63.73±2.52(2)	63.36±3.18(6)	63.60±3.09(3)	63.56±1.87(4)	63.47±3.07(5)	**65.24±5.87(1)**
HEA	**73.70**±5.46(1)	73.56±3.76(2)	73.42±5.03(3)	72.51±7.37(6)	72.9±5.59(5)	73.24±2.45(4)
SPH	61.10±6.07(4)	61.52±5.73(3)	60.20±5.25(6)	60.24±6.95(5)	61.61±4.41(2)	**67.75±6.16(1)**
PIM	67.19±3.14(6)	68.49±3.07(3)	68.29±3.23(4)	68.81±3.96(2)	67.86±3.23(5)	**72.16±3.91(1)**
GLA	78.53±3.50(5)	**80.34±4.20(1)**	79.43±2.99(3)	73.66±6.14(6)	78.92±2.10(4)	80.24±3.56(2)
DIA	67.86±3.30(6)	68.26±2.85(4)	68.6±3.14(3)	68.12±2.21(5)	68.66±3.09(2)	**71.84±4.14(1)**
BTSC	62.69±4.71(4)	63.08±4.32(2)	62.87±4.13(3)	62.07±4.67(6)	62.25±3.95(5)	**64.48±4.93(1)**
MPE	**78.33±1.19(1)**	77.88±1.28(2)	75.49±0.77(4)	75.32±0.49(5)	74.94±1.66(6)	77.23±2.02(3)
Wilcoxon	~	~	+	+	+	**N/A**
平均值	74.15	74.52	73.95	72.21	73.48	76.37

表 4.7 CART 基分类器下每个算法的 F_1-score（平均值±标准差（排名））

	CART	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	55.06±2.11(6)	55.38±2.47(5)	56.02±2.56(3)	55.48±2.1(4)	56.57±2.28(2)	**60.17±1.69(1)**
BRE	92.40±1.59(3)	92.37±1.74(4)	92.24±1.64(5)	92.12±1.6(6)	92.43±2.12(2)	**92.56±1.48(1)**
UMI	58.71±5.07(5)	**70.15±4.94(1)**	68.14±2.98(2)	57.05±2.44(6)	62.85±1.65(4)	66.62±5.51(3)
ILPD	59.40±3.21(6)	61.1±2.88(3)	60.74±3.33(5)	60.88±2.23(4)	61.16±2.97(2)	**69.90±10.63(1)**
AUS	82.00±3.18(5)	82.06±3.39(3)	82.0±4.04(4)	81.19±3.64(6)	82.14±3.15(2)	**82.20±2.66(1)**
YALEB	56.18±2.94(5)	61.94±1.98(2)	60.22±1.41(3)	51.43±1.76(6)	57.48±2.85(4)	**70.59±9.90(1)**
ORL	38.04±5.19(6)	51.85±3.78(3)	**57.92±1.93(1)**	44.08±5.01(5)	45.32±4.02(4)	55.61±6.26(2)
FER	21.26±2.95(6)	**53.12±5.44(1)**	48.89±6.02(3)	28.23±3.50(5)	44.25±5.5(4)	53.07±10.45(2)
PAL	49.39±2.72(6)	61.0±3.22(3)	62.22±2.95(2)	49.78±2.61(5)	55.94±2.5(4)	**65.16±8.21(1)**
AR	36.78±2.19(5)	**55.19**±1.34(1)	42.71±1.94(4)	5.36±2.85(6)	43.84±4.22(3)	54.82±11.03(2)
GER	63.73±2.52(2)	63.36±3.18(6)	63.6±3.09(3)	63.56±1.87(4)	63.47±3.07(5)	**65.24±5.87(1)**
HEA	**73.70±5.46(1)**	73.56±3.76(2)	73.42±5.03(3)	72.51±7.37(6)	72.9±5.59(5)	73.24±2.45(4)
SPH	61.10±6.07(4)	61.52±5.73(3)	60.20±5.25(6)	60.24±6.95(5)	61.61±4.41(2)	**67.75±6.16(1)**
PIM	67.19±3.14(6)	68.49±3.07(3)	68.29±3.23(4)	68.81±3.96(2)	67.86±3.23(5)	**72.16±3.91(1)**
GLA	57.07±7.49(6)	**71.61±4.41(1)**	63.3±7.62(4)	61.5±10.04(5)	71.5±3.41(2)	65.68±8.04(3)
DIA	67.86±3.3(6)	68.26±2.85(4)	68.6±3.14(3)	68.12±2.21(5)	68.66±3.09(2)	**71.84±4.14(1)**
BTSC	62.69±4.71(4)	63.08±4.32(2)	62.87±4.13(3)	62.07±4.67(6)	62.25±3.95(5)	**64.48±4.93(1)**
MPE	41.01±3.68(5)	48.08±3.08(3)	48.16±1.25(2)	40.4±2.28(6)	43.5±2.12(4)	**51.61±7.29(1)**
Wilcoxon	+	~	+	+	+	N/A
平均值	57.98	64.56	63.31	58.49	61.87	

从表 4.2、表 4.3、表 4.4、表 4.5、表 4.6、和表 4.7 的实验结果中，本文可以得到以下结论：

(1)从整体上看，STALN 的分类准确率显著高于对比算法，这充分说明了 STALN 算法的优越性。STALN 充分考虑了数据的全局性，在每一次迭代过程中挖掘全部有标签样本的信息，显著地提升了分类器的性能。

(2)由表 4.2 和表 4.3 本文可以得到，在 18 个数据集上，STALN 以 3NN 作为

基分类，其 Accuracy 全部高于对于算法，其分类准确率相较于 3NN、STDP、SERTRED、STOPF 和 STDPNaN 分别高 9.68%、9.89%、8.57%、6.92%、和 7.00%。此外，STALN 的 F1-score 在 18 个数据集中有 17 个数据集排名第一，相较于对比算法分别高 13.83%、14.09%、11.6%、8.85%、和 9.33%。这些数据表明 STALN 的分类性能在 3NN 基分类器下显著高于对比算法。

(3)由表 4.4 和表 4.5 本文可以看出，STALN 以 SVM 作为基分类器，其 Accuracy 在 18 个数据集中有 9 个数据集排名第一，F1-score 在 18 个数据集有 8 个数据集排名第一。STALN 的分类性能相较于以 3NN 作为基分类器的分类性能略低，但是 STALN 仍然获得了最高的平均 Accuracy 和 F1-score，这说明了 STALN 在 SVM 基分类器下通过考虑数据的全局信息，提升了分类器的性能。

(4)由表 4.6 和表 4.7 本文可以得到，STALN 以 CART 作为基分类器，其 Accuracy 和 F1-score 在 18 个数据集中有 12 个数据集排名第一，并且获得了最高的平均 Accuracy 和 F1-score。这说明了 STALN 以 CART 作为基分类器，充分利用了无标签样本与全部有标签样本的关系，进而能更加准确地筛选出高置信样本，提高算法的分类性能。

(5)在表 4.2 和表 4.3 中，STALN 基于 3NN 基分类器的统计检验结果显著优于其他对比算法。表 4.4 和表 4.5 中，STALN 基于 SVM 基分类器的统计检验结果显著优于 SVM，STDP 和 STOPF，与 SETRED 和 STDPNaN 没有显著差异。表 4.6 和表 4.7 中，STALN 基于 CART 基分类器的统计检验结果显著优于 SETRED，STOPF 和 STDPNaN，与 STDP 没有显著差异。

