学校代码：10284
分 类 号：TP181
密　　级：公开
U D C：004.8
学　　号：DZ1733025

南京大学

博 士 学 位 论 文

论 文 题 目　基于重构误差的原型学习算法研究

作 者 姓 名　张天玥

专 业 名 称　计算机科学与技术

研 究 方 向　机器学习

导 师 姓 名　申富饶 教授

2024 年 5 月 26 日

答辩委员会主席 杨明 教授

评　　阅　　人 盲审

论文答辩日期　2024 年 5 月 23 日

研究生签名：

导师签名：

Prototype Learning Methods based on Reconstruction Error

by

Tianyue Zhang

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

May 26, 2024

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：基于重构误差的原型学习算法研究

计算机科学与技术 专业 2017 级博士生姓名：张天玥

指导教师（姓名、职称）：申富饶 教授

摘　要

当前分类/聚类研究通常针对独立同分布的数据进行，同时在部署后用于输出二值化的标签预测结果。而现实世界中的数据通常是动态的、复杂的，数据的分布会因为新类型数据的到来而产生变化，同时单个二值化标签也很难完整地描述输入数据。因此为更好地处理实际应用场景中的数据，对于非独立同分布的增量学习环境，以及将预测标签扩展为实数标签的模糊聚类问题、标签分布问题则仍亟待研究。原型学习作为与人类认知相近的，通过学习代表原型来表示数据分布的学习框架已经在机器学习领域有多年的研究历史。由于原型集合增加或者删减的灵活性，以及原型模型拥有通用的表征学习能力，能够用于构建处理模糊数据以及在增量学习环境下进行学习。因此，本文从重构误差这一角度出发，将该损失函数作为原型学习框架的基础训练目标，并研究如何根据提出的原型学习框架形成对模糊聚类、标签分布、数据增量问题的学习算法。本文取得的成果包括：

1. **基于重构误差的原型学习框架。**在无其他任务信息的帮助下，如何学习合适的原型对数据分布进行表示是原型学习框架的基础。本文使用重构误差作为基本原型学习框架的训练目标，从而衡量原型集合对产生数据的基本模式的学习情况，能够促进原型集合对数据中关键信息的掌握。在此基础上，本文对比四个以重构误差为基础的在线线性原型学习方法并在多个领域的数据集上进行实验评估，得到结论为该学习框架能够学习到通用的表征表示，但面对具体任务时仍需要根据具体任务对模型进行相应的改进。
2. **基于模糊原型竞争学习框架的在线增量模糊聚类算法。**考虑到数据分析中出现数据分布中位于交叠区域的数据难以进行聚类，以及物理意义上输入

特征对不同的概念具有不同的符合程度，模糊聚类任务要求模型对输入特征预测其对于所有数据簇的归属程度。此外，聚类模型也常常面临需要在线增量进行学习的情况。本文提出基于重构误差的模糊原型竞争学习框架，并据此形成模糊自组织增量神经网络算法用于解决在线增量模糊聚类问题。该算法能够学习拓扑连接的原型图来对数据分布进行表示，并根据输入特征与原型的归属程度来确定其对于不同簇的归属程度。在学习过程中，网络随着数据分布的变化而增加或删除神经元的功能使其能够在增量学习环境中取得可塑性-稳定性的平衡。定性与定量实验验证了该模糊聚类算法在模糊聚类与硬聚类问题上的表现。

3. **基于非负原型线性学习框架的标签分布学习算法。**在有标签的监督学习中，单个标签可能不足以描述输入数据，同时多个标签的符合程度又有所区别，因此研究者们提出标签分布学习任务用于解决使用多个标签在不同程度上描述输入数据的问题。本文提出基于重构误差的非负原型线性学习框架，并据此形成非负原型基底学习算法用于解决标签分布问题。该算法使用原型向量来作为标签的支撑向量，从而将输入特征表示为原型向量的线性组合，并使用该线性系数对最大熵模型进行训练从而得到对输入特征在标签集合上的标签分布预测。多个数据集以及指标上进行的实验验证了本文算法在该问题上的分类性能相较以往算法有所提升。
4. **基于单原型学习框架的数据增量学习算法。**深度神经网络相比传统机器学习模型在大型数据集上具有优势，但在类增量学习环境下体现出灾难性遗忘的特性，从而难以进行多个任务的连续学习。数据增量问题使用增量的数据批训练网络，更加为神经网络对抗遗忘与更新参数带来了挑战。本文采用基于重构误差的单原型学习框架，提出基于该原型学习框架的原型重构误差用于训练网络，并据此形成监督对比与原型重构学习算法用于解决数据增量问题。该算法将原型重构与对比学习损失函数结合用于网络训练，并使用网络提取的嵌入特征更新原型分类器，达到了原型分类器与神经网络共同进行训练的结果。除此之外，网络还使用了回放样本内存对抗网络的遗忘问题。在平衡与非平衡的多个数据集上进行的分类性能对比，验证了本文算法在数据增量环境中的有效性。

关键词：原型学习；增量学习；标签分布学习；模糊聚类

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Prototype Learning Methods based on Reconstruction Error

SPECIALIZATION: Computer Science and Technology

POSTGRADUATE: Tianyue Zhang

MENTOR: Professor Furao Shen

ABSTRACT

Current classification/clustering research is usually carried out for independent and identically distributed data, and it is also used to output one-hot label prediction results after deployment. However, the data in the real world is usually dynamic and complex, and the distribution of data will change with the arrival of new types of data. At the same time, it is difficult for a one-hot label to describe the input data completely. Therefore, in order to better deal with the data in practical application scenarios, it is still urgent to study the non-independent and identically distributed incremental learning environment, as well as the Fuzzy Clustering problem and Label Distribution problem of expanding prediction labels into real-numbered labels. Prototype learning, as a learning framework similar to human cognition, has been studied in the field of machine learning for many years. Because of the flexibility of adding or deleting prototype sets and the universal representation learning ability of prototype models, it can be used to construct and process fuzzy data and learn in incremental learning environment. Therefore, from the perspective of reconstruction error, this thesis takes this loss function as the basic training goal of prototype learning framework, and studies how to solve three specific problems of Fuzzy Clustering, Label Distribution learning and Data Incremental learning by combining the prototype frameworks. The achievements of this thesis include:

1. **Prototype Learning Framework based on Reconstruction Error.** Without the help of other task information, how to learn appropriate prototypes to represent data distribution is the basis of prototype learning framework. In this thesis, the

reconstruction error is used as the training goal of the basic prototype learning framework, so as to measure the learning situation of the prototype set on the basic pattern of generating data and promote the prototype set to grasp the key information in the data. Besides, this thesis compares four online linear prototype learning methods based on reconstruction errors and makes experimental evaluation on data sets in many fields. The conclusion is that the learning framework can learn universal representations, but the model still needs to be improved according to specific tasks.

2. **Online Incremental Fuzzy Clustering Algorithm based on Fuzzy Prototype Competitive Learning Framework.** Considering that it is difficult to cluster the data located in the overlapping area in data distribution in data analysis, and the input features have different degrees of conformity to different concepts in physical sense, the Fuzzy Clustering task requires the model to predict the membership of the input features to all data clusters. In addition, clustering models are often faced with the need for online incremental learning. In this thesis, a competitive learning framework of fuzzy prototype based on reconstruction error is proposed, and a Fuzzy Self-Organizing Incremental Neural Network algorithm is formed to solve the Online Incremental Fuzzy Clustering problem. The algorithm can learn the prototype diagram of topological connection to represent the data distribution, and determine the membership degree of input features to different clusters according to the membership degree of prototypes. In the process of learning, the network adds or deletes the function of neurons with the change of data distribution, so that it can achieve a plasticity-stability balance in the incremental learning environment. Qualitative and quantitative experiments verify the performance of the proposed algorithm on Fuzzy Clustering and Hard Clustering problems.
3. **Label Distribution Learning Algorithm based on Non-negative Prototype Linear Learning Framework.** In supervised learning, a single label may not be enough to describe the input data, and the coincidence degrees of multiple labels are different, thus researchers put forward the Label Distribution learning task to solve the problem of using multiple labels to describe the input data in different

degrees. In this thesis, a non-negative prototype linear learning framework based on reconstruction error is proposed, and a Non-negative Prototype Basis Learning algorithm is formed to solve the Label Distribution problem. In this algorithm, the prototype vectors are used as the support vector of labels, so that the input features are represented as a linear combination of prototype vectors, and the maximum entropy model is trained by using this linear coefficient, so that label distribution predictions of input features are obtained. Experiments on several data sets and indicators demonstrate that the classification performance of this algorithm is improved compared with previous algorithms.

4. **Data Incremental Learning Algorithm based on Single Prototype Learning Framework.** Compared with the traditional machine learning model, deep neural networks have advantages on large data sets, but it shows the characteristics of catastrophic forgetting in the incremental learning environment, which makes it difficult to learn multiple tasks continuously. The Data Incremental problem uses streaming data batches to train the network, which brings more challenges to the neural network against forgetting and updating parameters. In this thesis, a single prototype learning framework based on reconstruction error is adopted, and the prototype reconstruction error based on this prototype learning framework is proposed to train the network, thus a Supervised Contrastive and Prototype Reconstruction learning algorithm is formed to solve the Data Incremental problem. In this algorithm, prototype reconstruction and supervised contrastive learning loss function are combined for network training, and the prototype classifier is updated with embedded features extracted from the network, so that the prototype classifier and neural network can be trained together. In addition, the network also uses replay memory to counter the forgetting problem of the network. The comparisons of classification performance on balanced and unbalanced data sets verify the effectiveness of the proposed algorithm in the incremental environment.

KEYWORDS: Prototype learning; Incremental learning; Label distribution learning; fuzzy clustering

目　录

第一章　绪论

1.1　引言

机器学习是一门致力于通过已有的数据提炼出数据模式，从而进行预测、识别等任务的学科，机器学习算法和模型的学习能力可以概括为“抽象”的能力：即从大量数据中发现概率与规律，用于任务的决策。如何构建有效的算法模型能够对输入数据的规律进行总结，是机器学习研究的重点。分类与聚类任务，作为最为广泛研究的研究目标，分别探究在有监督与无监督的条件下如何将数据归纳到已知的模式中，而这种归纳方式在机器学习模型上体现为对输入数据的标签进行预测。为提升机器学习模型的可靠性，预测准确性的提升是机器学习领域重点追求的目标。然而，现实世界是动态的、模糊的，单个标签通常不足以描述实体对象的全部性质，并且随着时间的推移数据分布逐渐会产生变化。传统机器学习模型则通常采用一维整数标签对数据进行描述，并且在独立同分布的数据上训练完成后即进行部署，无法继续训练学习新的数据分布。尽管机器学习模型已经在众多研究领域上取得了十足的进展，在更加贴合人类学习情况的训练环境中的算法表现仍然亟待提升。

在寻求解决问题方案时，人类的学习能力一直是算法领域参考的重点，因为人类具有优秀的凝炼知识、提炼概念的能力，可以在大脑中将信息总结为抽象的概念。神经科学、认知科学等多个领域的研究人员认为，该学习能力可由两种方式实现：包括通过原型（prototype）集合表示知识及通过代表样例（exemplar）学习知识。代表样例学习是指通过记录同一类别事物的典型样例来表示某一概念，这些样例是实际存在的；而原型学习则是指将样例分布的中心趋势总结为抽象的原型样例，这些原型样例是由具体样例混合形成的，并非实际存在。这两种学习共同存在于人类的大脑中。且可以通过脑成像、行为实验等方式证实[1]。例如，当提到“猫”这个类别时，人们也会记住几只让人印象深刻的猫咪，这则是猫这个概念中所保存的代表样例；同时，人们脑海中可能会浮现多个体现不同类型的

猫特征，但又并非现实中存在的抽象样本，这是大脑所提炼生成的概念原型。对于其他实体，人们将其与脑海中已有的猫的原型与代表样例进行比对，则能够识别出该实体是否属于猫这个概念。在机器学习领域，上述使用学习到的抽象原型概括数据分布信息的原型学习（Prototype Learning）算法从不同的研究方向持续受到研究者们的关注[2][3][4]。该类算法使用通常少于输入数据数量的原型向量集合来描述输入数据分布，该过程可以认为是将输入向量由其所在的特征空间转换到由原型所组成的嵌入空间进行表示，从而达到对数据中重要信息进行特征提取的结果。此时，原型集合的组合可对输入向量进行重构表示，从而将输入向量表示为系数向量。原型学习算法已经被国内外研究者应用于多种实际应用中，其大部分应用为分类、聚类等需要进行概念总结的场景中，例如人物搜寻[5]、目标检测[6]、视频分类[7]、图像分割[8][9][10]、文本分类[11] 等，对原型学习算法的综述也已有发表[12][13]。尽管原型学习相关研究已有多年历史，但是研究者们通常从不同的问题与思路出发提出自己的原型学习算法。是否能够总结出通用的原型学习框架，并在此基础上来提出不同的问题的解决方案，是本文研究的重点。

对原型学习框架进行研究，首要出发点是如何学习到好的原型集合。考虑到该框架的通用性，即在不同任务目标的情况下也能够判定原型的质量，通过输入特征来自监督检查原型集合是否保留了关键信息是一种可行的方案。如同可以通过学生是否能够为别人讲解知识点来判断该学生对知识点的掌握情况，判断原型集合对输入数据进行还原的能力也可以判断原型集合的质量。使用原型集合及权重系数对原始数据进行还原的过程被称为重构（reconstruction），而重构数据与原始数据之间的差别也被称为重构误差（Reconstruction Error）。重构误差越小，则意味着模型对原始数据的还原能力更好。因此，本文采用重构误差作为原型学习框架的基本学习目标。

根据进行数据重构时是否通过原型向量线性变换而得到，原型学习框架可以被分为两类：一类将原型向量视为重构时的基底，使用原型向量的线性组合来逼近输入向量，这种也被称为线性模型；其余类型的原型学习模型均可被归为非线性模型中。以重构误差为目标并加入启发式的规则对原型的性质加以限定，发展出多种无监督原型学习算法，例如限定基底与系数均为正数的非负矩阵分解[4]，采用竞争学习方式每次只对单个原型进行更新的自组织图方法[2]。当有标签信息时，标签作为额外的信息描述了输入数据之间的相似程度，因此在进行原

型学习时，引入类别信息对原型组合时的系数加以限制能够提升在该学习环境下原型表示的质量。例如半监督的非负矩阵分解会在基底矩阵和系数矩阵的计算时引入标签矩阵[14]；学习矢量量化算法则会使得相同类别的原型更易被输入数据激活，而非相同标签的原型向量则降低与该输入向量的相似程度[15]。因此，以重构误差为基础的通用型原型学习框架既可以采用自监督的方式进行表征学习，又可以在引入标签信息的情况下学习到具有任务特性的原型。重构原型学习算法的作用可以从两个方面进行概括，即通过原型对聚类/分类的数据分布进行表示，从而便于进行模型对于标签的预测；或是将输入特征转换为系数向量，从而提高模型的任务性能。

灵活性也是原型学习算法的一大优点，即当输入数据分布发生变化时，原型学习算法可以通过调整原型集合来适应新的数据分布。传统机器学习模型常常采用训练-部署的模式，即一旦训练完成进行实际使用后，就不再对模型进行更新。但现实世界通常是时刻变化的，新类型的数据时有产生。能够学习该类新出现的数据，并且尽量维持在已学习数据上的表现的模型被称为增量学习模型。无论是线性[16][17] 还是非线性[18][19][20] 的模型，在线增量学习的原型学习模型都有着大量的相关研究。以重构误差为基础框架的原型学习算法可以通过修改、增加甚至删除原型的方式来降低新数据所带来的重构误差提升，尝试满足在学习过程中变化的任务目标。

随着硬件与互联网的发展，用于解决具体任务所搜集、存储的数据集规模越来越大。同时，该类数据集的信息也逐渐完善，例如 ImageNet[21] 这样包含所有数据标注的大型数据集用于模型训练。这既为机器学习模型带来了提升模型质量的机遇，同时也提升了模型处理数据所需要的参数量以及训练量。随着对激活函数、损失函数、网络结构等一系列模块的改进，深度神经网络逐渐成为具有极高扩展性的学习模型，因此被研究者们视为对大数据进行表征学习的最佳工具，即能够从高维数据中提取具有区分性的特征。虽然深度神经网络在 “训练-部署” 这类批训练环境中表现十分良好，但其仍然在增量学习类环境中遭受了遗忘问题，使得最终网络性能急剧下降[22]。研究者们指出，线性分类层在增量学习环境中遗忘明显[23][24]。考虑到原型学习与深度神经网络正好拥有互补的优势与劣势，因此结合这两种学习方式进行共同学习成为解决复杂环境下大型数据集处理的研究趋势。相较于深度神经网络-线性分类层这一结构，原型向量由深度神

经网络所提取的特征直接计算得到。当特征提取网络的特征分布产生变化后，则计算得到的原型也相应进行进行变化，无需再进行训练。尽管大模型能够带来的性能提升已经足够惊艳，如何建立一个更加灵活，能够适应复杂环境变化的模型仍是许多研究者都在探索的方向，而原型学习算法与特征提取算法等更多机器学习算法模块的结合将是一条可以探索的道路。

考虑到重构原型学习框架本身具有通用性、灵活性、可扩展性强的优点，本文开展了对以重构误差为基础的原型学习框架的研究，并以此为基础延伸出以此原型学习框架解决包含实数化标签预测以及增量学习研究方向在内的三个原型学习算法。本文对具体算法目标的研究均根据如下思路进行展开，即首先通过分析具体问题所需要的模型能力而对基础的学习框架进行补充与改动，其次补充具体算法所需要的其他模块，最终验证模型在对应问题中的预测能力。

1.2 国内外研究现状

对原型学习算法进行综述前，首先需要对其所解决的机器学习任务进行定义。本文着重关注使用模型对输入特征进行标签预测的任务，即通过机器或人工定义的标签集合对输入特征进行标记，该问题包含了有监督的分类问题、无监督的聚类问题以及它们的扩展问题。其次，本节通过对原型重构时采用的系数形式将原型学习算法分为线性算法与非线性算法，并分别对国内外相关工作进行阐述。

1.2.1 形式化标签预测问题

模型在训练时获得训练数据集 $\mathcal{D}_{train} = \{\mathcal{X}_{train}, \mathcal{Y}_{train}\}$ 中获得 N 个训练样本对的集合 $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_N$ 或从数据流 $\mathcal{S}_{train}$ 中持续不断的采样训练样本对 $(\boldsymbol{x}_t, \boldsymbol{y}_t)$。其中，$\boldsymbol{x} \in \mathbb{R}^d$ 表示 d 维数据，$\boldsymbol{y} \in \mathbb{R}^C$ 表示对于 C 个类别形成的类别标签。在批量训练的情况下，N 个输入数据可整体表示为输入特征矩阵 $\mathbf{X}_{train} \in \mathbb{R}^{N \times d}$，以及输入标签矩阵 $\mathbf{Y}_{train} \in \mathbb{R}^{N \times C}$。当输入数据为无标签数据时，对应标签的所有维度置为 0。单分类问题中采用独热（one-hot）编码形式，标签向量只有一位置为 1，其余置为 0。在标签预测的实数化扩展研究中[25] 中，标签通过 C 维实数向量进行表示。标签预测任务可定义为如下形式：

定义 1.1 (标签预测任务) 标签预测任务可记为元组 $\langle \mathbb{X}, \mathbb{Y}, \theta, P \rangle$，其中 $\mathbb{X}$ 为输入特征空间，$\mathbb{Y}$ 为标签向量空间，$\theta : \mathbb{X} \to \mathbb{Y}$ 为模型预测函数，$P : \mathbb{Y} \times \mathbb{Y} \to \mathbb{R}$ 为任务评价函数。

在后续的原型学习算法中，本文使用 $\boldsymbol{p}_j$ 表示原型集合中的第 j 个原型，K 表示原型集合的大小，v_i^j 表示第 j 个原型对第 i 个输入特征进行重构时所使用的系数。

1.2.2 线性原型学习算法

主成分分析（Principal Component Analysis，PCA）是经典线性降维算法，其想法可以追溯到 1901 年[26]，并在后续研究中被命名为 PCA 算法[3]。其主要思路为：将数据映射到嵌入空间后，使得低维嵌入之间的区分度较大，即增大嵌入特征数据的方差总和，该优化问题通过引入拉格朗日算子进行优化可以得到数据的主成分方向为输入向量的协方差矩阵的特征向量，即将输入向量降到 K 维后，输入向量可以由协方差矩阵的最大 K 个特征值对应的特征向量进行线性重构。该问题也可以从原型线性重构的方向去理解[27]，即将输入向量通过 K 个基底向量线性组合，以及 $d-K$ 个基底向量采用在整个数据集上共享的系数进行表示，并最小化该表示与原始输入向量之间的重构误差。算法目标为最小化原型在所有输入特征上的重构误差：

$$\min_{\mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \left\| \boldsymbol{x}_i - \left(\sum_{j=1}^{K} v_i^j \boldsymbol{p}_j + \sum_{j=K+1}^{d} z_j \boldsymbol{p}_j \right) \right\|_2^2 . \tag{1-1}$$

对上式求偏导后再进行拉格朗日算法求解，同样可以得到前 K 个原型为输入向量协方差矩阵的特征向量。主成分分析作为机器学习的代表性算法，在其基础上拥有多种改进方法，例如将系数系数化使得可解释性与高维数据表现更好的稀疏主成分分析（Sparse Principal Component Analysis，SPCA）算法[28][29][30]，能够在在线增量环境下进行学习的增量主成分分析算法（Incremental Principal Component Analysis，IPCA）算法[16][31] 等。在人脸识别[32]、图像压缩[33]、色彩处理[34]、基因数据处理[35] 等不同领域的应用中，PCA 以及改进算法被大量应用并获得了良好的效果。除此之外，有多篇文献对 PCA 算法的改进与应用进行综

述[36][37][38]。

非负矩阵分解（Non-negative Matrix Factorization，NMF）[4] 则主要应对非负数据进行，并在重构误差的基础上加入了非负性的限制，即要求所有基底向量与系数向量都非负。非负性的限制更多考虑了对于模型可解释性的需求，例如对于图像数据，分解出来的基底非负则可以看做是基底图像，而非负系数则对应着基底图像的叠加构成了输入图像。NMF 本身仍然是从重构误差角度出发学习基底向量，并在此基础上加入了基底向量与系数向量的非负性限制。NMF 通常是将所有数据整合成数据矩阵来进行考虑，将非负输入数据矩阵标记为 $\mathbf{X}_+$，则基底矩阵 $\mathbf{P}$ 与系数矩阵 $\mathbf{V}$ 可通过如下学习目标优化得到

$$\min_{\mathbf{P}} \|\mathbf{X}_+ - \mathbf{V}\mathbf{P}\|_F, \quad \text{s.t.}\, \mathbf{V} \geqslant 0, \mathbf{P} \geqslant 0. \tag{1-2}$$

其中 $\|\cdot\|_F$ 表示矩阵的 Frobenius 范数。由于当两个矩阵都是可变参数的情况下，该目标函数是非凸的，因此上述式子可以通过交替优化或引入辅助函数的方式进行优化[4]。非负矩阵分解的改进通常通过引入不同的约束或正则化项[39][40]，或是引入额外的权重矩阵以提升矩阵分解的效果[41]。已有综述论文[42][43] 更为深入地介绍改进的矩阵分解算法。

1.2.3 无监督非线性原型学习算法

当输入数据无法通过原型的线性组合来进行重构时，即系数向量无法组成线性空间时，该类原型学习算法可以统一归结为非线性模型。线性算法引入核技巧使得模型能够处理非线性可分数据，以计算效率为代价提升算法性能，例如核 PCA 算法[44]、核 NMF 算法[45] 等；除此之外，也有将线性系数重构改进为局部线性重构的局部线性 PCA 算法[46]。这部分章节将重点介绍提出时即采用非线性原型重构的无监督算法。

非线性无监督模型常采用局部线性重构的形式，该学习形式类似于人类的神经元激活，即每次输入特征只能够激活足够相似的原型特征。胜者通吃（Winner-Taskes-All，WTA），即每次只激活最相似的单个原型的学习规则被用于多种非线性原型学习方案中，同时该种学习方式也被称为竞争学习（Competitive Learning，CL）。该规则来源于对于生物神经元激活行为的研究，并广泛应用于原型学习相

关的算法中。其规定每次输入特征只能够激活与其相似程度最高的原型特征，其余特征都不会据此进行学习。在后续算法的改进中，根据模型训练的需要原型激活个数可以扩展到有限个原型进行共同激活。每个输入向量只激活单个原型时，以欧式距离作为相似度衡量为例，每次输入特征激活距离最小的原型。具体来说，对于输入特征 $\boldsymbol{x}_i$，其激活的原型索引 $j(i)$ 为：

$$j(i) = \arg\min_j \left\| \boldsymbol{x}_t - \boldsymbol{p}_j \right\|_2 . \tag{1-3}$$

此时只有激活的原型重构系数为 1，其余原型的重构系数为 0，则全局的训练目标为最小化在所有输入特征上的重构误差之和：

$$\min_{\mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} \left\| \boldsymbol{x}_i - \boldsymbol{p}_{j(i)} \right\|_2^2 . \tag{1-4}$$

尽管每次学习时只降低了部分原型的重构误差，但可以认为其他未激活的原型重构系数为 0，因此从全局来看仍然降低了全局的重构误差[47]。

当对原型进行在线学习时通常使用梯度下降法进行更新，即对公式（1-4）进行求导并得到更新公式

$$\boldsymbol{p}_{j(i)} = \boldsymbol{p}_{j(i)} + \alpha(\boldsymbol{x}_t - \boldsymbol{p}_{j(i)}). \tag{1-5}$$

为使得得到的原型学习结果收敛到稳定状态，学习率 α 需要选择随着时间减少并逐渐收敛到 0 的参数[48]。如前文所分析的，当每次激活的原型数量有限时，原型会只根据与自己相关的输入特征进行更新。理想状态下，输入特征只更新与其属于同一模式的原型；如果是新数据分布采样的输入样例，则可以通过新增原型避免其对其他数据分布的原型进行修改。需要注意的是，由于新增原型对模型的参数进行了扩张，若模型训练的内存空间有限，则需要考虑为模型的原型数量设定上限，并规定在超过上限时对原型的覆盖规则。

自组织增量神经网络类算法是采用竞争学习原型进行学习的经典算法类型，自组织（self-organizing）一词强调了原型之间的竞争交互与不采用全局损失函数更新的特点。**自组织图（self-organizing map，SOM)**[2] 由一维或二维且包含特征向量的原型网格组成，网格将原型的共同激活关系表示为拓扑连接上的邻居

关系。如果将初始化的原型网格看作是一系列关键点构成的 “平面”，那么后续学习过程则是移动这些关键点使得形变后面的 “平面” 能够描述数据的分布。在二维平面上，第 j 个原型的坐标可记为 $\boldsymbol{c}_j = \left(c_{j1}, c_{j2}\right)$，其邻居则是与该坐标相邻坐标所对应的原型。在 SOM 中相互连接的原型具有共同激活学习的性质，即输入特征在更新获胜原型时邻居原型进行更新。因此，SOM 的原型更新公式在公式 (1-5) 的基础上进行了如下的改动：

$$\boldsymbol{p}_k = \boldsymbol{p}_k + \alpha \cdot h_{SOM}(\left\|\boldsymbol{c}_w - \boldsymbol{c}_{j(i)}\right\|)(\boldsymbol{x}_t - \boldsymbol{p}_w), \quad w \in A(j(i)). \tag{1-6}$$

其中 $A(j(i))$ 包含了获胜原型 $j(i)$ 及其所有邻居原型的索引，邻居原型的学习幅度通过邻接函数 $h_{SOM}(\cdot)$ 来控制幅度，因此该函数通常随着距离的增大而减小。

SOM 开创了对自组织神经网络的研究热潮，研究者们也发现了该算法的一些缺陷，例如 SOM 的节点与网格形状需要提前定义，不能够随着学习而改变，因此研究者们提出了可以动态确定结构和原型数量的算法，例如增长 SOM 算法（growing SOM）[49][50]；另一方面，尽管 SOM 采用拓扑结构试图解决初始化和特征空间的影响，SOM 中仍然可能存在无法被更新的死原型以及被过量更新的原型。在 SOM 的基础上提出的**神经气体（Neural Gas，NG）方法**[51] 则不再使用网格提前排布好原型节点之间的拓扑结构，而是通过原型特征之间的相似度排序来计算邻接关系。每次获取输入特征后，所有原型根据其与输入样例的相似度进行排序，即按照距离从小到大的排序为第 j 个原型特征赋予排序值 r_j。此时，原型通过如下更新公式来进行更新：

$$\boldsymbol{p}_j = \boldsymbol{p}_j + \alpha \cdot h_{NG}(r_j)(\boldsymbol{x}_t - \boldsymbol{p}_j), \quad j \in A(j(i)). \tag{1-7}$$

