分类号 _____ 密级 _____
U D C _____ 编号 10741

兰州财经大学
LANZHOU UNIVERSITY OF FINANCE AND ECONOMICS

硕 士 学 位 论 文

论文题目 数据编辑的 Self-training 分类算法研究

研 究 生 姓 名：李 冰

指导教师姓名、职称：聂飞平教授

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：2023 年 5 月 20 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：李冰 签字日期：2023 年 5 月 20 日

导师签名：聂峰 签字日期：2023 年 5 月 20 日

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：李冰 签字日期：2023 年 5 月 20 日

导师签名：聂峰 签字日期：2023 年 5 月 20 日

Research on Self-training classification algorithm with data editing

Candidate : Bing Li

Supervisor: Feiping Nie

摘 要

随着社会经济的发展，数据集的规模越来越大，其中只有少量数据有标签，而数据的标注费时且代价高昂。半监督分类算法可以使用少量有标签样本和大量无标签样本进行学习。自训练作为一种经典的半监督学习框架成为了研究的热点，但自训练算法的性能主要依赖于高置信度样本点的选取，在迭代过程中一旦出现噪声样本将会极大程度地影响算法的分类性能，为了处理数据中的噪声或误标记样本，研究人员提出了许多基于数据编辑的半监督分类算法，然而大多数编辑算法大多使用欧式距离计算样本间距离，且时间复杂度均不低于$O\left(n^2\right)$，不适用于大规模高维数据集。

综上所述，现有的半监督自训练算法存在两个问题：一是现有的自训练算法在选择高置信度样本点时缺乏对噪声样本的处理且时间复杂度高，二是欧式距离在高维数据集上容易出现维度诅咒，针对这两个问题，本文提出了两个算法。

一、提出了快速球簇划分编辑的半监督 Self-training 算法（EBSA）。EBSA 将数据集划分为稳定区域和争议区域，在此基础上提出了球簇划分编辑算法，用来识别稳定区域内的误标记样本点，并对其进行编辑，提升了高置信度样本点选取质量。EBSA 在每次迭代中仅需计算样本点与球簇中心的距离，计算量小，速度快。实验结果表明，与对比算法相比，EBSA 算法不仅运行速度快且提升了算法性能。

二、提出了块估计近邻编辑 Self-training 算法 MDSF。MDSF 使用不相似性度量方法计算样本之间的距离，并定义了块估计近邻关系，进而构建块估计近邻图，在此基础上提出了块估计近邻编辑算法对数据进行编辑，提升了高置信度样本的选取质量。因为使用不相似性度量方法，所以在高维数据集上算法性能较好。大量实验结果表明，与同类算法相比， MDSF 在高维数据集上性能明显优于对比算法。

关键词：半监督 Self-training 数据编辑 分类算法

Abstract

With the development of social economy, the scale of datasets is becoming larger and larger, with only a small amount of data labeled, and data annotation is time-consuming and expensive. Semi-supervised classification algorithms can use a small number of labeled samples and a large number of unlabeled samples for learning. Self-training, as a classic semi-supervised learning framework, has become a research hotspot, but the performance of the self-training algorithm mainly depends on the selection of high confidence sample points. Once noise samples appear in the iterative process, the classification performance of the algorithm will be greatly affected. In order to deal with noise or mislabeled samples in data, researchers have proposed many semi-supervised classification algorithms based on data editing. However, self-training algorithms often use Euclidean distance to calculate the distance between samples, and the time complexity of most editing algorithms is no less than $O\left(n^2\right)$ and are unsuitable for large-scale high-dimensional datasets.

In summary, existing semi-supervised self-training algorithms have two problems: firstly, they lack the processing of noisy samples and have high time complexity when selecting high-confidence sample points. Secondly, Euclidean distance is prone to dimensional curse on high-dimensional datasets. In response to these two problems, this paper

proposes two algorithms.

（1） A semi-supervised self-training algorithm (EBSA) for fast ball cluster partitioning and editing is proposed. EBSA divides the dataset into stable regions and controversial regions. Based on this, a ball cluster partitioning and editing algorithm is proposed to identify and edit mislabeled sample points in stable regions, improving the quality of sample selection with high confidence. In each iteration, EBSA only needs to calculate the distance between the sample point and the center of the ball cluster, which requires less computation and is fast. Experimental results show that compared with the comparison algorithm, the EBSA algorithm not only runs faster but also improves the performance of the algorithm.

（2） A block estimation nearest neighbor editing self-training algorithm (MDSF) is proposed. MDSF uses a dissimilarity metric method to calculate the distance between samples, and defines a block estimation neighborhood relationship. Then, it constructs a block estimation neighborhood graph. Based on this, it proposes a block estimation neighborhood editing algorithm to edit the data, improving the quality of selecting high-confidence samples. The algorithm performs better on high-dimensional datasets because of the use of similarity measures. A large number of experimental results show that compared with similar algorithms, MDSF performs significantly better on high-dimensional

datasets than the comparison algorithm.

Keywords： Semi-supervision; Self-training; Data editing; Classification algorithm

目录

1 引 言

1.1 问题的提出

随着科学技术的不断发展，社会越来越互联网化，数字化，大数据时代随之到来。有监督分类需要大量的有标签数据，然而获得全部数据标签会花费很多时间且代价高昂，有些情况无法获得数据的全部标签，随着大数据时代的到来，获得数据的全部标签更成为一件不可能的事情。为了充分利用无标签数据的信息，20 世纪 70 年代，出现了半监督学习[51,53]。研究者们提出了自训练(Self-training)[18,22,23,30,67,74,77]、直推学习[35](Transductive Learning)、生成式模型[4,45](Generative Model)等半监督学习方法，后来，出现了协同训练[32] (Co-Training)和转导支持向量机[10](Transductive Support Vector Machine，TSVM)等新方法。实践中发现，大部分数据集都只有少量数据有标签，这给半监督学习提供了良好的应用场景。半监督学习同时使用有标签数据和无标签数据训练模型，可以充分利用无标签数据改善学习器的性能。

半监督学习被广泛应用于文本分类[44]，数据挖掘[21]，根据不同的应用场景，半监督学习可以分为半监督分类[14,18,26,33,41,60]，半监督回归[43,82,84]，半监督聚类，半监督降维四类。半监督分类可以利用无标签数据中的潜在信息，帮助有标签数据训练分类器，生成的分类器比仅使用有标签训练的分类器更好，从而弥补有标签数据样本的不足，因此在实践中应用更加广泛。半监督聚类算法的思想是如何利用先验信息以更好地指导无标签样本的划分过程。现有的算法多数是在传统聚类算法基础上引入监督信息发展而来，基于不同的聚类算法可以将其扩展成不同的半监督聚类算法。半监督回归算法的思想是通过引入大量的无标签样本改进监督学习方法的性能，训练得到性能更优的回归器。现有的方法可以归纳为基于协同训练(差异)的半监督回归和基于流形的半监督回归两类。半监督降维算法的思想是在大量的无标签的样本中引入少量的有标签的样本，利用监督信息找到高维数据的低维结构表示，同时保持数据的内在固有信息。而利用的监督信息既可以是样例的类标签，也可以是成对约束信息，还可以是其他形式的监督信息，主要的半监督降维方法有基于类标签的方法、基于成对约束等方法。

问题 1：自训练 (Self-training)是一种常用的半监督学习算法[5,28,78,79]框架。Self-training 通过利用有标签数据训练一个分类器，然后利用分类器对无标签数据进行分类，从中选择高置信度样本点及其预测标签添加进有标签数据集，迭代训练直至得到

优化后的分类器。自训练在很大程度上依赖于高置信度样本的选择，一旦选取的高置信度样本点中包含错误标签，在迭代训练过程中这种错误将会一直存在，从而降低模型的性能。因此，在基于 Self-training 的算法中如何选取高置信度点至关重要。

问题 2：现有经典的自训练算法 STDP-DE[64](A self-training semi-supervised classification algorithm based on density peaks of data and differential evolution)，SNNRCE[59] (Semi-supervised learning based on nearest neighbor rule and cut edges)， ENaN[68]（Adaptive edited natural neighbor algorithm），ELS[75]（A Fast Parameter-Free Edition Algorithm With Natural Neighbors-Based Local Sets for k Nearest Neighbor），STDPNF[24]（A self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor），STSFCM[15]（Using clustering analysis to improve semi-supervised classification）和 MLSTE[61]（Semi-supervised multi-label image classification based on nearest neighbor editing）均使用欧式距离计算样本之间距离，欧式距离在低维空间上应用效果好，但在高维数据集上会出现“维度诅咒”。SDTC[54] (Semi-supervised self-training for decision tree classifiers)使用马氏距离计算样本之间距离，马氏距离表示数据的协方差距离，与欧式距离相比，马氏距离考虑到了数据各种特性之间的相互联系，但马氏距离的计算不够稳定。因此，如何计算高维数据集样本间的距离是提升自训练算法性能的关键点。

1.2 研究现状

本文主要研究半监督分类算法，半监督分类算法主要有基于差异的方法，基于图的方法，生成式方法和判别式方法，此外，本文总结了基于深度学习的半监督算法，并详细介绍了半监督自训练算法的研究现状。

1.2.1 半监督分类算法

基于差异的方法有：Miryam Elizabeth 等人[58]总结了 29 种半监督异常点检测算法，并在 KEEL 数据库的 95 个不平衡数据集上进行了实验，实验表明，BRM[7](Bagging-RandomMiner)分类器相比于其他 28 种算法具有更好的性能。判别式方法有：Andrea Cappozzo 等人[8]提出了一种自适应判别分析规则 AMDA(adaptive mixture discriminant analysis)，AMDA 基于模型生成分类器，可以在一组无标签集合中检测到无

标签的样本的类，使用高斯混合模型使分类器更具有鲁棒性。生成式方法有：ENDO Yasunori 等人[69]提出了两种基于 FCM(fuzzy *c*-means clustering)聚类的半监督分类算法 SSFCM(Semi-Supervised Fuzzy *c*-Means Clustering)和 ESFCM(semi-supervised entropy regularized fuzzy *c*-means clustering)， SSFCM 可以更广泛的应用于多种数据，比传统的 FCM 算法更加灵活。

基于图[6, 29, 57]的半监督学习算法多使用 k-Nearest Neighbor 来构造图。Feiping Nie 等人[70]提出一种自适应拉普拉斯图的半监督算法 ALGSSL(A Semi-supervised Learning Algorithm via Adaptive Laplacian Graph), ALGSSL 构造拉普拉斯图更新构造稀疏图，而不是直接使用初始图，降低了直接构造图过程中由于异常点和错误的数据特征造成的影响，并通过正则化参数调整将错误标签更正，提升了算法性能。Junliang Ma 等人[33]提出了一种融合局部和全局结构相似性的扩展标签传播算法 FLGSS(Semi-Supervised Classification With Graph Structure Similarity and Extended Label Propagation)，FLGSS 算法首先利用 KNN 构造初始结构图，然后利用不同的预测算法提取全局和局部特征信息并进行融合，将标签相关性与样本相似度进行结合，增强了数据相关性。

1.2.2 基于深度学习的半监督算法

研究者们在深度学习[38, 51]方向对半监督算法也进行了一系列研究。文献[53]提出了一种半监督深度学习的多级放大迭代训练算法，该算法使模型可以在使用无标签数据时克服伪标签的干扰并将对模型有用的特征作为研究重心，显著提高了青光眼诊断的准确性。

文献[72]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[60]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。

1.2.3 半监督自训练算法

本节对现有的半监督自训练算法进行了研究，总结了相关算法的研究现状。王宇等

[59]提出了基于最近邻规则和切边的自训练算法 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。以所有训练样本为基础为每个无标签的样本构建相对邻域图，然后对切边都连接到来自同一类训练样本的无标签样本进行标记后加入训练样本集，对新标记的样本标签进行统计检验，然后用最近邻规则进行迭代直到得到最优分类器。SNNRCE 在减少分类误差方面是有效的，特别是在半监督学习的早期阶段。在 SNNRCE 分类的第一步中，只有那些切边等于零的测试样本才会被分类。因此，初始阶段的分类误差低于标准自我训练的分类误差。这对分类是有益的，因为早期阶段的小错误可以减少后续迭代中的错误强化，从而提高分类性能。

卫志华等人[61]提出了多标签自训练与编辑的算法 MLSTE(Multi-Label Self-training with Editing)，MLSTE 算法结合 ENN (Edited Nearest Neighbors) 数据编辑技术，通过搜索样本点的 k 个最近邻的标签分布来评估样本点是否具有高置信度。MLSTE 算法是针对半监督多标签分类问题而设计的，而以前的算法只考虑半监督学习或多标签学习。ML-kNN（Multi-Label-k-Nearest Neighbor，多标签 k-Nearest）是一种具有代表性的基于 kNN 的多标签分类算法。由于 ML-kNN 不包括半监督学习和数据编辑，因此还对数据集进行了测试，以测试带有数据编辑的半监督学习是否具有良好效果。李俊南等人[22]提出了基于最优路径森林的自训练算法 STOPF(Self-training method based on an optimum path forest)，STOPF 在整个数据集上构建 OPF[1,13,22](Optimum-path forest)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。STOPF 包括三个主要步骤。第一步是在整个数据集上构造 OPF，以揭示特征空间的结构和分布。第二步，利用特征空间的结构和分布来帮助自训练方法标记无标签的样本。然后将这些样本添加进有标签样本集中。在第三步，可以用更新后的有标签数据集训练新的半监督分类器。与以前的算法相比，STOPF 的主要优点是：STOPF 不需要任何用户定义的参数；STOPF 可以训练有效的监督分类器，而不考虑关于数据集分布和形状的假设；与以前的其他算法相比，STOPF 在密度不均匀的数据集上效果更加明显；STOPF 不需要测量置信度，只要初始有标签数据样本能够确定各自的类簇，STOPF 在彼此重叠是数据集上也具有良好的性能，但通过随机选择很难找到这样的有标签样本；STOPF 采用了 3 个自训练算法、12 个 UCI 数据集和 3 个基本分类器，包括 3NN、SVM

和 Cart。实验结果清楚地表明，STOPF 算法在提高 3NN、SVM 和 Cart 的基础分类器的分类精度方面比以前的算法更有效。在实验中，还讨论了有标签数据样本比例对算法造成的影响。当有标签样本的百分比相对较低时，STOPF 通常比其他算法更好；从时间复杂度的实验中可以看出，STOPF 算法相对耗时。这是因为在构造 OPF 时，每次迭代时需要重新计算所有样本的两两之间的距离，但是一些相对距离比较远的样本不应该被考虑在内，这些较远的样本点的计算增加了时间复杂度。STOPF 算法的一些特点： STOPF 算法可以在不考虑数据集的形状和分布的情况下训练一个好的分类器，此外，除了基本分类器的参数外，STOPF 算法不考虑任何参数的选择。

吴迪等人[64]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 STDP-DE(A Self-training Semi-Supervised Classification Algorithm Based on Density Peaks of Data and Differential Evolution), STDP-DE 利用差分进化[62]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。STDP-DE 首先利用密度峰值聚类算法 DPC[47](Density peak clustering)发现数据潜在空间结构，然后在 STDP[63](Self-training semi-supervised classification based on density peaks of data)算法框架中利用差分进化优化 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，将优化后的数据点添加进高置信度样本集中进行迭代训练，得到优化后的分类器。

Jafar Tanha 等人[54]提出了一种决策树 Self-training 算法 SDTC(Semi-supervised Self-training for decision tree classifiers), SDTC 利用基分类器将有标签数据集分为正、负两类，然后使用 Mahalanobis 距离[37]计算每个无标签样本与正、负两类样本均值的距离 p、q，将 $w=|p-q|$ 作为样本分数，选择前 p% 个高分数样本作为高置信度样本。甘海涛等人[15]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 算法将 SSFCM(semi-supervised fuzzy c-means)算法[16, 56]和 SVM[71, 81](Support Vector Machine)进行结合，形成一种半监督分类框架，首先利用 SSFCM 发现整个数据集的空间结构和分布，其次利用 SVM 对有标签样本集进行训练得到初始分类器，然后利用 SSFCM 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。Livieris 等人[31]提

出了一种自动调节自训练半监督算法 AAST (Auto-Adjustable Self-training Semi-Supervised Algorithm)，AAST 算法将所有的分类器放入分类器池中，对分类器池中的每一个分类器进行自训练，然后选出分类器池中分类性能最好的分类器，性能最好指对数据集的所有无标签数据进行标记，标记无标签数据数量最多的分类器。AAST 的一个明显优点是，通过使用不同的学习算法来利用学习模型的误差多样性，并且动态地选择具有最高预测性能的分类器作为最终的分类器。然而，所提出的算法的有效性和计算成本取决于参数 k 的值。随着参数 k 值的增加，基分类器在被评估之前利用无标签数据中的隐藏信息迭代次数也会增加，因此，计算成本和时间成本显著增多。

Junnan Li 等人[23]提出了一种基于自然邻居实例生成的集成自训练算法框架 BoostSTIG，BoostSTIG 介绍了一种新的基于自然邻居实例生成的 boosting 自训练框架，旨在解决或缓解自训练方法中的标签错误问题，它与大多数现有的集成方法和自训练方法兼容。在 BoostSTIG 中，boosting 方法可以通过提高自训练过程中的预测精度来帮助现有的自训练方法解决或减少错误标记。为了使 boosting 方法适用于 SSL，提出了一种具有自然邻居的实例生成技术 IGNaN 来放大初始标记数据。通常，BoostSTIG 包括两个步骤：（1）使用具有自然邻居的实例生成技术来生成新的标记样本并扩大初始标记集；（2） boosting 方法用于帮助自训练方法训练给定的分类器。此外，与处理自训练方法中错误标记的代表性解决方案相比， BoostSTIG 先使用自然邻居生成无标签样本标签来扩大有标签样本集，然后使用集成方法生成自训练分类器，有效改进了有标签样本中误标记的情况。

N. Piroonsup 等人[46]提出了半监督自训练数据预处理方法，使用半监督集群技术分析标记数据的充分性来提升自训练分类器性能，并提出共同标记和联合标记这两个改进数据集标记的方法。为了估计标记的数据分布，应用了种子半监督聚类。聚类结果用于根据相关簇的特点将训练数据分为两组，即未知簇和标记簇。对这两组数据的分析发现，未知聚类的准确性明显低于标记聚类的准确性。为了提高半监督分类的准确性，提出了两种方法来增加未知聚类中标记数据的数量。第一种方法是应用主动学习技术的主动标记。主动标记方法通过选择未知聚类中信息量最大、最具代表性的数据，然后为所选数据赋予类标签。使用主动标记来改进标记数据集可以比增加标记聚类中的标记数据和增加未知聚类中的标签数据更加具有提高自训练分类准确性的目的。增加未知聚类中的标记数据比增加标记聚类中的标签数据更能显著提高分类精度，从未知聚类的质心中选择数据进行标记，这些数据是聚类中最具代表性的。结果表明，用主动标记改进的标记数

据集显著提高了半监督分类的性能。但是，主动标记需要额外手动标记所选数据。为了克服这一限制，提出了第二种方法，即共同标记。共同标记方法应用有效的分类器来自动标记未知聚类中的选定数据。与手动标记不同，共同标记在学习过程中训练共同分类器，并将其应用于标记未知聚类中的数据。为了训练协同分类器，研究了六种分类算法，即近 est 近邻、三近邻、神经网络、支持向量机、决策树和随机森林，以评估半监督分类精度是否有所提高。使用未标记数据的半监督分类的准确性偶尔低于仅使用少数标记数据的监督分类，自训练准确性的下降是由于使用了标记数据集不足的未标记数据。随机森林是尝试分配类标签的六种方法中最好的方法，随机森林的共同标记能够提高因标记数据不足而退化的自训练分类器的性能，通过应用与随机森林的联合标记作为预处理步骤来改进标记的数据集，可以提高半监督分类的准确率。在 UCI 和 Thai 文档图像数据集上的大量实验证实了有标记数据的不足会降低半监督自训练算法的性能。

Junnan Li 等人[26]提出了一种局部中心核的自训练半监督算法框架 LCSSC(An effective framework based on local cores for self-labeled semi-supervised classification), LCSSC 使用了局部核心[11]的思想改善自训练过程中有标签数据不足的缺陷，可以用于处理球形数据和非球形数据。LCSSC 使用带种子的半监督 k 均值（SSKMS）来聚类整个无标签和有标签的数据，然后通过将最接近无标签聚类的质心的代表性无标签样本添加到有标签集合中来改进初始有标签数据，其中这些无标签数据由给定分类器预测。然而，该方法仍存在一些技术缺陷。它无法处理非球面数据，因为 SSKMS 很难有效聚类非球面数据。当初始有标签数据极其稀少时，[53]中方法改进的初始有标签数据可能仍然不足。在该方法中，改进的有标签数据的数量与初始有标签数据的数目相关。如果初始有标签数据极为稀少，则添加的代表性样本将很少，因为 SSKMS 生成的无标签聚类，将每个有标签样本视为类中心，导致改进的有标签集不足。因此，在自标记方法中，由于目前存在上述技术缺陷，很难通过现有工作来改进不充分的初始有标签数据。LC-SSC 框架与任何自标记 SSC (Semi-supervised classification)方法兼容，受到局部核的新颖概念的启发[11]，通过研究球面或非球面数据分布，可以促进 LCSSC 框架在球面或非球形数据集上工作。此外，局部核心还可以帮助 LCSSC 框架有效地改善自标记方法中初始有标签数据不足的情况，即使在初始有标签数据极其稀缺的情况下也是如此。一般来说，LCSSC 框架包含两个步骤：通过将预测的局部核添加到有标签集合中来找到局部核，并改进不充分的初始有标签数据，其中通过主动标记或共标记来预测局部核；使用任何 SSC 自标记方法在改进的有标签数据和更新的无标签数据上训练给定的预测模型。当初

始有标签样本数据不足时，实验结果清楚地验证了 LCSSC 框架在改进自标记 SSC 方法方面的有效性。

综上所述，在多种半监督分类方法中，自训练[41, 80, 83]因为其简单易实现且不需要任何假设的特性成为使用最广泛的半监督分类框架之一。通过自训练得到的分类器的性能主要依赖自训练过程中高置信度样本的选择质量，在自训练中错误标签一旦出现就会一直存在于后续的迭代过程中，从而影响分类器的性能。因此，在基于 Self-training 的算法中如何选取高置信度点至关重要。集成学习作为常用的一种处理错误标签的方法在生活中被广泛使用，但是集成分类器在只有少量有标签的数据集面前无法生成合适的分类器。

令 n 表示输入样本的个数，d 表示维度，t 表示迭代次数，k 表示类簇数。SNNRCE 算法主要在于构造切边邻近图，然后使用最近邻规则进行分类，其整体算法复杂度为 $O\left(tdn^3\right)$。MLSTE 算法使用 ENN 技术进行数据编辑，计算 ENN 的时间复杂度为 $O\left(n^3\right)$，然后使用 KNN 基分类器进行自训练，总体复杂度为 $O\left(kdn^3\right)$。STOPF 算法需要构建 OPF，构建 OPF 的时间复杂度为 $O\left(n^2\right)$，然后使用 KNN（K=3）基分类器进行自训练，总体时间复杂度为 $O\left(tdn^3\right)$。STSFCM 算法主要在于训练 SVM，训练 SVM 的时间复杂度为 $O\left(n^3\right)$，故算法整体复杂度为 $O\left(n^3\right)$。通过以上分析发现，目前的半监督自训练算法需要计算样本间的两两距离，时间复杂度比较高。

1.3 创新点

本文主要有两个创新，分别为：（一）本文从 Ball-k-means 算法[65]得到启发提出了一种快速球簇划分编辑的半监督 Self-training 算法(Semi-supervised fast Self-training algorithm with partition editing of sphere clusters -EBSA)，Ball-k-means 算法首先使用 k-means 算法[52]选出球簇中心，然后将数据集划分为“稳定区域”和“活跃区域”，将“活跃区域”划分为一个个环形区域，在计算中仅需计算各球簇中心之间距离和环形区域中点与球簇中心距离，减少了计算量，降低了算法时间复杂度。

