南京大學

研究生毕业论文
(申请博士学位)

论文题目 基于神经网络的增量与开放式机器学习算法研究

作者姓名 徐百乐

专业名称 计算机科学与技术

研究方向 人工智能

指导教师 申富饶教授

年 月 日

学　　号: DG1633021

论文答辩日期: 2023 年 8 月 21 日

指 导 教 师: (签字)

A Research of Incremental and Open Machine Learning Algorithms based on Neural Networks

by

Baile Xu

Supervised by

Professor Furao Shen

A dissertation submitted to

the graduate school of Nanjing University

in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science and Technology

Department of Computer Science and Technology

Nanjing University

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于神经网络的增量与开放式机器学习算法研究

计算机科学与技术 专业 2016 级博士生姓名：徐百乐

指导教师（姓名、职称）：申富饶教授

摘　要

人工智能研究的出发点之一是使用计算机系统模拟人类的学习、推理、决策等智能行为。近年来，机器学习技术作为人工智能研究领域中使用的主要工具取得了飞速的发展，然而机器学习与人类的学习模式却存在巨大的区别。与封闭的机器学习范式不同，人类的学习是一个终生的、开放的过程。我们可以将人类学习的这种特点归纳为“开放性”和“增量性”。“开放性”是指人类思考问题的方式是开放的，在面对未知问题和学习未知事物时，能够利用过去学过的知识对问题进行解读；“增量性”是指人类在一生当中不断地积累新知识，而且能够将新的知识融入自身知识体系中，而不会对自身体系形成破坏。

本文研究面向真实动态环境的开放增量式学习算法，使智能系统突破传统学习模式的限制，在一定程度具备和人类的学习过程类似的开放性和增量性。本文研究无监督学习环境与监督学习环境两种不同情境下的增量开放式学习问题，具体的研究工作包括：

1. 在无监督学习环境中，本文以自组织增量学习神经网络 SOINN 的架构为基础，提出一种 DenSOINN 算法来解决无监督的增量开放式学习问题。DenSOINN 采用了一种自适应距离度量来对数据流进行增量式的规范化，并提出了一种结合密度连通方法与密度峰值方法的聚类算法，使用这种方法解决了 SOINN 类模型中的聚类分割难题。
2. 对于监督学习环境下的类增量学习问题，本文重点研究如何在限制了机器学习系统复习历史样本的情况下进行增量学习。本文分析了现有的增量学习算法面临的挑战，着重分析了目标抑制现象与开放空间风险两项主要问题。针对目标抑制现象提出了任务分解的方法，针对开放空间风险提出了

基于原型的深度神经网络模型，结合两项方案设计了基于原型的网络加固算法 PBNC。

3. 面向监督学习中的开放性问题，本文提出了一种基于对比学习的开集识别算法 ConOSR。本文设计了一种新型的对比表征学习算法 SupCon-ST，使对比学习算法能够与软目标相结合，扩大了对比学习方法的应用范围。在此基础上设计了一种对比开集识别方法 ConOSR，使用对比学习方法提升了深度神经网络的特征学习质量，在开集识别问题上取得了良好效果。

关键词：神经网络；增量学习；聚类；开集识别

南京大学研究生毕业论文英文摘要首页用纸

THESIS: A Research of Incremental and Open Machine Learning Algorithms based on Neural Networks

SPECIALIZATION: Computer Science and Technology

POSTGRADUATE: Baile Xu

MENTOR: Professor Furao Shen

ABSTRACT

One of the starting points of artificial intelligence research is to use computer systems to simulate human intelligent behaviors such as learning, reasoning, and decision-making. In recent years, machine learning technology has achieved great improvements as the main tool of artificial intelligence research. However, there is a significant difference between machine learning and human learning models. Human learning is a lifelong, open process, but the traditional machine learning paradigm is a closed process process. We can summarize this characteristic of human learning as "openness" and "incremental". "Openness" means that the way humans think about problems is open. When facing unknown problems and learning unknown things, they can use the knowledge they have learned in the past to interpret the problems. "Incremental" means that human beings continue to accumulate new knowledge throughout their lives, and can integrate new knowledge into their own knowledge system without damaging the system.

This paper studies the open and incremental learning algorithms in the dynamic environments, which enables the intelligent system to break through the limitations of the traditional machine learning methods, and mimics the openness and incrementality of the human learning process to a certain extent. We study the problem of open and incremental learning in two different contexts, unsupervised and supervised learning. The main work of this paper includes:

1. In the unsupervised learning environment, this paper proposes a DenSOINN algorithm to solve the problem of unsupervised incremental open learning

based on the structure of SOINN, a self-organizing incremental learning neural network. DenSOINN adopts an adaptive distance metric to normalize the data stream incrementally, and proposes a clustering algorithm combining density connectivity method and density peak method, which solves the clustering segmentation problem in SOINN class model.

2. For the class incremental learning problem in supervised learning environment, this paper focuses on how to conduct incremental learning under the condition of restraining the learning system from rehearsing old samples. This paper analyzes the challenges faced by the existing incremental learning algorithm, focusing on two main problems: target inhibition and open space risk. Aiming at the phenomenon of target suppression, a task decomposition method is proposed, and a prototype-based deep neural network model is proposed for the risk of open space. Combined with two schemes, a prototype-based network reinforcement algorithm PBNC is designed.
3. For the problem of supervised learning in open space, this paper proposes an open set recognition algorithm ConOSR based on comparative learning. A new contrastive representation learning algorithm SupCon-ST is designed, which can combine the contrastive learning algorithm with soft learning targets and expand the application scope of the contrastive learning method. On this basis, a comparative open set recognition method ConOSR is designed, which uses contrastive learning to improve the quality of representation learning, and achieves good results in benchmark open set recognition experiments.

Keywords: Neural Networks; Incremental Leaning; Clustering; Open Set Recognition

目　　录

插图目录

表格目录

第一章　绪论

1.1　研究意义

自从人工智能的概念诞生以来，人工智能就一直是计算机科学中的热门和前沿研究领域。人工智能学科的基本研究目的是使用计算机构建人工系统，从而模拟人类的学习、推理、决策等智能行为。近年来，机器学习技术是人工智能研究领域中使用的主要工具，并且在数据挖掘、计算机视觉、自然语言处理、机器人智能等计算机科学研究中作为核心方法发挥了巨大的作用，衍生出的相关成果被广泛应用在自然科学与社会科学的方方面面，已经成为了信息化社会的基石之一。然而，当回归人工智能研究的出发点来观察目前主要的机器学习技术，会发现机器与人类的学习模式存在相当大的区别。

目前主流的机器学习方法是面向某一特定的问题设计模型和学习算法，让模型通过学习从问题相关的统计数据中采样的训练数据集来优化其中的参数，并将训练得到的模型部署到具体的智能系统中，从而应用于解决真实世界中的问题。这种应用方式使机器学习模型仅仅作为解决的某一特定实际问题的工具，而不需要考虑其他任何相关信息，同时在训练完成后的部署应用过程中模型接收的输入对象也被限定在了这一特定问题的范围之中。这种传统的机器学习方式被称为“孤立学习”[1]或者“封闭学习”[2]。

另一方面，人类的学习是一个终生的、开放的过程。人类从出生开始就以自己的方式去观察并尝试理解周围的世界，从开放的环境中学到知识和技能。人类能够充分利用过去的学习经验来学习未曾接触新事物，将其纳入自身的知识体系之中，在遇到新问题时也能够活用已经掌握的知识来进行解读，尝试解决方案。人类学习的这种特点可以被归纳为“开放性”和“增量性”。“开放性”是指人类思考问题的方式是开放的，在面对未知问题和学习未知事物时，能够利用过去学过的知识对问题进行解读；“增量性”是指人类在一生当中不断地积累新知识，而且能够将新的知识融入自身知识体系中，而不会对自身体系形成破

坏。

一个简单的例子可以描述这两类学习模式的区别。首先，如果一个幼童和一个机器分类模型分别需要学会认识猫和狗这两种动物，此时幼童只需几张动物照片就能学会猫狗之间的区别，而机器学习模型则需要使用成千上万张图片作为训练样本；当这项学习任务完成后，如果给儿童展示一张兔子的照片，那么儿童能够分辨出照片上的动物并非猫和狗，但机器则会将其识别为其中之一；最后，如果儿童被告知新照片上的动物是兔子，那么他就学会了辨认这种新的动物，但机器却不能仅靠学习兔子的新训练样本来完成三种动物的分类任务，而是需要将猫、狗、兔的图片混合在一起重新进行学习。这种区别背后的原因是，儿童在进行这项学习任务之前已经在成长过程中具备了观察世界的能力，能够从所观察的图像中捕捉到含有对象关键特征的视觉词汇，进而将观察对象和自己大脑中对世界的认知体系相互匹配。而使机器具备类似的认知体系在目前人工智能研究的发展阶段是一个非常困难的任务，需要极多的训练数据量、模型容量作为支撑，训练消耗的算力开支也远非一般机构所能负担。

综上所述，尽管传统的机器学习方法能够很好地完成了面向特定实际问题的任务，这种封闭的学习方式依然限制了它们在现实世界中适应动态环境的能力。例如分类模型只学到了区分一类识别对象和其他类别的关键信息，这种信息并不能用来识别新的类别；而如果想要让机器学会认识新的类别，需要混合新旧类别的训练样本同时进行训练，否则就会破坏模型对旧类别的分类能力。当在真实环境进行预测时，测试数据类型、概率分布很可能与训练环境不符，算法的性能难以得到保证。研究开放增量式的机器学习算法也符合人工智能应用的发展需求。智能系统的用户希望模型可以增量式地学习在部署应用阶段产生的新业务数据，从而无需时常统合所有的业务数据重新训练模型；智能机器人的使用者希望机器人能够适应其所处的环境，拥有自行观察和学习的能力，可以在运行过程中变得越来越聪明。因此，本文研究面向真实动态环境的开放增量式学习算法，使智能系统突破传统学习模式的限制，在一定程度具备和人类的学习过程类似的开放性和增量性，从而适应动态发展的真实环境，这项研究工作具有重要的研究意义和应用价值。

1.2 相关研究领域

面向上文所介绍的研究背景，研究者从机器学习研究的早期阶段就已经开始关注此类问题，尤其是在近几年的深度学习研究中，面向增量性与开放性的研究课题获得了更多关注。另外，除去这两项特点以外，本文的研究问题与传统的机器学习问题并没有太大的区别。因此在介绍本文的研究工作之前，本节先简要回顾一些传统的机器学习问题范式和与本文研究内容相关的研究领域。

1.2.1 传统机器学习范式

传统的机器学习范式中，最主要的两类是监督学习 (Supervised Learning) 和无监督学习 (Unsupervised Learning)。区分者两类学习范式的依据是模型在学习过程中是否获得了来自外界的指导，即其学习的训练样本是否包含标签。

监督学习 监督学习使用的训练数据集包含数据样本 X 与标签 Y 两部分，其学习目的是利用训练数据样本来学习出一种从样本属性的定义域到标签值域的函数映射关系 f，使得该映射关系能够根据输入的测试样本 x 中的数据预测其标签 y。有监督学习的训练过程是优化模型参数以最小化 $f(x)$ 和 y 之间的误差。监督学习包含了多种子问题，它们决定了标签 y 的形式。其中最基本的是分类问题和回归问题。在传统的监督学习范式里，y 的取值范围在整个训练过程和后续预测过程中是一致的，而且所有训练样本在训练初期即可获得。本文的后续章节中介绍了多个开放与增量环境下的监督学习问题，其中以加入各种预设条件的分类问题为主。

无监督学习 无监督学习的学习对象是无标记信息的数据样本。其学习目的是获取数据内在的本质特征和规律，将其展示给训练者观看或应用于后续的下游任务。传统无监督学习的主要学习任务包括聚类[3]、降维[4]、密度估计[5]和数据可视化[6]等等。近年在深度学习领域则重点关注生成模型[7]和自监督表征学习[8]等问题。无监督学习各种子问题在问题定义与学习目标上具有比较明显的差别，但它们的共性则是用机器学习模型学习数据中有价值的内在结构。以密度估计算法为例，对于无标签输入数据集 X，密度估计算法要求机器学习模型定义某个概率密度分布 P，使 X 的分布与 P 近似。在传统密度估计算法中，分

布 $\boldsymbol{P}$ 的形式以及其中的超参数需要被预先设定，模型在一次训练过程中即可学习到所有数据的分布。然而，在开放增量式的学习过程中，未知数据的数据分布无法被预先估计，因此模型在学习过程中随时可能遇到不符合其参数定义的数据集合。本文的研究内容涉及了无标签数据的增量与开放式学习问题。

1.2.2 神经网络与表征学习

神经网络是以人工神经元与神经链接为基础的机器学习模型，它将信息分布储存于网络内的神经元中[9]。神经网络的关键优势之一是它们拟合非线性函数的能力，通过非线性的神经元激活函数与多层神经元的堆叠，神经网络具备充分逼近复杂非线性函数的能力。

神经网络通过学习输入数据而进行函数拟合的过程被称为训练[10]。网络根据训练数据调整神经元之间连接的权重，通过最小化衡量网络预测输出和实际输出之间差异的代价函数，模型可以随着训练过程的推进逐渐提高性能。这个过程通常使用优化算法如梯度下降来进行，它迭代地调整权重来最小化代价函数。随着近年来神经网络架构和训练算法的进步，以深度神经网络为基础的深度学习研究领域发展格外迅速。深度神经网络由许多层相互连接的神经元组成，在提取复杂数据（如图像、语音和自然语言）的特征方面特别有效。深度神经网络在目前许多人工智能应用领域取得了最佳结果，包括目标识别、语音识别、机器翻译等。

相对于传统机器学习模型而言，深度神经网络的优势在于其进行表征学习的能力。表征学习是机器学习研究领域中的重要课题之一[11]。当人类对事物进行描述时，通常只会选择性地描述对它的少量关键特征，而不是对事物的细节进行充分的详细描述。因为事物的关键信息通常隐含在其表面的大量无用信息之下。这反映了人类认识世界与的朴素方式，即归纳和演绎。人类会尝试从含有大量原始属性的数据集合中归纳出如何描述事物的关键特征，并使用这种描述方法来认识新的事物。表征学习在机器学习中非常重要，因为它作为一种基本技术，可以将变量的使用引导到对于给定机器学习系统最有效和最有效的内容。除了较为直观的表格类数据之外，原始的样本属性常常是高维、复杂、缺少可解释性的，无法形成对目标的有效描述。表征学习的意义在于从原始数据中提取出能够有效描述事物的特征，从而使基于这些特征训练的机器学习模型能够掌

握不同样本之间的语义区别，从而更准确地预测新的未见数据。本文的研究过程中采用了神经网络模型作为主要的研究工具，研究在面向不同的问题如何设计神经网络模型与训练方法，使神经网络能有效地对数据进行表征，完成学习任务。

1.2.3 增量学习

增量学习 (incremental learning) 是指当新的训练数据被输入到学习系统时，算法能够使用新数据快速更新现有模型，从而使模型在保持现有记忆的同时学习到新数据中心蕴含的知识，而不需要重新训练整个样本集[1][12]。根据定义，增量学习的学习样本具有很强的时序性，因此增量学习经常被用于处理大规模流数据。

增量学习通常要求在学习新的数据样本时，不能完全“遗忘”先前学习到的有效信息。显然，人的学习任务具有增量式的特性：人通过感官系统获得感知数据，因此学习样本是以人的感知顺序被输入到大脑当中，同时人不能长时间记忆所有感知到的样本数据。

增量学习可以继续细分为以下类型的学习任务[13]：第一类任务中的数据样本具有增量性，但样本类别、学习任务的目标都是固定的，这一类任务被称为“样本增量”或“域增量”；第二类任务中的数据样本和样本类别都具有增量性，样本类别随新样本的增量输入而增加，但所有类别都属于同一个分类任务，这一类任务被称为“类增量”；第三类任务则要求学习者能够针对新的问题进行学习，学习任务在不同的训练阶段随着样本的输入不断变化即“任务增量”。

增量学习的应用广泛。当下的互联网、金融、传感器监控等应用领域会实时产生大量的数据流，对这些数据流进行实时挖掘是一类有价值的数据挖掘任务，其中就需要用到增量学习技术。近年新兴的 Bert[14]、GPT[15-16]等大语言模型被广泛使用，它们的参数量极其庞大，训练代价高昂。因此很多最新的研究工作关注如何对大模型进行增量式微调。例如，GPT-3[16]中首先使用了 Prompt-Tuning 训练机制，在微调预训练模型时固定模型权重，只针对新的任务和数据训练和更新提示词的相关的一小部分参数。Low Rank Adaption[17]只通过微调一个低秩矩阵即可使模型适应新增的任务和数据，从而节省微调的时间和内存消耗，而且可以与 Prompt-Tuning 共同使用，也是实现大模型增量微调的常用技术之一。

1.2.4 开集识别

开集识别 (Open Set Recognition)[18]是模式识别和机器学习领域中的一个新兴的研究问题。因为在实际场景中，输入数据可能包含在训练过程中未见过的类别样本，因此需要识别系统能够适当地识别和处理这些实例。开集识别旨在处理测试过程中出现的新颖类别或未知类别，而传统的闭集 (Closed Set) 识别则仅能将输入分类到预定义的类别集合中。增量学习体现了监督学习中训练过程的开放性，而开集识别则体现了测试过程的开放性。

开集识别的动机源于闭集识别的局限性[2]，闭集识别假设测试数据仅由来自已知类别的测试样本组成。在实际应用中，这个假设往往是不现实的，因为模式识别系统在部署运行之后无法保证输入到系统中的样本全部属于已知的类别。属于新类别的测试样本在实际应用中随时可能会出现，要求识别系统能够有效地处理它们。开集识别在各个领域中都有应用，包括人脸识别、网络安全、语音识别和异常检测。考虑一个经过特定个体训练的人脸识别系统，在部署过程中，系统很有可能会遇到不属于训练集的个体面孔。在这种情况下，开集识别方法变得至关重要，能够将这些未知面孔识别为离群值或新颖类别，而不是错误地将它们分配给已知类别。

在此基础上，开集识别领域的研究者提出了一种被称为“开放世界识别”的新问题[19-20]。开放世界识别是指在一个动态变化的数据集中，能够不断地检测并添加新的类别，同时在预测时能够处理大量未见过的类别。这一问题可以被看做将开集识别问题与类增量学习问题进行了融合，对识别算法提出了更高的要求。

1.3 增量与开放式学习

本文中所研究的的“增量与开放式学习”，与传统的机器学习范式相比，它们具有一些鲜明的特点，需要关注的问题、克服的障碍也与传统学习范式有明显的区别。相比而言，更接近人类学习的特点。本节先从认知理论出发讨论人类学习为何具备增量性与开放性的，然后介绍机器学习系统中增量与开放式学习的特点。

1.3.1 人类认知发展理论

认知心理学家让. 皮亚杰提出了认知图式 (scheme) 理论分析人类的认知发展。图式理论认为儿童通过一系列的阶段来构建和发展对世界的知识。皮亚杰将图式定义为关于世界各方面的基本知识单元，也是获取知识的过程。图式可以是物体、行为、事件、概念等的心理表征，也可以是对这些表征进行操作和变化的规则和策略。例如，儿童学习认识狗时，需要认识狗的体型特征、生活习性、典型行为等，这就是幼儿对狗的图式。图式是动态的，它们会随着经验的增加而不断地调整和更新。

皮亚杰认为人类通过适应（adaptation）的过程来改变和发展他们的图式。适应包括两个方面：同化（assimilation）和调节（accommodation）。同化是指将新的信息或经验纳入现有的图式中，使之与自己的认知结构相一致。同化不会改变现有的认知结构，而是直接将新的知识纳入，对图式起到巩固的作用。调节是指根据新的信息或经验来修改或创建新的图式，使之与外部环境相适应。当现有的认知不能解释新的信息时，就需要改变认知以适应，即通过调整图式进行调节；如果难以通过调整图式达成调和，则重新构造图式。

例如，儿童有了一个关于狗的认知图式，当他看到同样四足行走的动物时，就会将其看做是狗，这属于同化的范畴。当他后来第一次认识猫的时候，他建立一个新的关于猫的认知图式，并且调整他对于狗的认知图式以便于区分两种动物，这属于调节的范畴。

人类的认知发展过程即是自身图式认知结构的进化过程，即不断地通过同化与调节两种方式使认知发展从一个平衡状态过渡到另一个平衡状态。一方面，人类不会盲目地用自己的图式同化一切外界事物，而是能够发现自身认知结构不能解释的外界事物，这体现了人类认知发展的开放性。另一方面，人类不断地使用通过调节系统使自身的认知结构产生变化以适应外界输入的信息，这体现了人类认知发展的增量性。

1.3.2 增量与开放式机器学习

仿照人类认知系统，本文所研究的增量与开放式学习范式具有如下的特点：

1. 训练数据以数据流的形式输入到学习系统，学习系统无法存储全部的数据

流，也无法控制数据流的输入顺序。

2. 属于新类别的数据样本可能会在学习过程中的任何时刻出现，学习系统无法预先知道有多少种不同类别的数据将要被学习。
3. 学习系统能够在学习进行的任何时刻对输入的测试数据进行预测，测试数据可能属于它学习过的任何一类，这要求学习系统在学习新信息时不能遗忘已学习到的旧知识。
4. 测试数据也可能属于未曾学习过的全新类别，这要求学习系统辨别属于全新类别的测试数据。

由此，本文给出一种形式化的描述。有监督的增量开放式学习，是对于输入数据流

$$S = \left\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_n, y_n), \cdots\right\}, \boldsymbol{x}_i \in \mathcal{R}^d, y_i \in \{0, 1, 2, 3, \cdots, \}, \tag{1.1}$$

学习到一种从数据空间 $\mathcal{R}^d$ 到标签集合 Z^+ 的映射关系 f。其中，d 表示输入样本的特征维度，y_i 是样本 x_i 的标签。在学习的任何时刻，映射关系 f 能够对测试样本 $\boldsymbol{x}_t$ 进行预测，得到预测标签 $y_t \in \{-1, 0, 1, 2, 3, \cdots, k\}$，其中 k 是学习系统已经学习过的类别数，类标 $y_t = -1$ 则代表学习系统判定 $\boldsymbol{x}_t$ 不属于已知的任何一类。

同理，本文给出无监督环境下的增量开放式学习范式的形式化描述。对于无标签数据流

$$S = \left\{(\boldsymbol{x}_1), (\boldsymbol{x}_2), \cdots, (\boldsymbol{x}_n), \cdots\right\}, \boldsymbol{x}_i \in \mathcal{R}^d \tag{1.2}$$

机器学习模型学习出一种从数据集 $\mathcal{R}^d$ 标签集合 Z^+ 的映射关系 f，使得被映射到同一类标 (即聚类) 下的样本尽可能相似，被映射到不同类标的样本尽可能不相似。其中标签集合中的元素数量、即聚类数 k 未知，学习系统能够在学习过程中跟随样本分布的变化自主地调整 k。无监督的增量开放式学习要求学习者自主对观察到的各类事物特征样例进行归类或区分，从而不断学习到新的事物。

上述学习范式的增量性体现在训练数据被组织为数据流的形式，学习系统在任意时刻都只能学习一小部分、甚至仅单个训练样本。而学习过程中随时可能进入测试阶段，此时要求学习系统根据过去所学习过的所有知识进行预测，而不是仅测试最近学习过的知识。开放式学习在监督学习和无监督学习中有不同

的体现。监督学习环境下，增量学习范式要求模型具有可动态扩展的开放性，但这仅体现了训练阶段的开放性；测试阶段的开放性则体现在测试数据的真实类别可能是训练过程中没有出现过的，要求学习系统在预测时发现没有学习过的未知样本。无监督学习问题中通常没有独立的测试阶段，因此增量学习同样也是开放式的学习，这种开放性体现在学习系统中预设的重要先验参数能够在学习过程中被后验地修正，例如不预先假定数据的分布与潜在聚类的数量，使模型能够跟随增量学习的进展而动态扩展。

1.4 本文的主要贡献与创新点

本文以基于神经网络的机器学习算法作为主要工具，面向增量与开放式学习问题展开研究。文中研究主要按如下基本顺序进行：首先，从无监督学习环境开始，研究无标签数据流的增量与开放式学习问题。然后，在监督学习环境中，研究神经网络在增量学习中必须要面对的灾难性遗忘现象。最后，面向监督学习环境中的开放性问题，深入研究神经网络如何应对开放空间风险。

在无监督增量学习环境中，数据流被组织成序列的形式依次输入，学习系统的学习任务是归纳数据流中存在的聚类。本文在这一问题上的研究思路为：设计一个竞争神经网络模型，使模型中的神经元节点与节点之间的连接形成对数据分布的归纳描述，随后在神经网络的数据结构上使用传统的聚类算法生成聚类结果。具体来说，在无监督增量与开放式学习方面，本文提出了一种竞争神经网络模型用来生成数据流的可信代表集，并提出了一种基于密度的聚类算法来对竞争神经网络的数据结构进行有效地分割，同时解决了聚类数目的自动确定问题。面向聚类中的距离计算，本文提出了一种自适应的距离度量，用来在增量学习环境下对度量函数进行动态优化。

在有监督增量与开放式学习方面，本文分析了学习系统需要克服的障碍，即灾难性遗忘与开放空间风险两项问题。面向类增量学习问题，本文以严格的类增量学习范式作为研究问题，限制了主流研究中常用的复习法，分析实现类增量学习所需解决的关键问题，提出了一种基于原型与网络加固的增量学习算法。最后，本文重点关注监督学习的开放式预测过程，提出了一种基于对比学习的开集识别算法来提升神经网络抵御开放空间风险的能力。

本文的创新点归纳如下：

1. 面向聚类过程中的样本间距离计算，本文提出了一种自适应距离度量框架，可以在增量学习过程中并行优化度量函数与模型参数。本文提出了一种能够动态实现数据白化或数据规范化的度量函数优化算法，其中包括了协方差矩阵逆矩阵的增量式计算方法。
2. 面向无监督增量与开放式学习问题，本文提出了开放式竞争型神经网络 DenSOINN，其网络结构在学习过程中动态构建。本文提出了一种基于密度的聚类方法来对 DenSOINN 的神经元进行聚类，提供了用竞争型神经网络处理聚类问题的一种有效方案。
3. 面向有监督增量学习中的类增量学习，本文分析了类增量学习范式中存在的两项关键问题，即目标抑制与开放空间风险，并分别提出了网络分解、基于原型的分类两项针对性的措施，并与知识蒸馏、Focal Loss 等网络训练方法综合成类增量学习算法 PBNC。
4. 本文提出了一种新的对比学习框架 SupCon-ST，实现了对比学习与实数标签向量的结合，从而让对比学习算法能够与标签平滑、Mixup、知识蒸馏等神经网络训练方法结合。
5. 面向开集识别问题，本文提出了一种基于对比学习的框架 ConOSR，通过对比学习提升了深度网络对于图像的表征质量，提高了识别未知样本的能力。

1.5 本文的组织结构

综上所述，本文的研究内容安排如图1.1所示，各章节的设置如下：

第一章为绪论，对本文的研究意义以及相关研究领域进行了介绍，对本文所要研究的问题进行了形式化的描述，随后对本文的主要内容与贡献进行概括。

第二章为相关工作，介绍了神经网络领域中与本文研究内容高度相关的现有研究成果，主要是形成本文研究成果的基础工作。这一章中介绍了作为第三章研究基础的竞争型神经网络与第四、五章研究基础的深度卷积神经网络。

第三章首先从无监督学习环境开始展开本文的研究工作。面向无标签数据流的增量与开放式学习问题，提出了一种基于竞争神经网络与自适应距离度量的增量开放式无监督学习算法 DenSOINN。

第四章面向监督学习环境中的类增量学习问题开展研究。虽然可以采用与

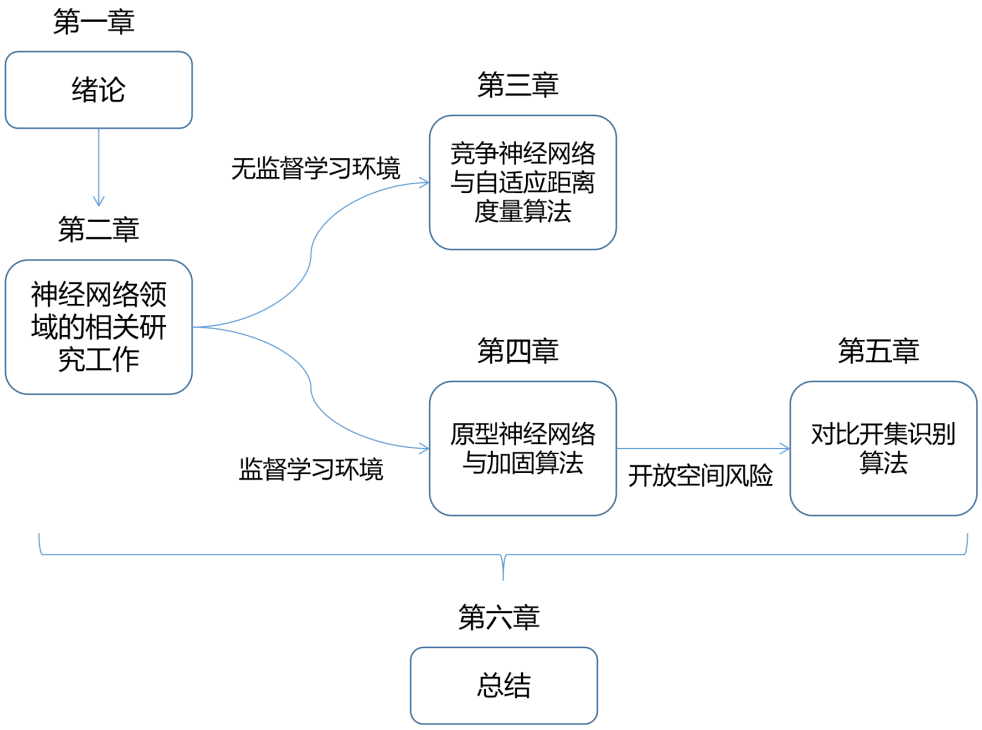


图 1.1　本文内容结构图

无监督增量学习中类似的方法对训练数据提取可信代表集，通过复习可信代表集的方式实现增量学习，但本文将研究重心放在另一个关键问题上，关注神经网络如何在不缓存训练样本的情况下实现增量学习。第四章从理论上分析目标抑制与开放空间风险两项障碍，并且受到第三章中神经元竞争思路的启发，提出了一种基于原型与网络加固的增量学习算法 PBNC。

第五章继续对第四章中涉及的开放空间风险问题进行深入研究。以面向开放式测试环境的开集识别问题作为研究对象，提出了一种有监督的对比表征学习算法 SupCon-ST，并在此基础上设计了一种对比开集识别算法 ConOSR。

第六章为总结和展望，对本文的内容进行了概括性的描述，并展望后续的研究方向。

第二章　相关研究工作

在后续章节中，本文将介绍三种基于神经网络的算法来分别解决无监督环境与监督环境下的增量开放式学习问题。在此之前，本章将详细介绍与本文所研究的神经网络模型相关的现有研究工作。因为上一章介绍了与本文研究内容有关的研究领域，并且后续章节中会在介绍各个研究子问题之后综述相关的研究工作，因此本章并不会从问题出发、在广度上展开来覆盖大量的研究成果，而是重点讲述与本文研究高度相关的少数神经网络模型，从而为不熟悉神经网络领域的读者提供必要的背景知识，以使本文的内容自成体系。

本文的研究工作中主要使用了两类神经网络模型。第一类是自组织竞争型神经网络，本文使用这类神经网络来处理大规模无标签数据，通过训练使网络中的神经元、连接等数据结构形成对于数据分布的压缩表示。这种方式使增量学习的大规模数据流转化为小数据，进而可以在小数据的基础上应用传统的无监督学习方法获取结果。第二类是深度前馈型神经网络，以卷积神经网络为主。在监督学习问题上，本文使用图像数据集为主要实验对象，但因为图像数据是一类较为复杂的数据，传统的机器学习算法需要结合图像特征提取技术才能有较好的效果。深度卷积神经网络则可以实现对图像原始数据的表征学习，而且表征效果好于传统的图像特征提取算法。

2.1　自组织竞争型神经网络

聚类和拓扑学习是无监督学习的两个重要部分，两者都是试图从无标签的数据集中发现隐含的信息。聚类算法的目的是为了挖掘出数据中潜在的全局结构信息，而拓扑学习则关注数据的局部邻域信息，这种局部信息能够在一定程度上反映原始数据在特征空间上的拓扑结构。不同于一般神经网络基于损失函数的反向传递来训练，它们运用竞争学习（competitive learning）策略，依靠神经元之间互相竞争逐步优化网络，训练过程中，所有神经元相互竞争对当前输

入的响应权力，获胜的神经元更新自身参数以适应新的输入。这样的竞争机制导致最终网络收敛后的结果是不同区域的神经元对不同的输入模式更为敏感，一旦该模式出现就会有更大的几率在竞争中被激活。历史上有多种竞争型神经网络模型被提出用于解决聚类和拓扑学习问题，比较有代表性的是 Kohonen 提出的自组织映射网（Self-Organizing Map，简称 SOM）[21]。

自组织映射网络 SOM 是由 Kohonen 于 1982 年提出的一种无监督、自组织、自学习的竞争型神经网络，又称 Kohonen 网。SOM 使用近邻关系函数（neighborhood function）来维持输入空间的拓扑结构。SOM 的目标是用低维（通常是二维或三维）目标空间的点来表示高维空间中的所有点，同时尽可能地保持点间的距离和邻近关系，也即拓扑关系。在接收外界输入模式时，将会分为不同的对应区域，各区域对输入模式有不同的响应特征，而这个过程是自动完成的，其特点与人脑的自组织特性类似。自组织神经网络无需提供标签信息，能够对外界未知环境或样本空间进行学习和模拟，并对自身的网络结构进行适当的调整。

神经生物学研究表明在人的感觉通道上一个很重要的组织原理是神经元有序地排列着，并且往往可以反映出所感觉到外界刺激的某些物理特性。如在听觉通道的每一个层次上，其神经元与神经纤维在结构上的排列与外界刺激的频率关系十分密切，对于某个频率，相应的神经元具有最大的响应，这种听觉通道上的有序排列一直延续到听觉皮层，尽管许多低层次上的组织是预先排好的，但高层次上的神经组织则是通过学习自组织而形成的。自组织映射神经网络作为一种聚类和高维可视化的无监督学习算法，也正是通过模拟以上人脑对信息存储和处理的特点而发展来的一种神经网络，随后发展为应用最广泛的自组织神经网络方法。

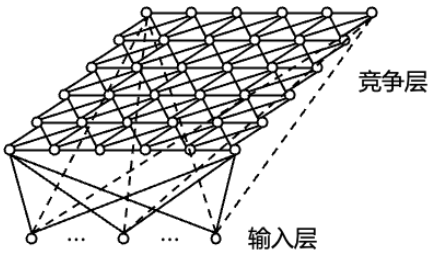


图 2.1 自组织映射网络结构

SOM 人工神经网络是一个可以在一维或二维的处理单元阵列上形成输入信

号的特征拓扑分布，结构如图 2.1所示。网络模拟了人类大脑神经网络自组织特征映射的功能。该网络由输入层和竞争层组成，两层之间各神经元通过双向连接，网络没有隐藏层，有时竞争层各神经元之间还存在横向连接。其中输入层的神经元个数的选取按输入网络的向量个数而定，输入神经元为一维矩阵，接收网络的输入信号，竞争层则是由神经元按一定的方式排列成一个二维节点矩阵。输入层的神经元与竞争层的神经元通过权值相互联结在一起。当网络接收到外部的输入信号以后，竞争层的某个神经元便会兴奋起来。

自组织神经网络的学习过程模拟了生物神经元之间的兴奋、协调、抑制和竞争作用的信息处理的动力学原理。网络通过对输入模式的反复学习可以使权重向量空间与输入模式的概率分布趋于一致，即概率保持性。网络的竞争层的各神经元竞争对输入模式的响应机会，获胜神经元有关的各权重朝着更有利于它竞争的方向调整，即以获胜神经元为圆心，对近邻的神经元表现出兴奋性侧反馈，而对远邻的神经元表现出抑制性侧反馈，近邻者相互激励，远邻者相互抑制。

SOM 网络的学习过程包括四个步骤：一是初始化，对所有连接权重都用较小的随机值进行初始化；二是竞争，对于每种输入模式，竞争层神经元计算他们各自的判别函数值，这里判别函数可以定义为输入向量和每个神经元的权重向量之间的平方欧几里德距离。具有最小判别函数值的神经元宣布为“胜利者”。换句话说，权重向量最接近输入向量（即与其最相似）的神经元即为胜利者。这样，连续的输入空间通过神经元之间的一个竞争过程被映射到神经元的离散输出空间；三是合作，在神经生物学中，一组兴奋神经元内存在横向的相互作用。当一个神经元被激活时，最近的邻居节点往往比那些远离的邻居节点更兴奋，并且存在一个随距离衰减的拓扑邻域。获胜的神经元决定了兴奋神经元拓扑邻域的空间位置，从而为相邻神经元之间的合作提供了基础；四是适应，激活神经元通过适当调整相关的连接权重，减少与输入模式相关的判别函数值，使得获胜的神经元与相似输入模式的后续应用的响应增强。SOM 训练过程的直观可视化效果如图 2.2所示。