其中 $h_{NG}(r_j) = \exp(-r_j/\lambda)$ 为 NG 的邻接函数，λ 为控制每个原型学习幅度的参数。当 λ 在学习初期设置得足够大时，就可以使所有原型都进行学习，避免出现无法更新的原型特征。在后续学习中，随着原型学习逐渐收敛，该参数可以逐步缩小以减少无关原型特征产生振荡。当 λ 接近 0 时，将只有获胜原型进行更新。

虽然 NG 解决了原型更新的问题，且无需提前设置原型的拓扑结构，但是原

型集合的数量 K 仍需要提前进行定义。该定义需要根据数据分布的先验知识对原型数量进行估计，但先验知识在无监督环境中难以获取。一种解决方案是采用网格搜索等参数搜索手段在实验中多次尝试选取最好的参数；另一种则是通过在学习过程中增加或减少原型数量以适应数据分布表示的需求，例如增长型神经气体算法（Growing Neural Gas，GNG）[52] 就是可以增加原型数量的神经气体算法。需要注意的是，GNG 的节点增加并没有设置阈值，因此原型数量可以一直增长，对于存储空间有限的设备可能是无法接受的。除去规定原型数量上限之外，也可以通过删除处于低数据密度区域的原型来减少原型数量。在持续对新类别数据进行学习的环境中，该删除方式也可被视为一种最近最少使用（Least Recently Used）缓存机制，即保留最近被更多新任务数据激活的原型，而删除旧任务中较少使用的原型。

同样在训练过程中能够处理新类别数据的**自组织增量神经网络（Self Organinzing Incremental Neural Network，SOINN）**[19] 使用原型之间的拓扑连接状态与原型的激活次数来控制原型的删除，从而达成对原型数量的控制。除此之外，学习过程中则使用激活阈值判断当前原型的学习状态，使得已经足够被已有原型表示的输入特征才能够更新原型，而将其他无法激活至少两个原型的输入特征视为新的数据分布并加入原型集合中。对于获胜原型 $\boldsymbol{p}_w$ 及与其相互连接的原型，SOINN 的原型将通过如下方式进行更新：

$$\begin{aligned} &\boldsymbol{p}_w = \boldsymbol{p}_w + \alpha \cdot h_{SOINN}(w, j(i))(\boldsymbol{x}_t - \boldsymbol{p}_w), \quad w \in A(j(i)). \\ &h_{SOINN}(w, j(i)) = \begin{cases} 1, w = j(i) \\ \frac{1}{100}, w \neq j(i) \end{cases}. \end{aligned} \tag{1-8}$$

该公式表明在 SOINN 算法中，激活原型将进行大幅度的更新；同时认为相互连接的原型具有相似性，因此获胜原型的邻居原型也据此进行更新。原始 SOINN 模型是两层结构，在学习完第一层原型后，还会将学习的原型作为第二层模型的输入进行学习，这使得在线学习的情况下需要随时判断是否应该进入第二层的学习。SOINN 的改进算法通常采用一层结构，更适宜在在线增量环境中使用。增强 SOINN（Enhanced SOINN，ESOINN）[20] 增加了对原型周围输入特征分布密度的判断，使得模型能够更好地将属于不同数据分布的原型相互分离，属于相同数

据分布的原型相互连接，避免两个相近簇因为交叠的输入特征而产生了异簇原型之间的拓扑连接。负载均衡 SOINN（Load Balancing SOINN，LB-SOINN）[53] 则在 ESOINN 的基础上引入了除欧式距离外多种相似度衡量方式，使得该类模型能够应用于更高维数的输入特征空间；同时采用负载均衡的方式平衡每个原型获胜的次数，并使得 ESOINN 的分离和合并能够更加稳定。相比 ESOINN，LB-SOINN 可以获得更为稳定的聚类结果，同时也可以用于文本分类等输入维度更高的应用场景。局部分布 SOINN（Local Distribution SOINN，LD-SOINN）[54][55] 则为原型引入了局部协方差矩阵，将原型视为高斯混合模型进行训练。相较于传统 EM 算法所训练的高斯混合模型，LD-SOINN 的训练速度更快，所需输入数据量更少。SOINN 原型学习算法作为一个基础模块，已经被结合其他技术从而应用于多种应用场景中，例如医疗诊断[56][57]，异常检测[58]，机器人智能[59][60][61] 等。已有论文[62] 对 SOINN 算法及其改进算法进行了深入的列举与探讨。

1.2.4 有监督非线性原型学习算法

在监督学习条件的分类问题中，除可以使用无监督原型学习算法作为任务无关的特征提取模块之外，也可以对原型进行类别标记使其能够完成分类任务。当使用标签信息时，原型的竞争方式则演变成只有与输入特征标签相同的原型可以激活。该激活方式可以使用指示函数 $\mathrm{I}(\boldsymbol{x}_i, \boldsymbol{p}_j)$ 来表示，该函数在 $y_{\boldsymbol{x}_i} = y_{\boldsymbol{p}_j}$ 的时候取值为 1，其余时间取值为 0。

最简单的原型分类算法可以追溯到**最近邻（Nearest Neighbor, NN）算法**[63]。NN 算法在训练过程中只做一件事，那就是将所有的训练样例保存下来。而在测试的时候计算测试样本与所有训练样例之间的相似程度，将该测试样本的类别指定为与它相似程度最大的训练样例。对于其重构误差的分析也显而易见，即该算法由于保留了所有的训练数据，因此对于训练特征的重构误差永远为 0。该算法可以拓展为 k-最近邻算法（k-NN），即考虑距离最近的 k 个邻居的标签，采用投票式或加权投票式确定最终的预测标签。该方法将所有训练样本都视为代表该类别的原型，每个原型都负责其附近的分类边界，因此 k-NN 算法形成的分类边界将十分复杂，同时没有对原始数据集起到降低数据复杂度的效果。尽管 k-NN 是一个简单有效的分类方法，即使在算法复杂度越来越高的今天，其仍然拥有一定的应用场景。理论证实，在 k 选择合适时，k-NN 能够达到与贝叶斯最优

分类算法相近的效果[13]。但是不可否认的是，k-NN 在原始特征空间中进行搜索的代价十分高昂，同时决策边界十分复杂，存在过拟合现象。同时，需要足够大的数据集使样本在特征空间中有足够的密度来支持 k 的选择。该方法在数据集和原始特征空间都较小时能够有效发挥其作用，同时也有一系列对其的改进使其能够在更多场景发挥作用。例如，采用无监督原型学习算法 SOINN、k-means 对输入特征进行预先选择以减少 k-NN 的计算复杂度[64]，或是采用近似搜索的方式减少计算复杂度[65] 等。部分综述论文则更为详细地对比并分析了不同最近邻的改进方法[66][67][68]，国内对于该类算法的研究与应用同样形成了多个论文成果[69][70][71]。

上述无监督的竞争学习模型在监督的环境中也可以通过引入类别信息来进行学习。在提出自组织图的论文[2] 中，Kohonen 同样提出了引入类别标签的原型更新方式，该类算法也被称为**学习矢量量化（Learning Vector Quantization，LVQ）类算法**。当输入特征到来时，模型中同类别的原型应当可能靠近，不同类别的原型应当尽可能远离，从而避免将该输入特征分类为该类别。上述学习方式在论文中作为原型更新公式被提出，即：

$$\begin{aligned} \boldsymbol{p}_{j(i)} &= \boldsymbol{p}_{j(i)} + \alpha(\boldsymbol{x}_t - \boldsymbol{p}_{j(i)}), \\ \boldsymbol{p}_{j(i)^-} &= \boldsymbol{p}_{j(i)^-} + \alpha(\boldsymbol{x}_t - \boldsymbol{p}_{j(i)^-}). \end{aligned} \tag{1-9}$$

其中，$j(i)$ 以及 $j(i)^-$ 分别为同类别中与输入特征欧式距离最小的原型以及不同类别中与输入特征欧式距离最小的原型。然而，这个更新公式无法作为某个损失函数的导数，也无法对应于具体的重构误差函数。因此，研究者们在该算法的基础上进一步对原型学习的框架进行分析，并提出例如泛化 LVQ（Generalized LVQ，GLVQ）[72] 的算法为原型学习策略提供了重构误差的学习目标：

$$\min_{\mathcal{P}} \sum_{i=1}^{m} f\left(\frac{d^+ - d^-}{d^+ + d^-}\right). \tag{1-10}$$

其中，$d^+ = \left\|\boldsymbol{x}_i - \boldsymbol{p}_{j(i)}\right\|_2^2$，$d^- = \left\|\boldsymbol{x}_i - \boldsymbol{p}_{j(i)^-}\right\|_2^2$ 分别表示了输入特征对于同类原型和异类原型的距离度量。该损失函数仍可从重构误差的角度去理解：使用同类输入特征的重构误差减去异类特征的重构误差，此时对于异类特征的重构误差越

大越好。根据上述误差函数，则原型可以根据梯度下降法进行学习，即

$$\begin{aligned}\boldsymbol{p}_{j(i)} &= \boldsymbol{p}_{j(i)} + \alpha f' \frac{d^-}{(d^+ + d^-)^2}(\boldsymbol{x}_t - \boldsymbol{p}_{j(i)}),\\ \boldsymbol{p}_{j(i)^-} &= \boldsymbol{p}_{j(i)^-} - \alpha f' \frac{d^+}{(d^+ + d^-)^2}(\boldsymbol{x}_t - \boldsymbol{p}_{j(i)^-}).\end{aligned} \tag{1-11}$$

其中 $f(\cdot)$ 可以选择 sigmoid 函数。与此相同，更多算法[73][74] 也采用引入启发式规则的方式使得原始 LVQ 的原型更新公式可以通过目标函数而推导得到。

随着数据集规模及数据维数逐渐增加，模型所需要表示的映射也愈加复杂，因此模型的参数量也随之增加。深度神经网络拥有可以批量性增加模型参数，同时又能够通过通用的梯度下降等方法进行训练使得网络收敛到一个较优目标值的能力，展示了在不同任务上的惊艳表现，因此逐渐获得研究者们的青睐。为解决不同数据类型的应用问题，例如图像数据[75]、自然语言数据[76]、音频与语音数据等[77]，研究者们分别提出了不同的网络结构与训练目标，使得机器学习模型在不同任务上的性能取得多次突破。特别在大规模语言模型上，最新的 GPT 系列模型[78] 拥有 1.76 万亿的参数规模，并已经能够在多个使用场景中接受自然语言的提问并给出合理且有价值的回答，无论是相关研究者还是普通用户都能够从中受益。因此，探究如何结合深度神经网络以及原型学习算法，是扩大原型学习框架应用范围的必要途径。

深度神经网络的突出表现得益于它的表示学习能力，即多层神经网络构成了一个复杂的映射函数，输入数据能够通过神经网络被转化为低维嵌入特征，而该低维特征使得数据更具有区分性。例如，通过对卷积神经网络的可视化研究[79] 可以看出浅层卷积核将输入图像表示为竖线、圆形等基本轮廓的线性组合，而高层次卷积层则将它们组成更复杂模式的线性组合，最终得到的特征向量每个维度都代表某种图像特征是否被激活。此时，一些图像中对于最终任务无用的信息被抛弃，图像被转换到信息密度更高的低维空间中。可以认为，深度神经网络是目前性能最强的表征学习方法。

尽管如此，深度神经网络的表征能力也必须采用合适的损失函数进行训练才能够得到。线性分类层、Softmax 激活函数以及交叉熵损失函数结合进行分类训练的方式自从提出以后变由于其简单有效的表现成为分类问题的首选训练方式[80]。但是，该训练模式也并非在任何场合都能够表现良好。例如，在增量学习

环境中存在对于最近新学习类有偏好（bias）；输出原型与类的个数捆绑，根据新增类的数量输出原型也要增加等缺陷[81]。论文[24] 详细考察了卷积神经网络在增量学习环境中的参数遗忘问题，线性分类层的参数变动程度也远大于多数卷积层参数。因此，采用其他分类模型代替线性分类层成为提升神经网络性能的一种手段。考虑到原型学习框架在变化学习环境中的灵活性，深度神经网络可以视为原型学习框架中扩展参数量的手段，通过结合使用深度神经网络作为前端，原型模型作为后端，可以同时赋予模型更为强大的学习能力和灵活的适应能力。

目前与深度神经网络结合的原型学习模型可以分为两类，一类是在预测时直接使用保存的回放样本计算得到的原型集合，不随着网络的训练过程而进行更新；另一类是随网络对原型进行更新，即使用公式（1-4）中定义的重构误差指导原型分类器进行学习，仅需将激活原型的方式修改为上述定义的指示函数 $\mathrm{I}(\boldsymbol{x}_i, \boldsymbol{p}_j)$。深度增量学习网络 **iCaRL[82] 算法**中使用原型模型替代了线性分类层的功能，具体来说是使用最近回放样本均值（Nearest-Mean-of-Exemplars）的分类模型对卷积神经网络所提取的特征向量进行分类。该原型模型使用单个均值特征向量作为原型向量代表一个类别，每个类别的原型向量通过在内存中保存的回访样本的特征向量求取均值得到。该原型模型并没有学习的功能，仅仅在预测的时候进行原型的计算。其他增量学习方法[23] 也使用了仅在预测阶段进行更新的原型分类模型，并通过实验对比验证了使用原型进行分类在多数方法中都优于采用线性分类层进行分类。除此之外，部分工作[83] 中则使用随训练环境同步进行更新的最近均值（Nearest-Class-Mean，NCM）分类模型，原型在学习的期间就通过减少对同类训练样本重构误差的方式进行更新。该类随时更新的原型模型同时还对特征提取网络的训练有所帮助，即通过损失函数使得同类别的输入特征尽可能靠近该类别的原型向量，增加数据的可分性。

在采用原型学习模型的同时，研究者们对训练分类网络所使用的交叉熵损失函数进行了考察，探索是否有更适合在动态环境中进行网络训练的损失函数。在论文[84] 中对比了在类增量环境中 ResNet 网络采用 Softmax 函数进行精调（fine-tune），采用 Softmax 进行精调但是使用 NCM 分类器，以及使用度量学习（Metric Learning）损失函数及 NCM 分类器三种训练方式，其结果证明第三种学习方式的遗忘程度远小于前两种，即度量学习配合原型分类器在类增量环境中相比原始的交叉熵函数具有优势。近些年，新提出的对比学习损失函数被广泛用于训练

各种问题下的网络嵌入特征提取功能[85][86]。该损失函数的基本思想是使得同类数据的嵌入特征尽可能靠近，异类特征尽可能地远离，使得输入数据的嵌入特征形成相互分离的簇。因此，无论数据原本是否具有标签，只要能对样本的正负有所规定，则就可以使用对比学习进行网络训练。而这一对比学习损失函数同样助益于原型分类器。对比常用的交叉熵损失函数，对比学习损失函数能够学习到类间距离更小的特征。该结论由论文[87] 通过在 CIFAR10、CIFAR100 等具体实验上得到证实，即对比损失函数学习的数据特征将比通过交叉熵损失函数学习的特征更靠近其原型均值。此时，通过原型对数据进行重构，数据的重构误差将更小。因此，对比损失函数与原型分类器在增量环境下的深度神经网络训练中共同发挥了重要的作用。

1.3 待解决问题

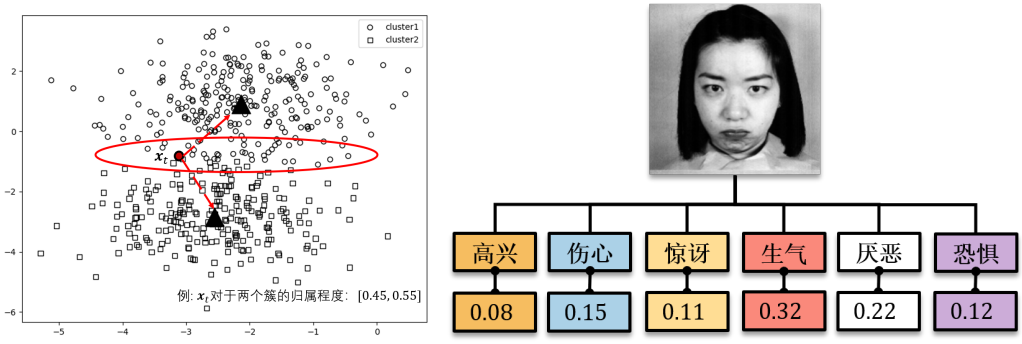
主流对标签预测类方法的研究主要着力于提升算法的预测准确率，即在分类问题上的标签预测准确率，无监督问题上的聚类质量等。时至今日，从计算快速、应对少量可分性强数据的线性模型，到训练时长、应对大规模复杂数据集的非线性及深度模型，不同参数规模的模型被应合理用于适配的任务环境中。然而，对于更为贴合实际场景的模糊性、动态性问题，目前取得的研究成果还不能令人满意，对现有的算法模型提出了较大的考验。例如在增量学习的论文[83] 中，将增量学习模型与传统批量训练的分类性能对比来看，其训练结果仍然无法与已有的批训练方法性能相比。因此考虑到原型学习模型通用性、灵活性、以及可扩展性的优势，本文使用原型学习框架所形成的算法解决下述三个具体研究问题，使得机器学习模型能够在更为复杂的实际数据环境中进行学习。

1.3.1 模糊聚类问题

在无监督的情况下，聚类算法通常通过数据之间的相似性来将输入特征集合划分为属于不同数据模式的子集，但是该划分方式是否合适仍然值得商榷。从物理意义的角度考虑，如果将每个簇看作是一个概念的实例，那么日常生活中所接触到的概念可以分为主观概念和客观概念，例如将包含狗和猫的图像划分到两个不同的簇中，则每个簇所对应的是狗和猫这样不随主观认知所改变的概念，

可将其称为客观概念。如果对多个年收入进行聚类以形成“富有”这个概念，那么拥有九千万收入和一百万收入的人显然对于这个概念的归属程度是不同的；从数据分析的角度进行考虑，当两个数据模式所产生的输入特征较为相似时，根据高密度输入特征的分布可以划分出两个数据簇，但其数据分布通常产生重叠区域（overlap）[88]。例如图1-1a所示的两个人工生成的高斯分布数据簇，使用 k-Means 算法进行聚类时所得到的两个簇划分结果是十分合理的。但对于图中椭圆区域所圈出的输入特征，硬聚类的做法是将其划分为相似程度更高的聚类。但从数据分布上推测，该数据可能由两个数据模式混合产生，因此将其划分为其中一个簇都可能丢失另一个簇的信息。因此，若将其分别计算对于两个聚类的归属程度，则可以观察得到其对于两个数据模式的符合程度，获得更多属于数据的描述信息。

根据以上分析，模糊聚类 *(Fuzzy Clustering)* 问题定义模型应学习输入空间到归属程度向量的函数，该归属程度向量的不同维度分别代表了输入特征对某个簇的符合程度。该问题可以视为对聚类问题的扩展，其困难之处在于簇的数据分布与输入数据对于每个簇的归属程度都是未知的，因此需要两个方向进行交替优化，使得算法模型逐渐收敛到能够表示数据分布的状态。



(a) 模糊聚类问题：来源于人工生成的数据集。

(b) 标签分布问题：来源于标签分布问题的公开数据集。

图 1-1 模糊聚类问题与标签分布问题来源的示例图像。

1.3.2 标签分布问题

在监督学习领域，主流研究通常采用单个最佳标签对输入数据的性质进行描述，即采用单标签预测模型对数据进行描述。然而，实际生活中的事物通常有

多个特征，单标签则遗漏了数据特征所反应的其他性质。尽管将单标签扩展为多标签可以使用标签子集来描述多个相关特征[89][90]，但仍不能够反应不同的标签对实例的不同描述权重。例如实际生活中人类的情感经常混合存在，使用模型对人类情感进行分析时，若将其简单归结为单一的基本情感则可能错误理解对方的沟通意图[91]，从而导致后续决策的失误。对图1-1b所示的公开数据集样例，该图像所示的女性表情所代表的情感是复杂的，因此需要通过不同程度上的基础情绪的组合来作为对该图像的特征描述。数据集中，每张表情图像都被标记为几种基本情绪的混合，共包括快乐、悲伤、惊讶、愤怒、厌恶和恐惧六种情绪。总之，在对客观世界进行分析时，部分场景中使用实数化的多维度分析才能够对数据进行合理描述。

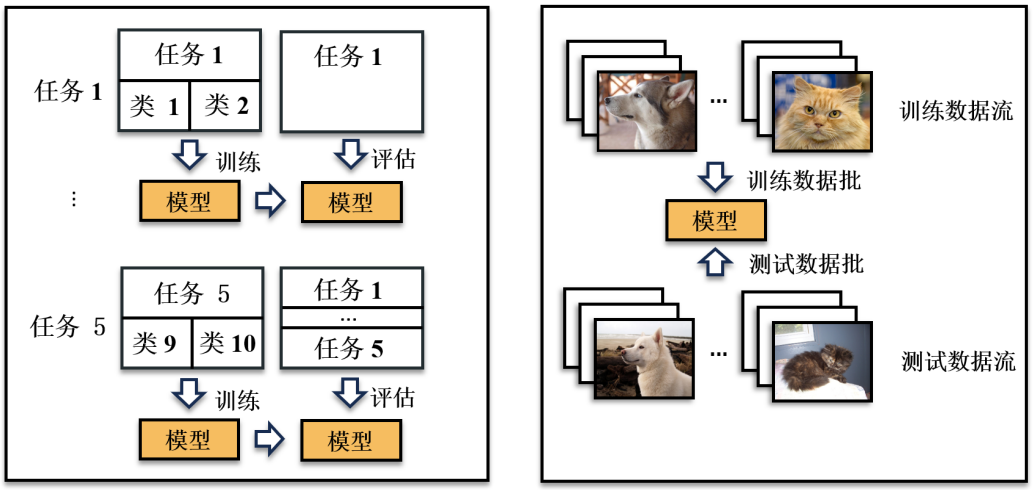
根据以上分析，标签分布 *(Label Distribution)* 问题定义模型应学习输入空间到标签分布向量的函数，该标签分布向量的不同维度代表了某个标签对于输入特征的符合程度。该问题可以视为对单标签和多标签预测问题的扩展，其困难之处在于标签之间可能存在相关性，因此不能将其作为不同标签独立进行预测的单标签预测集合模型进行解决。与模糊聚类相比，标签分布问题拥有人工标注的真实标签，使得模型的优化目标更为明确；但标签之间的相似性与输入特征之间的相似性可能并不统一，且数据分布中不一定对应类标存在输入特征高密度聚集的不同区域使得模型能够将不同类别的数据划分开，因此无法使用模糊聚类的相似度判断思路解决标签分布问题。

1.3.3 数据增量问题

增量学习（Incremental Learning）问题指对训练数据非同分布的情况下如何模型进行学习的研究。如同人类的终身学习一般，人们希望模型在能够持续更新的同时不遗忘已经学习到的知识，即保留所有已经学习到的知识而对学习过的数据进行正确的预测。这种学习模式与传统机器学习模式不同，强调训练是持续化的，而非部署使用后就无法对模型进行更新。可以看出，增量学习过程可以与任一学习问题进行结合，使得解决该问题的模型能够在动态环境中进行持久性的学习。

类增量学习是本文所关注的研究重点，其描述了在学习进行过程中，采用了模型从未见过的新类别数据对模型进行训练。此时，对模型每次进行评估时，都

会使用所有训练过的类别的数据进行实验，无论在当前学习阶段模型是否输入过该类别的新数据。常用的类增量实验设置通常会为学习过程划分不同的学习阶段，每个学习阶段也被称为任务（task）[92]。不同任务阶段之间训练的数据类别之间没有交集，这使得模型在下个阶段无法复习上个阶段的知识，因此要求模型能够在保持旧知识的稳定性与学习新知识的可塑性之间取得平衡[22]。增量学习为模型所带来的困难本质上是由于数据分布的更改导致模型参数随着新的任务目标而变化，因此失去了对旧任务的处理能力[93]。需要注意的是，在本文中在线学习（Online Learning）与增量学习问题不同，更注重的是数据是否是以流式形式，也就是每次输入单个样例对模型进行更新。在线学习与增量学习可同时进行，即流式数据也会在学习过程中接触到新的数据类别，此时模型面临的挑战无疑也更大。



(a) 类增量问题示意图。 **(b) 数据增量问题示意图。**

图 1-2 类增量问题与数据增量问题的模型学习方式比较。

部分研究者们[83] 认为上述类增量的实验环境较为理想化，无法适应实际生活中的动态学习环境。例如野生动物的图像检测系统，该系统既随时可能发现新类别的动物，需要模型进行少量数据更新后就立即可以投入实际应用，而非等待任务数据学习完毕后才能够对系统进行更新；又需要面临不同类别数据之间可能产生不平衡的问题。类增量与数据增量环境下模型的学习方式比较如图1-2所示。

根据以上分析，*数据增量*（*Data Incremental*）问题定义模型应学习输入空间到标签集的函数，该标签集随着训练轮次的增加而扩展。相比类增量问题而言，

数据增量问题中使用小批量的样本对模型进行训练则更增加了模型在训练数据分布产生变化时对抗遗忘以及更新模型的困难。因此，如何增强模型的对抗遗忘、增加模型对新数据的学习能力以及处理非平衡数据流的能力，则是数据增量问题研究的重点。

1.4　本文工作

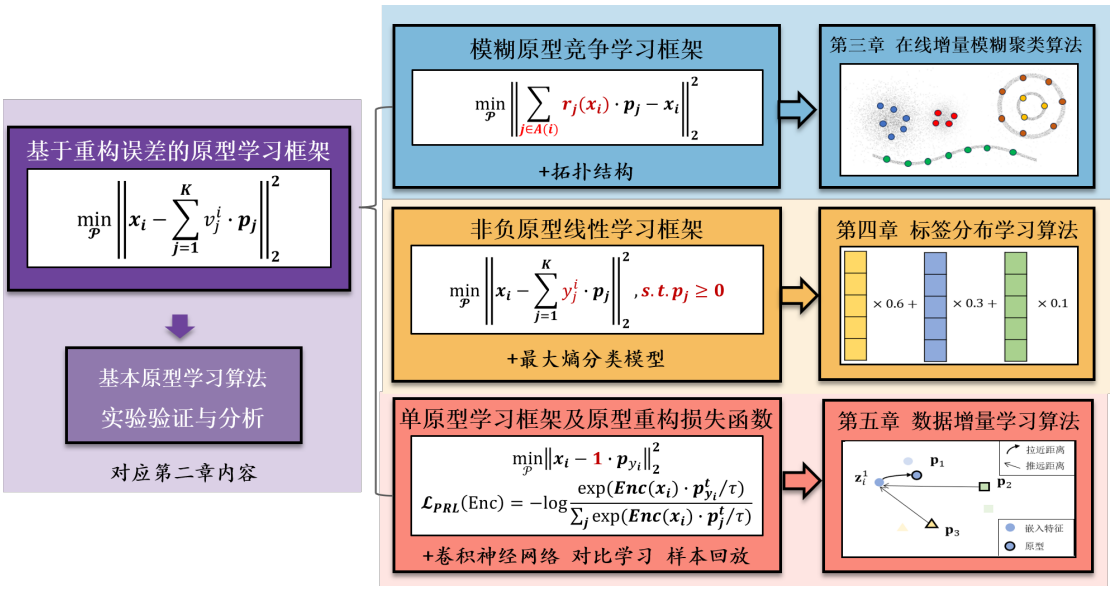


图 1-3　本文各个章节之间的逻辑结构图。

本文试图通过以原型学习框架为基础的算法逐一解决以上三个研究问题。首先，本文对以重构误差为基础的原型学习框架进行总结，并且对原型学习框架的表征学习能力在多个应用领域的数据集上进行验证与分析。在基本原型学习框架的基础上，本文提出如何根据面对的问题场景对基本学习框架进行变换，以及与其他算法模块的组合从而解决相应任务。上述内容主要对应本文的第二至第五章节。

在第二章中，考虑到原型模型可以在无监督的情况下进行通用性的表征学习，本文总结以重构误差为基础的原型学习框架并进行分析。其次，使用上述基本框架所形成的无监督原型学习算法在多个领域数据集上进行重构误差及简单分类任务的评估。根据实验分析可以得出结论，不同的原型学习框架以及算法的其他模块对最终任务指标评估有所影响，因此必须考虑问题特性提出对应的原型学习框架。