EBSA 利用球簇划分方法将数据集划分为稳定区域和争议区域，使用球簇划分编辑算法评估稳定区域内的样本是否为误标记样本点，对误标记样本点进行编辑，且删除争议区域内的样本，提升了高置信度样本选取质量。由于在每次迭代中仅需计算样本点与

球簇中心的距离，计算量小，速度快。

（二）自 1970 年以来，研究者们提出，距离度量缺少差异性这一关键因素，即密度较高区域的两个样本的相似性低于密度较低区域两个样本的相似性。现有的距离度量方式如欧式距离、马氏距离，计算过程中没有考虑差异性这一关键因素。基于此，本文提出一种块估计近邻编辑 Self-training 算法 MDSF， MDSF 使用不相似性度量方式计算样本距离，并定义了块估计近邻关系，基于块估计近邻关系构建块估计近邻图，提出了块估计近邻编辑算法对数据进行编辑，提升了高置信度样本的选取质量。

1.4 本文的章节安排

本文其他部分工作描述如下：第二部分为相关工作介绍，相关工作包括本文使用的 Self-training 框架，本文提出的两个算法所使用的对比算法介绍以及部分数据编辑技术介绍。第三部分为提出的算法快速球簇划分编辑的半监督 Self-training 算法（EBSA）详细说明与介绍与 EBSA 算法的实验结果与分析。第四章为提出的块估计近邻编辑 Self-training 算法 MDSF 详细说明与介绍与 MDSF 算法的实验结果与分析，第五部分是对全文的总结和本文工作需要进一步完善和深入研究的方向的介绍。

2 相关工作

这一部分列举了本文出现的重要符号和公式，介绍了一些基本理论知识和对比算法。

2.1 符号及说明

$X=\{x_1,x_2,x_3,\cdots,x_n\}$， X 表示包含 n 个样本的数据空间， $x_i\in\Re^{d\times1}$ 表示维度为 d 的一个样本。

$Y=\{y_1,y_2,y_3,\cdots,y_n\}$，$Y$ 表示有 k 个可能的类的样本标签空间，y_i 表示样本 x_i 的标签。

$L=\{(x_1,y_1),(x_2,y_2),\cdots,(x_l,y_l)\}$： L 表示样本数据及其对应标签集合，即有标签样本集。

$U=\{x_1,x_2,\cdots,x_u\}$： U 表示无标签样本集。

$S=\{x_1,x_2,\cdots,x_f\}$： S 表示具有 f 个高置信度样本的集合。

2.2 自训练(Self-training)

半监督学习通过结合无标签和有标签数据信息进行学习。自训练是典型的半监督学习框架之一。在自训练过程中，首先在有标签数据集上训练一个基分类器[73]，然后在无标签数据集中选择高置信度样本点集加入有标签样本集中，进行迭代标签。Self-training 算法步骤如下：

2.3 图编辑自训练(SETRED)

SETRED[27] (Self-training with Editing)利用一种特定的数据编辑方法从自标记数据中识别被错误标记的样本点，以此来消除错误标记的样本点（噪声点）对整个训练进程的影响。SETRED 在自训练的每次迭代过程中加入局部切边权重(CEW-Cut Edge Weight)来评估新标记的样本点是否具有高置信度，然后仅将具有高置信度的自标签的数据加入有标签数据集中，迭代得到优化后的分类器。

Self-training 算法

输入　有标签样本集L，无标签样本集U

输出　分类器H

1. 初始化高置信度样本集$S=\varnothing$
2. WHILE$U\neq\varnothing$ DO
3. 在有标签样本集L上训练分类器H
4. 利用分类器H给无标签样本集U赋予标签
5. 从无标签样本集U选取部分样本组成高置信度样本集合S，并由H赋予伪标签
6. 更新$L\leftarrow L\cup S$，$U\leftarrow U-S$
7. END WHILE
8. 当分类器H稳定或$U=\varnothing$，输出分类器H

SETRED 算法步骤：

1) 选择基分类器，训练有标签数据集L得到初始分类器H；
2) 从无标签数据集U中随机选择部分数据形成U',使用H选择U'中更高置信度的点，构成集合L'；
3) 使用$L\cup L'$构造相关邻接图G，G中的两个具有不同类标签的点之间连接的边称为切边；在G中任意一个样本点x_i，与其有边相连的样本点集合称为x_i的近邻集N_i，一个近邻集N_i的样本点拥有相同的类标签，当某一个样本点与其近邻之间有多条切边时，那这个样本点称为存疑点；
4) 使用局部切边权重方法识别L'被错误标记的样本点，删除误标记样本点，生成L''加入训练集中;
5) 生成最终的分类器H'。

2.4 STDPNF

基于密度峰值的自训练的无参数噪声过滤方法 STDPNF[24](A Self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor)，STDPNF 提出了一个新的本地噪音过滤器，它可以同时使用有标签数据和无标签数据来去除噪声，克服了自训练方法中现有局部噪声过滤器的技术缺陷。STDPNF 重新定义密度峰值聚类算法 DPC[47](Density peak clustering)发现的数据的空间结构，使用自训练方法标记无标签样本和扩展有标签数据集，将局部噪声过滤器过滤噪声后的样本加入训练集中重新训练后得到分类器。

STDPNF 算法步骤：

1) 用 DPC 发现数据集特征空间的结构和分布，为所有的样本点标记顺序(order);
2) 根据标记顺序在无标签样本集U中寻找对应的x_i，当所有的样本点都可以根据标记顺序查找到时，根据有标签样本和标记的顺序进行训练，得到分类器H；当无标签数据集U中部分标记顺序对应的x_i找不到时，将这些样本点加入基分类器中重新训练，并使用局部噪声过滤器 ENaNE 进行噪声过滤并标记，将新标记的样本点加入有标签数据集L中；
3) 使用扩展的有标签数据样本集对给定分类器进行重新训练，得到新的分类器H'

2.5 STDP-CEW

结合密度峰值和切边权值的自训练算法 STDP-CEW[30](Self-training Algorithm Combining Density Peak and Cut Edge Weight)， STDP-CEW 利用密度聚类算法发现数据集的空间结构，从中选出具有代表性的无标签数据进行标记，使用切边权值进行假设检验来评估样本是否具有高置信度，最后将具有高置信度的点加入有标签数据集，迭代得出优化后的分类器。

STDP-CEW 算法步骤：

1) 利用 DPC 发现整个样本空间的潜在结构，找出有标签数据的“上一个”样本集L'和“下一个”样本集L''，“上一个”和“下一个”都是以局部密度为基础，从低密度指向高密度，“下一个”指比某一样本点局部密度大的点，且两点之间有连

接关系；

2) 选择决策树作为基分类器，将初始有标签数据 L 进行训练得到初始分类器 H；
3) 对 L 中所有有标签样本的“上一个”和“下一个”无标签样本进行标记，使用切边权值进行假设检验来评估“上一个”和“下一个”样本是否具有高置信度，将具有高置信度的样本点及其标签添加进有标签数据集中进行训练，直到所有样本的“上一个”和“下一个”的样本集合为空；
4) 生成最终的分类器 H'。

2.6 STDPNaN

一种基于密度峰值和自然邻居的自训练算法 STDPNaN[74](Self-training method based on density peaks and natural neighbors), STDPNaN 使用集成分类器来提高 Self-training 算法的标签预测能力。STDPNaN 通过引入自然近邻提出了一种无参数密度峰值聚类算法 DPCNaN(parameter-free density peaks clustering)，DPCNaN 通过使每个样本 x_i 指向其最近的具有较高局部密度的样本来发现整个数据集的空间结构及分布，STDPNaN 在 DPCNaN 构造的数据空间基础上对 L 中所有有标签样本的 “下一个”和“上一个”无标签样本进行标记，将标记过的样本点添加进有标签样本集 L 中，迭代得到优化后的分类器 H。

STDPNaN 的主要优点是（1） 它是无参数的；（2） 由于采用了集成分类器，它具有更好的预测能力；（3） 它可以处理球面或非球面分布的数据集。通过实验分析和验证，STDPNaN 在改进方面优于最先进的算法 KNN、SVM 和 CART 的分类精度而不受标记数据数量的限制。权衡分类精度和运行时间，STDPNaN 要求的运行时间也可接受。尽管集成学习可以提高自训练过程的预测能力，但也增加了计算时间。

2.7 基于块的不相似性度量

Kai Ming Ting[55]等人提出了基于块的不相似性度量，令 F 表示概率密度函数，D 表示样本数据，$H \in \Psi(D)$ 表示将空间 D 划分为一个个不重叠的非空域的层次划分模型。令 x_i 表示样本数据 D 中第 i 个样本，令 $1(\bullet)$ 表示指示函数，令 $R(x_i, x_j | H; D)$ 表示包含 x_i 和 x_j 的在 H 和 D 下的最小域：

$$R\left(x_i,x_j\left|H;D\right.\right)=\underset{h\subset H,s.t.\left\{x_i,x_j\right\}\in h}{\arg\min}\sum_{z\in D}1\left(z\in h\right) \tag{2.7.1}$$

令 $P_F\left(\Delta\right)$ 表示使用概率密度函数 F 计算得到的 Δ 的概率，令 $R\left(x_i,x_j\left|H;D\right.\right)$ 的期望概率 $m\left(x_i,x_j\left|H;D\right.\right)$ 为样本 x_i 和 x_j 关于 D 和 F 的基于块的不相似性。

$$m\left(x_i,x_j\left|H;D\right.\right)=E_{\Psi(D)}\left[P_F\left(R\left(x_i,x_j\left|H;D\right.\right)\right)\right] \tag{2.7.2}$$

令 $H_b\in\Psi\left(D\right)\left(b=1,\cdots,B\right)$ 为有限个模型，令 $P\left(R\right)=\frac{1}{\left|D\right|}\sum_{z\in D}1\left(z\in R\right)$，则公式(2.7.2)变为：

$$m_e\left(x_i,x_j\left|D\right.\right)=\frac{1}{B}\sum_{b=1}^{B}P\left(R\left(x_i,x_j\left|H_b;D\right.\right)\right) \tag{2.7.3}$$

2.8 STDP

Di Wu 等人[63]提出了一种基于密度峰值的半监督分类自训练框架 STDP (Self-training semi-supervised classifification based on density peaks of data)，STDP 首先使用密度峰值算法发现整个数据空间的潜在结构，然后整合数据空间的真实结构并将其添加进自训练过程中进行迭代训练得到优化后的分类器。STDP 有三个优点：（1）STDP 不受初始有标签数据和整个数据空间分布的限制；（2） STDP 是一个没有先决条件的建设性模型；（3） STDP 适用于任何监督算法，通过使用大量的无标签数据来提高算法性能。STDP 在使用 SVM、KNN 和 CART 的不同监督算法作为基础分类器的情况下，选择了两种具有代表性的自标记 SSC 算法进行比较，分析了有标签样本数据比率的影响、截断距离 d_c 的影响以及噪声的影响等其他问题。

总之，STDP 比半监督 3NN 训练和半监督 FCM 更有效，可以提高数据集 Gauss50、Gauss50x 和钞票认证上的监督算法的性能。STDP 算法适合于非球形分布数据，且使用 SVM 的监督算法比使用 KNN 或 CART 的监督算法具有更好的性能，可能是 STDP 算法的学习策略与支持向量机的结构风险最小化原理相似，该算法基于通过发现数据密度峰值而发现的数据空间结构迭代学习分类器。此外，在数据集具有大量强重叠数据（如数据集波形）的情况下，STDP 算法可能会失去有效性。

2.9 数据编辑技术

(1) Depuration data editing (DE)

J.S. Saanchez 等人[49]提出了一种净化数据编辑技术 DE，DE 首先搜索每个有标签样本 x_i 的 k 近邻构成集合 N_{x_i}，当 N_{x_i} 中有超过 k' 个样本的标签为 y 时，$y_i = y$，其中 $\frac{(k+1)}{2} \le k' < k$，DE 可以修改被误标记的样本标签并在训练过程中过滤噪声数据。

(2) Local sets edition

LSEdit[25](Local sets edition)：Junnan Li 等人提出了一种局部集合编辑技术 LSEdit，LSEdit 首先使用自然近邻搜索每个样本的自然近邻构成局部集合，然后使用噪声因子函数来评估每个样本是否为噪声样本点，将经过编辑的样本点及其标签添加进编辑集合来过滤噪声数据。

(3) Cut edges weight statistic

CEWS[39](Cut edges weight statistic)：CEWS 首先构造相关邻接图 G，G 中的两个具有不同类标签的点之间连接的边称为切边；在 G 中任意一个样本点 x_i，与其有边相连的样本点集合称为 x_i 的近邻集 N_i，一个近邻集 N_i 的样本点拥有相同的类标签，当某一个样本点与其近邻之间有多条切边时，那这个样本点称为噪声点，最后使用局部切边权重进行假设检验。

(4) Relative Neighborhood Graph Edition

RNGE[50](Relative Neighborhood Graph Edition)：RNGE 首先构造近邻无向图 $G=(V,E)$，$V=X$，E 为边组成的集合，当 $\forall x_k \in X, k \ne i, j$，$(x_i, x_j) \in E \Leftrightarrow \|x_i - x_j\|^2 \le \|x_i - x_k\|^2 + \|x_j - x_k\|^2$ 时，x_i 和 x_j 称为图邻居，RNGE 给出了相关近邻图边的定义，当 $\forall x_k \in X, k \ne i, j$，$(x_i, x_j) \in E \Leftrightarrow \|x_i - x_j\| \le \max(\|x_i - x_k\|, \|x_j - x_k\|)$，在非近邻的点之间添加新的边后进行编辑。

3 快速球簇划分编辑的半监督 Self-training 算法 （EBSA）算法

现有的半监督 Self-training 算法，如：SNNRCE、MLSTE、STSFCM、SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法计算复杂度高。为了降低半监督 Self-training 算法计算复杂度，提升半监督分类算法的性能，本文提出快速球簇划分编辑的半监督 Self-training 算法(EBSA)。

3.1 定义

定义 1 簇心和簇半径

令C'表示一个簇，x_i表示属于C'的第i个样本，$\left|C'\right|$表示C'包含的样本个数。令ς表示簇心，r表示簇半径。

$$\varsigma = \frac{1}{\left|C_p\right|}\sum_{i=1}^{\left|C_p\right|} x_i \tag{3.1.1}$$

$$r = \max\left(\left\|x_i - \varsigma\right\|\right) \tag{3.1.2}$$

簇心ς为簇所有样本求均值，簇半径r为属于簇C'的样本x_i与簇心ς的最大距离。

定义 2 球簇

以簇心ς为球心，簇半径r为半径的球形区域称为球簇。

定义 3 球簇类别

$$k_{C_p} = \arg\max_i \left|F_i\right| \tag{3.1.3}$$

球簇C_p的球簇类别为球簇C_p内样本数目最多的样本标签。

定义 4 球簇近邻簇

令C_p和C_q表示两个不同的球簇，令ς_p和ς_q表示这两个球簇的球心，r_p表示球簇C_p的半径。

球簇C_p和C_q之间的距离l_{pq}：

$$l_{pq} = \left\|\varsigma_p - \varsigma_q\right\|_2 \tag{3.1.4}$$

当l_{pq}满足公式(3.1.5)时,C_q称为C_p的近邻簇。

$$\frac{1}{2}l_{pq} < r_p \tag{3.1.5}$$

这种近邻关系是非对称的。任意两个球簇C_p和C_q可能存在以下 3 种关系：

1）C_p和C_q互为近邻簇；

2）C_p是C_q的近邻簇，C_q不是C_p的近邻簇；

3）C_p和C_q没有近邻关系。

给定球簇C_1，C_2，C_3和C_4，如图 1 所示：

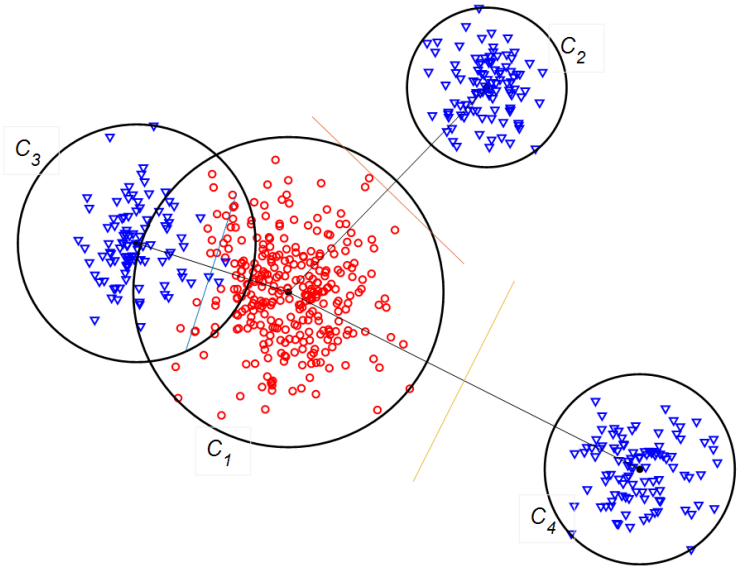


图 3.1 球簇近邻关系图

图 3.1 中线段为球簇C_1和C_2，C_1和C_3，C_1和C_4的距离l_{12}，l_{13}，l_{14}的垂直平分线。由图 3.1 可知，C_1和C_3互为近邻簇，C_2是C_1的近邻簇，C_1不是C_2的近邻簇，C_1和C_4没有近邻关系。

定义 5 稳定区域和活跃区域

令N_{C_p}表示C_p的近邻簇的球心点的集合：

$$\varsigma_q \in N_{C_p}\left(N_{C_p} \neq \varnothing\right) \tag{3.1.6}$$

$$\hat{r} = \frac{1}{2}\min\left(l_{pq}\right)_{\varsigma_q \in N_{C_p}} \tag{3.1.7}$$

当C_q的球心满足公式(3.1.7)时，C_p的稳定区域C_{pw}为以ς_p为簇心，以$\hat{r}$为半径形成的球形区域。C_p的活跃区域C_{ph}为$C_p - C_{pw}$。

定义 6 待划区域

给定球簇C_p和C_q，C_q为球簇C_p的近邻簇，ς_p和ς_q分别表示这两个球簇球心，r_p和r_q分别为两个球簇的半径。

$$\tilde{r} = \frac{1}{2} l_{pq} \tag{3.1.8}$$

球簇C_p的待划区域C_{pdh}为以ς_q为簇心，以$\tilde{r}$为半径的球形区域。

定义 7 争议区域

球簇C_p的争议区域为C_{pzy}：(C_q为C_p的近邻簇)

$$C_{pzy} = C_{qdh} \cap C_p \tag{3.1.9}$$

给定球簇C_p和C_q，C_q为球簇C_p的近邻簇，ς_p和ς_q分别表示这两个球簇球心，r_p和r_q分别为两个球簇的半径。$\forall x_i \in C_{pzy}$，$\left\| x_i - \varsigma_p \right\|_2 \le r_p$，$\left\| x_i - \varsigma_q \right\|_2 \le r_q$，基于聚类假设进行划分时，点$x_i$既可以被分类到球簇$C_p$中，也可以被分类到球簇$C_q$中，因此争议区域内的样本容易被误分的可能性增大,这种错误划分在Self-training迭代过程中会被放大，从而导致分类器性能下降。

3.2 球簇划分编辑算法

给定球簇C_p和C_q，C_q为球簇C_p的近邻簇，ς_p和ς_q分别表示这两个球簇球心，r_p和r_q分别为两个球簇的半径。球簇C_p的争议区域为C_{pzy}。设C_p中有b个样本点，$X_p = \{x_1, x_2, x_3, \cdots, x_b\}$为$C_p$内所有样本点的集合，$Y_p = \{y_1, y_2, y_3, \cdots, y_b\}$为$C_p$内所有样本点的标签的集合，令球簇$C_p$内标签为$K_1$的样本数量最多，令$C_{pw}$表示球簇$C_p$的稳定区域,令$X_{pw} = \{x_1, x_2, x_3, \cdots, x_w\}$为$C_{pw}$内所有样本点的集合，$Y_{pw} = \{y_1, y_2, y_3, \cdots, y_w\}$为$C_{pw}$内所有样本点的标签的集合，$\forall j \in [1, w]$，$if\ \exists y_j \in Y_{pw} \ne K_1$，$y_i \to K_1$。

算法 1 描述了球簇划分编辑过程。图 3.2 是球簇划分编辑示意图：

Algorithm 1 球簇划分编辑算法(EC)

输入 有标签样本集L，无标签样本集U，聚类数目 K

参数 高置信度样本集 S，邻居类簇N_{pq}

输出 球簇稳定区域集合C_{pw}，争议区域集 N_{pzy}

$S=\varnothing$ ，$C_p=\varnothing, N_{pq}=\varnothing$

在有标签样本集L上训练分类器H

$\forall x_i \in U, y_i = H(x_i)$

for $i=1,2,\cdots K$

    $C_{y_i} = C_{y_i} \cup x_i$

End for

For each C_p in C

    使用公式(3.1.1)，(3.1.2)，(3.1.3)计算球簇半径r_p，球簇中心ς_p，球簇类别 k_{C_p}

End for

计算两两球簇中心之间的距离l_{pq}

For each C_p in C

    If $\frac{1}{2} l_{pq} < r_p$

        $N_{pq} = N_{pq} \cup C_q$

    End if

End for

For C_q in N_{pq}

    由公式(3.1.7)计算出稳定区域C_{pw}

    计算球簇划分区域C_{pdh}

19	计算争议区域C_{pzy}
20	$N_{pzy}=C_{pzy1}\cup C_{pzy2}\cup\cdots\cup C_{pzyo}$
21	End for
22	for x_i in C_{pw}
23	if $y_i\neq k_{C_p}$
24	$y_i==k_{C_p}$
25	end if
26	End for

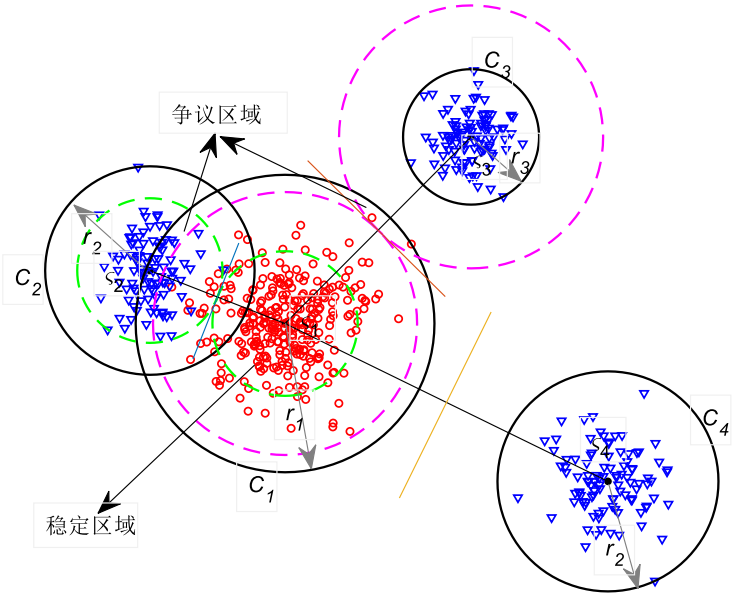


图 3.2 球簇划分编辑

球簇划分编辑就是先将球簇进行划分，然后将稳定区域内的点进行编辑。因为在 Self-training 算法中，噪声点无可避免，因此需要将稳定区域内的样本进行编辑，降低误标记样本的影响，就此提升 Self-training 算法的分类性能。

在图 3.2 中，用C_1，C_2，C_3和C_4分别表示四个不同的球簇，令ς_1，ς_2，ς_3和ς_4表示这四个球簇质心，使用r_1，r_2，r_3和r_4表示这四个球簇的半径，C_2和C_3为球簇C_1的近邻簇，球簇C_1的稳定区域和争议区域如图所示：其他球簇的稳定区域与争议区域划分与