图中蓝色部分的区域表示训练数据的分布，白色圆圈表示从该分布中选取的当前训练基准面。可以看出 SOM 节点首先在数据空间中任意位置初始化，然后选取最靠近训练基准的节点（黄色区域），并向该位置移动。经过多次迭代后，

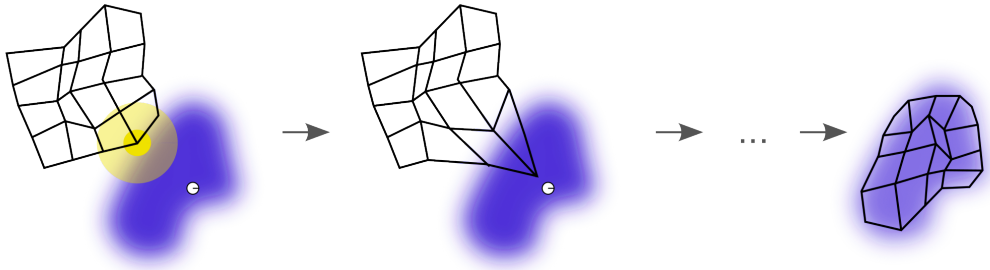


图 2.2　自组织映射网络训练过程示意图

网络会趋近于数据分布。然而，SOM 算法依然存在一些局限性，比如：聚类数目和初始网络结构固定，需要预先设定聚类数目和初始的权值矩阵；可能会出现一些始终不能获胜的“死神经元”，同时会有一些因为经常获胜被过度利用的神经元，这样就不能充分利用所有神经元信息而影响聚类质量；要想往 SOM 网络中加入新的类别必须先完整的重新学习之后才可进行；数据的输入顺序会影响甚至决定了输出的结果，数据量少时尤为明显；连接权值初始值、计算策略，参数选择不当时会导致网络收敛时间过长，甚至不能收敛。

然而，有研究表明这样的竞争神经网络难以在保持可塑性的情况下获得稳定的学习结果，这就是 Grossberg 和 Carpenter 提出的稳定性-可塑性困境[22] (Stability-Plasticity Dilemma)：在试图构造一个能够实时地适应不断变化的环境的自适应学习系统时，如果一个系统过于稳定，则不善于适应快速变化的环境。反之，系统对外界的刺激过于敏感就会导致系统本身没有办法稳定地保存先前学习到的知识，甚至无法收敛到一个稳定的状态。因此，在面对缺乏先验知识且外部输入模式随时间变化的情况时，网络的自组织性和算法的增量性是一个学习系统应该具有的特性。

自组织增量学习神经网络 (SOINN)[23]是在自组织映射网络的基础上结合竞争 Hebbian 学习规则[24]、拓扑表示网络[25]及增量学习等概念发展而来的。它的主要功能是以无监督的方式在线增量式地生成大规模训练数据的可信代表集与节点之间的拓扑结构。SOINN 神经网络及其衍生模型被用于解决聚类[23][26]、分类[27]、密度估计[28][29]、联想记忆[30]、数据可视化[6]等多种机器学习问题。

初始的 SOINN 神经网络[23]包括两层竞争学习层，第一层用于对原始数据输入生成代表神经元节点，而第二层使用第一层的神经元节点作为输入，以同样的方式进行学习得到更加简化的网络结构。后来 SOINN 神经网络被简化为单个竞争层，即 E-SOINN[26]与 Adjusted-SOINN[31]。该单层网络能够对原始输入

数据样本进行在线学习，从而生成作为数据代表集的神经元节点及网络拓扑。

SOINN 神经网络的拓扑结构生成主要依赖于竞争 Hebbian 学习[24]。即对于网络神经元节点集合 $N = \{s_n\}$，当有新的数据样本 x 输入进来时，网络首先寻找与 x 最相似的两个神经元节点 s_i，s_j：

$$s_i = \underset{s \in N}{argmin} \|s - x_t\|_2 \tag{2.1}$$

$$s_j = \underset{s \in N/\{s_i\}}{argmin} \|s - x\|_2 \tag{2.2}$$

如果输入数据样本在两个获胜神经元的激活阈值 th_i, th_j 之内，即：

$$\|s_i - x\|_2 < th_i \quad and \quad \|s_j - x\|_2 < th_j \tag{2.3}$$

则最近邻神经元 s_i 学习输入样本 x，并调整权重和阈值参数，同时更新两个获胜节点之间的连接关系。否则，认为 x 属于一个新的模式，从而形成一个新的神经元节点。以这种方式，网络能够不断在线增长。

对比 SOINN 算法，E-SOINN 仅采用单层神经网络，第二层网络的去除使得 E-SOINN 无需判断何时停止第一层学习和开始第二层学习，更适合在线学习任务。E-SOINN 改进了聚类代表点的表示，引入了节点密度属性，从而判断网络中的哪一部分结构对应了数据分布中不同聚类的重叠区域。当在节点之间建立连接时，E-SOINN 增加了基于密度的条件来判断是否需要连接，并在一定次数的学习迭代后并删除位于密度重叠区域的连接。E-SOINN 也去除了 SOINN 中的一些冗余操作，使得调试算法更为便捷。Adjusted-SOINN 则是面向监督学习设计的简化版的 E-SOINN 模型，保留单层结构的同时去除了节点密度的部分，使学习过程更加简洁。

SOINN 神经网络及早期变体使用欧几里得距离来计算样本特征之间的距离，但是这种距离度量方式在高维空间中往往并不能够有效的表示样本差异程度，因此衍生模型 LD-SOINN（Local Distribution SOINN）[29]对每个神经元节点引入了局部协方差矩阵，用来保留更多的局部数据分布信息，并使用神经元节点的局部区域的 Mahalanobis 距离来计算输入样本与神经元节点之间的距离，

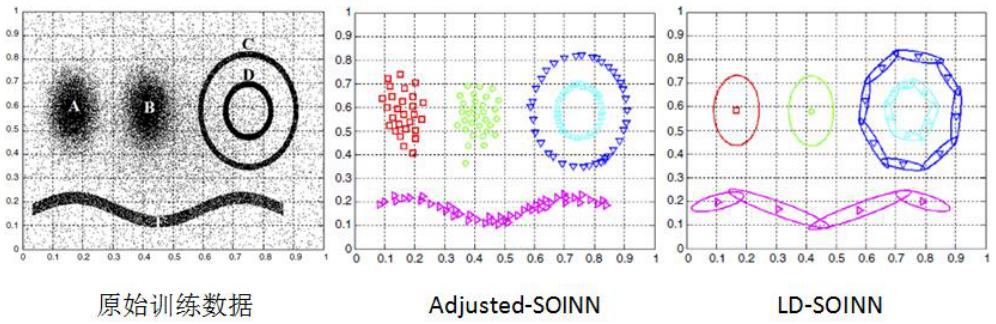


图 2.3　Adjusted-SOINN 与 LD-SOINN 在人工数据集上的实验结果

即：

$$d(x, s_i) = \sqrt{(x - s_i)\boldsymbol{M}_i^{-1}(x - s_i)}, \quad i = 1, 2, \cdots, |N| \tag{2.4}$$

其中，$\boldsymbol{M}_i$ 表示神经元 s_i 局部的协方差矩阵。同时，算法还能够对邻近的并具有相似主成分的神经元节点进行合并，从而获取更加简洁的数据拓扑表示方式。图2.3表示了在人工数据下 Adjusted-SOINN 与 LD-SOINN 的增量式学习结果 (图片来自论文[29])。

SOINN 类神经网络的学习方式符合无监督增量学习的范式，而且其网络结构的自组织性很好地体现了学习系统的开放性。本文第三章以 SOINN 神经网络为基础，提出了基于自适应距离度量与密度聚类的无监督增量开放式学习算法，解决了困扰 SOINN 类模型的聚类分割问题。

2.2　深度神经网络

深度学习是当前机器学习领域的热门研究分支，它使用深度神经网络从数据中学习并执行图像识别、自然语言处理、语音合成等任务。深度神经网络的灵感来自于人脑的结构和功能，但并不是它的精确复制品。与传统机器学习算法相比，深度神经网络可以处理大量复杂的非结构化数据，例如图像、文本、音频和视频，并使用多层网络结构自底向上地从原始数据中提取从简单到复杂的多级特征。

图2.4展示了深度学习与传统机器学习的区别。以图像数据的识别为例，传统方法通过人工设计的算法提取 SIFT[32]、HOG[33]等图像特征，在这些特征的

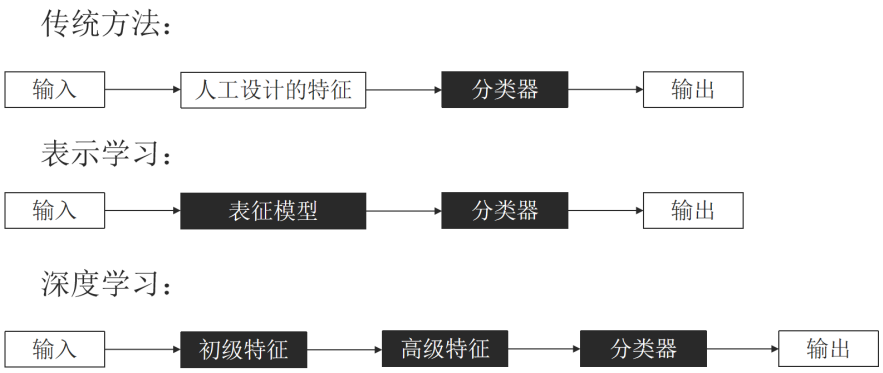


图 2.4　传统机器学习、表征学习与深度学习的区别，图中黑色框代表学习系统中可学习的部分。

基础上用机器学习方法训练分类器。表示学习[11]则是同时使用机器学习方法学习图像特征编码器和分类器。深度学习[34]属于表示学习的一种，能够通过深度神经网络学习出从颜色、形状等低级特征到高级语义特征的多级特征表示。

深度神经网络通常是多层的前馈网络模型，通过计算网络输出与数据标签计算目标函数值，并通过梯度反向传播来优化网络参数，实现目标函数的最大化或最小化。用来优化网络参数的算法有随机梯度下降[35]、动量法[36]、Adam[37]、Amasgrad[38]等等。深度学习研究者通常借助 Tensorflow、pytorch 等深度学习平台来将算法实现为程序代码。这些平台内置了张量运算、求导、计算图等功能，以及基础的神经网络组件与优化器，大幅减轻了研究者的编程工作量。

深度卷积神经网络 (Deep Convolutional Neural Network, DCNN) 是最常用的深度学习模型类型之一，它专为处理视觉数据而设计。DCNN 自下而上有多个网络块组成，每一块又包含卷积层、池化层、批归一化层等多种神经层。

卷积是 DCNN 的核心操作。它是一种使用称为卷积核或过滤器的小矩阵过滤和转换输入数据的方法。卷积操作将卷积核的每个元素与输入数据的一小块同尺寸区域中的对应元素相乘，并将结果相加来产生单个输出值。然后卷积核以一定的步幅（例如一次一个像素）在整个输入数据上滑动，依次与不同区域执行卷积操作，得到一个称为特征图的输出矩阵。

卷积的数学定义如下：令 x 为大小为 $m\times n$ 的输入矩阵，令 w 为大小为 $k\times k$ 的核矩阵。然后，x 和 w 的卷积记为 $x*w$ 并且是一个大小为 $(m-k+1)\times(n-k+1)$ 的矩阵，其中的元素 $(x*w)_{i,j}$ 由下式给出：

$$(x * w)_{i,j} = \sum_{a=0}^{k-1} \sum_{b=0}^{k-1} x_{i+a,j+b} w_{a,b} \tag{2.5}$$

卷积运算可以推广到更高的维度，例如单张 RGB 图像对应的三维张量或一批图像数据对于的四维张量。在这种情况下，内核也具有与输入相同的维数，并且沿着除最后一个维度（对应于通道或特征的数量）之外的所有维度执行卷积。例如，如果有一个大小为 $b \times c \times m \times n$ 的输入张量 x，其中 b 是批量大小，c 是通道或特征的数量，m 和 n 是空间维度，一个大小为 $c \times k \times k$ 的核张量 w，那么 x 和 w 的卷积就是一个大小为 $b \times c \times (m-k+1) \times (n-k+1)$，其中每个元素 (i, j, l, r) 由下式给出：

$$(x * w)_{i,j,l,r} = \sum_{a=0}^{k-1} \sum_{b=0}^{k-1} x_{i,j,l+a,r+b} w_{j,a,b} \tag{2.6}$$

卷积操作的一些特点使它特别适合提取图像特征：

1. 卷积是可叠加的线性操作，因此简单的卷积核能够组合成复杂的卷积核，复杂的卷积核能被分解。
2. 卷积操作满足交换律和结合律，因此输入和卷积核能够互相交换，或是卷积的顺序而不影响结果。
3. 卷积具有平移不变性，因此能够保留输入的空间结构，例如有一个输入 x 和一个内核 w，并且若将输入的元素移动某个量 (s, t)，则有 $(x_{i+s,j+t}) * w = (x * w)_{i+s,j+t}$。这一特点使卷积核能够检测图像中出现的特定特征，而且在输出矩阵中保留特征在输入中的空间位置。

卷积神经网络通常在卷积层之后添加池化层，用于减少特征图的空间维度，同时保留特征通道。池化层的工作原理是对特征图的小区域应用池化操作，并为每个区域生成单个输出值。假设 x 是大小为 $m \times n \times c$ 的输入特征图，其中 m 和 n 是空间维度，c 是通道数。设 f 为池过滤器的大小，s 为步幅。然后，池化后的输出特征图是大小为 $\lfloor \frac{m-f}{s} + 1 \rfloor \times \lfloor \frac{n-f}{s} + 1 \rfloor \times c$ 的张量。当进行最大池化时，输出特征图中每个元素 (i, j, k) 由下式给出：

$$y_{i,j,k} = \max_{a=0}^{f-1} \max_{b=0}^{f-1} x_{si+a,sj+b,k} \tag{2.7}$$

当进行平均池化时:

$$y_{i,j,k} = \frac{1}{f^2} \sum_{a=0}^{f-1} \sum_{b=0}^{f-1} x_{si+a,sj+b,k} \tag{2.8}$$

池化层在卷积神经网络中起着重要的作用。首先，它们降低了特征图的空间维度，从而减少了网络中参数和计算量，提高了效率。其次，它们整合了卷积层提取的特征，这使网络模块对输入中的小变化不太敏感，防止了过度拟合。

卷积神经网络具有多种典型架构,例如 AlexNet[39]、VGGNet[40]、ResNet[41]和 DenseNet[42]等。本文的研究工作中主要使用了 VGGNet 和 ResNet 两种架构。VGGNet 由多个 3x3 卷积层块和最大池化层组成，网络深度从 11 层到 19 层不等，每个卷积层的卷积核数量从 64 个到 512 个。VGG Net 在 2014 年的 ImageNet 挑战赛上取得了最佳成绩，此后作为各种计算机视觉算法与应用中的骨干架构被广泛应用。

然而,VGGNet 与此前的深度网络一样存在一个问题，就是梯度消失现象使网络深度到达一定程度时便无法通过继续增加深度来提升网络性能。论文[41]提出了 ResNet 网络架构，其中首次使用了残差块结构来克服梯度消失问题。残差块在基本的卷积模块之外加入了跨层连接，让残差块的输出 $z = f(x) + x$。残差结构使梯度能沿着跨层连接传播，避免了梯度消失问题，因此网络能够使用非常深的多层架构，而不会出现性能随层数增加而饱和或退化的现象。

本文第四章、第五章的研究工作中使用了深度卷积神经网络来进行图像数据的表征，但并未对深度卷积网络的架构进行优化。本文专注于应对增量与开放式学习环境中存在的困难，而且提出了对应的优化目标来改进深度网络的训练机制。

2.3 本章小结

本章介绍了几种与本文研究相关的神经网络模型。竞争神经网络特别是 SOINN 网络是本文第三章研究工作的基础。而深度卷积神经网络则在本文第四、五章的实验中作为骨干网络架构发挥了重要作用。本章内容对这些神经网络模型的原理及特点进行了较为完整的描述，有助于读者理解本文后续的内容。

第三章　基于竞争神经网络与自适应距离度量的增量开放式无监督学习算法

在增量开放式的无监督学习任务中，输入的数据是一个有序的无标签样本序列，也被称为数据流。本章将介绍一种用于增量式无监督学习的竞争性神经网络，称为基于密度的自组织增量神经网络 (DenSOINN)。DenSOINN 是一个自组织的竞争网络，它以在线的神经元竞争学习学习为基本学习规则，结合竞争 Hebbian 学习规则和拓扑学习，能够以自适应生长的方式学习适应数据的分布的神经元节点。DenSOINN 能够提取数据流的可信代表集，同时具备初步的密度估计能力，并根据密度连通区域进行聚类。通过基于密度的聚类机制，DenSOINN 能够发现任意形状的聚类，并抑制来自噪声样本的负面影响。此外，DenSOINN 采用了一个自适应的距离度量机制，使其对于非归一化的输入数据也能获得学习的良好性能。

3.1　引言

无监督学习是一种重要的机器学习技术，包括聚类、密度估计等研究问题，在过去几十年中吸引了大量的研究兴趣[43-44]。它已广泛应用于数据挖掘、自然语言处理、计算机视觉等领域[45-48]。本章主要面向无监督学习中的聚类问题，以具有增量学习方式的数据流聚类为研究点。聚类的主要任务是将给定数据集中的相似模式分组为有意义的簇。数据流聚类是近年来引起广泛关注的一种特殊类型的聚类，它受到涉及大数据集和实时学习任务的应用的推动。数据流是组织成有序序列的一组模式。与传统的离线算法相比，流聚类模型有许多不同之处[49]。

首先，训练样本以序列的形式被连续输入到学习系统，学习系统无法控制输入序列。流聚类模型不能随机访问模式或迭代学习整个数据集。只有少量样本可以存储在内存中，需要在处理后丢弃。

其次，数据流中的底层数据分布可能是不稳定的。流聚类模型应该能够学习新的数据分布，而不会忘记以前学习的知识。

第三，数据流中的样本数量可能非常大，而算法可用的时间和存储空间是有限的。流聚类算法应该使用简洁的聚类表示模型，并且其计算时间与样本数量应当是线性关系。

根据聚类任务的需要，可能会考虑其他一些问题。在某些任务中，输入数据可能包含错误和噪声，这些任务就需要算法能够检测异常样本并采取相应措施进行去噪[50][23]。某些应用可能认为最近的数据比旧数据更有价值，因此聚类算法采用时间窗模型来满足这种需求[50][51]。

然而，在所有现有的模型中仍有一个问题没有解决。聚类的本质是根据样本之间的相似与不相似划分同类与异类，而作为样本之间不相似性的度量，距离度量在聚类过程中起着重要的作用。对于 k-means[52]和 DBSCAN[53]等经典聚类算法来说，选择合适的距离度量可以大大提高它们的性能。在许多真实世界的数据集中，不同属性的范围变化很大，例如人类年收入的取值范围远大于年龄的取值范围。一些广泛使用的距离度量 (如欧几里德距离) 在这种情况下并不适用，因为具有相对较宽范围值的属性会对距离值产生较高的影响。为了确保每个特征在距离度量中贡献相等，学习系统通常使用数据归一化 (data normalization) 对数据集进行预处理技术，以此来统一属性的取值范围[54] 。然而，数据归一化算法很难在数据流的约束条件下实现，因为它们需要计算整个数据集的统计特征，但这在数据访问受限时是不可行的。

针对上述问题，本章提出了一种新的开放式增量学习方法——基于密度的自组织增量学习神经网络 (Density based Self Organizing Neural Network, DenSOINN)，应用于数据流的聚类问题。本章内容的主要研究贡献如下:

(1) 采用自适应的距离度量来逼近数据归一化的效果，使得算法既能够适用于归一化的数据，也能够适用于原始数据;

(2) 提出了一种具有增量无监督学习能力的新型竞争神经网络，能够提取数据流的可信代表集;

(3) 提出了一种新的基于密度的方法来从竞争神经网络模型中提取聚类。

本章的其余部分组织如下: 第 2 节回顾相关工作，第 3 节描述 DenSOINN 的细节，第 4 节给出分析和讨论，第 5 节给出在人工和真实世界数据集上的实

验结果，第 6 节总结本章内容。

3.2 相关工作

3.2.1 增量式聚类算法

在过去的几十年里，研究者们通过改进传统聚类算法的方式，提出了许多增量式聚类算法，例如 BIRCH[55]，CluStream[56]，DenStream[50]，StreamKM++[57]，StrAP[58]等。这些算法可以概括为两个步骤：一个在线学习步骤，从输入的数据流中提取数据代表集；一个离线聚类步骤，生成最终的聚类结果。

在线学习步骤中，输入的训练样本被总结为特定的数据结构，即代表集，以便算法能够在不存储这些数据点的前提下学习数据的分布。这种数据结构最早在 BIRCH 算法中被引入。该数据结构称为聚类特征向量（CF），有三个组成部分：数据点的数量 N，数据点的线性和 LS，以及数据点的平方和 SS。从这些组成部分中可以计算出局部分布信息，例如聚类均值 $\mu = \frac{LS}{N}$ 和标准差 $\delta = \sqrt{\frac{SS}{N} - (\frac{LS}{N})^2}$。聚类特征向量可以增量地维护，并且可以通过简单地相加两个向量来合并。CluStream 将聚类特征向量扩展为一种称为微簇 (micro-cluster，即微型聚类簇) 的数据结构，通过在向量中添加时间特征来维护微簇中数据点时间戳的统计摘要。DenStream 也将其数据摘要结构命名为微簇，它包含三个与 BIRCH 中的聚类特征向量类似的组成部分。DenStream 可以检测输入数据中的异常值，因此它对训练样本中的噪声不太敏感。DenStream 采用了一种阻尼窗口模型，使用一个用户定义的参数来控制对新数据的重视程度，给最近学习的训练样本分配更高的权重并降低旧样本的权重。StreamKM++ 通过微簇的合并和删除机制来维护其代表集。StrAP 记录了那些与现有微簇不匹配的输入样本作为异常值，并使用一个异常样本池来存储这些异常值。

离线聚类步骤通常是传统聚类算法的变体。在线学习步骤产生的数据代表集被视为一组带有不同权重的伪训练样本，这些样本被输入到聚类算法中以获取最终的聚类结果。K-means[52][59]和 DBSCAN[53]是最广泛使用的标准聚类算法。BIRCH、Clustream 和 StreamKm++ 在它们的离线聚类步骤中采用了 K-Means。StreamKm++ 还使用了 K-means++[60]算法来进行聚类中心的初始化。DenStream 使用 DBSCAN 来对微簇进行聚类。StrAP 是基于 Affinity Propaga-

tion[61]扩展而来的一种基于消息传递的聚类方法，在被提出后引起了许多研究兴趣。早期数据流聚类算法的综述可以在[62]中找到。近年来，研究者还提出了一些新的数据流聚类算法，包括 Str-FSFDP[63]、IDEStream[64]和 CEDAS[65]等。在线聚类算法的缺陷通常也与其基础的离线聚类算法有关。K-Means 需要预先确定聚类数 k，因此不具备增量学习所需的模型开放性，并且倾向于生成大小相似的球形聚类。DBSCAN 可以检测出适合数量的任意形状的聚类，但它有两个对于聚类性能至关重要的超参数，但有时很难选择合适的参数值，尤其是在高维数据上。此外，DBSCAN 在分离重叠聚类时也有一定的困难。

竞争神经网络是另一种可以用于流式聚类的模型，比如自组织映射网络 (Self Organizing Map, SOM)[66]、生长型神经气体 (Growing Neural Gas, GNG)[67]和自组织增量神经网络（Self Organizing Incremental Neural Network, SOINN)[23]。在竞争神经网络的在线学习过程中，每个输入样本都会触发神经节点之间的竞争，胜者将得到奖励。随着学习过程的进行，节点被训练成为历史输入数据的代表集。然而，如何从网络中得到最终的聚类仍然是一个问题。一些基于竞争神经网络的增量式聚类算法，如 AING[68]和 G-Stream[51]，都将每个节点作为一个聚类。但是这两种网络中节点的数量都大约随着输入数据点数量呈线性增长直到达到预定义限制，在大多数情况下这个限制远远超过了真实分类数目。SOINN 将拓扑图中每个连通分量作为一个聚类：从一个未分类节点开始，将所有与它有路径相连节点标记为特定标签。然而这种方法不能完美地分离重叠区域，并且对参数设置非常敏感。基于 SOINN 还有一些改进模型，包括 E-SOINN[26]、Adjusted-SOINN[31]和 LB-SOINN[69]。这些模型都进行了一些创新来检测网络中重叠区域，但它们继续使用基于连通分量划分聚类的方法，聚类的结果依旧不够稳定。

近年来的聚类研究工作通常与深度学习相结合，将聚类与无监督表征学习共同作为目标来优化神经网络，起到了相辅相成的作用。代表性的研究工作包括 Deep Embedding Clustering[70], Deep Cluster[71], DERC[72]等等。因为优化深度神经网络需要使用完整的数据集反复进行迭代训练，而增量式的训练则需要面临“灾难性遗忘”的难题，因此深度聚类算法还没有广泛应用到增量聚类领域。论文[73]中提出了一种在线的深度聚类方法，但其作用是在一个训练周期内作为优化深度表征模型的辅助，整体的表征学习框架并非增量学习。论文[74]提

出了一种基于对比学习的增量式聚类算法，在真正意义上实现了深度聚类与增量学习的结合。论文[75]将深度在线聚类应用于视频流中的动作分割。总的来说，基于深度学习的增量式聚类研究成果数量还比较少，因为深度神经网络模型的灾难性遗忘特性，研究难度大于从传统聚类算法中衍生出的增量式算法。但因为深度神经网络的强大学习能力，这一方向的研究工作仍有较大的潜力可供发掘。

3.2.2 基于密度的聚类

在基于密度的聚类方法中，一个簇被定义为一组形成连接的高密度区域的输入模式，该区域与其他簇由连接的低密度区域分隔开来。基于密度的聚类并不以最小化类内距离作为优化目标，因此不能据此定义损失函数。与基于中心的聚类算法相比，基于密度的簇内样本之间的最大距离更远。密度聚类方法在某些特定应用中表现良好，因为它们具有以下优点：

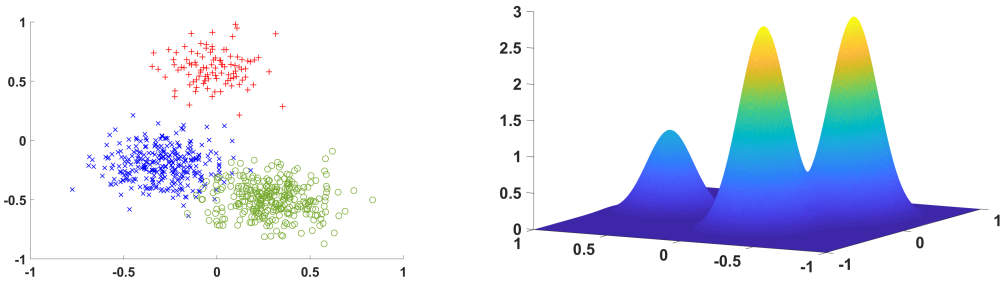
(1) 不需要假设簇的数量。

(2) 能够生成任意形状的簇。

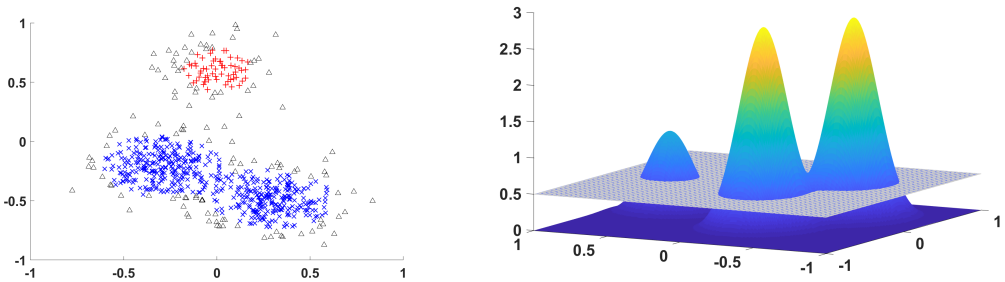
(3) 可以识别输入模式中的异常值。因此，密度聚类方法通常对噪声具有鲁棒性。

基于密度的方法需要解决两个问题，即如何估计密度分布和如何定义密度连通性。在许多基于密度的算法中，例如 DBSCAN[53]和 OPTICS[76]，样本 x 的密度是通过计算距离样本 x 半径 ϵ 内的样本数量来计算的。DENCLUE[77]使用核密度估计来估计概率密度分布。一对样本之间的密度连通性通常是基于它们之间的距离来定义密度连通性，例如在 DBSCAN 中，如果它们之间的距离不超过阈值 ϵ，则定义两个样本直接连接。

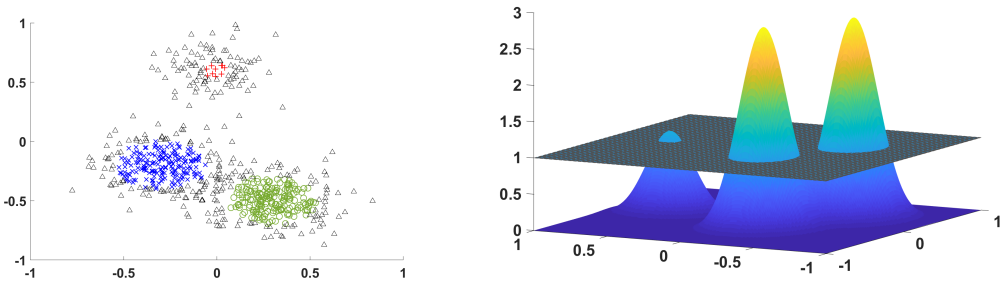
基于密度连通性的算法可以被理解为在概率密度函数中定义一个阈值，并且将概率密度高于阈值的连通区域视为簇。但是，这种方法面临一个难题，如图3.1所示。如果密度阈值设置得太低，则算法无法分离重叠的簇。如果密度阈值设置得太高，则会将太多的样本视为噪声。在局部密度差异较大的数据集中，特别难以选择适当的密度水平。此外，密度水平的选择对参数敏感，但有时很难设置参数。以 DBSCAN 为例，邻域的最大半径 ϵ 和使一个样本成为核心点的最小邻居数 ($minPts$) 共同决定了这一阈值，但这两个参数在高维数据集中很难设



(a) 一个示例数据集及其概率密度分布



(b) 密度阈值过低时的聚类结果: 未能成功分离具有重叠区域的聚类



(c) 密度阈值过高时的聚类结果：太多样本被当做噪声

图 3.1　一个示例数据集及其概率密度分布。左栏显示了二维空间中的数据点和聚类结果。右栏显示了概率密度分布和每次聚类时使用的密度阈值。数据集由从三个高斯分布中采样的样本组成。各个类中的样本以不同形状和颜色的标记显示。被聚类算法标记为噪声的样本显示为黑色三角形。

置。

近年来，Rodriguez 等人[78]提出了一种新颖的基于密度的算法，其中通过找到数据分布中的密度高峰来执行聚类，因此被称为密度峰值法。密度峰值法方法可以描述为：簇中心被具有较低局部密度的邻居包围，并且它们与具有较高局部密度的任何点相距相对较远。”该算法避免了找到适当密度水平以分离簇的挑战性问题。然而，密度估计方法与 DBSCAN 中相同，仍然需要预设距离阈值。

3.3 基于密度的自组织增量网络

基于密度的自组织增量网络（Density Based Self Organizing Incremental Neural Network, DenSOINN）采用单层网络结构，与传统竞争神经网络（如自组织映射、神经气和 E-SOINN）类似。网络结构可以用有向图 $G =< V, E >$ 表示，其中 V 是神经节点集合，E 是连接集合。特征空间上的相邻数据点被映射到同一个节点或两个相邻节点。每个节点 $i \in V$ 是由映射到它的输入模式形成的微簇的原型。E 中的边是有向边，从节点 i 到节点 j 的边被记为 (i, j)。作为一种聚类算法，V 被分成许多子集，每个子集代表一个簇。聚类结果用标签向量 $\boldsymbol{l} = (l_1, l_2, ..., l_{|V|})^T$ 表示。如果 $l_i = l_j$，则节点 i 和 j 在同一簇中。当收到聚类请求时，DenSOINN 输出 V、E 和 $\boldsymbol{l}$。图.3.2 显示了 DenSOINN 的学习流程图。

与大多数流式聚类算法一样，DenSOINN 可以总结为两个步骤：网络结构的在线训练和神经节点的离线聚类。在在线学习步骤中，在每轮学习开始时调整距离度量。当训练样本被输入给网络时，将触发节点之间的竞争，并找到获胜者和亚军。如果获胜者和亚军都被激活，则在它们之间建立连接并更新获胜者的特征向量。否则，将训练样本添加到网络种作为新节点。网络每学习一定数量的训练样本后执行去噪过程，删除网络中孤立的节点。当收到聚类请求时，DenSOINN 开始离线聚类步骤。首先将网络切割成连通分量，然后将每个子图上的密度峰值节点分配为簇中心，并相应地对其他节点进行分组。本节的后续内容将详细描述每个步骤。

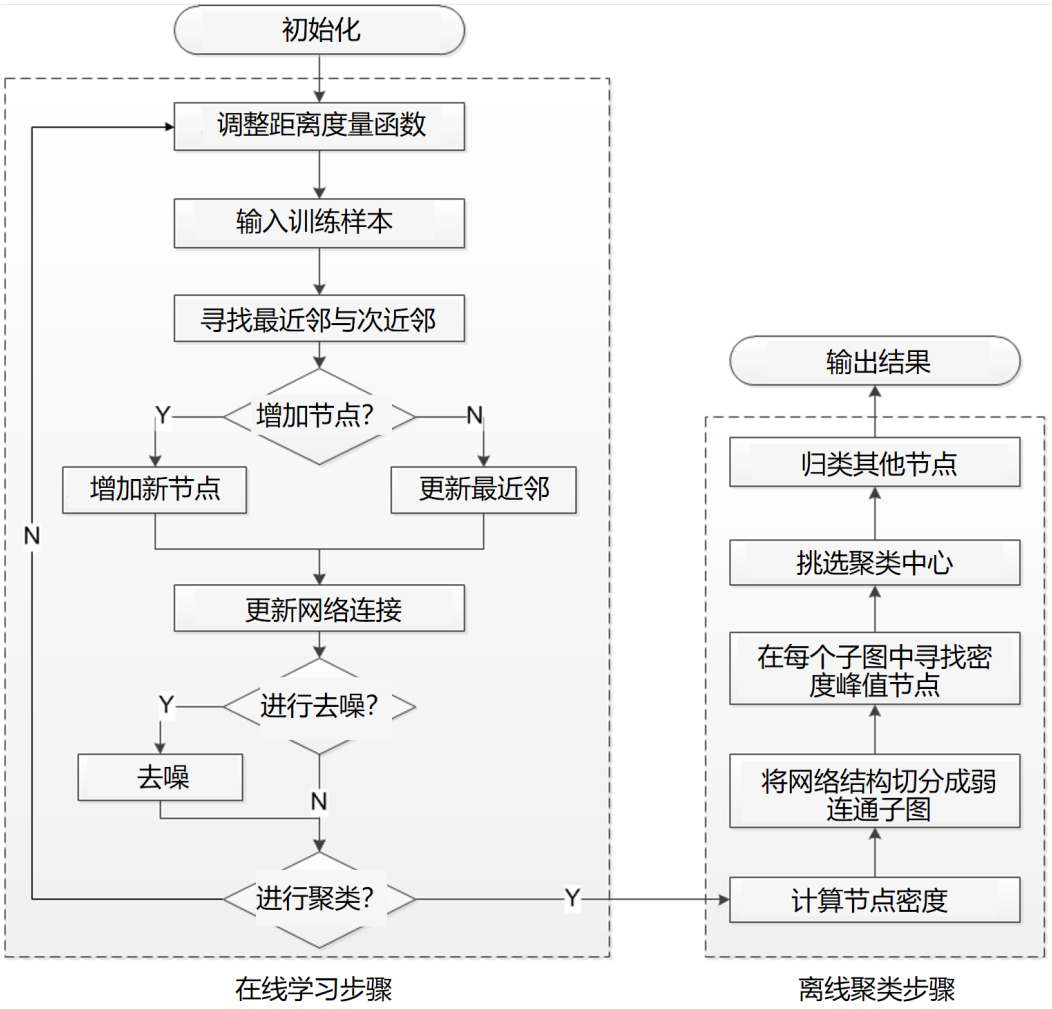


图 3.2　DenSOINN 的学习流程图

3.3.1 自适应距离度量

本章引言中讨论过，由于训练数据的访问受限，数据归一化在增量式聚类中难以实现。为了解决这个问题，DenOINN 引入了一种自适应的距离度量方法，其基础是参数可变的马氏距离 (Mahalanobis Distance)，公式为：

$$d_A(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \boldsymbol{y})^T A(\boldsymbol{x} - \boldsymbol{y})} \tag{3.1}$$