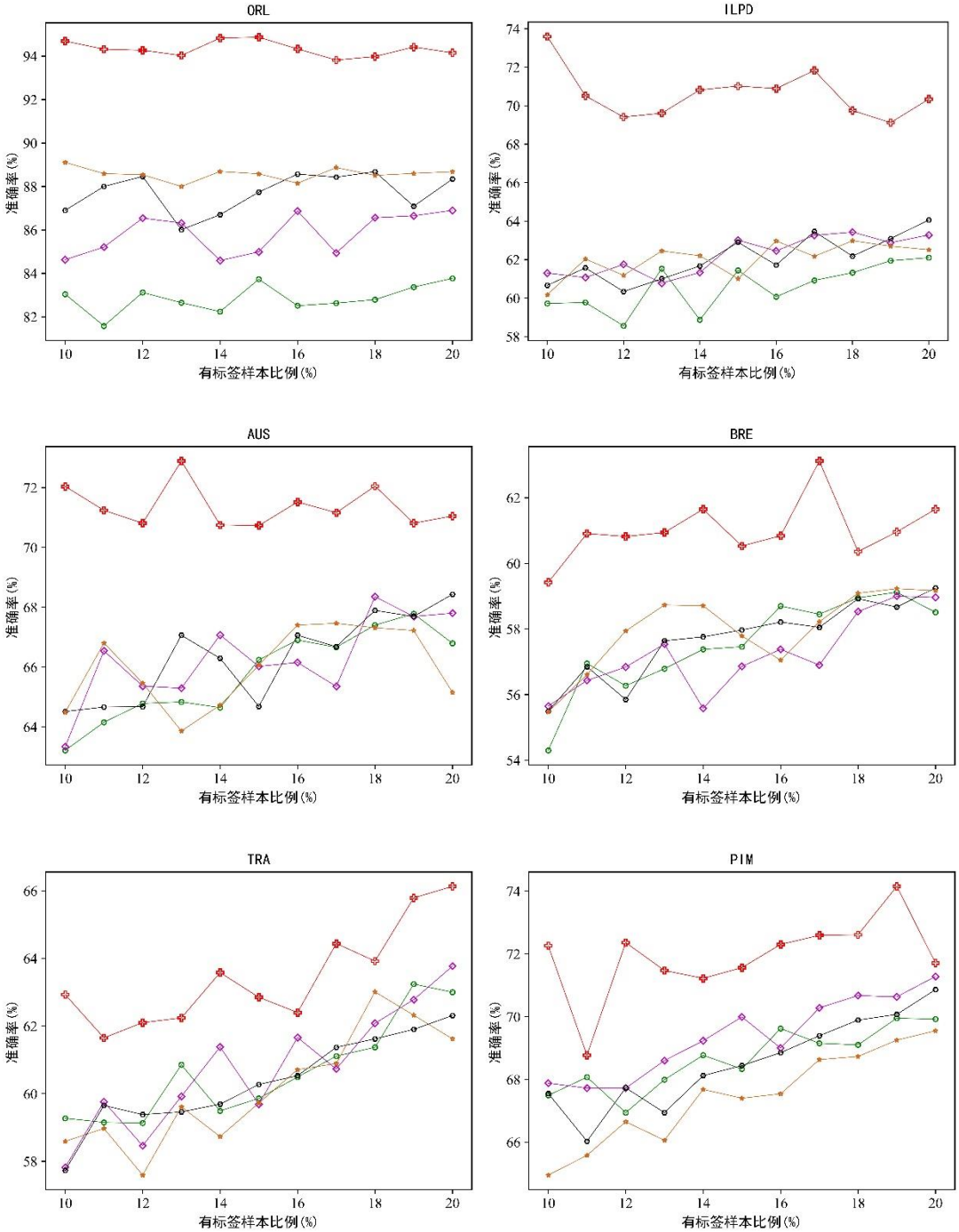
(6)在数据集 MPE 数据集上 STALN 在三个不同基分类器的 Accuracy 值均未获得最优性能。这是因为 MPE 具有 70 个类别，在类别过多的数据集上 STALN 求取局部邻居标签众数时易出现偏差进而导致被划分为噪声样本。因此，STALN 不适用与类别过多的数据集。

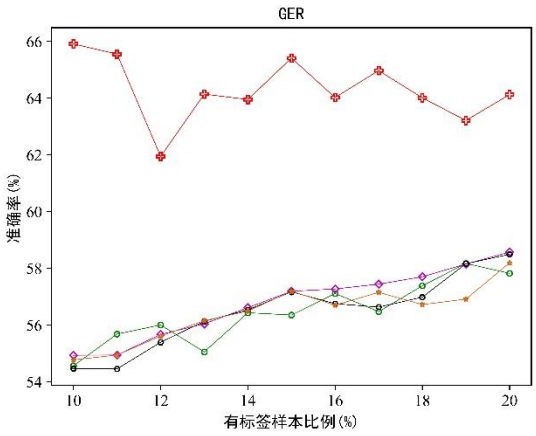
4.7 有标签样本比例对分类性能的影响

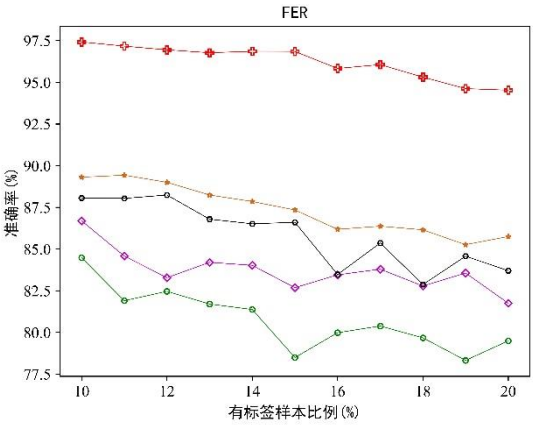
为验证不同比例的有标签样本对分类准确性 的影响，有标签样本比例在[ ]

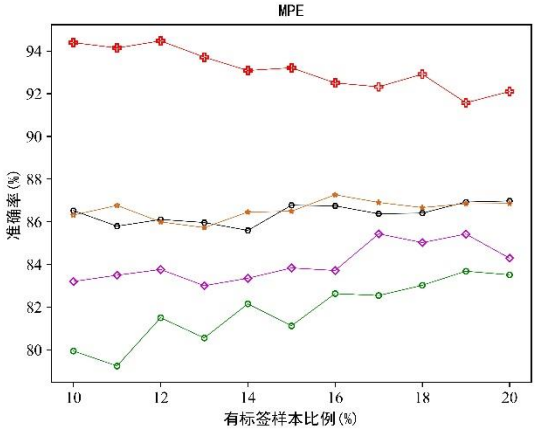
10%,20%内取值，步 长为 1%。STALN 采用 3NN 为基分类器在 16 个数 据集上进行实验，每次实验运行 50 次，并记录平均准确率。实验结果如图 4.3 所示。

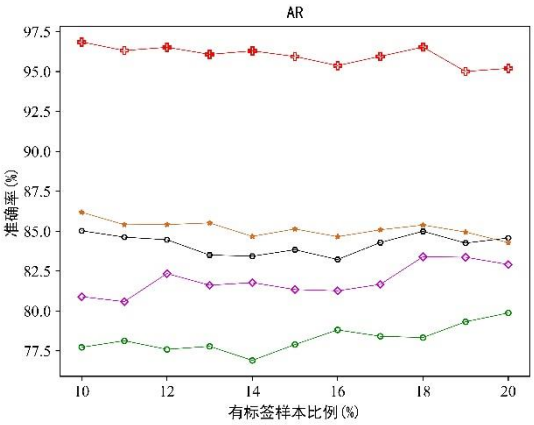
0 次，并记录平均准确率和标准差。实验结果如图 4.3 所示。

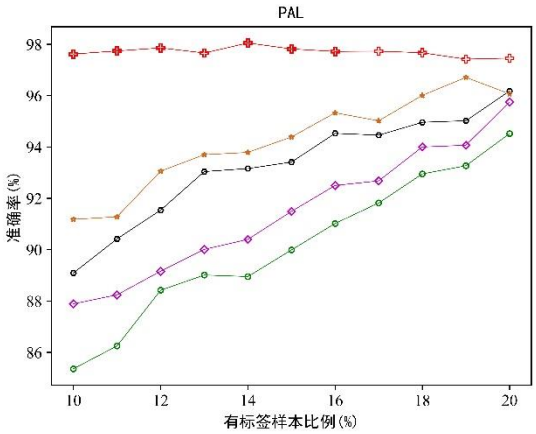


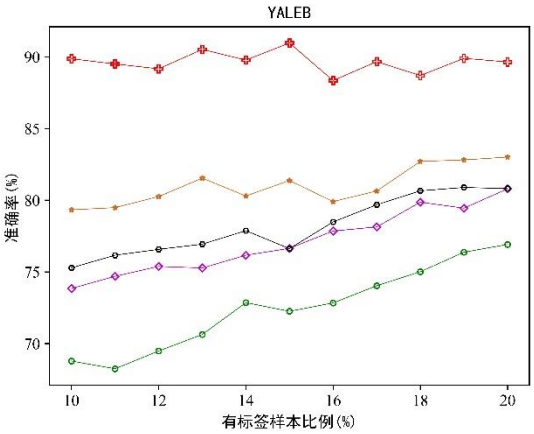

GER
准确率(%)
有标签样本比例(%)