在第三章中，考虑到原型集合可以作为模糊集合的支撑向量对数据分布进行表示，本文在模糊原型竞争学习框架的基础上提出在线模糊增量算法。该方法无需提前定义类别与原型的数量，可以根据在线增量环境中的数据分布表示需要增加或删除原型数量。在此基础上，拓扑结构也被用于定义原型之间的相邻关系，使得原型对于数据分布的表示更为合理。

在第四章中，考虑到原型集合可以表示每个标签所对应的数据模式，本文在非负线性原型学习框架的基础上提出标签分布算法。该方法将类别分布视为数据由基本模式线性组合的系数，因此可以通过重构误差学习到对应数据模式的原型。在此基础上，该方法将输入特征转换为原型表示时的系数起到特征提取的作用，使得在嵌入空间进行优化的最大熵模型提升预测性能。

在第五章中，考虑到原型集合可以替代深度神经网络的线性分类层以减少遗忘程度，本文在单原型学习框架的基础上提出数据增量学习算法。该方法使用单原型表示不同类别的数据分布，并提供原型重构损失函数用于训练深度神经网络。在此基础上，回放样本技术及对比损失函数被用于促进神经网络的训练，使得原型模型与深度神经网络模型能够共同训练提升算法的预测性能。

上述章节的逻辑关系如图1-3所示。第二章节总结了基本的原型学习框架，而第三至五章节则改进该基本原型学习框架并用于形成解决研究问题的算法方案。

第二章　基于重构误差的原型学习框架

2.1　引言

本文的研究目标是使用以重构误差为基础的原型学习框架来解决机器模型问题中的动态性与模糊性问题。因此，本章将从通用的学习框架开始讨论，并引出后续改进的三种原型学习框架用于具体研究问题。

首先，本章对基本的重构误差原型学习框架进行定义和分析。重构误差能够考察原型集合还原输入特征的能力，因此展示了该原型集合是否对数据背后的基本模式有所掌握。同时，该学习框架仅需要输入特征自身就能够进行自监督学习，且系数可以控制对于不同原型的更新程度。因此，该原型学习框架具有通用性与灵活性的优势，能够在任务无关的情况下进行表征学习。

其次，为验证以重构误差为基础的原型学习框架，且未引入对数据分布进行预设的情况下形成的原型学习算法具有通用性的表征学习能力，本章在 12 个多领域数据集上，展示基于重构误差的原型学习框架在无标签的基础下在增量学习问题上的表现。通过该实验结果可以看出，重构误差作为误差衡量与例如分类准确率的任务指标并不完全统一，且原型学习框架与算法的其他模块会影响最终分类任务的指标评判。因此，后续章节中将首先对基本的原型学习框架进行改进，再引入与具体问题相关的算法模块，最终使得原型学习算法能够更好地解决对应问题。

2.2　基本重构误差学习框架

2.2.1　目标函数定义

本文所采用的以重构误差为目标的原型学习框架可通过如下方式进行定义：模型学习到含有 K 个原型的原型集合，记为 $\mathcal{P} = \{\boldsymbol{p}_1, ..., \boldsymbol{p}_K\}$。对于输入向量 $\boldsymbol{x}_i$，输入向量可以由原型向量进行带权组合而重构得到，并通过输入向量与重构

向量之间的差别来衡量该原型基底对于当前输入向量的表示能力。对于输入向量 $\boldsymbol{x}_i$，原型学习框架的学习目标是最小化上述重构误差

$$\min_{\mathcal{P}} \mathcal{L}_{recon}(\mathcal{P}) = \min_{\mathcal{P}} \left\| \boldsymbol{x}_i - \sum_{j=1}^{K} v_i^j \boldsymbol{p}_j \right\|_2^2. \tag{2-1}$$

v_i^j 即为对应原型 $\boldsymbol{p}_j$ 在重构输入向量 $\boldsymbol{x}_i$ 时所使用的系数。此时，输入向量 $\boldsymbol{x}_i \in \mathbb{R}^{1\times d}$ 可通过原型集合以及维度为 K 的系数向量 $\boldsymbol{v}_i \in \mathbb{R}^{1\times K}$ 进行表示。

在上述原型系统中，原型在减小重构误差的驱动下对数据进行关键信息的提炼。从数据的角度进行考虑，数据中通常存在冗余信息，其表示的维度通常小于其特征维度。以图像数据为例，通常人们看到的有意义的图像，像素之间必然遵守一些规律，例如图片通常是由相同的或规律变化的颜色区域组成，这意味着这些相同数值或遵守一定规律变化的像素数值。那么通过原型向量表示模型从数据中提炼出的模式规律时，这些模式通常代表该数据集中包含的背景、物体等等信息。将图片总结为上述模式的组合，则可以仅通过原型集合与系数还原出原始图像的关键信息。即使数据中的各个维度完全相互独立，通常也会有包含信息的关键程度区分。当原型数量小于输入数据的实际维度时，则无法完全通过原型重构出输入数据，此时就会产生重构误差。重构误差越大，表示损失的信息越多，因此在系统学习的过程中，应当尽量减少重构误差大小，学习到包含更多关键信息的原型。图2-1以一张经典的 512×512 机器学习图像 Lenna 与奇异值分解方法（Singular Value Decomposition，SVD）为例，通过选取前 K 个奇异向量与其对应奇异值进行线性组合来重构原始图像，并观察该图像与原始图像之间的视觉与数值差异。由于仅靠矩阵分解即可得到奇异值与奇异向量，该方法可以视为一个无需进行学习只需要调整原型数量参数 K 进行矩阵分解的原型学习算法。可以看出，重构误差 $\mathcal{L}_{recon}$ 能够从数值上表示图像还原的质量，而选取 $K = 200$ 时，重构图像就已经拥有肉眼无法分辨的质量。因此，该图像数据中确实存在冗余，使得图像可以使用小于原始维数的向量数量来进行表示；同时，重构误差越小，则说明该原型集合所表示原型学习到图像的更多信息，重构时的结果也越好。

综上所述，数据由基本的模式组合而成，但该模式通常是未知的。本文意图让模型所学习到的原型集合包含基本模式的信息，因此它也能够通过系数组合

(a) 原始图像 **(b)** $K = 200, \mathcal{L}_{recon} = 3.51$ **(c)** $K = 100, \mathcal{L}_{recon} = 8.43$ **(d)** $K = 50, \mathcal{L}_{recon} = 15.86$

图 2-1 使用奇异值分解（SVD）对 Lenna 灰度图像进行处理，并选取前 K 大奇异值所对应的奇异向量进行图像重构，重构图像与原始图像的差值矩阵的 F-范数记为重构误差 $\mathcal{L}_{recon}$。将上述奇异值作为原型集合来描述该图像，则 200 以上的原型集合已经能够实现较小的重构误差，同时达到视觉上几乎完全还原原始图像的效果。

重构出接近原始数据的特征。根据重构误差对重构特征和输入特征差别的衡量，可以一定程度上判断原型对于数据模式的学习情况，从而评估原型集合的质量。因此，重构误差可以作为学习目标指导原型集合的学习。

2.2.2 框架分析

在应用重构误差框架形成具体学习算法时，首先需要考虑如何设定合适的原型数量 K 以及选择重构形式。

原型数量可以根据对数据分布的描述需求而定。这里可以考虑一种极端情况，即保留所有的训练样例作为原型，这也是最近邻算法的思路。当需要进行标签预测时，直接通过计算所有训练特征与测试特征之间的相似度即可预测测试特征的标签为最相似训练特征的标签。最近邻算法不需要训练过程且重构误差为 0，在数据量适当时甚至能够取得很好的结果，因此在当今的机器学习算法中仍然被广泛使用。然而，当数据量过大时，测试特征与所有输入特征之间的距离计算将带来极大的计算代价，且分类性能将会下降[94]。从一个方面来说，原型学习算法需要在计算代价和重构损失中取得平衡；从另一个方面来说，重构误差为 0 并不代表能够完全解决具体任务，还需要考虑原型学习方案带来的影响。在不同的原型学习算法中，通常采用人工控制原型数量参数或初始化定量原型进行学习；或是设定软性的阈值条件，使得算法根据重构误差或者任务表现自动选择合适的阈值对原型数量进行调控。图2-1的例子采用了人工调控的方式，选取 $K = 200$ 甚至 $K = 100$ 都可以达到视觉上令人较为满意的结果；同时也可以设定重构误差的阈值，当重构损失降到该数值以下时就选取该原型数量作为最

终结果。

重构形式可以通过原型是否进行线性组合来重构输入向量来进行分类讨论，即将原型学习算法据此分类为线性模型与非线性模型。线性模型学习到了嵌入空间的基底向量，并使用基底向量对数据进行表示。这类模型通常只使用矩阵计算，复杂度较低，但表达能力有限；而非线性模型则对原型的重构有更加复杂多样的处理，使得原型向量能够表示更为复杂的数据结构。在图2-1的例子中，原型集合通过奇异值系数来进行线性组合。可以看出，SVD 方法已经能够对该图像进行达到较好效果的还原，因此采用线性方法可以达到更快的计算速度。但从更多非线性的算法改进中可以看出，仍有许多数据无法使用线性模型进行学习。

在重构误差的定义与讨论中可以获得如下待证实的推论：首先，重构误差可以验证原型学习集合的学习质量，从而提升模型的任务指标，但该提升幅度是有限的；其次，原型的数量需要通过人工或算法进行精心设置，对算法结果有所影响；最后，线性与非线性模型的选择与数据集相关。考虑到上述框架的实际应用情况需要实验来证实，因此下节将选择由基本的重构误差框架构成的算法，以验证上述待证实的推论。

2.3　对原型学习框架的实验验证

2.3.1　算法说明

本节选取 4 个在线的线性原型学习算法进行实验以验证上节中的推论，这些算法包括增量主成分分析 IPCA[16]，无偏协方差无关增量主成分分析（Candid Covariance-free Incremental Principal Component Analysis，CCIPCA）[31]，增量式正交成分分析（Incremental Orthogonal Component Analysis，IOCA)[17] 以及进化式正交成分分析（Evolutionary Orthogonal Component Analysis，EOCA）。这些算法采用了基本的重构误差学习框架，对数据分布没有预先假设，仅仅通过数据的重构误差来进行原型的学习，因此可作为通用的原型学习方法可以应用于多种数据集上。同时，IPCA 与 CCIPCA，IOCA 与 EOCA 都采用了相互正交的基底原型进行特征表示。IPCA 与 CCIPCA 是对协方差矩阵的特征向量进行更新计算从而得到基底向量；而 IOCA 与 EOCA 则将输入特征进行归一化且正交化后加入基底向量。采用正交基底的优势在于，首先正交的基底在进行特征表示时相

较于非正交的效果更好[95][96]；其次当对正交化的基底进行规范化使得其范数为1时，原型之间的将更线性独立，此时计算的效率与稳定性也将更高。假定需要表示维度为10的空间，那么使用10个相互正交的基底就可以进行表示，若这些基底之间相互不正交，则可能最后计算得到多于10个的基底进行表示。同时，原型矩阵的条件数也将使得其数学计算更为稳定[27]。最后，可以看到在使用正交基底进行系数计算时，其系数不再需要求矩阵的逆，而是可以直接通过基底与输入向量之间的内积计算得到，因此可以提升算法整体的计算效率。

IPCA与CCIPCA延续了PCA的原型学习思路，即通过计算协方差矩阵的特征值来作为原型集合。这两个算法所面临的问题是如何在处理数据流时更新并计算协方差矩阵的特征值。IPCA保留每次已经学习到的协方差矩阵的特征向量及其对应的特征值，并通过重构时得到的残差向量来实现对于更新后的协方差矩阵的近似，而CCIPCA则通过特征向量的性质，即对于第j个特征向量$\boldsymbol{p}_j$，其对所有输入特征$\boldsymbol{x}_t$满足$\lambda_j\boldsymbol{p}_j=\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{x}_t\boldsymbol{x}_t^T\boldsymbol{p}_j$，来对初始化的原型集合进行估计，从而避免了重复求解特征值的问题并降低了计算的复杂程度。这两个算法的原型数量需要通过人工设定的方式进行控制，因此需要在实验中选择多个原型数量设置来比较最终的算法结果。

IOCA与EOCA的原型学习是增量式的，即随着数据的到来逐渐扩展其原型集合的数量，而其扩展的阈值是通过重构误差来进行判定的。如果当前基底集合对于输入向量的重构误差较小，则无需增加基底向量；而当重构误差较大时，则需要增加基底向量的数量。当前已经学习到包含有K个基底的原型集合$\mathcal{P}=\{\boldsymbol{p}_1,...,\boldsymbol{p}_K\}$，则可以通过施密特正交化过程（Gram-Schmidt process）来判断当前输入向量$\boldsymbol{x}_t$使用基底进行表示后获得的残量，即

$$\boldsymbol{r}_1^t=\boldsymbol{x}_t, \tag{2-2}$$

$$\boldsymbol{r}_{j+1}^t=\boldsymbol{r}_j^t-\boldsymbol{p}_j^T\boldsymbol{r}_j^t\boldsymbol{p}_j, j=1,\cdots,K. \tag{2-3}$$

经过所有原型表示后得到的$\boldsymbol{r}_{K+1}^t$即为当前输入特征的残差向量。可以认为，该残量过大时，将残量进行标准正交化加入原型集合则能够降低对当前输入特征表示的重构误差。那么如何确定当前的对于已经学习过的输入特征，该标准正交化的基底通过阈值$\frac{\|\boldsymbol{r}_{K+1}^t\|_2}{L_{max}^t}\geqslant f(\frac{K}{n})$来判断是否残量$\boldsymbol{r}_{K+1}^t$应当加入原型基底。如

果满足该阈值，则将 $\boldsymbol{r}_{K+1}^{t}$ 标准化后加入原型集合。阈值中的 L_{max}^{t} 表示在当前时间节点 t 时所计算过的所有输入特征向量模的最大值。因此，使用阈值控制的这两个算法在实验中无须对原型数量进行设置。

考虑到在线数据难以做到及时的数据清洗，并且对数据的均值等信息在学习前难以进行估计，EOCA 增加了辅助子空间分割学习过程。具体来说，EOCA 在每个阶段的子空间中仍进行与 IOCA 相同的基底学习过程，而在学习到收敛阶段后，将辅助子空间的基底与主空间的基底进行合并。在合并时需要注意的是基底向量的正负号在学习时可以通过系数调整而不影响学习结果，但是合并时会造成合并的结果不唯一。因此需要找到两个子空间向量之间的配对关系，该配对使得原型向量之间的夹角最小。原型向量将辅助子空间对应的基底原型矩阵记为 $\mathbf{P}_2$，主空间对应的记为 $\mathbf{P}_1$，则可以通过计算 $\mathbf{P}_1^T\mathbf{P}_2$ 的 SVD 分解得到转换矩阵，将两个基底矩阵中的原型转换为一一对应关系后再进行合并。

综上，上述四个算法采用了基本的原型学习框架，IPCA 与 CCIPCA 通过对数据协方差矩阵的特征向量进行估计得到原型集合，并且需要人工设定原型数量参数；而 IOCA 与 EOCA 则采用了相同的原型学习框架，即通过阈值控制残量向量是否加入原型集合成为新的正交化原型基底，因此通过算法自动估计所需原型数量。

2.3.2　实验设置

本节实验中选择 12 个不同类型的数据集，来比较上述四个在线原型学习算法的实验性能。数据集的具体细节如表格2-1所示。该表格展示了数据集的应用领域、特征维数、训练集以及测试集规模，以及类别个数。

为验证上述线性原型算法的实验结果，本节采用两项指标，首先是相对重构误差（relative reconstruction Error，E），该指标与已定义的（绝对）重构误差相比能够减少输入特征的模对结果的影响。其具体计算公式如下：

$$E = \frac{1}{N}\sum_{t=1}^{N}\frac{\|\boldsymbol{x}_t - \sum_{j=1}^{k}\boldsymbol{b}_j\boldsymbol{b}_j^{\top}\boldsymbol{x}_t\|_2}{\|\boldsymbol{x}_t\|_2}. \tag{2-4}$$

在此基础上，本节采用了简单的分类任务作为任务样例对上述算法结果进行衡量。为避免分类模型对结果的影响，本节采用最近邻分类器，对原型算法

表 2-1 来自不同领域的 12 个数据集，该表格展示了数据集的应用领域、特征维数、训练集以及测试集规模，以及类别个数。

数据类型	数据集	# 特征	# 训练集	# 测试集	# 类别
数学	Hill-Valley[97]	100	606	606	2
物理	Ionosphere[97]	34	251	100	2
化学	Musk[97]	166	476	6598	2
图像	OptDigit[97]	64	3823	1797	10
物理	Sonar[97]	60	104	104	2
物理	Waveform[97]	21	900	4100	3
物理	IJCNN1[98]	22	49990	91701	2
音频	ISOLET[97]	617	6238	1559	26
生物	protein[99]	357	17766	6621	3
物理	SensIT Vehicle[100]	50	78823	19705	3
图像	USPS[101]	256	7291	2007	10
图像	CIFAR10[102]	3072	50000	10000	10

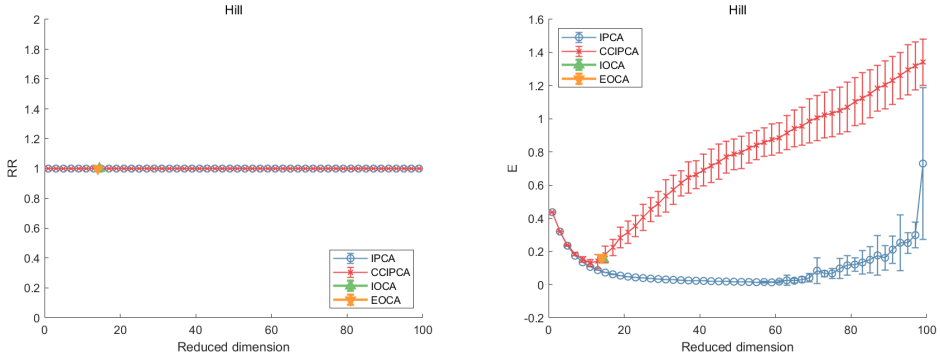
将输入特征转换的系数特征进行分类，并根据分类准确率（recognition rate，RR）来评估分类结果。RR 指标由分类正确的测试样例数量除以总体测试样例的数量得到。该分类器没有分类模型的训练过程，因此也可以更为直观地反映出不同原型学习框架在分类任务评估体系下的表征学习能力。

考虑到在线环境中需要考虑噪声数据对输入特征的影响，同时由于无法进行数据预处理过程，算法必须拥有对少量噪声数据的鲁棒性，因此在 12 个数据集上加入噪声数据作为噪声数据流实验环境评估上述四个算法的表现。噪声数据通过较大的模来试图破坏算法的学习过程，其通过公式 $20L\boldsymbol{z}$ 来产生，其中 $\boldsymbol{z}$ 是随机产生的单位向量，而 L 表示数据集中数据的平均模，即 $L = \frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{x}_i\|_2$。规定噪声率 C 来控制数据集中噪声的数量，即数据集中最多产生 $\lfloor CN \rfloor$ 个噪声，并且这些数据将先于正常数据输入。在实验中分别测试了 $C = 0.1$ 以及 $C = 0.01$ 的结果。

对于 IPCA、CCIPCA 的原型数量设置，本文会首先根据 IOCA 与 EOCA 得到的原型数量结果，来选择其周边一定范围内的原型数量进行比较。在噪声数据上，则将这两个算法都设置为与 EOCA 相同的原型数量。

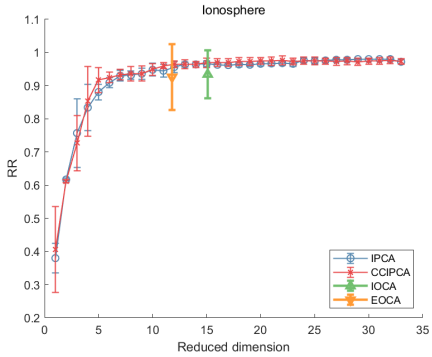
2.3.3 结果分析

图2-2、2-3与2-4展示了上述四个在线原型算法在 12 个数据集上的指标表现。首先，对比重构误差与识别率曲线的增长或下降趋势，对可以人工指定原型数量

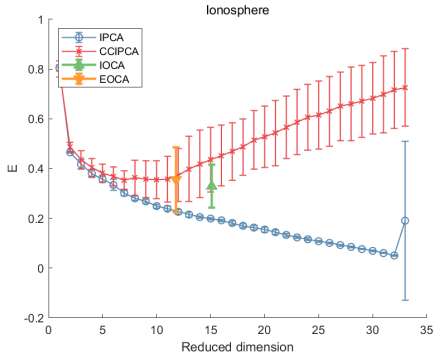


(a) Hill, RR

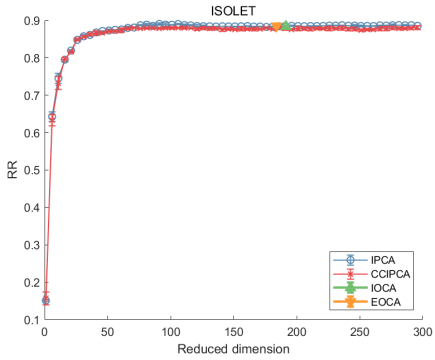
(b) Hill, E



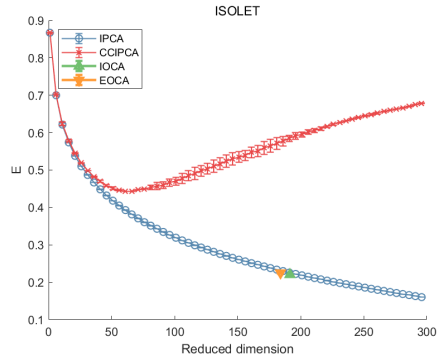
(c) Ionosphere, RR



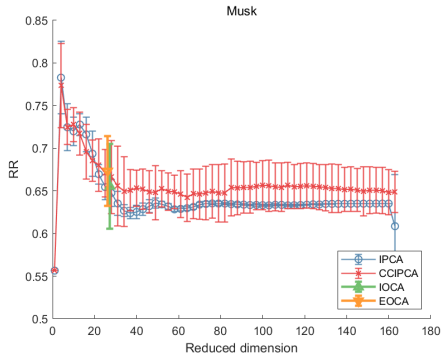
(d) Ionosphere, E



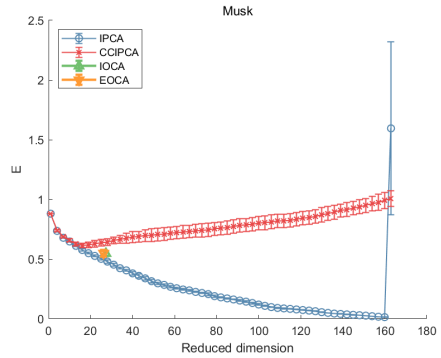
(e) ISOLET, RR



(f) ISOLET, E

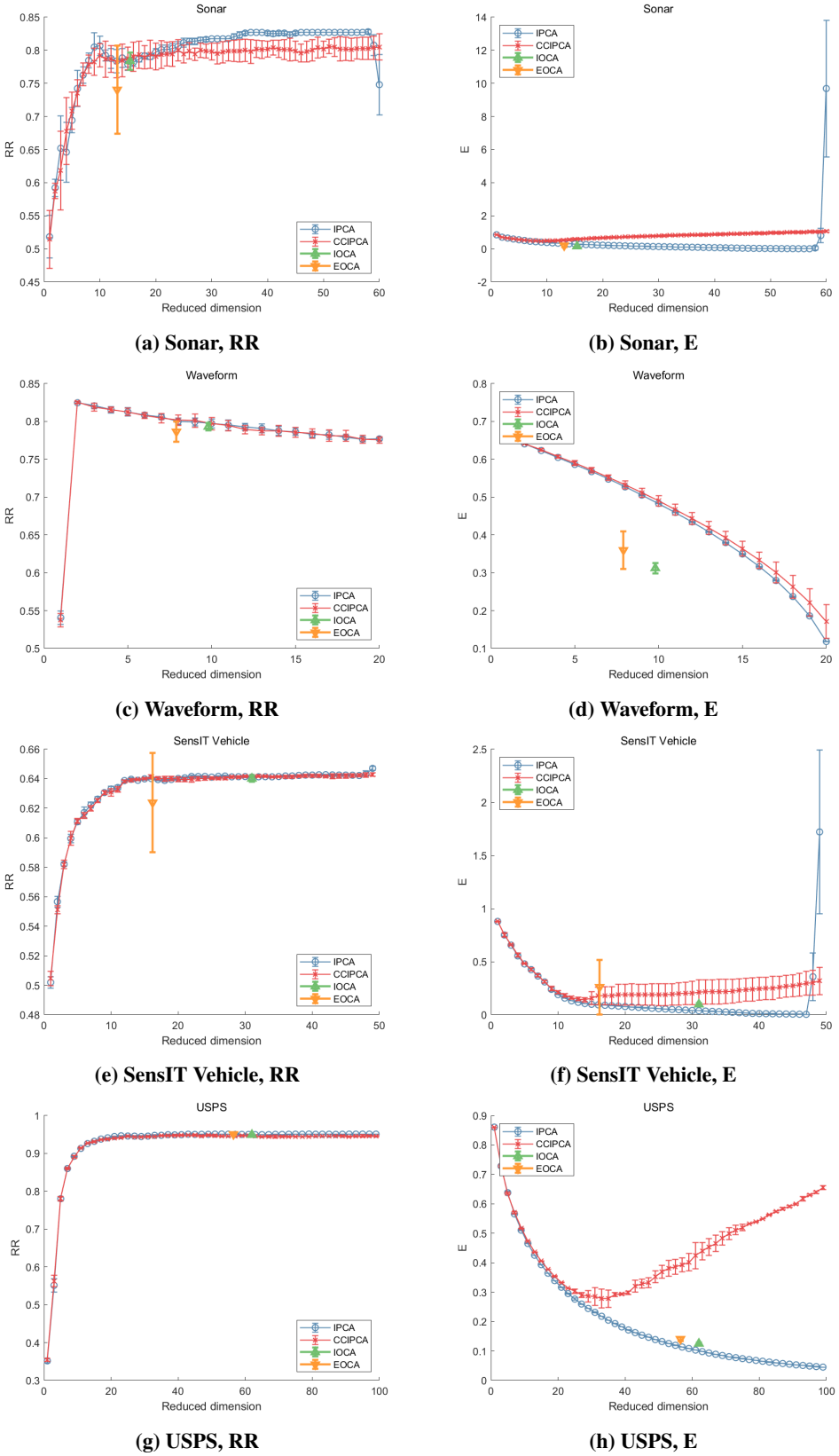


(g) Musk, RR



(h) Musk, E

图 2-2　四种原型学习方法在 Hill、Ionosphere、ISOLET、Musk 数据集上的表现。横轴为原型数量，纵轴为识别率 RR 以及重构误差 E，中心标记表示结果的均值，竖线表示结果的方差。