这里 $\boldsymbol{A}$ 是一个半正定参数矩阵。距离度量是“自适应”的，因为 $\boldsymbol{A}$ 根据输入数据的统计特征在增量学习过程中动态更新。马氏距离可以看作原始输入空间中任意线性缩放和旋转的欧几里得距离。DenSOINN 在原始数据上使用这种距离度量来近似归一化数据上的欧几里得距离。具体来说，参数矩阵 $\boldsymbol{A}$ 适应特征的不同值范围，使特征对最终距离的贡献大致成比例。

在本文描述的算法和实验中，$\boldsymbol{A}$ 中的元素是根据 Min-Max 归一化方法计算得到的。假设第 i 维特征的范围为 $[min_i, max_i]$，Min-Max 归一化的目标范围为 $[0, 1]$，则对于样本 $\boldsymbol{x}$，Min-Max 归一化的公式为：

$$x_i^{'} = \frac{x_i - min_i}{max_i - min_i} \tag{3.2}$$

将样本 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 经过归一化之后得到的样本记为 $\boldsymbol{x'}$ 和 $\boldsymbol{y'}$，则可以得到以下定理来计算参数矩阵 $\boldsymbol{A}$。

定理 3.1 设 $\boldsymbol{A}$ 为马氏距离的对角参数矩阵，其中 $\boldsymbol{A_{ii}} = \frac{1}{(max_i - min_i)^2}$，则 $d_A(\boldsymbol{x}, \boldsymbol{y}) = d_{euc}(\boldsymbol{x'}, \boldsymbol{y'})$。

证明 根据公式 (3.1)：

$$d_A(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{n} A_{ii}(x_i - y_i)^2} = \sqrt{\sum_{i=1}^{n} (\frac{x_i - y_i}{max_i - min_i})^2}$$

而 $\boldsymbol{x}^{'}$ 和 $\boldsymbol{y}^{'}$ 之间的欧几里得距离为

$$d_{euc}(\boldsymbol{x}', \boldsymbol{y}') = \sqrt{\sum_{i=1}^{n}(\frac{x_i - min_i}{max_i - min_i} - \frac{y_i - min_i}{max_i - min_i})^2} = \sqrt{\sum_{i=1}^{n}(\frac{x_i - y_i}{max_i - min_i})^2} \quad \square$$

因此可以得到 $d_A(\boldsymbol{x}, \boldsymbol{y}) = d_{euc}(\boldsymbol{x}', \boldsymbol{y}')$。

在增量式聚类中，特征空间中各维度的最大值和最小值无法直接计算，但只需要在在线学习步骤中记录输入数据在各维度上的最大值和最小值，当新输入的样本打破最大值或最小值记录时相应地调整 $\boldsymbol{A}$ 即可解决这个问题。

然而，Min-Max 归一化的一个主要问题是训练数据的最大值和最小值很容易受到异常值的影响，因此 DenSOINN 使用其网络节点特征向量的最大值和最小值代替。给定网络节点 j，其特征向量的值代表 j 在 n 维特征空间中的坐标，记为 $\boldsymbol{w_j} = (w_{j,1}, w_{j,2}, ..., w_{j,n})^T$。因为节点的特征向量是节点所代表的微簇中所有样本的均值，而且 DenSOINN 网络具备去除噪音节点的功能，所以节点特征向量中元素的最值相对不容易受到噪声影响。由于节点位置在每轮学习中都会改变，$\boldsymbol{A}$ 需要相应地进行更新。在每轮学习开始时，首先使用算法 1 调整参数矩阵 $\boldsymbol{A}$。本节的后文将省略参数矩阵 $\boldsymbol{A}$ 并使用 $d(\boldsymbol{x}, \boldsymbol{y})$ 表示由此度量计算出的 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 之间的距离，除非 $\boldsymbol{A}$ 的计算方式与算法 1不同。

3.3.2 在线网络训练

DenSOINN 的在线学习步骤使用训练数据集中的样本迭代训练其节点和连接，每个循环中仅学习一个训练样本。

DenSOINN 中的每个节点代表一个由历史训练样本构成的微簇。每个节点 i 使用三个属性来描述其所代表的微簇：（1）特征向量 $\boldsymbol{w_i} = (w_{i,1}, w_{i,2}, ..., w_{i,n})^T$，表示微簇的质心，也是节点 i 在特征空间中的“位置”；（2）权重 m_i，计算微簇中累积的训练样本数量；（3）激活阈值 t_i，描述其与其他节点的关系。

如果 i 与其他节点之间存在连接，则 t_i 计算为 i 和其邻居之间的最大距离：

$$t_i = \max_{j \in N_i} d(\boldsymbol{w_j}, \boldsymbol{w_i}) \tag{3.3}$$

Algorithm 1 计算自适应距离度量的参数矩阵 A

输入:　V: 节点集合;
输出:　A: 马氏距离的 $n \times n$ 参数矩阵;

初始化 A 为零矩阵;
for 输入空间的每个维度 i do
    $max_i = -\infty, min_i = \infty$;
    for 每个节点 $j \in V$ do
        if $w_{j,i} > max_i$ then
            $max_i = w_{j,i}$;
        else
            if $w_{j,i} < min_i$ then
                $min_i = w_{j,i}$;
            end if
        end if
    end for
    if $max_i = min_i$ then
        $A_{ii} = 0$;
    else
        $A_{ii} = \frac{1}{(max_i - min_i)^2}$;
    end if
end for
return A

在公式 (3.3) 中，N_i 表示节点 i 的邻居集合，即 $N_i = \{j \in V | (i,j) \in E$ or $(j,i) \in E\}$。如果 N_i 为空，则 t_i 计算为 i 和其他节点之间的最小距离：

$$t_i = \min_{j \in V} d(\boldsymbol{w_j}, \boldsymbol{w_i}) \tag{3.4}$$

从节点 i 到其邻居 j 的边，记为 (i,j)，具有一个权重 $m_{i,j}$，当 i 的位置改变时减小，当 i 和 j 都被激活时增加。在算法开始时，网络初始化为一个空图 $G =< V, E >$，其中 V 和 E 都是空集。

在处理每个输入的训练样本之前需要先使用算法 1 调整距离度量。

每次迭代中的学习过程如图.3.3 所示。

当输入模式 $\boldsymbol{x}$ 到达时，如果 $|V| < 2$，则 $\boldsymbol{x}$ 形成一个新的微簇，即一个新节点 i。微簇的质心 $\boldsymbol{w_i} = \boldsymbol{x}$，其权重，即累积的模式计数，$m_i = 1$。如果 $|V| \geq 2$，则在 V 中的节点之间触发竞争，用以下算式找到最近邻 s_1 和次近邻 s_2：

$$s_1 = arg \min_{i \in V} d(\boldsymbol{x}, \boldsymbol{w_i}) \tag{3.5}$$

$$s_2 = arg \min_{i \in V - \{s_1\}} d(\boldsymbol{x}, \boldsymbol{w}_i) \tag{3.6}$$

如果 $d(\boldsymbol{x}, \boldsymbol{w}_{s_1}) \leq t_{s_1}$ 且 $d(\boldsymbol{x}, \boldsymbol{w}_{s_2}) \leq t_{s_2}$，则 $\boldsymbol{x}$ 可以激活 s_1 和 s_2。在这种情况下判定 $\boldsymbol{x}$ 属于由 s_1 表示的微簇，将 m_{s_1} 的值增加 1，并按以下方式更新 $\boldsymbol{w}_{s_1}$：

$$\boldsymbol{w}_{s_1} = \boldsymbol{w}_{s_1} + \frac{1}{m_{s_1}}(\boldsymbol{x} - \boldsymbol{w}_{s_1}) \tag{3.7}$$

当 s_1 的位置发生变化时，从 s_1 出发的旧连接可能不再有效。因此 DenSOINN 加入了连接权重机制，使每次 s_1 的位置改变时，其连接的权重都会减小，当某个连接的权重低于阈值时，连接将被删除。删除阈值默认设置为从 s_1 开始的所以连接权重之和的 10%。对于每个边 $(s_1, i) \in \boldsymbol{E}$，连接的权重更新如下：

$$m_{s_1,i} = 2^{-\gamma} m_{s_1,i} \tag{3.8}$$

这里 γ 是一个预定义的超参数，用于控制连接权重的衰减。

如果 $\boldsymbol{x}$ 同时激活了 s_1 和 s_2，但网络中不存在从 s_1 到 s_2 的连接，则将一个新连接 (s_1, s_2) 添加到 $\boldsymbol{E}$ 中，并设置 $m_{s_1,s_2} = 1$，否则将 m_{s_1,s_2} 增加 1。然后对于每个连接 $(s_1, i) \in \boldsymbol{E}$，采用上式更新其权重，如果 $m_{s_1,i} < \frac{\sum_{(s_1,j) \in E} m_{s_1,j}}{10}$，则从 $\boldsymbol{E}$ 中删除边 (s_1, i)。本章的下一节将进一步分析连接管理方案。

如果 $d(\boldsymbol{x}, \boldsymbol{w}_{s_1}) > t_{s_1}$ 或 $d(\boldsymbol{x}, \boldsymbol{w}_{s_2}) > t_{s_2}$，则 $\boldsymbol{x}$ 不能激活 s_1 和 s_2。在这种情况下将添加一个微簇到网络中，由新节点 r 表示，其特征向量 $\boldsymbol{w}_r = \boldsymbol{x}$，权重 $m_r = 1$。然后在 $\boldsymbol{E}$ 中增加一条新节点 r 和最近邻 s_1 之间的连接。DenSOINN 的单次迭代学习过程如图3.3所示。

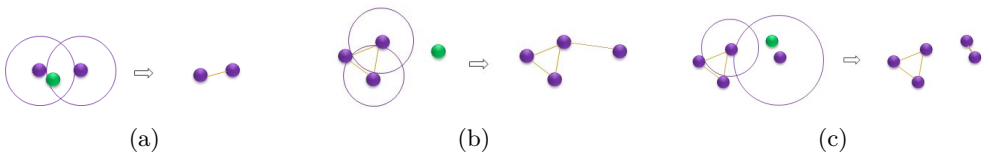


图 3.3　DenSOINN 单次迭代训练过程。在图中，绿色节点表示输入给网络的训练样本，其他节点表示网络中的节点。在图（a）中，最近的网络节点被拉向训练样本，并在两个节点之间建立连接。在图（b）和（c）中，新节点被加入到网络中，并在新节点与其最近邻之间建立新连接

网络最大容量可以选择使用一个上限 $MaxNodes$ 进行控制，以防网络中节

点数量过大导致计算时间不可接受。如果 $|V| \geq MaxNodes$, 则网络不再产生新的节点, 输入样本总是按同时激活 s_1 和 s_2 进行处理。

每当训练过程迭代一定次数之后，DenSOINN 对其中的节点进行去噪，基于以下因素判断一个节点 i 是否是噪声节点：首先，噪声节点较少被训练样本激活，因此其权重应低于所有节点平均权重的一定比例；其次，噪声节点的邻居数量很少，当两个节点的权重相等时，邻居少的节点更可能是噪声。因此，节点 i 在满足以下条件时会被判定为噪声：

$$m_i < \frac{10^{1-|N_i|}\epsilon}{|V|} \sum_{j \in V} m_j \tag{3.9}$$

在线训练过程重复进行，直到数据流中的所有样本都已被学习，或学习系统收到用户给出的聚类请求。算法2给出了上述过程的伪代码描述。

3.3.3 基于密度的节点聚类

当全部训练数据都已学习完毕，或者用户发送了聚类请求时，DenSOINN 对网络中的节点进行聚类并输出聚类结果。由于距离度量的影响，由节点表示的微簇呈椭圆形。然而，真实数据中类簇的形状是不确定的，因此 DenSOINN 采用基于密度的离线聚类方法，目的是将微簇组合成可能是任何形状的大型类簇。

节点 i 的密度，记为 p_i，定义为其 Voronoi 区域 $\boldsymbol{R}_i$ 的平均概率密度。将 $\boldsymbol{R}_i$ 的体积表示为 $vol(\boldsymbol{R}_i)$，则数据点 $\boldsymbol{x}$ 位于 $\boldsymbol{R}_i$ 中的概率为 $P(\boldsymbol{x} \in \boldsymbol{R}_i) = p_i vol(\boldsymbol{R}_i)$，并且可以通过 $P(\boldsymbol{x} \in \boldsymbol{R}_i) \approx \frac{m_i}{\sum_{j \in V} m_j}$ 来估计。因此可以得到：

$$p_i \approx \frac{m_i}{vol(\boldsymbol{R}_i) \sum_{j \in V} m_j} \tag{3.10}$$

在高维特征空间中计算 Voronoi 区域的体积很困难。一种简单的估计方式是 $vol(\boldsymbol{R}_i) \propto d_i^n$，其中 n 是特种空间的维数，d_i 是节点 i 到其邻居的平均距离：

$$d_i = \frac{1}{|N_i|} \sum_{j \in N_i} d(\boldsymbol{w_i}, \boldsymbol{w_j}) \tag{3.11}$$

然而，在真实的数据集中，这种估计并不奏效。因为数据通常分布在高位空

Algorithm 2 DenSOINN 的在线训练算法

输入: 输入数据集 X；连接权重衰减速度 γ；两次去噪过程的间隔 $DenoisingInterval$；去噪强度 ϵ；

输出: 节点集合 V；边集合 E;

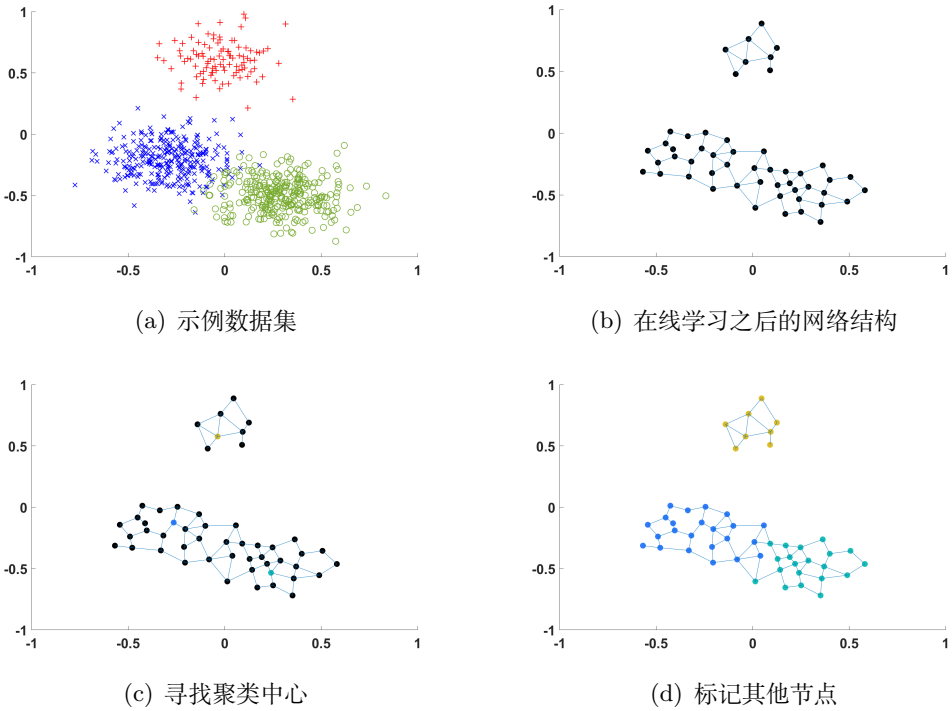
用 $V = \phi$，$E = \phi$ 初始化网络;
for 每个样本 $\boldsymbol{x} \in X$ do
    if $|V| < 2$ then
        创建一个新节点 r，使 $\boldsymbol{w}_r = \boldsymbol{x}$ 并将其添加到 V，转到步骤 2;
    end if
    通过算法1调整距离度量;
    使用公式 (3.5) 和 (3.6) 找到 $\boldsymbol{x}$ 的最近邻 s_1 和最近邻 s_2;
    使用公式 (3.3) 和 (3.4) 计算激活阈值 t_{s_1} 和 t_{s_2};
    if $|V| < MaxNodes$ 且 $(d(\boldsymbol{x}, \boldsymbol{w}_{s_1}) > t_{s_1}$ 或 $d(\boldsymbol{x}, \boldsymbol{w}_{s_2}) > t_{s_2})$ then
        创建节点 r，使 $\boldsymbol{w}_r = \boldsymbol{x}$ 并将其添加到 V;
        建立连接 (r, s_1) 并将其添加到 E;
    else
        将 m_{s_1} 增加 1，并使用公式 (3.7) 更新 $\boldsymbol{w}_{s_1}$;
        使用公式 (3.8) 减小从 s_1 开始的连接的权重;
        搜索并删除小权重的连接;
        if (s_1, s_2) 不存在 then
            建立连接 (s_1, s_2) 并将其添加到 E:
        else
            将权重 m_{s_1,s_2} 增加 1;
        end if
    end if
    if 已学习的样本数量是 $DenoisingInterval$ 的整数倍 then
        计算节点的平均权重;
        使用公式 (3.9) 搜索并删除噪声节点及其边缘;
    end if
end for
return V, E

间中具有较低内在维数的流形上。将内在维数记为 n'，因此 $vol(R_i) \propto d_i^{n'}$。学习系统无法事先得知 n' 的值，但实验中发现将 n' 设置为 2 时通常可以获得较好的结果。结合上述公式，可以将 p_i 估计为：

$$p_i \propto \frac{m_i}{1 + d_i^2} \tag{3.12}$$

上式在分母中添加了一个正则项以防止邻域半径非常小的节点密度过高。

DenSOINN 中节点的离线聚类可以被认为是“密度连通”和“密度峰值”方法的组合。密度连接区域被定义为网络图中弱连接组件。聚类过程分为三步，首先将网络划分为弱连接子图，然后在每个子图上找到密度峰值并将其指定为簇中心，最后将剩余节点划分到类簇中。图3.4是聚类过程的一个简单示例。



(a) 示例数据集　(b) 在线学习之后的网络结构

(c) 寻找聚类中心　(d) 标记其他节点

图 3.4　在图.3.1所示数据集上的聚类过程。网络图被分割成弱连通子图，然后每个子图上的密度峰值节点被选为聚类中心。剩余的节点按密度从高到低的顺序，依次被标记为距离它最近的密度更高节点的同类。

以弱连通分量 $SG =< SV, SE >$ 为例，对于每个节点 $i \in SV$，找到密度高于 i 的最近节点 q_i，将该节点表示为 q_i：

$$q_i = arg \min_{j \in SV \ and \ p_i < p_j} d(\boldsymbol{w_i}, \boldsymbol{w_j}) \tag{3.13}$$

然后将节点 i 分配一个 “峰值分数”，表示为 h_i：

$$h_i = p_i d(\boldsymbol{w_i}, \boldsymbol{w_{q_i}}) \tag{3.14}$$

如果节点 i 满足以下条件，则选择节点 i 作为聚类中心：

$$h_i \geq \alpha h_{mean} \tag{3.15}$$

上式中 α 是预定义的超参数，h_{mean} 是 SV 中节点的平均峰值分数。如果节点 i 的密度在 SV 中最高，则无法找到 q_i，因此无法计算 h_i。在这种情况下设定 $h_i = \infty$，这意味着节点 i 总是被选择为聚类中心，并且在计算 h_{mean} 时不考虑 h_i。

如果节点 i 没有被选择为聚类中心，则将其与 q_i 分组到同一聚类中。为了确保在标记 i 时已经将 q_i 分配到一个聚类中，在此步骤前先按节点密度的降序对 SV 进行排序，因此始终可以在标记 i 之前标记 q_i。

上述方法的形式化描述如算法 3所示。

3.4　算法分析

3.4.1　关于自适应距离度量的分析

上一节描述了自适应距离度量，它以增量方式近似 Min-Max 归一化。自适应距离度量的效果类似于简单的度量学习算法。度量学习的目标是选择一个合适的距离度量，以提高特定基于度量的学习算法（如 k-means 和 k-nearest neighbor 分类器）的性能。但在现有的度量学习框架中，度量函数是在用于基于度量的算法之前学习的。而 DenSOINN 中的自适应距离度量是在增量学习的过程中同时学习和使用的，这使其与度量学习算法有所区别。除了 Min-Max 归一化之外，还可以通过另外方式计算参数矩阵 $\boldsymbol{A}$ 以实现其他效果。本节用两个例子说明自适应距离度量可以实现的其他数据规范化方法。

第一个例子是白化变换 (whitening)。数据白化是一种广泛使用的预处理方法，可以消除特征之间的相关性。当将参数矩阵 $\boldsymbol{A}$ 分配为输入数据集协方差矩阵 $\boldsymbol{\Sigma}$ 的逆时，输入空间中的马氏距离等效于变换空间中的欧几里得距离。在增

Algorithm 3 基于节点密度的聚类

输入:　V: 节点集合, E: 边集合, α: 聚类中心选择参数;
输出:　l: 节点的聚类标签;

for 每个节点 $i \in V$ do
　用公式 (3.12) 和 (3.11) 计算节点密度 p_i;
end for
将图 $G =< V, E >$ 分成连通分量 $\{SG_1, SG_2, ...\}$;
$y = 1$;
for 每个连通分量 $SG_t =< SV_t, SE_t >$ do
　按节点密度降序排列 SV_t 中的节点;
　for 每个节点 $i \in SV_t$ do
　　用公式 (3.13) 找到具有更高密度的最近节点 q_i;
　　用公式 (3.14) 计算峰值得分 h_i;
　end for
　计算平均峰值得分 h_{mean};
　for 每个节点 $i \in SV_t$ do
　　if $h_i \geq \alpha h_{mean}$ then
　　i 被选为聚类中心: $l_i = y, y = y + 1$;
　　else
　　将 i 与 q_i 分组到同一聚类中: $l_i = l_{q_i}$;
　　end if
　end for
end for
$l = (l_1, ..., l_{|V|})^T$;
return l

量式无监督学习过程中，由于训练数据的可访问性受限，算法不能直接计算协方差矩阵 Σ 及其逆矩阵，但是可以使用以下定理增量地学习近似 Σ^{-1} 的矩阵 $\boldsymbol{A}$。

定理 3.2 将前 n 个训练样本 $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ 的平均值和协方差矩阵分别表示为 $\overline{\boldsymbol{x}}_n$ 和 Σ_n，则 Σ_n^{-1} 的增量计算方法为：

$$\Sigma_n^{-1} = \frac{n}{(n-1)}\Sigma_{n-1}^{-1} - \frac{[\Sigma_{n-1}^{-1}(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})][(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})^T\Sigma_{n-1}^{-1}]}{(n-1)[1 + \frac{1}{n}(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})^T\Sigma_{n-1}^{-1}(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})]} \tag{3.16}$$

证明 显然，$\overline{\boldsymbol{x}}_n$ 可以用下式增量计算得出：

$$\overline{\boldsymbol{x}}_n = \overline{\boldsymbol{x}}_{n-1} + \frac{(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})}{n}$$

然后可以按以下方式更新 Σ_n：

$$\begin{aligned}
\Sigma_n &= \sum_{i=1}^{n} \frac{(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_n)(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_n)^T}{n} \\
&= \sum_{i=1}^{n} [\frac{(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_{n-1})(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_{n-1})^T}{n} + \frac{(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})^T}{n^3}] \\
&\quad \frac{(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})(\boldsymbol{x}_i - \overline{\boldsymbol{x}}_{n-1})^T + (\boldsymbol{x}_i - \overline{\boldsymbol{x}}_{n-1})(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})^T}{n^2} \\
&= \frac{(n-1)\Sigma_{n-1}}{n} + \frac{(n-1)(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})^T}{n^2}
\end{aligned}$$

然后，Σ_n^{-1} 可以使用 Sherman-Morrison 公式进行增量计算，该公式可以写成

$$(\boldsymbol{M} + \boldsymbol{b}\boldsymbol{c}^T)^{-1} = \boldsymbol{M}^{-1} + \frac{\boldsymbol{M}^{-1}\boldsymbol{b}\boldsymbol{c}^T\boldsymbol{M}^{-1}}{1 + \boldsymbol{c}^T\boldsymbol{M}^{-1}\boldsymbol{b}} \qquad \square$$

令 $\boldsymbol{M} = \frac{(n-1)\Sigma_{n-1}}{n}$, $\boldsymbol{b} = (n-1)(\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1})$，$\boldsymbol{c} = \frac{\boldsymbol{x}_n - \overline{\boldsymbol{x}}_{n-1}}{n^2}$，代入上式后化简，即可得到公式 (3.16).

如果训练数据的分布是稳定的，那么 Σ_n^{-1} 在学习部分训练数据后将近似于 Σ^{-1}。令参数矩阵 $\boldsymbol{A} = \Sigma_n^{-1}$，即可实现数据的增量式白化。

第二个例子是数据的 z-norm 规范化，同样可以通过定义矩阵 $\boldsymbol{A}$ 的计算方式以近似其效果。z-norm 规范化的公式为 $x_i' = \frac{x_i - \overline{x}_i}{\delta_i}$，其中 $\overline{x}_i$ 和 δ_i 是第 i 个特

征的平均值和标准差。很容易推断出 $d_{euc}(\boldsymbol{x}', \boldsymbol{y}') = \sqrt{\sum_{i=1}^{n}(\frac{x_i - y_i}{\delta_i})^2}$。

将此度量函数与马氏距离相结合，可以得到 A 的第 i 个对角元素为 $A_{ii} = \frac{1}{\delta_i^2}$。模型接收到每个输入样本时，通过更新 δ_i^2 来得到 A_{ii} 的值。记 n 个输入模式后第 i 个特征的均值和方差分别为 $\overline{x}_{i,n}$ 和 $\delta_{i,n}^2$，则其增量计算公式如下：

$$\overline{x}_{i,n} = \overline{x}_{i,n-1} + \frac{x_i - \overline{x}_{i,n-1}}{n} \tag{3.17}$$

$$\delta_{i,n}^2 = \frac{(n-1)\delta_{i,n-1}^2 + (x_i - \overline{x}_{i,n})(x_i - \overline{x}_{i,n-1})}{n} \tag{3.18}$$

综上所述，在本章提出的自适应距离度量方法中，参数矩阵 A 能够使用不同的计算方式来近似不同的数据规范化方法，以适用于各种应用数据。只要能够为数据规范化算法中的参数找到增量计算的方式，就可以用自适应距离模拟这种算法。

3.4.2　DenSOINN 中的密度连通性

上一节在介绍密度聚类算法时曾提到 DenSOINN 中的连通子图对应了数据分布的密度连通区域。本小节将讨论 DenSOINN 的图结构与密度连通性之间的关系。

DenSOINN 的边是按照竞争式 Hebbian 规则构建的，这一规则与 Voronoi tessellation 密切相关。假设特征空间是 n 维实空间，输入数据的概率分布函数是未知函数 $P(x)$，则与节点 i 和 j 相关联的第二阶 Voronoi 区域 $\boldsymbol{R}_{ij}$ 定义为：

$$\boldsymbol{R}_{ij} = \{\boldsymbol{x} | \forall k \notin \{i, j\}, d(\boldsymbol{x}, \boldsymbol{w_i}) \leq d(\boldsymbol{x}, \boldsymbol{w_j}) \leq d(\boldsymbol{x}, \boldsymbol{w_k})\} \tag{3.19}$$

只有当 $\boldsymbol{R}_i$ 和 $\boldsymbol{R}_j$ 相邻时，$\boldsymbol{R}_{ij}$ 才不为空。

每次 DenSOINN 学习输入样本 $\boldsymbol{x}$ 时，它都会寻找其最近邻 s_1 和次近邻 s_2。显然 $\boldsymbol{x} \in \boldsymbol{R}_{s_1 s_2}$。如果 $\boldsymbol{x}$ 可以激活 s_1 和 s_2，则创建边 (s_1, s_2) 或增加 m_{s_1,s_2}。否则，一个新节点 r 被创建，其特征向量 $\boldsymbol{w_r} = \boldsymbol{x}$。在这种情况下，$d(\boldsymbol{x}, \boldsymbol{w_r}) = 0 \leq d(\boldsymbol{x}, \boldsymbol{w}_{s_1}) \leq d(\boldsymbol{x}, \boldsymbol{w}_{s_2})$，因此 $\boldsymbol{x} \in \boldsymbol{R}_{rs_1}$ 并创建连接 (r, s_1)。由此可以得出结论：每个连接 (i, j) 对应一个二阶 Voronoi 区域 $\boldsymbol{R}_{ij}$，其中 $p(\boldsymbol{x} \in \boldsymbol{R}_{ij}) = \int_{\boldsymbol{R}_{ij}} P(x) > 0$。

通过这种方法构建的网络通常是高度连通的，因为只要二阶 Voronoi 区域的中概率密度大于 0 即存在对应的连接。因此，DenSOINN 在学习过程中动态删除代表相对低密度区域的边，以使网络图中的连通分量表示数据分布的高密度区域。如果 $p(\boldsymbol{x} \in \boldsymbol{R}_{ij}|\boldsymbol{x} \in \boldsymbol{R}_i)$ 很小，则将第二阶 Voronoi 区域 $\boldsymbol{R}_{ij}$ 视为低密度，并删除对应的连接。删除连接的另一个原因是节点位置在整个在线学习步骤中是持续变化的，使得 Voronoi 区域相应地发生改变。在这种情况下，在经过一些学习轮之后，原本连接在一起的节点可能不再相邻。

低密度区域由权重相对较小的网络连接表示。如果节点 i 有 k 条从它出发的连接，且样本 $\boldsymbol{x}$ 位于 $\boldsymbol{R}_i$ 中，则 $\boldsymbol{x}$ 属于哪个二阶 Voronoi 区域的概率遵循参数为 $\boldsymbol{\mu} = (\mu_1, ..., \mu_k)$ 的多项式分布。

若要找到包含 $\boldsymbol{x}$ 的概率较低的区域，需要考虑参数 $\boldsymbol{\mu}$ 的分布，它是具有超参数 $\boldsymbol{\phi} = (\phi_1, ..., \phi_k)$ 的狄利克雷分布。

$\boldsymbol{\mu}$ 的概率分布和期望值计算如下：

$$p(\boldsymbol{\mu}|\boldsymbol{\phi}) = \frac{\Gamma(\sum_{j=1}^{k} \phi_j)}{\prod_{j=1}^{k} \Gamma(\phi_j)} \prod_{j=1}^{k} \mu_j^{\phi_j - 1} \tag{3.20}$$

$$E(\boldsymbol{\mu}) = \frac{\boldsymbol{\phi}}{\sum_{j=1}^{k} \phi_j} \tag{3.21}$$

参数 $\boldsymbol{\phi}$ 的值由每个二阶 Voronoi 区域中有效样本的数量来计算得到，这些数量记录在从节点 i 开始的边的权重中。考虑节点位置都稳定的简单情况，这意味着不会添加新节点，现有节点的位置也不会改变，Voronoi 区域也不会改变。在这种情况下，ϕ_j 就是 $\boldsymbol{R}_{ij}$ 中样本的计数。

但是，节点和 Voronoi 区域的位置会不时发生变化。每次 i 的位置发生变化时，其关联二阶 Voronoi 区域中历史训练样本的计数变得不那么有效。为了记录有效样本的数量，DenSOINN 采用了一个阻尼窗口模型，为最近出现的样本分配更高的有效性。每个样本 $\boldsymbol{x} \in \boldsymbol{R}_i$ 的有效性随着节点 i 位置变化次数呈指数衰减。衰减函数为 $e(\boldsymbol{x}) = 2^{-\gamma t}$，其中 $\gamma > 0$，t 是输入 $\boldsymbol{x}$ 后 i 位置更改次数。

连接 (i, j) 的权重计算方式为 $m_{i,j} = \sum_{\boldsymbol{x} \in \boldsymbol{R}_{ij}} e(\boldsymbol{x})$。每次当学习样本 $x \in \boldsymbol{R}_{ij}$ 时，模型更新节点 i 的位置，然后将从 i 开始的边权重乘以 $2^{-\gamma}$ 减少，并将 $m_{i,j}$

增加 1。这种方式能够在线地计算节点 i 对应的各二阶 Voronoi 区域中历史训练样本的有效性之和。在此基础上，$p(\boldsymbol{x} \in R_{ij} | \boldsymbol{x} \in R_i)$ 的期望值计算如下：

$$E(p(\boldsymbol{x} \in R_{ij} | \boldsymbol{x} \in R_i)) = \frac{m_{i,j}}{\sum_{(i,r)\in E} m_{i,r}} \tag{3.22}$$

如果 $E(p(\boldsymbol{x} \in R_{ij} | \boldsymbol{x} \in R_i)) < 0.1$，则 R_{ij} 被视为低密度区域，其对应的连接 (i, j) 被从 E 中删除。这样即可找到低密度的二阶 Voronoi 区域，并从网络中删除表示这些区域的连接。

参数 γ 的效果类似于基于密度的聚类算法中设置密度阈值的参数，例如 DBSCAN 中的半径参数 ϵ 和最小邻居数 $minPts$。但是，γ 要容易调整得多，因为它不受特征的维数和取值范围的影响。

综上所述，DenSOINN 通过引入连接的权重调整与删除机制，使网络中的连通子图与数据分布的高密度区域对应起来，从而能够实现类似 DBSCAN 的基于密度连通区域的聚类。但是，这种方法在分割存在重叠区域的聚类时存在困难，因此仍然需要采用密度峰值机制在每个子图上找到聚类中心。

3.4.3　计算复杂度分析

假设输入给模型的数据流是一个包含 $|X|$ 个训练样本的流，其中每个样本用一个 n 维特征向量表示。每个在线学习轮次中，最耗时的计算是找到输入样本的最近邻和次近邻节点，其时间复杂度是 $O(n|V|)$。这里的 $|V|$ 是网络中神经元节点的数量。然而，在在线学习步骤中，$|V|$ 会随时间而变化。将 $|V|$ 的最大值记为 k，则在线学习过程的总时间复杂度为 $O(nk|X|)$。

如果离线聚类过程开始于在线学习过程完整结束后，则 $|V|$ 的值近似于 k。离线聚类步骤中最耗时的计算是找到每个节点的具有更高密度的最近节点，这需要计算同一子图中每对节点之间的距离。在最坏情况下，整个网络图结构仅有一个连通分量，此时计算复杂度为 $O(nk^2)$。因为 k 的值远小于 $|X|$，因此 DenSOINN 算法的时间复杂度为 $O(nk|X|)$。

在空间复杂度方面，DenSOINN 模型需要存储节点属性和连接图结构。每个节点的属性占用 $n+2$ 个内存单元，因此存储节点的总内存空间为 $O(nk)$。如果连接图被保存为一个有向关联矩阵，则需要至多 $O(k^2)$ 个内存单元来存储。因

此，DenSOINN 的总空间复杂度为 $O(nk) + O(k^2)$。

3.5 实验验证

3.5.1 数据集、方法和评价标准

本节通过人工和真实数据集上进行了实验来评估 DenSOINN 算法的聚类质量。实验中使用的数据集的属性列在表格中.3.1。请注意，在将数据集提供给聚类算法之前，数据集未进行归一化。

实验中使用的人工数据集是低维的，以便可以将数据和聚类结果以可视化的方式展现出来。第一个人工数据集是一个二维数据集，通过从密度不同的两个的高斯分布中随机抽样创建，其名记为 Artificial I。第二个数据集是一个 Swiss Roll 数据集，其构造方法是：首先创建一个由 4 个二维高斯分布的混合组成的二维数据集，接着对数据点使用 Swiss Roll 映射 $(x, y) \longrightarrow (x\cos x, y, x\sin x)$，将其转换成三维空间中的点。上述人工数据集如图3.5所示。

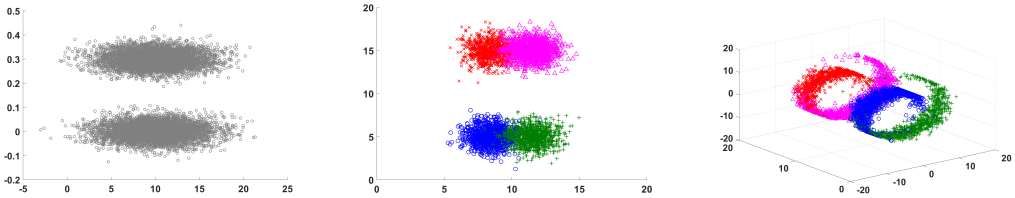
表 3.1 实验中使用的数据集

数据集	类别数量	样本数量	特征数量
Artificial I	2	10000	2
Swiss Roll	4	4000	3
London Air Quality	-	2976	2
Segment	7	2310	19
Pendigits	10	10992	16
HAR	6	10299	561
Usps	10	9298	256
Yale Face	10	5850	1200
MNIST	10	70000	780
Cover Type	7	581012	54
KDD99	23	4898431	72
URL Reputation	2	2396130	64

本组实验使用了四个数据流聚类算法作为比较基线，具体包括：G-Stream[51]，StrAP[58]，在线 K-Means[59]和 StreamKM++[57]。

除了 StreamKM++ 之外的比较算法是用 Matlab 实现的。StreamKM++ 实验使用了由论文[57]的作者提供的 C++ 代码进行，并从该代码中删除了大部分控制台输出，以准确记录聚类过程的计算时间。

为了从不同的方面比较这些算法，五个聚类验证度量被用来来评估实验结果，这些度量可以分为两组。



(a) Artificial I: 两个二维高斯分布的混合

(b) Swiss Roll: 4 个二维高斯分布的混合

(c) Swiss Roll: 三维 Swiss Roll 数据从二维高斯混合模型中转化得到

图 3.5　人工数据集。Artificial I **数据集由** 10000 **个从两个高斯分布中均匀采样得到的样本组成。**Swiss Roll **数据集是一个三维数据集，通过对从有** 4 **个分量的高斯混合模型中采样出的样本进行** swiss roll **映射得到。**