C_1相同，争议区域为绿色虚线与黑色实线围成区域，紫色虚线与黑色实线围成区域。

3.3 高置信度样本选取算法

在 Self-training 算法中，如何选取高置信度样本是至关重要的一步。在提出的算法 EBSA 中，先将数据集进行球簇划分，然后划分出各个球簇的稳定区域与争议区域，在对球簇中的样本进行数据编辑的基础上选取高置信度样本。算法 2 描述了高置信度样本选取算法：

算法 2 高置信度样本选择(SG)	
输入	球簇稳定区域集合C_{pw}，球簇集合 C，争议区域 N_{pzy}
参数	球簇C_p中的所有样本的集合 X_p
输出	高置信度样本集合S
1	$S=\varnothing$
2	For C_p in C
3	$S^{'}=S\cup eC_p, S^{''}=X_p-N_{pzy}$
4	$S=S^{'}\cup S^{''}$
5	End for

争议区域的样本在分类过程中会影响分类性能，因此将属于球簇C_p的争议区域的样本删除，留下经过编辑的稳定区域的样本和无争议区域的样本作为高置信度样本，在后续的迭代过程中可以降低误分类造成的影响，提升分类器性能。

下面举例说明高置信度样本的选取过程：在图 3.3 中，C_1，C_2和C_3表示球簇，令ς_1，ς_2和ς_3表示这三个球簇质心，使用r_1，r_2和r_3表示这三个球簇的半径，C_2和C_3为球簇C_1的近邻簇。

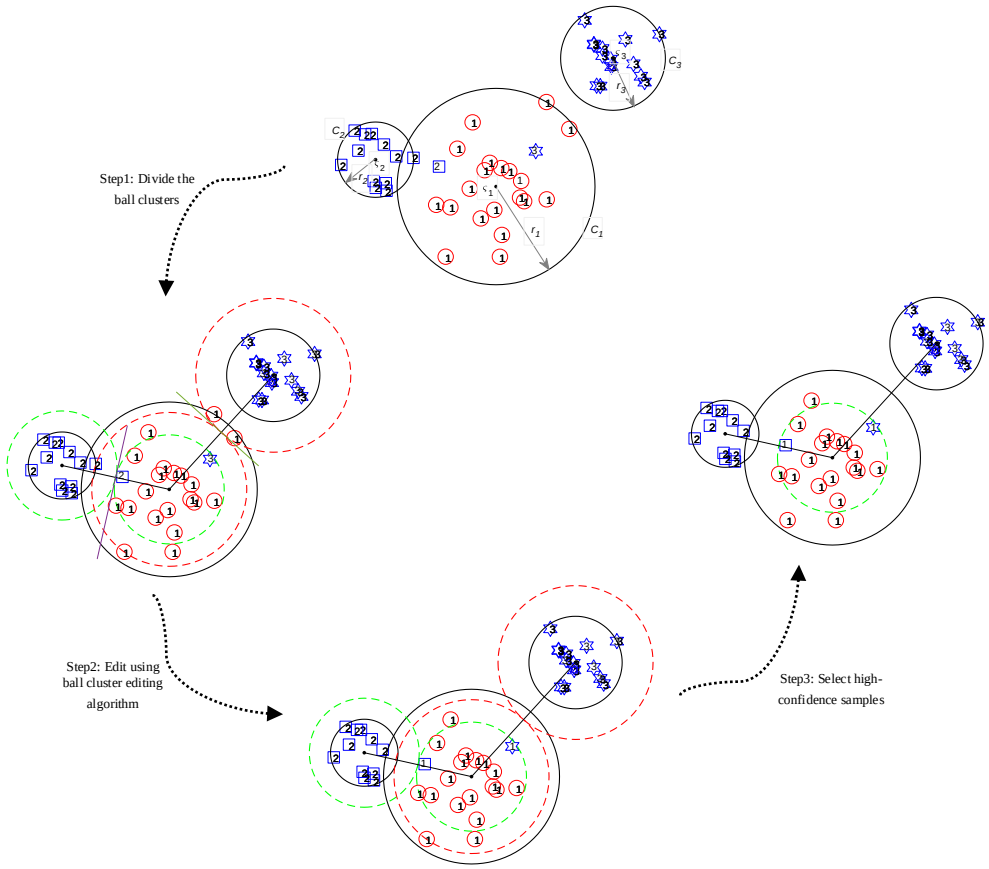


图 3.3 高置信度样本选择过程

在图 3.3 中，(a)为原始空间样本分布图，(b)为对(a)进行球簇划分后的图，(c)为经过球簇编辑后的样本图，图(c)中标签为 1 的样本点即为球簇C_1的高置信度样本。因为在提出的算法 EBSA 中，不需要计算样本点两两之间的距离，只需要计算出球簇中心，球簇半径和球簇质心之间距离，因为已经划分出稳定区域与争议区域，而高置信度样本的选取只需要将稳定区域和争议区域内的样本进行球簇编辑，因此可以在不降低分类器性能的前提下降低计算复杂度，提升分类器速度。

3.3 快速球簇划分编辑的半监督 Self-training 算法 （EBSA）算法

EBSA 将数据集先利用决策树[9, 34, 40, 48]基分类器进行分类，将分类后的样本进行球簇划分，划分出稳定区域和争议区域，由基分类器给这些样本赋予标签，利用球簇划分编辑算法对每个球簇内的样本点进行编辑，将编辑后的样本添加进高置信度样本集中，使用 Self-training 方法训练迭代得到优化后的分类器。将 EBSA 算法总结如下：

算法 4 EBSA 算法

输入 有标签样本集L，无标签样本集U，球簇类别总数K

参数 高置信度样本集 S

输出 分类器H

1. $S=\varnothing$
2 在有标签样本集L上训练分类器H
3. For $i=1,2,\cdots K$
4. $y_i=H(x_i)$，$C_{y_i}=C_{y_i}\cup x_i$
5. End for
6. 计算球簇C
7. WHILE
8. 用算法 1 计算球簇稳定区域集合C_{pw}和球簇争议区域集合N_{pzy}
9. 用算法 2 选择高置信度样本点生成高置信度样本集合S
10. 更新有标签样本集$L\leftarrow L\cup S$
11. 使用更新后的有标签样本集训练分类器H
12. END WHILE
13. 输出分类器H

3.4 实验设置

实验在 32G 内存、64 位 Windows 10 操作系统和 Inter Core i9 处理器的环境下进行。IDE 编程环境为 MATLAB2019b 版本。

实验所用数据集均为公开数据集。数据集详细信息如表 3.1 所示：

在 20 个数据集中，AR、COIL20、MINIST2k2k 、ORL、Palm、UPS 和 YaleB 均为图像数据集，且数据集都比较大，Isolet 是口语音频数据集，数据集比较大， Solar 为太阳耀斑数据集，Sonar 为声纳数据集，Cleve 、Heart 、Solar 和 Sonar 数据集都比较小。使用了 5 个来自 uci 数据库的小数据集，Heart 为心脏单质子发射计算机断层扫描（SPECT）图像诊断信息数据集。

表 3.1 数据集

序号	数据集	样本数	维数	球簇数
1	AR[36]	1680	1024	120
2	Cleve[20]	303	13	4
3	COIL20[2]	1440	1024	20
4	Isolet[12]	1560	617	2
5	UPS[19]	2007	256	10
6	Heart[3]	270	13	2
7	MINIST2k2k[66]	4000	784	10
8	ORL[1]	400	1024	40
9	Palm[2]	2000	256	100
10	Sonar[3]	208	60	2
11	Solar[4]	323	12	6
12	YaleB[17]	2414	1024	38
13	BUPA	345	6	2
14	FERET[76]	1400	1024	200
15	MSRA25	1799	256	12
16	Yeast	1484	1470	10
17	autouni	205	25	6
18	Ecoli	336	7	8
19	pendigits	3498	16	10
20	uspst	2007	256	10

选取同类的 SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法进行对比实验。对比算法的运行参数根据原文设置，具体情况如表 3.2 所示：

[1] http://www.uk.research.att.com/facedatabase.html
[2] https://www.gwern.net/Crops
[3]https://machinelearningmastery.com/standard-machine-learning-datasets/
[4] https://www.kaggle.com/anikannal/solar-power-generation-data

表 3.2 对比算法参数设置

算法	参数
SETRED	$\theta = 0.1$
STDP-CEW	$\alpha = 2$， $\theta = 0.1$
STDPNF	$\alpha = 2$

α 为 DPC 算法中的距离截取阈值，θ 为置信度水平阈值。

值得说明的是，所提出的 EBSA 算法不需要输入参数，这有助于 EBSA 算法的广泛使用。

实验中，选择决策树作为基分类器。决策树运行参数如下：使用基尼多样性指数作为划分准则进行属性划分，训练和测试时所使用的类簇名称与标签集合 Y 的标签一致，根据标签集合 Y 中的类频率设置每个类的先验概率，当节点进行再划分时，要求每个节点样本数不少于 2 个，将所有特征数作为划分时所考虑的最大特征数，最大分裂数为数据集所有样本数-1，即 $n-1$。

为了验证提出的算法 EBSA 的分类性能，做了三个实验，实验介绍如下：

（1）实验 1：在有标签样本比例为 10%的 20 个数据集上与四个对比算法进行比较；

（2）实验 2：为了研究少量有标签样本比例对实验的影响，在 2%~20%的有标签比例下进行实验；

（3）实验 3：为了验证提出的算法 EBSA 具有数据编辑的效果，可以降低噪声数据对分类性能的影响，在 10%的有标签比例下与其他数据编辑算法进行实验，选择 1%~10%的有标签样本进行标错作为噪声数据进行实验，研究 EBSA 算法去噪的性能。

为了评估分类性能是否显著，在 95%的置信度水平上进行了威尔科克森(Wilcoxon)符号秩检验。符号“+”，“−”，“~”分别表示提出的算法 EBSA 与对比算法相比，有更高、更低，相同的置信度。

使用准确率（Accuracy）和 F 分数（Fscore-Micro）作为分类评价指标，上述指标可由表 3.3 混淆矩阵计算得出。

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{3.5.1}$$

$$precision = \frac{TP}{TP + FP} \tag{3.5.2}$$

$$recall = \frac{TP}{TP + FN} \tag{3.5.3}$$

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 * precision * recall}{precision + recall} \tag{3.5.4}$$

表 3.3 混淆矩阵

		predicted class	
		positive	negative
actual class	positive	TP(True Positive)	FN(False Negative)
	negative	FP(False Positive)	TN(True Negative)

3.5 分类性能分析

随着大数据时代的到来，获得数据的全部标签难以实现，在现实生活中，大部分数据集仅有小部分数据包含标签。因此，本文为了研究各算法在具有少量标签样本的数据集上的性能，参照一些半监督分类算法[15, 31]选择样本的比例进行选择。随机抽取数据集中 10%带有原始标签的样本数据形成有标签集合，将剩余的 90%的样本数据的标签删除后作为无标签样本进行实验。避免实验结果可能出现的偶然性，本文所有实验均进行了 50 次。本文实验经过统计检验，实验结果分析呈现显著性。

在给定的 20 个基准数据集上进行实验，初始有标签比例为 10%，即训练数据集中有 10%的样本有标签，90%的样本无标签，5 个算法均运行 50 次。分类性能指标选择准确率（Accuracy）和 F 分数（Fscore-Micro），计算 50 次运行结果的平均值（mean）和方差（std）。实验结果如表 3.4、表 3.5 所示。

从表 3.4、表 3.5 中的实验结果可得如下结论：

（1）提出的算法 EBSA 在 19 个数据集上的分类准确率均高于其他四个对比算法，在 pendigits 数据集上低于 SETRED、STDP-CEW 和 STDPNaN 算法，高于 STDPNF 算法。F 分数在 16 个数据集上高于其他四个对比算法，在 AR、Yeast 和 Ecoli 数据集上比 STDPNF 算法低 0.07%，0.05%和 0.03%，但高于其他三个对比算法 SETRED、STDP-CEW 和 STDPNaN。由此看出，在数据集 Cleve 、COIL20、Heart 、Isolet 、MINIST2k2k 、ORL、Palm、Solar 、Sonar 、UPS、YaleB、BUPA 、FERET32、MSRA25、autouni 和 uspst 上，提出的算法 EBSA 的分类性能（准确率和 F 分数）均高于对比算法 SETRED、

STDP-CEW、STDPNaN 和 STDPNF；

表 3.4 各算法在 20 个数据集上的分类性能（准确率）（均值±标准差）

Accuracy	EBSA	SETRED	STDPCEW	STDPNF	STDPNaN
AR	**96.23±0.61**	81.58±1.03	78.87±0.95	96.18±0.47	79.50±1.52
Cleve	**86.17±2.13**	79.94±4.28	79.04±5.62	84.38±3.78	79.88±5.64
COIL20	**92.61±1.07**	83.70±1.45	80.97±2.23	90.89±1.42	82.59±1.93
Heart	**74.55±3.81**	72.98±3.82	74.88±3.61	70.36±7.52	69.56±4.49
Isolet	**72.36±1.81**	69.46±3.53	70.01±2.38	71.78±1.88	67.87±2.93
MINIST2k	**85.94±0.62**	78.20±0.95	77.08±1.31	85.91±1.28	77.64±1.01
ORL	**95.58±0.66**	83.19±1.48	80.46±0.76	94.79±0.88	81.30±1.27
Palm	**95.46±0.47**	79.90±1.30	75.98±0.81	95.22±0.55	77.80±1.51
Solar	**86.25±1.54**	81.35±4.01	84.61±2.06	81.30±2.69	81.43±3.20
Sonar	**75.75±2.81**	65.63±9.00	66.36±4.95	73.91±4.93	65.87±4.30
UPS	**88.23±1.39**	82.93±1.46	82.33±1.52	87.82±1.70	82.60±1.74
YaleB	**91.93±1.68**	72.04±1.45	67.51±1.18	91.42±1.93	68.73±1.76
BUPA	**67.41±4.72**	63.7±3.6	61.22±3.34	66.95±3.12	62.71±3.86
FERET32	**97.75±0.5**	89.9±1.11	86.86±0.96	97.67±0.38	88.13±1.06
MSRA25	**90.63±0.87**	87.53±0.91	86.88±1.62	90.4±0.64	86.45±1.18
Yeast	**93.51±0.02**	80.22±1.69	81.51±2.21	93.42±0.15	80.97±2.6
autouni	**79.91±1.36**	76.67±3.56	74.46±4.63	79.68±2.08	73.19±6.14
Ecoli	**93.03±0.18**	87.87±2.09	88.26±3.36	92.73±0.73	89.77±2.49
pendigits	90.37±0.57	91.53±1.23	91.55±0.74	89.88±0.97	**91.68±0.6**
uspst	**88.34±1.03**	83.48±1.65	82.58±1.41	87.76±1.12	82.9±0.95
WSR-test	N/A	+	+	~	+
Ave.ACC	**87.39**	79.36	78.37	86.5	78.34
Ave.std	1.32	2.3	2.1	1.79	2.34

表 3.5 各算法在 20 个数据集上的分类性能（F 分数）（均值±标准差）

fs	EBSA	SETRED	STDPCEW	STDPNF	STDPNaN
AR	92.20±1.86	51.51±3.80	39.85±4.48	**92.27±1.11**	42.90±4.81
Cleve	**75.68±5.88**	68.39±8.23	68.50±9.08	71.23±9.30	68.45±7.36
COIL20	**92.20±1.30**	80.90±2.00	76.98±3.35	90.10±1.76	79.32±2.72
Heart	**74.55±3.81**	72.98±3.82	74.88±3.61	70.36±7.52	69.56±4.49
Isolet	**72.36±1.81**	69.46±3.53	70.01±2.38	71.78±1.88	67.87±2.93
MINIST2k	**85.13±0.74**	73.60±1.40	71.92±1.96	85.08±1.55	72.76±1.49
ORL	**91.44±2.01**	55.19±8.16	42.59±5.31	90.00±2.11	46.39±5.99
Palm	**92.76±0.65**	62.21±1.99	51.82±2.66	91.86±1.18	57.07±4.30
Solar	**75.92±7.09**	72.82±3.86	77.63±2.91	57.87±11.47	72.14±5.85
Sonar	**75.75±2.81**	65.63±9.00	66.36±4.95	73.91±4.93	65.87±4.30
UPS	**87.64±1.64**	80.19±1.97	79.36±2.10	87.25±2.00	79.72±2.35
YaleB	**90.73±2.07**	61.70±2.63	52.22±2.67	90.32±2.40	54.62±3.42
BUPA	**67.41±4.72**	63.7±3.6	61.22±3.34	66.95±3.12	62.71±3.86
FERET32	**91.73±1.99**	48.36±7.33	34.48±3.92	91.63±1.84	45.53±5.45
MSRA25	**90.39±0.98**	86.1±1.13	85.25±2.08	90.14±0.73	84.72±1.5
Yeast	93±0.31	73.85±5.18	76.74±2.89	**93.05±0.46**	75.42±4.36
autouni	**73.79±4.17**	62.52±8.84	59.58±6.77	70.26±8.22	61.69±9.58
Ecoli	89.66±1.37	79.61±4.8	80.22±6.74	**89.69±1.93**	84.75±4.29
pendigits	90.14±0.63	90.91±1.4	90.93±0.85	89.7±1.1	**91.08±0.69**
uspst	**87.77±1.21**	80.91±2.21	79.68±1.91	87.19±1.32	80.15±1.29
WSR-test	N/A	+	+	~	+
Ave.ACC	**84.99**	69.83	66.95	83.19	67.99
Ave.std	2.2	3.94	3.41	3.06	3.79

（2）在 AR、COIL20、ORL、Palm、YaleB、 FERET32、MSRA25 这 7 个图像数据集上，提出的算法 EBSA 的分类准确率分别为 **96.23%、92.61%、95.58%、95.46%、91.93%、97.75%、90.63%和 90.9%**， F 分数分别为 92.20%、92.20%、**91.44%、92.76%、90.73%、91.73、90.39%和** 90.14%，分类性能均超过了 90%，验证了提出的算法 EBSA

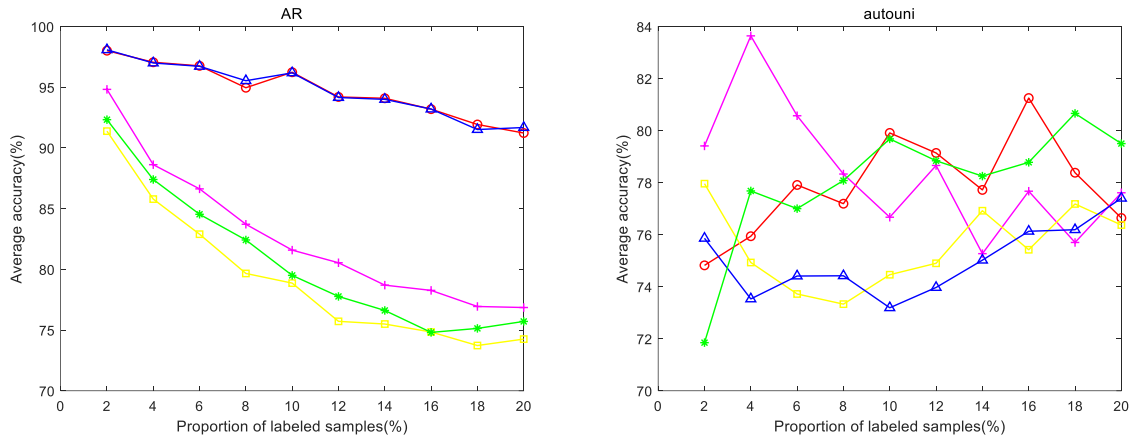
的良好的分类性能，显示出 EBSA 处理高维图像数据的能力；

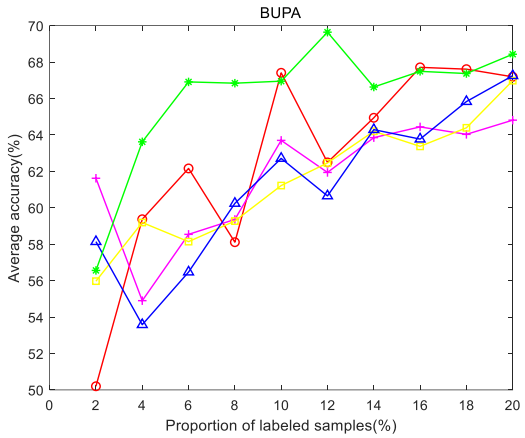
（3）EBSA 算法在 20 个数据集上的平均分类准确率为 87.39%，分别比对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF 高出 8.03%、9.02%、0.89%和 9.05%，平均 F 分数为 84.99%，分别比对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF 高出 15.16%、18.04%、1.8%和 17%，证明了 EBSA 算法的良好的分类性能。EBSA 算法在 20 个数据集上的平均分类标准差为 1.32，分别比对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF 低 0.98、0.78、0.47 和 1.02，平均 F 分数标准差为 2.2，分别比对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF 低 1.74、1.21、0.86 和 1.59，证明了 EBSA 算法的鲁棒性。

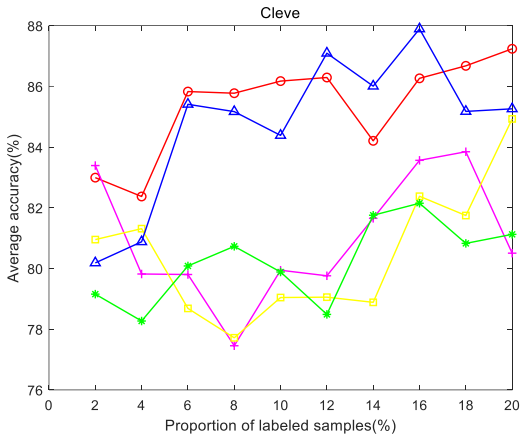
（4）由统计检验结果可知，EBSA 的分类性能显著优于对比算法 SETRED、STDP-CEW 和 STDPNaN，与 STDPNF 分类性能相差无几；20 个数据集类型多样，有图像数据集，大型数据集，小型数据集，实验结果证明了 EBSA 算法具有广泛适用能力。

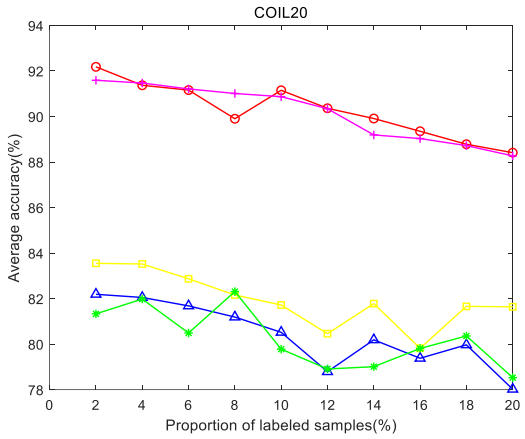
3.6 有标签样本比例对算法分类性能的影响

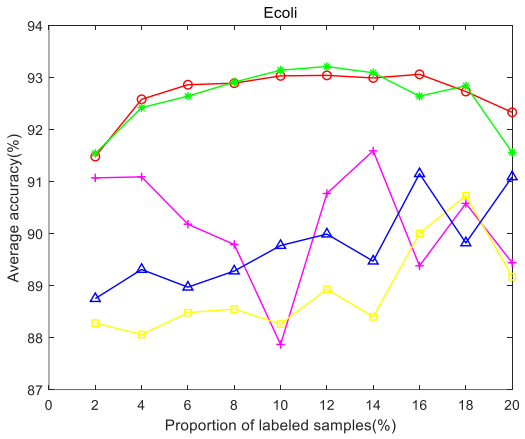
为验证有标签样本比例对对比算法分类性能的影响，针对少数有标签数据，进行一个梯度测试实验，随机选取有样本标签比例为 2%~20%，步进为 2%，在 20 个数据集上实验。实验结果如图 3.4 和表 3.6 所示。表 3.6 的分类性能为 5 个算法在 2%~20%的有标签比例上的分类性能的均值和标准差。