FER
准确率(%)
有标签样本比例(%)


MPE
准确率(%)
有标签样本比例(%)


AR
准确率(%)
有标签样本比例(%)


PAL
准确率(%)
有标签样本比例(%)


YALEB
准确率(%)
有标签样本比例(%)

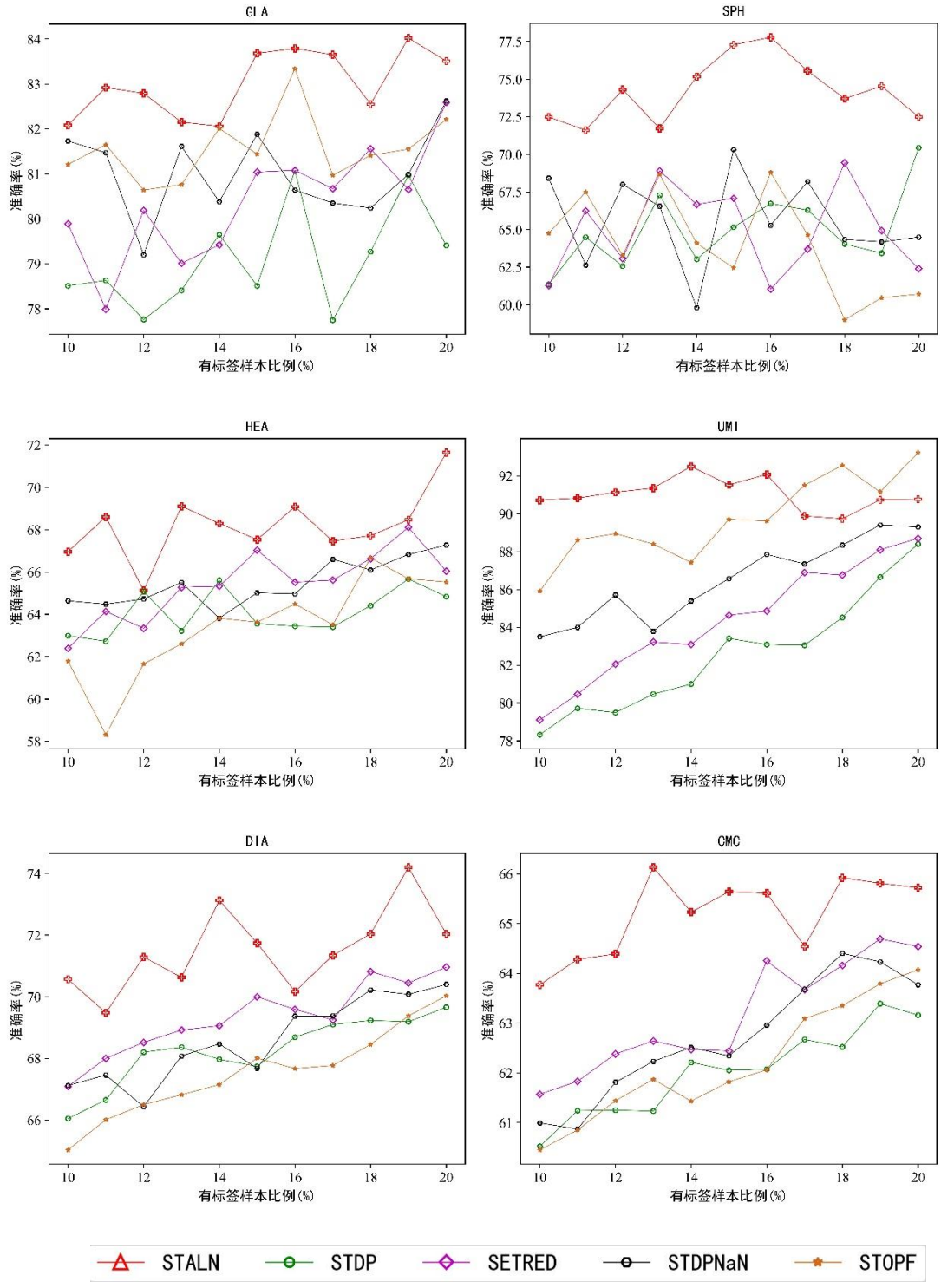


图　4.3　不同比例有标记样本下的算法分类准确率

从图 4.3 可以得到以下结论：

(1)从整体上看，在除 UMI 外的其它数据集上，STALN 分类性能在不同比例有标签样本下显著高于对比算法。

(2)在 ORL，FER，MPE，AR，PAL，YALEB 和 UMI 图像数据集上，STALN 在不同标签比例下 Accuracy 准确率较高且变化比较平稳，这说明了 STALN 基于数据的全局信息自适应地选择无标签样本的局部邻居，在图像数据集上 STALN 相较于对比算法更加鲁棒。

(3)在 UMI 数据集上，有标签比例为 17%~20%时 STALN 的 Accuracy 低于 STOPF，因为 STOPF 利用 OPF 揭示了数据潜在的空间结构，能够选出高质量的高置信度样本，从而提升分类器的性能。但是从整体上看，STALN 在 UMI 数据集上的 Accuracy 高于 STOPF。

(4)在 HEA、GLA 和 SPH 数据集上，STALN 的 Accuracy 在不同标签比例下波动较大，可能是因为这些数据集的样本量比较小，当有标签样本量较少时，STALN 无法有效地挖掘有标签样本的信息。

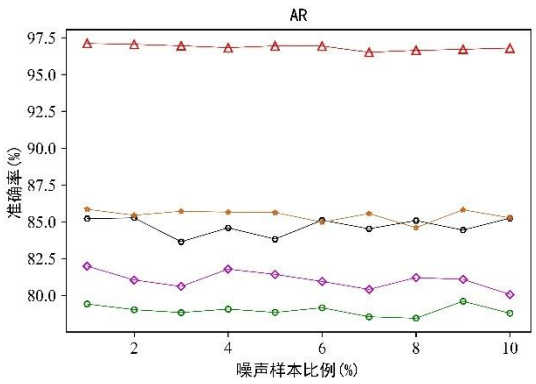
4.8 噪声实验分析

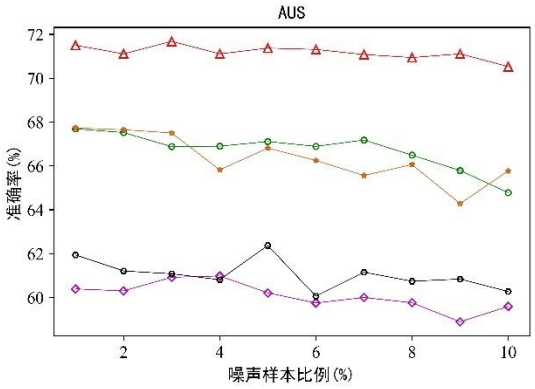
在 self-training 算法的迭代过程中，将误分类的无标签样本添加进训练集，错误将会随后的迭代过程中扩大，将严重降低分类器的性能。STALN 充分考虑了数据的全局信息，挖掘全部有标签样本的信息，因此 STALN 能够过滤噪声样本。为了验证本文所提的算法过滤噪声样本的能力，本文进行了噪声实验。