外部度量使用数据集中的样本标签作为“基准事实”，并将它们与聚类产生的样本标签进行比较。本章在实验中使用的外部度量是准确性、标准化互信息(Normalized Mutual Information, NMI)[79]和调整兰德系数(Adjusted RandIndex, ARI)[80]。

内部度量的计算仅依赖于输入数据和聚类划分。本章的实验中，轮廓系数(Silhouette coefficient, SC) 和量化误差被采用作为内部度量，但由于 SC 计算成本过高，在样本数量超过 50,000 的数据集上不使用此度量。均方根误差(RMSE) 被用来衡量量化误差。RMSE 是模型中每个数据点和其最近示例 (Den-SOINN/GStream 的神经节点和在线 K-Means/StrAP/StreamKM++ 的聚类中心) 之间平方欧几里得距离的均值的平方根。准确性、NMI、SC 的值范围为 $[0, 1]$，ARI 的值范围为 $[-1, 1]$，RMSE 的值范围为 $[0, +\infty]$。准确性、NMI、ARI 和 SC 的值越高，聚类性能越好，但 RMSE 越小越好。

本节还报告了各个算法找到的类簇数量，因为对于一个可以自行确定聚类数量的算法来说，聚类数量是否合理也是一个可以用来判断其性能的指标。本节也报告了各个算法的执行时间以比较其时间效率。

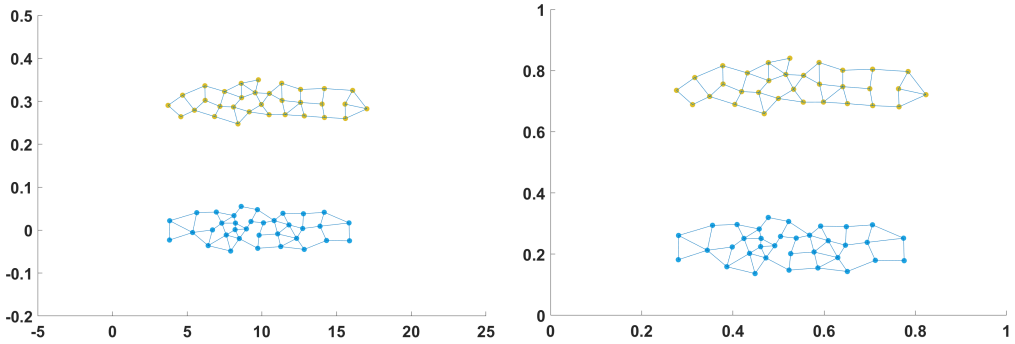
3.5.2　人工数据集上的实验

本小节用可视化的形式展示在人工数据集上进行的实验。这些图中，神经网络的节点和连接由彩色点和它们之间的线条来说明。属于同一个聚类簇的 Den-SOINN 节点被标记为相同颜色的点。

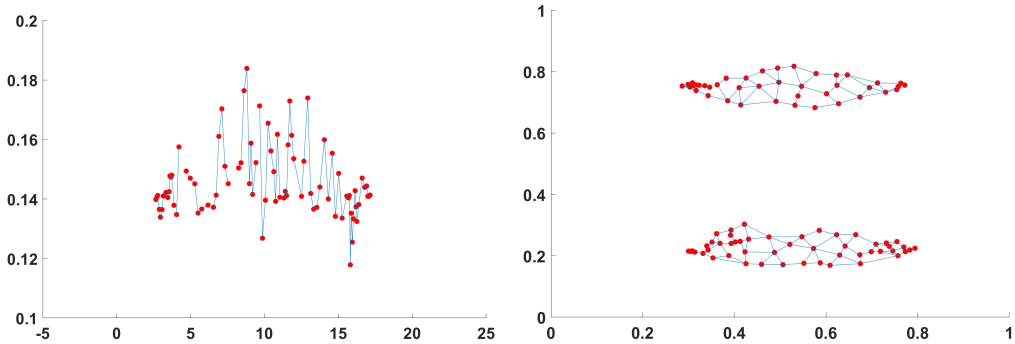
第一个实验在 Artificial I 数据集上进行，旨在验证自适应距离度量的效果。Artificial I 数据集由从两个高斯分布中均匀采样的 10000 个样本组成，如

图3.5(a)所示。为了模拟未经归一化的数据集，数据点的横坐标与纵坐标取值范围不同。DenSOINN 与 G-Stream[51]分别在原始数据和经过 Min-Max 归一化处理后的数据上进行实验并比较实验结果。DenSOINN 的参数设置为 $\gamma = 0.25$，DenoisingInterval=100，$\epsilon = 1$，$\alpha = 5$；G-Stream 使用[51]中建议的默认参数。DenSOINN 和 G-Stream 的最大节点数量没有限制。结果如图3.6所示。

从图中可以看到，DenSOINN 在两种环境下都表现良好，结果几乎相同。而 G-Stream 在未经归一化的原始数据上没能取得理想结果。这表明 DenSOINN 中的自适应距离度量能够在未经归一化的原始数据上胜任距离计算任务，其效果近似于在归一化数据上计算的欧氏距离。



(a) DenSOINN 在原始数据上的结果　(b) DenSOINN 在规范化数据上的结果

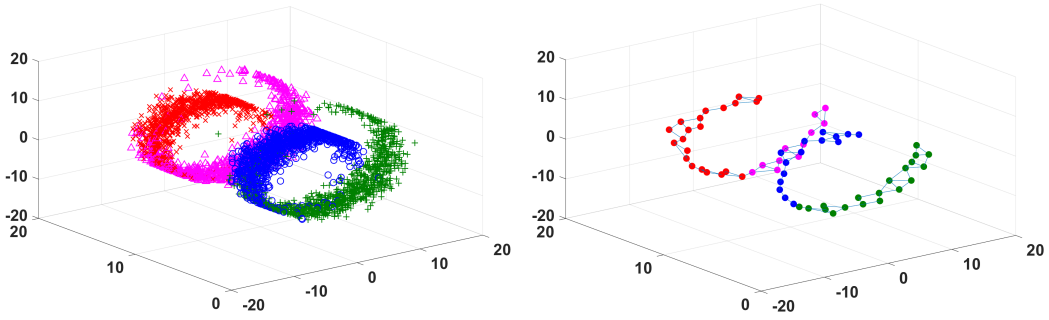


(c) G-Stream 在原始数据上的结果　(d) G-Stream 在规范化数据上的结果

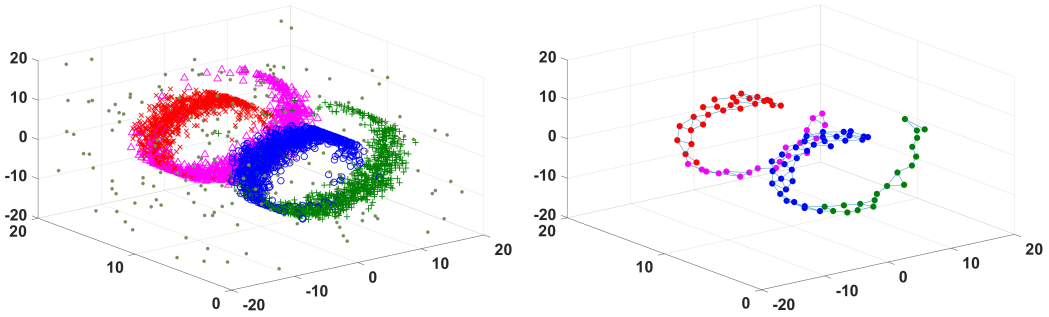
图 3.6　Artificial I 上的实验结果。DenSOINN 与 G-Stream 在原始数据和通过 Min-Max 归一化进行处理的数据上进行对比。神经网络的节点和连接由彩色点和它们之间的线条表示。G-Stream 中的每个节点都代表一个聚类，DenSOINN 的同颜色节点属于同一聚类。

第二个实验在 Swiss Roll 数据集上进行。数据中添加了一定比例的噪声以测试 DenSOINN 的鲁棒性。超参数设置为 $\gamma = 0.5, \mathit{DenoisingInterval} = 100, \epsilon = 0.5, \alpha = 6$。

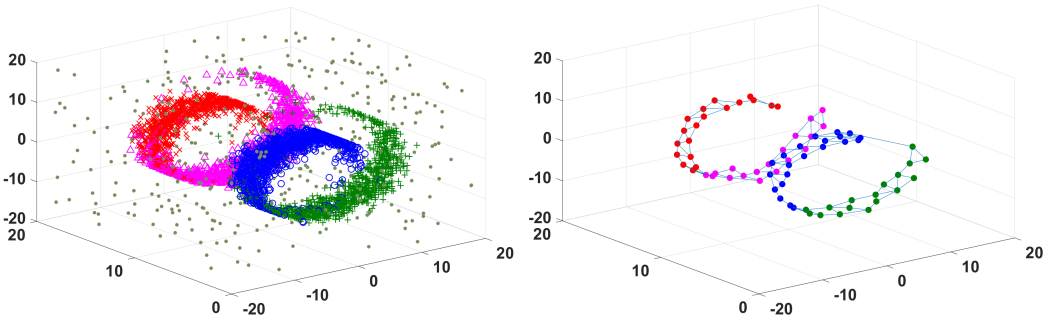
图3.7展示了这个实验的结果。从图中可以看出，随着噪声量的增加，DenSOINN 网络包含更多的噪声节点和连接，但聚类仍然很好地被分离开，并且聚



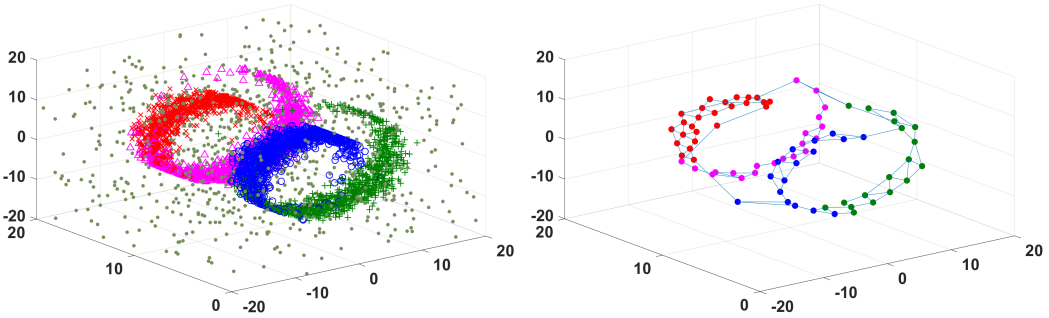
(a) 0% noise, NMI=0.7299



(b) 5% noise, NMI=0.7474



(c) 10% noise, NMI=0.7555



(d) 20% noise, NMI=0.7657

图 3.7　在添加了噪声的 Swiss Roll 数据集上的实验结果。左列展示了不同噪声等级的训练数据，右列展示了 DenSOINN 的学习结果

类质量没有受到影响。这种现象的原因是 DenSOINN 的聚类结果主要取决于聚类中心的选择。尽管网络在噪声等级高的时候包含了表示噪声数据的节点和连接，但因为噪声节点的密度很低，因此既没有被选为作为聚类中心也没有对其他节点的类别归属产生影响，聚类质量没有受到损失。

3.5.3　真实数据集上的实验

表 3.2　真实数据集上的实验结果

Dataset		DenSOINN	G-Stream	StrAP	online K-Means	StreamKM++
Segment	聚类数	6.6 ± 1.1	23 ± 2.2	42.8 ± 3.8	7^*	7^*
	Acc	0.6150 ± 0.0648	0.6442 ± 0.0300	$\mathbf{0.7801 \pm 0.0051}$	0.5815 ± 0.0327	0.5441 ± 0.0350
	ARI	$\mathbf{0.4214 \pm 0.0724}$	0.2802 ± 0.0378	0.1906 ± 0.0066	0.3652 ± 0.0398	0.3602 ± 0.0549
	NMI	$\mathbf{0.6176 \pm 0.0330}$	0.4713 ± 0.0291	0.5301 ± 0.0056	0.5207 ± 0.0379	0.5292 ± 0.0456
	SC	0.0415 ± 0.0366	0.1990 ± 0.0435	0.4402 ± 0.0088	0.4102 ± 0.0310	$\mathbf{0.4698 \pm 0.0086}$
	RMSE	60.19 ± 10.95	91.24 ± 2.25	$\mathbf{39.8 \pm 6.06}$	90.45 ± 1.70	77.8 ± 0.4834
	Time	0.3358 ± 0.0187	0.0546 ± 0.0077	3.4540 ± 0.6214	0.0201 ± 0.0022	0.0893 ± 0.0055
Pendigits	聚类数	9.1 ± 0.9944	72.6 ± 4.326	40 ± 0	10^*	10^*
	Acc	0.6974 ± 0.0596	0.8896 ± 0.0095	$\mathbf{0.8908 \pm 0.0010}$	0.6852 ± 0.0282	0.7188 ± 0.0307
	ARI	$\mathbf{0.5642 \pm 0.0568}$	0.2801 ± 0.0124	0.3702 ± 0.0129	0.5121 ± 0.0329	0.5437 ± 0.0312
	NMI	$\mathbf{0.7387 \pm 0.0347}$	0.6413 ± 0.0073	0.6760 ± 0.0073	0.6516 ± 0.0146	0.6800 ± 0.0126
	SC	0.3916 ± 0.0390	0.2001 ± 0.0256	0.3377 ± 0.0045	0.4279 ± 0.0174	$\mathbf{0.4667 \pm 0.0198}$
	RMSE	$\mathbf{35.32 \pm 0.70}$	50.65 ± 0.64	51.65 ± 0.74	69.6 ± 1.90	67.5 ± 0.44
	Time	2.1440 ± 0.0769	0.6545 ± 0.0204	2.8410 ± 0.1311	0.0861 ± 0.0075	0.5515 ± 0.0028
Usps	聚类数	5.7 ± 0.94	77.5 ± 4.478	69.3 ± 2.263	10^*	10^*
	Acc	0.5378 ± 0.0570	0.8464 ± 0.0090	$\mathbf{0.8531 \pm 0.0013}$	0.6633 ± 0.0309	0.6975 ± 0.0275
	ARI	0.4623 ± 0.0981	0.2941 ± 0.0435	0.2687 ± 0.0022	0.4796 ± 0.0362	$\mathbf{0.5046 \pm 0.0239}$
	NMI	$\mathbf{0.6283 \pm 0.05019}$	0.5724 ± 0.0096	0.5712 ± 0.0020	0.5764 ± 0.0211	0.5943 ± 0.0162
	SC	0.1121 ± 0.0686	0.1080 ± 0.0212	0.1614 ± 0.0133	0.2318 ± 0.0170	$\mathbf{0.2413 \pm 0.0106}$
	RMSE	$\mathbf{4.53 \pm 0.04}$	5.38 ± 0.02	5.62 ± 0.02	6.14 ± 0.02	6.09 ± 0.01
	Time	6.261 ± 0.290	0.683 ± 0.028	4.759 ± 0.443	0.2267 ± 0.0039	8.899 ± 0.596
HAR	聚类数	4.8 ± 0.9189	79.7 ± 4.322	40.7 ± 2.45	6^*	6^*
	Acc	0.6038 ± 0.0767	$\mathbf{0.8331 \pm 0.0096}$	0.7873 ± 0.0158	0.5849 ± 0.0599	0.6064 ± 0.0128
	ARI	$\mathbf{0.4798 \pm 0.0873}$	0.1169 ± 0.0063	0.2075 ± 0.0110	0.4283 ± 0.0703	0.4505 ± 0.0254
	NMI	$\mathbf{0.6252 \pm 0.0599}$	0.4743 ± 0.0054	0.4955 ± 0.0091	0.5781 ± 0.0530	0.5996 ± 0.0059
	SC	0.2082 ± 0.0935	0.0321 ± 0.0074	0.0429 ± 0.0069	$\mathbf{0.2712 \pm 0.0800}$	0.2414 ± 0.0237
	RMSE	$\mathbf{3.63 \pm 0.06}$	3.71 ± 0.01	4.63 ± 0.02	4.30 ± 0.09	4.22 ± 0.01
	Time	9.573 ± 1.565	1.071 ± 0.045	17.26 ± 0.657	0.408 ± 0.006	19.326 ± 0.714
Yale Face B	聚类数	8 ± 1.63	49.1 ± 3.14	82.4 ± 3.13	10^*	10^*
	Acc	0.6328 ± 0.1091	0.9128 ± 0.0234	$\mathbf{0.9412 \pm 0.0026}$	0.6431 ± 0.0659	0.7672 ± 0.0542
	ARI	0.5907 ± 0.1428	0.4331 ± 0.0273	0.2504 ± 0.0177	0.4821 ± 0.0901	$\mathbf{0.6272 \pm 0.0732}$
	NMI	0.7746 ± 0.0671	0.7317 ± 0.0151	0.6770 ± 0.0045	0.6979 ± 0.0459	$\mathbf{0.7931 \pm 0.0383}$
	SC	0.1660 ± 0.0696	0.1265 ± 0.0242	0.2189 ± 0.0103	0.2282 ± 0.0379	$\mathbf{0.2656 \pm 0.0084}$
	RMSE	$\mathbf{781.9 \pm 20.65}$	1077 ± 14.41	1009 ± 6.92	1243 ± 34.2	1204 ± 3.39
	Time	30.022 ± 3.553	1.562 ± 0.042	9.365 ± 3.317	0.854 ± 0.018	26.052 ± 1.538
MNIST	聚类数	11.4 ± 1.83	91.4 ± 6.45	94 ± 6.55	10^*	10^*
	Acc	0.6156 ± 0.0742	$\mathbf{0.7907 \pm 0.0070}$	0.7684 ± 0.0266	0.5490 ± 0.0329	0.5992 ± 0.0081
	ARI	$\mathbf{0.3996 \pm 0.0891}$	0.1554 ± 0.0099	0.1428 ± 0.0033	0.3419 ± 0.0296	0.3853 ± 0.0060
	NMI	$\mathbf{0.5763 \pm 0.0428}$	0.5004 ± 0.0038	0.4601 ± 0.0107	0.4665 ± 0.0234	0.4998 ± 0.0114
	RMSE	$\mathbf{1337 \pm 9.5}$	1436 ± 4.16	1695 ± 4.83	1607 ± 7.50	1608 ± 1.62
	Time	191.9 ± 22.06	14.14 ± 0.24	2215 ± 808.3	5.49 ± 0.076	146.622 ± 1.570
CoverType	聚类数	14.1 ± 1.72	84.6 ± 4.74	34.3 ± 1.15	7^*	7^*
	Acc	$\mathbf{0.5637 \pm 0.0145}$	0.5626 ± 0.0042	0.5449 ± 0.0051	0.4917 ± 0.0037	0.4919 ± 0.0024
	ARI	$\mathbf{0.0284 \pm 0.0306}$	0.0050 ± 0.0003	0.0093 ± 0.0013	0.0035 ± 0.0029	-0.0048 ± 0.0018
	NMI	$\mathbf{0.1461 \pm 0.0296}$	0.0988 ± 0.0015	0.0969 ± 0.0046	0.0740 ± 0.0041	0.0731 ± 0.0012
	RMSE	$\mathbf{459.5 \pm 15.55}$	479.4 ± 27.39	535.4 ± 5.50	894 ± 9.95	880.6 ± 0.72
	Time	258 ± 18.76	71.34 ± 3.69	8176 ± 368.8	5.888 ± 0.16	77.364 ± 0.367
KDD99	聚类数	4.6 ± 1.14	36.6 ± 6.18	33 ± 3.46	23^*	23^*
	Acc	0.9641 ± 0.0189	$\mathbf{0.9747 \pm 0.0186}$	0.9603 ± 0.0017	0.9538 ± 0.0264	0.7361 ± 0.1126
	ARI	$\mathbf{0.8685 \pm 0.1065}$	0.6503 ± 0.0052	0.6955 ± 0.0022	0.5280 ± 0.0446	0.4967 ± 0.3807
	NMI	$\mathbf{0.8075 \pm 0.0707}$	0.6572 ± 0.0036	0.6798 ± 0.0041	0.6144 ± 0.0075	0.4760 ± 0.2991
	RMSE	$9.888 \times 10^5 \pm 11.49$	$9.879 \times 10^5 \pm 625.5$	$9.887 \times 10^5 \pm 10.35$	$9.731 \times 10^5 \pm 5109$	$\mathbf{1880 \pm 47.43}$
	Time	153.3 ± 10.97	68.77 ± 4.156	7670 ± 258	16.17 ± 0.097	72.307 ± 0.730

本小节在真实世界数据集上将 DenSOINN 和其他聚类算法进行对比。实验使用了 KDD99 的 10% 采样集，而不是完整数据集。在每个实验中，每个算法

在给定数据集上重复运行 10 次。数据集被组织成随机顺序的输入序列，每个样本只能被学习一次。模型在学习了完整的输入序列后进行聚类。

比较算法的参数是相关论文作者提供的代码中的默认参数。基于 K-Means 的算法中的聚类数设置为数据集中的真实类数。DenSOINN 和 G-Stream 的最大节点数量没有限制。DenSOINN 的参数设置为：$\gamma = 0.25$，$DenoisingInterval = 200$，$\epsilon = 0.5$，$\alpha = 5$。其中的一个特例是 MNIST，在这个数据集上 $\alpha = 3.5$。

表3.2报告了聚类质量的平均值和标准差，以及聚类数和运行时间（秒）。根据外部度量标准，DenSOINN 在大多数实验中实现了最高的平均 NMI 和 ARI。它的准确率 (Acc) 低于 G-Stream 和 StrAP，但部分原因是 DenSOINN 生成的聚类要少得多。准确率和聚类数量通常成反比，当聚类数量超过实际类别的数量时，增加聚类数可以提高准确性。例如，在每个聚类仅包含一个样本的极端情况下，准确性将为 1。总体而言，DenSOINN 学习的类簇与标记数据集中的真实类别最相似。值得注意的是，在 KDD99 和 Segment 上，DenSOINN 表现明显优于其他算法。原因是在这些数据集上有一部分特征的取值范围远远超过其他特征，而 DenSOINN 中的距离度量可以应对这个问题。

在内部度量标准方面，在所有实验中，DenSOINN 的轮廓系数都不如基于 K-Means 的算法好，但在大多数实验中，DenSOINN 的 RMSE 最小。这种现象说明 DenSOINN 的节点可以形成输入数据的良好代表集，因此样本与其代表节点之间的 RMSE 更小。但是与 K-Means 算法学习到的聚类相比，聚类在特征空间中的形状不太紧密。这是因为 DenSOINN 是使用基于密度的聚类方法来形成聚类的。数据集中的真实分类不一定在特征空间上是致密的，而可能具有任意形状。轮廓系数以及大多数内部度量标准强调类内相似性和类间差异，因此这些标准自然倾向于学习紧密簇的聚类算法。基于密度的聚类算法则榆次相反，强调簇内密度连接和簇间密度分离。类簇可以是任意形状，因此使用这些方法进行聚类得到的轮廓系数相对较差。

通过表3.2 中每个算法报告的聚类数，可以看到 DenSOINN 在大多数实验中报告的聚类数接近数据集中的真实类数，但在 KDD99 和 CoverType 这种不平衡数据集上报告的类数误差较大。相比之下，G-Stream 和 StrAP 报告的聚类数量太多了。

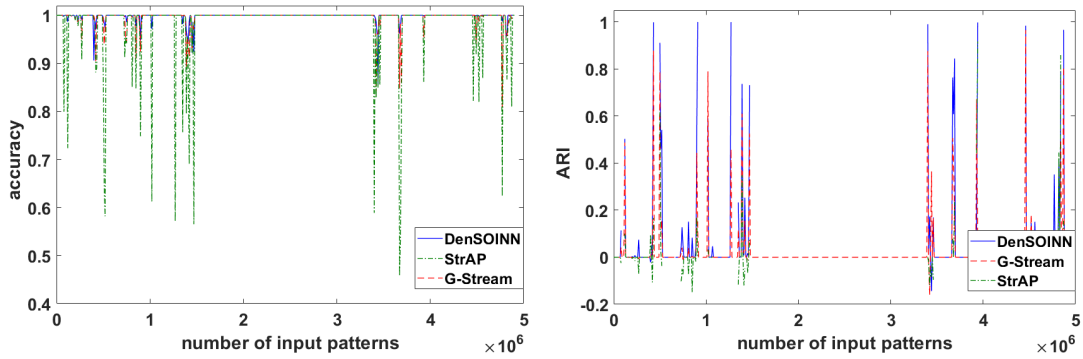
在算法的执行时间方面，在线 K-Means 算法是最高效的，因为它的在每个

学习轮次中进行的距离计算次数最低。DenSOINN 和 G-Stream 的效率低于在线 K-Means，因为两个网络中的节点数高于 K-Means 的簇数 k。G-Stream 在小数据集上比 DenSOINN 效率高很多，但是在像 KDD99 和 CoverType 这样的大数据集上这个优势并不明显。因为 DenSOINN 的节点数量在开始时增长很快但之后保持稳定，而 G-Stream 的大小随着输入模式的数量近似线性增长。基于目前的参数设置，G-Stream 的网络节点数量比 DenSOINN 少。StrAP 比其他方法慢，尤其是在大型和高维数据集上。StreamKm++ 在大多数数据集中也比 DenSOINN 快，但主要原因是 C++ 程序相对于 Matlab 程序的运行速度优势。

3.5.4　演化数据流的实验

本小节在三个大规模演化数据流上进行了实验，并展示了学习过程中的阶段性聚类结果。这些实验将 DenSOINN 与 G-Stream、StrAP 进行了比较。

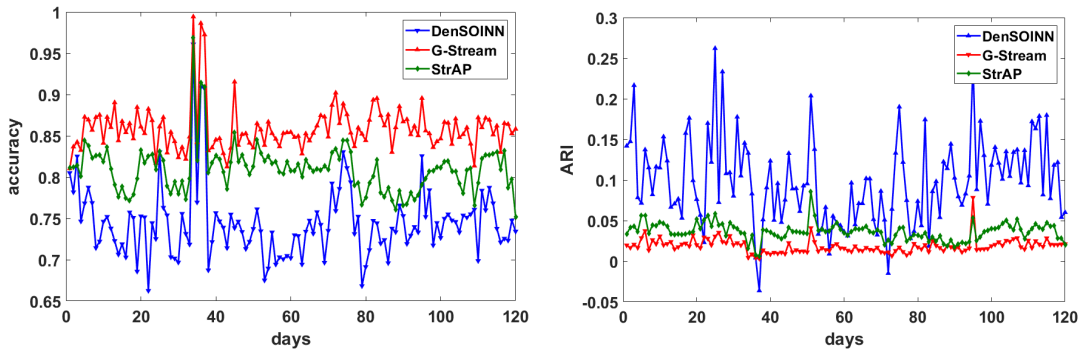
第一个实验是在 KDD99 数据集上进行的，数据输入顺序是原数据集中的样本顺序。数据流被分成 500 个阶段，每个阶段有 10000 个样本。在每个阶段之后，算法进行聚类，并使用本阶段内学习的数据评估聚类质量。DenSOINN 和 G-Stream 都有一个参数来控制最大节点数，本实验中设置为 1000。本组实验使用准确率 (Accuracy) 和 ARI 来评估本实验中的聚类质量。准确率和 ARI 随时间的比较如图 3.8 所示。



(a) Accuracy　　(b) ARI

图 3.8　DenSOINN、G-Stream 和 StrAP 在 KDD99 数据流上的阶段性结果

从图中可以看出，DenSOINN 的准确率和 ARI 优于 G-Stream 和 StrAP。KDD99 数据流在大多数学习阶段中都仅包含来自单个类的数据，此时所有结果中的准确度均为 1，ARI 均为 0。在数据流中有多个类的阶段，准确率会下降，



(a) Accuracy　　(b) ARI

图 3.9　DenSOINN、G-Stream and StrAP 在 URL Reputation 数据流上的阶段性结果

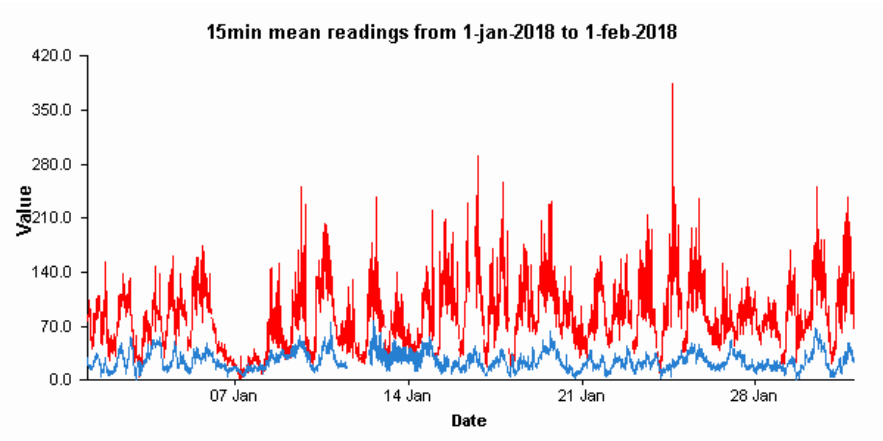


图 3.10　伦敦空气质量监测数据。红线表示二氧化氮的量，蓝线表示 PM10 颗粒物的量。

ARI 会升高。可以观察到，DenSOINN 的准确性和 ARI 在大多数情况下都优于比较方法。

第二个实验是在 URL Reputation 数据集[81] 上进行的。该数据集是 ICML-09 URL 数据的 120 天子集，包含 240 万个样本和 320 万个特征。实验中使用了 64 个数字特征。学习每天的数据后给出聚类请求，然后评价聚类质量。与之前的实验相同，DenSOINN 和 G-Stream 的最大容量被限制在 1000，评估聚类质量的度量是准确性和 ARI。每个阶段的准确率和 ARI 的比较如图 3.9 所示。在这个实验中，与 G-Stream 和 StrAP 相比，DenSOINN 的准确性较差，而其 ARI 是最好的。出现这种现象的原因是 URL Reputation 数据集只包含两个类，但 G-Stream 和 StrAP 都报告了太多的聚类。相比之下，DenSOINN 仅有个位数的聚类数量，导致其准确性不高，但报告的数据区分与样本的分类更相似。

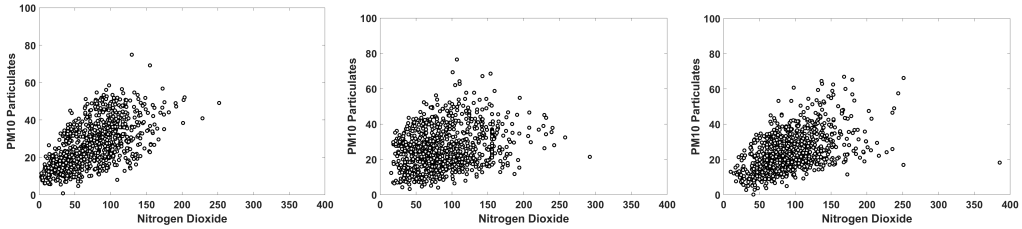
第三个实验使用伦敦空气质量数据[82] 进行。实验数据是从网络上获取的 Westminster - Marylebone Road 站点从 1-Jan-2018 到 1-Feb-2018 的监测数据。监测站每 15 分钟测量一次空气质量，因此数据集包含 2976 个样本。二氧化氮（$\mu g/m^3$）和 PM10 颗粒物（$\mu g/m^3$）的量被用作数据特征，如图 3.10 所示。数据集分为三个子集，每个子集包含大约 10 天内收集的数据。在本实验中，由于数据集较小且维数较低，因此更改了 G-Stream 和 StrAP 的参数，G-Stream 中设置节点创建间隔为 100，StrAP 中设置重启条件为 $MaxCache = 300$。数据点和结果如图 3.11 所示。数据集是未标记的，因此使用内部度量轮廓系数 (SC) 和 RMSE 来评估聚类质量，并将结果进行可视化。

从图中可以看出，DenSOINN 的网络结构很好形成了输入数据的代表集。DenSOINN 在第一个时期报告了三个聚类，在其他时期报告了四个聚类。相比之下，G-Stream 的节点分布在输入数据中心附近的一个小区域。StrAP 的样本也形成了很好的数据代表集，但代表点的数量相当大。DenSOINN 的轮廓系数仍然比 G-Stream 和 StrAP 差，但它的 RMSE 是最小的。

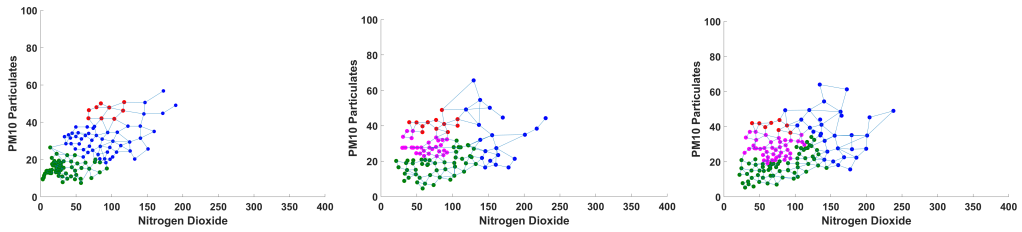
3.5.5 参数设置和灵敏度分析

本小节通过实验 DenSOINN 分析超参数的影响。DenSOINN 中共有五个参数，根据其作用可分为三组。

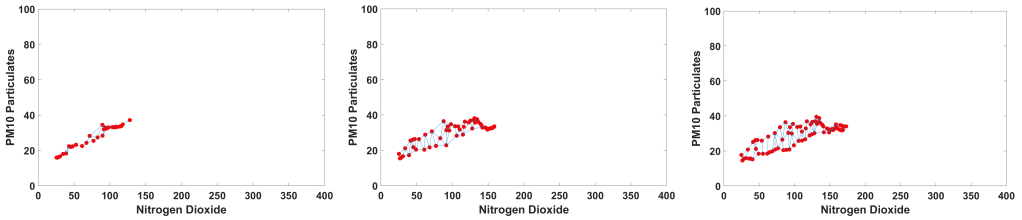
第一组由 γ、$DenoisingInterval$ 和 ϵ 组成。$DenoisingInterval$ 和 ϵ 分别



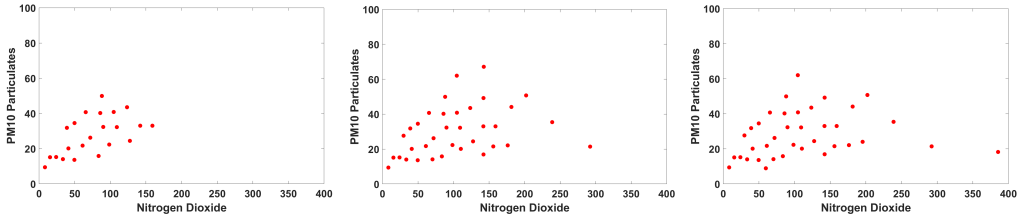
(a) 数据分布



(b) DenSOINN 学习结果, $SC = (0.1537, 0.0882, -0.1389)$, $RMSE = (4.34, 5.09, 6.07)$



(c) G-Stream 学习结果, $SC = (-0.0207, 0.1372, 0.1432)$, $RMSE = (13.44, 13.59, 11.77)$



(d) StrAP 学习结果, $SC = (0.4915, 0.4494, 0.4370)$, $RMSE = (7.37, 9.53, 6.28)$

图 3.11　伦敦空气质量数据和聚类结果。从左到右三列依次显示每个 10 天的数据和学习结果。

控制去噪的频率和强度。γ 影响最近未更新的边的生存时间，这进一步影响去噪过程中节点的删除。这些参数共同影响节点数，节点数对执行时间和内存使用具有决定性作用。这些参数的较为合理的设置范围分别是 $\gamma \in [0.1, 1]$、$DenoisingInterval \in [50, 200]$ 和 $\epsilon \in [0.2, 1]$。设置较小的 γ 将为 DenSOINN 提供更好的拓扑保持能力，在具有良好分离的集群和小噪声的数据集上工作良好，如本节中使用的人工数据集。然而，现实世界的数据集通常包含重叠的簇和有噪声的数据，因此有必要及时去除过时的边缘。设置较大的 $DenoisingInterval$ 会降低去噪频率，但如果 $DenoisingInterval$ 太大或 ϵ 太小，网络将包含许多不必要的节点，导致计算速度降低。反之，如果 $DenoisingInterval$ 太小或 ϵ 太大，则会删除太多节点，网络结构会变得不稳定。本章的真实数据实验中设置 $DenoisingInterval = 200$、$\epsilon = 0.5$，因为它们在大多数真实世界数据集上运行良好，但根据输入数据的特性调整这些参数可以提高聚类质量或时间效率。在学习数百个样本的小型数据集时，推荐设置 $DenoisingInterval = 200$ 和 $\epsilon = 0.25$。如果输入数据集非常大，默认参数下的运行时间过长，此时设置 $DenoisingInterval = 100$ 和 $\epsilon_1 = 1$ 可以大大减少节点数，加快训练速度。