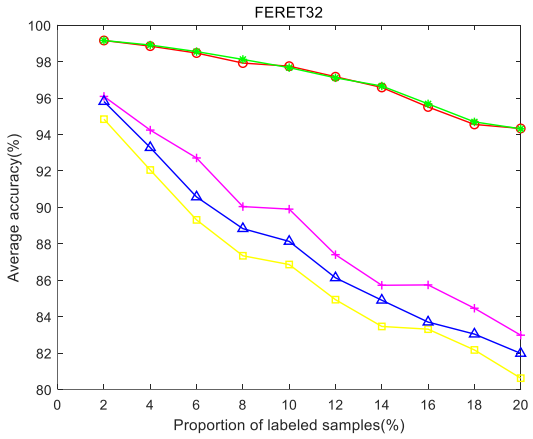


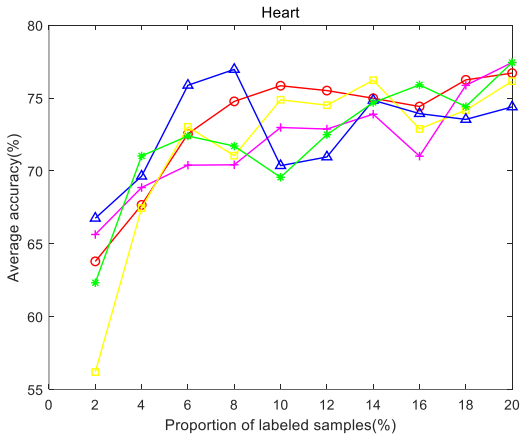

BUPA
Average accuracy(%)
Proportion of labeled samples(%)

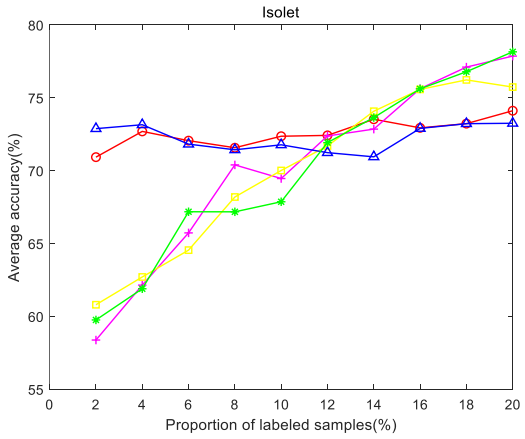

Cleve
Average accuracy(%)
Proportion of labeled samples(%)

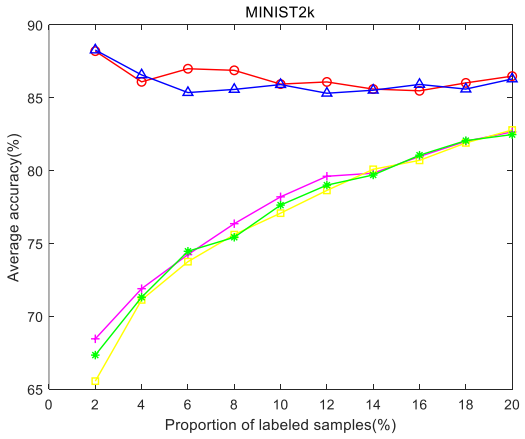

COIL20
Average accuracy(%)
Proportion of labeled samples(%)

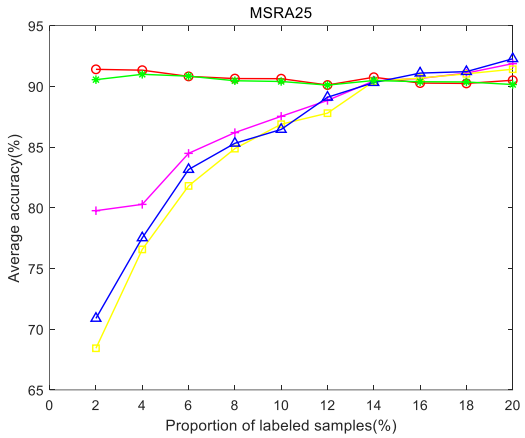

Ecoli
Average accuracy(%)
Proportion of labeled samples(%)

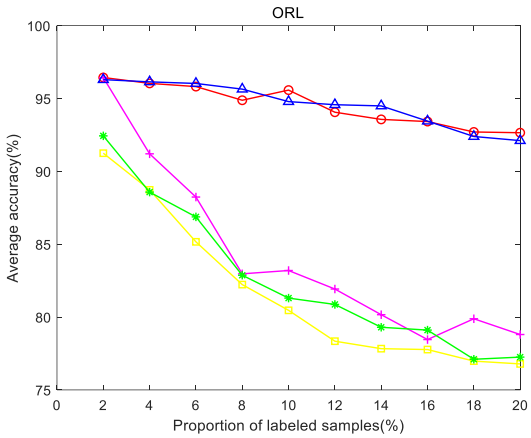

FERET32
Average accuracy(%)
Proportion of labeled samples(%)

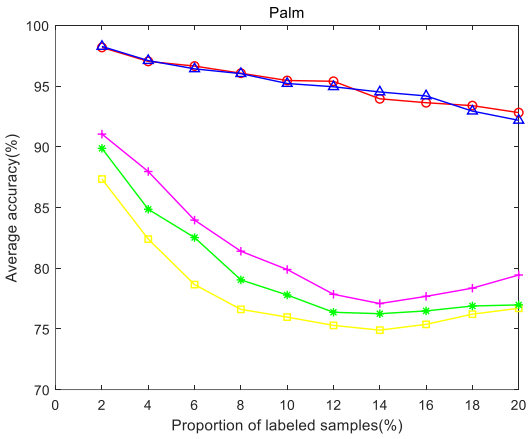

Heart
Average accuracy(%)
Proportion of labeled samples(%)

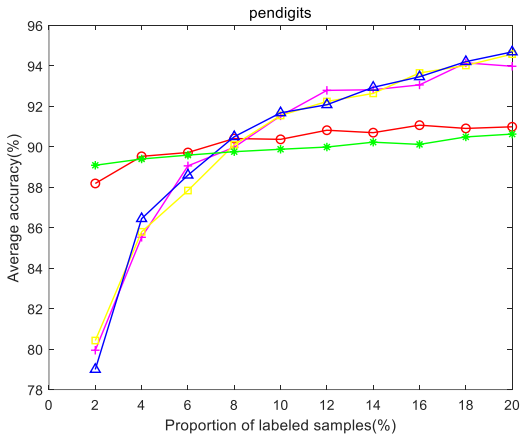

Isolet
Average accuracy(%)
Proportion of labeled samples(%)

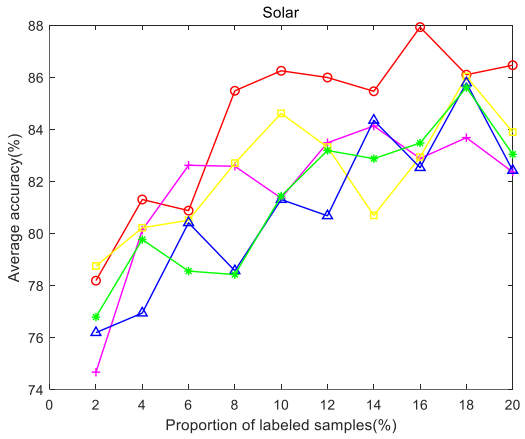

MINIST2k
Average accuracy(%)
Proportion of labeled samples(%)

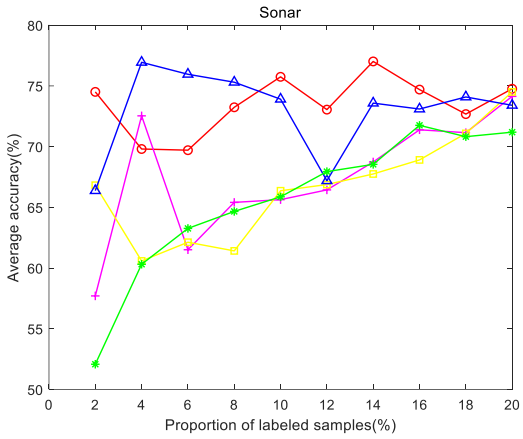

MSRA25
Average accuracy(%)
Proportion of labeled samples(%)

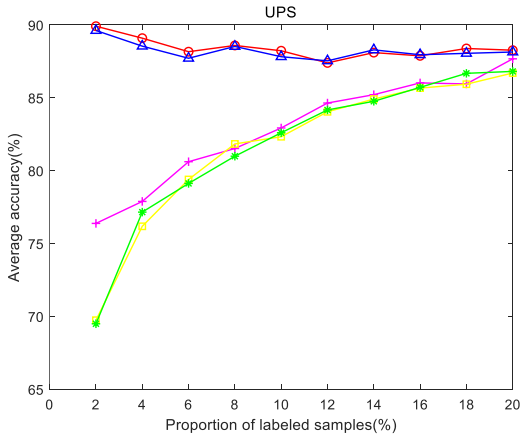

ORL
Average accuracy(%)
Proportion of labeled samples(%)

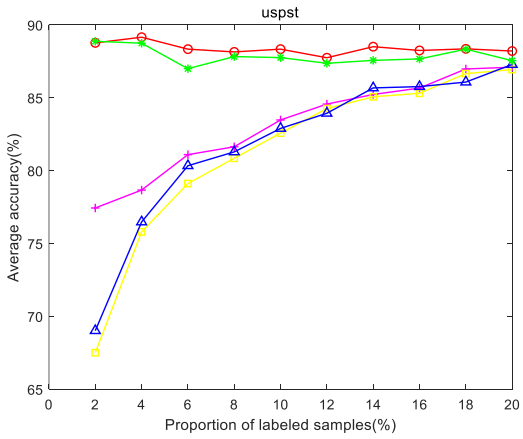

Palm
Average accuracy(%)
Proportion of labeled samples(%)


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


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


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


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


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

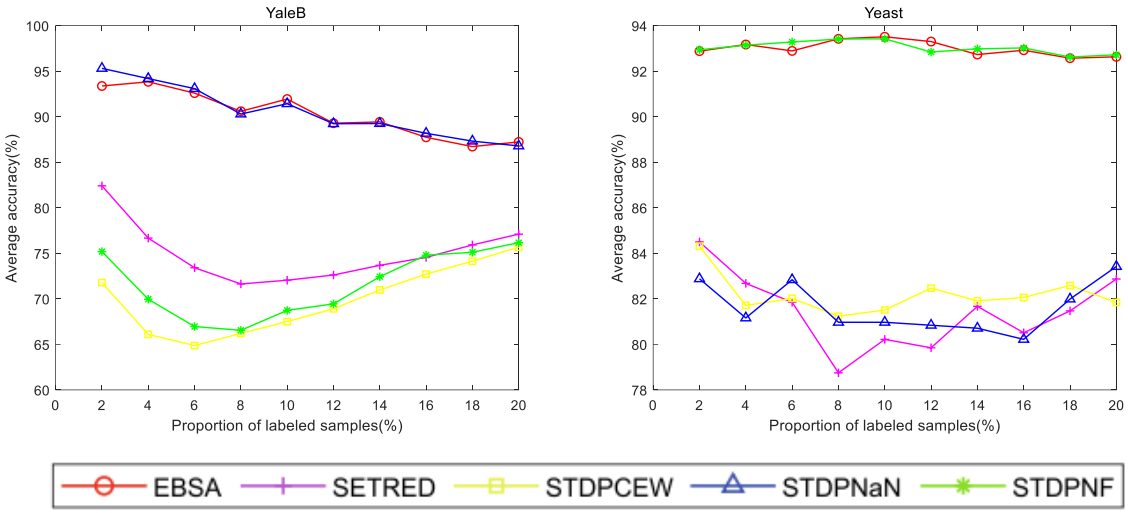


图 3.4 有标签样本比例对算法分类性能的影响

从图 3.4 和表 3.6 可得到如下结论：

（1）由图 3.4 可以看出，随着有标签样本比例增加，提出的算法 EBSA 在 AR、COIL20、MINIST2k、ORL、Palm、UPS、YaleB、FERET32、Yeast、uspst 和 Ecoli 这 11 个数据集上的分类性能始终优于 SETRED、STDP-CEW 和 STDPNaN 算法，且优势十分明显，体现了 EBSA 算法良好的分类性能。

（2）在 AR、COIL20、ORL、Palm 和 YaleB 数据集上，随着有标签比例的增加，提出的算法 EBSA 的分类性能在逐渐下降，这是因为提出的算法 EBSA 是以数据类簇为分类依据，随着有标签比例的增加，可能会给球簇划分与编辑造成一些干扰，从而影响分类性能；在 AR、ORL 和 Palm 数据集上，SETRED、STDP-CEW 和 STDPNaN 算法 3 个对比算法的分类性能都随有标签比例增加而下降，且下降的趋势相比于提出的算法 EBSA 更加明显；

（3）EBSA 算法的平均准确率为 86.19%，相比于对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF，高出 6.08%、7.6%、0.19%和 7.25%，验证了 EBSA 算法良好的分类性能。 EBSA 的平均标准差为 1.71，分别比 SETRED、STDP-CEW 和 STDPNaN 低 2.12，2.55 和 2.57，体现了 EBSA 算法分类性能的稳定性；

（4）从表 3.6 看出提出的算法 EBSA 在 Cleve、COIL20、Heart 、Isolet、MINIST2k2k、Palm、Solar 、Sonar 、UPS、MSRA25、uspst 和 Ecoli 这 11 个数据集上的分类性能高于 SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法，这 11 个数据集有图像数据集，大型数据集和小型数据集，体现出 EBSA 算法的良好的泛化能力。

表 3.6 有标签样本比例对算法分类性能的影响

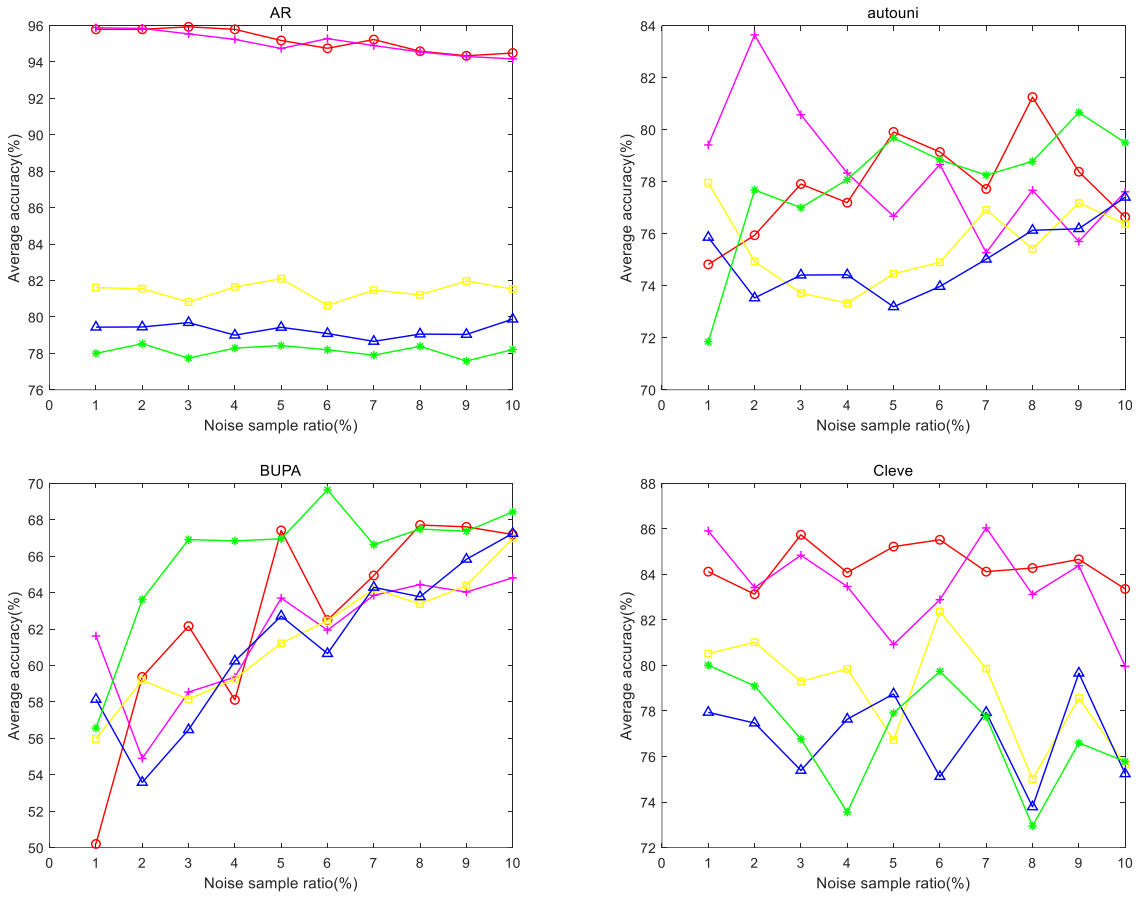
Accuracy	EBSA	SETRED	STDPCEW	STDPNF	STDPNaN
AR	94.77±2.25	82.67±5.84	79.26±5.82	**94.81±2.26**	80.62±5.92
Cleve	**85.38±1.63**	80.97±2.08	80.47±2.2	84.74±2.45	80.25±1.32
COIL20	**92.45±0.51**	84.98±3.54	82.36±6.19	91.89±1.1	82.87±6.02
Heart	**73.25±4.23**	71.94±3.44	71.65±6.02	72.72±3.18	72.19±4.2
Isolet	**72.59±0.94**	70.19±6.44	69.96±5.71	72.26±0.91	70±6.26
MINIST2k	**86.38±0.81**	77.43±4.65	76.73±5.38	86.03±0.89	77.05±4.93
ORL	94.51±1.42	84.13±5.96	81.55±5.19	**94.59±1.52**	82.57±5.14
Palm	**95.26±1.77**	81.47±4.75	77.95±3.97	95.18±1.85	79.71±4.6
Solar	**84.4±3.14**	81.79±2.76	82.36±2.26	80.91±3.06	81.31±2.81
Sonar	**73.52±2.37**	67.46±5.17	66.64±4.38	72.98±3.49	65.64±6.03
UPS	**88.40±0.69**	82.89±3.73	81.67±5.31	88.22±0.59	81.75±5.38
YaleB	**90.27±2.59**	75.00±3.22	69.89±3.71	90.51±2.92	71.53±3.65
BUPA	62.72±5.63	61.72±3.22	61.52±3.37	**66.04±3.66**	61.29±4.31
FERET32	97.03±1.74	88.93±4.4	86.49±4.53	**97.09±1.73**	87.64±4.56
MSRA25	**90.67±0.43**	87.11±4.39	84.99±7.5	90.47±0.27	85.73±6.9
Yeast	93.00±0.33	81.44±1.68	82.17±0.85	**93.04±0.27**	81.6±1.1
autouni	77.89±1.9	78.35±2.46	75.52±1.54	**78.03±2.41**	75.01±1.35
Ecoli	**92.7±0.49**	90.18±1.1	88.89±0.86	92.56±0.58	89.76±0.81
pendigits	90.27±0.9	90.29±4.49	90.28±4.45	89.92±0.48	**90.36±4.74**
uspst	**88.38±0.38**	83.19±3.37	81.41±6.02	87.87±0.6	81.88±5.57
WSR-test	N/A	+	+	~	+
Ave.ACC	86.19	80.11	78.59	86	78.94
Ave.std	1.71	3.83	4.26	1.71	4.28

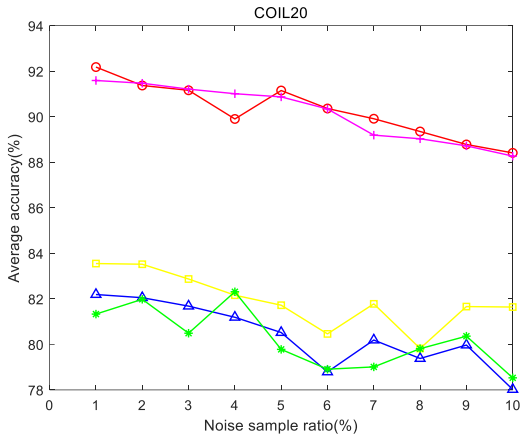
（5）随着有标签比例的增加，提出的算法 EBSA 在 AR、COIL20、Isolet、MINIST2k2k、ORL、Palm、UPS、YaleB、FERET32、Yeast、MSRA25、pendigits 、uspst 和 Ecoli 这 14 个数据集上的分类性能都比较平稳。在不同的有标签比例条件下，EBSA 算法的分类性能曲线比较平滑，这显示了 EBSA 算法的具有很好的鲁棒性。在表 3.6 中，

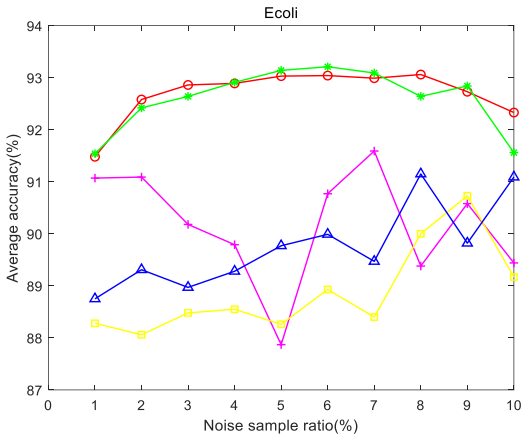
提出的算法 EBSA 在这 14 个数据集上的标准差均小于 SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法，与图 3.4 中得出的结论一致。

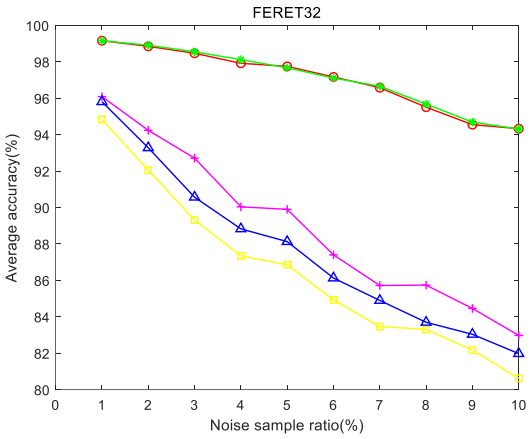
3.7 噪声比例实验分析

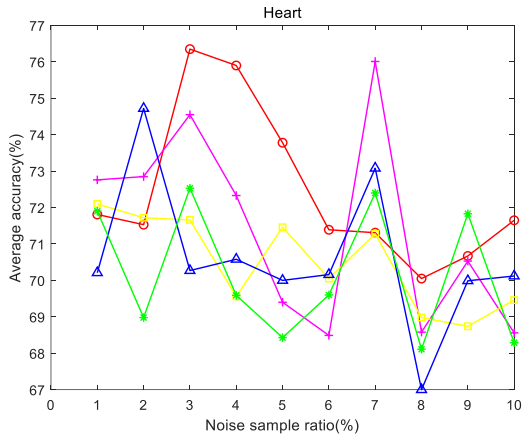
EBSA 算法在迭代训练的过程中使用球簇划分对数据进行编辑，因此具有一定的去噪能力，能够有效地纠正数据集中标错的数据。为了验证 EBSA 的去噪能力，进行了噪声实验。因为本文主要研究在只有少量有标签的数据集上的算法性能，因此在有标签比例为 10%的数据集上进行噪声实验，因为有标签样本比例较少，噪声数据不宜过多，否则会大大影响实验效果，因此从有标签样本中随机选取[1%,10%]的数据，步进为 1%，并赋予其错误标签，然后进行实验。选择准确率作为分类性能评价指标，每个数据集上各算法均进行了 50 次实验，计算各次实验结果的均值和标准差作为实验结果。实验结果如图 3.5 和表 3.7 所示：