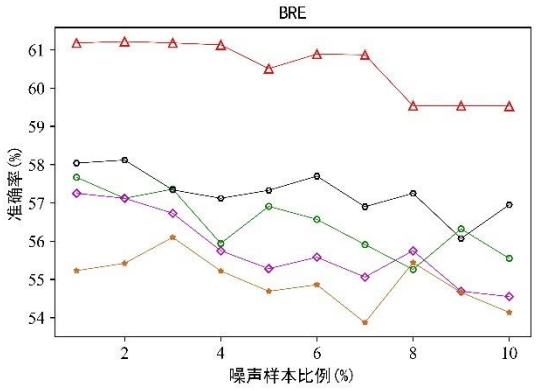
本文随机选择 10%的有标记样本作为训练集，并随机从训练集中抽取 1%到 10%的样本赋予错误标签。本文选择 Accuracy 作为评价指标，在 3NN 基分类器上进行噪声实验，每次实验运行 50 次并记录平均值和标准差。噪声实验的结果见表 4.8 和图 4.4

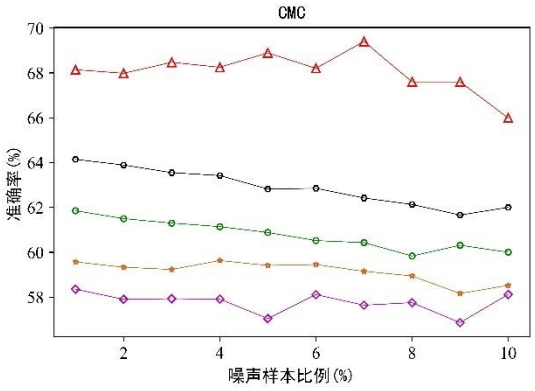
表 4.8 每个算法在噪声数据集上的平均 Accuracy 与标准差（排序）

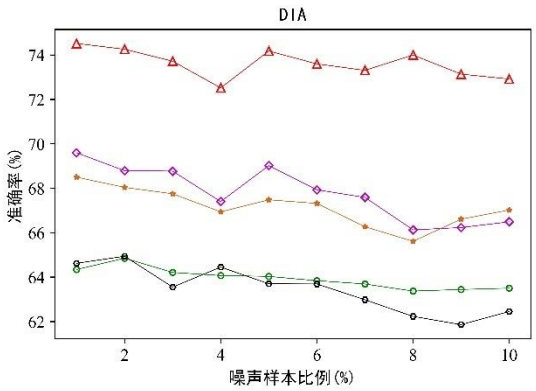
	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	60.78±0.83(3)	57.76±0.63(5)	59.14±0.66(4)	62.89±0.72(2)	**68.05±2.87(1)**
BRE	56.46±1.88(3)	55.78±1.32(4)	54.96±1.52(5)	57.28±1.23(2)	**60.56±2.45(1)**
UMI	81.71±1.51(4)	78.53±1.05(5)	86.59±1.40(2)	84.55±0.91(3)	**91.79±0.92(1)**
ILPD	62.29±1.79(3)	60.62±1.53(4)	59.63±3.35(5)	62.50±1.11(2)	**65.27±2.11(1)**
AUS	66.72±1.36(2)	60.08±2.12(5)	66.34±1.81(3)	61.05±3.03(4)	**71.17±1.51(1)**
YALEB	68.97±0.59(5)	71.83±0.66(4)	79.65±0.88(2)	75.08±1.08(3)	**90.72±2.46(1)**
ORL	81.85±0.94(5)	85.63±1.08(4)	88.51±0.68(2)	87.79±0.96(3)	**94.61±0.93(1)**
FER	83.94±0.39(5)	86.80±0.57(4)	90.48±0.44(2)	88.58±0.70(3)	**96.71±0.92(1)**
PAL	82.42±0.75(5)	85.73±0.63(4)	91.94±0.49(2)	89.02±0.66(3)	**96.87±0.58(1)**
AR	78.97±0.66(5)	81.05±0.89(4)	85.44±0.44(2)	84.69±1.01(3)	**96.85±0.80(1)**
GER	52.75±1.08(5)	55.62±0.92(3)	53.22±0.93(4)	56.60±1.28(2)	**68.19±3.80(1)**
HEA	55.29±3.52(4)	63.95±3.39(2)	59.52±3.68(3)	53.59±4.88(5)	**64.49±2.77(1)**
SPH	58.55±2.48(5)	61.79±4.82(3)	62.12±3.05(2)	61.50±1.99(4)	**75.05±7.72(1)**
PIM	68.95±1.97(2)	63.44±1.74(4)	56.77±1.56(5)	65.19±2.09(3)	**78.86±5.46(1)**
GLA	77.72±2.09(4)	75.55±2.63(5)	77.74±3.09(3)	**83.13±2.07(1)**	82.94±2.76(2)
DIA	63.93±1.37(4)	67.8±1.71(2)	67.15±1.39(3)	63.45±2.14(5)	**73.62±2.62(1)**
BTSC	58.71±0.78(3)	51.40±1.56(5)	58.92±1.21(2)	57.28±1.70(4)	**61.65±5.04(1)**
MPE	78.78±0.93(5)	85.54±0.67(3)	87.20±0.75(2)	84.34±0.99(4)	**94.69±0.93(1)**
Wilcoxon	+	+	+	+	N/A
平均值	68.82	69.38	71.41	71.03	79.56

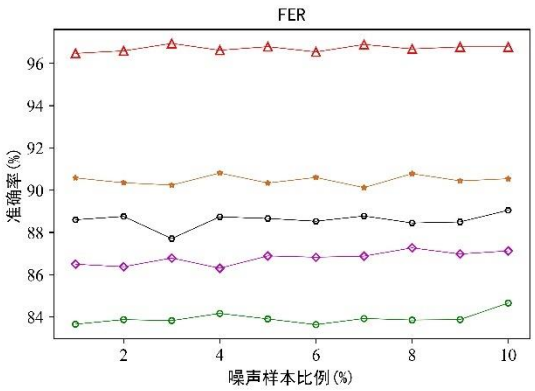

AR
准确率(%)
噪声样本比例(%)

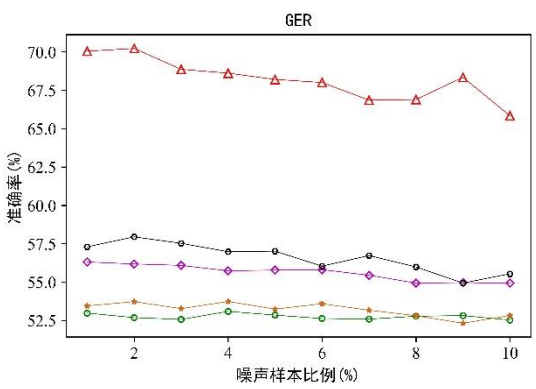

AUS
准确率(%)
噪声样本比例(%)

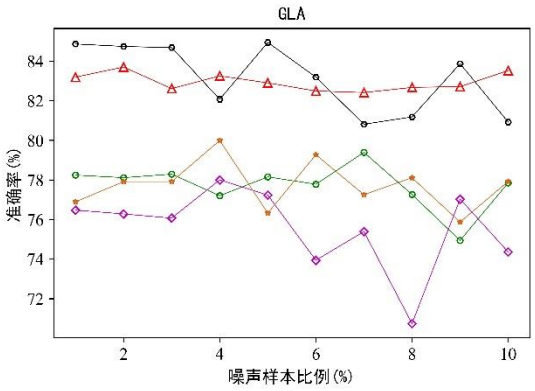

BRE
准确率(%)
噪声样本比例(%)