第二组实验在 γ、$DenoisingInterval$ 和 ϵ 的不同设置下对真实世界的数据集进行了实验。实验中使用了三组 $DenoisingInterval$ 和 ϵ，分别对应三个级别的去噪。每组 $DenoisingInterval$ 和 ϵ 分别使用 0.25、0.5 和 1 三个 γ 值。本组所有实验中都设置 $MaxNode = 1000$ 和 $\alpha = 5$。在每个实验中，DenSOINN 在给定数据集上使用每个参数设置重复运行 10 次，平均结果如表 3.3 所示。

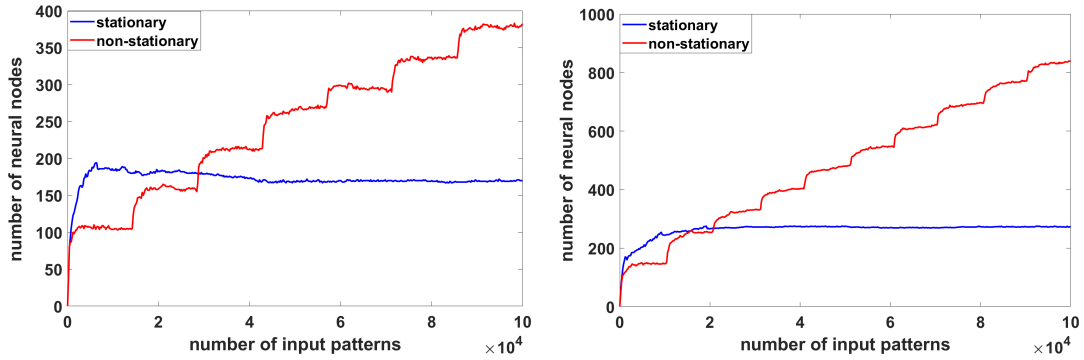
可以观察到，设置 $DenoisingInterval = 200$，$\epsilon = 0.5$ 在大多数实验中得到了最好的结果，但在 MNIST 上数据没有充分分离，导致结果不够理想。设置 $\epsilon = 0.25$ 可能会导致节点过多和训练时间变长，而且其结果明显比默认参数差。设置 $DenoisingInterval = 100$ 和 $\epsilon = 1$ 可以大大减少节点数量和训练时间，但代价是聚类质量降低。更改 γ 只会轻微影响结果，但在大多数比较中，设置 $\gamma = 0.25$ 会导致更少的神经节点和更好的聚类质量。

这组实验中另一个可观察到的现象是 DenSOINN 中的节点数量不依赖于训练数据集的大小。例如，学习 KDD99 产生的节点数比任何其他数据集都少得多。事实上，DenSOINN 能够在短时间内生成合适数量的节点来表示数据分布，并且网络规模保持稳定，除非数据分布发生变化。但是，它对数据分布的变化很

表 3.3　DenSOINN 在不同参数下的实验结果

数据集	指标	$StreamSpeed = 200$, $\epsilon = 0.25$			$StreamSpeed = 200$, $\epsilon = 0.5$			$StreamSpeed = 100$, $\epsilon = 1$		
		$\gamma = 0.25$	$\gamma = 0.5$	$\gamma = 1$	$\gamma = 0.25$	$\gamma = 0.5$	$\gamma = 1$	$\gamma = 0.25$	$\gamma = 0.5$	$\gamma = 1$
Segment	Clusters	27	27.8	31.6	6.6	7.2	9	3.2	4.4	5.4
	Acc	0.8377	0.8390	0.8449	0.6518	0.5930	0.6115	0.4319	0.5374	0.5377
	ARI	0.3464	0.3254	0.3084	0.4731	0.3900	0.3982	0.2648	0.3663	0.3561
	NMI	0.6142	0.6083	0.6031	0.6341	0.5993	0.5996	0.5024	0.6049	0.5762
	神经元数量	770.2	786.4	818	149	162	162.6	61.6	75.6	73.2
Pendigits	Clusters	17.5	19	18.8	9.3	12.6	12	8	7.6	8.3
	Acc	0.8302	0.8214	0.8531	0.7618	0.8184	0.8398	0.684	0.6655	0.6709
	ARI	0.5995	0.5728	0.6149	0.6375	0.6532	0.6832	0.5012	0.4903	0.5484
	NMI	0.7444	0.7413	0.7649	0.7697	0.7877	0.7906	0.7301	0.7157	0.7156
	神经元数量	801.7	808.3	825.3	244.3	256	278.7	81.67	77.33	69.33
Usps	Clusters	17.6	17	16.8	6.4	6	6.8	2.8	3.2	3.8
	Acc	0.7785	0.7727	0.7620	0.5661	0.5662	0.5673	0.3681	0.4052	0.4336
	ARI	0.4876	0.4965	0.4968	0.4426	0.4390	0.3871	0.2309	0.2918	0.2776
	NMI	0.6923	0.6838	0.6914	0.6636	0.6467	0.6358	0.4571	0.5110	0.4882
	神经元数量	743.5	729.1	732.6	223.5	232	238	85	82.6	77.2
HAR	Clusters	16.6	16.2	18.2	6.4	6.2	4.4	2.2	2.4	2
	Acc	0.7717	0.7649	0.7721	0.6579	0.6566	0.5734	0.3600	0.4259	0.3545
	ARI	0.3892	0.3879	0.3975	0.4878	0.4782	0.4497	0.3279	0.3941	0.3299
	NMI	0.5819	0.5832	0.5895	0.6116	0.6072	0.6030	0.5389	0.6057	0.5468
	神经元数量	430.3	424.6	400.2	93	104.7	69.3	29.4	25	19.2
Yale Face B	Clusters	19.8	20.4	23.2	8.4	9.6	10.8	6.2	6.2	7
	Acc	0.8750	0.8486	0.8753	0.6670	0.7333	0.7393	0.4549	0.4643	0.5416
	ARI	0.6021	0.5732	0.5925	0.5797	0.6459	0.6351	0.2935	0.2681	0.3762
	NMI	0.8011	0.7905	0.7971	0.7992	0.8347	0.8356	0.6152	0.6415	0.7056
	神经元数量	891.1	893.6	889.9	217.5	216.6	222.8	90.1	97.4	86.4
MNIST	Clusters	13.8	15.6	13.8	5.4	6.4	4.4	1.2	2.2	2.6
	Acc	0.5329	0.5974	0.6134	0.3612	0.3776	0.3631	0.1321	0.2096	0.2494
	ARI	0.3248	0.3741	0.4048	0.2063	0.2111	0.1844	0.0100	0.0619	0.0817
	NMI	0.5572	0.5910	0.5994	0.4173	0.4321	0.4115	0.04439	0.1743	0.2704
	Neural nodes	615.4	593.2	580.2	237.8	229.6	186.6	64.4	58.6	40.8
CoverType	Clusters	37.6	39.2	44.6	13.4	15.2	18.2	2	2.4	3.6
	Acc	0.6284	0.6320	0.6378	0.5621	0.5766	0.5946	0.4898	0.501	0.5036
	ARI	0.0404	0.0398	0.0422	0.0178	0.0300	0.0452	0.0121	0.0114	0.0003
	NMI	0.1787	0.1794	0.1832	0.1448	0.1488	0.1573	0.0381	0.0438	0.0446
	神经元数量	725.2	827.6	784	218.2	214.2	196	47.2	39.4	20.8
KDD99	Clusters	19.8	20.4	23.2	8.4	9.6	10.8	6.2	6.2	7
	Acc	0.9751	0.9666	0.9701	0.9716	0.9691	0.8333	0.8206	0.7391	0.7244
	ARI	0.8223	0.7515	0.7145	0.7705	0.8160	0.6456	0.7669	0.5720	0.3965
	NMI	0.7420	0.7134	0.7110	0.7637	0.7812	0.6385	0.7441	0.5370	0.3869
	神经元数量	93	85	58.2	51.6	35	8.9	15.3	9	4

敏感。另一个实验测试了 Pendigits 和 Segment 在数据分布稳定和不稳定时的网络生长过程。实验中的训练数据是从原始数据集中随机抽取的 100,000 个样本。在模拟数据分布不稳定的情况时，样本的输入序列根据类别标签进行排序。本实验中参数设置为 $\gamma = 0.25$，$DenoisingInterval = 200$，$\epsilon = 0.5$，$\alpha = 5$，不限制网络容量。网络规模的增长如图 3.12 所示。

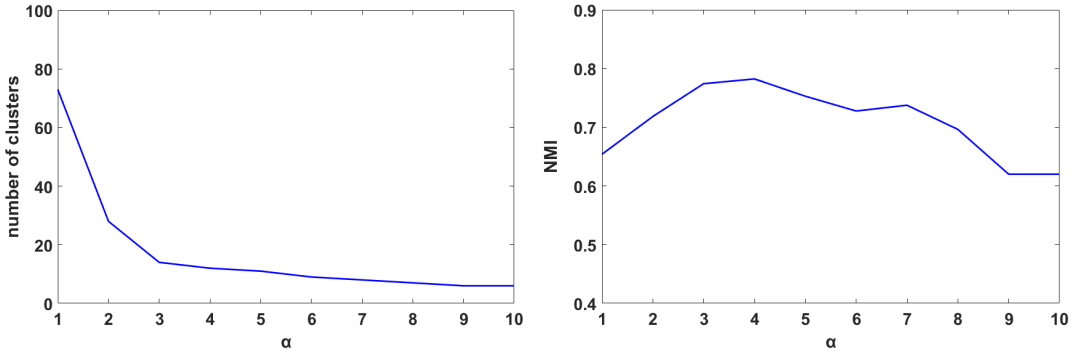


(a) Segment 数据集上的网络生长　　(b) Pendigits 数据集上的网络生长

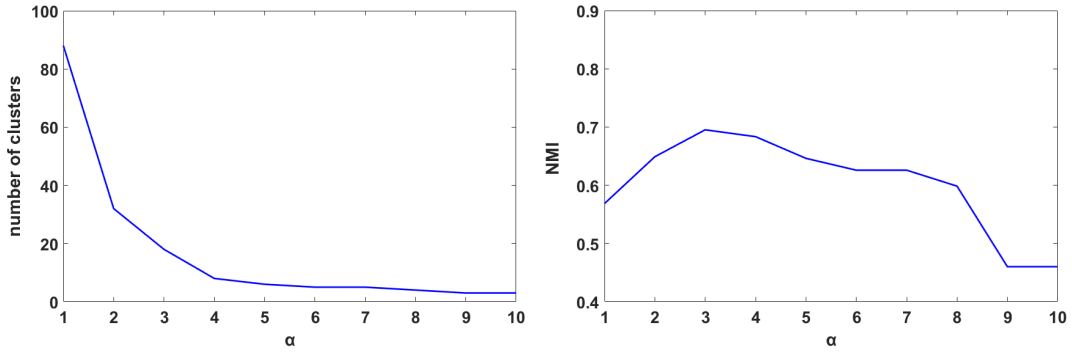
图 3.12　Segment 和 Pendigits 数据集上的网络节点数与输入样本数。绿线是数据分布稳定时的网络生长。蓝线是数据分布不稳定时的网络生长。

从图3.12可以看出，网络节点的数量在开始时增长很快，之后在数据分布稳定时保持稳定。在不稳定环境中，每次样本的类别发生变化时网络的节点数量都会明显增加。本章所提出的方法面向的是增量学习，其中旧数据和新数据被认为同样重要，因此 DenSOINN 在保持旧结构的同时学习新知识。但是，在数据分布一直在变化的极端情况下，网络会不断增加新的节点，网络规模会变得非常大。因此参数 $MaxNodes$ 控制网络的最大容量，以防输入数据流极度不稳定导致计算成本不可控。

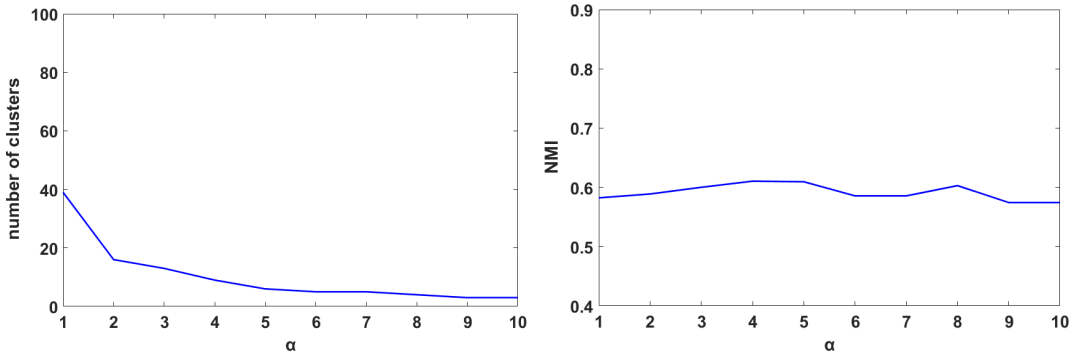
最后一个参数 α 控制网络的每个连接子图中的聚类中心数。α 的合理范围是 $\alpha \in [2, 10]$。α 对聚类结果有关键影响，但它的值并不难设置。本节在真实世界数据集上进行聚类时，α 固定为 $\alpha = 5$，由此产生的聚类数通常接近数据集中的标签类数。最后一组实验使用 α 的不同设置下在 Pendigit、Usps 和 Segment 数据集上进行，记录了每个实验中的聚类数量，并通过 NMI 评估聚类质量。结果如图 3.13 所示，聚类的数量随着 α 的增加而减少，直到达到最小值，即网络图结构中连通分量的数量。可以在右栏中看到，设置 $\alpha \in [3, 6]$ 时聚类质量较高。



(a) Pendigits



(b) Usps



(c) Segment

图 3.13　α 不同设置下的实验结果。左列展示了聚类数量与 α 的关系，右列展示了 NMI 与 α 的关系。

3.6　本章小结

本章介绍了一种新的竞争性神经网络，称为基于密度的自组织增量神经网络 (DenSOINN)。该方法面向无监督增量学习，能够学习出大规模数据流的代表集，并发现其中的聚类。DenSOINN 还应用了竞争性 Hebbian 学习来学习输入数据的拓扑表达式。面向增量学习中难以对数据整体进行规范化的问题，本文采用了一种自适应距离度量，使样本对之间的距离近似于在规范化数据上计算的欧几里得距离，从而使 DenSOINN 能良好地应用于未经规范化的原始数据。通过结合传统的基于密度连通性的聚类和基于密度峰值的聚类，DenSOINN 能够发现任意同时找到合适数量的任意形状的簇。本章的研究成果形成了一篇会议论文与一篇期刊论文，分别发表于国际会议 IJCNN-16 与期刊《Neural Networks》。

然而，DenSOINN 算法依旧存在一些缺点。真实世界数据集的实验结果表明，与基于 K-Means 的算法相比，DenSOINN 的聚类质量对样本的输入序列更敏感。此外，算法中的超参数较多，超参数的最优值不易设定。本文的后续研究工作将在努力提高 DenSOINN 的稳定性，并尝试提供一种自动方法来寻找模型中的最优参数。

第四章　基于原型与深度网络加固的类增量学习算法

灾难性遗忘现象是阻碍人工神经网络实现增量学习的主要因素。经过训练的神经网络在学习新知识之后，其中保存的旧有记忆会出现严重的丢失。现有克服灾难性遗忘的方法主要面向多任务持续学习问题，即连续训练一个神经网络模型执行多个任务。类增量学习将单一分类任务中不同类别的训练数据分为多个部分，让模型在多个训练阶段中分别学习一部分类别。在类增量学习问题的解决方案上，大多数方法仍然依赖于存储过去的训练数据用来在新的训练阶段进行回放。这种训练机制依赖辅助训练数据的支持，如存储的样本或由生成模型生成的伪样本。本章研究一类更严格的类增量学习问题，即如何在不使用辅助数据或生成模型的情况进行类增量学习，分析这种增量学习环境种存在的难题，并提出一种新方法来应对这些挑战。本章的各节内容安排如下：第 1 节介绍本章的研究意义，第 2 节介绍相关工作。第 3 节描述了类增量学习问题以及受限制的学习系统需要面对的主要困难。第 4 节提出了受限类增量学习的解决方案, 第 5 节进行实验验证，第 6 节总结本章内容。

4.1　引言

近年来，深度神经网络（DNN）的快速发展对人工智能的各个研究领域都产生了巨大的影响[39,41]。然而，关于深度神经网络的研究工作依然面临着一些困难问题的挑战。其中的难题之一是如何使经过训练的神经网络能够进一步从数据流中学习的新知识，就像人类能够不断从日常生活中学习新概念一样。这种学习场景称为类增量学习，其中不同类别的训练数据被分成批次逐渐输入到学习系统，并且新的类别随着时间的推移逐步出现[83]。类增量学习要求分类模型能够持续不断更新，既学习到新类中的知识，同时又保留对以前学习过的类别 (基类) 知识的记忆[84]。然而，灾难性遗忘现象[85] 是神经网络实现增量学习的主要障碍——经过训练的神经网络很难在不导致基类预测性能下降的情况下

学习新类，因为网络中关键的权重发生了变化，以适应新的训练目标[86]。

灾难性遗忘问题的一种简单解决方案是复习旧数据[87]：当网络学习新的输入数据时，它可以利用一些被人为保留的基类训练数据来巩固它的记忆。复习法可以显着缓解灾难性遗忘现象，但也需要额外的存储空间和训练时间。此外，复习法在认知科学理论上并不合理，因为人类在学习新知识时并不总是需要重新访问旧资源。然而，最近的大多数类增量学习方法仍然依赖于复习[88-89]。只有少数方法不需要存储基类的训练数据[90-91]

本章研究一种更严格的类增量学习算法，在学习过程中避免使用辅助数据和生成模型，以探究类增量学习算法需要面对的基本障碍，并开发一种具有更好时间空间的方法。本章分析了受限类增量学习的两个主要障碍：目标抑制和开放空间风险，并提出了一种名为基于原型的网络加固 (Prototype based Network Consolidation, PBNC) 的新方法作为解决方案。本章内容的主要研究贡献如下：

- 分析了类增量学习中的目标抑制现象。这一现象使得 Elastic Weight Consolidation (EWC)[86] 等多任务持续学习方法在类增量学习中失效了。本章从理论上证明，若要在不访问辅助数据的情况下实现类增量学习，对分类层神经元进行任务分解是必要的。
- 讨论类增量学习中的开放空间风险问题。虽然类增量学习与开集识别问题存在一些交叉研究，但在以往的类增量学习研究工作中很少涉及开放空间风险。
- 针对以上两个问题，引入了一种基于原型的类增量学习方法，它采用端到端的训练机制来同时训练特征提取器和类的原型。

4.2 相关研究工作

目前的深度学习研究领域已有很多关于类增量学习和灾难性遗忘现象的研究成果。本节分析这些成果的贡献和局限性。

4.2.1 基于复习法的类增量学习

正如在本章第一节中介绍的，大多数类增量学习方法都建立在复习法的基础之上。根据复习先前学习的数据的不同机制，这些类增量学习方法可以分为

两组。

第一组方法在每个训练阶段采样少量训练样本，将其存入一个样本集中。论文[87] 中首次提出了复习的概念，即将旧样本重新输入给模型以巩固模型的记忆。ICaRL[89] 使用深度神经网络来学习特征表示，用最近类均值法进行分类。论文[88]提出了一个端到端训练的增量学习神经网络来同时学习特征表示和分类器。Class Matrix Sketching[92]使用低维矩阵作为过往训练数据的总结。最近的方法[93]重点关注如何提高内存效率，它存储了样本的特征表示而不是其原始数据。总而言之，这些方法有效利用被存储的训练样本并逐步学习特征表示。然而，这些方法总是需要一定量的存储空间来缓存样本集。

另一类方法利用生成模型生成伪训练样本。Deep Generative Replay (DGR)[94] 训练一个生成对抗网络 (GAN)[95] 模型来帮助训练分类模型。论文[96]中也使用了 GAN 来生成辅助训练数据。另一项研究[97]提出了借助变分自编码器 (VAE)[98] 和网络蒸馏[99]改进 DGR。Generative Feature Replay[100]研究基类和新类之间的数据不平衡问题，提出特征生成来降低 generative replay 的复杂度。这些方法不需要空间来缓存历史样本集，但必须训练一些生成模型作为辅助，等待生成模型生成伪数据后再训练分类模型。

一般来说，大多数现有的类增量学习方法都需要使用真实的或伪造的历史训练样本巩固模型的记忆。然而，增量学习的动机是通过渐进式学习的方式来模拟人类学习的方式、减少训练开销[84]。换句话说，理想情况下的增量学习应该只用新增的训练数据来训练模型，而不是依赖缓存数据或生成模型的辅助。

4.2.2 受限环境下的增量学习

有一部分研究工作面对受限环境下的增量学习问题。Elastic Weight Consolidation (EWC)[86] 将基于 Fisher 信息矩阵测得的重要权重进行加固。突触智能 (Synaptic Intelligence, SI)[101] 随着时间的推移积累和利用任务相关信息以防止遗忘。Learning without Forgetting (LwF)[102] 基于网络蒸馏使模型的新输出接近旧输出。这些方法都引入了一些正则项，用来在增量训练过程中巩固网络记忆。

然而，上述模型主要用于任务增量学习，和类增量学习有较大的区别。任务增量学习中的不同任务被分离开来，并且可以在测试阶段通过描述符来识别

测试样本属于哪个任务。论文[97]明确区分了类增量学习和任务增量学习的区别。一些研究[103-104] 根据是否提供任务描述符将增量学习的测试模式分为单头测试和多头测试。这些研究中的实验结果还表明，任务增量学习方法通常在多头测试中运行良好，但在没有任务描述符的单头测试中往往会失效。类增量学习是一个典型的单头设置场景，在这种设置下这些方法的性能显着下降。

一些后来的研究工作汲取了论文[102]中的经验，基于网络蒸馏开发出了有效的类增量学习算法。在论文[89] 中，作者将 LwF 用于特征增量学习，并在隐藏层之上添加线性分类层，得到了 LwF.MC 算法。Learning Without Memorizing (LWM) 通过整合 Grad-CAM[105]生成的注意力图扩展了 LwF.MC。Deep Model Consolidation (DMC)[91]同时使用有标签的训练数据和无标签的辅助数据来进行知识蒸馏。

此外，可扩展的动态神经网络也是实现增量学习的常见方式。Dynamically Expandable Networks[106]在学习新任务的时候向网络中动态增加新的神经元，并且根据神经元的重要程度进行剪枝，清理无用的神经元。MEMO[107]提出了一种高效的网络扩展机制，将深度网络的上层结构分解为面向不同任务的特化区块，在扩展的时候仅增加新的特化区块。DyTox[108]实现了可扩展的 ViT[109]模型，通过 task token 实现网络扩展，比起扩展网络骨干结构具有更高的效率。

大多数类增量学习研究都假定了不同增量学习阶段的类别不会重叠，然而在真实应用中类别重叠是经常发生的情况[110]。带有类别重叠的增量学习算法在一些文献[111-112]中被称为” 模糊类增量学习”(Blurry Class Incremental Learning)。虽然这类问题更接近于真实应用，但本文为了分析类增量学习的关键难题，还是采用了更常见的学习范式作为研究对象。

4.3 问题分析

本节为受限环境下的类增量学习问题给出定义，并进一步讨论了这一问题下的两个主要挑战：目标抑制和开放空间风险。

4.3.1 问题定义

在训练过程中，有标签的数据集 $\mathcal{D}$ 被分成若干个子集，在多个训练阶段中连续输入给模型 $\mathcal{M}$。在第一个训练阶段，模型 $\mathcal{M}$ 被初始化，然后学习 $\mathcal{D}$ 的第一个子集 $\mathcal{D}_1$。第一个训练阶段结束后 $\mathcal{M}$ 的状态记为 $\mathcal{M}_1$。在第 k 个增量学习阶段，模型 $\mathcal{M}$ 从状态 $\mathcal{M}_{k-1}$ 开始，在数据子集 $\mathcal{D}_k$ 上进行训练，训练结束后进入状态 $\mathcal{M}_k$。数据子集 $\mathcal{D}_k = \{(\boldsymbol{x}_1^k, \boldsymbol{y}_1^k), (\boldsymbol{x}_2^k, \boldsymbol{y}_2^k), ..., (\boldsymbol{x}_{n_k}^k, \boldsymbol{y}_{n_k}^k)\}$，其中 $\boldsymbol{x}^k$ 是训练数据的输入向量，$\boldsymbol{y}^k$ 是 one-hot 标签向量。令 $\mathcal{C}_k$ 为 $\mathcal{D}_k$ 中的类别集合，模型 $\mathcal{M}_k$ 应该能够对集合 $\mathcal{C}_0 \cup \mathcal{C}_1 \cup ... \cup \mathcal{C}_k$ 中的所有类进行分类。参考此前类增量学习研究中的设置，假设不同训练阶段出现的类别不重叠，即 $\mathcal{C}_i \cap \mathcal{C}_j = \varnothing$ if $i \neq j$。

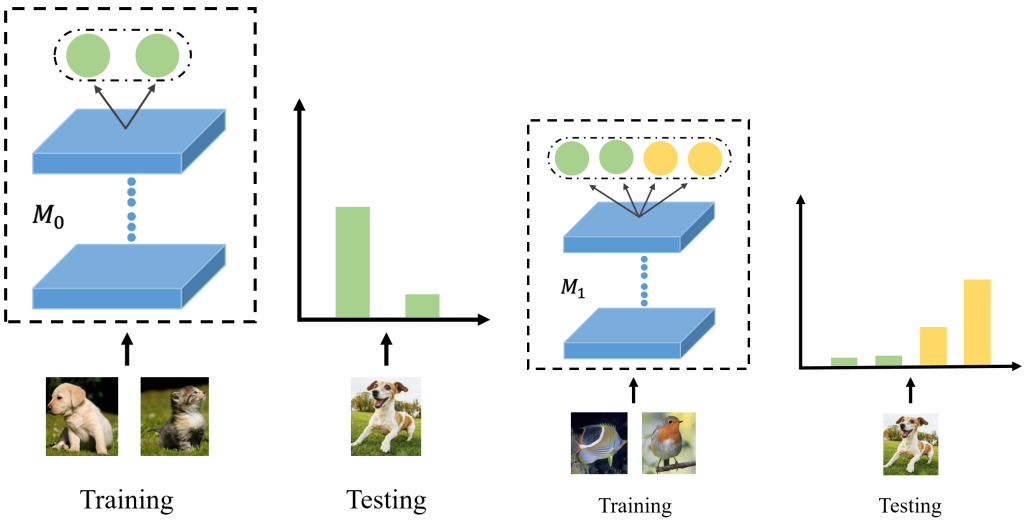
在基于复习的算法中，模型 $\mathcal{M}_k$ 学习 $\mathcal{D}_k$ 的所有数据的同时，还使用从 $\mathcal{D}_1$ $\mathcal{D}_{k-1}$ 中采样的基类训练样本巩固了记忆。将用于复习的基类样本集合记为 $\mathcal{D}_b'$，通常对其中的样本数量进行限制，以限制空间占用随着增量学习阶段的推进而不断增加。样本数量限制的结果则是出现了偏向于新类别 $\mathcal{C}_k$ 的不平衡。这种不平衡随着增量训练的推进而增加，因为 $\mathcal{D}_b'$ 中总样本数量不变，每个类别的样本数量变少。再受限制的类增量学习设定下，$\mathcal{D}_b'$ 的容量为 0，这使得类别失衡变得无限大，在这种情况下带来了两大挑战。

4.3.2 目标抑制

本小节用一个例子来讨论类增量学习中的目标抑制问题，如图 4.1所示。这里假设模型已经被训练来对一组基类 $\mathcal{C}_b$ 进行分类，并在新的训练状态下学习一组新类 $\mathcal{C}_n$. 大多数类增量学习分类器都在输出层使用 softmax 计算类别概率，用交叉熵损失进行训练。然而，当训练结束时，对于任何测试样本，网络预测的新类概率必定会超过基类概率。

本文称这种现象为*目标抑制* (target suppression)，而这一现象是类增量学习区别于任务增量学习的重要特征。给定一个输入样本 $(\boldsymbol{x}, \boldsymbol{y})$，令 $\boldsymbol{z} = h(\boldsymbol{x})$ 为从网络倒数第二层输出的特征向量，$\boldsymbol{W}_i$ 是输出层第 i 个线性节点的权重。使用上文中的定义，能够推导出以下定理：

定理 4.1 *如果 $\boldsymbol{z} \geq 0$，则通过梯度下降将所有类别的交叉熵损失最小化到当前任*



(a) 初始状态　　(b) 增量学习结束后的状态

图 4.1　目标抑制问题的一个例子。在学习了仅包含新类样本的增量训练数据后，模型预测输出中的新类概率将大幅抑制基类概率。

务：

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = -\sum_{i}^{|\mathcal{C}_b|+|\mathcal{C}_n|} y_i \log \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_{j}^{|\mathcal{C}_b|+|\mathcal{C}_n|} \exp(\ bm{W}_j \boldsymbol{z})} \tag{4.1}$$

会导致任何基类 i 对应的输出层神经元权重 $\boldsymbol{W}_i$ 单调递减。

证明 损失函数 $\mathcal{L}$ 关于权重向量 $\boldsymbol{W}_i$ 的偏导数是

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_i} &= -y_i(1 - \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_j \exp(\boldsymbol{W}_j \boldsymbol{z})})\boldsymbol{z} + \sum_{j \neq i} y_j \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_j \exp(\boldsymbol{W}_j \boldsymbol{z})} \boldsymbol{z} \\ &= \sum_j y_j \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_j \exp(\boldsymbol{W}_j \boldsymbol{z})} \boldsymbol{z} - y_i \boldsymbol{z} \\ &= \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_j \exp(\boldsymbol{W}_j \boldsymbol{z})} \boldsymbol{z} - y_i \boldsymbol{z}. \end{aligned} \tag{4.2}$$

在限制设置下，$y_i = 0$ 在增量学习阶段始终为真。已知 $\boldsymbol{z} \geq 0$，所以可以得到：

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_i} = \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_j \exp(\boldsymbol{W}_j \boldsymbol{z})} \boldsymbol{z} \geq 0. \tag{4.3}$$

学习率为 η 的梯度下降会导致 $\boldsymbol{W}_i := \boldsymbol{W}_i - \eta \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_i} \leq \boldsymbol{W}_i$，即 $\boldsymbol{W}_i$ 单调递减。 □

从上面的证明可以看出，目标抑制是由于交叉熵误差的反向传播以及输出层神经元与当前训练数据中的类别不对应导致的。当模型被限制访问基类的训

练样本时，模型只会学习不属于这一类的负例，导致了输出层神经元权重的降低。即使在增量学习阶段冻结 $\boldsymbol{W}_i$ 以防止其衰减，对应于新类 j 的神经元权重 $\boldsymbol{W}_j$ 也会大大超过 $\boldsymbol{W}_i$，这是由损失函数的梯度性质所决定的。

以往的研究工作[103-104]中已经分析过，当 EWC 和 LwF 方法使用 Equation(4.1) 作为损失函数时无法正常发挥作用。但一个可能的解决方案是修改目标函数，使其只考虑当前可用的类：

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = -\sum_{i}^{|\mathcal{C}_n|} y_i \log \frac{\exp(\boldsymbol{W}_i \boldsymbol{z})}{\sum_{j}^{|\mathcal{C}_n|} \exp(\boldsymbol{W}_j \boldsymbol{z})}. \tag{4.4}$$

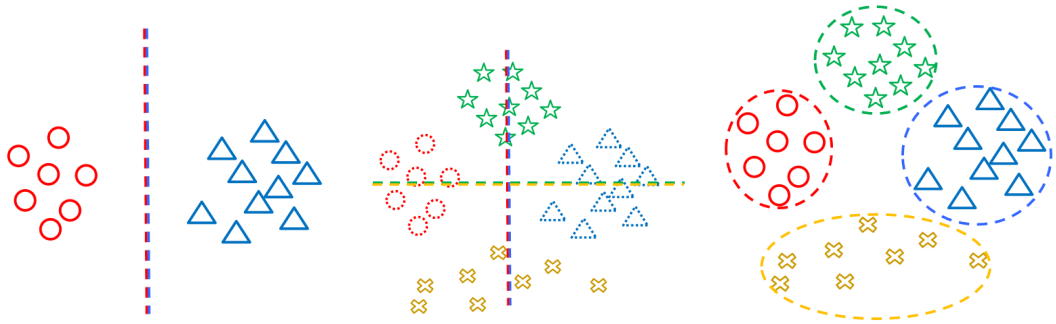
这种损失函数通常用于任务增量学习，其中模型的输出层通常具有多头结构，每个头对应一个任务，任务描述符在测试时可用。在这种情况下，EWC[86] 和 LwF[102]等方法可以在一定程度上减轻灾难性遗忘现象。但因为类增量学习中不像任务增量学习一样可以在测试阶段使用任务描述符判断用哪一个输出头进行预测，因此只能使用所有所有神经元的输出计算分类概率，这就导致了另一个问题，即下面将要介绍的开放空间风险。

4.3.3 开放空间风险

上文的分析得到的结论之一是在增量学习阶段忽视分类层中对应基类的神经元可以避免产生目标抑制现象。然而，这带来了一个新的问题：在基类神经元训练完成后，新类别被增量输入，基类神经元并未学习如何正确将新类别的样本识别为负例。在受限设置下，新类别对应的输出层神经元也不能访问基类的训练样本，无法学习到新类和基类的区别。这个问题被称为为开放空间风险 (open space risk)[2]，这意味着在训练阶段中看不到的新类可以出现在测试中。专注于降低开放空间风险的研究领域被称为开集识别[18]，其中分类器需要准确地对训练阶段学习过的类别进行分类，同时在测试阶段检测出属于未知类别的测试样本。

示例如图4.2所示。在图4.2(a) 中，一个模型被训练来识别初始状态下的两个类。在图 4.2(b) 中，模型以增量学习的方式学习两个新类。在这里，基分类器面临一个两难的境地：如果训练误差反向传播给它们，目标抑制会使它们的权重衰减；否则，它们就无法将新的例子识别为反面例子。另一方面，新分类器

是在没有基类示例的情况下训练的，因此它们也无法学习基类和新类之间的决策边界。因此，该模型没有学会同时区分这四个类别。如图 4.2(c) 所示，假设该模型是一个理想的开集识别模型，则可以给出一个解决方案，其中各类别对应的决策边界限制了类别的区域，为看不见的类保留空间。



(a) 初始状态　(b) 增量学习结束后的状态　(c) 理想的开集识别模型给出的解

图 4.2　类增量学习中的开放空间风险。在初始状态中，模型能够识别两个类。在增量学习结束后中，基分类器无法识别新类样本，而新分类器在没有基类示例的情况下进行训练，也无法识别基类样本。而理想的开集识别模型则能够给出更好的解决方案。

4.4　基于原型的网络加固算法

4.4.1　概述

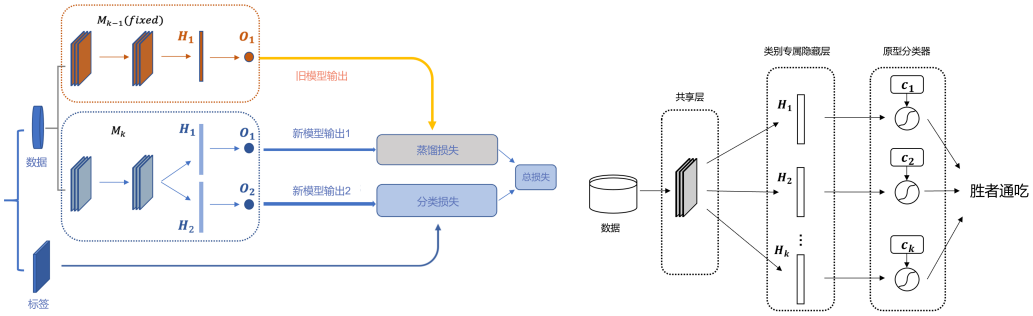
上一节分析了受限环境下类增量学习的两大障碍。针对上述问题，本节提出了一种基于原型的网络加固算法 (Prototype-based Network Consolidation, PBNC)。PBNC 的框架如图 4.3 所示，将一个多分类问题分解为多个二分类问题，并相应地采用多头网络结构，它由两个基本组件组成：

(1) 面向不同任务与数据类型而进行特征学习的深度网络，即共享层。(2) 由多个二元分类器组成的多头分类层。每个类别对应一个轻量化的网络结构，记为“分类头”。

每当训练数据中出现新类别 i 时，在分类层中添加一个对应的分类头。在 PBNC 的默认设置中，每个分类头由一个全连接层 $\boldsymbol{H}_i$ 和一个输出单元 $\boldsymbol{O}_i$ 组成。输出单元 $\boldsymbol{O}_i$ 的参数是类别 i 的原型向量 $\boldsymbol{c}_i$。样本 $\boldsymbol{x}$ 属于类别 i 的概率是根据它的特征向量与 $\boldsymbol{c}_i$ 的距离计算的。

在第 k 个增量学习阶段开始时，对应新类别的新分类头被添加到 $\mathcal{M}_{k-1}$ 中，

如图 4.3(b) 中的 H_2 所示。之后，训练数据通过被传入 $\mathcal{M}_{k-1}$，记录所有旧的分类头（即 H_1）的输出数字，将其作为知识蒸馏的学习目标。训练数据的标签向量和蒸馏目标分别是 $\mathcal{M}_k$ 中新头和旧头的期望输出，以使网络 $\mathcal{M}_k$ 的输出接近这些学习目标。在测试阶段，一个测试样本被传入网络的所有分类头中，选择输出概率最高的分类头作为该样本的预测标签。