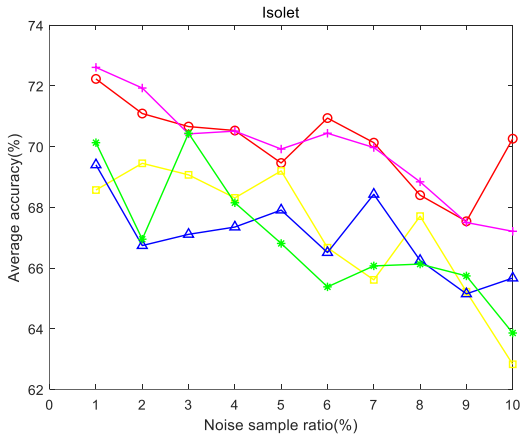


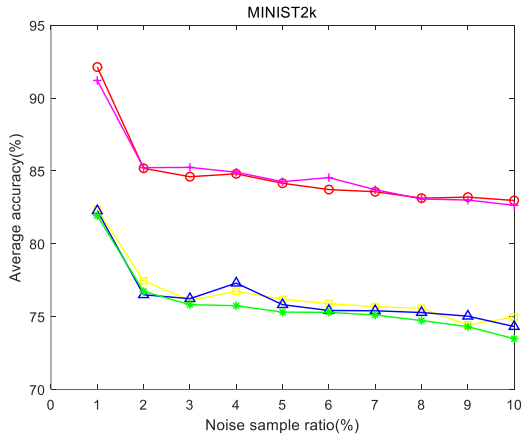

COIL20
Average accuracy(%)
Noise sample ratio(%)

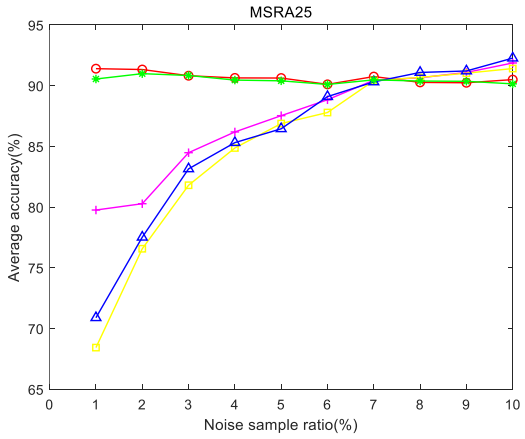

Ecoli
Average accuracy(%)
Noise sample ratio(%)

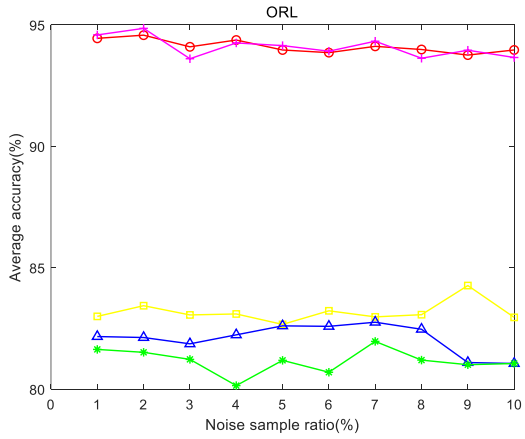

FERET32
Average accuracy(%)
Noise sample ratio(%)

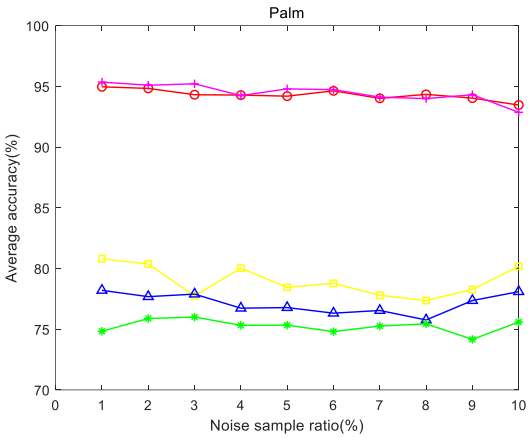

Heart
Average accuracy(%)
Noise sample ratio(%)

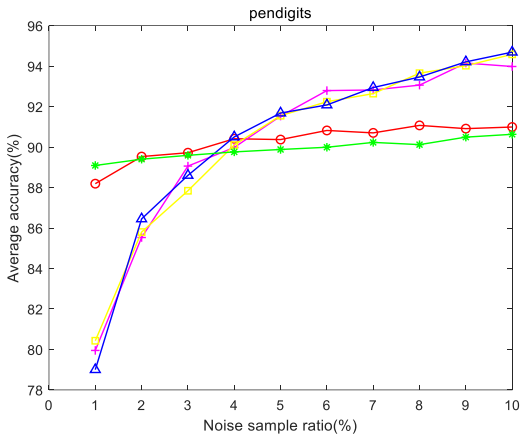

Isolet
Average accuracy(%)
Noise sample ratio(%)

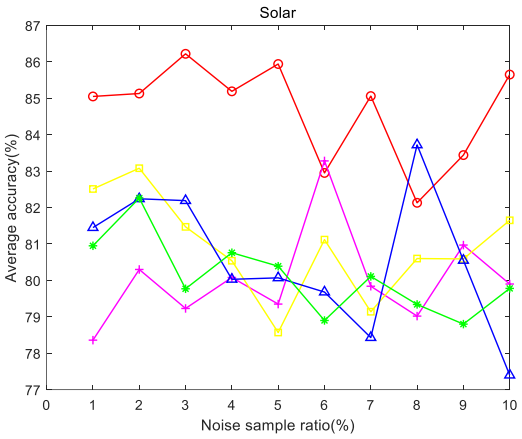

MINIST2k
Average accuracy(%)
Noise sample ratio(%)

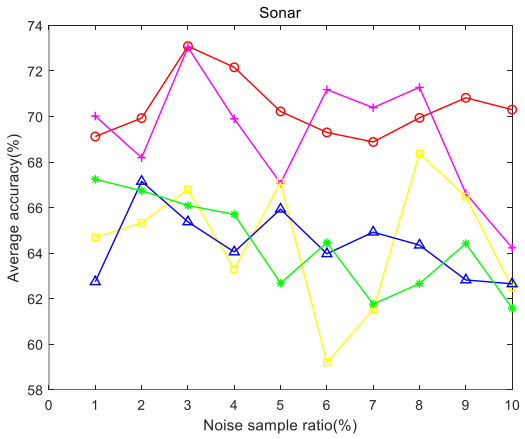

MSRA25
Average accuracy(%)
Noise sample ratio(%)

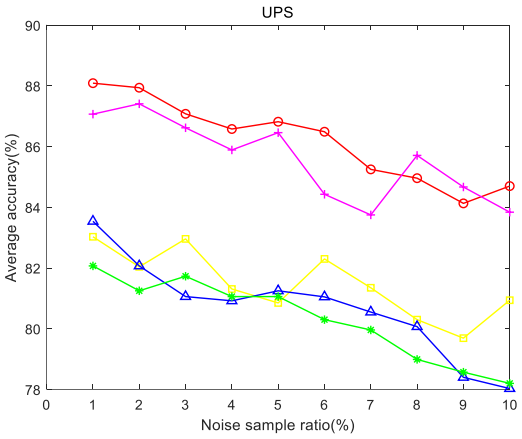

ORL
Average accuracy(%)
Noise sample ratio(%)

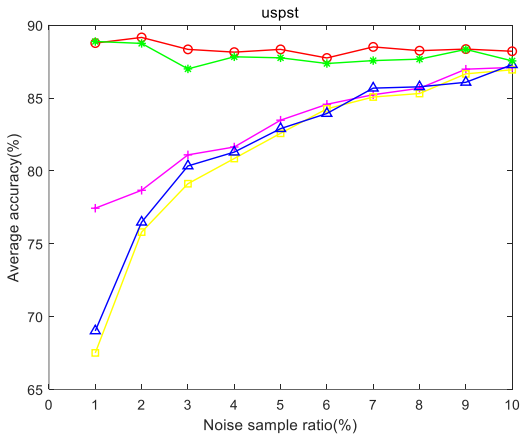

Palm
Average accuracy(%)
Noise sample ratio(%)

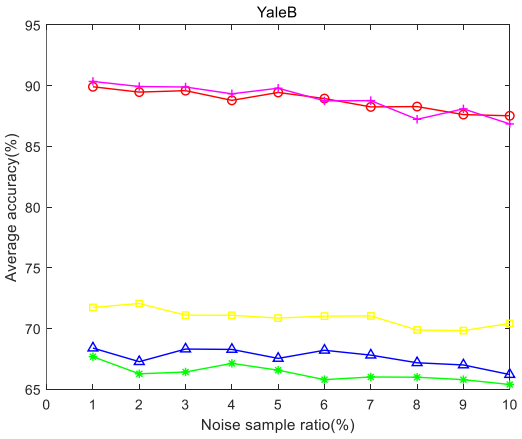

pendigits
Average accuracy(%)
Noise sample ratio(%)


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


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


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


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


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

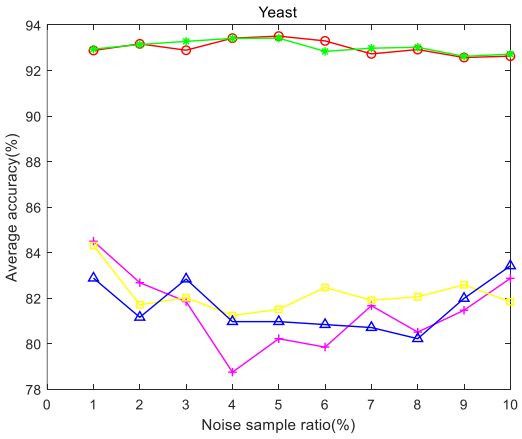

Yeast
Average accuracy(%)
Noise sample ratio(%)

图 3.5 噪声比例对各算法的影响

表 3.7 噪声比例对各算法的影响

Accuracy	EBSA	SETRED	STDPCEW	STDPNF	STDPNaN
AR	**95.19±0.61**	81.46±0.46	78.13±0.31	95.04±0.61	79.27±0.37
COIL20	**90.26±1.21**	81.92±1.2	80.25±1.3	90.17±1.25	80.4±1.4
Cleve	**84.42±0.87**	78.89±2.4	77.02±2.41	83.5±1.96	76.89±1.88
Heart	**72.44±2.16**	70.5±1.27	70.17±1.8	71.41±2.69	70.61±2.04
Isolet	**70.12±1.36**	67.26±2.15	66.97±2.07	69.94±1.72	67.05±1.28
MINIST2k	84.75±2.7	76.54±2.22	75.85±2.31	**84.78±2.45**	76.36±2.23
ORL	94.06±0.33	83.18±0.43	81.17±0.5	**94.10±0.42**	82.1±0.6
Palm	94.3±0.43	78.98±1.25	75.26±0.55	**94.47±0.74**	77.14±0.83
Solar	**84.68±1.36**	80.93±1.38	80.11±1.04	80.03±1.35	80.58±1.89
Sonar	**70.38±1.34**	64.53±2.87	64.33±2.08	69.19±2.63	64.4±1.48
UPS	**86.2±1.37**	81.48±1.1	80.32±1.35	85.59±1.34	80.69±1.61
YaleB	88.78±0.84	70.9±0.71	66.3±0.68	**88.9±1.19**	67.62±0.71
BUPA	63.35±2.25	60.12±1.17	60.35±1.48	**65.13±1.45**	59.57±1.1
FERET32	97.08±0.35	89.8±0.21	87.01±0.25	**97.22±0.32**	88.09±0.28
MSRA25	**88.46±0.9**	85.39±1.67	84.7±1.24	88.35±1.34	85±1.53
Yeast	**87.24±3.12**	76.55±1.89	76.17±2.01	87.15±3.47	76.1±2.64
autouni	76.15±1.16	75.6±1.31	72.99±1.93	**77.07±2.15**	72.41±1.09
Ecoli	89.04±1.55	87.11±1.16	86.75±1.72	**89.14±1.82**	86.3±1.35
pendigits	**89.68±0.48**	89.98±0.77	89.71±1.11	88.63±0.74	89.79±0.93
uspst	**86.15±1.06**	81.5±1.42	80.88±1.1	85.51±1.16	80.8±1.22
WSR-test	N/A	+	+	~	+
Ave.ACC	84.64	78.13	76.72	84.27	77.06
Ave.std	1.27	1.35	1.36	1.54	1.32

从图 3.5 和表 3.7 中可以得出以下结论：

（1）随着噪声样本比例增加，提出的算法 EBSA 的分类性能在 AR、COIL20、ORL、

Palm 和 FERET32 数据集上的分类性能均高于 90%，比性能最低的对比算法 STDP-CEW 提升了 17.06%，10.01%，12.89%、19.04%和 10.07%，在 Cleve、MINIST2k、Solar、UPS、YaleB、MSRA25、Yeast 、Ecoli 和 uspst 数据集上的分类性能均高于 80%，比性能最低的对比算法 STDP-CEW 提升了 7.4%、8.9%、4.57%、5.88%、22.48%、3.76%、11.07%、2.29%和 5.27%，表 3.7 中的均值与图 3.5 中结论一致，由此可以证明提出的算法 EBSA 的良好的去噪声的能力；

（2）EBSA 算法的平均分类准确率为 84.64%，相比于对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF，性能分别提升了 6.51%、7.92%、0.37%和 7.58% ，由此可以看出 EBSA 算法具有数据编辑效果，可以降低噪声样本对分类性能的影响。

（3）由图 3.5 可以看出，EBSA 算法在 AR、COIL20、ORL、Palm 、UPS、YaleB 和 FERET32 数据集上分类性能没有受到噪声比例的增加影响，随着噪声比例的增加，分类性能曲线在水平方向波动，没有较大的抖动，验证了 EBSA 算法良好的数据编辑能力。

（4）提出的算法 EBSA 的分类性能在 AR、COIL20、Cleve、Isolet、MINIST2k、ORL、Palm、Solar、Sonar、UPS 和 YaleB 这 11 个数据集上的分类性能远优于 SETRED、STDP-CEW 和 STDPNaN 算法，略优于使用局部过滤器的 STDPNF 算法，证明 EBSA 算法的球簇划分编辑方法去噪声的性能优良，表 3.7 的实验数据与图 3.5 一致，证实了图 3.5 结论的真实有效性。

3.8 迭代实验分析

为证明 EBSA 算法实验的真实性，以 UPS 数据集为例，将 EBSA 算法迭代过程细节进行描述，在 UPS 数据集上，EBSA 算法进行三次迭代后达到收敛。

首先，随机抽取 10%的样本赋予样本标签，初始有标签样本数量为 200 个。第一次迭代，进行了数据编辑和高置信度样本的选取，得到了 353 个高置信度样本，第二次迭代，得到了 387 个高置信度样本，第三次迭代得到了 388 个样本后，达到了实验收敛条件，即高置信度样本集无新增数据，使用得到的高置信度样本进行训练获得了一个新的分类器。

3.9 算法运行时间分析

EBSA 算法的具有输入样本个数的线性时间复杂度，与对比算法相比，时间复杂度大幅降低。表 3.8 中，括号中的数字表示在五个算法中，该算法的运行时间在当前数据集下进行实验的排序（从时间最短到最长进行排序）。在 20 个数据集上进行实验，实验结果如表 3.8 所示。

令 n 表示输入样本的个数，d 表示维度，t 表示迭代次数，k 表示球簇数。使用决策树基分类器的时间复杂度为 $O(n\log n)$。EBSA 算法计算球簇簇心的时间复杂度为 $O(n)$，计算球簇半径的时间复杂度为 $O(kn)$，计算各个球簇中的样本与该球簇中心的距离的时间复杂度为 $O(kn)$，计算球簇中心距离的时间复杂度为 $O(k^2)$，因为 $k \ll n$，故 EBSA 算法的整体时间复杂度为 $O(t(2kn+n\log n+n+k^2))$。

从表 3.8 中可以得出以下结论：

（1）在 20 个数据集上，提出的算法 EBSA 的运行时间均大幅低于对比算法 SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法；表 3.8 中的数据显示 EBSA 的运行时间最短，其次是 STDPNaN 算法，之后是 STDPNF 算法，SETRED 和 STDP-CEW 算法运行时间最长。这是因为 STDP-CEW 算法计算密度峰值的时间复杂度为 $O(n^2)$，构造相关邻接图的时间复杂度为 $O(tdn^3)$，故 STDP-CEW 算法的整体时间复杂度为 $O(tdn^3)$。STDPNaN 算法和 STDPNF 算法主要在于使用 DPC 发现空间结构和寻找自然近邻 NaN，其时间复杂度分别为 $O(n^2)$ 和 $O(n\log n)$，故 STDPNaN 的整体时间复杂度为 $O(n^2)$，ENaNE 在 STDPNF 中的运行时间依赖于 ENaNE 中自训练算法的运行时间 t_e，故 STDPNF 的整体时间复杂度为 $\max\{O(n^2),t_e\}$。

（2） 在 20 个数据集上，EBSA 算法的运行时间明显小于其他四个对比算法，由平均时间可以得到，EBSA 算法的运行时间是 SETRED、STDP-CEW、STDPNaN 和 STDPNF 算法的 1.17%、0.12%、1.83%和 2.59%，运行速率得到了极大的提升。EBSA 算法的运行时间排序全都为 1，证明了 EBSA 算法在时间上领先于其他对比算法，验证了理论分析的正确性。

表 3.8 各算法在 20 个数据集上的运行时间

time	EBSA	SETRED	STDPCEW	STDPNF	STDPNaN
AR	**22.53(1)**	705.05(5)	320.15(4)	130.03(3)	52.29(2)
Cleve	**0.02(1)**	1.27(4)	0.55(3)	0.12(2)	0.12(2)
COIL20	**3.43(1)**	74.61(4)	145.03(5)	23.95(3)	17.99(2)
Heart	**0.01(1)**	1.24(4)	0.32(5)	0.09(2)	0.16(3)
Isolet	**1.83(1)**	26.86(4)	77.74(5)	53.75(2)	11.21(2)
MINIST2k	**2.87(1)**	380.67(4)	1965.18(5)	332.93(3)	161.29(2)
ORL	**0.95(1)**	28.24(5)	6.07(4)	6.60(3)	1.91(2)
Palm	**8.11(1)**	253.86(4)	281.93(5)	32.40(2)	33.15(3)
Solar	**0.03(1)**	2.14(4)	0.43(3)	0.20(2)	0.20(2)
Sonar	**0.02(1)**	1.29(5)	0.27(4)	0.23(3)	0.14(2)
UPS	**0.42(1)**	45.67(4)	208.04(5)	39.73(3)	18.18(2)
YaleB	**9.86(1)**	312.61(4)	446.26(5)	164.95(3)	57.53(2)
BUPA	**0.03(1)**	1.65(5)	0.62(4)	0.11(2)	0.26(3)
FERET32	**11.14(1)**	472.56(5)	111.73(4)	76.92(3)	20.88(2)
MSRA25	**1.03(1)**	35.15(4)	91.09(5)	9.37(2)	15.42(3)
Yeast	**4.2(1)**	55.19(5)	25.09(4)	11.44(3)	8(2)
autouni	**0.03(1)**	1.67(5)	0.23(4)	0.13(3)	0.1(2)
Ecoli	**0.08(1)**	2.96(5)	0.98(4)	0.9(3)	0.25(2)
pendigits	**0.24(1)**	143.37(4)	1514.17(5)	9.45(2)	91.85(3)
uspst	**0.44(1)**	37.91(3)	201.14(5)	40.66(4)	18.55(2)
Ave.time	3.36	129.2	269.85	52.63	26.8
Ave.rank	1	4.3	4.5	2.6	2.3

（3）因为 Cleve、Heart、Solar、Sonar、BUPA、autouni 和 Ecoli 数据集相对于其他 13 个数据集来说比较小，因此提出的算法 EBSA 在这七个数据集上的运行时间均不超过 0.05s，与其他四个对比算法相比，速度大大提升，在 Cleve、Solar、Sonar、autouni 和 Ecoli 数据集上是仅次于 EBSA 算法的 STDPNaN 算法运行时间的 16.67%、15%、14.29%、30%和 32%，在 Heart 和 BUPA 数据集上是仅次于 EBSA 算法的 STDPNF 算法

运行时间的 11.11%，27%；在图像数据集 MINIST2k、UPS 上，EBSA 的运行时间是仅次于 EBSA 算法的 STDPNaN 算法运行时间的 1.78%和 2.31%；在大型数据集 USPS 和 PIE 上，EBSA 的运行时间是仅次于 EBSA 算法的 STDPNF 算法运行时间的 0.4%和 2.79%，由此可知，在大型数据集上，EBSA 算法快速的特性更加明显；整体而言，提出的算法 EBSA 的具有输入样本个数的线性时间复杂度，时间复杂度最低。实验结果与理论分析一致，验证了 EBSA 具有快速的特性。

3.10 不同基分类器的实验

为了研究不同基础分类器对所提出的高置信度样本选择和编辑的影响，在随机森林、KNN（K=1）、SVM 和决策树上进行了实验。让 DT 表示决定树，RF 表示随机森林。所有实验进行了 50 次，记录了平均值和标准偏差并列出了每个分类器的运行时间。实验结果如表 3.9 和 3.10 所示。

表 3.9 在各分类器上的分类性能

Accuracy	SG+KNN	SG+RF	SG+SVM	SG+DT
AR	99.12±0.02	99.1±0.06	99.09±0.09	96.23±0.61
Cleve	75.71±4.19	87.29±1.65	73.21±6.76	86.17±2.13
COIL20	95.94±0.05	95.95±0.02	95.97±0.01	92.61±1.07
Heart	70.79±7.57	78.9±1.05	57.79±6.7	74.55±3.81
Isolet	76.31±0.07	76.31±0.11	76.32±0.14	72.36±1.81
MINIST2k	91.37±0.01	91.37±0.01	91.37±0.01	85.94±0.62
ORL	97.01±0.06	97.25±0.04	97.31±0.05	95.58±0.66
Palm	99.06±0.02	99.05±0.02	99.04±0.14	95.46±0.47
Solar	83.13±3.13	87.47±2.11	83.22±0.45	86.25±1.54
Sonar	77.94±0.36	78.03±0.17	53.37±0.01	75.75±2.81
UPS	92.04±0.02	92.64±0.03	92.3±0.19	88.23±1.39
YaleB	96.57±0.01	97.47±0.21	97.46±0.17	91.93±1.68
BUPA	65.94±3.12	71.73±1.76	42.58±1.67	67.41±4.72
FERET32	98.97±0.16	99.27±0.12	98.99±0.13	97.75±0.5

续表 3.9

Accuracy	SG+KNN	SG+RF	SG+SVM	SG+DT
MSRA25	92.87±0.01	92.86±0.01	92.86±0.01	90.63±0.87
Yeast	92.51±0.04	93.23±0.08	93.48±0.06	93.51±0.02
autouni	70.12±4.56	83±1.71	58.92±4.8	79.91±1.36
Ecoli	91.15±0.15	92.4±0.64	92.31±0.51	93.03±0.18
pendigits	92.15±0.07	92.13±0.06	89.03±0.46	90.37±0.57
uspst	92.65±0.02	92.64±0.01	92.26±0.19	88.34±1.03
Ave.ACC	88.3	90.43	84.75	87.39
Ave.std	1.08	0.45	1.03	1.32

表 3.10 在各分类器上的运行时间

runtime	SG+KNN	SG+RF	SG+SVM	SG+DT
AR	23.45	26.25	691.56	22.53
Cleve	0.02	0.49	2.05	0.02
COIL20	2.58	6.76	116.84	3.43
Heart	0.03	0.41	0.44	0.01
Isolet	1.45	4.16	0.96	1.83
MINIST2k	12.58	9.78	109.21	2.87
ORL	0.2	3.46	28.98	0.95
Palm	1.09	21.07	314.27	8.11
Solar	0.04	0.57	0.16	0.03
Sonar	0.01	0.27	0.03	0.02
UPS	1.07	4.61	3.72	0.42
YaleB	6.11	15.08	255.71	9.86
BUPA	0.01	0.41	0.14	0.03
FERET32	12.28	17.19	943.54	11.14
MSRA25	0.76	4.11	3.71	1.03
Yeast	3.34	6.27	19.13	4.2
autouni	0.03	0.4	5.45	0.03

续表 3.10

runtime	SG+KNN	SG+RF	SG+SVM	SG+DT
Ecoli	0.03	0.72	0.33	0.08
pendigits	0.15	4.26	38.83	0.24
uspst	1.12	4.58	3.45	0.44
Ave.time	10.33	11.59	825.92	7.86