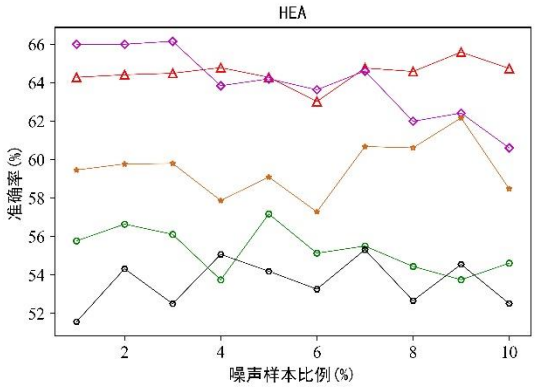

CMC
准确率(%)
噪声样本比例(%)

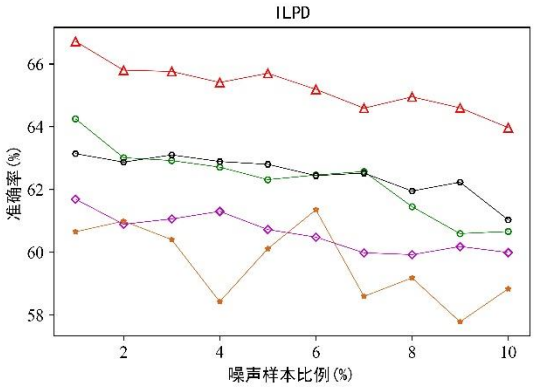

DIA
准确率(%)
噪声样本比例(%)

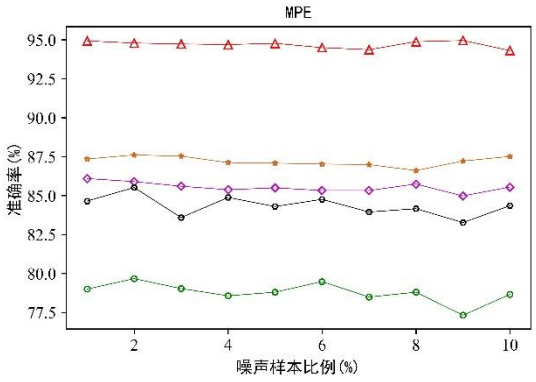

FER
准确率(%)
噪声样本比例(%)

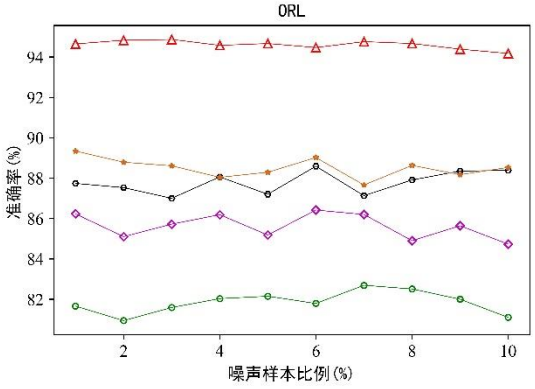

GER
准确率(%)
噪声样本比例(%)

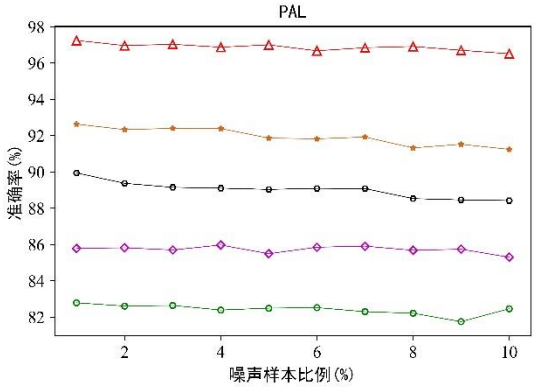

GLA
准确率(%)
噪声样本比例(%)

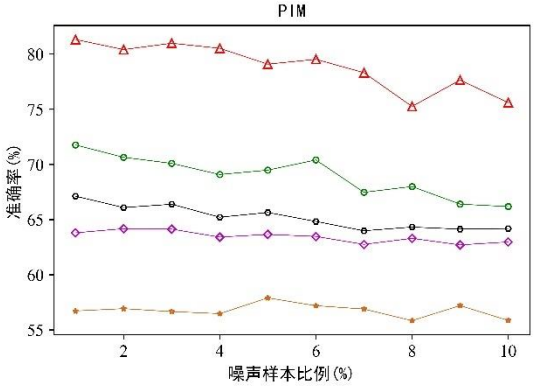

HEA
准确率(%)
噪声样本比例(%)

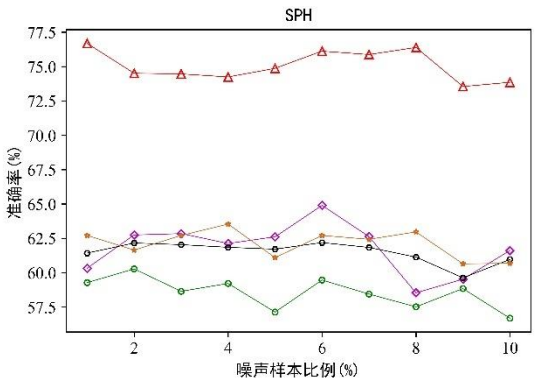

ILPD
准确率(%)
噪声样本比例(%)

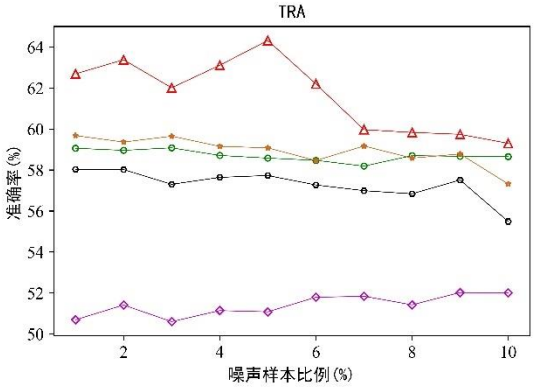

MPE
准确率(%)
噪声样本比例(%)


ORL
准确率(%)
噪声样本比例(%)


PAL
准确率(%)
噪声样本比例(%)


PIM
准确率(%)
噪声样本比例(%)


SPH
准确率(%)
噪声样本比例(%)


TRA
准确率(%)
噪声样本比例(%)

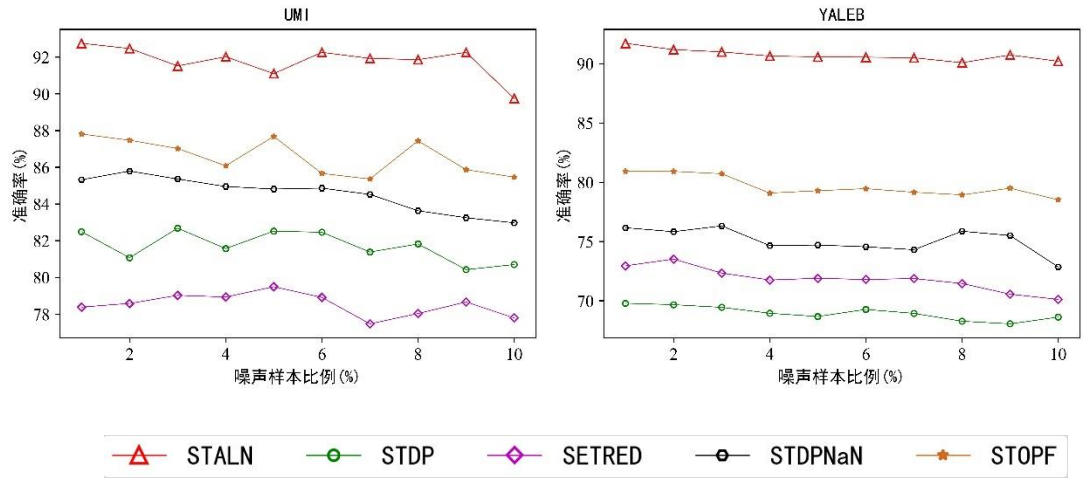


图 4.4　含不同比例噪声样本下的算法分类准确率

从表 4.8 和图 4.4 中，可以得到以下结论：

(1)STALN 的 Accuracy 在 18 个数据集中有 16 个数据集显著优于对比算法，这说明了本文所提出的算法具有极佳的噪声过滤能力。

(2)在 GLA 和 HEA 数据集上，STALN 的 Accuracy 略低于对比算法，因为有标签样本量较少，STALN 利用全局有标签样本信息，选取的无标签样本局部邻居不合适。