(a) PBNC 的训练框架　　(b) PBNC 的测试框架

图 4.3　PBNC 的训练与测试框架示意图

4.4.2　基于原型的二元分类

上一节讨论了克服目标抑制现象的可行方案之一是阻止基类对应的分类层单元在后续的增量学习阶段参与误差反向传播。为此，本节提出一种方案以应对目标抑制，即将进行多分类的全连接层分解为由多个二元分类头组成的网络结构。在第 k 个增量学习阶段，记 $\mathcal{C}_b$ 为模型 $\mathcal{M}_{k-1}$ 学习过的基类集合，$\mathcal{C}_k$ 为新类集合。在训练阶段开始时，$|\mathcal{C}_k|$ 个新的二元分类头被添加到共享层之上。只有这些分类头学习新的训练样本，而基类对应的分类头则不参与计算分类损失。类 i 对应的分类器由全连接隐藏层 $\boldsymbol{H}_i$ 和 sigmoid 输出单元 O_i 组成。给定一个输入样本 $\boldsymbol{x}$，分类头输出此样本属于其对应类别的概率：

$$\hat{y}_i(\boldsymbol{x}) = p(y_i = 1 \mid \boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{1 + e^{-o_i(\boldsymbol{x})}}, \tag{4.5}$$

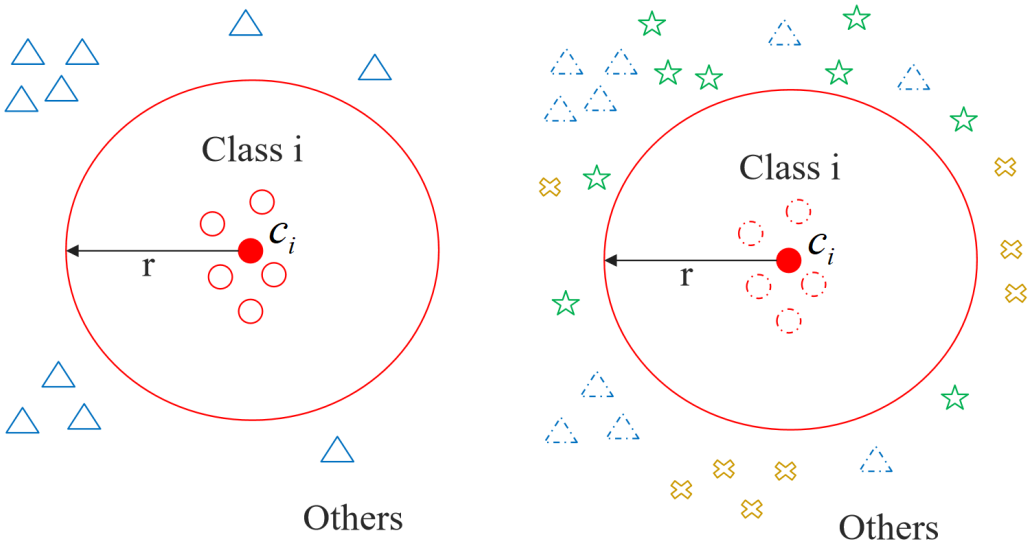
其中 $\boldsymbol{\theta}$ 表示网络中的参数。

这样，网络的输出就变成了一组伯努利分布，而不是单个 softmax 层输出的多项式分布。在测试阶段，可以通过选择所有二元分类器的最高输出分数来预测测试示例的标签。虽然所有类别的概率之和不等于 1，但依然可以选择概率

最高的类别作为样本的预测标签。优化一个分类头不会影响其他分类头的参数，因此防止了误差反向传播导致的目标抑制现象。

此外，本节提出一种通过基于原型的分类方法以降低开放空间风险，以此来降低多个分类头在融合时的互相干扰。在常见的分类神经网络中，输出层本质上是一个线性分类器，它建立在隐藏层学习到的特征之上。这种结构将数据的特征表示训练成线性可分的，但目标函数中通常没有减少类内方差或扩大类间边际的项。如图 4.2 所示，线性分类器导致无限的开放空间风险。

基于原型的分类单元能够在一定程度上降低开放空间风险。图 4.4 中显示了一个示例。



(a) 初始状态　　(b) 增量学习阶段

图 4.4　基于原型的分类单元示例。在初始状态 (a) 中，单元 O_i 学习超球形决策边界以识别类别 i。在学习阶段 (b) 中，开放空间风险相对线性分类单元更少。

对于每个类别 i，模型学习一个原型向量 $\boldsymbol{c}_i$ 作为其在特征空间中的代表点。给定输入样本 $\boldsymbol{x}$，设 $h_i(\boldsymbol{x})$ 是类 i 对应的分类头隐藏层 $\boldsymbol{H}_i$ 输出的特征表示，则输出单元 O_i 给出的概率是根据 $h_i(\boldsymbol{x})$ 和 $\boldsymbol{c}_i$ 之间的距离得到的：

$$o_i(\boldsymbol{x}) = r - ||\boldsymbol{c}_i - h_i(\boldsymbol{x})||_2, \tag{4.6}$$

其中 $||\cdot||_2$ 是向量的 2-范数。r 是预设的超参数，其意义为特征空间中超球形决

策边界的半径。然后类 i 对应的二元交叉熵损失计算如下：

$$\mathcal{L}_i(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = -y_i \log \hat{y}_i(\boldsymbol{x}) - (1 - y_i) \log (1 - \hat{y}_i(\boldsymbol{x})). \tag{4.7}$$

与 iCaRL 通过样本均值计算的原型不同，PBNC 中的原型是与特征层的网络参数联合优化的。最小化这个损失函数使得从 $\boldsymbol{H}_i$ 中提取的正例特征向量聚集在 $\boldsymbol{c}_i$ 周围，负例特征向量与 $\boldsymbol{c}_i$ 的平方距离超过 r，确保正例和负例之间的间距。

对于类增量学习而言，这个解决方案具备了充分的适用性和可扩展性。虽然网络使用多头结构，但它在测试阶段不需要任务描述符，因此可适应单头测试。每个二元分类头的结构也是可选的。在最简单的形式中，每个头只需要一个输出单元，此时全部分类头的总参数量等于 softmax 输出层中的参数量。在 PBNC 的默认结构中，每个二元分类头由一个使用 ReLU 激活函数的全连接隐藏层和一个输出单元组成。隐藏层不是实现类增量学习的必要网络结构，但它可以提高分类头的性能，降低多个整合分类头的难度。

4.4.3　网络加固

通过多个二分类器的网络结构，类增量学习问题被分解为多个二分类任务以应对输出层目标抑制现象，但共享层的参数依旧会受到灾难性遗忘的影响。对于从单个多分类任务分解而来的多个二分类任务，任务增量学习方法能够起到减少灾难性遗忘的作用。网络加固的目标是在训练数据中缺少基类样本时，使基类对应的分类头输出保持稳定。因此本节使用了基于知识蒸馏的网络加固方法。

网络的记忆体现在它的输入输出行为上。知识蒸馏迫使学生模型的行为接近另一个教师模型，从而将教师模型的记忆传授给学生模型。将模型 $\mathcal{M}_{k-1}$ 作为教师模型、$\mathcal{M}_k$ 作为学生模型，使用知识蒸馏中将 $\mathcal{M}_{k-1}$ 的记忆传授给 $\mathcal{M}_k$，这种训练方式巩固了模型关于基类的记忆。PBNC 使用了基于距离的知识蒸馏，鼓励输入给模型的训练样本的特征向量与基类原型之间的距离在增量学习阶段中保持不变。将 $\mathcal{M}_{k-1}$ 的网络参数记为 $\boldsymbol{\theta}^{k-1}$，通过最小化最小二乘蒸馏损失来

巩固网络输出：

$$\mathcal{L}_i^D(\boldsymbol{x}, \boldsymbol{\theta}^{k-1}, \boldsymbol{\theta}) = (o_i(\boldsymbol{x}|\boldsymbol{\theta}) - o_i(\boldsymbol{x}|\boldsymbol{\theta}^{k-1}))^2. \tag{4.8}$$

尽管蒸馏减轻了灾难性遗忘，但学习新知识和巩固旧知识之间的不平衡仍然存在。因此 PBNC 通过将二元交叉熵分类损失替换为论文[113]中提出的 Focal Loss 来进一步缓解不平衡，其定义为

$$\mathcal{L}_i^F(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = \begin{cases} -(1-\hat{y}_i(\boldsymbol{x}))^2 \log \hat{y}_i(\boldsymbol{x}) & y_i = 1 \\ -\alpha \hat{y}_i^2(\boldsymbol{x}) \log (1-\hat{y}_i(\boldsymbol{x})) & y_i = 0 \end{cases}. \tag{4.9}$$

Focal Loss 损失使网络专注于学习一小部分困难的训练样本。公式 (4.9) 是 Focal Loss 的 α 平衡形式 (α-balanced form)，并且将原公式中的聚焦参数固定为 2。

结合训练样本分类的 Focal Loss 和网络加固的蒸馏损失，得到 PBNC 的总损失函数：

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = \sum_{i \in \mathcal{C}_k} \mathcal{L}_i^F(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) + \lambda \sum_{j \in \mathcal{C}_b} \mathcal{L}_j^D(\boldsymbol{x}, \boldsymbol{\theta}^{k-1}, \boldsymbol{\theta}). \tag{4.10}$$

上式中蒸馏损失可以被视作正则项，系数 λ 控制分类损失和蒸馏损失的平衡。

4.5　实验

本节介绍的验证 PBNC 性能的实验设置并展示结果。首先介绍实验中使用的数据集和参与比较实验的基准方法，然后描述了评估方式和实现细节，包括网络架构和参数设置。

4.5.1　数据集、评估指标和基线

本节使用四个数据集进行实验：MNIST[114]、CIFAR-10、CIFAR-100[115] 和 ImageNet-200[116]。每个数据集中的类分为几个批次，每个批次包含的类不与其他批次重叠。整个增量学习过程同样分为几个阶段，模型在每个阶段学习一个批次的训练样本，并且无法访问其他批次的数据。在每个学习阶段之后，使用当前阶段为止学习的所有类的测试样本来验证模型性能，记录准确性。在整个数

据集都被学习过之后，使用除初始状态之外的所有结果记录计算平均增量精度 (average incremental accuracy)[89]。表 4.1展示了实验数据集的详细信息。

表 4.1 实验数据集的详细信息

数据集	MNIST	CIFAR-10	CIFAR-100	ImageNet-200
类别数量	10	10	100	200
每批次类别数量	2, 5	2, 5	10, 20	10, 20
每类训练样本数量	5000	5000	500	500
每类训练样本数量	1000	1000	100	50
测试标准	top-1 准确率	top-1 准确率	top-1 准确率	top-5 准确率

实验中使用三种基线方法作为比较：固定表示法 (fixed representation, FR)、LwF.MC[89] 和 LwM[90]。FR 使用与 PBNC 同样的任务分解机制，在第一批次的数据上训练初始模型 $\mathcal{M}_1$，并在以后的增量学习状态中冻结共享层的参数，仅训练新创建的二元分类头。因为在类增量学习算法使用辅助数据（存储示例、未标记数据等）或模型对效果提升幅度很大，面向受限环境的类增量学习方法数量稀少，因此本文实验中的对比方法选择范围较窄。

4.5.2 实现细节

为了比较的公平性，本节的实验固定了神经网络的基本架构和相关的超参数，使用 Lenet-5[114]作为 MNIST 上所有模型的基本架构，ResNet-18[41]作为在 CIFAR-10、CIFAR-100 和 ImageNet-200 上的模型基本架构。PBNC 去掉了模型的最后一个全连接隐藏层与输出层，替换为各个类对应的分类头，每个分类头的隐藏层都是一个具有 10 个节点、使用 ReLU 激活函数的全连接层。所有算法都使用 Adam 优化器[37]进行优化，随机梯度下降的样本批次大小为 128。在 MNIST / CIFAR-10 上，每个增量学习阶段的 epoch 数设置为 60，在 CIFAR-100/ImageNet-200 上的 epoch 数设置为 100。所有实验中，初始学习率被设为 5×10^{-4}，使用学习率调节机制在每个训练阶段的进度达到 25%/50%/75% 时将学习率减半。

PBNC 中的超参数通过网格搜索进行了优化。具体设置为：公式 (4.10) 中的 λ 在 ImageNet-200 上为 0.001，在其他数据集上为 0.01；公式 (4.9) 中的 α 在 ImageNet-200 上为 0.5，在其他数据集上为 1；公式 (4.6) 中的 r 固定为 5。

4.5.3 实验结果

表 4.2 类增量学习模型的平均增量精度

数据集		MNIST		CIFAR-10		CIFAR-100		ImageNet-200	
每批次类别数量		2	5	2	5	10	20	10	20
模型	FR	0.380	0.676	0.302	0.486	0.219	0.306	0.390	0.522
	LwF.MC	0.759	0.891	0.614	0.767	0.437	0.520	0.628	0.694
	LwM	0.748	0.885	0.613	0.764	0.436	0.518	0.647	0.697
	PBNC	0.864	0.922	0.682	0.815	0.487	0.552	0.662	0.704

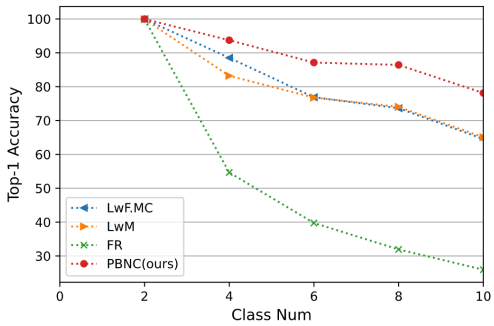
所有实验的详细结果如图 4.5 所示，平均增量精度在表 4.2 中报告。图表中的数值均为 5 次实验的平均结果。平均增量精度衡量了模型在增量学习过程中的整体性能。一般来说，模型的分类准确率会随着增量学习阶段的增加而降低，更好的算法能让模型的准确率下降更慢，使其平均增量精度更高。

在所有的实验中，FR 的准确率是最差的，主要是因为共享层仅在第一个阶段中进行学习，缺乏泛化能力。LwF.MC 和 LwM 的结果在大部分实验中接近。LwM 在 ImageNet-200 上的表现优于 LwF.MC，但在 MNIST 上的表现略差。这表明 LwM 提出的注意力蒸馏方法在复杂图像上效果更好，而在手写数字等非常简单的图像上并不总是起到正面效果。另一方面，用 grad-cam 计算注意力的方法仅能用于图像数据。

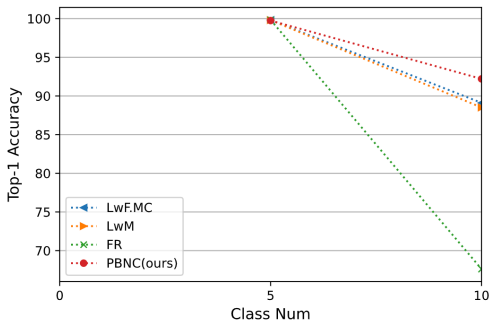
在像 MNIST 和 CIFAR-10 这样的小数据集上，PBNC 的性能比其他方法要好得多，尤其是当每批次的类数很少时。造成这种现象的原因是 PBNC 采用了基于原型的分类，降低了开放空间风险。分类器只能学习如何区分本批次的类之间的差异，因此当每批次的类数较少时，开放空间风险更为显着。当增量训练过程完成时，PBNC 在这些数据集上比 LwF.MC 和 LwM 提高了约 5%-13%。

在每批次具有更多类的更大数据集上，PBNC 的优势不那么明显，主要是因为模型的二元分类器可以学习更多的负样本，因此开放空间风险不那么显着。具体来说，PBNC 在早期训练状态下的表现并不优于基线，但其准确性比其他方法下降得更慢。

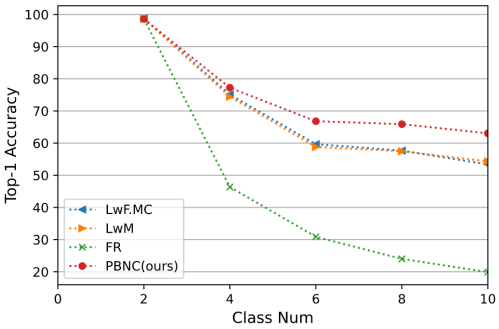
总之，PBNC 在所有实验中都优于基线，其优势在小批量上更为显着，因为每个训练阶段中的开放空间风险更为明显。



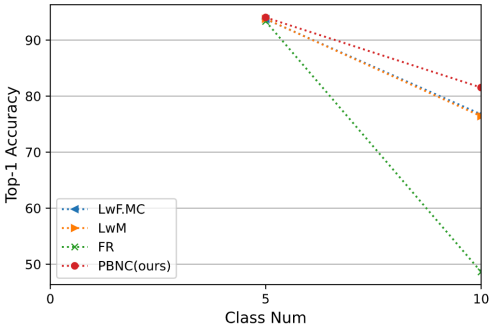
(a) MNIST（每批 2 个类）



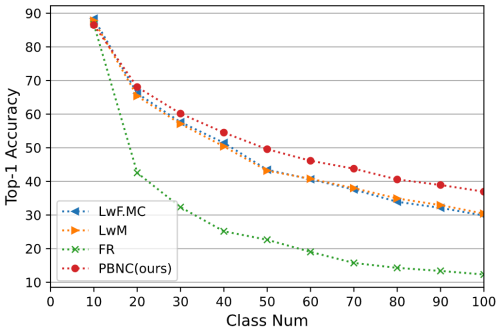
(b) MNIST（每批 5 个类）



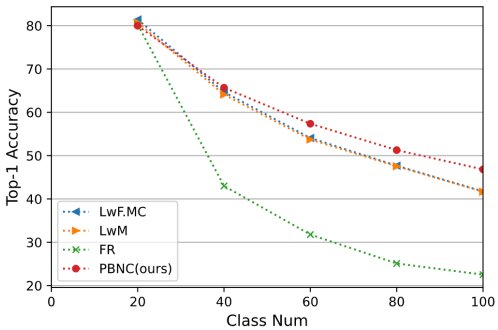
(c) CIFAR-10（每批 2 个类）



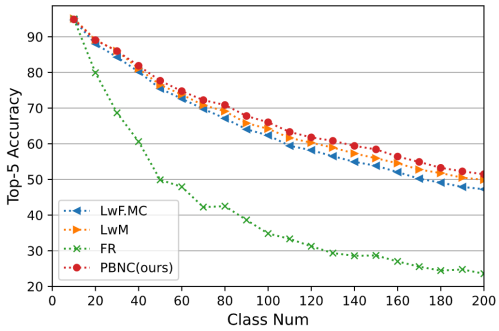
(d) CIFAR-10（每批 5 个类）



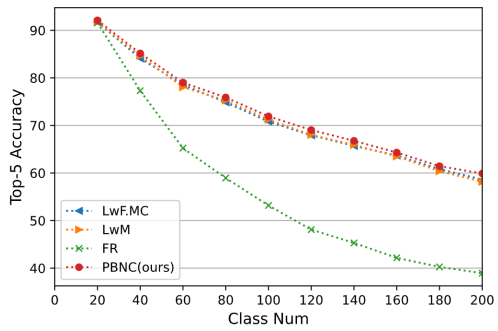
(e) CIFAR-100（每批 10 个类）



(f) CIFAR-100（每批 20 个类）



(g) ImageNet-200（每批 10 个类）



(h) ImageNet-200（每批 20 个类）

图 4.5　PBNC 和基线方法的性能比较。图中展示了每个训练阶段结束后的测试准确率。PBNC 在大多数增量学习阶段下优于基线方法。

4.5.4　消融实验

为了验证 PBNC 一些详细设计的有效性，本小节用一组消融实验来比较 PBNC 及其两种变体。

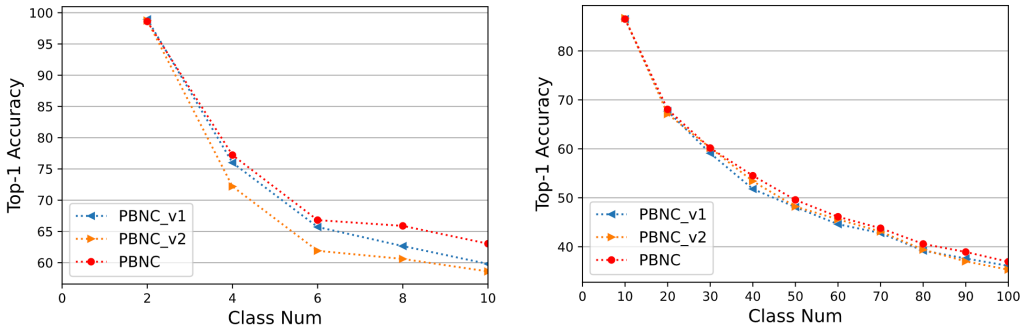
变体 1：公式 (4.9) 中的 Focal Loss 损失被替换为常用的二元交叉熵损失。

变体 2：公式 (4.8) 中基于距离的知识蒸馏损失被替换为基于二分类概率的蒸馏损失，定义为

$$\mathcal{L}_i^D(\boldsymbol{x}, \boldsymbol{\theta}', \boldsymbol{\theta}) = -\hat{y}'_i \log \hat{y}_i(\boldsymbol{x}) - (1 - \hat{y}'_i) \log (1 - \hat{y}_i(\boldsymbol{x})), \tag{4.11}$$

其中 $\hat{y}'_i = p(y_i = 1|\boldsymbol{x}; \boldsymbol{\theta}')$，$\boldsymbol{\theta}'$ 是旧参数。

这三种方法在 CIFAR-10（每批 2 个类）和 CIFAR-100（每批 10 个类）上进行了比较，结果如图 4.6 所示。



(a) CIFAR-10　　(b) CIFAR-100

图 4.6　PBNC 及其变体在 CIFAR-10 和 cifar100 上的性能比较。PBNC 的结果略好于其变体。

当整个训练过程完成时，PBNC 在 CIFAR-10 上比其变体提升约 3% - 4%，在 CIFAR-100 提升约 1% - 2%。此外，还可以看到对于蒸馏损失的优化更为重要。这些结果证实了 PBNC 中 Focal Loss 和网络加固方法的有效性。

4.6　本章小结

本章研究了受限环境下的类增量学习问题，在训练过程限制使用辅助数据和模型。首先分析了这一问题中的两个主要挑战：目标抑制和开放空间风险，从理论上证明了目标抑制是由单头分类层中的误差反向传播引起的，并且可以通过将多分类问题分解为多个元分类器来克服，并提出使用基于原型的分类来

降低开放空间风险。基于这些分析，本章设计了一种基于原型的网络加固算法 (PBNC) 受限环境下的类增量学习。PBNC 在多个基准数据集的实验中取得了优秀的效果。本章的研究成果形成了一项国家发明专利，并且获得了专利授权。

然而，PBNC 的实验效果依旧不如基于复习的类增量学习方法。PBNC 的改进方向与复习法没有冲突，因此可以允许网络在增量学习阶段访问从基类的训练数据中采样的部分训练样本，来进一步提高 PBNC 的性能。其次，开放空间风险仍然是亟待解决的重要问题，本文的下一章将详细介绍一种专门面向这一问题的算法。

第五章　基于对比学习的开集识别算法

在传统的识别任务中，模型只被训练来识别学习过的目标类别，但在实际应用中很难为所有潜在类别收集到训练样例。在测试阶段，当模型收到来自未知类别的测试样本时，它们会错误地将样本分类到已知类别中。开集识别（Open Set Recognition）是一类更现实的识别任务，它要求分类器能够检测出测试样本属于某个未知的类别，同时保持对已知类的高分类精度。本章从表征学习的角度研究了如何提高深度神经网络的开集识别性能。本章采用监督对比学习来提高特征表示的质量，提出了一种新的监督对比学习方法，使模型能够从软标签中学习，并在此基础上设计了一个开集识别框架。这种框架能够在对比训练深度神经网络时利用标签平滑和混合等训练技巧，从而提高在开集识别任务中检测未知样本的鲁棒性和传统分类任务的准确性。本章在多个基准数据集和测试场景上对所提出的方法进行实验，验证了所提出方法有效性。

5.1　引言

传统的识别算法在一个封闭集合的假设下工作，即训练数据和测试数据共享相同的标签和特征空间。然而，现实中通常很难收集到涵盖测试样本所有潜在类别的训练样例，而传统的分类器会将任何测试样本分类到其中一个训练类别中，即使它的真实类别还没有被学习到。开集识别是面向这个挑战而提出的一个新型识别问题，在测试过程中可能会出现来自未知类的样本，要求识别算法在检测未知测试样本的同时保持已知类的高分类准确率[2]。

在传统的多类分类网络中，输出层通常使用 softmax 函数来生成训练类的概率分布。softmax 函数由于其封闭性，不能估计测试样本属于未知类的概率，因此不适用于 OSR。一个直接的解决方案是对 softmax 函数输出的最大概率设置阈值[117] 以拒绝低置信度的估计，这为开集识别研究提供了一个简单的基线。然而，深度神经网络通常表现出对预测的过度自信，当样本属于未知类别时网

络的预测也具有很高的置信度。

尽管开集识别研究领域取得了很大进展，但最近的一项研究[118] 表明，简单地在闭集分类器上使用最先进的训练机制可以显着提高它们在开集识别任务中的性能。这一发现表明有可能使用更好的表示学习技术提高深度神经网络的特征表达能力，以此来增强网络在开集识别任务上的表现。受这项研究工作的启发，本章针对开集识别任务研究了一种更有效的表示学习机制。

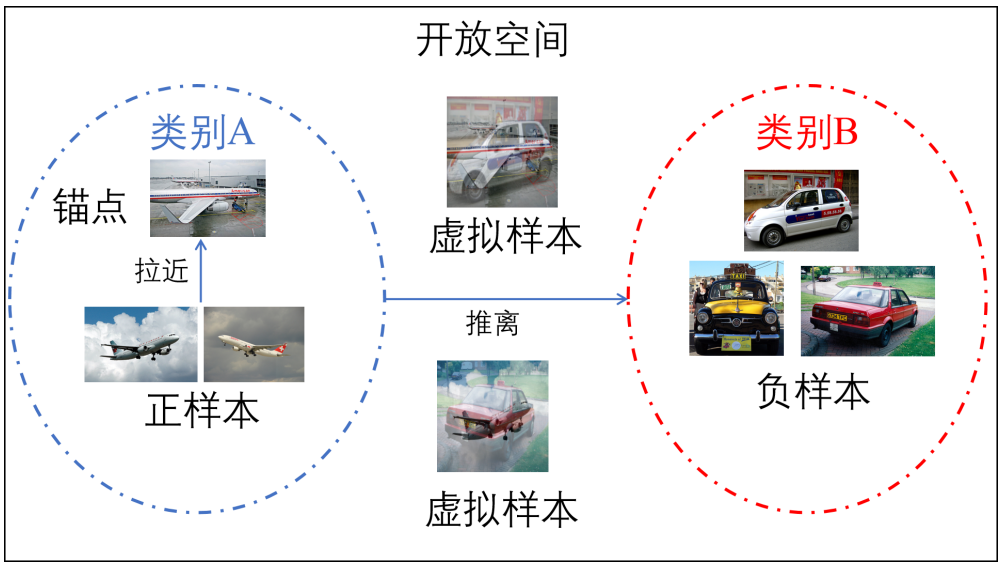


图 5.1　本章提出的方法的总览。在特征空间中由属于类别 A 的锚点 (anchor)，有监督的对比学习将同样属于 A 类中的正样本向锚点拉近，同时将属于 B 类的负样本推离锚点。Mixup 算法生成了虚拟样本，用来模拟开放空间中的未知样本。

本章的研究使用有监督的对比学习 (Supervised Contrastive Learning, SupCon)[119]来对属于同一类的正样本对和属于不同类的负样本对进行比较，训练高质量的数据表征网络。在实验中观察到，对比学习得到的表征模型更适合用来检测未知目标。本章使用 mixup 算法生成语义模糊的虚拟样本，以便模型可以在训练阶段将来自已知类的真实训练样本与虚拟样本进行对比。Mixup 算法的原理是将一对样本数据与标签同时进行随机的线性组合，得到一个虚拟的新样本。虚拟样本的标签是实向量，而非分类问题中常用的 one-hot 编码，这种标签也被称为“软标签”。为了将带有软标签的虚拟样本引入对比学习框架，本章设计了一种增强的监督对比学习方法，该方法允许训练样本对之间建立一种基于相似性的对比关系，而非简单的“非正即负”二元关系。这种改进提高了监督对比学习算法在开集识别和传统闭集分类任务上的表现。

本章的研究贡献总结如下：

1. 提出了一种基于对比学习的开集识别方法，简称对比开集识别 (Contrastive Open Set Recognition, ConOSR)。本章通过实验分析了为什么对比学习得到的特征可以提高分类器在开集识别任务中的性能。

2. 改进了监督对比学习算法，使其具有从软目标学习的能力 (Supervised Contrastive Learning with Soft Targets, SupCon-ST)。本章设计的改进方法在封闭集分类中优于标准的 SupCon 算法，并且还提高了 ConOSR 的性能，使其在基准数据集上的表现达到了目前最先进的水准。

5.2 相关工作

5.2.1 开集识别

开集识别首先在论文[2]中被提出，一些重要的相关术语也在该论文中被定义,例如开放空间和开放空间风险。最近的一篇综述论文[18] 将开集识别方法归纳为判别式方法和生成式方法两类。大多数最新的判别式方法都是基于深度神经网络的方法, 通过使用各种异常值检测机制增强输出层, 使深度网络具有未知检测能力。Openmax[120] 通过测量测试样本的激活向量与已知类的平均激活向量之间的距离, 来估计测试样本属于未知类的概率。Reciprocal Points Learning[121] 引入了一个名为 reciprocal point 的新概念，以便为特征空间中的每个已知类建模潜在的开放空间。PROSER[122] 将占位符分配给特征空间中的未知类，以此来模拟开放空间并预测未知数据的分布。PROSER 还使用特征混合来生成虚拟样本作为占位符。CVAECapOSR[123] 使用胶囊网络作为特征编码模型，目的是学习已知类的紧凑特征表示。

生成式方法可以进一步分为实例生成方法和非实例生成方法。实例生成通常使用生成对抗网络 (GAN)[124] 生成伪样本来模拟未知测试样本。在 Openmax 算法的基础上, G-Openmax[125]结合生成模型对其进行了改进, 它通过使用条件 GAN 生成的未知样本训练深度网络。OSRCI[126] 训练编码器-解码器 GANs, 生成类似训练样本、但不属于任何已知类的反事实示例, 并使用反事实示例增强训练数据。最近，OpenGAN[127] 提出使用 GAN 的鉴别器网络作为开放集似然函数, 使用真实世界数据作为未知类别的训练样本来改进 GAN 的训练。OpenGAN 在图像分类和像素分割任务中的效果明显优于现有的 OSR 方法。非实例生成方

法训练编码器-解码器网络以辅助未知样本检测。CROSR[128] 在检测样本的步骤中同时使用分类层的预测输出和样本的特征表示进行重建样本，根据样本的重建质量来判断样本是否属于未知类别。GFROSR[129] 使用重建模型作为一种数据增强方式，迫使编码网络学习能够捕获对象关键结构的特征。生成式方法通过建模数据的分布为识别系统提供了更多信息，但训练生成模型显着增加了识别系统的总训练成本。

5.2.2 对比学习

对比学习是表征学习的一个子领域，近年来引起了广泛的研究关注。大多数对比学习方法都是自监督的[130-133]，它们不依赖于特定任务提供的监督信息。自监督对比学习的一个主要问题是如何在没有监督信息的情况下获得正负样本对。MOCO[131] 和 SimCLR[132] 使用单个训练样本的多个视图作为正样本对，将不同的训练样本作为负样本对，但它们需要大量的负样本对才能获得良好的性能，这使得他们的训练开销非常大。BYOL[134] 和 SimSiam[133] 使用孪生网络结构和停止梯度来避免使用负样本对，因此它们可以将训练数据划分为较小的批次。在 ImageNet 分类任务中，最近的自监督对比学习方法取得了接近监督学习的结果。

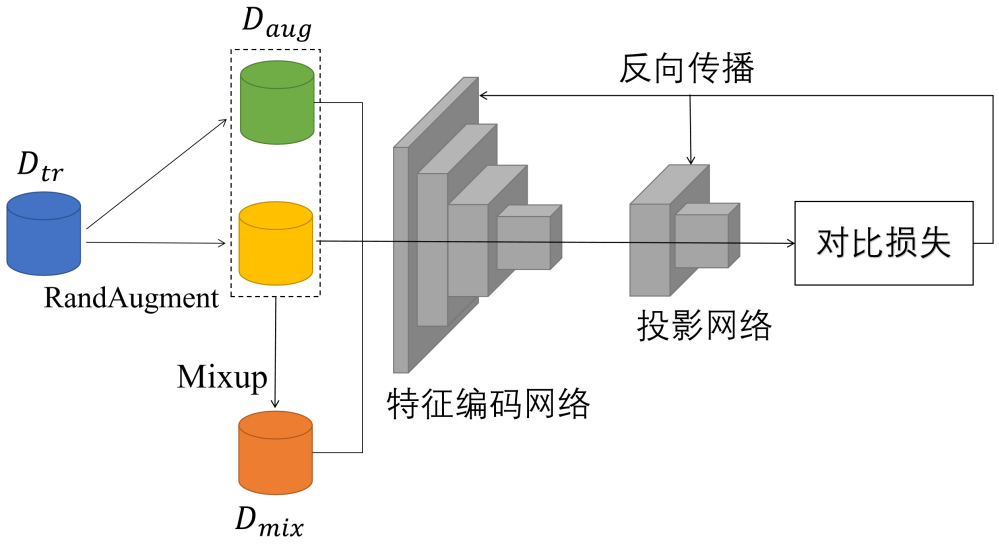
监督对比学习 (SupCon)[119] 则使用数据集上的标签来划分正负样本对，最大化正样本对之间的相似性和负样本对之间的差异。SupCon 在分类精度方面大大优于自监督方法，并在多个封闭集分类任务中取得了超过普通监督学习方式的结果。然而 SupCon 并没有像自监督方法那样被研究者们关注，因为它不能学习无标签的训练数据。

关于对比学习为何能起到提升表征质量作用的原因，近年来的一些研究工作对此进行了探讨。论文[135]指出了对比学习获得的表征具有 Alignment 和 Uniformity 两种关键属性。论文[136]分析了数据增强在自监督对比学习中发挥的重要作用，即来自同一类别中不同样本的增强视图可能高度相似。对比学习使统一样本不同视图的特征相似，进而让同一类别中的不同样本特征相似。论文[137]提出了一种观点，认为基于 InfoNCE 损失函数[130]的对比学习方法具有分离数据生成因子的作用，类似于非线性独立成分分析，即对比学习的过程是数据生成的逆过程。

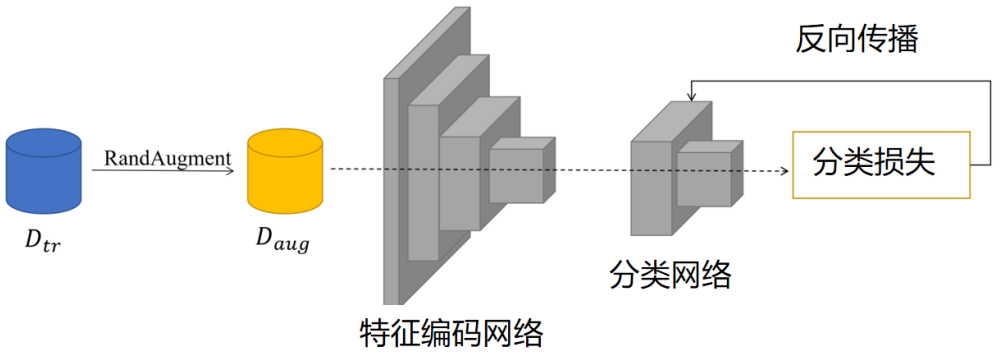
本章的研究工作中也使用了对比学习技术来提升网络表征的质量，这种提升同时体现在了已知类分类准确率和未知类检测精度两方面。然而，过去的对比学习算法因为严格定义了正负样本，而没有考虑样本标签的概率分布，因此和本章研究思路中的 Mixup 算法无法共同使用。因此本章对监督对比学习算法进行了改进，提出了适用于软标签学习的新型对比学习算法。

5.3　对比开集识别

本节将详细描述所提出的对比开集识别（ConOSR）算法。图 5.2展示了 ConOSR 的训练过程。



(a) 第 1 步：对比表征学习



(b) 第 2 步：训练分类器

图 5.2　ConOSR 的训练过程总览

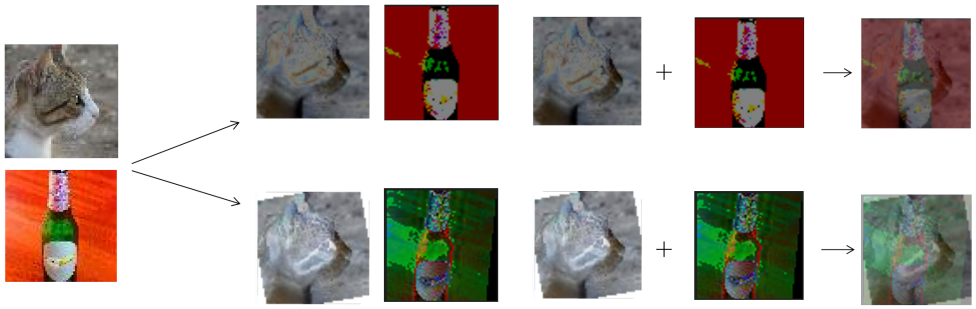
ConOSR 包括了一个对比学习步骤和一个分类器训练步骤。在对比学习阶段，网络模型有特征编码器网络和投影网络两个模块。数据预处理模块使用 Ran-

dAugment[138]算法生成训练数据 $\mathcal{D}_{tr}$ 的两个增强视图，然后对它们执行 Mixup 算法[139]，得到一批虚拟训练样本。之后，在增强数据 $\mathcal{D}_{aug}$ 和混合数据 $\mathcal{D}_{mix}$ 上计算的对比损失，并以最小化对比损失为目标对编码器网络和投影网络进行反向传播优化。

在分类器训练阶段，投影层网络被丢弃，编码器网络的参数被固定。$\mathcal{D}_{tr}$ 经过 RandAugment 算法预处理，然后通过编码器网络前向传播得到特征表示，最后通过分类网络得到预测输出。这一阶段的损失函数是交叉熵损失，用来反向传播优化分类网络。网络参数收敛后，使用 $\mathcal{D}_{tr}$ 估计各类别的拒绝阈值，用来检测不属于已知类的测试样本。本节的各小节将详细介绍上述框架中的各个组件。

5.3.1　数据增强和对比学习

如图 5.2 所示，ConOSR 采用了两种不同的数据增强技术。RandAugment[138] 和 Mixup[139] 是近年新提出的数据增强方法，广泛应用于各个领域。图 5.3 显示了两种增强方法的示例。



(a) RandAugment　　(b) Mixup

图 5.3　ConOSR 使用的数据增强技术。(a) RandAugment **对输入图像的视觉元素进行随机变换，如旋转、平移、变色等，同时保持其语义内容；**(b) Mixup **对两个样本的内容和标签进行线性组合得到虚拟样本。**

给定一张训练图像，RandAugment 从 14 个可用变换中随机选择 N 个变换，然后按顺序将所选变换应用于图像。变换的幅度由超参数 M 控制。RandAugment 丰富了训练样本的视觉信息，同时保持其语义内容不变，使模型可以学习到具有不变性的特征表示。

对于一个训练批次 $(\boldsymbol{x}_i, y_i)_{i=1}^{n}$ 中的每个训练样本 $(\boldsymbol{x}_k, \boldsymbol{y}_k)$，首先使用 RandAugment 生成两个增强视图 $\widetilde{\boldsymbol{x}}_{2k}$ 和 $\widetilde{\boldsymbol{x}}_{2k+1}$。增强函数的随机性确保了 $\widetilde{\boldsymbol{x}}_{2k}$ 和 $\widetilde{\boldsymbol{x}}_{2k+1}$ 在训练阶段的视觉差异。

在使用 RandAugment 增强图像的同时，ConOSR 使用标签平滑 (label smoothing) 对训练样本的标签进行增强。如果数据集中的类别总数为 m 且训练示例 $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ 属于第 k 类，则生成平滑标签 $\tilde{\boldsymbol{y}}_i = (\tilde{y}_{i1}, \tilde{y}_{i2}, \ldots, \tilde{y}_{im})$ 的公式为：

$$\tilde{y}_{ij} = \begin{cases} \sigma & j = k \\ \dfrac{1-\sigma}{m-1} & otherwise \end{cases} \tag{5.1}$$

.