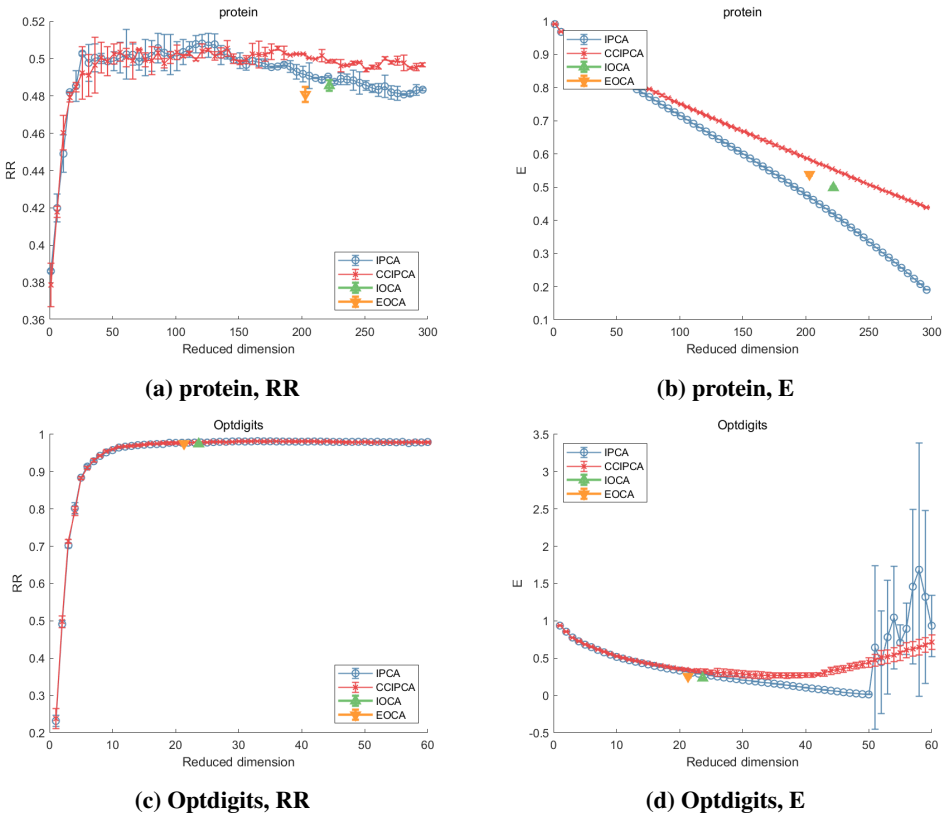
(a) Sonar, RR (b) Sonar, E

(c) Waveform, RR (d) Waveform, E

(e) SensIT Vehicle, RR (f) SensIT Vehicle, E

(g) USPS, RR (h) USPS, E

图 2-3 四种原型学习方法在 Sonar、Waveform、SensIT Vehicle、USPS 数据集上的表现。横轴为原型数量，纵轴为识别率 RR 以及重构误差 E，中心标记表示结果的均值，竖线表示结果的方差。



(a) protein, RR　　**(b) protein, E**

(c) Optdigits, RR　　**(d) Optdigits, E**

图 2-4　四种原型学习方法在 protein、Optdigits 数据集上的表现。横轴为原型数量，纵轴为识别率 RR 以及重构误差 E，中心标记表示结果的均值，竖线表示结果的方差。

的 IPCA 与 CCIPCA 来说较为特殊的是 Waveform 与 Musk 数据集。这两个数据集都在原型数量增长的初期达到了最优的识别率，并随着原型数量的增加反而识别率逐渐降低。从重构误差的角度考虑，Waveform 的重构误差逐渐降低，而在 Musk 数据集上甚至出现了重构误差逐渐上升的情况。考虑到 IPCA、CCIPCA 实际上是对特征向量进行估计而非实际对协方差矩阵进行特征求解，因此可以认为多余的原型向量难以进行优化求解，反正作为噪声向量影响了最终的性能指标。除此以外的数据集则可以得到较为普适的结论，即随着原型数量的增长，结果的基本趋势是重构误差降低，识别率上升；但最终结果的提升幅度的增加是逐渐放缓的，具体表现为识别率的提升曲线与重构误差的下降曲线的逐渐平滑。IOCA 与 EOCA 方法自主确定的原型数量也基本较为适宜，结果通常位于其他两个算法上升区间的平稳区域，为其他算法的参数设置提供参考数值区间，并且自身在不同指标上也实现了较好的性能。

其次，IOCA 与 EOCA 采用了相同的原型学习框架，因此对比其在正常数据集上的表现结果，可以认为在普通数据集上这两者性能相当。EOCA 与 IOCA 相

比较通常最终确定了较少的原型数量，其最终指标表现在 Hill、Ionosphere、Isolet、Optdigits 数据集上与 IOCA 算法相当，在 Musk、Waveform 数据集上有所提升，protein、Waveform、USPS 数据集上指标互有胜负，其余数据集上则不如 IOCA 算法。可以认为，EOCA 在学习过程中实际上的阈值控制较 IOCA 更为严格，尽管所采用的公式是相同的；同时，两个算法在正常数据集上的表现基本相当。

最后需要注意的是，识别率与重构误差在结果衡量上并不统一。例如在 Musk 数据集上，EOCA 取得了更好的分类性能，但是在重构误差上与 IOCA 其实是相当的；而在 Sonar 数据集上，重构误差相当的 IOCA 与 EOCA 算法，却是 EOCA 取得了分类性能上的优势。可以认为，重构误差代表着通用表征学习的能力，而识别率代表着与任务相关的表征学习能力，而这二者在仅依靠重构误差进行训练时并不统一。因此，可以认为重构误差所判断的好的表征并不一定是对任务更为有效的表征。

IPCA，CCIPCA，IOCA 与 EOCA 四个原型算法在噪声数据集上的表现如表格2-2所示。对比正常数据集上的实验结果可以看出，四个算法在噪声环境中都有不同程度上的性能下降。EOCA 在 $C = 0.01$ 的设置下的六个数据集上达到了最优分类性能，九个数据集上达到了最佳重构误差性能；在 $C = 0.1$ 的设置下的九个数据集上达到了最优分类性能，十个数据集上达到了最佳重构误差性能。考虑到 EOCA 采用的原型学习框架与 IOCA 相同，仅通过引入辅助子空间来减少噪声数据的影响。因此，在数据分布根据任务不同有特殊分布情况时，引入其他算法模块对特殊数据分布进行处理能够较为有效地提升任务性能。

由于该四项算法仅采用矩阵与向量的线性计算，因此能够估计理论上的算法复杂度。本文将上述算法的计算复杂度与在实验中的实际训练时间统计在表2-3中，其中运行时间是由 Matlab tic/toc 命令进行 10 次运行的记录并求取平均值得到。由于其通常只涉及矩阵的单次计算，线性模型的一大优势是进行训练模型与预测时的计算优势。除去 IPCA 由于需要随输入向量进行多次特征值的计算而复杂度更高，其余快速方法的复杂度都是 $O(NdK)$。当然，复杂度分析中的常数、实现细节和机器状态都会影响实际运行时间，因此最终他们的实际运行时间也会存在差异。需要注意的是，EOCA 算法由于对基底矩阵的合并操作使用到了 SVD 分解算法，其复杂度是 $O(dK^2)$，但由于该操作总共只需要进行 $O(\frac{N}{d})$，则最终复杂度为 $O(NK^2)$，与 $O(NdK)$ 合并后复杂度仍为 $O(NdK)$。可以看出，

表 2-2　四种在线原型学习算法在不同噪声率 C 的噪声数据集上的表现。结果以 $mean \pm std$ 表示，最佳结果以粗体表示，下划线表示优于 IOCA 的 EOCA 算法结果。

(a) $C = 0.01$

数据集	指标	EOCA	IPCA	CCIPCA	IOCA	数据集	指标	EOCA	IPCA	CCIPCA	IOCA
Hill	K	13.0	13.0	13.0	9.0	IJCNN1	K	10.0	10.0	10.0	12.6
	RR(%)	100 ± .00	100 ± .00	100 ± .00	100 ± .00		RR(%)	91.63 ± 0.00	**95.88 ± 1.66**	93.95 ± 1.79	95.63 ± 1.45
	E	**.171 ± .000**	.173 ± .002	.280 ± .025	.354 ± .002		E	**.350 ± .000**	.603 ± .022	.626 ± .019	.537 ± .020
Ionosphere	K	12.0	12.0	12.0	8.0	ISOLET	K	147.0	147.0	147.0	65.0
	RR(%)	92.00 ± .00	**94.90 ± 1.50**	94.00 ± 1.56	78.90 ± 1.10		RR(%)	**88.71 ± .00**	88.68 ± .18	86.40 ± .40	80.12 ± .11
	E	.415 ± .000	**.221 ± .014**	.372 ± .083	.525 ± .012		E	**.272 ± .000**	.313 ± .001	1.926 ± .091	.679 ± .003
Musk	K	29.0	29.0	29.0	7.0	CIFAR10	K	341.0	341.0	341.0	503.0
	RR(%)	**67.38 ± .00**	62.66 ± .60	64.63 ± 5.02	65.97 ± 9.04		RR(%)	37.23 ± .13	36.24 ± .21	**38.85 ± .05**	32.38 ± .44
	E	**.517 ± .000**	.525 ± .010	.853 ± .002	.832 ± .010		E	**.122 ± .002**	.521 ± .001	2.714 ± .070	.355 ± .008
OptDigits	K	19.0	19.0	19.0	28.5	protein	K	206.0	206.0	206.0	146.0
	RR(%)	**97.55 ± .00**	95.76 ± .50	–	96.28 ± .39		RR(%)	47.47 ± .00	49.27 ± .53	–	**46.55 ± .62**
	E	**.297 ± .000**	.679 ± .012	–	.444 ± .013		E	.549 ± .000	**.525 ± .001**	–	.726 ± .002
Sonar	K	16.0	16.0	16.0	8.0	SensIT Vehicle	K	22.0	22.0	22.0	25.0
	RR(%)	78.84 ± .00	**79.71 ± .84**	78.90 ± 1.94	71.87 ± 2.18		RR(%)	**63.89 ± .00**	63.50 ± .23	63.30 ± .19	63.51 ± .18
	E	.158 ± .000	**.152 ± .002**	.385 ± .06	.342 ± .003		E	**.126 ± .000**	.362 ± .013	.774 ± .052	.519 ± .016
Waveform	K	8.0	8.0	8.0	9.7	USPS	K	54.0	54.0	54.0	75.0
	RR(%)	**79.49 ± .00**	75.03 ± 3.79	74.22 ± .61	72.63 ± 3.01		RR(%)	**95.12 ± .00**	93.48 ± .19	93.92 ± .26	93.57 ± .35
	E	**.343 ± .000**	.603 ± .045	.892 ± .088	.403 ± .020		E	**.143 ± .000**	.510 ± .004	1.435 ± .073	.533 ± .004

(b) $C = 0.1$

数据集	指标	EOCA	IPCA	CCIPCA	IOCA	数据集	指标	EOCA	IPCA	CCIPCA	IOCA
Hill	K	15.0	15.0	15.0	43.3	IJCNN1	K	10.0	10.0	10.0	13.2
	RR(%)	100 ± .0	100 ± .0	100 ± .0	100 ± .0		RR(%)	93.45 ± .02	**95.60 ± .79**	95.30 ± 1.0	95.2 ± 1.1
	E	**.146 ± .008**	.255 ± .004	.262 ± .014	.340 ± .002		E	**.350 ± .000**	.379 ± .079	.810 ± .013	.527 ± .018
Ionosphere	K	12.0	12.0	12.0	16.4	ISOLET	K	167.0	167.0	167.0	256.0
	RR(%)	92.00 ± .00	89.90 ± 8.24	**92.90 ± 5.80**	91.30 ± 6.09		RR(%)	**88.54 ± .20**	86.72 ± .51	84.93 ± .47	87.68 ± .51
	E	.415 ± .000	**.2420 ± .016**	.255 ± .017	.443 ± .019		E	**.272 ± .000**	.313 ± .001	1.926 ± .091	.679 ± .003
Musk	K	29.0	29.0	29.0	50	CIFAR10	K	366.6	367	367	1239.2
	RR(%)	**67.36 ± .04**	66.99 ± 8.5	66.59 ± 6.4	59.42 ± 8.2		RR(%)	**37.15 ± .10**	33.89 ± .16	34.09 ± .19	34.37 ± .31
	E	**.517 ± .000**	.838 ± .005	.879 ± .045	.659 ± .008		E	**.122 ± .001**	.201 ± .001	.438 ± .023	.396 ± .004
OptDigits	K	18.0	18.0	18.0	31.0	protein	K	206.0	206.0	206.0	153.8
	RR(%)	**97.55 ± .00**	93.90 ± .5	–	96.69 ± .4		RR(%)	**47.53 ± .0**	47.52 ± .55	–	46.86 ± .88
	E	**.297 ± .000**	.300 ± .049	–	.436 ± .011		E	.549 ± .000	**.533 ± .001**	–	.713 ± .002
Sonar	K	16.0	16.0	16.0	13.0	SensIT Vehicle	K	22.0	22.0	22.0	26.6
	RR(%)	**78.84 ± .00**	76.63 ± 1.11	75.29 ± 1.70	76.64 ± 2.61		RR(%)	**63.89 ± .00**	63.57 ± .20	63.53 ± .35	63.63 ± .23
	E	**.158 ± .000**	.309 ± .010	.792 ± .096	.513 ± .019		E	**.126 ± .000**	.362 ± .013	.774 ± .052	.519 ± .016
Waveform	K	8.0	8.0	8.0	11.6	USPS	K	54.0	54.0	54.0	111.6
	RR(%)	**79.49 ± .00**	69.58 ± 3.4	70.54 ± 2.8	74.12 ± 3.0		RR(%)	**95.07 ± .11**	93.15 ± .44	92.81 ± .34	94.22 ± .24
	E	**.343 ± .000**	.403 ± .014	.544 ± .031	.376 ± .021		E	**.142 ± .003**	.287 ± .029	.311 ± .066	.475 ± .006

表 2-3　IPCA、CCIPCA、IOCA 与 EOCA 的理论计算复杂度与实际运行时间的对比。

理论时间复杂度	IPCA	CCIPCA	IOCA	EOCA
	$O(NdK^2)$	$O(C_1NdK)$	$O(C_5NdK)$	$O(C_6NdK)$
实际运行时间 (s)	IPCA	CCIPCA	IOCA	EOCA
Hill-Valley	4.09	0.42	0.02	0.02
IJCNN1	4.8	2.9	0.32	0.34
Ionosphere	0.05	0.03	<0.01	0.01
ISOLET	96.5	23.0	6.8	2.5
Musk	1.76	0.39	0.02	0.02
Optdigits	2.50	0.72	0.07	0.08
protein	111.4	48.5	9.99	7.90
Sonar	0.05	0.02	<0.01	<0.01
SensIT	54.3	15.3	1.87	2.06
Waveform	0.11	0.08	0.02	0.02
USPS	50.09	5.16	0.84	0.95
CIFAR10	5938.0	2812.5	618.7	943.3

由于 IPCA 需要多次的特征值求解，因此其实际运行复杂度较高，而其余算法则有着相近的算法复杂度。

实际运行时间也基本符合算法复杂度的分析。从表格可以看出，上述四个算法方法的运行速度基本可以总结为 IPCA < CCIPCA < IOCA ≈ EOCA。可以看出，尽量避免矩阵的特征值计算可以带来更多的计算优势。虽然实验设置通常有所不同，但通过当前数据集可以简要对比线性模型与引入深度神经网络的原型学习算法。在较为大型的数据集，例如 CIFAR10 数据集上，线性模型计算复杂度低，只需要 CPU 设备即可进行训练且仅需要几十分钟-两个小时，而深度神经网络则需要 GPU 设备且需要训练多个小时；但性能上，仅从本文第五章节的实验结果也可看出，该章节中的原型学习模型能够达到 70%+ 的识别率，远远超过了线性模型。但是对于一些较为小型的例如 Hill、Optdigits 等数据集，线性模型已经能够达到接近 100 的识别率，因此也不需要大型模型来解决这类问题。因此，采用何种形式的原型学习算法进行处理，则需要根据任务与数据集规模进一步进行分析。

上述实验证实，以重构误差为基础的原型学习框架在在不同领域的无监督数据集上可以学习到通用的表征提取能力，其既可以作为数据压缩的算法对数据的维数进行降低以便于传输或保存，同时还可以作为例如分类问题等其他机器学习问题的特征提取算法，为其他方法提供更好学习的表征空间。同时，对于增量学习环境，原型学习框架也可以通过重构误差来自主控制增加原型的阈值，使得模型能够应对新来的数据分布。

但是对于不同的应用问题，可以从实验结果中看到重构误差的持续降低并不意味着最终任务将一直随之提升。根据前文所分析的，重构误差验证了原型集合中对于数据基本模式的包含情况。那么随着原型数量的增加，即使其中增加了冗余信息也对数据的表示影响不大。但是，具体任务中原型集合只需要任务相关的信息即可进行预测，甚至只学习到部分具有区分性的数据模式的情况下能够达成更好的结果。此时，冗余信息对最终任务的结果影响就不大，甚至可能反而影响模型预测。因此，能够重构数据的表征学习并不等于在具体任务上有区分性的表征学习，这使得基本原型框架仍需要考虑到具体任务信息来进行改进，从而学习到包含任务关键信息的原型集合。

2.4　小结

这一章节中总结了基本的重构误差原学习框架，并使用其形成的算法在多个数据集上进行实验验证并分析结果。在无任务信息的条件下，重构误差能够训练出具有通用性的原型模型用于表征学习。但是，在实验中观察发现该表征学习系统对于分类任务时，重构误差与分类准确性并不一致，因此仅采用重构误差作为不同任务下的算法目标仍是不足够的。因此，后续章节中本文提出在通用学习框架上进行改进而得到的任务专用原型学习框架，并且引入其他算法模块用于更好地应对不同的研究问题。

第三章　基于模糊原型竞争学习框架的在线模糊聚类算法

3.1　引言

考虑到实际数据分析中对交叠区域数据的处理，以及输入特征对于聚类所对应的物理意义有不同的符合程度，模糊聚类算法需对输入特征预测对于所有簇的归属程度，可以视为对传统聚类问题的扩展。尽管理想中的模糊聚类算法能够为数据带来更为准确的聚类描述，但是如何通过没有标签信息的数据划分出不同的聚类仍然是该类问题的难点。此时数据对于聚类的归属程度也同样是未知的，模型需要从数据中挖掘出该聚类的数据分布函数。同时，本章工作还试图从在线增量学习的角度来解决上述模糊聚类问题。在线增量学习为模糊聚类算法带来的困难是无法预先定义聚类的数量，因为随着训练的过程数据分布随时会产生变化，从而需要对新的聚类簇进行表示。因此，模型如何学习数据分布的变化但又对已学过的知识进行保存，则同样是需要解决的重点问题。

考虑到原型学习框架在无监督环境下能够使用原型集合表示模糊聚类中每个簇的分布，还能够自主进行原型数量的增加以对数据流中的簇数量进行估计。因此，为解决在线增量模糊聚类问题，本章提出模糊原型竞争学习框架，并结合自组织增量神经网络算法的拓扑结构学习方式，提出模糊自组织增量神经网络算法对数据进行聚类表示。该算法本身具有能够在在线增量环境下进行学习的能力，能够随着训练的进行自行增加或者删除神经元节点。同时，采用多个原型描述同一个聚类既可以对非高斯分布的数据进行描述，同时还可以作为聚类的支撑节点，用于预测输入特征对于该簇的归属程度。在实验章节中，本文采用人工数据实验以及可视化展示了模糊聚类的效果，并通过数值指标证实了该算法在这一问题上的有效性。

3.2　研究背景

模糊聚类是聚类技术与数学中的模糊集（Fuzzy Set）概念相结合的产物。通常的聚类算法将输入数据集合通过不同的相似度判定准则而划分成没有交集的子集，这种聚类方式可以被称为是硬聚类。但在实际应用中，并不是每个输入数据都会完美符合其所属的模式，而更有可能是多种模式的混合，这一现象在数据分析上体现为数据簇之间的重叠区域：即数据位于两个高密度簇分布的边缘区域，无法很好的判定其到底属于哪个簇。这一现象在 1968 年就由 Nagy[103] 在数据分析时所观察得到。为了解决这个问题，在 Zadeh[104] 提出的模糊集概念上，Ruspini[88] 提出模糊聚类学习模式：即为输入数据学习其到每个簇的归属程度，而非仅将其划分到某个簇中而不属于其他任何簇。这种聚类方式不同于硬聚类中忽略、强制分开，甚至强行将位于两个簇中间低密度区域的数据视为噪声的做法，而是为描述复杂的数据分布提供了一种手段。考虑到现实场景中，无监督数据通常并非一下子完全得到而是逐步收集得到。此时，输入数据来源的隐藏分布很有可能是动态变化的，而非独立同分布的。例如，在图像类别识别场景中，随着任务的变化，所需要识别的目标类别也会有所增加；在病症诊疗系统中，可能随时会发现从未见过的病例，其症状也有可能会产生变化。物理世界是动态的环境，而将人类所具有终身学习的能力是机器学习模型所必须的。因此，本章试图解决线增量学习场景中的模糊聚类问题。

模糊聚类算法大多由已有聚类算法改进产生，其主要包含以下三种算法及其衍生算法，即模糊 C-均值（Fuzzy C-Means，FCM），模糊自适应共振理论（Fuzzy Adaptive Resonance Theory，Fuzzy ART），自组织映射网络（Self-Organizing Map，SOM）。需要注意的是，模糊聚类技术可以看作是对聚类技术的软化，从对数据进行二值化的划分变成了计算实数的归属度向量。因此，上述模糊聚类算法基本都是对已有二值化算法的模糊化计算改进，其最终对于数据的聚类是否是模糊性的可以根据具体任务的需求而进行改变。

FCM[105] 是对 K-Means 方法进行模糊聚类处理后形成的改进方法，时至今日仍是模糊聚类算法的热门研究方向。通过实验对比[106] 可以看出，FCM 相比于 K-Means 提升了对于复杂数据分布的处理能力，但是使用了更多时间用于模型训练的收敛。但由于其只使用均值来描述数据，仍然无法很好地处理非高斯分布的

输入数据，因此有多种改进方式对此方向进行改进，例如采用核函数将原始数据映射到高维空间的方法[107]，以及 Gustafson-Kessel 算法[108]，Fuzzy C-Varities[109] 等。一些改进工作[110] 则能够解决类别不平衡的聚类问题。除此之外，增量 K-Means 算法[111] 能够在在线环境中进行学习。除此之外，Online FCM 等一系列改进算法[112][113] 则将在线数据分批进行处理从而应对数据流的问题。论文[18] 将核方法应用于在线 FCM 算法上。Fuzzy ART[114] 则是 ART 算法[115] 与模糊逻辑计算方式的结合，其可以通过补码解决神经元类的增长过多的问题。由于 ART 系列的算法本身就具有增量式增加新神经元用于学习在线知识的能力，因此其可以直接应用在在线增量的学习环境中。论文[116][117] 中的改进提升了 Fuzzy ART 算法的鲁棒性和聚类准确性。如绪论章节所述，SOM 类算法的改进中许多采用了软竞争学习策略，即在某个神经元获胜之后，不仅这个神经元获得了学习的机会，其他竞争失败的神经元也有一部分能够进行学习，这与 winner-take-all，也就是硬竞争学习策略相对。研究者们指出，该类算法采用软竞争策略使得所有神经元在竞争过后都能进行幅度不同的学习的神经网络是广义上的用于模糊聚类的神经网络[118]。同时，SOM 的模糊版本 Fuzzy SOM[119] 采用了输入向量到神经元的归属度函数来确定神经元的学习率，同样也属于软竞争学习策略。在具体应用领域，图像处理[120][121]，生物医疗[122][123][124] 等领域都有模糊聚类算法的应用。

本章节提出模糊自组织增量神经网络（Fuzzy Self-Organizing Incremental Neural Network, Fuzzy SOINN）用于聚类与模糊聚类问题。该网络结合模糊原型学习与自组织增量神经网络 SOINN[19][20] 的拓扑学习模型，能够在在线增量的环境下保留旧任务相关的原型，同时根据激活次数以及阈值来控制原型节点的增加，从而在网络的稳定性和可塑性中取得平衡。同时使用归属函数确定输入特征对于原型的激活程度，从而使得原型进行软竞争的情况下将归属程度作为学习权重进行学习。最终，模型学习到的原型集合能够对数据分布进行表示，并根据对于原型的归属程度预测输入特征对于聚类的归属程度。对于本章工作的具体贡献，可总结如下：

- 提出模糊原型竞争学习框架引入输入特征的归属程度用于原型学习，使得原型能够作为数据簇的支撑原型表示数据分布，且能够在在线增量的环境下学习动态数据分布；
- 结合拓扑结构的学习形成模糊聚类算法 Fuzzy SOINN，从而实现对输入数

据的归属程度预测；

- 实验章节中分别通过人工数据集上的可视化结果与公开数据集上的指标评估结果验证了本章工作在聚类问题与模糊聚类问题上的有效性。

3.3　本章工作

3.3.1　问题定义

本章对模糊聚类问题进行定义，而硬聚类可以视为模糊聚类的一种特殊情况包含其中，该定义遵循绪论给出的标签预测定义1.1。在线模糊聚类问题下，模型将不断学习在线数据流传来的无标签数据，并试图从相似的模式中归结出聚类，并判断样本对于这些聚类的归属程度。模型从数据流 $\mathcal{S}_{train}$ 中持续不断的采样输入数据样本 $\boldsymbol{x}_t$，且 $\boldsymbol{x}_t \in \mathbb{R}^d$ 表示 d 维数据。聚类模型使用 $\boldsymbol{\theta}$ 进行表示，模型的学习目标是学习从输入空间到输出空间的映射，且为从输入空间到输出空间的映射 $\theta : \mathbb{R}^d \to \mathbb{R}^C$，其中 C 是模型产生的聚类数量。在评估模型时，模型从测试数据流 $\mathcal{S}_{test}$ 中获得测试数据 $\boldsymbol{x}_l$，模型预测 $\hat{\boldsymbol{y}}_l = f_\theta(\boldsymbol{x}_l)$ 作为测试数据的预测簇标签，其中 $\hat{\boldsymbol{y}}_l \in \mathbb{R}^C$。此时，$\hat{y}_l^c$ 代表第 l 个测试样例对第 c 个簇的归属程度。

3.3.2　模糊原型竞争学习框架

对所需要解决的问题进行拆解分析，可得到如下关键点：首先是该问题属于无监督聚类问题，那么采用非线性的竞争学习原型能够对数据分布进行表示，并将数据高密度聚集的区域划分为不同的数据簇；其次，该问题作为扩展后的模糊聚类问题，需要原型集合能够作为数据簇的支撑向量，即该原型时完全属于这个数据簇的，从而能够预测其他输入特征对于该数据簇的归属程度。同时，原型根据输入特征进行学习时，不同输入特征所对应的学习程度应该有所不同，因此学习权重中应该加入输入特征的归属程度值加以衡量。最后，模型接受在线增量的输入数据流，那么应当只有部分足够相似的激活原型能够随着输入特征而学习，避免输入特征对于旧原型所学习到的数据分布产生影响。

根据上述分析，本文提出如下**模糊原型竞争学习框架**：

$$\min_{\mathcal{P}} \left\| \boldsymbol{x}_t - \sum_{\mathcal{A}(\boldsymbol{x}_t)} r_j(\boldsymbol{x}_t)\boldsymbol{p}_j \right\|_2^2. \tag{3-1}$$

其中，$r_j(\boldsymbol{x}_t)$ 代表了输入特征 $\boldsymbol{x}_t$ 对于原型 $\boldsymbol{p}_j$ 的激活程度，$\mathcal{A}(\boldsymbol{x}_t)$ 则表示只有部分原型能够被输入特征激活从而进行学习，这也使得模型能够在增量数据流中保持稳定性-可塑性平衡。由于数据输入是在线方式，因此算法采用梯度下降的方式逐一对原型进行更新，其更新公式具体为

$$\boldsymbol{p}_j = \boldsymbol{p}_j + \alpha r_j(\boldsymbol{x}_t)(\boldsymbol{x}_t - \boldsymbol{p}_j). \tag{3-2}$$

其中 α 为原型进行学习时的学习率。

在提出上述原型学习框架的基础上，首先需要考虑如何进行输入特征的归属度程度计算，而该归属程度可以通过衡量输入特征与原型向量的相似程度得到。第 j 个原型可以用三元组 $\boldsymbol{n}_j =< \boldsymbol{p}_j, \boldsymbol{\sigma}_j, acc_j >$ 来描述，其中，$\boldsymbol{p}_j \in \mathbb{R}^d$、$\boldsymbol{\sigma}_j \in \mathbb{R}^d$ 以及 acc_j 分别代表原型向量，激活阈值和原型的累计激活次数，原型的维数 d 与输入向量相同。第 j 个神经元对于输入向量 $\boldsymbol{x}_t$ 的激活程度可以通过如下公式进行计算：

$$r_j(\boldsymbol{x}_t) = \exp\left(-\sum_{k}^{d} \frac{\left(x_t^k - p_j^k\right)^2}{2(\sigma_j^k)^2} \right). \tag{3-3}$$

该自组织网络可以根据上述激活程度与每个原型的阈值之间的数值关系来判定哪些原型被激活。具体来说，输入向量 $\boldsymbol{x}_t$ 所能够激活的原型集合可以通过如下判断公式得到：

$$\mathcal{N}(\boldsymbol{x}_t) = \left\{ \boldsymbol{p}_a \middle| r_a(\boldsymbol{x}_t) > r_a\left(\boldsymbol{p}_a + \sqrt{2}\boldsymbol{\sigma_a}\right) \right\}. \tag{3-4}$$

该式为原型划分了圆形的激活范围，即以原型向量为中心，半径为 $\sqrt{2}\boldsymbol{\sigma_a}$ 的范围边界上的点作为基准点，激活程度大于等于基准点激活程度的输入向量均可激活当前原型。