(3)总的来说，随着噪声比例的增加，所有算法的分类性能下降。在 18 个数据集上，STALN 的平均准确率为 79.56%，相较与 STDP、SETRED、STOPF 和 STDPNaN 准确率分别高于 10.74%、10.18%、8.15%和 8.53%。实验结果充分说明了 STALN 能高效地过滤噪声样本。

4.9 运行时间分析

令 n 表示数据集样本的个数，t 表示 self-training 算法的迭代次数，n_1 表示有标记样本的数量，n_2 是无标记样本的数量。则 STALN 的时间复杂度 $O(n^2+tn_1n_2)$，其中 $n_1+n_2=n$。STDP 计算样本点局部密度的时间复杂度为 $O(n^2)$ 与计算峰值的时间复杂度为 $O(n^2)$，因此 STDP 总的时间复杂度 $O(n^2)$。STOPF 构建 OPF 的时间复杂度是 $O(n^2)$，难以在整个数据集上构建 OPF，尤其是在大数据

集上。SETRED 算法的时间复杂度主要在于构造构建 RNG，其时间复杂度为 $O(n^3)$，所以 SETRED 总的复杂度为 $O(n^3)$。STDPNaN 计算每个样本的自然邻居和构建 DPC 结构的时间复杂性分别为 $O(n\log n)$ 和 $O(n^2)$，因此 STDPNaN 总的时间复杂度为 $O(n^2)$。本文以 3NN 作为基分类器，选择 10%的有标签样本进行运行时间分析实验。记录 5 个算法在 18 个数据集上行 50 次的时间均值（秒），实验结果见表 4.9。

表 4.9 每个算法在不同数据集上的运行时间(排名)

	STDP	SETRED	STOPF	STDPNaN	STALN
CMC	0.36(2)	2.43(3)	16.50(5)	11.17(4)	0.29(1)
BRE	0.15(2)	0.80(3)	4.93(5)	1.80(4)	0.11(1)
UMI	4.16(1)	22.08(2)	333.94(5)	36.28(3)	40.02(4)
ILPD	0.06(1)	0.57(3)	2.81(5)	0.65(4)	0.17(2)
AUS	0.13(1)	0.90(3)	7.34(5)	1.04(4)	0.13(2)
YALEB	7.28(1)	43.47(3)	888.18(5)	182.93(4)	21.38(2)
ORL	0.22(1)	3.08(4)	11.27(5)	2.32(3)	0.59(2)
FER	2.40(1)	19.04(3)	179.44(5)	24.71(4)	6.12(2)
PAL	1.42(1)	13.25(3)	114.67(5)	25.79(4)	4.80(2)
AR	3.31(1)	22.46(3)	320.10(5)	38.62(4)	10.49(2)
GER	0.13(1)	0.82(3)	12.25(5)	1.59(4)	0.14(2)
HEA	0.05(1)	0.38(4)	1.16(5)	0.19(3)	0.07(2)
SPH	0.05(1)	0.95(4)	1.37(5)	0.49(3)	0.21(2)
PIM	0.07(1)	0.62(3)	7.25(5)	0.89(4)	0.21(2)
GLA	0.03(1)	0.32(4)	0.59(5)	0.13(3)	0.05(2)
DIA	0.09(1)	0.64(3)	7.14(5)	0.78(4)	0.09(2)
BTSC	0.08(1)	0.69(3)	6.04(5)	1.00(4)	0.24(2)
MPE	13.35(1)	112.11(3)	1769.03(5)	235.38(4)	55.90(2)
平均值排名	1.11	3.17	5	3.72	2

从表 4.9 中本文可以看出:

(1)在 UMI 数据集上 STALN 的运行时明显高于 STDP、SETRED 和 STDPNaN，这是因为 UMI 数据集的维度很高，在计算欧式距离时会发生“维度灾难”，STALN 依赖欧式距离选取的局部邻居受到影响。

(2)STALN 在 self-training 算法迭代的过程中需要计算无标签样本与全部有标签样本的平均距离，这需要消耗一定的计算时间。总的来说，STALN 在 18 个数据集上的平均时间排名第二，仅仅低于 STDP ，实验的结果与理论分析一致。

4.10 本章小结

针对传统的 self-training 算法仅仅考虑无标签样本与其邻近有标签样本的关系而忽略了全部有标签样本的信息这一问题，提出了一种自适应局部邻居过滤的 self-training 算法(A Self-training Algorithm for Adaptive Local Neighbor Filtering, STALN)。STALN 挖掘全部有标签样本的信息，自适应地选择无标签样本的局部有标签邻居。一方面自适应地选择邻居避免不同参数对算法性能的影响，另一方面通过结合数据的全局信息与数据的局部信息，能准确地预测样本的标签。为了验证所提算法的性能，本文进行了大量的实验，实验结果表明了所提算法显著优于对比算法。然而，由于高维数据下传统的欧式距离计算很困难，导致本文算法在高维数据集上的分类性能有所下降。

5. 总结与展望

5.1 总结

本文主要是对半监督学习的 self-training 分类进行了研究。首先介绍了 self-training 分类的研究背景与意义, self-training 分类任务面临的问题是有标记样本量少,解决问题的关键在于从大量的无标记样本中找到高置信度样本。在半监督学习下，可以利用少量的有标签样本和更多的无标签样本，这一方面使得标记样本的成本大幅下降，另方面也更符合现实世界中数据状况。之后介绍了半监督学习和 self-training 算法思想、几个典型的 self-training 算法，用到的基分类器等相关知识。

第一个研究工作中利用基于 self-training 框架， 提出了基于块的不相似性度量和原型树，设计了鲁棒的半监督数据编辑算法。基于块的不相似度量通过平行分隔建立树重新定义样本密度和峰值。在分布复杂数据上，基于块的不相似性计算样本之间的距离比传统的几何距离更科学。另一方面，目前的 self-training 方法在高置性度样本的选取过程未能充分利用数据集的潜在空间结构。因此，本文用基于块的不相似性度量和原型树度量样本间关系选取高置信度样本，并进一步对高置信度样本进行噪声过滤提升算法性能。本文在 14 个公开数据集和 4 个已有 self-training 方法开展实验，验证了方法的有效性。

在第二个研究工作中，提出了基于区局信息的自适应局部邻居过滤 self-training 的算法。STALN 挖掘全部有标签样本的信息，自适应地选择无标签样本的局部有标签邻居。一方面自适应地选择邻居避免不同参数对算法性能的影响，另一方面通过结合数据的全局信息与数据的局部信息，能准确地预测样本的标签。为了验证所提算法的性能，本文进行了大量的实验，实验结果表明了所提算法显著优于对比算法。然而，由于高维数据下传统的欧式距离计算很困难，导致本文算法在高维数据集上的分类性能有所下降，在未来将研究引入降维算法以提高在高维数据上分类器的性能。

综上所述， 本文提出两种高置信度样本选择方法，并将其应用到 self-

training 算法中，提升高置信度样本的选取质量，从而提升了分类器的性能。

5.2 展望

基于鲁棒的高置信度样本的选择方法是 Self-training 近年研究热点之一，它可以有效地利用过滤和编辑高噪声样本，提高模型的分类性能。然而，这个领域仍然存在着很多挑战和问题，需要进一步的研究和探索。而且本文提出的 self-training 算尽管在性能上略好于相关的对比算法，但仍然存在一些尚未解决的问题对于未来的工作，可以考虑从以下两个方面改进.