从表 3.9 和表 3.10 中，可以得出以下结论：

（1） SG+RF 获得了最高的平均分类精度，SG+KNN 仅次于 SG+RF。SG+DT 的分类精度与 KNN 相似，但略低于随机林。SG+SVM 的分类精度最低。

（2） SG+DT 的平均运行时间小于 SG+KNN，SG+SVM 的运行时间最长。由于随机森林集成了多个决策树，SG+RF 的运行时间远高于 SG+DT 和 SG+KNN。

（3） 总之，SG+DT 比其他树算法运行得更快。尽管 SG+DT 的分类精度较低但运行时间远小于 SG+RF。

4 块估计近邻编辑的 Self-training 算法（MDSF）

现有的自训练算法 SETRED、STDPCEW、STDPNaN、STDP、STDPDE、STDPNF 和 SNNRCE 在选择高置信度样本时，均使用欧氏距离来计算样本间距离，但欧式距离不适用于高维数据集上，曼哈顿距离比欧式距离更适用于高维数据，但曼哈顿距离计算的距离不是最短路径，无法在实际应用中起到更好的效果。基于块的不相似性度量在高维数据集上有良好的表现能力，且充分考虑到样本距离度量的差异性，因此基于块的不相似性度量计算样本之间距离，提出块估计近邻编辑 Self-training 算法。

4.1 定义

定义 1 块相似邻居

当样本 x_i 和 x_j 的不相似性小于 0.5，则认为样本 x_i 和 x_j 相似。当样本 x_i 和 x_j 相似，且样本 x_j 和 x_i 相似，则样本 x_i 和 x_j 称为块相似邻居。$MN_\alpha(x_i)$ 表示包含 α 个样本的样本 x_i 的块相似邻居集合。

定义 2 块邻数，离群点和离群集

$MNb(X)=\{g_1,g_2,\cdots,g_n\}$ 表示块相似邻居集合，当样本 x_i 在其他样本的块相似邻居集合中出现的次数为 g_i，样本 x_i 的块邻数 $MNb(x_i)=g_i$。

当样本 x_i 的块邻数 $MNb(x_i)=0$，样本 x_i 称为离群点。离群点形成的集合称为离群集，离群集 $LQ=\{x_i\,|MNb(x_i)=0\}$。

定义 3 块相似近邻

样本 x_i 的块相似近邻集合 $MDN(x_i)$ 为：

$$x_j\in MDN(x_i)\Leftrightarrow\left(x_i\in MN_\alpha(x_j)\right)\&\&\left(x_j\in MN_\alpha(x_i)\right) \tag{4.1.1}$$

定义 4 块稳定结构

当每个样本至少有一个块相似邻居的样本点时，数据集达到块稳定状态，这种状态称为块稳定结构。

$$(\forall x_i)(\exists x_j)(\alpha\in\lambda)\vee(x_i\neq x_j)\rightarrow\left(x_i\in MN_\alpha(x_j)\right)\vee\left(x_j\in MN_\alpha(x_i)\right) \tag{4.1.2}$$

公式(4.1.2)中，循环次数 $\alpha \in [1,\lambda]$， $\lambda \le n$， $\lambda = \max\left|\left\{x_j \middle| x_j \in MDN(x_i)\right\}\right|$。

定义 5 样本的局部密度和峰值

样本 x_i 的局部密度计算如下：

$$\rho_i = \sum_{j \ne i} \Delta\left(l_{ij} - \varepsilon\right) \tag{4.1.3}$$

$$\Delta(\varphi) = \begin{cases} 1, \varphi < 0 \\ 0, \varphi \ge 0 \end{cases} \tag{4.1.4}$$

样本 x_i 的峰值计算如下：

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i}\left(l_{ij}\right), others \\ \max\limits_{j}\left(l_{ij}\right), \forall j \ne i, \rho_j < \rho_i \end{cases} \tag{4.1.5}$$

公式(4.1.3)中 $l_{ij} = m_e\left(x_i, x_j\right)$，表示样本 x_i 和 x_j 的不相似性，ε 表示截断阈值。

定义 6 近邻类别

$MDN\left(x_i\right)$ 为样本 x_i 的块相似近邻样本所组成的集合，则样本 x_i 的近邻类别 l_i 为：

$$CM(i) = \arg\max_i \left|F_{Mi}\right| \tag{4.1.6}$$

$$l_i = y_{CM(i)} \tag{4.1.7}$$

其中 F_{Mi} 表示 $MDN\left(x_i\right)$ 中标签为 y_i 的样本集合，$CM(i)$ 表示 $MDN\left(x_i\right)$ 中出现次数最多的样本标签。

当样本 x_i 标签 y_i 不同于样本 x_i 的近邻类别 l_i，样本 x_i 为噪声样本。

4.2 块估计近邻搜索算法

为了寻找到所有样本的块相似近邻，提出了块估计近邻搜索算法。算法 1 是块估计近邻搜索算法的伪代码，第二步计算样本的不相似性矩阵，第 3-9 步，搜索样本的块相似近邻并记录离群点，第 10 步，计算离群点的个数，当遍历所有样本，离群点数目不变时，算法达到稳定状态，具体步骤如第 11-20 步。在算法 1 中，MDN 表示样本的块近邻相似集合，$RMN_k\left(x_i\right)$ 表示样本 x_i 的最 k 相似近邻集合，M 表示计算得到的不相似性度量矩阵，λ 记录了样本所能拥有的最多的块近邻相似样本的数量。算法 1 描述了块

估计近邻搜索算法。

Algorithm 1 MDN_Search

输入 X

输出 MDN, M, λ

1 $\alpha = 1$， $\forall x_i \in X$， $MDN(x_i) = \varnothing, MN_\alpha(x_i) = \varnothing, MNb(x_i) = \varnothing, RMN_k(x_i) = \varnothing$

2 使用公式(2.7.2)在 X 上计算不相似性度量矩阵 M

3 For x_i in X

4 If $m(x_j, x_i | H; D) < 0.5$

5 $MN_\alpha(x_i) = MN_\alpha(x_i) \cup \{x_j\}$

6 $MNb(x_j) = MNb(x_j) + 1$

7 $RMN_k(x_j) = RMN_k(x_j) \cup \{x_i\}$

8 End if

9 End for

10 $qv = |LQ|$

11 If qv 不发生改变

12 $\lambda = \alpha$

13 For x_i in X

14 $MDN(x_i) = RMN_\lambda(x_i) \cap MN_\lambda(x_i)$

15 end for

16 return MDN, λ

17 else

18 $\alpha = \alpha + 1$

19 Go step 3

20 End if

4.3 高置信度样本选择算法

在 MDSF 算法中，使用不相似性度量方式计算出样本的局部密度和峰值，以确定样本之间的关系指向。得到样本的关系指向后可以构造出块估计近邻关系图。在 MDSG 算法中第 2-4 步，计算每个样本的局部密度和峰值，第 5-10 步，找到每个无标签样本最相似的密度高于它的有标签样本确定关系指向，在第 11-20 步，依照样本之间的关系指向将所有有标签样本的上一个和下一个样本赋予标签后添加进有标签样本集。算法 2 中，*order* 记录每个样本在经过多少次迭代后被标记，ε 表示截断阈值，*arrows* 表示样本的关系指向方向，箭头从无标签样本指向有标签样本。

关系指向图示例如下：图 4.1 中，正方形表示类 1 中的样本，圆圈表示类 2 中的样本，五角星表示类三中的样本，实心表示有标签样本，空心表示无标签样本。样本 x 的关系指向为：从样本 x 指向距离样本 x 最近的密度大于样本 x 的样本点。

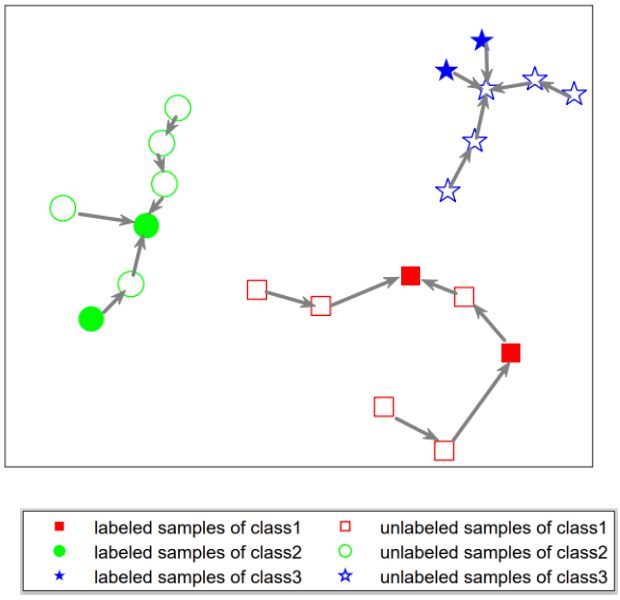


图 4.1 关系指向图

自训练算法得到的分类器的性能主要依赖高置信度样本的选取质量，提出了 *MDNE* 算法，将每个样本的近邻类别与样本标签进行对比，将与近邻类别不一致的样本作为噪声点删除，从而提升了高置信度样本的选取质量。在算法 3 中，第 2-5 步，选择每个样本的前 *order* 个相似近邻生成块近邻相似集合 *MDN*，根据 *MDN* 计算样本的近邻类别，第 6-9 步，对样本进行编辑，当样本标签与它的近邻类别不一致，则该样本为噪声样本，选择标签与近邻类别一致的样本作为高置信度样本。

Algorithm 2 MDSG

输入 L,U,ε,M

输出 $order$

1 $X=[L;U],count=1,order=\varnothing$

2 For x_i in X, DO

3 $order(i)=0$，使用公式(4.1.3)和(4.1.5)计算局部密度和峰值

4 End for

5 For x_i in U

6 For x_j in L

7 $P(i)=\arg\min_{j:\rho_i<\rho_i} l_{ij}$

8 $arrows(i)=P(i)$

9 End for

10 End for

11 重复步骤，直到有标签样本集L的所有的下一个样本点都从无标签样本集U中被选择。

12 If 无标签样本x_i是有标签样本集L的下一个样本点。

13 $L=L\cup x_i,U=U-x_i,order(x_i)=count$

14 End if

15 $count=count+1$

16 重复步骤，直到有标签样本集L的所有的上一个样本点都从无标签样本集U中被选择

17 If 无标签样本x_i是有标签样本集L的上一个样本点

18 $L=L\cup x_i,U=U-x_i,order(x_i)=count$

19 End if

20	$count = count + 1$

Algorithm 3 MDNE	
输入	$X, MDN, order$
输出	ES
1	$ES = \varnothing$
2	For x_i in X
3	选择前 $order$ 个块估计最相似近邻样本。
4	使用 MDN 计算 l_i。
5	End for
6	For x_i in X
7	If $y_i = l_i$
8	$ES = ES \cup x_i$
9	End if
10	End for

4.4 块估计近邻编辑的 Self-training 算法（MDSF）

MDSF 算法首先使用块相似近邻搜索算法找出每个样本的块近邻集合，然后使用基于块的不相似性度量方式计算距离矩阵，得到样本局部密度和峰值后，根据密度与峰值确定样本的关系指向，构造块估计近邻关系图，在有标签样本集上训练初始分类器，根据块估计近邻关系使用初始分类器为无标签样本赋予标签，将与样本块近邻类别一致的样本作为高置信度样本添加进有标签样本集中进行训练，迭代优化，得到最终的分类器。MDSF 算法总结如下：

在算法 4 中，第 2 步搜索样本的块估计近邻，第 3 步记录样本在多少次迭代后被标记，第 4 步初始化基分类器，第 7-11 步，选择无标签样本集的一个子集为其赋予标签，第 12 步进行数据编辑选择高置信度样本点，第 14-16 步，更新有标签样本集和无标签样本集，第 19 步使用更新后的有标签样本集训练得到优化后的分类器。

Algorithm 4 MDSF

输入 L,U,ε

输出 H

$X=[L;U], order=\varnothing, TU=\varnothing$

$[MDN，M]=MDN\_Search(X)$

$order=MDSG[L,U,\varepsilon,M]$

在有标签样本集 L 上训练分类器 H

$count=1$

While $count<\max(order)$, Do

For x_i in U

If $order(x_i)=count$

$TU=TU\cup x_i$

End if

End for

使用分类器 H 为样本集 TU 赋予标签

$ES=MDNE[TU,MDN,order]$

更新有标签样本集 $L\leftarrow L\cup ES$

更新无标签样本集 $U\leftarrow U-ES$

在更新后的有标签样本集 L 上训练分类器 H

$count=count+1$

End while

在最终得到的有标签样本集 L 上训练分类器 H

4.5 实验设置

实验在 32G 内存、64 位 Windows 10 操作系统和 Inter Core i9 处理器的环境下进行。IDE 编程环境为 MATLAB2019b 版本。

实验所用数据集均为公开数据集。在 16 个数据集中，AR、COIL20、FERET 、ORL、Palm、YaleB 和 Yeast 均为图像数据集，且数据集都比较大， Heart 为心脏单质子发射计算机断层扫描（SPECT）图像诊断信息数据集， Sonar 为声纳数据集，还使用了 8 个 UCI 数据集，Australian ，BUPA， Cars，Cleve，Heart，Sonar， Vehicle 和 Zoo。

数据集详细信息如表 4.2 所示：

表 4.2 数据集

序号	数据集	样本数	维数	簇数	缩写
1	AR[36]	1680	1024	120	AR
2	Australian	690	14	2	AUS
3	BUPA	345	6	2	BUP
4	Cars	392	8	3	CAR
5	Cleve[20]	303	13	4	CLE
6	COIL20[2,42]	1440	1024	20	COL
7	FERET32x32[76]	1400	1024	200	FER
8	Heart[3]	270	13	2	HEA
9	Solar	208	60	2	SOL
10	ORL[5]	400	1024	40	ORL
11	Palm[6]	2000	256	100	PAL
12	Sonar[7]	208	60	2	SON
13	Vehicle	323	12	6	VEH
14	YaleB[17]	2414	1024	38	YAL
15	Yeast	1484	1470	10	YEA
16	Zoo	101	16	8	Zoo

[5] http://www.uk.research.att.com/facedatabase.html
[6] https://www.gwern.net/Crops
[7] https://machinelearningmastery.com/standard-machine-learning-datasets/

本文所提算法为自训练算法，选取同类的 SETRED、STDP-CEW、STDPNaN 和 STDP 算法进行对比实验。为验证所提算法的去噪能力，与数据编辑技术 ENN，DE 和 LSEdit 在自训练框架下进行对比实验。对比算法的运行参数根据原文设置，具体情况如下所示：在 SETRED、STDP-CEW 和 STDP 算法中，α 为 DPC 算法中的距离截取阈值，θ 为置信度水平阈值，设置 $\alpha=2$，$\theta=0.1$。在算法 DE 中，设置 $K=3,k=2$。在 MDSF 算法中，$\varepsilon=0.5$。

实验中，选择 KNN 作为基分类器，KNN 的运行参数如下：K=3，使用欧式距离进行度量选择距离样本最近的 K 个数。在实验中，使用准确率和标准差作为评价指标对分类性能进行评估。为了验证所提算法的显著性，在 95%的置信度水平下使用 Wilcoxon 符号秩检验进行统计检验，符号“+”、“~”和“-”分别表示 MDSF 算法性能优于，等于和差于对比算法。

4.6 分类性能与分析

在给定的 16 个基准数据集上进行实验，初始有标签比例为 10%，即训练数据集中有 10%的样本有标签，90%的样本无标签，8 个算法均运行 50 次。根据其他文章[24,31,46]，分类性能指标选择准确率（Accuracy），计算 50 次运行结果的平均值（mean）和方差（std）。实验结果如表 4.3 所示。

表 4.3 各算法在 16 个数据集上的分类性能（准确率）（均值±标准差）

Accuracy	DE	ENN3	LSEdit	SETRED	STDPCEW	STDPNaN	STDP	MDSF
AR	82.02(5)	84.35(3)	85.31(2)	81.42(7)	81.96(6)	84.22(4)	77.45(8)	**98.79(1)**
	±1.27	±1.01	±0.81	±0.77	±1.34	±1.95	±1.14	**±0.06**
AUS	63.59(8)	65.39(3)	65.66(2)	64.06(6)	65.31(4)	65.24(5)	64.17(6)	**70.53(1)**
	±3.96	±1.88	±2.61	±2.07	±2.64	±4.08	±2.76	**±1.64**
BUP	59.89(5)	60.7(3)	60.94(2)	60.37(4)	58.96(7)	58.73(8)	59.57(6)	**67.88(1)**
	±4.89	±3.14	±4.04	±4.06	±4.05	±3.86	±2.8	**±3.76**
COI	90.45(7)	91.22(5)	91.64(3)	91.61(4)	88.23(8)	92.14(2)	90.96(6)	**94.7(1)**
	±0.92	±1.41	±0.78	±0.83	±1.75	±1.17	±0.83	**±0.23**

续表 4.3

Accuracy	DE	ENN3	LSEdit	SETRED	STDPCEW	STDPNaN	STDP	MDSF
FER	88.95(5) ±0.6	89.80(3) ±1.05	90.09(2) ±0.63	86.32(7) ±0.62	87.41(6) ±1.16	89.20(4) ±1.20	83.17(8) ±0.87	**99.14(1) ±0.13**
SOL	79.44(4) ±0.06	79.31(5) ±0.08	81.09(3) ±0.19	78.46(6) ±1.77	81.95(2) ±0.49	76.87(8) ±0.18	78.06(7) ±0.08	**82.95(1) ±3.42**
ORL	85.25(5) ±1.00	86.99(4) ±1.40	87.54(3) ±2.18	85.11(6) ±1.8	84.1(7) ±1.33	89.78(2) ±1.13	83.87(8) ±1.57	**96.47(1) ±0.03**
PAL	86.47(6) ±0.55	88.92(4) ±1.06	90.10(2) ±0.82	87.68(5) ±0.92	85.59(8) ±0.7	89.68(3) ±0.65	85.63(7) ±1.24	**98.66(1) ±0.02**
SON	59.39(8) ±6.32	61.72(6) ±6.73	61.96(4) ±5.43	65.03(2) ±2.53	59.54(7) ±5.29	63.53(3) ±5.08	61.78(5) ±5.35	**69.72(1) ±1.02**
VEH	69.96(7) ±1.98	70.73(3) ±1.99	70.83(2) ±1.65	70.72(4) ±1.52	73.09(2) ±1.84	70.22(6) ±0.97	70.54(5) ±0.89	**78.44(1) ±1.8**
YAL	70.1(7) ±3.18	76.26(2) ±1.1	75.09(4) ±2.26	73.06(5) ±1.51	70.50(6) ±1.20	75.88(3) ±1.78	68.97(8) ±1.33	**95.97(1) ±1**
CAR	65.04(8) ±5.22	68.44(4) ±3.35	68.46(3) ±3.46	67.89(6) ±2.41	69.76(2) ±3.65	68.22(5) ±3.47	66.58(7) ±4.59	**78.59(1) ±6.56**
CLE	77.12(6) ±3.67	80.01(2) ±1.37	80.8(2) ±1.39	79.12(3) ±3.48	78.16(4) ±4.64	76.63(7) ±5.65	77.82(5) ±5.81	**83.98(1) ±4.98**
HEA	63.75(2) ±2.12	62.77(4) ±5.8	62.45(6) ±4.73	62.62(5) ±2.49	59.74(8) ±5.73	62.17(7) ±3.97	63.33(3) ±3.94	**66.77(1) ±4.77**
YEA	86.09(7) ±0.46	86.22(5) ±0.17	86.22(5) ±0.05	87.47(3) ±2.34	84.6(8) ±2.75	87.51(2) ±2.46	87.2(4) ±1.97	**89.25(1) ±2.51**
Zoo	84.24(8) ±3.58	87.01(3) ±3.72	84.29(7) ±2.8	87.05(2) ±3.26	85.89(5) ±1.69	86.65(4) ±2.22	85.05(6) ±3.64	**87.13(1) ±2.73**
WSR-test	+	+	~	+	+	+	+	**N/A**
AVE.Acc	73.38	74.89	76.87	74.26	73.04	74.91	72.86	**84.31**
AVE.Std	3	2.54	1.87	2.18	2.4	2.65	2.51	2.37
AVE.Rank	6.05	3.9	3.35	4.7	5.7	4.5	6.1	**1**

从表 4.3 的实验结果可得如下结论：

（1）在所有的 16 个数据集上，提出的算法 MDSF 的分类性能均高于对比算法 DE、LSEdit、ENN3、SETRED、STDP-CEW、STDPNaN 和 STDP，这是因为 MDSF 算法使用的度量方式可以将边缘数据进行压缩，提升高置信度样本的选取质量，从而提升分类器性能；

（2）MDSF 算法的平均标准差在所有的算法中排序为 3，这是因为每次选取的有标签样本均为随机抽取，在不相似性度量的过程中，每次不同的数据样本及标签会导致误差较大。

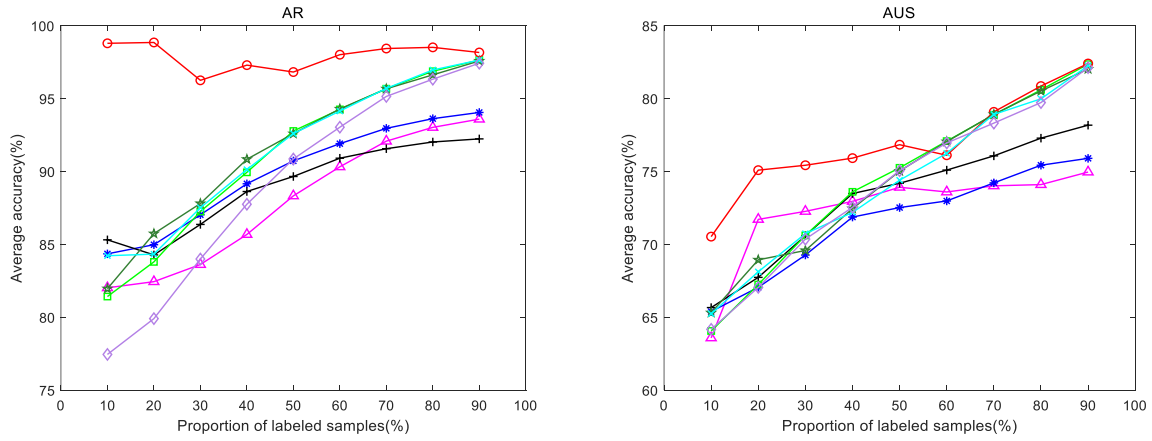
（3）在 AR、COI、FER、PIE、ORL 和 YAL 这 6 个数据集上，提出的算法 MDSF 的分类性能均超过了 90%，验证了提出的算法 EBSA 的良好的分类性能，这 6 个数据集均为图像数据集，显示出 MDSF 处理高维图像数据的能力；

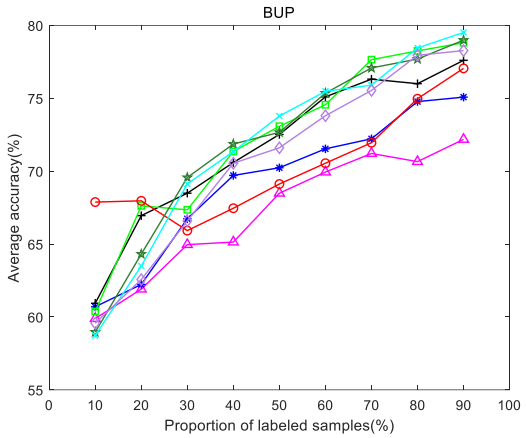
（4）与 DE、ENN3、SETRED、STDP-CEW、STDPNaN 和 STDP 算法相比，在 95% 的置信度水平上进行的统计检验实验结果显示的算法 MDSF 实验结果显著，验证了实验的有效性。