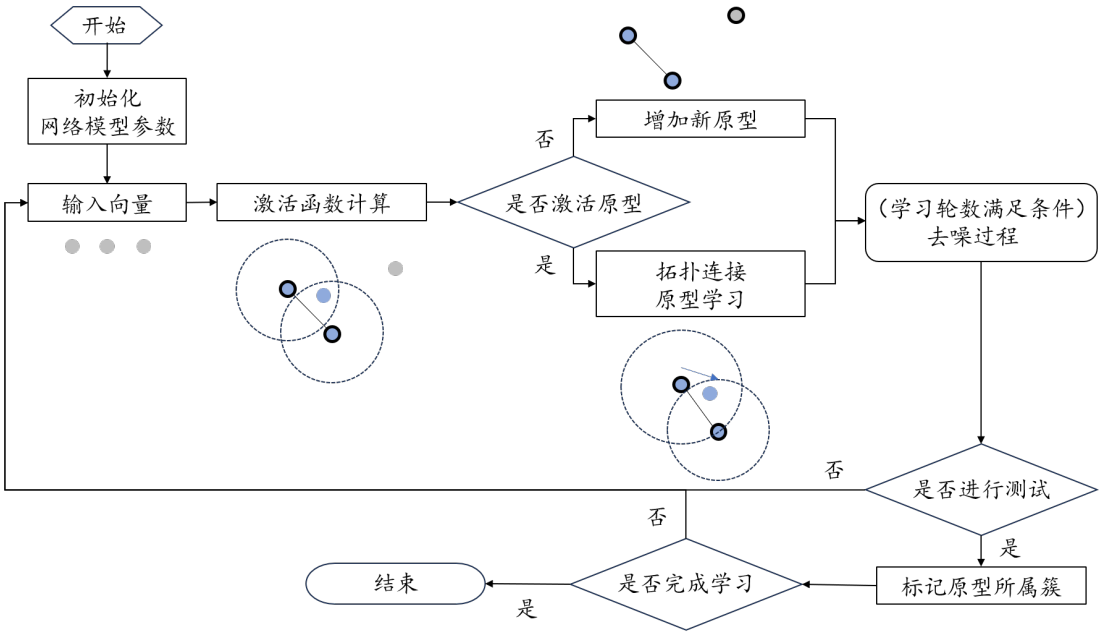


图 3-1　Fuzzy SOINN 学习流程总结。

根据上述原型学习框架，本章提出模糊自组织增量神经网络 Fuzzy SOINN 算法。该算法学习相互连接的原型节点用于对输入数据进行拓扑表示。网络模型包含两个集合，即原型集合 $\mathcal{P}$ 与连接集合 $\mathcal{C}$。初始化阶段，网络模型的原型与连接均初始化为空集，并通常采用首次输入的两个输入数据来得到最初的两个原型节点。在学习过程中，模型将根据输入分布的情况来对神经元节点及其连接进行增加或者删减。在原型的激活方式上，本章工作采用了软竞争方式，获胜原型与其连接原型都能够进行学习。整体学习流程如流程图3-1所示。

输入特征输入模型后，首先计算它对于神经网络中原型的归属度函数，并根据该归属度以及原型的激活阈值判断是否能够激活足够多的原型进行学习。若没有激活，则将输入数据作为新的原型加入模糊网络的原型集合中，否则对激活的原型及其拓扑结构进行学习。如果学习轮数进行了 λ 次，则进行原型节点的去噪过程。若需要对测试数据进行预测，则网络对原型进行标签标记。如果还有新的数据需要进行学习，则返回第一个步骤继续对下个训练数据进行学习。

3.3.3　原型学习

当通过公式（3-4）激活的原型数量小于两个时，则说明当前输入向量可能代表着未学习到的新模式或是需要对当前聚类的边界加以拓展。除此之外，模糊

网络的初始化中也需要将前两个输入向量直接以新原型的方式加入网络。可将原型初始化的方式总结为：

$$\boldsymbol{n}_{new} =< \boldsymbol{p}_{new} = \boldsymbol{x}_t, \boldsymbol{\sigma}_{new} = \boldsymbol{\sigma}_0, acc_{new} = 0 > . \tag{3-5}$$

如果是对网络进行初始化，那么 $\boldsymbol{\sigma}_0$ 将被初始化为用户设定的参数。否则，$\boldsymbol{\sigma}_0$ 将被设定为距离输入向量欧式距离最小的原型与当前输入向量的坐标差值，即 $\boldsymbol{\sigma}_0^k = \left|p_{nearest}^k - x_t^k\right|$。由于需要对两个激活程度最高的原型进行拓扑连接，因此激活的神经元节点必须有两个或以上，才能够对原型进行激活学习。

在激活集合为非空集的情况下，则需要对原型向量及拓扑连接进行更新。获胜原型即为当前输入向量激活程度最高的原型，即 $n_w = \arg\max_{n_i \in N} r_i(\boldsymbol{x})$。除去获胜原型需要根据原型学习的公式（3-2）进行更新以外，由于邻居节点是与激活原型共同激活的相关原型，因此它们也同样按照公式（3-2）进行学习。具体来说，邻居原型的学习幅度为获胜原型的 $\frac{1}{100}$，因此最终进行学习的原型集合 $\mathcal{A}(\boldsymbol{x}_t)$ 中包含了获胜原型与其邻居原型。同时，获胜节点的激活范围以及类似激活次数激活次数可以根据输入样例进行更新为：

$$\begin{aligned} acc_w &= acc_w + 1, \\ \sigma_w^k &= \sqrt{(\sigma_w^k)^2 + \frac{1}{acc_w^* + 1}(x_t^k - (p_w^k))^2}. \end{aligned} \tag{3-6}$$

上式为激活原型增加了累积激活次数参数，并且将其激活范围逐渐扩大。在具体实验时，一开始的初始化参数将设置得较小，使得其激活范围能够进行一定程度上的扩大调整。需要注意的是，模型中将获胜原型激活次数的倒数作为学习率来进行节点向量的更新。因此，原型向量的学习率随时间变化的序列为调和级数序列。文献[118] 提到，在线学习能够通过保证学习率的收敛性来对批量学习的迭代过程进行模拟，而上述学习率则能够满足该论文提出的三个条件，因此作为学习率是适合的。

在竞争学习中，同时被激活的原型之间会建立连接，因此网络模型将获胜原型与次获胜原型进行连接以表明它们可能描述了相似的特征。次获胜原型的确定方式与获胜原型相同，是在除去获胜原型的原型集合中寻找输入向量的归

属度最高的原型结点。原型之间的连接通过参数 $age_{j1,j2}$ 来描述，该参数表示该连接的两个节点原型 n_{j1} 与节点 n_{j2} 作为获胜结点和次获胜结点激活的次数。如果这条连接已经存在并且两个端点分别是这次输入向量所激活的获胜结点和次获胜结点，$age_{j1,j2}$ 会被重置为 0；所有与获胜节点相互连接但未被激活的节点则表明这两个原型存在并不是相关原型的可能，所以它们与获胜原型之间连接的年龄参数 $age_{j1,j2}$ 加 1。当连接的 $age_{j1,j2}$ 超过了算法规定的阈值 age_{max} 的时候，这条连接将会被删除。考虑到两个原型在更新后可能会偏离原始的位置，则此时两个原型无法同时激活，因此它们则不再相互连接。

3.3.4　去噪过程

去噪过程建立在以下启发式假设之上，即噪声周围的输入向量密度应当低于正常点附近的输入向量密度，因此若某个原型是由噪声向量产生，其被激活的次数应该远低于所有原型的平均值。具体去噪操作可描述为：在模型进行 λ 学习步后的学习之后，将删除一些激活次数远小于其余原型的原型。在本章中，噪声被定义为如下的原型集合：

$$\mathcal{N}_{noise} = \left\{ n_o \middle| acc_o < 0.1 \cdot \frac{\sum_j^{|N|} acc_j}{|N|} \right\}. \tag{3-7}$$

3.3.5　解决模糊聚类问题

当模型需要对测试向量进行预测时，需要对原型所属的簇进行划分。由于本章工作认为所学习的原型都是簇的支撑原型，因此可以看作其对于本身的簇归属程度为 1，其余簇的归属程度为 0。根据学习阶段的拓扑连接方式，其只有在两个原型被共同激活时才能够进行连接，因此可以认为这两个原型属于同一个簇。因此，原型的簇标签可以通过标签传播算法得到。首先，随机选取一个原型并对其标记一个新的簇标签，然后将该标签传播给所有与其连接的原型。当所有原型的标签都被标记时就完成了簇的划分；否则，重新选择一个无标签的原型重复上述操作。上述原型聚类过程总结在算法3.1中。若将原型集合及其连接看作图结构，则可以认为聚类过程就是将每个联通子图划分为一个簇。

当对测试数据进行簇预测时，输出方式依据具体的应用而决定。如果需要使

算法 3.1 原型聚类过程

输入： 未完全指定类标的原型集合，原型连接集合

1: 无论此原型是否带有之前学习得到过的类标记，将所有原型都初始化为未分类的
2: 选择原型集合中一个未标记的原型，为其分配一个类标
3: 使用递归方式将所有与其存在连接的原型分配相同类标，直到所有通过连接搜索到的原型都已经被标记了类标
4: 寻找下一个未标记的原型，若还存在，则返回第 2 步继续进行聚类；若不存在，则所有原型都已经被指定了聚类结果中的某一个类类标，算法结束，输出原型集合

输出： 指定聚类类标的原型集合

用硬聚类类型的输入样本聚类标签结果，则将输入样本划分为归属度函数最高的原型所属的类；如果需要得到软聚类的归属度矩阵，则计算输入样本到所有原型的激活程度，选取每个聚类中归属度值最高的原型所得到的归属度值，将其归一化，就是原型到每个聚类的归属度向量。

综上所述，算法3.2给出了本章模型的学习与预测的流程。

3.4 实验验证

3.4.1 人工数据集验证

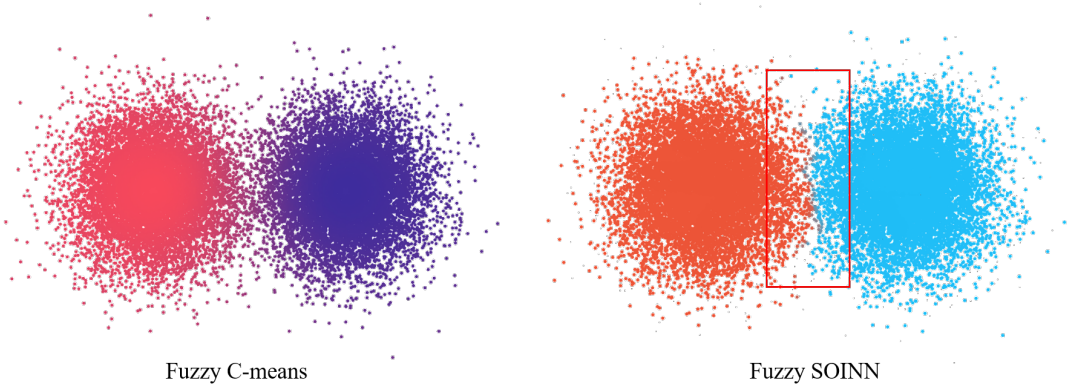


图 3-2 在高斯分布且线性可分的两个个数据分布上分别使用 fuzzy C-means 以及本章工作 Fuzzy SOINN 算法进行模糊聚类后，每个数据的归属度结果展示。两个数据簇分别用不同颜色进行表示，若归属度约趋于平均，则该数据特征的颜色约趋于两种颜色的平均。

本章设计了人工数据集用于展示模糊聚类的效果，并将本章工作与 Fuzzy C-means 算法进行比较。第一个实验从高斯分布的两个数据簇中产生 2 维的 20000 个数据样本，这两个类别的数据时线性可分的。两种算法对数据的模糊标签预测

算法 3.2 Fuzzy SOINN 算法的训练和预测流程

训练阶段:

输入：训练数据流 $\mathcal{S}_{train}$，用户定义参数 arg_{max}，λ，σ_0

1: 初始化原型集合 $\mathcal{P}$ 与连接集合 $\mathcal{C}$ 为空集
2: 输入向量 $\boldsymbol{x}_t$，如果这是网络的前两个输入特征，则直接使用其加入原型集合中，使用公式（3-5）进行初始化
3: 寻找激活原型集合。如果激活原型集合中原型数量小于 2，则将其通过公式（3-5）初始化后加入新的原型网络中，并继续学习下一个输入向量；否则，继续进行下一步学习
4: 对激活的原型通过公式（3-2）、（3-6）进行更新，并处理其之间的连接
5: 如果输入向量的数量是 λ 的倍数，则根据公式（3-7）寻找噪声原型进行删除
6: 若还有输入向量或算法未停止，则继续学习过程；否则，通过算法3.1 对节点进行聚类，停止训练

输出：包含原型集合 $\mathcal{P}$ 与连接集合 $\mathcal{C}$ 的网络模型

测试阶段:

输入：测试数据 $\boldsymbol{x}_l$

1: 使用公式（3-3）对测试数据 $\boldsymbol{x}_l$ 计算其对所有原型的归属程度
2: 根据归属程度确定预测标签 $\hat{\boldsymbol{y}}_l$

输出：预测标签 $\hat{\boldsymbol{y}}_l$

结果如图3-2所示。第二个实验从非高斯分布的三个数据簇中产生 2 维的 20000 个数据样本，两种算法对数据的模糊标签预测结果如图3-3所示。不同颜色代表数据对于聚类的归属程度相差较大。若归属两个类别的程度相差越小，则该数据样本的颜色更倾向于多种颜色的混合。例如在图3-2中红线所圈出的范围内，数据的颜色倾向于两个簇颜色的混合，这意味着其归属与两个簇的程度也相差不大。另一方面，本章工作的数据模糊程度比 Fuzzy C-means 低，因为因为其颜色在整体上更加纯粹，只有在重叠区域的颜色是渐变的。其原因主要是因为本章工作与 Fuzzy C-means 算法对于数据分布的假设不同。使用多个原型节点表示分布，因此只要处于该数据簇的 “势力” 范围内，则归属度将较高；而 Fuzzy C-means 只使用均值原型节点，因此只要远离均值原型节点的数据对于该数据簇的归属度就会下降。另一方面，可以看出在非线性可分的人工数据集上，Fuzzy C-means 对于 A、C 的两个数据簇不能够进行很好的表示。从直观视角考虑，以 C 数据簇为例，那么应该是大约处于如图所示 “均值线” 附近的数据归属度最高，越位于两侧的数据归属度越低。然而从 Fuzzy C-means 的实验结果来看，所画出的三个矩阵框内，位于均值线附近的数据颜色并不相同，反而是位于中间矩形框内的数

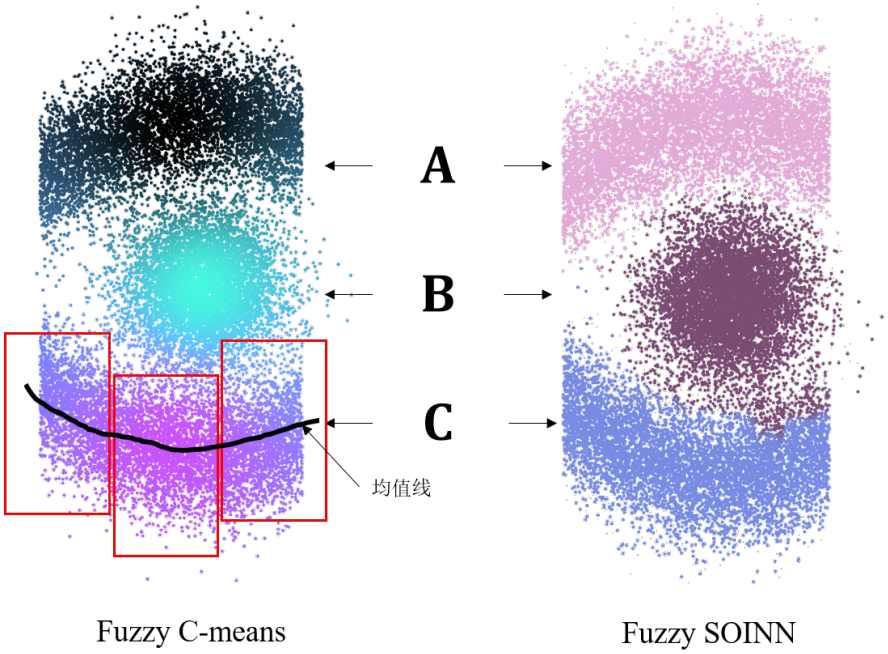
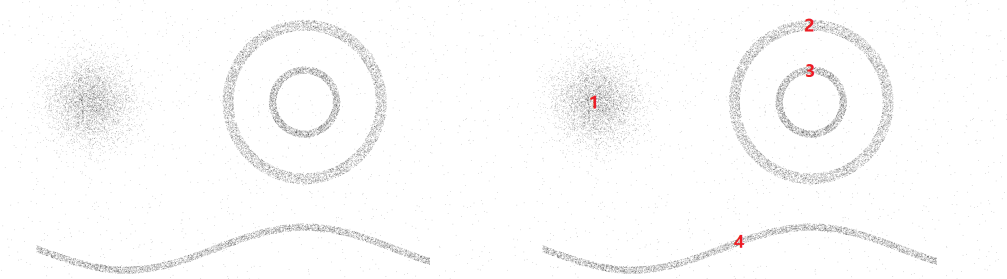


图 3-3 在非高斯分布且线性不可分的三个数据分布 A，B，C 上分别使用 fuzzy C-means 以及 fuzzy SOINN 算法进行模糊聚类后，每个数据的归属度结果展示。三个数据簇分别用不同颜色进行表示，若归属度约趋于平均，则该数据特征的颜色约趋于三种颜色的平均。

据归属度更高。因此，使用均值并不能很好地对非高斯分布的数据进行表示。

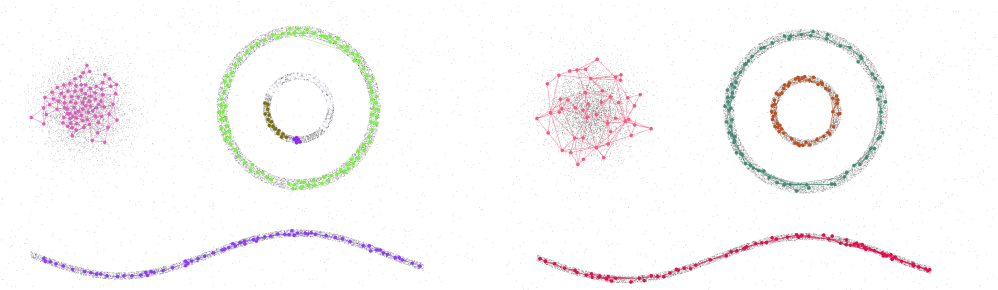


(a) 在线学习设置。

(b) 在线增量学习设置，数字表明类别输入的顺序。

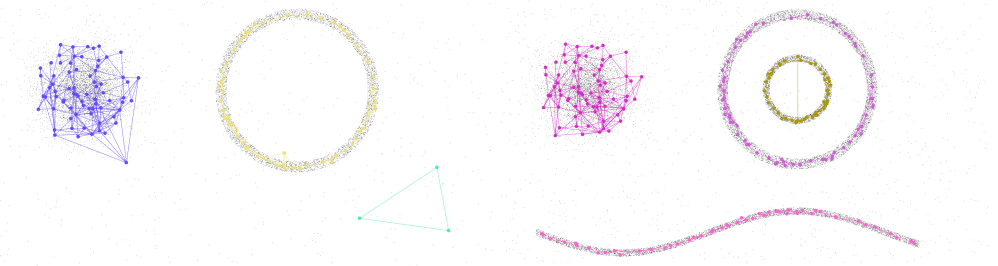
图 3-4 人工数据集的数据分布，包含高斯分布、圆环分布、以及 sin 型分布，数据集共包含 20000 个样本，以两种学习设置输入模型。

可以看出，本章工作在非线性可分、非高斯分布的数据集上也能够进行学习，因此本章采用多个人工数据分布的组合来展示本章工作所学习到的原型网络分布。该数据集包含 20000 个输入样本，噪声数据占据总数据的 5% 左右，并随机分布在输入数据的坐标范围中。在线学习实验中，数据按照随机的顺序输入进行学习；而在线增量实验还考察了本章工作在增量学习环境中的学习能力，即每次只输入一个类别的数据，输入完后再进行下一个类别数据的学习。上述两



(a) SOINN 算法学习结果。　　(b) 本章工作学习结果。

图 3-5　在线实验设置上 SOINN 算法与本章工作的模型训练完后原型分布情况。



(a) 学习完两个类别数据的原型学习结果。　　(b) 完成所有数据训练的原型学习结果。

图 3-6　在线增量实验设置上 SOINN 算法在不同阶段的原型学习结果展示，可以看出增量学习环境中本章工作在扩展原型集合时对已学习的原型知识保存较好。

个实验的数据设置在图3-4中进行表示。

在线设置中，本章工作与 SOINN 算法进行了原型结果的比较，结果如图3-5所示。可以看出，SOINN 和本章工作中的原型及其拓扑结构基本都能够对数据集进行表示，SOINN 在双圆环数据集上的表示较差，无法对位于内圆的数据进行原型的拓扑表示。同时，在原型的分布上，本章工作相较于 SOINN 网络模型也有一定的区别，SOINN 学习过程中使得原型保持在 Voronoi 三角的顶端，因此形成的拓扑结构较为规整；而本章工作采用了模糊归属程度的计算方式，使得原型不再通过拓扑结构保持 Voronoi 三角结构。该表示方式使得原型的表示结果较为杂乱，但同样能够作为模糊集合的支撑向量表示数据分布。从高斯分布数据可以看出，本章工作对于模糊集合的边缘数据也能够进行表示。也就是说作为支撑原型，本章工作所学习的数据分布相较于 Fuzzy C-means 以及 SOINN 更广，在计算归属程度时该项是优势还是劣势，由于模糊聚类指标的欠缺性，目前还暂时无法下定结论。增量学习设置中，图3-6展示了本章工作的学习中间结果以及最终学习结果。首先可以看出，中间结果中学习得到的原型分布与最终结果相差

表 3-1 聚类实验中所使用的 8 个数据集细节，该表格展示了数据集的特征维数、数据集样本数量以及类别个数。在线聚类实验设置中，训练集与测试集合并参与网络模型的训练与测试阶段。

序号	数据集	# 特征维数	# 样本	# 类别
1	Iris	4	150	3
2	Glass	10	214	6
3	Segment	19	2310	7
4	Wine	13	178	3
5	Letter	16	20000	26
6	USPS	256	9298	10
7	isolet	617	7797	26
8	COIL	1024	7200	100

不大，甚至表示高斯分布数据的原型中还除去了学习结果较差的原型，说明本章工作对于已学习的知识保存较好，甚至还能够在后续学习中对已学习的错误知识进行纠正。但也可以看到，增量学习得到的原型分布不如在线学习得到的结果好，例如大圆环数据上段对应的原型有所欠缺，以及 sin 分布数据右端对应的原型也较少，其次小圆环对应的原型也出现了连接两个相距较远的原型的情况。总之，本章工作无论在在线学习还是在线增量学习的环境中都能够得到描述数据分布的原型集合及其拓扑结构。

3.4.2 真实数据集实验设置

本次实验中所采用的 8 个数据集细节总结在表3-1中，除 COIL 数据集外其他数据集都来自于 UCI 数据库[125]。其中，Iris 数据集是早期用于模式识别的常用数据集，其包含共 150 个三种鸢尾花的花萼及花瓣的长宽数值，即共 4 维特征。Glass 数据集包含玻璃 10 种化学元素成分含量作为特征的 214 个数据，用于辨别该玻璃的来源（包括台灯、窗户、容器、车窗等）。Segment 数据集包含 7 个类别的图像数据，所有维度均是启发式方法手动提取的图像特征。Wine 数据集对酒的 13 种成分进行描述，并据此来对 3 种酒进行分类。Letter 数据集包含多种字体下的 26 个字母的两万张黑白图像，并经过特征工程提取得到 16 个属性组成每个输入样例，来源于多种字体，并经过随机化的旋转，标签为该图像所对应的英文字母。isolet 数据集包含来自 150 位受试者对 26 个英文字母进行朗读的录音文件，其特征包含轮廓特征、声母特征、前声母特征和后声母特征共 26 维特征。COIL 为黑色背景下共 100 种物体的 32×32 灰度图像数据集[126]。

本文共选取了四个模糊聚类以及相关的增量学习算法与本章工作在真实数据集上进行对比，包括 FCM、Online FCM（OFCM）以及 RL-FCM[127]。同时还包括 SOINN 相关算法，即进行一层学习的 SOINN 算法，增强 SOINN 算法（ESOINN)。最初的 SOINN 算法在进行第一层次聚类后还会对学习到的节点进行第二次学习，考虑到在线学习的特征以及 Fuzzy SOINN 的学习流程中也只有一层，因此实验中使用单层 SOINN 进行测试。ESOINN 能够对于低密度区域学习的原型进行分离，防止两个簇由于距离太近而被连接起来成为一个数据簇。

在实验中按照如下方式设置本章工作所使用的三个参数 σ_0，age_{max}，λ：在数据集 letter、isolet、以及 COIL 上，$\sigma_0 = 0.1$；在 Segment 上 $\sigma_0 = 1.0$，且其他数据集上都设置为 $\sigma_0 = 0.5$。在所有数据集上都设置 $age_{max} = 20$。λ 参数的设置与数据集样本的数量正相关，该参数控制着删除节点快慢，因此在复杂数据集上应该给予模型更多的学习时间，而在小型数据集上数据在更少的训练时长上就能够学习到数据的分布。因此，在 USPS、Letter、isolet 以及 COIL 数据集上设置 $\lambda = 1000$，在 Iris 以及 Wine 数据集上设置 $\lambda = 100$，且其他数据集上设置 $\lambda = 200$。对比算法 SOINN 以及 ESOINN 的结果均来源于 ESOINN 论文[128]。对于 SOINN 类型的在线学习算法，输入数据的顺序将影响算法的学习效果，因此最终学习结果将被记录为 10 次不同输入顺序得到的结果的均值。

在本次实验中采取标准化互信息（Normalized Mutual Information，NMI）对结果进行衡量。该指标常用于衡量聚类结果的质量。该指标的结果与标签的绝对值无关，因此可以体现出标签相对值之间的关系。由于该指标衡量的是两种标签指定的相似程度，因此该指标越大越好，且取值范围通常在 $[0, 1]$ 之间。NMI 的计算可以通过 python 中的 scikit-learn 计算包实现[129]。

3.4.3　实验结果分析

本章工作与其他基线算法在 8 个数据集上的比较结果总结在表3-2中。从总体表现来看，本章工作在 Segment、Wine 以及 isolet 数据集上的 NMI 指标优于所有方法。对比 FCM 类型方法，对于经典算法 FCM，本章工作在 Glass、Segment、Wine、isolet、COIL 这 5 个数据集上优于该方法。OFCM 则是 FCM 的在线版本，因此其结果相比 FCM 更差。相比较新提出的 RL-FCM 方法，本章工作在 Segment、Wine、isolet、以及 COIL 四个数据集上的 NMI 指标优于该方法。与在

表 3-2 本章工作 Fuzzy SOINN 与其他对比方法在 8 个真实数据集上进行 10 次运行的 NMI 均值结果，其中加粗结果为当前数据集下的各个方法中的最优结果，当结果右上角被 * 标记时代表该结果来自于其他论文。

数据集	Iris	Glass	Segment	Wine	Letter	USPS	isolet	COIL
FCM	0.750	0.360	0.515	0.417	0.344	0.450	0.634	0.478
OFCM	0.530	0.295	0.488	0.402	0.234	0.197	0.356	0.368
RL-FCM	**0.792**	0.570	0.618	0.430	**0.398**	0.500	0.635	0.492
SOINN*	0.554	**0.584**	0.382	0.276	0.229	0.507	0.635	**0.607**
ESOINN*	0.633	0.515	0.406	0.260	0.376	**0.607**	0.662	0.513
Fuzzy SOINN	0.701	0.425	**0.624**	**0.456**	0.291	0.325	**0.675**	0.522
原型数量	48	67	900	55	792	1025	1378	1890