（1）本文中第一个研究主要考虑了是数据集的空间结构和噪声过滤， 虽然该方法有效的提升分类的性能，但该方法不适用与小型非均衡数据集，具有较高的时间复杂度，以及超参数难以设置。未来可以探索适用于更多类型数据集以及无参数的快速方法。

（2）本文中第二个研究虽然无参且利用数据集的全局信息，但由于该算法使用的传统的欧式距离，导致本文算法在高维数据集上的分类性能有所下降，并且不适用于类别过多的数据急，在未来将研究引入降维方法以提高在高维数据上分类器的性能。

参考文献

[1] Larose Daniel T .Data Mining: Methods and Models[J].Technometrics, 2007, 49(4).

[2] 邵峰晶.数据挖掘原理与算法[M].水利水电出版社,2003.

[3] PangNing Tan,Michael Steinbach,Vipin Kumar,等.数据挖掘导论[M].人民邮电出版社, 2006.

[4] 胡镜清,刘保延,王永炎.中医临床个体化诊疗信息特征与数据挖掘技术应用分析[J].世界科学技术:中医药现代化, 2004, 6(1):14-16.

[5] 徐卓揆,刘德钦,林宗坚,等.WebGIS 与基于网络的数据挖掘整合应用研究[J].测绘科学, 2004, 29(3):3.

[6] 郝先臣,张德干,尹国成,等.用于电子商务中的数据挖掘技术研究[J].小型微型计算机系统, 2001, 22(7):4.

[7] 李莉平.数据挖掘技术在现代市场营销中的应用[J].云南财经大学学报, 2006, 22(5):40-45.

[8] Tom M Mitchell. Machine learning and data mining[J]. Communications of the ACM, 1999, 42(11):30-36.

[9] 刘红岩,陈剑,陈国青.数据挖掘中的数据分类算法综述[J].清华大学学报：自然科学版, 2002, 42(6).

[10] 江铃燚.有监督深度学习的优化方法研究综述[J].中国图象图形学报, 2023,963-983.

[11] Barlow, Horace B.Unsupervised learning[J]. Neural computation,1989,295-311.

[12] Hastie, Trevor, et al.Unsupervised learning[J].The elements of statistical learning: Data mining, inference, and prediction, 2009, 485-585.

[13] 刘建伟,刘媛,罗雄麟.半监督学习方法[J].计算机学报,2015,1592-1617.

[14] Xiaojin Zhu,Andrew B.Goldberg .Introduction to Semi-Supervised Learning[J].Synthesis Lectures on Artificial Intelligence and Machine Learning, 2009, 3(1):130.

[15] Sebastiano Bruno Serpico,Lorenzo Bruzzone,Fabio Roli.An experimental comparison of neural and statistical non-parametric algorithm for supervised classification of remote-sensing images.[J]. Pattern Recognition Letters, 1996, 17(13):1331-1341.

[16] Sim Sejin, Bae Jinsoo, Kim SeoungBum. Robust Semi-Supervised Regression for Vehicle Interior Noise Prediction[J]. IEEE Access,2024,12: 60-72.

[17] Christophe Rosenberger,Kacem Chehdi.Unsupervised clustering method with optimal estimation of the number of clusters: application to image segmentation[J]. Pattern Recognition, 2000,1:656-659.

[18] Manoranjan Dash,Huan Liu.Dimensionality reduction of unsupervised data[J].IEEE, 1997,532-539.

[19] Patel A .An Efficient Multilevel Association Rule Mining Based On Unsupervised Learning[J].2024，3.

[20] 管仁初.半监督聚类算法的研究与应用[D].吉林大学,2010.

[21] 王瑶.结合主动学习的半监督分类算法优化研究[D].大连理工大学,2024.

[22] 陈诗国,张道强.半监督降维方法的实验比较[J].软件学报, 2011, 22(1):16.

[23] Sejin Sim， Jinsoo Bae， Seoung Bum Kim.Robust Semi-Supervised Regression for Vehicle Interior Noise Prediction[J].IEEE Access,2024,12:20-72.

[24] Luyi Han，Yunzhi Huang，Haoran Dou.Semi-supervised segmentation of lesion from breast ultrasound images with attentional generative adversarial network[J].Compution Methods Programs Biomed,2023,189:105275.

[25] Gharebaghi, Farhad, Amiri. LP-MLTSVM: Laplacian Multi-Label Twin Support Vector Machine for Semi-Supervised Classification[J].IEEE Access,2022,10: 13738-13752.

[26] 张长帅.基于图的半监督学习及其应用研究[D].南京航空航天大学，2024.

[27] 李亚娥.基于图的半监督分类算法研究[D].陕西师范大学,2012.

[28] Xiangkui Lu,Jun Wu,Junheng Huang.Co-Training-Teaching: A Robust Semi-Supervised Framework for Review-Aware Rating Regression[J].ACM Transactions on Knowledge Discovery from Data,2024,18(14):1-16.

[29] Jian Wu. The self-training semi-supervised support vector machine based on wavelet entropy for the evaluation of the elderly gait[J].Chinese Journal of Biomedical Engineering, 2013, 32(5):588-594.

[30] A. Agrawala. Learning with a probabilistic teacher[J]. Information Theory, 1970, 16(4):373-379.

[31] Christopher Merz and Daniel C. St. Clair and William E. Bond.SeMi-supervised adaptive resonance theory(SMART2)[C]. International Joint Conference on Neural Networks. IEEE, 1992,3: 851-856.

[32] David Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods[J]. Proceedings of Annual Meeting of the Association for Computational Linguistics, 1995, 49:189-196.

[33] Alexander Gammerman, Vladimir Vovk, Vladimir Naumovich Vapnik. Learning by Transduction[C]. UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, 1998,24-26.

[34] Blum Avrim, Mitchell Tom. Combining Labeled and Unlabeled Data with Co-Training[C]. Proceedings of the 11th Annual Conference on Computational Learning Theory. 1998.

[35] Rajat Raina, Alexis Battle, Honglak Lee. Self-taught learning: transfer learning from unlabeled data[J]. International Conference on Machine Learning,2007,759-766.

[36] Ming Li, ZhiHua Zhou.SETRED: Self-training with Editing[J].Springer Verlag, 2005: 611-621.

[37] Dennis Wilson. Asymptotic properties of nearest neighbor rules using edited data [J]. IEEE Transactions on Systems, Man, and Cybernetics, 1972, (3):408-421.

[38] Kenneth Supowit. The relative neighborhood graph, with an applicationto minimum spanning trees[J]. Journ al of the ACM, 1983, 30(3):428-448.

[39] Fabrice Muhlenbach, Stéphane Lallich,Djamel A. Zighed. Identifying and handling mislabelled instances[J]. Journal of Intelligent Information Systems, 2004, 22(1):89-109.

[40] Yu Wang, Xiaoyan Xu, Haifeng Zhao, Zhongsheng Hua. Semi-supervised learning based on nearest neighbor rule and cut edges [J]. Knowledge-Based Systems, 2010, 23(6): 547-54.

[41] Zhihua Wei, Hanli Wang, Rui Zhao. Semi-supervised multi-label image classification based on nearest neighbor editing[J]. Neurocomputing, 2013, 119: 462-8.

[42] Dennis Wilson. Asymptotic properties of nearest neighbor rules using edited data [J]. IEEE Transactions on Systems, Man, and Cybernetics, 1972, (3):408-421.

[43] Jafar Tanha, Maarten van Someren, Hamideh Afsarmanesh. Semi-supervised self-training for decision tree classifiers[J]. International Journal of Machine Learning and Cybernetics, 2017,7: 355–370.

[44] XiZhao Wang, Ling-Cai Dong, JianHui Yan. Maximum ambiguity-based sample selection in fuzzy decision tree induction[J]. IEEE Transactions on Knowledge and Data Engineering, 24(8):1491–1505.

[45] Dasgupta S . Mahalanobis Distance[M]. 2005.

[46] Di Wu, Mingsheng Shang, Xin Luo.Self-training semi-supervised classification based on density peaks of data[J].Neurocomputing, 2018, 275:180-191.

[47] Alex Rodriguez, Alessandro Laio. Clustering by fast search and find of density peaks[J].American Association for the Advancement of Science, 2014, 1492-1496.