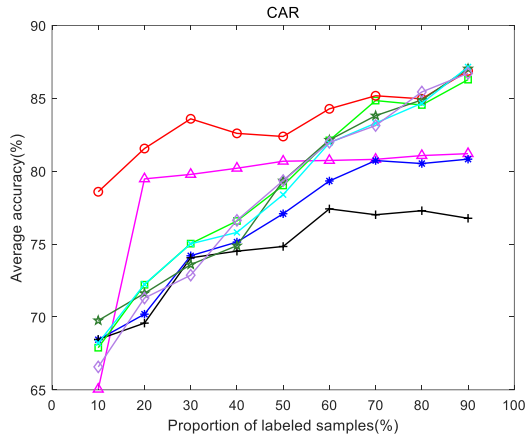
（5）16 个数据集类型多样，有图像数据集，大型数据集，小型数据集，实验结果证明了 MDSF 算法具有广泛适用能力。

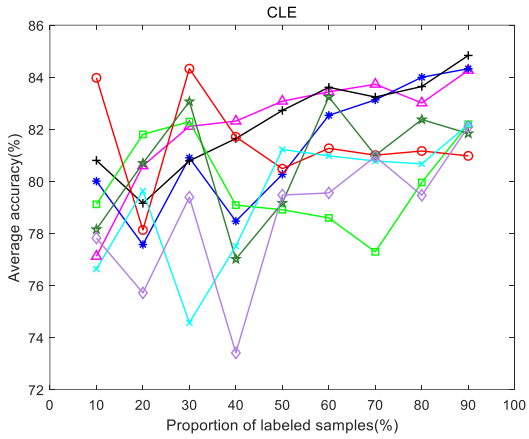
4.7 有标签样本比例对算法分类性能的影响

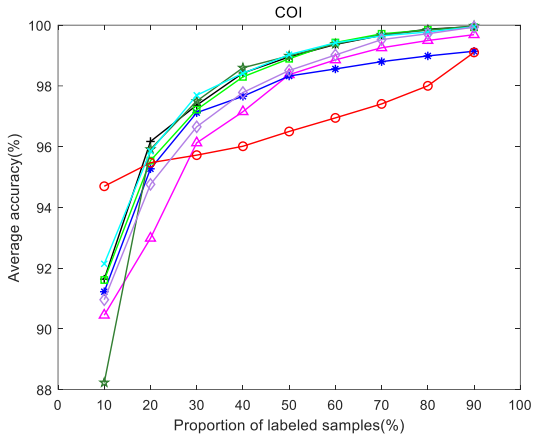
为验证有标签样本比例对对比算法分类性能的影响，随机选取有样本标签比例为 10%~90%，步进为 10%，在 16 个数据集上实验。实验结果如图 4.1：

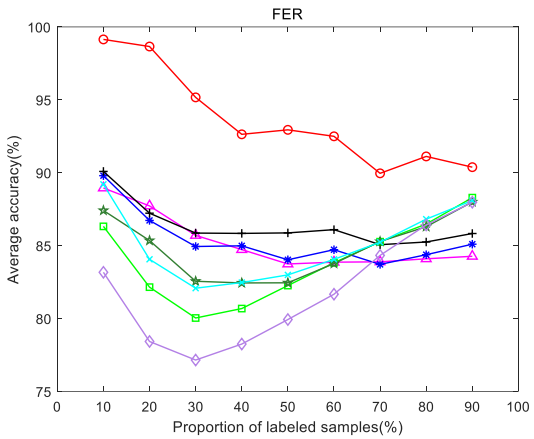


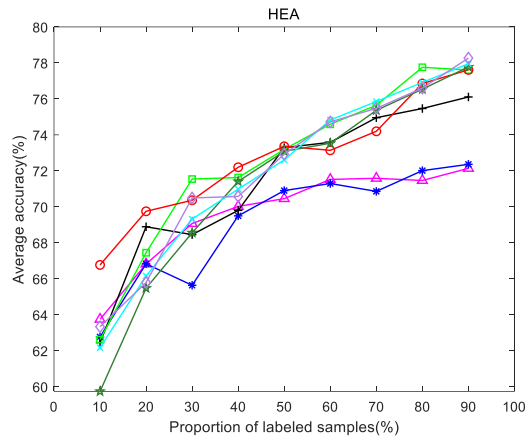

BUP
Average accuracy(%)
Proportion of labeled samples(%)

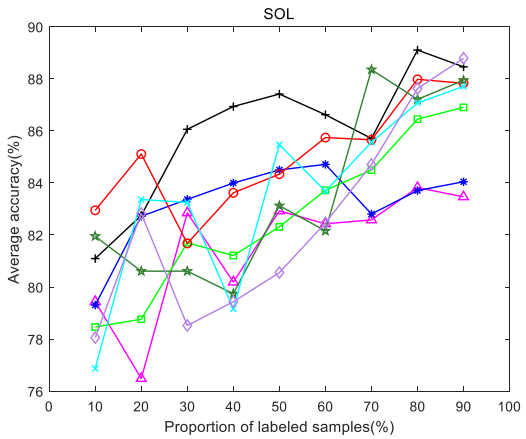

CAR
Average accuracy(%)
Proportion of labeled samples(%)

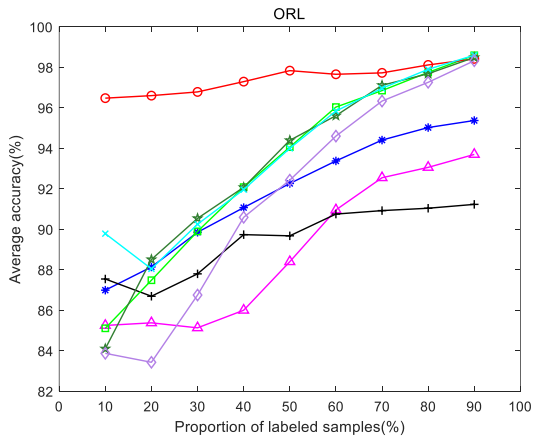

CLE
Average accuracy(%)
Proportion of labeled samples(%)


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


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


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


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


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

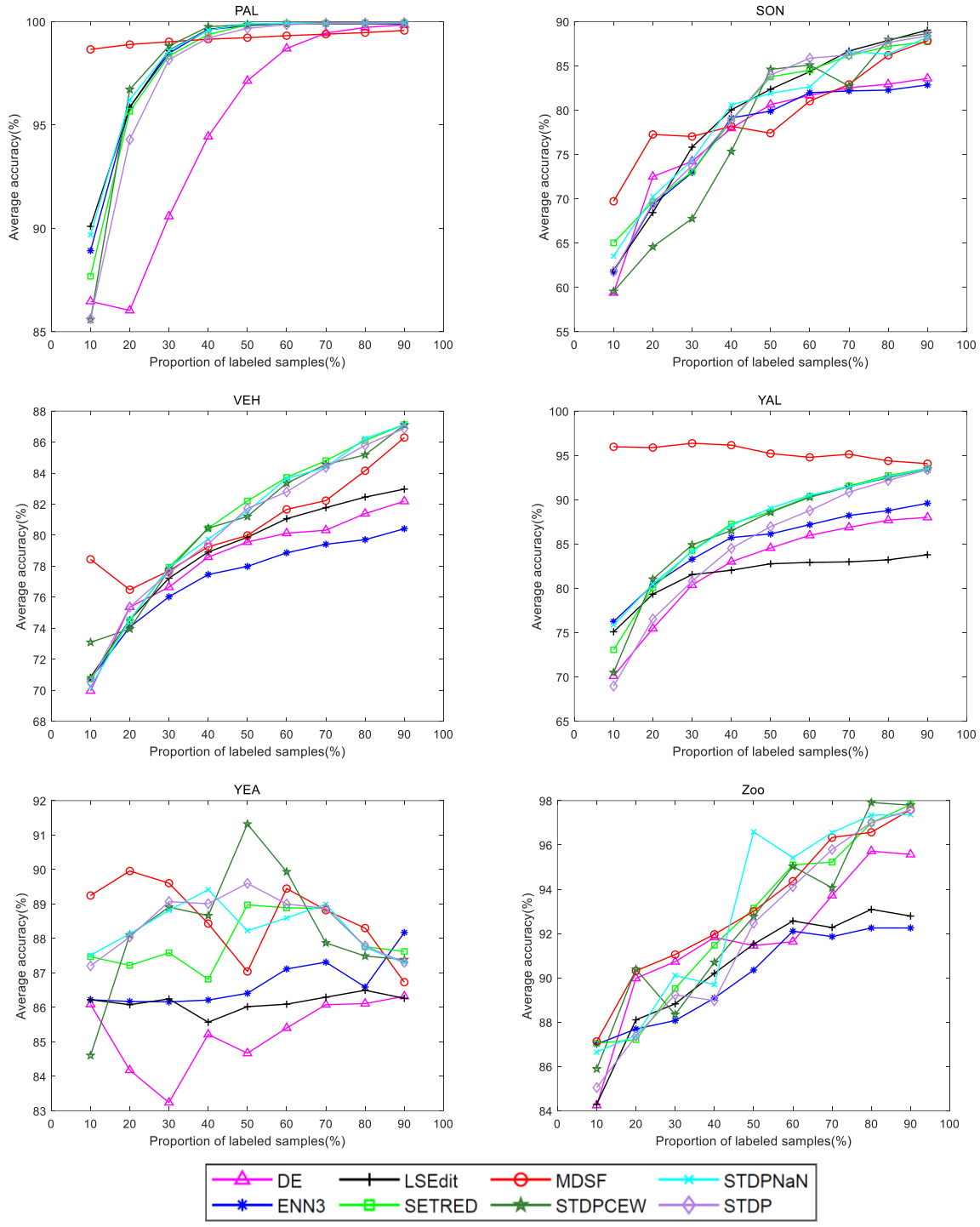


图 4.1 有标签样本比例对算法的影响

从图 4.1 可得到如下结论：

（1）随着有标签样本比例增加，提出的算法 MDSF 在 AR、ORL、CAR、YAL 和 FER 数据集上的分类性能始终优于对比算法 DE、LSEdit、ENN3、SETRED、STDP-CEW、STDPNaN 和 STDP，由此可以看出提出算法的良好的分类性能，这 5 个数据集均为图像数据集，体现了 MDSF 对图像数据集良好的处理能力；

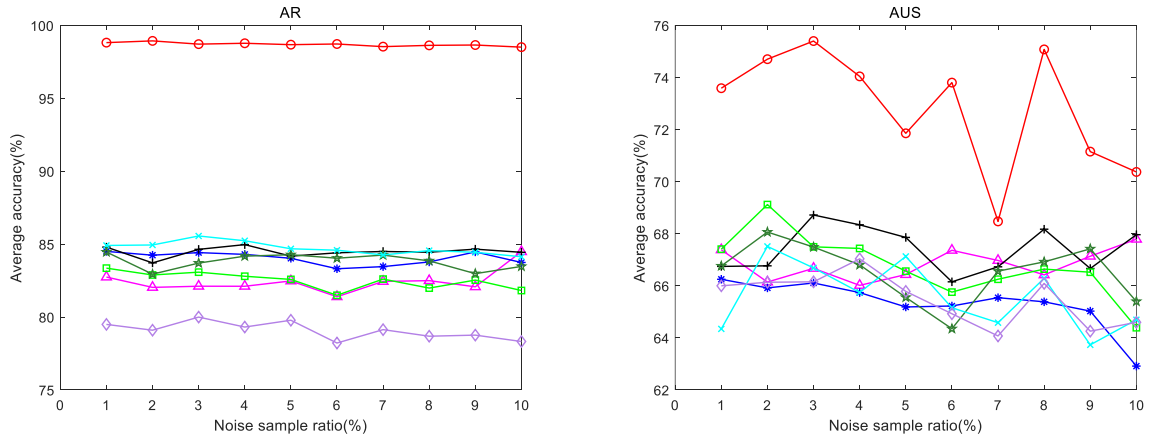
（2）提出的算法 MDSF 在样本比例较少时获得了较高的准确率，而其他对比算法在有标签样本比较多时，才能获得较高的准确率，因此提出的算法 MDSF 对于现实中大量数据集仅存在少量标签的情况更加适应。

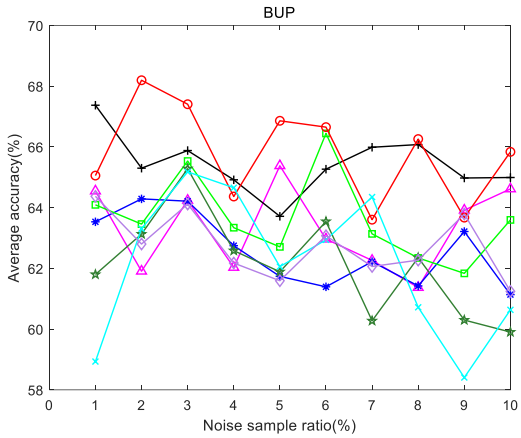
（3）随着有标签比例的增加，算法 MDSF 的分类性能达到饱和状态，这是因为在半监督算法中，随着有标签比例的增加，MDSF 算法在自训练过程中可能会误分类样本，从而影响分类器性能。

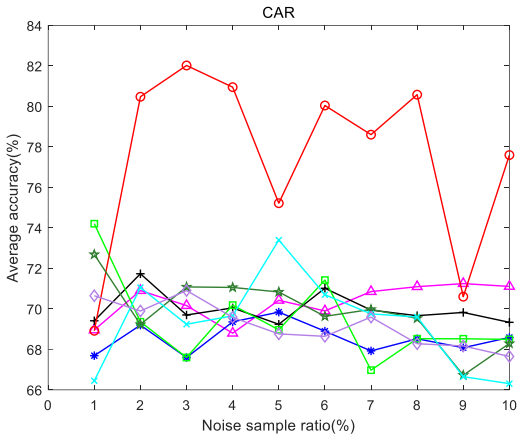
（4）随着有标签比例的增加，提出的算法 MDSF 在 AR、COI、HEA、BUP、ORL、PAL、CAR、SON、AUS 和 YAL 这 10 个数据集上的分类性能都比较平稳。在不同的有标签比例条件下，MDSF 算法的分类性能曲线比较平滑，这显示了 MDSF 算法具有很好的鲁棒性。

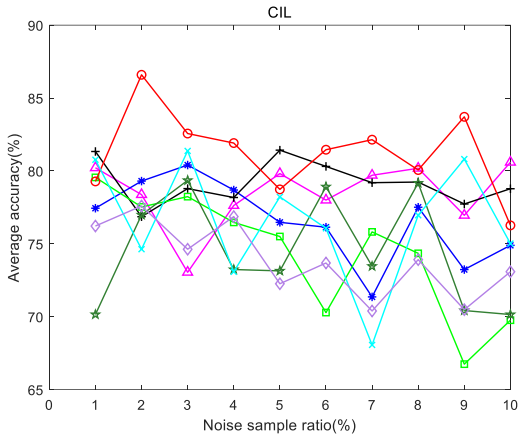
4.8 噪声比例实验分析

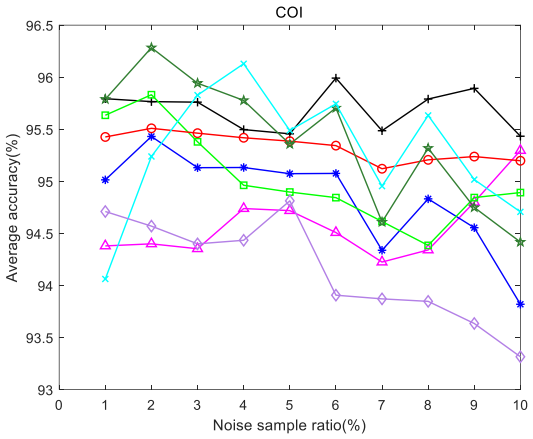
MDSF 算法在迭代训练的过程中对数据进行编辑，因此具有一定的去噪能力，能够有效地纠正数据集中标错的数据。为了验证 MDSF 的去噪能力，进行了噪声实验。随机选取 16 个数据集中 20%的样本赋予标签，在有标签样本集中随选取[1%,10%]的数据，步进为 1%，并赋予其错误标签，然后进行实验。选择准确率作为分类性能评价指标，每个数据集上各算法均进行了 50 次实验，计算各次实验结果的均值和标准差作为实验结果。实验结果如图 4.2 所示：

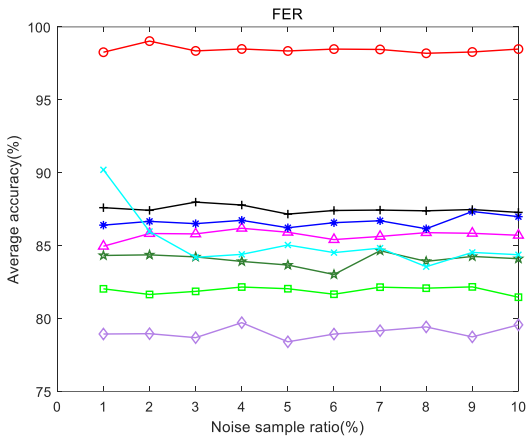


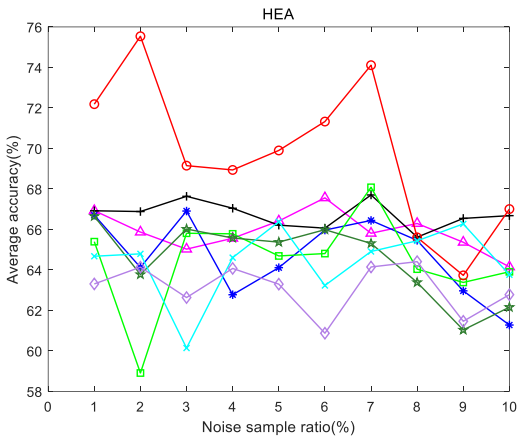

BUP
Average accuracy(%)
Noise sample ratio(%)

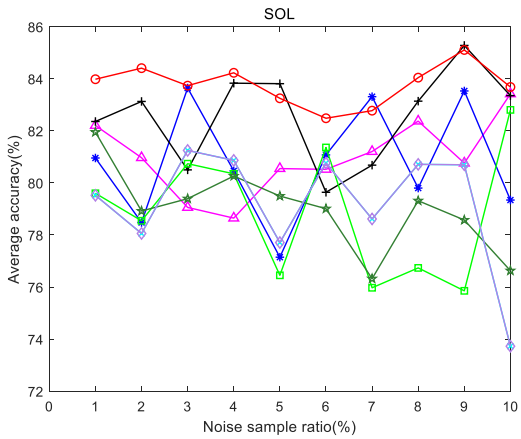

CAR
Average accuracy(%)
Noise sample ratio(%)

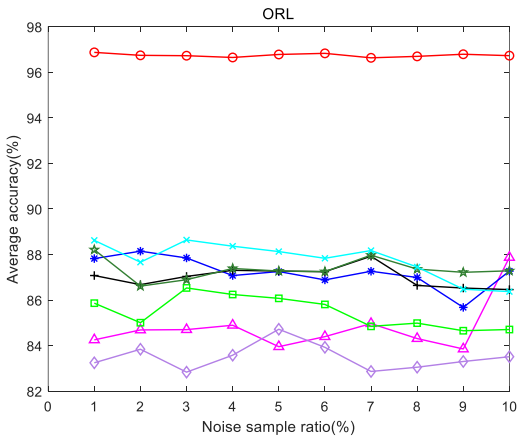

CIL
Average accuracy(%)
Noise sample ratio(%)


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


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


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


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


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

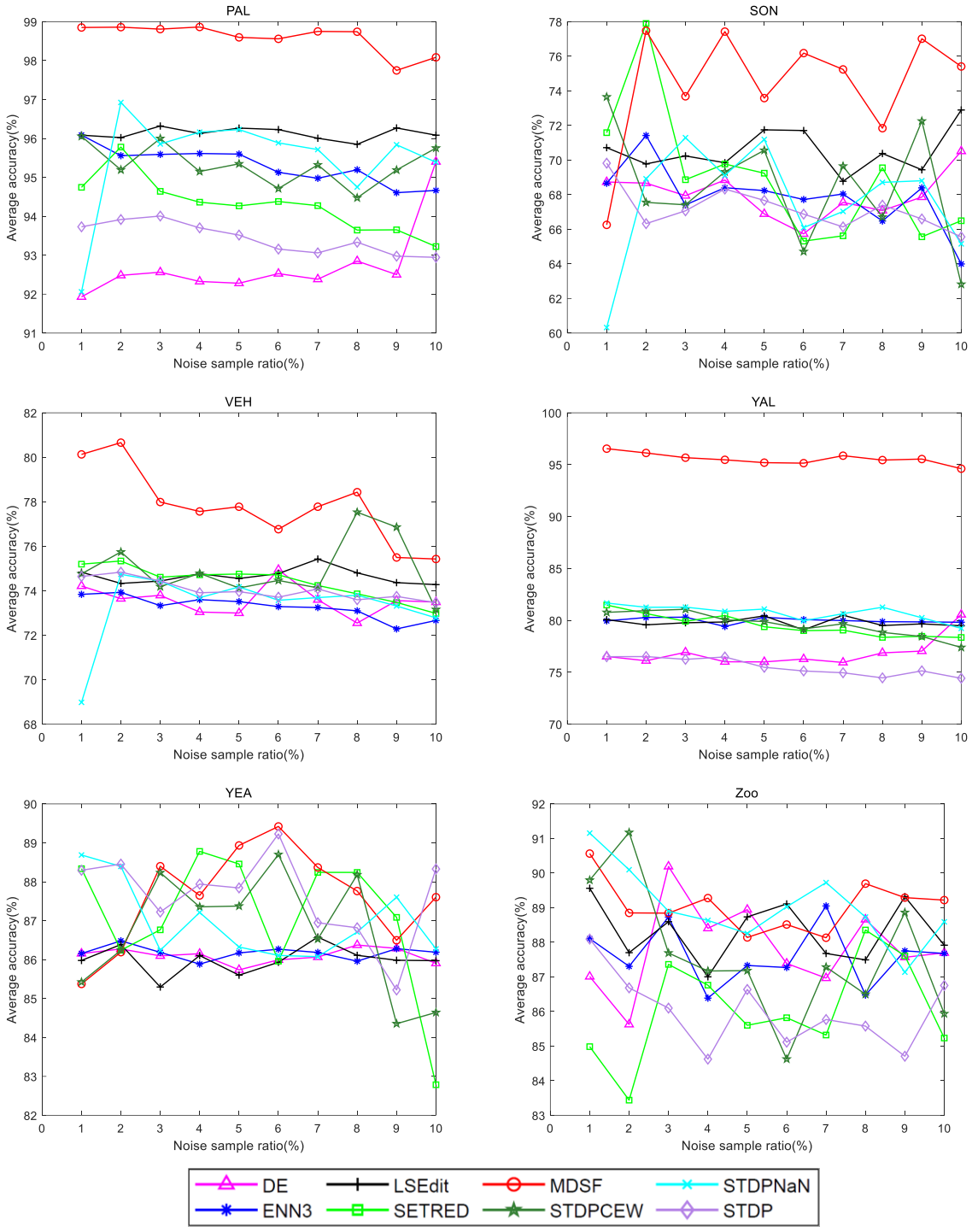


图 4.2 噪声比例对算法性能的影响

从图 4.2 中可以得出以下结论：

（1）提出的算法 MDSF 的分类性能在 AR、COI、FER、ORL、PAL 和 YAL 数据集上的分类性能均高于 90%，在 CLE、Zoo 和 YEA 数据集上的分类性能均高于 80%，由此可以证明提出的算法 MDSF 的良好的去噪声的能力；这是因为现有的数据编辑算法

DE、LSEdit、ENN3 和 SETRED 仅利用有标签样本集中的信息过滤噪声样本，在有标签比例较少的半监督分类算法中，适用性不强，而 MDSF 可以同时利用数据编辑技术过滤噪声样本，更适用于现在的大数据时代。

（2）提出的算法 MDSF 的分类性能在 AR、CLE、HEA、FER、AUS、ORL、PAL、CAR、SON、VEH 和 YAL 这 11 个数据集上的分类性能远优于 DE、LSEdit、ENN3、SETRED、STDP-CEW、STDPNaN 和 STDP 算法，证明 MDSF 算法的编辑方法去噪声的性能优良，这 11 个数据集中有 7 个图像数据集，由此可以看出 MDSF 算法对图像数据集良好的处理能力。