线增量的 SOINN 类算法相比，本章工作在 Iris、Segment、Wine、Letter、isolet 五个数据集上的聚类性能优于 SOINN 算法，并据此验证了本章所使用模糊原型网络的有效性。与对 SOINN 进行改进的 ESOINN 相比，也同样在五个数据集上具有优势。因此，本章工作无论是相较于改进前的 SOINN 算法还是在线模糊聚类算法都具有一定优势。此外，由于本章工作的时间、空间复杂度与原型数量相关，因此表格中展示了本章工作在不同数据集上学习到的原型数量结果。可以看出，随着数据集规模的逐渐增加，该原型数量也随之逐渐增加，因此会占用更多的时间、空间资源。当训练资源有限时，可以通过设定原型数量上限，并根据激活情况删除不活跃原型来控制算法的训练消耗，但势必会影响算法最终的聚类性能。

但是，本章工作在部分数据集上聚类性能仍有改进的空间。例如在 USPS、COIL 等较为高维的数据集上，本章工作通过归属度函数进行激活时，由于该归属度函数属于幂函数，且会随着维数上升而容易迅速减少，因此使得两个高维的原型难以被共同激活，从而会产生较多的聚类数量。使用 σ_0 对激活范围进行扩大则可能使不同簇的原型也被连接起来，因为簇之间存在模糊的数据，难以对数据进行区分。从实验结果中的原型数量也可以看出，随着数据规模的增加，逐渐需要更多原型向量才能够对高维数据进行表示。

3.4.4 参数敏感性分析

在这节中对本节算法所用参数 σ_0，age_{max}，λ 进行了敏感性分析，结果展示在表格3-3中。其中部分数据集的训练轮数不足当前 λ 参数，因此不进行该设置下的实验。并且算法结束时如果不是恰好进入去噪过程，则额外增加一次去噪过

表 3-3　本章工作在不同参数设置下的 8 个数据集上的 NMI 结果。

参数	设置	Iris	Glass	Segment	Wine	Letter	USPS	isolet	COIL
σ_0	0.05	0.60	0.32	0.47	0.37	0.27	0.25	0.67	0.50
	0.1	0.65	0.40	0.55	0.43	0.29	0.31	0.68	0.52
	0.5	0.70	0.43	0.58	0.46	0.26	0.33	0.64	0.48
	1.0	0.69	0.41	0.62	0.43	0.21	0.32	0.60	0.42
	2.0	0.57	0.37	0.60	0.32	0.19	0.20	0.62	0.44
age_{max}	10	0.69	0.43	0.61	0.44	0.29	0.33	0.67	0.51
	20	0.70	0.43	0.62	0.46	0.29	0.33	0.68	0.52
	50	0.68	0.43	0.60	0.43	0.28	0.31	0.68	0.48
	100	0.65	0.39	0.59	0.41	0.23	0.29	0.65	0.48
λ	50	0.69	0.40	0.55	0.42	0.20	0.23	0.63	0.43
	100	0.70	0.42	0.58	0.46	0.22	0.25	0.63	0.45
	200	0.67	0.43	0.62	0.44	0.25	0.29	0.64	0.47
	500	—	0.40	0.61	—	0.26	0.30	0.65	0.51
	1000	—	—	0.60	—	0.29	0.33	0.68	0.52
	2000	—	—	0.57	—	0.28	0.31	0.67	0.49

程。从结果可以看出，不同参数设置对于结果都有一定的影响。其中 σ_0 参数的设置对于 NMI 结果的影响是最大的，考虑到该参数控制了原型的激活范围，直接影响到原型的学习结果，因此原型学习结果对于该参数是十分敏感的。例如在 Letter 数据集上，$\sigma_0 = 2.0$ 时 NMI 指标从 0.29 下降到了 0.19；在 USPS 数据集上，$\sigma_0 = 2.0$ 时 NMI 指标从 0.33 下降到了 0.20，这些都是无法接受的聚类性能下降。因此，在实验中需要关注对于 σ_0 的设置。参数 age_{max} 与 λ 对于结果的影响则没有 σ_0 参数那么大，甚至部分 age_{max} 参数的结果不影响最终的聚类性能。除此之外也可以看出，参数设置基本都有合理的范围，当距离最佳参数设置较近时，参数的设置变化对算法性能的影响要小于超出该范围后对算法的影响。

综上所述，本章工作对于参数 σ_0 十分敏感，需要通过多次实验找到最佳参数；对于 age_{max}，λ 参数则可基本保持稳定的算法性能，在合理范围内的设置对算法的聚类结果影响不大。

3.5　小结

为解决在线对聚类归属程度进行预测的模糊聚类问题，本文提出了模糊原型竞争学习框架，并在此基础上提出了解决模糊聚类问题的模糊自组织增量神经网络。其思路主要是为聚类分布学习原型网络对数据分布进行表示，并且预测

输入特征的归属程度。在原型学习的基础上，本文采用的自组织网络结构为原型网络提供了拓扑结构以及共同激活的学习方式，使得网络能够自主对簇中的支撑原型进行划分。同时，网络模型中的新增原型与去噪功能使得网络在在线环境中也能够自主增加或删除原型向量，适应数据分布的变化，不需要被人工设定的聚类数量参数所束缚。在实验中，分别通过人工数据集以及实际数据集对算法的表现进行了评估。人工数据集可视化地展示了本章工作模糊聚类的结果，而真实数据集则通过聚类结果指标展示了本章工作在聚类上的性能。

本章工作只是对模糊聚类问题上初步的算法探索，其还有很大的改进空间。从算法思路上进行考虑，本章工作中将所有原型视为模糊集合的支撑原型，即所有原型都是完全属于该模糊集合的。但所有原型其实都从某种程度上描述了这个模糊集合，但是却都不能描述完全，这可能更符合人类直观的感受。因此，如何在学习支撑原型的同时确定其对于当前簇的描述程度，可以进一步在算法中进行考虑。从实验结果上进行考虑，在一部分数据集上不甚令人满意的结果表示了本章工作还有很大的改进空间，特别是对于高维数据的处理。如何改进算法使得其实验性能进一步改进是主要的探索方向。同时，如何采用合适的模糊聚类算法指标对算法进行考量也是可以进行探索的方向。

第四章　基于非负原型线性学习框架的标签分布学习算法

4.1　引言

考虑到不同标签对应性质对于输入特征的不同描述程度，标签分布学习模型为输入特征预测所有标签对其的归属程度，可以视为是对单标签与多标签学习问题的扩展。不同标签之间的关联关系，也使得该问题不能简单使用多个独立的回归模型进行解决。已有的标签分布算法大多考虑直接使用模型拟合标签分布的数值，并且在分类时考虑标签之间的关联性，从而将其作为一个特殊的分类问题进行解决，但对于输入特征空间是否能够进行转换而提升分类性能则研究较少。本章工作则从原型学习框架来考虑该问题，即采用原型基底来将标签具现化为向量形式，输入特征则可以表示为原型向量之间的线性组合，那么仍然可以使用重构误差的原型学习框架来作为训练目标。除此之外，将输入特征转换为系数向量起到了特征提取的作用，使得分类模型在更容易进行分类的嵌入空间中进行学习。

从上述角度出发，本章节提出非负原型线性学习框架，并结合最大熵分类模型提出非负原型基底学习方法来解决标签分布问题。使用原型学习框架学习到标签对应的原型向量表示，并据此将数据转换到原型表示的嵌入空间中。在此基础上，采用分类模型对嵌入特征进行标签分布的拟合，并最终完成对输入数据标签分布的预测。实验章节中采用了多种衡量分布相似度的评判指标，验证了原型学习框架及本章工作的有效性。

4.2　研究背景

从标签分布学习问题的提出[25] 开始，该问题受到了越来越多研究者们的关注。由于该问题可视为对分类问题的扩展，因此将该问题转换为分类问题进行解决是最初研究者们的思路。在此基础上，研究者们对已有的多标签分类或是分类问题的算法进行了改进，相较于单纯使用分类模型进行解决提升了算法的性能。

最终，研究者们提出了专用的标签预测模型用于解决该问题，使得标签分布问题能够进一步得到解决。遵循上述发展流程，对标签分布算法的概述可以根据算法提出思路分为三种类型，分别包括问题转换、改进传统标签预测算法以及提出针对性算法。

当希望完全使用已有分类算法对标签分布问题进行解决时，可以将该问题转化成为单标签的预测问题。例如使用支持向量机的方法 PT-SVM[130] 和 LDLM[131]，其主要思想是将每个实例及其标签分布转化成为多个带有单个标签的相同特征，因此模型每次只用预测单个标签对于该特征的置信度（confidence）或是概率（probability）。可以看出，这种方法会忽略标签之间的关联性，使用模型独立地对每个标签进行预测，同时论文[25] 中的结果也证明了该类算法的表现差于其他类型的标签分布算法。

其次是将传统的单标签或多标签问题的解决方案改造成为标签分布学习问题的解决方案，例如使用 k-最近邻的 AA-kNN[132] 和使用反向传播（backpropagation，BP）进行训练的三层神经网络 AA-BP[133]。AA-kNN 延续了 k-最近邻算法的懒惰学习（lazy learning）思想，即在预测时，对 k 个距离最近的特征的分布进行平均即为待预测特征的标签分布。而对于 BP 神经网络，则是将原本的 one-hot 形式的编码扩展为使用 softmax 函数进行归一化过的输出，并通过降低网络输出与真实标签分布的平方和损失来训练网络。

除上述算法外，也有多个专门为标签分布学习问题设计的算法。例如 IIS-LD[133] 和 BFGS-LD[25] 提出使用不同优化算法训练的最大熵模型（maximum entropy model）解决年龄预测问题。后续研究者们在上述最大熵模型的基础上引入了对标签或者代表特征之间相关性的限制[134][135][136]。LALOT[137] 使用最优传输理论（optimal transport theory）替代 KL 散度来表示真实标签分布与预测标签分布之间的差异。Duo-LDL[138] 同样考虑了标签之间的关联性，因此使用三层 MLP 神经网络并将网络输出设置为不同标签之间的分布差异。对于更为复杂的问题，深度神经网络也被使用进行年龄估计[139][140]、情绪分析[141][142]、面部姿势估计[143][144] 和视频分析[145] 等实际应用场景。

然而，以前的工作大多是直接从原始数据中学习分类模型，而没有借助对原始数据特征进行线性变换来将数据转换为嵌入空间特征进行学习的方法。因此，本章提出非负原型基底学习（Nonnegative Prototype Basis learning，NPB）算法用

于解决标签分布学习问题。该方法将原始数据表示为原型之间的重构系数，从而使得分类模型在更容易进行学习的空间中进行训练，从而得到更好的预测结果。考虑到原型表示了标签的具体性质，具有实际物理含义，因此在学习过程中限制原型非负。对于本章工作的具体贡献，可总结如下：

- 提出非负原型学习框架对于标签所对应的性质进行向量表示，使得输入特征能够根据其与不同性质的符合程度表示为原型向量的线性组合，也能够将输入特征转换为系数向量进行特征提取；
- 结合最大熵分类模型形成非负原型基底学习方法 NPB，从而实现对输入数据的标签分布预测；
- 实验章节中通过公开数据集上的多个指标评估结果验证了本章工作在聚类问题与模糊聚类问题上的有效性。

4.3 本章工作

4.3.1 标签分布问题定义

标签分布问题的定义遵循绪论中给出的标签预测问题定义1.1。具体来说，模型在训练时获得训练数据集 $\mathcal{D}_{train}=\{\mathbf{X}_{train},\mathbf{Y}_{train}\}$，并且 $\mathbf{X}_{train}=[x_i^k]_{N\times d}$ 表示训练集中共有 N 个 d 维数据，$\mathbf{Y}_{train}=[y_i^c]_{N\times C}$ 表示训练集的 N 个样本对于 C 个类别形成的类别标签分布。此时，y_i^c 代表第 c 个标签对于第 i 个样本的描述程度。由于规定所有标签对同一个输入样本的描述程度 $\boldsymbol{y}_i$ 为一个分布，因此数值上 $y_i^c\in[0,1]$，且 $\sum_j y_i^c=1$。

标签分布模型模型的学习目标是学习从输入空间到输出空间的映射 $\theta:\mathbb{R}^d\rightarrow\mathbb{R}^C$。在评估模型时，测试数据集 $\mathcal{D}_{test}=\{\mathbf{X}_{test},\mathbf{Y}_{test}\}$ 向模型输入特征矩阵 $\mathbf{X}_{test}=[x_l^k]_{l\times d}$。对于测试数据特征 $\boldsymbol{x}_l$，模型的输出 $\hat{\boldsymbol{y}}_l=f_\theta(\boldsymbol{x}_l)$，该输出应与评估数据集的标签 $\boldsymbol{y}_l$ 尽可能的接近。因此，对预测结果的表现评估基本采用不同的距离度量函数或相似度度量函数来比较预测标签分布矩阵与真实标签分布矩阵之间的差别。在实验章节中将介绍本章工作中所采用的具体度量函数的计算方式。

4.3.2　非负原型线性学习框架

可以看出，标签分布学习问题与模糊聚类问题类似，都需要预测输入特征对所有数据模式的归属程度。相比于模糊聚类问题，标签分布问题的优势在于输入特征对于每个类别的归属程度是已知的，因此可以据此学习出每个类别的支撑原型。那么首先可以考虑是否能够将模糊聚类方式应用于标签分布学习算法，而经过分析和实验就可以知道该方法是不可行的。无监督模型本身对于数据分布有如下先验假设：数据不是均匀分布在特征空间的，其会在部分空间中聚集成高密度区域，而其他特征空间区域的输入特征分布较少。但实际生活中可能存在数据较为均匀地分布，特别是标签分布问题下的数据尤甚，其平滑地在特征空间中移动就对应了标签不同的描述程度。在进行算法设计前，采用 FCM 对标签分布问题下的公开数据集训练数据进行学习，发现最终只能够学习到一个数据簇。综上考虑，在进行原型学习框架进行学习时必须引入标签分布的信息，才能够学习到合理的原型集合。本节提出采用单个对应原型基底来表示标签所对应的性质，并在此基础上进行重构误差函数的训练。考虑到使该原型向量具有一定的可解释性，其每个维度能够对应现实世界对事物的数值描述，因此本文学习到非负的原型基底。例如，图像数据每个位置的数据描述了这个坐标的像素值，因此通常对应一个范围在 0-1 之间或 0-255 之间的数值。而学习到对应标签的原型基底则可以可视化得到该标签所代表的图像向量。因此，在优化求解过程中，原型向量 $\boldsymbol{p}_j \geqslant 0$。对于第 i 个输入向量，第 c 个标签对其的描述程度为 y_i^c。若需要原型向量能够完全代表第 c 项特征，也即第 c 个标签能够完全描述该原型向量，那么其在所有类标上的标签分布应当是仅有第 c 个标签的描述程度为 1 且其余标签的描述程度都为 0。此时，其它实例特征可以使用原型向量的线性组合进行近似表示，也就是可以通过原型向量重构实例特征。可以认为，每个标签对于实例特征的描述程度可以认为是原型向量在重构时所占的权重。因此，改进后的原型学习框架将原重构误差函数中的重构系数从需要进行学习的参数变为已知的训练标签分布，那么实例特征 $\boldsymbol{x}_i$ 通过原型向量进行重构的公式可以写为：

$$\boldsymbol{x}_i = \sum_{j=1}^{K} y_i^j \boldsymbol{p}_j. \tag{4-1}$$

由于采用了每个标签与原型一一对应的学习方式，因此原型数量 K 与标签数量 C 一致。在对于该标签分布算法的论述中，对于原型相关的序号将采用 j 进行，对于标签相关的序号将采用 c 进行。

根据上述分析，本文提出如下**非负原型线性学习框架：**

$$\min_{\mathcal{P}} \left\| \boldsymbol{x}_i - \sum_{j=1}^{K} y_i^j \boldsymbol{p}_j \right\|_2^2, \quad \text{s.t. } \boldsymbol{p} \geqslant \mathbf{0}. \tag{4-2}$$

当所有训练数据同时进行训练时，可将所有数据集合表示成矩阵的形式。此时，原型集合也可以表示成矩阵 $\mathbf{P} = [p_j^k]_{C \times d}$，因此式（4-2）可写为降低重构误差矩阵的 F 范数，即：

$$\min_{\mathbf{P}} \|\mathbf{X}_{train} - \mathbf{Y}_{train}\mathbf{P}\|_F^2, \quad \text{s.t. } \mathbf{P} \geqslant \mathbf{0}. \tag{4-3}$$

其中 d 是输入特征的维数，C 是监督标签的数量。

基于上述原型学习框架，本章提出的 NPB 算法学习过程由图4-1进行描述。每个标签对应了数据中的不同特征，因此可以为每个标签类别学习一个代表其特征的原型向量，而数据则由描述这些标签的特征向量进行混合组成。此时，输入数据将从原特征空间转换到由每个标签组成的特征空间。此时，输入特征可以转换为原型表示的系数向量，并采用系数组成的系数矩阵进行分类模型的训练，从而对测试样本的标签分布进行预测。

4.3.3 线性表示

对原型集合进行学习后，可以将特征矩阵从原始特征空间转换到原型矩阵所形成的基底空间。考虑到原型矩阵重建时的误差，真实特征矩阵转化为的系数矩阵通常与真实标签分布有差异。从另一个角度考虑，系数矩阵是由原始特征空间降维至基底空间得到的嵌入特征，因此对系数矩阵使用标签分布预测模型能够得到更为准确的结果。在 $\mathbf{X}$ 和 $\mathbf{P}$ 固定的情况下，系数矩阵记为 $\mathbf{V} = [u_i^j]_{N \times C}$。考虑到减法也是自然的、可解释的，负数系数可以看作是从已有原型基底中减去该原型基底。因此在本章工作中放宽对重构系数的要求，使得系数的求解更为容易。综上，通过最小化重构误差将特征矩阵转化为基底空间中的系数矩阵 $\mathbf{V}$，其

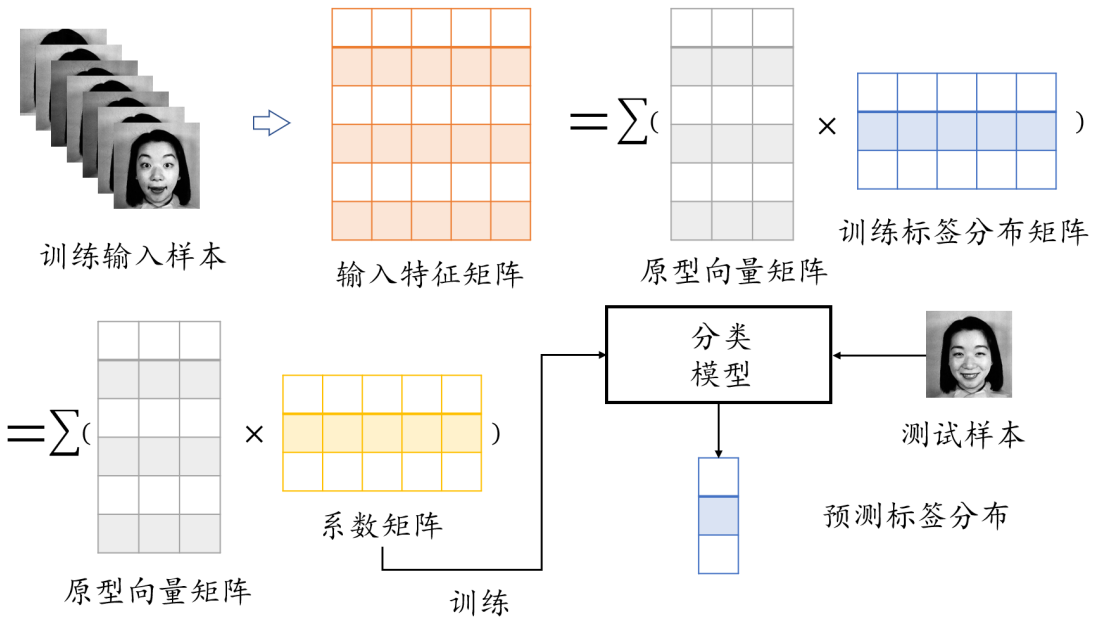


图 4-1　本章工作的学习流程。首先根据标签分布矩阵与输入特征矩阵求得标签集合所对应的原型基底矩阵，并据此得到输入特征在该原型矩阵上进行表示的系数矩阵。该系数矩阵将用于训练进行标签预测的分类模型，在预测时也通过原型集合表示的系数对测试样例的标签分布进行预测。

可通过最小化如下目标函数得到:

$$\min_{\mathbf{V}} ||\mathbf{X}-\mathbf{V}\mathbf{P}||_F^2. \tag{4-4}$$

这里的数据集 $\mathbf{X}$ 视计算场景而定，既可以在训练时使用训练特征 $\mathbf{X}_{train}$，又可以在测试时使用测试特征 $\mathbf{X}_{test}$。将训练特征矩阵 $\mathbf{X}_{train} \in \mathbb{R}^{N\times d}$ 的维数与系数矩阵 $\mathbf{V} \in \mathbb{R}^{N\times C}$ 的维数进行对比可以看出，当 $C < d$ 时，即当标签数量小于数据特征维度时，上述基底学习过程也可视为一种数据降维操作；否则，可以视为一种字典学习。

4.3.4　最大熵分类模型

通过原型基底的学习，输入特征从原始空间转换到了原型基底构成的表征空间，并通过系数表示为表征空间的向量。该空间的表示可以视为将输入向量表示成为每个标签所代表的模式的带权组合，对组合系数进行标签分布的预测相较于原始空间更为容易。使用最大熵模型 $p(\mathbf{Y}|\mathbf{V};\mathbf{W})$[25][135] 作为分类预测模型，其中 $\mathbf{W}=[w_c^j]_{C\times C}$。因此，对于训练标签分布中对第 i 个样本的第 c 个标签的预

测 $\widehat{y}_i^c$，可通过如下 Softmax 公式得到：

$$p(\widehat{y}_i^c|\boldsymbol{v}_i;\mathbf{W}) = \frac{\exp(\boldsymbol{w}_c\boldsymbol{v}_i^T)}{\sum_{c=1}^{C}\exp(\boldsymbol{w}_c\boldsymbol{v}_i^T)}. \tag{4-5}$$

为了构建学习分类模型的损失函数，本章模型采用 Kullback-Leibler（KL）散度作为距离度量，该度量常用于评估两个分布之间的距离，范围在 $[0,\infty]$ 之间，且越小表明两个分布越相似。因此，模型参数 **W** 可通过减少预测分布与真实分布之间的 KL 散度来优化，具体损失函数如公式 (4-6) 所示：

$$\begin{aligned}\mathcal{L}_{KL}(\mathbf{W}) &= \sum_{i=1}^{N}\sum_{c=1}^{C} y_i^c \ln\frac{y_i^c}{p(\widehat{y}_i^c|\boldsymbol{u_i};\mathbf{W})} \\ &= \sum_{i=1}^{N}\sum_{c=1}^{C} y_i^c \ln y_i^c - y_i^c p(\widehat{y}_i^c|\boldsymbol{u_i};\mathbf{W}).\end{aligned} \tag{4-6}$$

由于固定值对于模型的优化训练起不到作用，因此在具体训练过程中可以省去公式 (4-6) 中的固定值 $y_i^c \ln y_i^c$。对上述公式进行展开，可以得到最终进行优化的公式：

$$\begin{aligned}\mathcal{L}_{KL}(\mathbf{W}) &= \sum_{i=1}^{N}\sum_{c=1}^{C} y_i^c \ln\frac{\sum_{c=1}^{C}\exp(\boldsymbol{w}_c\boldsymbol{u}_i^T)}{\exp(\boldsymbol{w}_c\boldsymbol{u}_i^T)} \\ &= \sum_{i=1}^{N}\ln\sum_{c=1}^{C}\exp(\boldsymbol{w}_c\boldsymbol{u}_i^T) - \sum_{i=1}^{N}\sum_{c=1}^{C} y_i^c\boldsymbol{w}_c\boldsymbol{u}_i^T.\end{aligned} \tag{4-7}$$

其中 $\sum_c y_i^c = 1$，因此可以将前半部分公式合为 $\sum_{i=1}^{N}\ln\sum_{c=1}^{C}\exp(\boldsymbol{w}_c\boldsymbol{u}_i^T)$。

4.3.5 参数优化

本章工作中共有三个模型参数需要优化求解，即原型矩阵 **P**，原型表示系数矩阵 **V**，和最大熵模型的参数矩阵 **W**。由于已知标签分布，因此通过逐一更新参数 **P**, **V**, **W** 即可完成标签分布算法的训练过程。

原型矩阵 **P** 的优化可以归结为非负最小二乘法问题（NNLS）。采用 active set 优化方法[146] 可以快速求解。该算法同样被应用于非负矩阵分解算法中[147]，但是需要多次交替进行求解。求解原型矩阵可以被拆解为 d 个独立的 NNLS 子问题，每个子问题对于输入特征的第 k 个特征进行独立求解，即 $\mathbf{X}_{train} = [\boldsymbol{x}_k]$，

且 $\boldsymbol{x}_k \in \mathbb{R}^{N\times 1}$。对于 $\boldsymbol{x}_k$ 的优化，公式可以写为：

$$\min_{\boldsymbol{p}^k} ||\mathbf{Y}\boldsymbol{p}^k - \boldsymbol{x}^k||_2^2, \quad s.t.\, \boldsymbol{p}^k \geqslant 0. \tag{4-8}$$

系数矩阵 **V** 可以直接使用最小二乘法的闭式解法进行解决，该闭式解法的公式如公式 (4-9) 所示：

$$\mathbf{V} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{X}. \tag{4-9}$$

对于 **W** 的无约束优化，采用有限记忆准牛顿方法（L-BFGS）[148] 来快速优化可以进行求解，而无需像牛顿法一样计算 Hessian 矩阵的逆。使用 L-BFGS 方法进行优化求解时，必须计算目标函数的一阶导数，可通过如下公式计算为

$$\frac{\partial \mathcal{L}_{KL}(\mathbf{W})}{\partial w_c^j} = \sum_{i=1}^{N} \frac{\exp(\boldsymbol{w}_c\mathbf{u}_i^T)u_i^j}{\sum_{c=1}^{C}\exp(\boldsymbol{w}_c\mathbf{u}_i^T)} - \sum_{i=1}^{N} y_i^c u_i^j. \tag{4-10}$$

由于本文所涉及的基本为便于求解收敛的凸优化问题，因此在实验中还考虑了采用成熟优化包的流程作为对比。本章采用 SDPT3[149] 来计算求解原型矩阵 **P** 与参数矩阵 **V**，该优化方案在[150][151] 提供的 CVX 包中实现。为方便对比其他算法，因此最大熵模型未采用 SDPT3 优化方法进行优化。后续实验部分将对比不同优化算法的预测性能以及计算效率，对本章工作在实际使用场景中的优化方法提供选择的参考结果。

算法4.3中总结了本文所提出的标签分布学习算法的训练与测试过程。

4.4　实验验证

4.4.1　实验设置

本章选择了 13 个来自不同研究领域的真实世界数据集用于验证不同标签分布学习算法的效果。以 “Yeast” 开头命名的 10 个数据集来自于出芽酵母（budding Yeast Saccharomyces cerevisiae）上进行的生物实验[152]，每个数据集来自于一个不同的实验设置。其中包含 24 维的酵母基因特征共 2465 个，每个数据集的类标分布表示在不同时间节点上的基因表达水平。其中类标为不同的时间节点，而基因表示水平则经过标准化后作为时间节点对于该基因的描述程度。sJAFFE 为收

算法 4.3 标签分布学习算法的训练和预测流程

训练阶段:

输入：训练数据集 $\mathcal{D}_{train}$

1: 初始化原型矩阵 **P**，最大熵模型 **W**
2: 使用公式（4-3）对原型矩阵 **P** 进行优化求解
3: 使用公式（4-9）计算系数矩阵 **V** 的最优解
4: 使用公式（4-10）优化求解最大熵模型参数 **W**

输出：优化后的原型矩阵 **P**，最大熵模型 **W**

测试阶段:

输入：测试数据集 $\mathcal{D}_{test}$

1: 使用公式（4-9）计算测试特征 $\mathbf{X}_{test}$ 在原型矩阵 **P** 上的系数矩阵 $\mathbf{V}_{test}$
2: 使用最大熵模型最大熵模型 **W** 通过公式（4-5）来预测标签分布 $p(\hat{\mathbf{Y}}_{test}|\mathbf{X}_{test}, \mathbf{W})$

输出：预测标签矩阵 $\hat{\mathbf{Y}}_{test}$

表 4-1 标签分布问题所采用的 13 个数据集，该表格展示了数据集的特征维数、数据集规模、以及类别个数，在训练中使用使用 10 次 10 折交叉验证法对数据集进行划分。