[48] Haitao Gan, Nong Sang, Rui Huang, Xiaojun Tong, Zhiping Dan. Using clustering analysis to improve semi-supervised classification [J]. Neurocomputing, 2013, 101: 290-298.

[49] Marti Hearst. Support Vector Machines [J]. IEEE Intelligent Systems and their Applications, 1998,4(13):18-28.

[50] Ruixiang Zhang, Tong Che, Zoubin Ghahramani. Metagan: An adversarial approach to few-shot learning [C]. Neural Information Processing Systems.Curran Associates Inc. 2018.

[51] Yin Liu.Self-training algorithm combining density peak and cut edge weight[J]. International Conferences on Distributed Multimedia Systems, 2020:11-16.

[52] Junnan Li, Qingsheng Zhu, Quanwang Wu. A self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor[J]. Knowledge-Based Systems, 2019, 184: 104895.

[53] Suwen Zhao and Junnan Li. A semi-supervised self-training method based on density peaks and natural neighbors[J]. Journal of Ambient Intelligence and Humanized Computing, 2021, 12: 2939-2953.

[54] Qingsheng Zhu, Ji Feng, Jinlong Huang. Natural neighbor: A self-adaptive neighborhood method without parameter K[J]. Pattern Recognition Letters, 2016, 80: 30-36.

[55] Junnan Li, Qingsheng Zhu. Semi-supervised self-training method based on an optimum-path forest [J]. IEEE Access, 2019, 7:36388-99.

[56] Luis Claudio Sugi Afonso, Douglas Rodrigues, João Paulo Papa. Nature-inspired optimum-path forest [J]. Evolutionary Intelligence, 2023, 16: 317-328.

[57] 刘学文, 王继奎, 杨正国. 密度峰值隶属度优化的半监督 Self-Training 算法[J]. 计算机科学与探索, 2022, 16(9): 2078.

[58] Shuyin Xia, Daowan Peng, Deyu Meng. A Fast Adaptive k-means with No Bounds[J].IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 1: 99.

[59] Bing Li, Jikui Wang, Zhengguo Yang ,Jihai Yi, Feiping Nie.Fast semi-supervised self-training algorithm based on data editing[J]. Information Sciences,2023,626: 293-314.

[60] Ruixiang Zhang, Author PictureTong Che. MetaGAN: an adversarial approach to few-shot learning[C]. Neural Information Processing Systems.Curran Associates Inc. 2018.

[61] Haixia Bi, Jian Sun, Zongben Xu. A graph-based semi-supervised polsar image classification method using deep convolutional neural networks[J]. IEEE Transactions on Geoscience and Remote Sensing, 2020, 48(1): 66.

[62] Sofia Visa, Brian Ramsay, Anca Ralescu. Confusion matrix-based feature selection. [J]. Proceedings of The 22nd Midwest Artificial Intelligence and Cognitive Science Conference, 2011, 710(1):120-127,.

[63] QingJun Song , HaiYan Jiang, Jing Liu.Feature selection based on FDA and F1-score for multi-class classification [J]. Expert Systems with Applications, 2017, 81:22-27.

[64] Seyyed Meisam Taheri, G. Hesamian, A generalization of the wilcoxon signed-rank test and its applica tions [J], Statistical Papers, 2013, 54: 457–470.

[65] Martínez, A., Benavente, R., The are face database, Computer vision center, technical report, 1998.

[66] WenYi Zhao, Rama Chellappa, Arvind Krishnaswamy. Discriminant analysis of principal components for face recognition [J]. IEEE Computer Society, 1998, 73–85.

[67] Mohammad Shabbir Hasan, Xiaowei Wu, Layne Watson, Liqing Zhang. UPS-indel: a universal positioning system for indels [J]. Scientic reports., 2017, 7(1):1-13

[68] Athinodoros Georghiades, Peter Belhumeur, David Kriegman. From few to many:Illumination cone models for face recognition under variable lighting and pose. IEEE transactions on pattern analysis and machine intelligence , 2001, 23(555):643–660.

[69] 黄灵林. 基于 MPEG-7 的视频数据库存储检索技术研究[D]. 武汉理工大学.

[70] KaiMing Ting, GuangTong Zhou, FeiTony Liu. Mass estimation [J]. Machine Learning, 2013, 90(1):127-160.

[71] ShiFei Ding, Xiao Xu, YanRu Wang. Optimized density peaks clustering algorithm based on dissimilarity measure [J]. Journal of Software. 2020, 31(11):3321-3333.

[72] Bo Chen, KaiMing Ting, Takashi Washio. Half space mass: a maximally robust and e cient data depth method [J]. Mach. Learn. ,2015, 100(2-3):677-699.

[73] FeiTony Liu, KaiMing Ting, ZhiHua Zhou . Isolation forest [J], ICDM, IEEE ComputerSociety, 2008:413422.

致 谢

总以为来日方长，却不知时光匆匆，硕士三年求学时光转瞬即逝。始于 2021 年初秋，我离开家乡，第一次来到素有“黄河明珠”之称的兰州市，在这里我感受到西北人民的热情朴素，了解到塞北风光的雄奇壮阔，品尝到西北特有的美食佳肴。兰州财经大学不断见证我的成长，目光所及，校园里的点点滴滴汇聚心头，失去与收获，遗憾与欣喜，但更多的还是感谢。

盛行千里，不忘师恩。首先我要感谢我的导师李兵老师和王继奎老师，他们在忙碌的教学工作中仍不忘挤出时间关心我的学习和生活状况。他们是专业知识渊博的学者，给我提供了最好的学习环境与实验设备，在论文撰写过程中给予了我悉心的指导和宝贵的意见；他们也是我人生路上的良师，在我在生活迷茫时给我指点迷津，阐述道理，让我能够面对一次次的挫折，勇往直前。感谢 511 实验室易纪海老师和杨正国老师对我学业的帮助。在此，再次向李兵老师和王继奎老师致以崇高的敬意和由衷的感谢！

父母之爱，为之深远。其次我要感谢父母的养育之恩，你们一直坚定不移尊重、支持我的选择，并给予我无微不至的照顾。你们日以继夜，不辞辛苦的工作换来了我现在的一切，却也换来了日渐衰老的身体。希望我能尽快成为您依靠的肩膀，为您分担解忧。

山水一程，有幸相遇。我还要感谢 511 实验室的所有伙伴们。当我学习上有不懂地方时你们总会教我正确的思路与方法，是你们让原本枯燥乏味的科研生活变得多姿多彩。感谢我可爱的室友们，从陌生到熟悉，回到寝室后总会有无限的欢乐，从不觉得无趣。

最后，我要感谢百忙之中抽出时间来评审我的论文的各位专家教授以及答辩委员会的老师们，感谢你们对本文的指导与宝贵意见！

攻读硕士学位期间发表的论文及科研情况

发表论文：

[1] Wu Y, Duan H, Wang J. Identification model of poverty-prone population and analysis of poverty-causing factors[C]//International Conference on Intelligent Systems, Communications, and Computer Networks (ISCCN 2022). SPIE.2022，12332: 465-469

[2] Duan H, Chen C, Wang J. SMpeaks: a semi-supervised clustering algorithm based on density peaks[C]//6th International Workshop on Advanced Algorithms and Control Engineering (IWAACE 2022). SPIE, 2022, 12350:607-612

[3] Jihai Yi, Huiyu Duan, Jikui Wang, Zhengguo Yang, Feiping Nie. Structure preserved fast dimensionality reduction. Applied Soft Computing

参与课题：

[1]甘肃省自然科学基金项目“高维数据的鲁棒半监督多标签学习算法研究”(项目编号 22JR5RA554)，参与

[2]甘肃省高等学校创新基金项目“类簇结构保持的快速无监督线性降维算法 研究”(项目编号 2022A-092)，参与

[3]公共大数据国家重点实验室开放课题项目“图优化降维和聚类融合学习模 型与 37 算法研究”(项目编号 GZU-PBD2021-101) ，参与