Mixup 通过线性组合成对的训练样本来构建虚拟样本。给定从 $\mathcal{D}_{aug}$ 中随机抽取的两个训练样本 $(\widetilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)$ 和 $(\widetilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{y}}_j)$，虚拟示例 $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ 构造为：

$$\hat{\boldsymbol{x}} = \gamma \widetilde{\boldsymbol{x}}_i + (1-\gamma)\widetilde{\boldsymbol{x}}_j, \tag{5.2}$$

$$\hat{\boldsymbol{y}} = \gamma \tilde{\boldsymbol{y}}_i + (1-\gamma)\tilde{\boldsymbol{y}}_j, \tag{5.3}$$

其中 γ 是从 $[0,1]$ 区间上的均匀分布中随机选择的。

Mixup 增强在 ConOSR 框架中很重要，因为它生成的虚拟样本标签 $\hat{\boldsymbol{y}}$ 是 $\tilde{\boldsymbol{y}}_i$ 与 $\tilde{\boldsymbol{y}}_j$ 的线性组合。在理想情况下，标签为 $\tilde{\boldsymbol{y}}_i$ 或 $\tilde{\boldsymbol{y}}_j$ 的样本在特征空间中形成两个紧密的类簇，而虚拟样本则分布在类簇之间的区域。在测试阶段，落在原本属于虚拟样本的区域的测试样本会被判定为未知样本。通过这种方式，虚拟样本可以在训练阶段起到模拟未知样本的作用。

5.3.2 使用软目标的监督对比学习

对比学习中的深度网络结构由特征编码器 $\boldsymbol{\phi}(\cdot)$ 和投影网络 $\boldsymbol{\psi}(\cdot)$ 组成。编码器网络将 $\boldsymbol{x}_i$ 映射到 $\mathbb{R}^{de}$ 中的特征向量 $\boldsymbol{h}_i$ 然后，投影网络进一步将 $\boldsymbol{h}_i$ 映射到 $\mathbb{R}^{dp}$ 中的向量 $\boldsymbol{z}_i$ 上。$\boldsymbol{z}_i$ 被用于计算对比损失。对比学习的目标是在投影空间中最大化正样本对和负样本对之间的相似性差异。在论文[119]提出的监督对比学习算法 SupCon 中，对比损失的计算公式为:

$$\mathcal{L}^{sup} = \sum_i -\frac{1}{|P_i|}\sum_{j\in P_i} \log \frac{\exp(\boldsymbol{z_i}\cdot \boldsymbol{z_j}/\tau)}{\sum\limits_{k\neq i}\exp(\boldsymbol{z_i}\cdot \boldsymbol{z_k}/\tau)}, \tag{5.4}$$

其中 P_i 是属于与 i 属于同一类的样本组成的集合，τ 是温度超参数。然而，对训练集进行数据增强之后，样本中的标签是实数向量，无法简单地划分为同类或异类。因此需要对 SupCon 进行了改进，使它在计算对比损失时使用样本之间的相似性关系，而不是简单地将它们分为正负对。

给定一对标记样本 $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ 和 $(\boldsymbol{x}_j, \boldsymbol{y}_j)$，使用一个标签相似度函数来计算它们的语义相似程度，记为 $s(\boldsymbol{y}_i, \boldsymbol{y}_j)$。为了使改进算法在使用 one-hot 标签时等同于 SupCon，将 $s(\boldsymbol{y}_i, \boldsymbol{y}_j)$ 代入到公式 (5.4) 中不能改变其原本计算结果。因此，当 $\boldsymbol{y}_i$ 和 $\boldsymbol{y}_j$ 是 one-hot 编码的标签向量时，如果 $\boldsymbol{y}_i = \boldsymbol{y}_j$ 则 $s(\boldsymbol{y}_i, \boldsymbol{y}_j) = 1$，否则 $s(\boldsymbol{y}_i, \boldsymbol{y}_j) = 0$。考虑到这种情况，本章在没有特殊说明的情况下使用余弦相似度作为函数 $s(\boldsymbol{y}_i, \boldsymbol{y}_j)$：

$$s(\boldsymbol{y}_i, \boldsymbol{y}_j) = \frac{\boldsymbol{y}_i \cdot \boldsymbol{y}_j}{||\boldsymbol{y}_i|| ||\boldsymbol{y}_j||}, \tag{5.5}$$

接着定义 SupCon-ST 损失函数如下:

$$\mathcal{L}^{scst} = -\sum_i \sum_{j \neq i} \frac{s(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\sum\limits_{k \neq i} s(\boldsymbol{y}_i, \boldsymbol{y}_k)} \log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_j / \tau)}{\sum\limits_{k \neq i} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_k / \tau)}. \tag{5.6}$$

公式 (5.6) 的形式类似于交叉熵损失。当所有标签向量都是 one-hot 向量时，公式 (5.6) 等价于公式 (5.4)。与 SupCon 损失相比，SupCon-ST 损失的主要优点是它允许标签是任意实向量，因此可以在对比学习框架中使用标签平滑和 Mixup。除此之外，SupCon-ST 还使监督对比学习可以使用知识蒸馏等其他带有软目标的训练方案。

5.3.3　分类器训练和未知样本检测

ConOSR 的第二个训练阶段是在特征编码器 $\phi(\cdot)$ 之上训练一个轻量级分类网络 $f(\cdot)$。在这个阶段，ConOSR 仍然使用 RandAugment 和 label smoothing 来预处理训练数据，但不再使用 Mixup。

给定训练示例 $(\boldsymbol{x}, \boldsymbol{y})$，$\boldsymbol{x}$ 属于类别 i 的概率由 softmax 函数估计：

$$y_i' = \mathbb{P}(y_i = 1 | \boldsymbol{x}) = \frac{e^{f_i(\boldsymbol{\phi}(\boldsymbol{x}))}}{\sum_{j=1}^{k} e^{f_j(\boldsymbol{\phi}(\boldsymbol{x}))}}, \tag{5.7}$$

然后计算交叉熵损失：

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = -\sum_{i} y_i \log y_i'. \tag{5.8}$$

$f(\cdot)$ 的参数通过反向传播最小化交叉熵损失来优化，而 $\boldsymbol{\phi}(\cdot)$ 的参数是固定的。

在训练阶段结束时，模型根据训练样本的输出值分布来估计用于检测未知样本的拒绝阈值。对于类别 i 中的每个训练示例 $(\boldsymbol{x}, \boldsymbol{y})$，如果 $i = \arg\max_j f_j(\boldsymbol{\phi}(\boldsymbol{x}))$，即 $(\boldsymbol{x}, \boldsymbol{y})$ 被正确分类，则将分类器的输出值 $f_i(\boldsymbol{\phi}(\boldsymbol{x}))$ 添加到集合 $\boldsymbol{T}_i$ 中。在处理完所有训练示例后，每个集合 $\boldsymbol{T}_i$ 的 λ 百分位数被记录为类别 i 的拒绝阈值 ϵ_i。此处 λ 是预设的超参数。

在测试阶段，如果 $\max_i f_i(\boldsymbol{\phi}(\boldsymbol{x})) < \epsilon_i$，则测试样本 $\boldsymbol{x}$ 被标记为未知样本。拒绝阈值可以通过调整超参数 λ 来调整。λ 的默认值设置为 5，这表示允许训练集中 5% 的样本被误判为未知样本。然而，在测试数据上的误判率通常高于 λ。

5.4　算法分析

本节将对 SupCon-ST 方法进行理论分析，为了公式书写简洁，在此给出如下定义：

$$\boldsymbol{S}_{ij} = \frac{s(\boldsymbol{y}_i, \boldsymbol{y}_j)}{\sum\limits_{k \neq i} s(\boldsymbol{y}_i, \boldsymbol{y}_k)}, \boldsymbol{P}_{ij} = \frac{\exp(\boldsymbol{z_i} \cdot \boldsymbol{z_j}/\tau)}{\sum\limits_{k \neq i} \exp(\boldsymbol{z_i} \cdot \boldsymbol{z_k}/\tau)}, \tag{5.9}$$

因此 SupCon-ST 损失函数可被简化为:

$$\mathcal{L}^{scst} = \sum_{i} \mathcal{L}_i = \sum_{i} \sum_{j \neq i} \mathcal{L}_{ij} = -\sum_{i} \sum_{j \neq i} \boldsymbol{S}_{ij} log \boldsymbol{P}_{ij}. \tag{5.10}$$

5.4.1　SupCon-ST 损失的梯度推导

本小节通过分析其梯度推导来研究 SupCon-ST 的特性，首先从关于特定样本 $(\boldsymbol{x_i}, \boldsymbol{y_i})$ 的损失开始。

样本 $(\boldsymbol{x_i}, \boldsymbol{y_i})$ 在损失函数中可能扮演三个不同的角色。当 $(\boldsymbol{x_i}, \boldsymbol{y_i})$ 是锚点时，它与另一个样本 $(\boldsymbol{x_j}, \boldsymbol{y_j})$ 的对比损失 $\mathcal{L}_{ij}$ 对投影向量 $\boldsymbol{z}_i$ 的偏导数是：

$$\begin{aligned}\frac{\partial \mathcal{L}_{ij}}{\partial \boldsymbol{z_i}} &= -\frac{\partial}{\partial \boldsymbol{z_i}} S_{ij}\{\boldsymbol{z_i}\cdot\boldsymbol{z_j}/\tau - log\sum_{k\neq i}\exp(\boldsymbol{z_i}\cdot\boldsymbol{z_k}/\tau)\} \\ &= -\frac{S_{ij}}{\tau}\{\boldsymbol{z_j} - \frac{\sum\limits_{k\neq i}\boldsymbol{z_k}\exp(\boldsymbol{z_i}\cdot\boldsymbol{z_k}/\tau)}{\sum\limits_{k\neq i}\exp(\boldsymbol{z_i}\cdot\boldsymbol{z_k}/\tau)}\} \\ &= -\frac{S_{ij}}{\tau}(\boldsymbol{z_j} - \sum_{k\neq i} P_{ik}\boldsymbol{z_k}).\end{aligned} \tag{5.11}$$

当计算 $(\boldsymbol{x_i}, \boldsymbol{y_i})$ 与锚点 $(\boldsymbol{x_j}, \boldsymbol{y_j})$ 的对比损失时，$\mathcal{L}_{ji}$ 对 $\boldsymbol{z}_i$ 的偏导数为：

$$\begin{aligned}\frac{\partial \mathcal{L}_{ji}}{\partial \boldsymbol{z_i}} &= -\frac{\partial}{\partial \boldsymbol{z_i}} S_{ji}\{\boldsymbol{z_j}\cdot\boldsymbol{z_i}/\tau - log\sum_{k\neq j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_k}/\tau)\} \\ &= -\frac{S_{ji}}{\tau}\{\boldsymbol{z_j} - \frac{\boldsymbol{z_j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_i}/\tau)}{\sum\limits_{k\neq j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_k}/\tau)}\} \\ &= -\frac{S_{ji}}{\tau}(1 - P_{ji})\boldsymbol{z_j}.\end{aligned} \tag{5.12}$$

当计算锚点 $(\boldsymbol{x_j}, \boldsymbol{y_j})$ 和另一个样本 $(\boldsymbol{x_n}, \boldsymbol{y_n})$ 对比损失 $\mathcal{L}_{jn}$ 时，$\mathcal{L}_{jn}$ 对 $\boldsymbol{z}_i$ 的偏导数为：

$$\begin{aligned}\frac{\partial \mathcal{L}_{jn}}{\partial \boldsymbol{z_i}} &= \frac{\partial}{\partial \boldsymbol{z_i}} S_{jn}\log\sum_{k\neq j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_k}/\tau) \\ &= -\frac{S_{jn}}{\tau}\{\frac{\boldsymbol{z_j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_i}/\tau)}{\sum\limits_{k\neq j}\exp(\boldsymbol{z_j}\cdot\boldsymbol{z_k}/\tau)}\} = \frac{S_{jn}P_{ji}}{\tau}\boldsymbol{z_j}.\end{aligned} \tag{5.13}$$

然后可以推导出样本对比损失 $\mathcal{L}_i$ 和 $\mathcal{L}_j$ 对 $\boldsymbol{z}_i$ 的偏导数：

$$\begin{aligned}\frac{\partial \mathcal{L}_i}{\partial \boldsymbol{z_i}} &= \sum_{j\neq i}\frac{\partial \mathcal{L}_{ij}}{\partial \boldsymbol{z_i}} = -\frac{1}{\tau}(\sum_{j\neq i} S_{ij}\boldsymbol{z_j} - \sum_{j\neq i} S_{ij}\sum_{k\neq i} P_{ik}\boldsymbol{z_k}) \\ &= -\frac{1}{\tau}(\sum_{j\neq i} S_{ij}\boldsymbol{z_j} - \sum_{k\neq i} P_{ik}\boldsymbol{z_k}) \\ &= \frac{1}{\tau}\sum_{j\neq i}(P_{ij} - S_{ij})\boldsymbol{z_j}.\end{aligned} \tag{5.14}$$

$$\begin{aligned}\frac{\partial \mathcal{L}_j}{\partial \boldsymbol{z_i}} &= \frac{\partial \mathcal{L}_{ji}}{\partial \boldsymbol{z_i}} + \sum_{n\notin\{i,j\}} \frac{\partial \mathcal{L}_{jn}}{\partial \boldsymbol{z_i}} \\ &= \frac{1}{\tau}(S_{ji}P_{ji}\boldsymbol{z_j} - S_{ij}\boldsymbol{z_j} + \sum_{n\notin\{i,j\}} S_{jn}P_{ji}\boldsymbol{z_j}) \\ &= \frac{1}{\tau}(\sum_{n\neq j} S_{jn}P_{ji}\boldsymbol{z_j} - S_{ij}\boldsymbol{z_j}) = \frac{1}{\tau}(P_{ji} - S_{ji})\boldsymbol{z_j}.\end{aligned} \tag{5.15}$$

最后得到 $\mathcal{L}^{scst}$ 对 $\boldsymbol{z}_i$ 的偏导数：

$$\begin{aligned}\frac{\partial \mathcal{L}^{scst}}{\partial \boldsymbol{z_i}} &= \frac{\partial \mathcal{L}_i}{\partial \boldsymbol{z_i}} + \sum_{j\neq i} \frac{\partial \mathcal{L}_j}{\partial \boldsymbol{z_i}} \\ &= \frac{1}{\tau}\sum_{j\neq i}(P_{ij} + P_{ji} - S_{ij} - S_{ji})\boldsymbol{z_j}.\end{aligned} \tag{5.16}$$

梯度的最终形式简单易懂。给定锚点特征向量 $\boldsymbol{z}_i$，P_{ij} 可以被看作是对特征 $\boldsymbol{z}_j$ 是锚点同类的概率进行的估计，而 S_{ij} 是从样本标签计算出的 $\boldsymbol{z}_i$ 和 $\boldsymbol{z}_j$ 属于同类的概率。最小化 $\mathcal{L}^{scst}$ 能够对计算特征向量的编码器网络和投影网络进行参数优化，使 P_{ij} 与 S_{ij} 对齐。

5.4.2　对比学习的特性分析

正如在论文[119] 中讨论的那样，自监督对比学习的 InfoNCE 损失[132]是 SupCon 的一个特例，而 SupCon 则可被看做 SupCon-ST 在标签向量为 one-hot 编码时的特例。这些损失函数与 SupCon-ST 之间的差异是由语义相似度函数 $s(\cdot,\cdot)$ 的不同定义引起的。本小节讨论不同对比损失的特性。

论文[135]指出了 InfoNCE 对比损失的两个关键属性。

Alignment：锚点的特征向量应该在特征空间中接近正样本的特征向量。

Uniformity：归一化的特征向量应该均匀分布在单位超球面上。

根据这个分析，$\mathcal{L}^{scst}$ 可以分解为 $\mathcal{L}_{align}$ 和 $\mathcal{L}_{uniform}$ 两部分：

$$\begin{aligned}\mathcal{L}^{scst} &= -\sum_{i}\sum_{j\neq i}\frac{S_{ij}}{\tau}\boldsymbol{z_i}\cdot\ \boldsymbol{z_j} + \sum_{i}\log\sum_{k\neq i}\exp(\frac{\boldsymbol{z_i}\cdot\boldsymbol{z_k}}{\tau}) \\ &= \frac{1}{2\tau}\sum_{i}\sum_{j\neq i}S_{ij}(||\boldsymbol{z_i}-\ \boldsymbol{z_j}||^2-1) + \sum_{i}\log\sum_{k\neq i}\exp(\boldsymbol{z_i}\cdot\boldsymbol{z_k}/\tau) \\ &= \mathcal{L}_{align}+\mathcal{L}_{uniform}.\end{aligned} \tag{5.17}$$

从以上分解可以看出，语义相似度 $s(\cdot,\cdot)$ 的定义只影响 $\mathcal{L}_{align}$。S_{ij} 表示样本对 i 和 j 之间的对齐程度。自监督的 InfoNCE 损失仅将锚点的两种视图对齐，而 SupCon 则将对齐范围扩展到全部来自同一类的样本。SupCon-ST 进一步将对齐范围扩展到所有样本。理论上，当一个编码器将所有输入映射到单个特定的特征向量时，这个编码器就完美满足了 Alignment 特性。这样会产生名为特征彤缩的现象，但 Uniformity 特性让特征向量趋向于均匀分布，避免了特征彤缩现象的出现。

另一方面，$\mathcal{L}_{uniform}$ 与 $s(\cdot,\cdot)$ 无关，因此 Uniformity 特性在 InfoNCE 损失的所有变体中都是相同的。当特征向量的分布遵循单位超球面上的均匀分布时，$\mathcal{L}_{uniform}$ 被最小化。

Uniformity 特性进一步引出了对比损失的挖掘困难负样本的能力。具体来说，当 $\tau\to 0^+$ 时，有以下关于锚点 i 的 $\mathcal{L}_{uniform}$ 的近似值：

$$\begin{aligned}\lim_{\tau\to 0^+}\mathcal{L}^{i}_{uniform} &= \lim_{\tau\to 0^+}\log\sum_{j\neq i}\exp(\boldsymbol{z_i}\cdot\boldsymbol{z_j}/\tau) \\ &= \lim_{\tau\to 0^+}\frac{1}{\tau}\max_{j\neq i}\boldsymbol{z_i}\cdot\boldsymbol{z_j}.\end{aligned} \tag{5.18}$$

τ 越趋近于 0，$\mathcal{L}_{uniform}$ 越集中于分离锚点和与它距离最近的样本。然而，均匀性损失没有考虑样本之间的语义相似性。这样导致了一种结果，在有监督的对比学习中，因为距离锚点最近的样本更可能是锚点的同类样本，所以这样反而导致类内样本距离增加，挖掘硬负样本的能力被削弱。

这种现象在论文[140] 中被描述为正负耦合现象 (negative-positive-coupling effect)，这篇论文还提出了解耦对比损失 (decoupled contrastive loss) 来消除这种影响。解耦对比损失从 InfoNCE 损失的分母中的总和项中删除了正例。从上

面的分析可以看出，这个修改去掉了 $\mathcal{L}_{uniform}$ 中的正例，使得 $\mathcal{L}_{uniform}$ 可以专注于负例。在 SupCon-ST 中加入解耦机制将是本文后续的研究内容之一，有望对 SupCon-ST 起到改进作用。

5.5 实验验证

表 5.1 根据 AUC-ROC 指标评估的开集识别实验结果

数据集	MNIST	SVHN	CIFAR-10	CIFAR+10	CIFAR+50	TinyImageNet
Openness	22.54%	22.54%	22.54%	46.55%	72.78%	68.38%
Softmax	97.8	88.6	67.7	81.6	80.5	57.7
OpenMax	98.1	89.4	69.5	81.7	79.6	57.6
G-OpenMax	98.4	89.6	67.5	82.7	81.9	58.0
OSRCI	98.9	91.0	69.9	83.8	82.7	58.6
C2AE	98.9	89.2	71.1	81.0	80.3	58.1
RPL++	99.3	95.1	86.1	85.6	85.0	70.2
GFROSR	N.R	93.5	80.7	92.8	92.6	60.8
PROSER	N.R	94.3	89.1	96.0	95.3	69.3
ARPL	99.7	96.7	91.0	97.1	95.1	78.2
ConOSR (vanilla SupCon)	99.7	98.8	93.7	97.9	97.0	79.6
ConOSR (SupCon-ST)	99.7	99.1	94.2	98.1	97.3	80.9

本节通过在基准数据集上的实验将 ConOSR 与最先进的开集识别方法进行比较，测试了各种方法在传统的闭集分类和开集识别任务中的性能。在所有实验中，ConOSR 采用了 VGG19[40]作为特征编码器网络，与论文[126] 中使用的网络结构相同。深度 OSR 网络的性能也与其骨干网络的学习能力正相关。用来进行对比的 OSR 算法都是基于深度学习的方法，在各自的实验中也采用了同样的网络结构。对比学习步骤中的投影网络是一个具有两个全连接层的 MLP，均由 128 个节点组成。分类网络也是一个具有 128 个节点的全连接隐藏层的 MLP。

5.5.1 未知样本检测

最近关于开集识别的研究工作通常采用论文[126]中定义的实验方法。通过随机选择 k 个类作为已知数据，将多类分类数据集分为两个子集，留下剩余的类来模拟开集识别场景中的开放空间。数据集的拆分会显着影响开集识别实验的结果。因此，为了公平比较，本组实验中的数据集拆分与提出 ARPL 的论文[141]中的相同。实验中使用基准数据集信息如下：

MNIST \ SVHN \ CIFAR-10：这些数据集是有 10 个类的分类数据集，其中 6 个类被选为已知类，另外 4 个类被用作未知类。

CIFAR+10 \ CIFAR+50：从 CIFAR-10 中选择 4 个类作为已知类，从 CIFAR-100 中选择 10 \ 50 个类作为未知类。

TinyImageNet：TinyImageNet 由 200 个类组成。其中 20 个类作为已知类，其余 180 个类作为未知类。

本次实验旨在测试开集识别算法对未知样本的检测能力，因此选用了 AUROC 用作评估指标。AUROC 是一个与阈值无关的指标，可以解释为正例被分配的检测分数高于负例的概率[18]。为了比较公平，采用与论文[141]相同的数据拆分进行五次试验，取平均结果进行比较。

每个开集识别实验的难度由 Openness 衡量，在论文[126]中被定义为

$$Openness = 1 - \sqrt{K/M} \tag{5.19}$$

其中 K 和 M 表示分别是训练中使用的类别数量和测试中出现的类别总数。

表5.1显示了这组实验的结果。比较基线有 Softmax Thresholding[117], OpenMax[120], G-OpenMax[125], OSRCI[126], C2AE[142], RPL++[121]、GFROSR[129]、PROSER[122] 和 ARPL[141]，这些算法的实验结果引用自论文[122,141]。N.R 表示原论文没有报告相应的结果。

表5.1的最后两行报告了 ConOSR 框架的两种变体的实验结果。第一个变体在对比学习步骤中使用了基础的 SupCon 算法，而第二个变体则使用本章提出的 SupCon-ST。后文中的 ConOSR 均为第二种变体。

从表 5.1 可以看出，几乎所有的方法在数字识别数据集 MNIST 和 SVHN 上都取得了很好的效果。特别是 MNIST 上的结果已经基本饱和，难以再有明显提升。尽管如此，ConOSR 仍然将 SVHN 上的 AUROC 提高到 99.1%。在自然图像数据集上，ConOSR 也取得了比 SOTA 方法 PROSER 和 ARPL 更好的结果。与第二好的方法 ARPL 相比，ConOSR 在 TinyImageNet 数据集上的结果提高了 2.7%。

表5.1中的结果也表明，用 SupCon-ST 替换 SupCon 可以获得更好的未知样本检测结果。SupCon-ST 在 SVHN 和 CIFAR 等简单数据集上将 AUROC 提高了 0.2-0.5，在难度最高的数据集 TinyImageNet 上的提升则有 1.3。

5.5.2 闭集分类

另一组实验验证 SupCon-ST 在传统分类任务上的有效性，将 SupCon 算法和普通的 VGG19 网络作为对比基准。VGG19 网络的训练过程使用了与 SupCon-ST 相同的数据增强方法，但因为 SupCon 不能支持软标签，所以仅使用了 RandAugment。这组实验使用的数据集是 CIFAR-10、CIFAR-100 和 TinyImageNet 的前 100 类。表 5.2 报告了 5 次随机试验的平均结果。

表 5.2 闭集分类的平均准确率对比

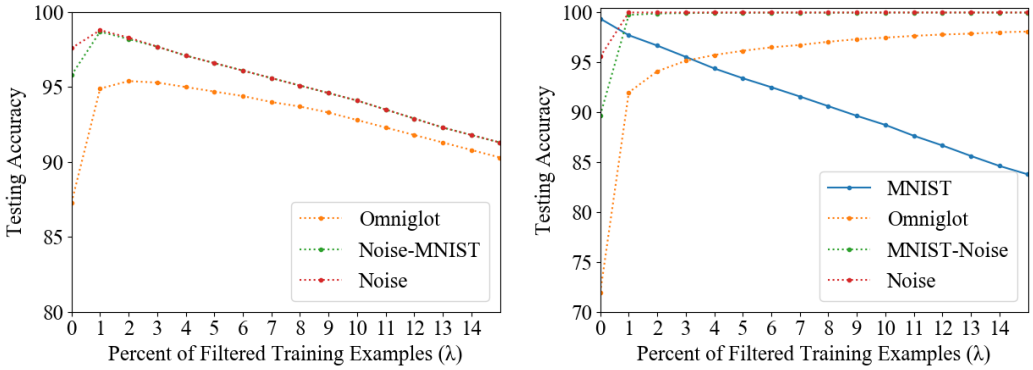
数据集	CIFAR-10	CIFAR-100	TinyImageNet
Plain VGG19	94.0	71.6	63.7
ARPL	94.1	72.1	N.R.
SupCon	94.1	72.4	63.7
SupCon-ST	94.6	73.0	66.1

从结果中可以看出，SupCon 的准确性与 VGG19 的分类准确率接近，仅在 CIFAR-100 上提升较为明显。而 SupConST 的准确率则有相对较大的领先优势。这些结果表明，有监督的对比学习提高了传统封闭集识别任务的分类准确性。然而，由于与软标签不相容，SupCon 能够使用的训练技巧比其他方法少，这导致了它的潜力没有得到充分发挥。

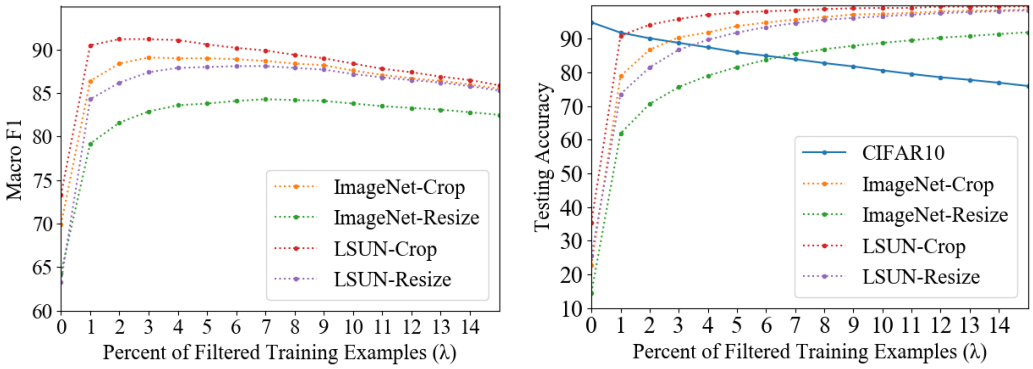
表中引用了 ARPL[141]的实验结果进行对比。论文[141]作者在进行 ARPL 的闭集分类实验时使用了 ResNet-34[41]作为其骨干架构，这是一个性能比 VGG19 更强的卷积网络。但另一方面，ARPL 的实验并没有使用那么多的数据增强。大多数现有的开集识别论文报告的闭集分类结果都低于同架构的普通卷积神经网络。ARPL 是少数优于普通卷积网络的方法之一，因为 ARPL 同样是以改进开集识别系统的表示学习部分为出发点的研究成果，因此它在传统分类任务中的表现也得到了提升。论文[118]中详细分析了闭集分类准确度与开集识别性能之间的正相关关系。

5.5.3 开集识别

本组实验遵循论文[122]中使用的实验方法，验证 ConOSR 在开集识别任务中的性能。在训练时，一个基准数据集中的完整训练集被用于训练开集识别模型。在测试期间，来自另一个数据集的测试样本被添加到测试集中，并被视为一



(a) MNIST 实验的 Marco F1　　(b) MNIST 实验的准确率



(c) CIFAR-10 实验的 Marco F1　　(d) CIFAR-10 实验的准确率

图 5.4　分类准确率、macro F1 与超参数 λ 的关系

个新类别。这组实验的评估指标是所有类别的 macro F1-index。

第一个实验使用 MNIST 数据集进行，在测试阶段向测试集中添加三种样本作为未知样本：Omniglot 数据集中的样本[143]、噪声样本和添加了噪声的 MNIST 样本 (MNIST-Noise)。与论文[122]中的实验方式相同，未知样本的数量设置为 10,000，使其数量等于 MNIST 数据集的测试样本数量。Omniglot 的测试集包含 13,180 个样本，因此本组实验选择按文件名升序排序前 10,000 张的图像。在合成噪声数据时，从 [0,255] 区间上的均匀分布中进行采样取整，得到噪声图像的每个像素的灰度值。MNIST-Noise 是通过在 MNIST 的测试样本上叠加噪声图像来合成的。

第二个实验在 CIFAR-10 数据集上进行，并在测试阶段加入另外两个数据集的测试集作为未知样本：TinyImageNet 和 LSUN[144]。CIFAR-10、TinyImageNet 和 LSUN 都有一个包含 10,000 张图像的测试集。为了消除 CIFAR-10 和 TinyImageNet & LSUN 之间的图像大小差异，本组实验使用两种不同的方式来处理未知图像：(1) 将图像大小调整为 32×32；(2) 从每张图像中随机裁剪出

表 5.3 MNIST 上的开集识别实验结果。三种不同来源的样本作为一个新的类添加到测试集，表中报告 11 个类的 Macro F1 指标。

数据集	Omniglot	Noise-MNIST	Noise
Softmax	59.5	64.1	82.9
OpenMax	68.0	72.0	82.6
CROSR	79.3	82.7	82.6
PROSER	86.2	87.4	88.2
ConOSR	95.4	98.7	98.8

表 5.4 CIFAR10 上的开集识别实验结果。TinyImageNet(TIN) 数据集和 LSUN 数据集的测试样本添加到测试集作为一个新的类，表中报告 11 个类的 Macro F1 指标。

数据集	TIN (Crop)	TIN (Resize)	LSUN (Crop)	LSUN (Resize)
Softmax	63.9	65.3	64.2	64.7
OpenMax	66.0	68.4	65.7	66.8
OSRCI	63.6	63.5	65.0	64.8
CROSR	72.1	73.5	72.0	74.9
GFROSR	75.7	79.2	75.1	80.5
PROSER	84.9	82.4	86.7	85.6
ConOSR	89.1	84.3	91.2	88.1

32×32 的小块。

这些实验的结果显示在表 5.3 和表 5.4 中，其中其他方法的结果引用自论文[122]。与未知类检测的 AUROC 指标不同，macro F1 指标会受到拒绝阈超参数 λ 的影响。因此，本组实验中 λ 最优值是通过区间 $[1, 15]$ 上的网格搜索来设置的，表中报告的是最佳的 macro F1 值。

在表 5.3 中可以看到当训练图像的背景干净时，检测噪声图像是一项简单的任务。检测来自 Omniglot 数据集的未知样本最具挑战性，主要是因为未知样本与训练样本一样干净。在这组实验中，ConOSR 明显优于其他方法。当未知样本是带有噪声的图像时，ConOSR 与第二好的方法 PROSER 之间的精度差距大于 10%。

在第二个实验中，ConOSR 在所有数据集上的表现均优于其他方法。从 Table 5.4可以看出，在以裁剪方式得到未知样本的情况下，OSR 的优势更加明显。这种现象表明 ConOSR 在检测语义不明确的图像时效果更好，因为对比学习算法侧重于学习用于区分不同类别的最明显的特征，而随机裁剪的图像块通常不含有的此类特征。当未知类的样本是经过缩放的图像时，ConOSR 与第二好的方法 PROSER 相比有大约 2% 的提升。

因为 macro-F1 很容易受到超参数 λ 的取值的影响，所以这组实验的超参数调试过程可以用来分析 λ 是如何影响结果的。图5.4展示了将 λ 的值设置为 $[0, 15]$ 区间内的整数时，已知类和未知类的分类准确率以及 macro F1-index 如何随 λ 的变化而变化。

增加 λ 会增加未知实例的分类精度，同时降低已知实例的分类精度。在简单的任务中，例如从 MNIST 图像中检测噪声异常值，当 $\lambda = 1$ 时未知样本的准确率就已经达到 100%，因此进一步增加 λ 只会导致结果下降。在 CIFAR-10 实验中，未知实例的分类准确率则只在设置较大的 λ 时才能接近极限。当已知类的分类准确率接近未知类的分类准确率时，macro F1-index 的值通常也比较高。默认设置值 $\lambda = 5$ 在 CIFAR-10 实验中也能产生良好的 macro F1-index。

5.5.4 分析实验

本小节用另一个实验来分析对比学习可以提高开集识别能力的原因。在这里首先提出一个简单的分析思路。

与许多基于深度网络的判别式开集识别算法类似，ConOSR 通过对网络的输出设置拒绝阈来检测未知样本，并当测试样本让网络产生低于拒绝阈的输出值时将其判定为未知样本。从深度神经网络的原理可以推断出，测试样本被拒绝的原因是因为网络的激活程度不够。换句话说，这类方法检测未知样本的原理是检测样本中是否缺失使其被识别为任何已知类别的必要特征，而不是检测是否出现训练阶段未曾学习过新特征。最近的一项研究[145] 将此属性命名为 “熟悉度假设”，并提供了强有力的证据来支持该假设。