序号	数据集	# 特征	# 数据集	# 类别个数
1	Yeast-alpha	24	2465	18
2	Yeast-cdc	24	2465	15
3	Yeast-cold	24	2465	4
4	Yeast-diau	24	2465	14
5	Yeast-dtt	24	2465	4
6	Yeast-elu	24	2465	14
7	Yeast-heat	24	2465	6
8	Yeast-spo	24	2465	6
9	Yeast-spoem	24	2465	2
10	Yeast-spo5	24	2465	3
11	s-JAFFE	243	213	6
12	Human Gene	36	30542	68
13	Movie	1869	7755	5

集自 10 名日本女性的面部表情数据集[153]，研究人员将这些表情总结为六种基础情绪的混合，其中包括：快乐、悲伤、惊讶、愤怒、厌恶以及恐惧。每张图像由 60 个日本受试者给出关于 6 种情绪的评分，而该评分经过归一化后进行平均即为数据集的真实标签分布。同时，该面部图像已经通过局部二进制模式（Local Binary Patterns，LBP）算法提取为 243 维的特征[154]。Human Gene 为描述人类基因和疾病之间关系的大规模数据集，其中每个基因被表示为 36 维的描述符向量，其标签对应 68 种疾病，标签分布对应着每种基因与该疾病的相关程度。Movie 数据集包括来自 Netflix 网站对于 7755 部电影的评分，评分等级从 1 到 5，该评

分的分布是由每个评分占总评分的百分比计算出来的。该评分由超过 5400 万评分者的结果平均得到，而每部电影从电影的元数据中提取了 1869 维的特征。数据集的规模及特征维数在表4-1中进行了总结。

本次实验采用了六个标签分布学习算法进行对比，包括 AA-kNN[132]、AA-BP[133]、IIS-LD[133]、BFGS-LD[25]、LALOT[137] 和 Duo-DL[138]。AA-kNN 和 AA-BP 是两种将输出空间转变为实数分布的改进型算法，而其余算法则都是为标签分布学习问题设计的算法。IIS-LLD 和 BFGS-LLD 采用相同的最大熵分类模型及 KL 散度作为损失函数学习标签分布的预测，其中 IIS-LLD 用 Gauss-Newton 方法优化模型，后者则使用 L-BFGS 方法进行优化。LALOT 用最优传输距离代替 KL 散度来捕捉特征空间的几何特征，该方法需要进行交替优化，因此会占用更多的计算时间。Duo-LDL 方法利用具有 $C(C-1)$ 输出神经元的三层 MLP 来捕捉所有标签分布的一对一关系来探索标签的相互依赖性。在实验中发现采用不同的优化方法会对结果产生影响，因此将分别记录两种优化方法的结果。其中使用 LBFGS 的方法为 NPB-BFGS，而采用 SDPT3 的被称为 NPB-SDPT3。

由于标签分布实验的数据集没有进行训练集和验证集的划分，本次实验使用 10 次 10 折交叉验证来评估所有方法，并记录其结果的平均值和标准差。此外，还记录了各方法在每个数据集上的排名，并根据各方法在所有数据集上的表现计算出平均排名。LALOT 的参数设置如下：对于大多数数据集，$\lambda = 0.2$，对于 Yeast-spoem，$\lambda = 2$，对于 s-JAFFE，$\lambda = 0.1$；对于大多数数据集，$C = 200$，对于 Yeast-alpha，Yeast-cdc 和 Yeast-elu，$C = 20$；对于所有数据集，$\eta = 1e{-}4$。AA-kNN 中的邻居数 k 被设定为 5。AA-BP 的隐层神经元数量被设定为 60。Duo-LDL 的参数与原论文相同[138]。

由于标签是分布而非整数，因此需要采用标签分布问题专用的评价指标对其进行评价。直觉上来说，两个标签向量之间的相似度越高或者距离越短，则预测标签越准确。根据论文[25] 所指出的，不同的衡量指标体现了对算法不同的评价方向，因此本次实验中共采用六种不同的距离指标来评估本章工作的性能，包括 *Cosine* 相似度, *Chebyshev* 距离, *KL* 散度, *Euclidean* 距离, *Canberra* 距离和 *Intersection* 相似度。*Cosine* 相似度和 *Intersection* 相似度越大表明两个分布越相近，因而算法表现越好；其他四个距离度量指标则是越小表明两个分布越相近，

因此算法表现越好。上述度量指标的计算公式总结如下：

$$Euclidean(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \sqrt{\sum_{c=1}^{C} (y_i^c - \hat{y}_i^c)^2}. \tag{4-11}$$

$$Cosine(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \frac{\sum_{c=1}^{C} y_i^c \hat{y}_i^c}{||\boldsymbol{y}_i||_2 ||\hat{\boldsymbol{y}}_i||_2}. \tag{4-12}$$

$$KL(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \sum_{c=1}^{C} y_i^c \ln \frac{y_i^c}{\hat{y}_i^c}. \tag{4-13}$$

$$Chebyshev(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \max_c |y_i^c - \hat{y}_i^c|. \tag{4-14}$$

$$Canberra(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \sum_{c=1}^{C} \frac{|y_i^c - \hat{y}_i^c|}{y_i^c + \hat{y}_i^c}. \tag{4-15}$$

$$Intersection(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \sum_{c=1}^{C} \min(y_i^c, \hat{y}_i^c). \tag{4-16}$$

其中 $\boldsymbol{d}_i$ 是第 i 个测试特征的真实标签分布，$\hat{\boldsymbol{d}}_i$ 是对应的模型预测标签分布。可以看出，Chebyshev 指标仅考虑标签分布中最大标签之间的差值，而除此之外的指标均平等地考察每个标签的预测结果之间的差异。

4.4.2 结果分析

上述共 8 种方法在 13 个数据集上的所有指标结果报告在表4-2和表4-3中, 结果数值的格式为均值 ± 方差（排名），且每个数据集上当前指标表现最好的结果被加粗表示。首先观察不同指标之间的结果对比，可以发现尽管采用了六种指标，且不同算法在不同指标上的优势不同，但是整体趋势基本是一致的。基本可以认为，前四种算法 AA-kNN、AA-BP、IIS-LLD 以及 LALOT 的整体表现不如后四种算法，但是 LALOT 算法在 Spo5、Dtt 两个数据集上的部分指标处取得了较好的表现。同时，Human Gene 数据集在 Cosine、Euclidean、KL、Canberra、Intersection 五个指标上的普遍结果都差于其他数据集，尤其是 Canberra 指标尤为突出。一方面可以认为该数据集较难进行学习，另一方面，Canberra 在 0 附近具有对小扰动敏感的特性[25]。而 Human Gene 数据集中由于基因可能只与某些病相关，因此标签分布中接近 0 的数值较多，因此产生了特殊的指标结果。总体来说，没有最优的判断指标，因此可以结合在不同指标上的排名综合考虑不同算

表 4-2　8 种标签分布学习方法在 13 个数据集上使用三种衡量标准：Chebyshev 距离，Cosine 相似度以及 Euclidean 距离的实验结果。

Chebyshev 距离

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	.0147 ± .0008(5)	.0401 ± .0022(8)	.0156 ± .0004(6)	.0138 ± .0002(4)	.0160 ± .0010(7)	.0135 ± .0003(3)	**.0134 ± .0004(1)**	**.0134 ± .0004(1)**
Cdc	.0173 ± .0005(6)	.0411 ± .0022(8)	.0184 ± .0005(7)	.0170 ± .0007(5)	.0165 ± .0005(3)	.0165 ± .0007(3)	**.0162 ± .0005(1)**	**.0162 ± .0005(1)**
Cold	.0542 ± .0017(5)	.0598 ± .0031(8)	.0545 ± .0017(7)	.0543 ± .0028(6)	.0512 ± .0015(3)	.0513 ± .0016(4)	.0510 ± .0018(2)	**.0510 ± .0017(1)**
Diau	.0385 ± .0012(5)	.0531 ± .0053(8)	.0397 ± .0011(6)	.0418 ± .0010(7)	**.0370 ± .0012(1)**	.0374 ± .0011(4)	**.0370 ± .0012(1)**	**.0370 ± .0012(1)**
Dtt	.0385 ± .0013(6)	.0470 ± .0042(8)	.0406 ± .0014(7)	.0374 ± .0014(5)	.0361 ± .0012(3)	.0360 ± .0013(4)	**.0359 ± .0012(1)**	**.0359 ± .0012(1)**
Elu	.0173 ± .0004(6)	.0409 ± .0023(8)	.0186 ± .0004(7)	.0170 ± .0004(5)	.0164 ± .0005(3)	.0165 ± .0005(3)	**.0163 ± .0004(1)**	**.0163 ± .0004(1)**
Heat	.0441 ± .0012(6)	.0534 ± .0035(8)	.0495 ± .0013(7)	.0435 ± .0011(5)	.0422 ± .0013(3)	.0425 ± .0013(4)	**.0422 ± .0012(1)**	**.0422 ± .0012(1)**
Spo	.0627 ± .0023(7)	.0684 ± .0031(8)	.0605 ± .0018(5)	.0606 ± .0020(6)	.0585 ± .0020(3)	.0586 ± .0021(4)	**.0583 ± .0018(1)**	**.0583 ± .0018(1)**
Spoem	.0904 ± .0047(7)	.0892 ± .0049(6)	.0905 ± .0036(8)	.0880 ± .0055(5)	.0871 ± .0037(2)	**.0870 ± .0037(1)**	.0873 ± .0037(3)	.0874 ± .0037(4)
Spo5	.0948 ± .0036(6)	.0949 ± .0036(7)	.0931 ± .0037(4)	**.0908 ± .0037(1)**	.0913 ± .0033(4)	.0914 ± .0040(5)	.0912 ± .0038(2)	.0912 ± .0038(2)
sJAFFE	.1141 ± .0108(3)	.1272 ± .0126(7)	.1194 ± .0130(5)	.1191 ± .0110(6)	.1291 ± .0120(8)	.1142 ± .0132(4)	**.0956 ± .0103(1)**	.0959 ± .0103(2)
Human Gene	.0648 ± .0018(8)	.0624 ± .0019(7)	.0534 ± .0016(4)	.0534 ± .0007(2)	.0534 ± .0007(2)	.0534 ± .0018(5)	.0534 ± .0018(5)	**.0533 ± .0018(1)**
Movie	.1542 ± .0048(7)	.1572 ± .0024(8)	.1508 ± .0016(6)	.1382 ± .0007(5)	.1240 ± .0032(3)	.1355 ± .0018(4)	.1199 ± .0256(2)	**.1197 ± .0024(1)**
Avg.Rank	5.92	7.62	6.00	4.77	3.46	3.69	1.69	**1.38**

Cosine 相似度

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	.9938 ± .0003(6)	.9391 ± .0059(8)	.9927 ± .0003(7)	.9939 ± .0002(5)	**.9946 ± .0003(1)**	.9943 ± .0003(4)	**.9946 ± .0003(1)**	**.9946 ± .0003(1)**
Cdc	.9924 ± .0004(5)	.9508 ± .0047(8)	.9915 ± .0004(7)	.9924 ± .0004(5)	**.9933 ± .0003(1)**	.9928 ± .0004(4)	**.9933 ± .0003(1)**	**.9933 ± .0003(1)**
Cold	.9872 ± .0008(5)	.9844 ± .0016(8)	.9871 ± .0009(6)	.9870 ± .0015(7)	.9886 ± .0008(2)	.9880 ± .0007(4)	.9886 ± .0008(2)	**.9886 ± .0007(1)**
Diau	.9866 ± .0008(5)	.9742 ± .0053(8)	.9861 ± .0007(7)	.9850 ± .0005(6)	**.9879 ± .0007(1)**	.9869 ± .0007(4)	**.9879 ± .0007(1)**	**.9879 ± .0007(1)**
Dtt	.9933 ± .0005(6)	.9898 ± .0021(8)	.9926 ± .0005(7)	.9934 ± .0005(5)	**.9941 ± .0004(1)**	.9940 ± .0005(4)	**.9941 ± .0004(1)**	.9941 ± .0005(3)
Elu	.9931 ± .0002(6)	.9557 ± .0042(8)	.9922 ± .0003(7)	.9934 ± .0001(5)	.9940 ± .0002(2)	.9940 ± .0003(3)	**.9941 ± .0002(1)**	.9940 ± .0003(3)
Heat	.9867 ± .0006(6)	.9782 ± .0030(8)	.9857 ± .0007(7)	.9871 ± .0005(5)	**.9880 ± .0006(1)**	.9878 ± .0006(4)	**.9880 ± .0006(1)**	**.9880 ± .0006(1)**
Spo	.9730 ± .0017(7)	.9679 ± .0029(8)	.9753 ± .0013(5)	.9745 ± .0014(6)	**.9770 ± .0012(1)**	.9768 ± .0013(4)	**.9770 ± .0012(1)**	**.9770 ± .0012(1)**
Spoem	.9764 ± .0023(8)	.9778 ± .0034(6)	.9774 ± .0015(7)	.9778 ± .0023(5)	**.9790 ± .0015(1)**	**.9790 ± .0015(1)**	.9788 ± .0016(3)	.9788 ± .0016(3)
Spo5	.9713 ± .0022(8)	.9723 ± .0019(7)	.9731 ± .0019(6)	**.9741 ± .0007(1)**	.9741 ± .0018(4)	.9741 ± .0016(2)	.9741 ± .0018(4)	.9741 ± .0016(2)
sJAFFE	.9337 ± .0182(3)	.9145 ± .0140(8)	.9314 ± .0104(5)	.9316 ± .0083(4)	.9100 ± .0100(7)	.9301 ± .0121(6)	**.9531 ± .0086(1)**	.9530 ± .0090(2)
Human Gene	.7687 ± .0046(7)	.6906 ± .0087(8)	.8334 ± .0040(5)	.8333 ± .0018(6)	**.8345 ± .0020(1)**	.8342 ± .0039(3)	.8342 ± .0039(3)	.8345 ± .0039(2)
Movie	.8802 ± .0026(8)	.8948 ± .0012(7)	.9067 ± .0023(6)	.9147 ± .0028(5)	.9264 ± .0032(3)	.9231 ± .0028(4)	.9298 ± .0027(2)	**.9299 ± .0032(1)**
Avg.Rank	6.15	7.77	6.30	5.00	2.00	3.62	1.69	**1.69**

Euclidean 距离

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	.0249 ± .0005(6)	.0798 ± .0046(8)	.0273 ± .0005(7)	.0245 ± .0004(5)	**.0231 ± .0006(1)**	.0236 ± .0003(4)	**.0231 ± .0006(1)**	**.0231 ± .0006(1)**
Cdc	.0301 ± .0007(6)	.0770 ± .0043(8)	.0320 ± .0007(7)	.0288 ± .0008(5)	**.0279 ± .0007(1)**	.0284 ± .0007(4)	**.0279 ± .0007(1)**	**.0279 ± .0007(1)**
Cold	.0724 ± .0023(6)	.0798 ± .0040(8)	.0728 ± .0024(7)	.0717 ± .0038(5)	.0681 ± .0025(3)	.0691 ± .0024(4)	**.0681 ± .0022(1)**	**.0681 ± .0022(1)**
Diau	.0567 ± .0016(5)	.0793 ± .0078(8)	.0591 ± .0014(6)	.0593 ± .0013(7)	**.0543 ± .0015(1)**	.0546 ± .0016(4)	**.0543 ± .0015(1)**	**.0543 ± .0015(1)**
Dtt	.0512 ± .0016(6)	.0627 ± .0056(8)	.0541 ± .0018(7)	.0511 ± .0017(5)	**.0480 ± .0015(3)**	.0481 ± .0017(4)	**.0479 ± .0017(1)**	**.0479 ± .0017(1)**
Elu	.0298 ± .0005(6)	.0750 ± .0041(8)	.0321 ± .0006(7)	.0293 ± .0004(5)	**.0278 ± .0006(4)**	**.0277 ± .0006(1)**	**.0277 ± .0006(1)**	**.0277 ± .0006(1)**
Heat	.0622 ± .0016(6)	.0792 ± .0051(8)	.0651 ± .0017(7)	.0615 ± .0013(5)	.0593 ± .0016(3)	.0594 ± .0016(4)	**.0592 ± .0015(1)**	**.0592 ± .0015(1)**
Spo	.0880 ± .0027(7)	.0975 ± .0043(8)	.0854 ± .0024(6)	.0851 ± .0029(5)	**.0816 ± .0021(1)**	.0822 ± .0026(4)	.0819 ± .0024(3)	.0819 ± .0022(2)
Spoem	.1279 ± .0066(6)	.1432 ± .0083(8)	.1280 ± .0050(7)	.1244 ± .0078(5)	.1233 ± .0048(2)	**.1231 ± .0050(1)**	.1235 ± .0053(3)	.1235 ± .0053(3)
Spo5	.1216 ± .0046(7)	.1216 ± .0047(8)	.1192 ± .0047(6)	**.1165 ± .0049(1)**	**.1165 ± .0049(1)**	.1170 ± .0040(5)	.1167 ± .0048(4)	.1167 ± .0041(3)
sJAFFE	.1564 ± .0100(7)	.1713 ± .0151(8)	.1531 ± .0131(3)	.1536 ± .0099(4)	.1542 ± .0162(5)	.1549 ± .0146(6)	**.1232 ± .0113(1)**	.1234 ± .0116(2)
Human Gene	.1058 ± .0020(7)	.1272 ± .0028(8)	.0868 ± .0018(6)	.0867 ± .0009(5)	.0864 ± .0015(3)	.0865 ± .0020(4)	.0863 ± .0019(2)	**.0863 ± .0019(1)**
Movie	.2564 ± .0321(8)	.2221 ± .0211(7)	.2004 ± .0145(6)	.1876 ± .0110(5)	.1789 ± .0044(3)	.1819 ± .0033(4)	.1738 ± .0033(2)	**.1736 ± .0023(1)**
Avg.Rank	6.38	7.92	6.31	4.77	2.38	3.79	1.69	**1.46**

法之间的优劣。

从平均排名可以对算法的整体性能进行分析，本章算法在所有指标上采用 SDPT3 优化方法的排名分别为 1.38，1.69，1.46，2.00，1.77，1.69，在所有算法的排名表现中最优，证实了本章工作的有效性。对比采用快速牛顿算法 L-BFGS 进行优化的结果来看，该优化方法的整体排名为 1.69，1.69，1.69，2.15，2.00，2.23，因此整体结果相较于采用 SDPT3 优化方法的较差，且在 Intersection 指标上不如 Duo-LDL 方法。可以看出，寻找近似解尽管能够提升算法的训练速度，但是对最终的分类性能仍然产生了影响，降低了算法的表现。无论在哪种指标上，本章工作都未取得好结果的数据集为 Spo5 与 Spoem，可以认为该数据集上未学习到良好的原型表示，因此最终降低了算法性能。总体来说，从排名进行分析可以认为本章工作相较于以往算法在分类性能上具有竞争力。

实验中对其他标签分布算法也进行了考察，可以比较已有标签分布算法的

表 4-3 8 种标签分布学习方法在 13 个数据集上使用三种衡量标准：KL 散度，Canberra 距离以及 Intersection 相似度的实验结果。

KL 散度

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	ENC-BFGS	NPB-SDPT3
Alpha	.0064 ± .0004(6)	.0771 ± .0084(8)	.0075 ± .0003(7)	.0062 ± .0002(4)	**.0056 ± .0004(1)**	.0063 ± .0003(5)	**.0056 ± .0004(1)**	**.0056 ± .0004(1)**
Cdc	.0083 ± .0005(6)	.0608 ± .0065(8)	.0092 ± .0005(7)	.0078 ± .0004(5)	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**	**.0074 ± .0005(1)**
Cold	.0142 ± .0018(6)	.0174 ± .0026(8)	.0144 ± .0020(7)	.0135 ± .0012(5)	**.0129 ± .0020(1)**	.0130 ± .0014(4)	**.0129 ± .0022(1)**	**.0129 ± .0018(1)**
Diau	.0151 ± .0012(5)	.0299 ± .0069(8)	.0159 ± .0011(6)	.0162 ± .0005(7)	**.0138 ± .0010(1)**	.0139 ± .0012(2)	.0140 ± .0011(3)	.0140 ± .0011(3)
Dtt	.0076 ± .0016(6)	.0114 ± .0027(8)	.0084 ± .0016(7)	**.0067 ± .0004(1)**	.0068 ± .0013(2)	.0071 ± .0015(5)	.0069 ± .0015(3)	.0069 ± .0015(3)
Elu	.0073 ± .0004(6)	.0540 ± .0058(8)	.0083 ± .0004(7)	.0070 ± .0003(5)	**.0065 ± .0004(3)**	.0066 ± .0005(4)	**.0064 ± .0004(1)**	**.0064 ± .0004(1)**
Heat	.0145 ± .0011(6)	.0244 ± .0038(8)	.0156 ± .0012(7)	.0137 ± .0010(5)	.0133 ± .0013(3)	.0135 ± .0012(4)	.0133 ± .0011(2)	**.0133 ± .0010(1)**
Spo	.0303 ± .0021(7)	.0368 ± .0037(8)	.0281 ± .0019(6)	.0272 ± .0021(5)	**.0258 ± .0017(1)**	.0265 ± .0018(4)	.0263 ± .0017(2)	.0263 ± .0018(3)
Spoem	.0291 ± .0037(8)	.0283 ± .0034(6)	.0291 ± .0035(7)	.0256 ± .0032(2)	**.0255 ± .0030(1)**	.0270 ± .0035(3)	.0273 ± .0037(4)	.0273 ± .0038(5)
Spo5	.0343 ± .0031(8)	.0339 ± .0032(7)	.0330 ± .0032(6)	.0295 ± .0024(2)	**.0293 ± .0022(1)**	.0324 ± .0031(5)	.0322 ± .0034(3)	.0322 ± .0034(3)
sJAFFE	.0712 ± .0231(4)	.0960 ± .0183(7)	.0700 ± .0089(3)	.0724 ± .0084(5)	.1061 ± .0112(8)	.0740 ± .0135(6)	**.0500 ± .0090(1)**	**.0500 ± .0090(1)**
Human Gene	.3010 ± .0084(7)	.4691 ± .0169(8)	.2264 ± .0072(3)	.2265 ± .0056(5)	.2358 ± .0110(6)	.2264 ± .0070(2)	.2264 ± .0072(3)	**.2262 ± .0072(1)**
Movie	.2008 ± .0102(7)	.1792 ± .0246(6)	.1368 ± .0121(5)	.4572 ± .0331(8)	**.1131 ± .0625(1)**	.1292 ± .0056(4)	.1210 ± .0049(3)	.1209 ± .0049(2)
Avg.Rank	6.31	7.54	6.00	4.54	2.31	3.77	2.15	**2.00**

Canberra 距离

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	.7582 ± .0289(6)	2.3521 ± .1282(8)	.7625 ± .0351(7)	.7343 ± .0182(5)	.6813 ± .0186(3)	.6845 ± .0170(4)	**.6812 ± .0174(1)**	.6813 ± .0171(2)
Cdc	.7172 ± .0215(7)	1.7152 ± .1055(8)	.7086 ± .0215(6)	.6871 ± .0183(5)	.6462 ± .0180(2)	.6487 ± .0161(4)	**.6461 ± .0173(1)**	.6466 ± .0157(3)
Cold	.2604 ± .0102(6)	.2681 ± .0143(8)	.2487 ± .0091(7)	.2579 ± .0098(5)	.2408 ± .0092(4)	.2402 ± .0090(3)	.2398 ± .0089(2)	**.2398 ± .0080(1)**
Diau	.4551 ± .0112(6)	.5675 ± .0310(8)	.4487 ± .0170(5)	.4851 ± .0103(7)	**.4331 ± .0100(4)**	.4312 ± .0103(2)	.4312 ± .0140(3)	**.4311 ± .0129(1)**
Dtt	.1821 ± .0071(7)	.2043 ± .0118(8)	.1812 ± .0051(6)	.1772 ± .0071(5)	.1689 ± .0060(3)	.1690 ± .0062(4)	.1687 ± .0065(2)	**.1686 ± .0062(1)**
Elu	.6442 ± .0143(7)	1.4885 ± .0672(8)	.6387 ± .0193(6)	.6253 ± .0152(5)	.5853 ± .0115(4)	.5831 ± .0142(3)	**.5823 ± .0128(1)**	.5825 ± .0130(2)
Heat	.3918 ± .0112(7)	.4589 ± .0286(8)	.3772 ± .0068(5)	.3792 ± .0011(6)	.3646 ± .0100(4)	.3642 ± .0072(2)	.3642 ± .0090(3)	**.3640 ± .0098(1)**
Spo	.5597 ± .0218(7)	.5992 ± .0417(8)	.5231 ± .0312(5)	.5258 ± .0216(6)	.5137 ± .0135(4)	.5133 ± .0145(3)	.5127 ± .0155(2)	**.5126 ± .0144(1)**
Spoem	.1914 ± .0089(8)	.1842 ± .0108(7)	.1840 ± .0099(6)	.1814 ± .0090(5)	.1812 ± .0072(4)	**.1799 ± .0082(1)**	.1808 ± .0092(3)	.1808 ± .0082(2)
Spo5	.2969 ± .0146(8)	.2912 ± .0170(7)	.2871 ± .0191(6)	.2831 ± .0121(5)	**.2821 ± .0100(1)**	.2829 ± .0101(4)	.2823 ± .0115(2)	.2824 ± .0104(3)
sJAFFE	.8431 ± .1131(5)	1.0462 ± .1250(7)	.8751 ± .0842(6)	1.0682 ± .0983(8)	.8142 ± .0700(3)	.8202 ± .0675(4)	**.7108 ± .0553(1)**	.7115 ± .0612(2)
Human Gene	16.2832 ± .8072(7)	22.7847 ± 1.8523(8)	14.5412 ± .6534(6)	14.4873 ± .4323(5)	**14.4423 ± .2176(1)**	14.4532 ± .2207(2)	14.4543 ± .2282(3)	14.4543 ± .2282(3)
Movie	1.2758 ± .0457(7)	1.2693 ± .0872(6)	1.1367 ± .0542(5)	2.2317 ± .1011(8)	1.0772 ± .0201(4)	1.0617 ± .0173(3)	1.0345 ± .0195(2)	**1.0337 ± .0175(1)**
Avg.Rank	6.77	7.62	5.85	5.77	3.15	3.00	2.00	**1.77**

Intersection 相似度

method	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	.9581 ± .0021(6)	.8772 ± .0081(8)	.9578 ± .0021(7)	.9592 ± .0024(5)	**.9624 ± .0009(1)**	.9621 ± .0009(4)	**.9624 ± .0009(1)**	**.9624 ± .0009(1)**
Cdc	.9528 ± .0028(7)	.8912 ± .0051(8)	.9532 ± .0021(6)	.9541 ± .0010(5)	**.9575 ± .0010(1)**	.9574 ± .0010(4)	**.9575 ± .0010(1)**	**.9575 ± .0010(1)**
Cold	.9362 ± .0024(7)	.9340 ± .0032(8)	.9376 ± .0022(5)	.9363 ± .0020(6)	**.9409 ± .0019(1)**	.9408 ± .0019(4)	.9409 ± .0021(3)	**.9409 ± .0019(1)**
Diau	.9371 ± .0021(6)	.9224 ± .0039(8)	.9381 ± .0020(5)	.9328 ± .0022(7)	.9402 ± .0017(3)	**.9403 ± .0017(1)**	.9402 ± .0017(3)	**.9403 ± .0017(1)**
Dtt	.9549 ± .0021(7)	.9502 ± .0021(8)	.9552 ± .0016(6)	.9563 ± .0015(5)	.9582 ± .0014(4)	.9583 ± .0015(3)	**.9584 ± .0014(1)**	**.9584 ± .0014(1)**
Elu	.9546 ± .0011(7)	.8992 ± .0054(8)	.9547 ± .0011(6)	.9564 ± .0022(5)	**.9591 ± .0010(1)**	.9589 ± .0009(2)	.9589 ± .0015(4)	.9589 ± .0009(2)
Heat	.9362 ± .0023(7)	.9251 ± .0054(8)	.9384 ± .0011(6)	.9871 ± .0005(5)	**.9406 ± .0014(1)**	.9402 ± .0016(4)	.9402 ± .0014(3)	.9403 ± .0016(2)
Spo	.9082 ± .0043(7)	.9022 ± .0069(8)	.9143 ± .0052(5)	.9134 ± .0031(6)	**.9156 ± .0023(1)**	.9155 ± .0023(4)	**.9156 ± .0023(1)**	**.9156 ± .0023(1)**
Spoem	.9072 ± .0043(8)	.9108 ± .0056(7)	.9109 ± .0054(6)	.9126 ± .0044(5)	.9128 ± .0058(2)	**.9131 ± .0038(1)**	.9127 ± .0042(3)	.9126 ± .0037(4)
Spo5	.9044 ± .0051(8)	.9062 ± .0054(7)	.9072 ± .0034(6)	.9088 ± .0043(3)	**.9088 ± .0033(1)**	.9086 ± .0031(5)	.9088 ± .0034(4)	**.9088 ± .0033(1)**
sJAFFE	.8552 ± .0215(4)	.8243 ± .0216(7)	.8513 ± .0147(5)	.8058 ± .0221(8)	.8310 ± .0123(6)	.8606 ± .0121(3)	**.8797 ± .0101(1)**	.8797 ± .0108(2)
Human Gene	.7433 ± .0128(7)	.6712 ± .0221(8)	.7828 ± .0098(6)	.7841 ± .0018(5)	**.7852 ± .0042(1)**	.7846 ± .0034(3)	.7846 ± .0028(2)	.7842 ± .0034(4)
Movie	.7801 ± .0056(7)	.7882 ± .0112(6)	.8004 ± .0100(5)	.6496 ± .0311(8)	.8221 ± .0040(3)	.8192 ± .0054(4)	.8282 ± .0034(2)	**.8284 ± .0032(1)**
Avg.Rank	6.77	7.62	5.69	5.62	2.00	3.23	2.23	**1.69**