4.9 算法时间复杂度分析

令 n 表示输入样本的个数，d 表示维度，t 表示迭代次数，c 表示类簇数，h 表示树的个数，ϕ 表示树的高度，MDSF 算法计算样本距离矩阵的时间复杂度为 $O\left(n^2 h\log\phi + h\phi\log\phi\right)$，寻找块相似近邻的时间复杂度为 $O\left(n\log n\right)$，高置信度样本选择算法的时间复杂度为 $O\left(n\right)$，故 MDSF 算法的整体时间复杂度为 $O\left(n^2 h\log\phi + h\phi\log\phi + tn + n\log n\right)$。STDP-CEW 算法计算密度峰值的时间复杂度为 $O\left(n^2\right)$，构造相关邻接图的时间复杂度为 $O\left(tdn^3\right)$，故 STDP-CEW 算法的整体时间复杂度为 $O\left(tdn^3\right)$。STDPNaN 算法主要在于使用 DPC 发现空间结构和寻找自然近邻 NaN，其时间复杂度分别为 $O\left(n^2\right)$ 和 $O\left(n\log n\right)$，故 STDPNaN 的整体时间复杂度为 $O\left(tn^2\right)$。DE 和 ENN3 算法主要在于搜索样本 k 近邻，时间复杂度为 $O\left(tn^2\right)$。LSEdit 算法主要在于寻找自然近邻 NaN，时间复杂度为 $O\left(n\log n\right)$。SETRED 算法主要在于构造相关邻接图，时间复杂度为 $O\left(tdn^3\right)$。STDP 算法主要在于使用 DPC 发现空间结构，时间复杂度为 $O\left(tn^2\right)$。综上所述，提出的算法 MDSF 的时间复杂度低于 STDP-CEW 和 SETRED，高于其他 5 个对比算法,相比于 MDSF 的分类性能，在时间上有所牺牲是可以接受的。

5 总结与展望

5.1 总结

传统的监督分类算法通常需要大量的有标签样本，然而现有的数据集有标签样本较少，为所有的数据样本进行标注代价高昂。为了充分利用无标签样本的潜在信息，半监督分类学习算法应运而生。半监督分类算法可以使用少量有标签样本和大量无标签样本进行学习。目前的 Self-training 算法框架下的半监督机器学习算法计算复杂度高，不适用于大数据场景的应用。针对这一问题，在球簇定义的基础上，提出了一种快速球簇划分的数据编辑方法，该方法不仅能够方便的处理噪声数据，而且可以快速的选取高置信度样本进行迭代训练。与对比的基于 Self-training 算法框架下的相比，EBSA 算法的训练速度得到大幅提升，同时 EBSA 算法还具有无参的特性，使其更方便进行实际应用。由于 EBSA 预设球簇的个数与数据类别的个数相等，在有标签样本数据较多的数据集上，初始球簇容易出现相互包含的情况，从而降低了算法的性能。在有标签数据较少的情况下，对于不平衡的数据集，容易出现数据量少的类别无法计算出球簇，从而降低了 EBSA 算法的性能。拟下一步对这两种情况进行研究。

现有的 Self-training 框架下的半监督机器学习算法大多使用欧式距离计算样本之间的距离，不适用于高维数据场景的应用。针对这一问题，基于块的不相似性度量方式，提出了一种块估计近邻编辑方法 MDSF。MDSF 算法使用不相似性度量方式计算距离矩阵，提升了算法在高维空间的分类性能。 MDSF 算法搜索样本的块估计近邻，根据样本的局部密度和峰值确定样本之间的关系走向构造块估计近邻关系图，提出了一个新的高置信度样本选取算法，该方法不仅能够方便的处理噪声数据，而且可以选取高质量的高置信度样本进行迭代训练。与基于 Self-training 框架下的其他算法相比，MDSF 算法在有标签样本比例为 10%的数据集上分类性能良好。 MDSF 算法在有标签样本数据较多的数据集上，与其他算法相比分类性能提升不明显。因此 MDSF 更加适用于有标签数据较少的情况，在 16 个数据集上进行的对比实验可以充分说明提出算法 MDSF 的良好分类性能。在噪声实验中，MDSF 算法在不同的噪声样本比例下分类性能优于其他的对比算法，且性能稳定，表现了 MDSF 算法良好的数据编辑能力。

5.2 展望

总体而言，所提出的 EBSA 具有以下优点：（1）与基于自训练框架的其他相关算法相比，EBSA 的训练速度显著提高。因此，EBSA 可以处理大规模数据。（2） EBSA 可以编辑噪声数据并选择高质量的高可信度样本。因此，学习型分类器的性能得到了提高。（3） EBSA 不需要超参数，这使得它更便于实际应用。

MDSF 算法具有以下优点：（1）使用了不相似性度量方法来衡量样本之间的关系，避免了欧式距离在高维数据集上出现的维度诅咒问题，因此，MDSF 更适合用于高维数据集进行分类；（2）构建了块估计近邻图，可以更详细刻画样本之间的关系，提升了高置信度样本点的选取质量，分类器学习性能得到了提升。

EBSA 算法有两个主要缺点：（1）初始标记数据分布严重影响所提出算法的性能。（2） EBSA 使用球簇分割和编辑算法来编辑噪声数据，使其无法处理非球面分布数据集。因为 EBSA 使用欧氏距离来测量样本之间的距离，所以它不适用于流行数据和混合数字数据。在未来，将从以下方面扩展的算法：（1）通过增加球簇的数量，将 EBSA 算法扩展到非球面分布数据集。（2） 对于分类和混合类型数据集，将采用适当的距离计算方法来测量样本的距离，以便 EBSA 算法可以应用于这些场景。

MDSF 算法有三个主要的缺点：（1）MDSF 算法主要依赖于块估计近邻关系图，因此关系图构建的准确率影响 MDSF 算法的分类性能。（2）因为 MDSF 算法使用不相似性度量方法计算样本相似性，在数据维度低的数据集上相比于其他对比算法运行时间长；（3）MDSF 算法不适合用于流行和混合类型数据集。在未来的工作中，计划研究如何构建准确率更高的关系图，研究 MDSF 算法与多种分类器的适配性。

综上所述，本文所提的算法均不适合用于流行和混合类型数据集，在下一步的工作中，将在提升算法性能和降低算法运行时间的基础上对模型进行修改使其适合流行数据和混合数据类型。

参考文献

[1] Afonso L C S, Rodrigues D, Papa J P. Nature-inspired optimum-path forest[J]. Evolutionary Intelligence, 2021: 1-12.

[2] Aljalbout E, Golkov V, Siddiqui Y, et al. Clustering with deep learning: Taxonomy and new methods[J]. arXiv preprint arXiv:1801.07648, 2018.

[3] Asuncion A, Newman D. UCI machine learning repository: Irvine, CA, USA, 2007.

[4] Bau D, Liu S, Wang T, et al. Rewriting a deep generative model[C]//European Conference on Computer Vision. Springer, 2020: 351-369.

[5] Berthelot D, Carlini N, Goodfellow I, et al. Mixmatch: A holistic approach to semi-supervised learning[J]. arXiv preprint arXiv:1905.02249, 2019: 5050–5060.

[6] Cai H, Liu B, Xiao Y, et al. Semi-supervised multi-view clustering based on constrained nonnegative matrix factorization[J]. Knowledge-Based Systems, 2019, 182. 10.1016/j.knosys.2019.06.006.

[7] Camiña J B, Medina-Pérez M A, Monroy R, et al. Bagging-RandomMiner: a one-class classifier for file access-based masquerade detection[J]. Machine Vision and Applications, 2019, 30(5): 959-974. 10.1007/s00138-018-0957-4.

[8] Cappozzo A, Greselin F, Murphy T B. Anomaly and Novelty detection for robust semi-supervised learning[J]. Statistics and Computing, 2020, 30(5): 1545-1571. 10.1007/s11222-020-09959-1.

[9] Charbuty B, Abdulazeez A. Classification based on decision tree algorithm for machine learning[J]. Journal of Applied Science and Technology Trends, 2021, 2(01): 20-28.

[10] Chen Y, Wang G, Dong S. Learning with progressive transductive support vector machine[J]. Pattern Recognition Letters, 2003, 24(12): 1845-1855.

[11] Cheng D, Zhu Q, Wu Q. A local cores-based hierarchical clustering algorithm for data sets with complex structures[C]//2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC). IEEE, 2018: 410-419.

[12] Cole R, Fanty M. Spoken letter recognition[C]//Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990. 1990.

[13] De Souza R W R, De Oliveira J V C, Passos L A, et al. A novel approach for optimum-path forest classification using fuzzy logic[J]. IEEE Transactions on Fuzzy Systems, 2019, 28(12): 3076-3086.

[14] Dornaika F, Baradaaji A, El Traboulsi Y. Soft Label and Discriminant Embedding Estimation for Semi-Supervised Classification[C]//2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021: 7250-7257.

[15] Gan H, Sang N, Huang R, et al. Using clustering analysis to improve semi-supervised classification[J]. Neurocomputing, 2013, 101: 290-298.

[16] Gan H, Tong X, Jiang Q, et al. Discussion of FCM algorithm with partial supervision[C]//Proceedings of the eighth international symposium on distributed computing and applications to business, engineering and science. 2009: 27-31.

[17] Georghiades A S, Belhumeur P N, Kriegman D J. From few to many: Illumination cone models for face recognition under variable lighting and pose[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001, 23(6): 643-660.

[18] Gu X. A self-training hierarchical prototype-based approach for semi-supervised classification[J]. Information Sciences, 2020, 535: 204-224.

[19] Hasan M S, Wu X, Watson L T, et al. UPS-indel: a universal positioning system for indels[J]. Scientific reports, 2017, 7(1): 1-13.

[20] He Z, Xu X, Deng S. Clustering mixed numeric and categorical data: A cluster ensemble approach[J]. arXiv preprint cs/0509011, 2005.

[21] Lai D T C, Miyakawa M, Sato Y. Semi-supervised data clustering using particle swarm optimisation[J]. Soft Computing, 2019, 24(5): 3499-3510. 10.1007/s00500-019-04114-z.

[22] Li J, Zhu Q. Semi-supervised self-training method based on an optimum-path forest[J]. IEEE Access, 2019, 7: 36388-36399.

[23] Li J, Zhu Q. A boosting Self-Training Framework based on Instance Generation with Natural Neighbors for K Nearest Neighbor[J]. Applied Intelligence, 2020, 50(11): 3535-3553. 10.1007/s10489-020-01732-1.

[24] Li J, Zhu Q, Wu Q. 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.

[25] Li J, Zhu Q, Wu Q. A parameter-free hybrid instance selection algorithm based on local sets with natural neighbors[J]. Applied Intelligence, 2020, 50(5): 1527-1541. 10.1007/s10489-019-01598-y.

[26] Li J, Zhu Q, Wu Q, et al. An effective framework based on local cores for self-labeled semi-supervised classification[J]. Knowledge-Based Systems, 2020, 197. 10.1016/j.knosys.2020.105804.

[27] Li M, Zhou Z-H. SETRED: Self-training with editing[C]//Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 2005: 611-621.

[28] Liu J, Liu Y, Luo X. Semi-supervised learning method[J]. Chinese Journal of Computers, 2015, 38(8): 1592-1617.

[29] Liu S, Ding C, Jiang F, et al. Auto-weighted Multi-view learning for Semi-Supervised graph clustering[J]. Neurocomputing, 2019, 362: 19-32. 10.1016/j.neucom.2019.07.011.

[30] Liu Y. Self-training algorithm combining density peak and cut edge weight[J]. J. Vis. Lang. Comput., 2020, 2020(1): 11-16.

[31] Livieris I E, Kanavos A, Tampakas V, et al. An auto-adjustable semi-supervised self-training algorithm[J]. Algorithms, 2018, 11(9): 139.

[32] Ma F, Meng D, Xie Q, et al. Self-paced co-training[C]//International Conference on Machine Learning. PMLR, 2017: 2275-2284.

[33] Ma J, Wang N, Xiao B. Semi-supervised classification with graph structure similarity and extended label propagation[J]. IEEE Access, 2019, 7: 58010-58022.

[34] Manapragada C, Webb G I, Salehi M. Extremely fast decision tree[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018: 1953-1962.

[35] Marcacini R M, Rossi R G, Matsuno I P, et al. Cross-domain aspect extraction for sentiment analysis: A transductive learning approach[J]. Decision Support Systems, 2018, 114: 70-80.

[36] Martínez A, Benavente R. The AR face database, 1998[J]. Computer vision center, technical report, 2007, 3(5).

[37] Mclachlan G J. Mahalanobis distance[J]. Resonance, 1999, 4(6): 20-26.

[38] Metzenthin E, Bartz C, Meinel C. Weakly Supervised Scene Text Detection using Deep

Reinforcement Learning[J]. arXiv preprint arXiv:2201.04866, 2022.

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

[40] Myles A J, Feudale R N, Liu Y, et al. An introduction to decision tree modeling[J]. Journal of Chemometrics: A Journal of the Chemometrics Society, 2004, 18(6): 275-285.

[41] Nartey O T, Yang G, Wu J, et al. Semi-supervised learning for fine-grained classification with self-training[J]. IEEE Access, 2019, 8: 2109-2121.

[42] Nene S, Nayar S, Murase H. Columbia image object library (COIL-20)[J]. Technical Report CUCS006-96., 1996.

[43] Nie F, Xu D, Tsang I W, et al. Flexible manifold embedding: a framework for semi-supervised and unsupervised dimension reduction[J]. IEEE Trans Image Process, 2010, 19(7): 1921-1932. 10.1109/TIP.2010.2044958.

[44] Olsson V, Tranheden W, Pinto J, et al. ClassMix: Segmentation-Based Data Augmentation for Semi-Supervised Learning[C]//2021 IEEE Winter Conference on Applications of Computer Vision (WACV). 2021: 1368-1377.

[45] Oord A V D, Dieleman S, Zen H, et al. Wavenet: A generative model for raw audio[J]. arXiv preprint arXiv:1609.03499, 2016.

[46] Piroonsup N, Sinthupinyo S. Analysis of training data using clustering to improve semi-supervised self-training[J]. Knowledge-Based Systems, 2018, 143: 65-80. 10.1016/j.knosys.2017.12.006.

[47] Rodriguez A, Laio A. Clustering by fast search and find of density peaks[J]. science, 2014, 344(6191): 1492-1496.

[48] Safavian S R, Landgrebe D. A survey of decision tree classifier methodology[J]. IEEE transactions on systems, man, and cybernetics, 1991, 21(3): 660-674.

[49] Sánchez J S, Barandela R, Marqués A I, et al. Analysis of new techniques to obtain quality training sets[J]. Pattern Recognition Letters, 2003, 24(7): 1015-1022.

[50] Sánchez J S, Pla F, Ferri F J. Prototype selection for the nearest neighbour rule through proximity graphs[J]. Pattern Recognition Letters, 1997, 18(6): 507-513.

[51] Shibata H, Hanaoka S, Nomura Y, et al. A versatile anomaly detection method for medical images with a flow-based generative model in semi-supervision setting[J]. arXiv

preprint arXiv:2001.07847, 2020.

[52] Shindler M, Wong A, Meyerson A. Fast and accurate k-means for Large Datasets[C]//nips. 2011: 2375-2383.

[53] Tang Y, Yang G, Ding D, et al. Multi-level Amplified Iterative Training of Semi-Supervision Deep Learning For Glaucoma Diagnosis[C]//2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2021: 747-750.

[54] Tanha J, Van Someren M, Afsarmanesh H. Semi-supervised self-training for decision tree classifiers[J]. International Journal of Machine Learning and Cybernetics, 2017, 8(1): 355-370.

[55] Ting K M, Zhu Y, Carman M, et al. Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure[C]//Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016: 1205-1214.

[56] Tong X, Jiang Q, Sang N, et al. The Feature weighted FCM algorithm with semi-supervised[J]. Proceedings of Eighth International Symposiumon Distributed Computingand Applications to Business, Engineering and Science, 2009: 22-26.

[57] Vashishth S, Yadav P, Bhandari M, et al. Confidence-based graph convolutional networks for semi-supervised learning[C]//The 22nd International Conference on Artificial Intelligence and Statistics. PMLR, 2019: 1792-1801.

[58] Villa-Pérez M E, Álvarez-Carmona M Á, Loyola-González O, et al. Semi-supervised anomaly detection algorithms: A comparative summary and future research directions[J]. Knowledge-Based Systems, 2021, 218: 106878.

[59] Wang Y, Xu X, Zhao H, et al. Semi-supervised learning based on nearest neighbor rule and cut edges[J]. Knowledge-Based Systems, 2010, 23(6): 547-554.

[60] Wei Z-Q, Bi H-X, Liu X. A graph-based semi-supervised polsar image classification method using deep convolutional neural networks[J]. ACTA ELECTONICA SINICA, 2020, 48(1): 66.

[61] Wei Z, Wang H, Zhao R. Semi-supervised multi-label image classification based on nearest neighbor editing[J]. Neurocomputing, 2013, 119: 462-468.

[62] Wu D, Luo X, Wang G, et al. A highly accurate framework for self-labeled

semisupervised classification in industrial applications[J]. IEEE Transactions on Industrial Informatics, 2017, 14(3): 909-920.

[63] Wu D, Shang M, Luo X, et al. Self-training semi-supervised classification based on density peaks of data[J]. Neurocomputing, 2018, 275: 180-191.

[64] Wu D, Shang M, Wang G, et al. A self-training semi-supervised classification algorithm based on density peaks of data and differential evolution[C]//2018 IEEE 15th international conference on networking, Sensing and Control (ICNSC). IEEE, 2018: 1-6.

[65] Xia S, Peng D, Meng D, et al. A fast adaptive k-means with no bounds[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020: 1-1.

[66] Xu J, Han J, Xiong K, et al. Robust and sparse fuzzy k-means clustering[C]//IJCAI. 2016: 2224-2230.

[67] Xuewen L, Jikui W, Zhengguo Y, et al. Self-Training Algorithm with Editing Direct Relative Node Graph-DRNG[J]. Computer Engineering and Applications, 2022: 1-14.

[68] Yang L, Zhu Q, Huang J, et al. Adaptive edited natural neighbor algorithm[J]. Neurocomputing, 2017, 230: 427-433. 10.1016/j.neucom.2016.12.040.

[69] Yasunori E, Yukihiro H, Makito Y, et al. On semi-supervised fuzzy c-means clustering[C]//2009 IEEE International Conference on Fuzzy Systems. IEEE, 2009: 1119-1124.

[70] Yuan Y, Li X, Wang Q, et al. A semi-supervised learning algorithm via adaptive laplacian graph[J]. Neurocomputing, 2021, 426: 162-173.

[71] Zhang H, Han Z, Li C. Support Vector Machines[D]. 2002.

[72] Zhang R, Che T, Ghahramani Z, et al. Metagan: An adversarial approach to few-shot learning[J]. InICML, 2018, 31: 2.

[73] Zhang S, Li X, Zong M, et al. Learning k for knn classification[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2017, 8(3): 1-19.

[74] Zhao S, Li J. A semi-supervised self-training method based on density peaks and natural neighbors[J]. Journal of Ambient Intelligence and Humanized Computing, 2021, 12(2): 2939-2953.

[75] Zhao S, Li J. ELS: A Fast Parameter-Free Edition Algorithm With Natural Neighbors-Based Local Sets for k Nearest Neighbor[J]. IEEE Access, 2020, 8:

123773-123782. 10.1109/access.2020.3005815.

[76] Zhao W, Krishnaswamy A, Chellappa R, et al.: Discriminant analysis of principal components for face recognition, Face Recognition: Springer, 1998: 73-85.

[77] Zhou M, Li Y, Lu H, et al. Semi-Supervised Meta-Learning via Self-Training[C]//2020 3rd International Conference on Intelligent Autonomous Systems (ICoIAS). IEEE, 2020: 1-7.

[78] Zhou X, Belkin M: Semi-supervised learning, Academic Press Library in Signal Processing: Elsevier, 2014: 1239-1269.

[79] Zhu X, Goldberg A B. Introduction to semi-supervised learning[J]. Synthesis lectures on artificial intelligence and machine learning, 2009, 3(1): 1-130.

[80] 杨艺, 蒋良孝, 李超群. 一种基于自训练的众包标记噪声纠正算法[J]. 自动化学报, 2023, 49(4): 1-15.

[81] 张浩然, 韩正之, 李昌刚. 支持向量机[D]. 2002.

[82] 张倩, 李明, 王雪松. 基于密度分布的半监督回归算法研究[J]. 工矿自动化, 2012(3): 2.

[83] 赵芳, 马玉磊. 自训练半监督加权球结构支持向量机多分类方法[J]. 重庆邮电大学学报:自然科学版, 2014.

[84] 赵志凯, 钱建生, 程健, et al. 基于流形正则化的多元时间序列半监督回归[J]. 中国矿业大学学报, 2011, 40(3): 7.

后 记

光阴似箭，日月如梭，三年的硕士研究生求学即将结束。回想期间的学习和生活，面对培育我的母校，心中无限感慨。时光匆匆如流水，转眼便是毕业时节，春梦秋云，聚散真容易。离校日期已日趋临近毕业论文的完成也随之进入了尾声。从开始进入课题到论文的顺利完成，一直都离不开老师、同学、朋友给我热情的帮助，在这里请接受我诚挚的谢意!

本学位论文是在我的指导老师聂飞平、王继奎老师的亲切关怀与细心指导下完成的。从课题的选择到论文的最终完成，老师始终都给予了细心的指导和不懈的支持，并且在耐心指导论文之余，老师仍不忘拓展我的文化视野，让我感受到了文学的美妙与乐趣。值得一提的是，老师宅心仁厚，闲静少言，不慕荣利，对学生认真负责，在他的身上，我可以感受到一个学者的严谨和务实，这些都让我获益匪浅，并且将终生受用无穷。毕竟“经师易得，人师难求”，希望借此机会向老师表示最衷心的感谢!

此外，本文最终得以顺利完成，也是与学院其他老师的帮助分不开的，虽然他们没有直接参与我的论文指导，但在开题时也给我提供了不少的意见，提出了一系列可行性的建议，他们是李强老师、李兵老师、米红娟老师、何江萍老师、杨正国老师、易继海老师、张克宏老师、李琰老师、丁晓阳老师、杨海军老师等，在此向他们表示深深的感谢!

最后，感谢曾经教育和帮助作者的所有老师。

攻读硕士学位期间发表的论文及科研情况

发表论文：

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

[2] 王继奎，杨正国，刘学文，易纪海，**李冰**，聂飞平*. 一种基于极大熵的快速无监督线性降维方法[J]. 软件学报，2023, 34(4): 1779-1795.

[3] Jikui Wang , Zhengguo Yang , Xuewen Liu , **Bing Li**， Jihai Yi, Feiping Nie*，Projected Fuzzy C-Means with Probabilistic Neighbors，Information Sciences，607(2022), 553-571.

[4] 刘学文, 王继奎*, 杨正国, **李冰**, 聂飞平，密度峰值隶属度优化的半监督 Self-Training 算法，计算机科学与探索，2022, 16(9): 2078-2088.

[5] 刘学文, 王继奎*, 杨正国, 易纪海, **李冰**, 聂飞平，近亲结点图编辑的 Self-Training 算法[J]. 计算机工程与应用，2022, 58(14):144-152.

[6] 刘学文，王继奎*，杨正国，李强，**李冰**，聂飞平，密度峰值优化的球簇划分欠采样不平衡数据分类算法，计算机应用, 2022, 42(5):9.

参与课题：

1.《图优化降维和聚类融合学习模型与算法研究》，项目编号：GZU-PBD2021-101

2.《贫困地区易返贫人群特征提取和返贫风险预测研究》，项目编号：21JR11RA132

3.《基于原型树的无监督学习方法与大数据应用研究》，项目编号：2019B-97

4.《统一的降维和聚类学习模型与算法研究》，项目编号：Lzufe2020B-010

5.《鲁棒的半监督多标签分类算法研究》，项目编号：2021B-147

6.《基于图框架的无监督线性降维算法研究》，项目编号：Lzufe2020B-011