根据论文[137]的分析，对比学习的作用类似于非线性独立成分分析。因此有监督的对比学习侧重于学习与类别标签相关的特征，即将一个类别与其他类别区分开的关键特征，而降低了跨类别通用特征的影响。这些关键特征存在于类别未知的预测样本中的可能性更低，从而降低了未知检测的难度。但另一方面，这种特征与类别高度相关，因此它们不能很好地泛化到另一个领域。

本文使用另一项实验验证上述分析，在 CIFAR-100 和 TinyImageNet 数据集上将对比学习得到的特征与普通 CNN 网络学习的特征进行比较。每个数据集根据标签顺序分为两半。前半部分用作开集识别任务中的训练数据，后半部分用于模拟测试阶段的未知数据。首先以开集识别的实验方式训练和测试模型，

表 5.5 开集识别能力与特征泛化能力的对比

数据集	CIFAR-100			TinyImageNet		
指标	准确率 (已知类)	准确率 (未知类)	AUROC	准确率 (已知类)	准确率 (未知类)	AUROC
Plain CNN	77.2	62.6	76.7	63.7	49.1	68.1
ConOSR (SupCon)	77.8	59.3	77.9	63.8	41.3	71.6
ConOSR (SupCon-ST)	79.5	60.5	79.1	66.1	45.4	72.1

记录 AUROC 分数和闭集分类精度。然后，固定特征编码器的参数，在它们之上用未知类的训练数据训练新的分类器。最后，记录新分类器在未知类上的准确率，通过这种方式来了解编码器提取的特征能否在未知数据上良好地泛化。

本组实验的结果如表 5.5所示。已知类的闭集分类结果与 Table 5.2 中的结果相似。SupCon 和普通 CNN 的精度接近，而 SupCon-ST 明显优于两者。当特征编码器迁移到另一个域时却产生了相反的现象，在普通 CNN 提取的特征上训练的新分类器要好于在对比学习到的表征上训练的新分类器。在 AUROC 分数方面，ConOSR 的两种变体在检测未知样本方面都比普通 CNN 更好。具体来说，SupCon 在闭集分类准确度方面与普通 CNN 接近，但仍然获得了更好的 AUROC 分数。SupCon-ST 在所有评估指标方面都优于 SupCon，表明使用 Mixup 和标签平滑能够带来更好的效果。

这个实验的结果为上面的分析提供了支持，这表明有监督的对比学习更“专注”于区分他们学习的类别。因此，学习到的特征更加独特，在另一个领域上的通用性不如更低层次的特征。但是，这一特点使分类器更容易检测到特征的缺失，这有利于它们在开集识别任务中的表现。

为了更好地展示所提出方法的这一特性，本文使用类激活图 (class activation map) 来展示对比学习和普通 CNN 网络学习的特征之间的差异。从 CIFAR-100 中随机选择 4 对已知/未知图像，使用已知类的分类器权重计算两幅图像的类激活图。热图是根据两个激活图中的最小/最大激活值绘制的。类激活图如图5.5所示。从类别激活图的比较中，可以看出 ConOSR 激活图中的热点区域比普通 CNN 激活图中的热点区域小得多，而且集中在目标的重要部分，例如婴儿的脸部、自行车的轮子等部位。通过比较未知图像的激活图，可以看到 ConOSR 为未知样本生成的激活图的颜色比普通 CNN 的激活图深得多，表明 ConOSR 对未知样本的激活程度比普通 CNN 的激活程度更低。以上结果也支持本节的分析，即对比学习更侧重于每个类别的最关键的特征，因此更容易检测特征的

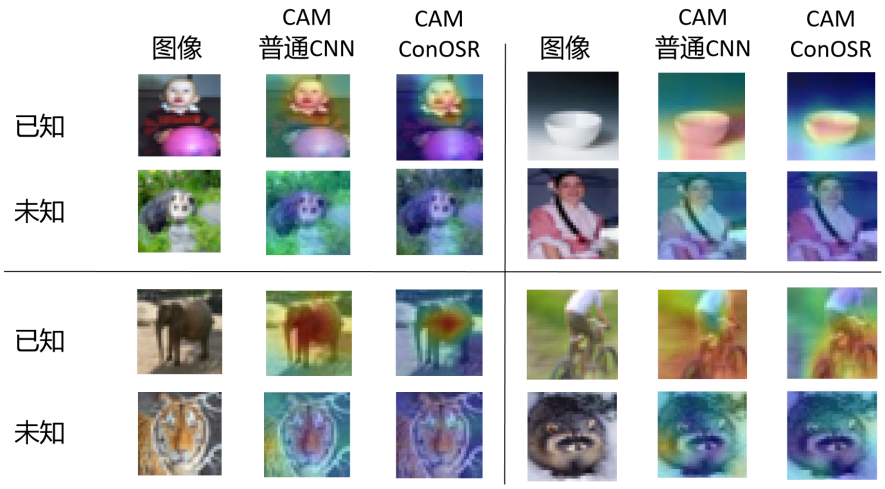


图 5.5　普通 CNN 网络和 ConOSR 的类激活图。从 CIFAR-100 中随机选择 4 对已知/未知图像，使用已知类的权重计算两幅图像的类激活图。

缺失。

5.6　本章小结

在现实世界的识别场景中，很难收集训练示例以覆盖所有潜在测试实例的类别。开集识别是针对这一困难的一种现实类型的识别任务，它要求分类器将测试样本与未见类区分开来，同时保持已见类的高分类精度。从表示学习的角度来看，本章提出了一种基于软目标的监督对比学习算法 SupCon-ST，并在此基础上提出基于对比学习的开集识别算法 ConOSR。SupCon-ST 能够在对比学习的训练阶段利用标签平滑和混合，从而使深度网络在开集识别任务中具有更好的鲁棒性，在闭集分类中也具有更高的准确性。本章的研究成果形成了一篇会议论文，发表于 AAAI-23。

然而，与常见的深度学习方法相比，本章所提出的方法在计算上效率不高。首先，与通常的有监督深度学习相比，对比学习需要更多的训练周期来收敛。其次，SupCon-ST 需要更多 GPU 显存才能正常工作。本章的实验在训练阶段必须根据类的数量设置每次输入给神经网络的小批样本数量 (batchsize)，尽量让每一批训练样本中都有全部类别的正样本对，因此在类别多的大规模数据集上各批次的样本数量会很大。更糟糕的是，如果每个训练批次包含 n 个训练样本，那么 RandAugment 算法会产生 $2n$ 个样本视图，而且 Mixup 算法会进一步组合出 $2n$ 个虚拟样本。因此，训练中的内存空间成本随着类别数量的增加而急剧

增加。

本章的后续研究工作将研究如何将所提出的方法与聚类方法相结合，以便降低算法的空间成本。另一个方向是将开集识别与增量学习结合，扩展到终身学习场景，即开放世界识别问题。将 SupCon-ST 用于传统的监督学习任务也是一个有潜力的研究问题。

第六章　总结和展望

6.1　工作总结

传统的机器学习与人类的学习方式有着明显的区别。人类的学习过程是开放的、终生的，而机器进行学习的方式则是一个封闭的训练的过程。本文研究模拟人类学习方式的机器学习算法，采用神经网络作为主要工具，面向增量与开放式的学习问题展开研究工作。

本文的研究工作分为无监督学习问题与监督学习问题两个部分。在无监督学习环境下，本文重点研究数据流的聚类问题。本文的第三章使用竞争型神经网络来在线增量地学习数据流的可信代表集，并提出了基于密度的聚类算法 DenSOINN，来生成数量不定、形状不定的聚类以适应数据分布。DenSOINN 还提出了一种自适应距离度量函数，能够增量地对数据流进行规范化。

在监督学习环境下，本文的第四章研究类增量学习问题。虽然模型能够借助从训练数据中采样的样本代表集，通过复习的方式实现增量学习，但这与增量学习的宗旨并不相符。因此本文限制了模型访问历史数据，在这种环境下分析增量学习需要克服的难题。通过对网络训练方式进行理论分析，本文指出了类增量学习中存在的目标抑制现象与开放空间风险，并提出了一种基于原型神经网络与网络权重加固的类增量学习算法 PBNC。

接着，本文在第五章进一步面向开放性的监督学习问题进行深入研究。本文从提升深度神经网络的表征学习能力为出发点，提出了新型对比表征学习算法 SupCon-ST。以此为基础设计了开集识别算法 ConOSR，并分析了对比学习在开集识别问题上的有效性。

总的来说，本文针对增量与开放式的学习问题展开了一系列研究，取得了多项研究成果并得到了研究同行的认可。

6.2 工作展望

本文以增量和开放式机器学习算法为主要研究内容，但这一研究领域的覆盖面广泛，存在着大量值得深入研究的问题，而本文只研究了其中一部分。本文的工作需要在今后的研究工作中不断深入和完善，今后将重点研究的方向包括：

1）完善算法的理论研究。神经网络作为一个黑盒子模型，其工作机制缺少可解释性。本文虽然对增量与开放式学习问题进行了一定的分析，但依然高度依赖神经网络模型本身的优秀学习能力，而对于算法有效性的理论分析仍有所欠缺。尤其是在增量学习问题上，当前主流的深度神经网络依旧受灾难性遗忘现象的困扰，而从根本上克服这一现象的方法仍待研究发现，这也就需要从神经网络的学习机制上进行更深入的理论分析。对神经网络的学习机制进行深入分析同样有助于对算法的执行效果进行更准确的判断，分析现有工作的不足之处，发现对算法进行改进的思路。

2）完善本文提出的对比表征学习算法框架，并将其应用于更多的学习问题上。第五章提出的对比表征学习框架 SupCon-ST 是本文最新的研究成果，不仅在开集识别问题上取得了良好效果，而且可用于提升神经网络在传统识别问题上的准确率。这一框架突破了过去的对比学习算法中存在的一些限制，使对比学习的训练样本标签可以是任意实数向量，因此有望被应用于更多的学习问题上。例如多标签分类、多输出回归等学习问题，其中的训练样本标签都并非 one-hot 向量，因此过去的对比学习算法无法在这类问题上应用。

另一方面，受到研究开展顺序的关系，本文尚未将这一框架用来解决增量学习问题。SupCon-ST 同样可以和本文第四章使用的知识蒸馏方法结合，因此将其加入 PBNC 的框架中并非难事。类增量学习同样面临着开放空间风险，加入对比学习来提升同类样本的特征相似度、分离不同类样本，这是一种有潜力的研究方案。

SupCon-ST 本身的性能也有进一步优化的空间。如第五章小结中所分析的，SupCon-ST 的数据增强步骤使用了太多的数据视图，大幅增加了训练开销。其他的对比学习算法同样存在着计算开销高昂的缺陷，因此如何对其训练效率进行优化也是一个潜在的研究方向。

3）研究增量式的无监督表征学习算法。无监督表征学习是无监督学习在深

度学习时代最受关注的研究课题。因为使用大规模数据训练的高质量表征模型可以被迁移到多个下游任务，而且效果已经达到了接近监督训练的水准。这种训练模式比在下游任务上直接训练模型要更加高效。本文在研究无监督学习问题时并未使用深度学习技术，因此算法在图像、自然语言等复杂数据上的表现与最新的深度聚类算法存在差距。

另一方面，对比学习算法也是无监督表征学习的常用方法。本文已经在无监督增量学习问题、对比学习方法上积累了一些研究基础，因此将其延伸到无监督表征学习领域是一种可行的思路。

4) 研究监督学习环境下增量学习与开集识别问题的融合。本文在监督学习领域的研究工作先后面向两个问题展开，而这两个研究问题中实际上存在着一些共性。增量学习与开集识别都是具有开放性的学习范式，但它们的开放性分别体现在监督学习的训练阶段和测试阶段。因此，将增量学习问题与开集识别问题结合，使模型在训练与测试阶段都表现出开放式的特点，这种范式即是被称为“开放世界识别”的学习范式。将本文第四章、第五章的研究内容进行综合，扩展到开放世界识别领域，这将是笔者后续的研究工作之一。

参考文献

[1] CHEN Z, LIU B. Lifelong Machine Learning[J]. Synthesis Lectures on Artificial Intelligence and Machine Learning, 2016, 10(3): 1-145.

[2] SCHEIRER W J, de REZENDE ROCHA A, SAPKOTA A, et al. Toward open set recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2012, 35(7): 1757-1772.

[3] HARTIGAN J A, HARTIGAN J. Clustering algorithms: vol. 209[M]. Wiley New York, 1975.

[4] JOLLIFFE I. Principal component analysis[M]. Wiley Online Library, 2002.

[5] 邓齐林, 邱天宇, 申富饶, 等. 一种自适应在线核密度估计方法[J]. 软件学报, 2020, 31(4): 16.

[6] DOU H, XU B, SHEN F, et al. V-SOINN: A Topology Preserving Visualization Method for Multidimensional Data[J]. Neurocomputing, 2021.

[7] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative Adversarial Nets[C]//Neural Information Processing Systems. 2014.

[8] KOLESNIKOV A, ZHAI X, BEYER L. Revisiting self-supervised visual representation learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 1920-1929.

[9] ANDERSON J A. An introduction to neural networks[M]. MIT press, 1995.

[10] LAROCHELLE H, BENGIO Y, LOURADOUR J, et al. Exploring strategies for training deep neural networks.[J]. Journal of machine learning research, 2009, 10(1).

[11] BENGIO Y, COURVILLE A, VINCENT P. Representation learning: A review and new perspectives[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(8): 1798-1828.

[12] SYED N A, HUAN S, KAH L, et al. Incremental learning with support vector machines[J]., 1999.

[13] ZHOU D W, WANG Q W, QI Z H, et al. Deep class-incremental learning: A survey[J]. arXiv preprint arXiv:2302.03648, 2023.

[14] DEVLIN J, CHANG M W, LEE K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[15] RADFORD A, WU J, CHILD R, et al. Language models are unsupervised multitask learners[J]. OpenAI blog, 2019, 1(8): 9.

[16] BROWN T, MANN B, RYDER N, et al. Language models are few-shot learners[J]. Advances in neural information processing systems, 2020, 33: 1877-1901.

[17] HU E J, yelong Shen, WALLIS P, et al. LoRA: Low-Rank Adaptation of Large Language Models[C/OL]//International Conference on Learning Representations. 2022. https://openreview.net/forum?id=nZeVKeeFYf9.

[18] GENG C, HUANG S J, CHEN S. Recent advances in open set recognition: A survey[J]. IEEE transactions on pattern analysis and machine intelligence, 2020.

[19] BENDALE A, BOULT T. Towards open world recognition[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1893-1902.

[20] JOSEPH K, KHAN S, KHAN F S, et al. Towards open world object detection[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 5830-5840.

[21] KOHONEN T. Self-Organized Formation of Topologically Correct Feature

Maps[J]. Biological Cybernetics, 1982, 43(1): 59-69. DOI: 10.1007/BF00337288.

[22] GROSSBERG S. Adaptive Pattern Classification and Universal Recoding: I. Parallel Development and Coding of Neural Feature Detectors[J]. Biological Cybernetics, 1976, 23(3): 121-134. DOI: 10.1007/BF00344744.

[23] SHEN F, HASEGAWA O. An incremental network for on-line unsupervised classification and topology learning[J]. Neural Netw, 2006, 19(1): 90-106.

[24] MARTINETZ T. Competitive Hebbian Learning Rule Forms Perfectly Topology Preserving Maps[C]//GIELEN S, KAPPEN B. ICANN '93. London: Springer London, 1993: 427-434. DOI: 10.1007/978-1-4471-2063-6_104.

[25] MARTINETZ T, SCHULTEN K. Topology Representing Networks[J]. Neural Networks, 1994, 7(3): 507-522. DOI: 10.1016/0893-6080(94)90109-0.

[26] FURAO S, OGURA T, HASEGAWA O. An enhanced self-organizing incremental neural network for online unsupervised learning.[J]. Neural Networks the Official Journal of the International Neural Network Society, 2007, 20(8): 893.

[27] SHEN F, HASEGAWA O. A fast nearest neighbor classifier based on self-organizing incremental neural network[J]. Neural Networks, 2008, 21(10): 1537-1547.

[28] QIU T, SHEN F, ZHAO J. Local adaptive and incremental gaussian mixture for online density estimation[C]//Pacific-Asia Conference on Knowledge Discovery and Data Mining. 2015: 418-428.

[29] XING Y, SHI X, SHEN F, et al. A Self-Organizing Incremental Neural Network Based on Local Distribution Learning[J]. Neural Networks, 2016, 84: 143-160. DOI: 10.1016/j.neunet.2016.08.011.

[30] SHEN F, OUYANG Q, KASAI W, et al. A general associative memory based on self-organizing incremental neural network[J]. Neurocomputing, 2013, 104: 57-71.

[31] SHEN F, HASEGAWA O. A fast nearest neighbor classifier based on self-organizing incremental neural network.[J]. Neural Networks the Official Journal of the International Neural Network Society, 2008, 21(10): 1537.

[32] LOWE D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60: 91-110.

[33] DALAL N, TRIGGS B. Histograms of oriented gradients for human detection[C]//2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05): vol. 1. 2005: 886-893.

[34] LECUN Y, BENGIO Y, HINTON G. Deep learning[J]. nature, 2015, 521(7553): 436.

[35] KIEFER J, WOLFOWITZ J. Stochastic estimation of the maximum of a regression function[J]. The Annals of Mathematical Statistics, 1952: 462-466.

[36] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors[J]. nature, 1986, 323(6088): 533-536.

[37] KINGMA D P, BA J. Adam: A Method for Stochastic Optimization[J]. international conference on learning representations, 2015.

[38] REDDI S J, KALE S, KUMAR S. On the convergence of adam and beyond [J]. arXiv preprint arXiv:1904.09237, 2019.

[39] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[40] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[41] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[42] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[43] JAIN A K, DUBES R C. Algorithms for clustering data[J]. Technometrics, 1988, 32(2): 227-229.

[44] JAIN A K, MURTY M N, FLYNN P J. Data clustering: a review[M]. ACM, 1999: 264-323.

[45] LEE R C T. Clustering Analysis and Its Applications[M]. Springer US, 1981: 169-292.

[46] SRIVASTAVA A, SAHAMI M. Text Mining: Classification, Clustering, and Applications[M]. 2009.

[47] MA A, ZHONG Y, ZHANG L. Adaptive Multiobjective Memetic Fuzzy Clustering Algorithm for Remote Sensing Imagery[J]. IEEE Transactions on Geoscience & Remote Sensing, 2015, 53(8): 4202-4217.

[48] ZHONG Y, MA A, ONG Y S, et al. Computational intelligence in optical remote sensing image processing[J]. Applied Soft Computing, 2018, 64: 75-93.

[49] SILVA J A, FARIA E R, BARROS R C, et al. Data stream clustering: A survey[J]. Acm Computing Surveys, 2013, 46(1): 13.

[50] CAO F, ESTER M, QIAN W, et al. Density-Based Clustering over an Evolving Data Stream with Noise[C]//Siam International Conference on Data Mining, April 20-22, 2006, Bethesda, Md, Usa. 2006: 328-339.

[51] GHESMOUNE M, LEBBAH M, AZZAG H. A new Growing Neural Gas for clustering data streams.[J]. Neural Networks the Official Journal of the International Neural Network Society, 2016, 78(3-4): 36-50.

[52] MACQUEEN J. Some Methods for Classification and Analysis of Multi-Variate Observations[C]//Proc. of Berkeley Symposium on Mathematical Statistics and Probability. 1967: 281-297.

[53] ESTER M, KRIEGEL H P, XU X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise[C]//International Conference Knowledge Discovery and Data Mining. 1996: 226-231.

[54] AKSOY S, HARALICK R M. Feature normalization and likelihood-based similarity measures for image retrieval[J]. Pattern Recognition Letters, 2001, 22(5): 563-582.

[55] ZHANG T, RAMAKRISHNAN R, LIVNY M. BIRCH: A New Data Clustering Algorithm and Its Applications[J]. Data Mining & Knowledge Discovery, 1997, 1(2): 141-182.

[56] AGGARWAL C C, HAN J, YU P S, et al. A framework for clustering evolving data streams[C]//VLDB. 2003: 81-92.

[57] ACKERMANN M R, RAUPACH C, SWIERKOT K, et al. StreamKM++: A clustering algorithm for data streams[J]. Journal of Experimental Algorithmics, 2012, 17: 2.4.

[58] ZHANG X, FURTLEHNER C, GERMAIN-RENAUD C, et al. Data Stream Clustering With Affinity Propagation[J]. IEEE Transactions on Knowledge & Data Engineering, 2014, 26(7): 1644-1656.

[59] LLOYD S. Least squares quantization in PCM[J]. IEEE Transactions on Information Theory, 1982, 28(2): 129-137.

[60] ARTHUR D, VASSILVITSKII S. k-means++:the advantages of careful seeding[C]//Eighteenth Acm-Siam Symposium on Discrete Algorithms. 2007: 1027-1035.

[61] FREY B J, DUECK D. Clustering by passing messages between data points[J]. science, 2007, 315(5814): 972-976.

[62] DING S, WU F, QIAN J, et al. Research on data stream clustering algorithms[J]. Artificial Intelligence Review, 2015, 43(4): 593-600.

[63] KHAN I, HUANG J Z, IVANOV K. Incremental density-based ensemble clustering over evolving data streams[J]. Neurocomputing, 2016, 191: 34-43.

[64] CHEN J Y, HE H H. A fast density-based data stream clustering algorithm with cluster centers self-determined for mixed data[M]. Elsevier Science Inc., 2016: 271-293.

[65] HYDE R, ANGELOV P, MACKENZIE A R. Fully online clustering of evolving data streams into arbitrarily shaped clusters[J]. Information Sciences, 2017, s 382-383: 96-114.

[66] KOHONEN T. Self-organized formation of topologically correct feature maps[J]. Biological Cybernetics, 1982, 43(1): 59-69.

[67] MARTINETZ T M, BERKOVICH S G, SCHULTEN K J. 'Neural-gas' network for vector quantization and its application to time-series prediction[J]. IEEE Trans Neural Netw, 1993, 4(4): 558-569.

[68] BOUGUELIA M R, BELAÏD Y, BELAÏD A. An Adaptive Incremental Clustering Method Based on the Growing Neural Gas Algorithm[C]// International Conference on Pattern Recognition Applications and Methods. 2013: 42-49.

[69] ZHANG H, XIAO X, HASEGAWA O. A Load-Balancing Self-Organizing Incremental Neural Network[J]. IEEE Transactions on Neural Networks & Learning Systems, 2014, 25(6): 1096-1105.

[70] XIE J, GIRSHICK R, FARHADI A. Unsupervised deep embedding for clustering analysis[C]//International conference on machine learning. 2016: 478-487.

[71] CARON M, BOJANOWSKI P, JOULIN A, et al. Deep clustering for unsupervised learning of visual features[C]//Proceedings of the European

conference on computer vision (ECCV). 2018: 132-149.

[72] YAN Y, HAO H, XU B, et al. Image clustering via deep embedded dimensionality reduction and probability-based triplet loss[J]. IEEE Transactions on Image Processing, 2020, 29: 5652-5661.

[73] ZHAN X, XIE J, LIU Z, et al. Online deep clustering for unsupervised representation learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 6688-6697.

[74] LI Y, YANG M, PENG D, et al. Twin contrastive learning for online clustering[J]. International Journal of Computer Vision, 2022, 130(9): 2205-2221.

[75] KUMAR S, HARESH S, AHMED A, et al. Unsupervised action segmentation by joint representation learning and online clustering[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 20174-20185.

[76] ANKERST M, BREUNIG M M, KRIEGEL H P, et al. OPTICS: ordering points to identify the clustering structure[J]. Acm Sigmod Record, 1999, 28(2): 49-60.

[77] HINNEBURG A, KEIM D A. An efficient approach to clustering in large multimedia databases with noise[C]//International Conference on Knowledge Discovery and Data Mining. 1998: 58-65.

[78] RODRIGUEZ A, LAIO A. Machine Learning. Clustering by fast search and find of density peaks[J]. Science, 2014, 344(6191): 1492.

[79] FRED A L N, JAIN A K. Robust Data Clustering[C]//Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. 2003: II-128-II-133 vol.2.

[80] HUBERT L, ARABIE P. Comparing partitions[J]. Journal of Classification, 1985, 2(1): 193-218.

[81] MA J, SAUL L K, SAVAGE S, et al. Identifying suspicious URLs: an

application of large-scale online learning[C]//International Conference on Machine Learning. 2009: 681-688.

[82] GROUP K E R. London Air Quality Network :: Welcome to the London Air Quality Network Data[Z]. http://www.londonair.org.uk/london/asp/datadownload.asp. 2015.

[83] XIAO T, ZHANG J, YANG K, et al. Error-driven incremental learning in deep convolutional neural network for large-scale image classification[C]//Proceedings of the 22nd ACM international conference on Multimedia. 2014: 177-186.

[84] ZHOU Z H, CHEN Z Q. Hybrid decision tree[J]. Knowledge-based systems, 2002, 15(8): 515-528.

[85] MCCLOSKEY M, COHEN N J. Catastrophic interference in connectionist networks: The sequential learning problem[G]//Psychology of learning and motivation: vol. 24. Elsevier, 1989: 109-165.

[86] KIRKPATRICK J, PASCANU R, RABINOWITZ N, et al. Overcoming catastrophic forgetting in neural networks[J]. Proceedings of the national academy of sciences, 2017: 201611835.

[87] ROBINS A, MCCALLUM S. Catastrophic forgetting and the pseudorehearsal solution in Hopfield-type networks[J]. Connection Science, 1998, 10(2): 121-135.

[88] CASTRO F M, MARINJIMENEZ M J, GUIL N, et al. End-to-End Incremental Learning[J]. european conference on computer vision, 2018: 241-257.

[89] REBUFFI S, KOLESNIKOV A, SPERL G, et al. iCaRL: Incremental Classifier and Representation Learning[J]. computer vision and pattern recognition, 2017: 5533-5542.

[90] DHAR P, SINGH R V, PENG K C, et al. Learning without memorizing[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern

Recognition. 2019: 5138-5146.

[91] ZHANG J, ZHANG J, GHOSH S, et al. Class-incremental learning via deep model consolidation[C]//The IEEE Winter Conference on Applications of Computer Vision. 2020: 1131-1140.

[92] MU X, ZHU F, DU J, et al. Streaming Classification with Emerging New Class by Class Matrix Sketching.[C]//AAAI. 2017: 2373-2379.

[93] HAYES T L, KAFLE K, SHRESTHA R, et al. REMIND Your Neural Network to Prevent Catastrophic Forgetting[C]//European Conference on Computer Vision. 2020: 466-483.

[94] SHIN H, LEE J K, KIM J, et al. Continual Learning with Deep Generative Replay[J]. neural information processing systems, 2017: 2990-2999.

[95] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial nets[C]//Advances in neural information processing systems. 2014: 2672-2680.

[96] WU Y, CHEN Y, WANG L, et al. Incremental Classifier Learning with Generative Adversarial Networks.[J]. arXiv: Computer Vision and Pattern Recognition, 2018.

[97] Van de VEN G M, TOLIAS A S. Generative replay with feedback connections as a general strategy for continual learning[J]. arXiv preprint arXiv:1809.10635, 2018.

[98] KINGMA D P, WELLING M. Auto-Encoding Variational Bayes[J]. international conference on learning representations, 2014.

[99] HINTON G E, VINYALS O, DEAN J. Distilling the Knowledge in a Neural Network[J]. arXiv: Machine Learning, 2015.

[100] LIU X, WU C, MENTA M, et al. Generative Feature Replay For Class-Incremental Learning[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020: 226-227.

[101] ZENKE F, POOLE B, GANGULI S. Continual Learning Through Synaptic Intelligence[J]. international conference on machine learning, 2017, 8: 3987-3995.

[102] LI Z, HOIEM D. Learning without Forgetting[J]. european conference on computer vision, 2017: 614-629.

[103] CHAUDHRY A, DOKANIA P K, AJANTHAN T, et al. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence[J]. arXiv:1801.10112, 2018.

[104] FARQUHAR S, GAL Y. Towards Robust Evaluations of Continual Learning[J]. arXiv:1805.09733, 2018.

[105] SELVARAJU R R, COGSWELL M, DAS A, et al. Grad-cam: Visual explanations from deep networks via gradient-based localization[C]// Proceedings of the IEEE international conference on computer vision. 2017: 618-626.

[106] YOON J, YANG E, LEE J, et al. Lifelong Learning with Dynamically Expandable Networks[C/OL]//International Conference on Learning Representations. 2018. https://openreview.net/forum?id=Sk7KsfW0-.

[107] ZHOU D W, WANG Q W, YE H J, et al. A Model or 603 Exemplars: Towards Memory-Efficient Class-Incremental Learning[C/OL]//The Eleventh International Conference on Learning Representations. 2023. ht tps://openreview.net/forum?id=S07feAlQHgM.

[108] DOUILLARD A, RAMÉ A, COUAIRON G, et al. Dytox: Transformers for continual learning with dynamic token expansion[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 9285-9295.

[109] DOSOVITSKIY A, BEYER L, KOLESNIKOV A, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale[C/OL] //International Conference on Learning Representations. 2021. https://o penreview.net/forum?id=YicbFdNTTy.

[110] XIE J, YAN S, HE X. General incremental learning with domain-aware categorical representations[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 14351-14360.

[111] BANG J, KIM H, YOO Y, et al. Rainbow memory: Continual learning with a memory of diverse samples[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 8218-8227.

[112] BANG J, KOH H, PARK S, et al. Online continual learning on a contaminated data stream with blurry task boundaries[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 9275-9284.

[113] LIN T Y, GOYAL P, GIRSHICK R, et al. Focal Loss for Dense Object Detection[C]//2017 IEEE International Conference on Computer Vision (ICCV). 2017: 2999-3007.

[114] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[115] KRIZHEVSKY A, HINTON G. Learning multiple layers of features from tiny images[R]. Citeseer, 2009.

[116] YAO L, MILLER J. Tiny imagenet classification with convolutional neural networks[J]. CS 231N, 2015.

[117] HENDRYCKS D, GIMPEL K. A baseline for detecting misclassified and out-of-distribution examples in neural networks[J]. arXiv preprint arXiv:1610.02136, 2016.

[118] VAZE S, HAN K, VEDALDI A, et al. Open-Set Recognition: A Good Closed-Set Classifier is All You Need[C]//International Conference on Learning Representations. 2021.

[119] KHOSLA P, TETERWAK P, WANG C, et al. Supervised contrastive learning[J]. Advances in Neural Information Processing Systems, 2020, 33:

18661-18673.

[120] BENDALE A, BOULT T E. Towards open set deep networks[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 1563-1572.

[121] CHEN G, QIAO L, SHI Y, et al. Learning Open Set Network with Discriminative Reciprocal Points[C]//VEDALDI A, BISCHOF H, BROX T, et al. Computer Vision – ECCV 2020. Cham: Springer International Publishing, 2020: 507-522.

[122] ZHOU D W, YE H J, ZHAN D C. Learning Placeholders for Open-Set Recognition[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 4401-4410.

[123] GUO Y, CAMPORESE G, YANG W, et al. Conditional Variational Capsule Network for Open Set Recognition[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 103-111.

[124] GOODFELLOW I J, POUGET-ABADIE J, MIRZA M, et al. Generative Adversarial Nets[C]//International Conference on Neural Information Processing Systems. 2014.

[125] GE Z, DEMYANOV S, CHEN Z, et al. Generative openmax for multi-class open set classification[J]. arXiv preprint arXiv:1707.07418, 2017.

[126] NEAL L, OLSON M, FERN X, et al. Open set learning with counterfactual images[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 613-628.

[127] KONG S, RAMANAN D. OpenGAN: Open-Set Recognition via Open Data Generation[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2021: 813-822.

[128] YOSHIHASHI R, SHAO W, KAWAKAMI R, et al. Classification-Reconstruction Learning for Open-Set Recognition[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition

(CVPR). 2019.

[129] PERERA P, MORARIU V I, JAIN R, et al. Generative-discriminative feature representations for open-set recognition[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 11814-11823.

[130] VAN DEN OORD A, LI Y, VINYALS O. Representation learning with contrastive predictive coding[J]. arXiv e-prints, 2018: arXiv-1807.

[131] HE K, FAN H, WU Y, et al. Momentum contrast for unsupervised visual representation learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 9729-9738.

[132] CHEN T, KORNBLITH S, NOROUZI M, et al. A simple framework for contrastive learning of visual representations[C]//International conference on machine learning. 2020: 1597-1607.

[133] CHEN X, HE K. Exploring simple siamese representation learning[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 15750-15758.

[134] GRILL J B, STRUB F, ALTCHÉ F, et al. Bootstrap your own latent-a new approach to self-supervised learning[J]. Advances in Neural Information Processing Systems, 2020, 33: 21271-21284.

[135] WANG T, ISOLA P. Understanding contrastive representation learning through alignment and uniformity on the hypersphere[C]//International Conference on Machine Learning. 2020: 9929-9939.

[136] WANG Y, ZHANG Q, WANG Y, et al. Chaos is a Ladder: A New Theoretical Understanding of Contrastive Learning via Augmentation Overlap [C/OL]//International Conference on Learning Representations. 2022. ht tps://openreview.net/forum?id=ECvgmYVyeUz.

[137] ZIMMERMANN R S, SHARMA Y, SCHNEIDER S, et al. Contrastive learning inverts the data generating process[C]//International Conference

on Machine Learning. 2021: 12979-12990.

[138] CUBUK E D, ZOPH B, SHLENS J, et al. Randaugment: Practical automated data augmentation with a reduced search space[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020: 702-703.

[139] ZHANG H, CISSE M, DAUPHIN Y N, et al. mixup: Beyond empirical risk minimization[J]. arXiv preprint arXiv:1710.09412, 2017.

[140] YEH C H, HONG C Y, HSU Y C, et al. Decoupled contrastive learning [C]//European Conference on Computer Vision. 2022: 668-684.

[141] CHEN G, PENG P, WANG X, et al. Adversarial Reciprocal Points Learning for Open Set Recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021: 1-1. DOI: 10.1109/TPAMI.2021.3106743.

[142] OZA P, PATEL V M. C2AE: Class Conditioned Auto-Encoder for Open-Set Recognition[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019.

[143] LAKE B M, SALAKHUTDINOV R, TENENBAUM J B. Human-level concept learning through probabilistic program induction[J]. Science, 2015, 350(6266): 1332-1338.

[144] YU F, ZHANG Y, SONG S, et al. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop[J]. arXiv preprint arXiv:1506.03365, 2015.

[145] DIETTERICH T G, GUYER A. The Familiarity Hypothesis: Explaining the Behavior of Deep Open Set Methods[J]. arXiv preprint arXiv:2203.02486, 2022.

致　　谢

在本论文的完成过程中，我得到了许多人的帮助和支持，借此机会，我向他们表达我的衷心感谢。

首先，我要感谢我的导师申富饶教授，他不仅在学术上给予了我指导和启发，还在生活上给予了我关怀和鼓励。他严谨的治学态度、广博的知识面、精湛的研究技能、敏锐的洞察力和热情的教学风格，都深深地影响了我，使我受益匪浅。在本论文的选题、设计、实验、分析和撰写等各个环节，他都给予了我细致的指导和建议，让我在思考问题、解决问题和表达问题上有了很大的进步。在本论文的完成过程中，他多次与我沟通和交流，耐心地解答我的疑惑和困惑，鼓励我克服困难和挑战，激发我创新和探索。在此，我向他表示最崇高的敬意和最诚挚的感谢。

其次，我要感谢南京大学计算机系 RINC 课题组的同学和朋友们，他们在我遇到困难时给予了我帮助和支持，在我忙碌和压力时给予了我陪伴和安慰，在我成功和快乐时给予了我分享和祝福。他们是我学习生活中不可或缺的伙伴，也是我人生道路上宝贵的财富。在本论文的完成过程中，他们给予了我很多的帮助和建议，让我在学习上有了更多的交流和合作，在生活上有了更多的乐趣和温暖。在此，我向他们表示最真挚的友情和最深切的感谢。

再次，我要感谢我的父母和家人们，他们是我最坚强的后盾，无论我身处何方，他们总是给予我无私的爱和无条件的信任。他们为我的成长付出了无数的辛劳和牺牲，他们为我的未来寄予了无限的期望和祝福。他们是我最亲爱的人，也是我最感激的人。在本论文的完成过程中，他们给予了我很多的关心和支持，让我在心理上有了更多的安定和力量，在物质上有了更多的保障和便利。在此，我向他们表示最深情的爱意和最衷心的感谢。

最后，我要感谢我的母校南京大学，它为我提供了优良的学习环境和丰富的校园资源，让我在这里度过了难忘的时光。在这里，我不仅学到了知识和技能，还培养了品德和气质。在这里，我不仅结识了老师和同学，还拓展了视野和

思维。在这里，我不仅收获了成绩和荣誉，还实现了梦想和目标。在这里，我感受到了南大的文化和精神，体验到了百年沉淀的历史和传承。在此，我向南大表示最深厚的敬意和最无限的眷恋。

在本论文的完成过程中，我也深刻地感受到了自己的不足和欠缺，以及本论文的局限和缺陷。我愿意虚心地接受各位评审老师和专家的批评和指正，以便于我在今后的学习和工作中不断地改进和提高。