性能：可以看出，对已有算法进行改进的 AA-BP 和 AA-kNN 算法相较其他算法较差，这也可以说明目前针对标签分布问题所设计的算法在预测上具有优势。较新的 Duo-LD 算法也具有一定的竞争优势，而这个算法则可以看作是 AA-BP 类神经网络算法针对标签分布的改进，即考虑了标签之间的关联而将标签之间的二元关系作为网络的预期输出。从该实验中也可以发现，Duo-LD 的新损失函数与 AA-BP 相比提高了预测性能。同样的，IIS-LLD 与 BFGS-LLD 仅采用了不同的优化方法，可以看出采用近似牛顿法进行优化学习的模型表现更好。

相较于在原始数据上直接使用最大熵模型，引入非负原型学习框架是否提升了模型的分类性能？对该问题的分析可以通过禁用原型学习模块的消融实验得到，该消融实验结果对比可以参考表4-2和表4-3中的 BFGS-LLD 方法的结果得到，因为该模型采用了相同的最大熵模型来学习标签分布而未采用本文提出的原型学习框架。从体现整体结果的排名可以看出，BFGS-LLD 算法的排名分别为 3.69，3.62，3.79，3.77，3.00，3.23，因此采用原型学习框架对整体性能是具

有提升作用的。但是在 Yeast-spoem 数据集上，BFGS-LLD 算法在六个指标上均优于本文算法，而本文算法在这个数据集上的表现也较差，因此可以认为在该数据集上原型无法很好地表示每个标签的性质，最终影响了最大熵模型的分类。除此之外，Human Gene 数据集的标签数量多于特征维数，此时原型学习框架相当于一种字典学习方法，可以从消融实验中分析该字典学习的优劣。在 Canberra 指标上，两种优化方案的结果均不如未采用原型学习框架；而其他指标则能够至少有一种优化方式表现更好。因此，大体上原型学习框架在该数据集上仍然对结果是正提升的。从消融实验的结果可以看出，本文提出的非负原型学习框架提升了标签分布算法的算法性能。

另外需要注意的是，从上述结果可以发现优化方法对非负原型框架的学习十分重要。在实验中发现，在 Human Gene 数据集上，原型基底矩阵 $\mathbf{P}^T\mathbf{P}$ 的条件数非常大，即在一次测试中达到6.66×10^{17}，这将导致逆矩阵的不可靠计算。因此，本文对 $\mathbf{P}^T\mathbf{P}$ 计算摩尔逆（moore-penrose inverse），以保持数值的稳定性。

总而言之，与只采用分类模型相比，非负原型学习框架确实提高了大多数数据集上标签分布学习的性能。与已有工作相比，本文工作是解决标签分布学习问题的具有竞争力的算法。由于本章算法没有人工参数设置，因此不再进行参数敏感性分析。

表 4-4　8 种标签分布学习方法在 13 种数据集上的运行时间（单位：秒）

方法/数据集	AA-kNN	AA-BP	IIS-LLD	LALOT	Duo-LDL	BFGS-LLD	NPB-BFGS	NPB-SDPT3
Alpha	1.22	3.82	1.61	44.24	5.21	1.16	6.51	3.67
Cdc	0.45	1.75	1.45	35.41	3.43	0.93	3.23	3.01
Cold	0.42	1.45	0.78	10.02	0.33	0.47	1.03	1.35
Diau	0.53	1.24	0.89	7.42	0.47	0.65	2.57	1.69
Dtt	0.37	1.17	0.57	8.55	0.38	0.46	0.89	1.36
Elu	0.91	1.38	1.16	20.12	3.05	0.92	2.79	3.04
Heat	0.22	0.89	0.45	13.89	0.78	0.59	2.38	1.62
Spo	0.39	1.27	0.44	11.99	0.49	0.59	2.62	1.67
Spoem	0.44	1.26	0.55	6.28	0.28	0.33	0.42	0.98
Spo5	0.33	1.08	0.89	7.91	0.23	0.39	0.75	1.12
s-JAFFE	2.73	3.46	2.48	305.45	7.65	27.16	1.52	1.77
Human Gene	76.88	35.90	36.91	778.32	92.28	150.87	82.31	231.96
Movie	43.86	75.53	46.01	6117.85	900.42	124.94	11.45	412.21

4.4.3　训练资源分析

标签分布算法多采用需要优化的参数进行，因此需要对模型参数所需训练的时长在同一实验环境中进行比较，训练时间总结在表4-4中，该表展示了 8 种

标签分布学习方法在 13 个数据集上训练到收敛所需时长。从整体趋势上来说，在 Human Gene 数据集与 Movie 数据集上所有方法都花费了更多的时间，因此也可以看出这两个数据集的处理难度较大。其他数据集上除 LALOT 方法、BFGS-LLD 在 sJAFFE 数据集上的时长外，其他方法的模型训练基本都在 10s 内结束。综合考虑这些训练时长可以看出，AA-kNN、AA-BP、IIS-LLD 以及 NPB-BFGS 是速度较快的算法，尽管在较难学习的 Human Gene 以及 Movie 数据集上的训练时长都没有超过 100 秒。Duo-LDL、BFGS-LLD 以及 NPB-SDPT3 则训练时长稍长，这些算法在不同数据集上各有优劣，比如 Duo-LDL 在 Movie 数据集上耗费时长较长，NPB-SDPT3 则在这两个数据集上的训练时长都长于 BFGS-LLD。相较于其他算法，LALOT 的训练时长较长。

相比较两种不同的优化算法，SDPT3 在 Human Gene 以及 Movie 数据集上花费的时间远多于 L-BFGS，因此 L-BFGS 优化算法拥有更大的计算优势。而对于 sJAFFE 数据集，本章所提出的两种优化方式下的算法训练时长都比未使用原型学习框架而仅使用最大熵模型的 BFGS-LLD 算法要短，可以体现出原型学习框架花费较少的计算代价使得数据转换到维数较少且更易进行学习的嵌入空间中，且随之降低了最大熵模型的训练时长。结合分类结果分析可以得出，在该数据集上采用原型学习框架无疑使各方面性能都有所提升。对比后两个数据集，这三种方法的训练速度为 NPB-BFGS>BFGS-LLD>NPB-SDPT3，结合分类性能来看可以认为 SDPT3 使用了更多的优化时间用于寻找更优的原型学习方案。

与其他对比方案相比，NPB-BFGS 在训练时长成本上更具有竞争力。NPB-SDPT3 在 Human Gene 和 Movie 数据集上花费了更多计算代价，但综合考虑上节中对分类结果的分析，该优化方案比 L-BFGS 优化方法更为可靠。总的来说，本文所提出的基于非负原型学习框架的 NPB-BFGS 和 NPB-SDPT3 方法能够有效解决标签分布问题。

4.5 小结

为解决预测实数化标签的标签分布问题，本文提出了非负原型线性学习框架，并在此基础上提出了解决标签分布问题的非负原型基底学习算法。其主要思路是为每个标签所描述的性质学习一个代表性的原型向量作为基底，并将数据

表示称为原型向量之间的线性组合，从而将输入特征转换为不同模式进行组合的系数空间。在此原型学习框架的基础上，本文继续在系数空间上进行最大熵模型的训练，从而获得标签预测模型。在实验中采用多个数据集对本文算法与其他标签分布算法进行对比，展示了该算法在解决标签分布问题上的优势。同时，通过算法运行时间对比与消融实验的证实了原型学习框架确实有效，且能够快速训练出模型。

该项工作对算法的性能进行了提升，然而依旧存在改进空间，可以作为未来的研究方向。从原型学习框架的角度进行考虑，该工作对于每个标签都学习单个原型向量进行表示，然而标签所对应的性质是否可以仅用单个向量进行描述？若能够增加原型集合的数量，则可能能够对输入数据进行更好地表征学习；从标签分布问题的角度进行考虑，考虑到标签可能随时间变化而增加，因此如何在增量学习环境下完成标签分布的学习，如何在标签增加后考虑新标签与旧标签的描述程度值关系，仍然值得研究者们继续在该问题上进行探索。

第五章　基于单原型学习框架的数据增量算法

5.1　引言

在第三、四章节中，本文提出了使用原型学习框架的线性模型以及非线性模型解决标签预测问题的方案，这两个算法计算复杂度低，模型训练所需要的资源较少，能够在训练数据较少的小型应用上较快地训练出一个可行模型。而对于更大规模的数据集分类问题，则使用深度神经网络可以做到有效地将数据转换到嵌入特征空间中，其中使用线性分类层与交叉熵损失函数是分类问题中神经网络常用的梯度下降训练方式。然而在类增量乃至数据增量问题中，深度神经网络模型及其线性分类器将会遭受严重的遗忘问题，即在接受新类别的训练数据后在旧类别数据上的分类性能将急剧下降。在动态的学习环境中，由于 Softmax 函数使得分类结果的归一化，则最近学习的类别将具有优势并使得网络对它们具有偏好；其次，分类层与特征提取层的解耦性使得在特征提取模型产生变化时，分类层通常需要重新进行训练[82]。此时，采用嵌入特征所学习得到的原型向量进行分类，即减轻了对最近任务的偏好，又能够随着特征提取模块的变化而及时更新。因此，结合原型学习框架使得深度神经网络在增量环境中的分类问题上更具有优势。

根据如上思路，本章将采用单原型学习框架进行分类，提出原型重构损失函数用于网络训练，并结合对比损失函数、回放样本技术形成数据增量学习算法。结合原型学习框架时，除去提供分类器的功能之外，原型向量为神经网络如何训练嵌入特征提供了目标。原型重构损失函数与监督对比损失函数使得同类特征向同类原型聚拢，保持异类嵌入特征尽量互相远离，使得嵌入特征在空间中更有区分性，便于进行分类。结合回放样本技术，本章工作降低了模型在数据增量环境中的遗忘现象。实验章节中对本章工作与对比算法在平衡数据流及非平衡数据流分别进行了验证，并最终验证了本章工作的有效性。

5.2　研究背景

近些年来的研究证实，深度神经网络拥有从大规模数据集中提取有用信息的能力，其能够学习将数据转换到具有意义的低维嵌入空间中。然而，数据集会随着世界的发展而逐渐扩充，甚至于对输入数据的分布产生影响。为减少对模型进行全部数据重训所产生的时间与空间的耗费，增量学习的深度神经网络模型仍然亟待研究。相较于类增量持续学习范式，本章所解决的数据增量学习问题则是在线增量与类增量问题的平衡。该学习范式中不再将学习阶段划分为多个任务，而是随时可以进行小批量样本的学习和模型评估。此外，来自真实世界的数据在不同类别之间经常是不平衡的，网络模型应该能够减少类别数据不平衡之间的差异。在更为灵活的学习与评估方式背后，小批量的学习样本无疑对模型的增量学习能力提出了更高的挑战。

然而，使用梯度下降训练的神经网络模型通常不具备终身学习的能力。大量实验表明，当神经网络仅使用新类别数据进行重新训练时，将几乎遗忘所有在旧类别数据上的预测能力而仅能处理新学习的数据样本，上述现象也被称为灾难性遗忘问题[155]。一部分研究人员认为，训练网络所使用的损失函数，其最小值会随着数据分布的不同而在不同的任务之间变化，从而导致网络为在新任务上达到较好的任务性能而对已经训练好的参数进行大量的改动，从而使得旧任务上的损失函数值变大[93]。除此之外，任务间边界混淆和最近任务分类偏移也被认为是导致神经网络在增量学习中出现问题的原因。本质上来说，在第一章节所提到的稳定性-可塑性困境即使在深度神经网络中也难以解决，其具体体现为为适应新任务所对网络权重进行的改动与为保证解决旧任务而对已学习好的权重的保留[156]。因此，解决上述产生遗忘现象的问题就成为研究者们在深度神经网络持续学习领域研究的重点。

数据重放、正则化和参数隔离三种技术并被证明对抵抗遗忘问题非常有效。其中，最为简单有效的一种技术是在 iCaRL 论文[82] 中提出的数据回放技术[155]，通过在额外内存中保留已训练样本的子集，并且在新任务来临时从内存中提取所有数据或采样部分数据，与新数据进行协同训练。这种新任务中被采样得到的旧任务样本被称之为回放样本。该方法可以看作是在从头重新训练所有数据的高性能和仅在新任务样本上进行训练所减少的计算和空间成本之间取得了平

衡，且这两个平凡的训练方式可以看作是数据重放方法的上界和下界。该方法最大的优势就是简单好用并且性能提升明显，由于数据对于训练模型的重要性是毋容置疑的，因此只要有旧任务的数据就能够对结果有所提升。此外，也有工作[157] 使用因果图证明了数据重放方法在旧数据和新模型之间建立因果路径，因此神经网络能够保存旧任务上的推理方式，从而对克服遗忘问题很有帮助。当然，数据回放方法也存在不少缺陷，例如需要额外的内存保存数据，尽管通常的做法是固定内存容量而平均分给每个类别以防止其随着类别的增加过多，然而对于大量的类别仍然需要一定的内存开销。同样，保存数据也可能带来安全性等诸多本因增量学习而该避免的问题。因此，研究者们提出可以使用生成式模型来产生回放样本，即将保存具体训练样本替换为保存该样本训练出的生成式模型，并使用生成模型在新的训练阶段中直接产生回放样本。可以想见的是，该类模型解决了需要保存旧任务样本的缺陷，但引入新的问题：如果生成式模型产生了不准确的回放样本，也会影响最终的训练效果。因此，如何选择合适的方式对模型进行增量学习，需要根据具体问题来进行权衡。除数据回放方法外，对于神经网络的损失函数进行约束也是一种缓解遗忘的手段。GEM 算法[158] 方法结合约束与回放样本以对抗网络对旧知识的遗忘；另一方面，iCaRL 方法[82] 使用蒸馏损失来迫使模型保持在旧类别数据上的输出，计算蒸馏损失需要保存上个任务训练好的模型，以该模型在新数据上的输出指导模型训练时在旧输出神经元上的输出。最后，参数隔离，即固定一部分与旧任务相关程度更高的参数是一种保持记忆的方式。考虑到网络的容量有限，根据任务数量扩展网络神经元数量以增加网络学习能力也是一种可行的方式。这同样需要面临新任务增加的计算和空间成本与网络表现性能之间的权衡。

对于数据增量模型，数据回放同样是一种有效的技术。Reservoir[159] 是一种快速从未知大小的内存中随机选取一部分样本的采样方法，而将其作为回放样本采样与神经网络简单结合就能够对效果有所提升[160]。另一项工作 MIR[161] 对采样的方法进行改进，选取在模型更新中受影响最大的回放样本，即该样本在新数据训练后的模型上的损失函数值与旧模型上相差最大。基于梯度的样本选择（Gradient based sample selection，GSS）方法[162] 与 GEM 方法的思想类似，认为回放样本计算得到的损失函数值是对新任务中数据损失函数值的约束，使得新任务的损失函数梯度方向与旧任务的损失函数梯度方向小于 90 度。因此，可

以通过选择内存中与其他样本的损失函数梯度方向相似程度最低，或是通过替换掉内存中损失函数的梯度方向与其他样本类似的回放样本，最大限度地增加内存中样本的多样性。持续原型进化（Continual Prototype Evolution，CoPE）方法[83] 总结了 DIL 问题的明确定义以及 learner-evaluator 学习框架。该方法采用持续样本嵌入特征均值持续更新原型，并且将同数据批次的同类样本和异类样本同样视为代理原型，与原型一同参与拉进类内距离，增加类间距离的损失函数用与训练网络。同时，为保证不同类别之间的数据平衡，还使得内存中不同类别的样本数量相同。除此之外，参数隔离方法 CN-DPM[163] 使用神经网络组成专家系统，并通过非参数贝叶斯模型来控制专家系统的扩张。

近年来，对比学习因其在无监督学习，同时也是自监督学习领域的优秀表现引起了研究者们的关注。在没有类别标签的情况下，对比学习认为同一类别的样本在特征嵌入空间中的相似程度应该尽量高，且不同类别的相似程度应该尽量低，因此网络训练时应该通过损失函数分别降低以及增加这两种嵌入特征之间的距离。对比损失可以被认为是三重损失（triplet loss）的扩展，在损失函数中计算了当前训练批的所有正负样本而非只选取一正一负。将上述无监督对比学习扩展到使用类标信息的监督对比损失[85] 也证明了其出色的性能。由于存在类标的信息，正负样本的划分更为明确，除去噪声样本外不会出现将同类样本划分为负样本的情况。由于对比学习对于同类特征的聚集效果，对比学习损失函数适用于采用近类均值 (neareat-class-mean，NCM) 分类器的连续学习方法，这种方法最近已经应用于类增量学习模型，例如 Co^2L[164] 中就采用改进的监督对比损失进行训练。监督对比重放 (Supervised Contrastive Replay，SCR) 方法[23] 同样使用对比损失函数进行训练，并使用重新计算的均值原型分类器对数据标签进行预测。

本章提出监督对比与原型重构学习算法（Supervised Contrastive with Prototype Reconstruction learning，SCPR）用于数据增量问题。该算法使用原型分类器提供的原型重构损失函数以及监督对比损失函数训练神经网络，并将结合样本回放降低模型在增量环境中的遗忘程度。卷积神经网络被视为特征提取器，并从输入数据生成嵌入特征。监督的原型分类器能够赋予神经网络不使用 Softmax 分类层也能够进行分类的能力，并与网络模型相互提供训练目标以共同训练。考虑到原型根据相似性度量来预测标签，网络应该充分聚集同类嵌入特征，并分离异类嵌入特征，以保证嵌入特征到同类原型的距离与其他原型相比最小。因此，

具有相同目标的监督对比损失以及同时使用的数据增加被用于加速网络的收敛，使得网络学习到更好的嵌入特征空间。从增量学习环境角度考虑，原型重构损失与原型分类器在训练神经网络的同时，在原型中有效地保留了已经学习过类别的信息。原型向量可以视为网络对旧类别数据提取的特征，并利用原型重构损失的训练迫使特征接近它们对应的原型以减轻了嵌入特征因新任务学习产生的漂移。对于本章工作的具体贡献，可总结如下：

- 采用单原型学习框架替代神经网络的分类层进行标签预测，并提出原型重构误差对卷积神经网络进行训练，促使输入特征在嵌入空间中能够聚集在原型向量的附近以便于分类；
- 结合回放技术、对比损失函数形成监督对比与原型重构学习算法，降低模型在增量环境中的遗忘程度，提升网络的表征学习能力；
- 实验章节中通过公开数据集在平衡与非平衡数据流上的指标评估结果验证了本章工作在数据增量问题上的有效性。

5.3 本章工作

5.3.1 数据增量问题定义

数据增量学习问题下，由于训练和测试是平行进行的，因此分别定义训练阶段 t 和测试阶段 te。模型在第 t 阶段接收来自训练数据流 $\mathcal{S}_{train}$ 的数据 $\mathcal{B}_t = \{(\boldsymbol{x}_i, y_i)\}_t$，在第 l 阶段接受来自测试数据流 $\mathcal{S}_{test}$ 的测试数据批次 $\mathcal{B}_{te} = \{(\boldsymbol{x}_l, y_l)\}_{te}$。可以看出，D 模型需要像类增量模型一样对不断扩展的输出空间进行预测。但与类增量不同的是，但并不是每个新的训练阶段 t 都会接受到来自新类的样本。因此，该问题中的训练和预测模型输出空间可以写成 $\{\mathcal{Y}_t\} \subseteq \{\mathcal{Y}_{t+1}\}$，$\{\mathcal{Y}_{te}\} \subseteq \{\mathcal{Y}_{te+1}\}$。在训练中，模型每次观察到的训练数据批大小表示为 N_b，将每次新数据到来时在重放样本内存中采样的样本数量表示为 N_s。

需要注意的是，在线持续学习（Online Continuous Learning）[165] 是不同的类增量学习模式。在该学习条件下，模型同样持续接受少量的数据批。然而，该学习问题定义仍然遵循训练-测试的划分任务边界的学习模式；另一方面，在线持续学习要求模型仅在同一数据批上训练一次，而数据增量学习放松了这个约束，以使得模型能够在同一个数据批上训练多次取得更好的效果。

5.3.2　单原型学习框架与原型重构损失函数

前面章节已经分析提到，使用原型分类器替代深度神经网络的分类层能够提升其在增量学习问题中的性能。此时，将深度神经网络视为特征提取器并记为$Enc(\cdot)$。由于神经网络参数与原型集合均为需要训练的参数，因此原型集合与特征提取器将协同进行训练，即使用原型损失函数更新网络参数，同时也使用网络所提取的嵌入特征更新原型分类器。本节中的原型分类器中仅使用单个原型。从统计学的角度考虑，选取单个原型来表示一个类别的数据趋势，那么采用均值是一种常用的方法，即对于类别c的均值原型计算方式为$\boldsymbol{p}_c = mean(Enc(\boldsymbol{x}_i)), i \in \{i|y_i = c\}$。而从重构误差的角度考虑，那么仅有同类别的嵌入特征能够激活该原型，因此重构误差中仅有相同类别的原型系数为1，其他原型的系数为0，具体来说原型分类器的学习目标为降低同类原型的重构误差，即形成**单原型学习框架**：

$$\min_{\mathcal{P}} \left\| \boldsymbol{x}_t - \boldsymbol{p}_{y_t} \right\|_2^2. \tag{5-1}$$

那么采用梯度下降法进行原型的更新学习时，更新公式则为上述重构误差函数的导数，即：

$$\boldsymbol{p}_{y_t}^* = \boldsymbol{p}_{y_t} + \alpha(\boldsymbol{x}_t - \boldsymbol{p}_{y_t}). \tag{5-2}$$

其中α为更新时的学习率。由于对比学习对嵌入特征进行了标准化操作，因此在原型进行分类时也应当进行相应的标准化操作。考虑到每次输入批次有多个嵌入特征可以进行更新，因此上式对于原型分类器的更新公式在具体网络训练中为：

$$\begin{aligned} \boldsymbol{p}_c &= \alpha \boldsymbol{p}_c + (1-\alpha)\frac{1}{|I_c|}\sum_{i \in I_c} \boldsymbol{z}_i \\ \boldsymbol{p}_c &= \frac{\boldsymbol{p}_c}{\| \boldsymbol{p}_c \|_2}. \end{aligned} \tag{5-3}$$

其中I_c为类别c的样本索引集合，$\boldsymbol{p}_c^t$则为训练阶段t时的c类别所属原型。

这两种原型分类器分别有其优缺点：更新原型时通常会采用较低的学习率来防止原型振荡，因此其对于特征的漂移相对重新计算不敏感，而在最终分类时可能会不够准确；相应地原型保留了一部分旧任务的特征分布，可以通过蒸馏损失等损失来阻止网络遗忘旧的知识。从预测的角度考虑，重新计算的结果与内存

中样本的数量很有关系，如果每个类别的样本数量过少，可能该样本难以很好表示整个类别的分布。

将类别 c 的原型记为 $\boldsymbol{p}_c$，对所有嵌入特征进行归一化，那么在 C 分类问题上，原型分类器对于特征 $Enc(\boldsymbol{x}_i)$ 的分类方式为

$$\begin{aligned} \hat{y}_i &= \max_{c=1,\ldots,C} \frac{Enc(\boldsymbol{x}_i) \cdot \boldsymbol{p}_c^T}{\sum_j Enc(\boldsymbol{x}_i) \cdot \boldsymbol{p}_j^T} \\ &= \max_{c=1,\ldots,C} Enc(\boldsymbol{x}_i) \cdot \boldsymbol{p}_c^T. \end{aligned} \tag{5-4}$$

即嵌入特征将被分类为相似度最高的原型的类别。

当模型已有拥有学习好的原型向量集合时，为降低原型对输入特征的重构误差应使得输入的嵌入特征更靠近相同类别的原型，并据此来训练特征提取网络。采用交叉熵损失来衡量同类嵌入特征分布与原型特征分布的差异，则该损失可作为神经网络训练的损失函数，在本文中称为**原型重构损失函数（Prototype Reconstruction Loss，PRL）**。使用 $\boldsymbol{z}_i = Enc(\boldsymbol{x}_i)$ 表示进行卷积神经网络提取的特征向量，则上述损失函数可以表示为：

$$\mathcal{L}_{PRL}(\boldsymbol{z}_i) = -\log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{p}_{y_i}^T / \tau)}{\sum_j \exp(\boldsymbol{z}_i \cdot \boldsymbol{p}_j^T / \tau)}. \tag{5-5}$$

尽管分别对特征提取模块和原型分类模块进行分析，但是在神经网络的训练初期嵌入特征输出完全随机，那么其产生的原型将会重合，无法起到分离异类特征作用。一种思路是进行交替优化，随着网络的训练逐渐更新原型，同时也使用更新的原型进行网络损失的计算，这种思路往往需要等待原型逐渐收敛，因此所花费时间较长。除此之外还有另一种思路，即同时使用原型损失函数以及其他损失函数同时进行训练，加速特征提取网络的收敛。本文采用对比学习损失函数[166] 与上述原型重构损失函数共同进行训练。该函数能够使得同类嵌入特征相互靠近，异类特征相互远离，可以使得同类嵌入特征更加聚集。从直觉上分析，这有利于减少原型重构的误差，提升分类性能，同时也无需线性分类层即可进行训练。

基于上述原型学习框架与损失函数，本章提出的 SCPR 算法的训练流程如图5-1所示。在训练阶段 t，从训练数据流 $\mathcal{S}$ 中采样得到的训练数据批 $\mathcal{B}_t$ 以及从

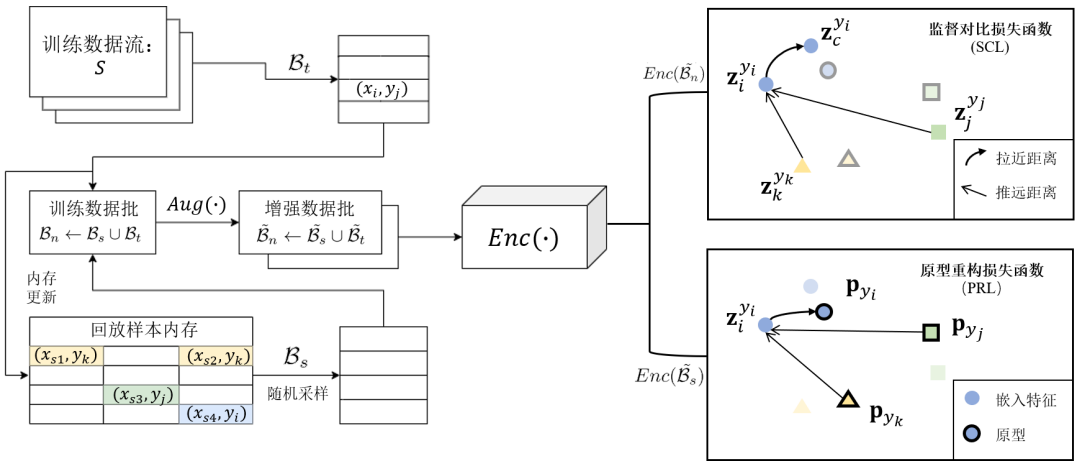


图 5-1　本章工作的训练流程图。

回放样本内存中随机采样得到的回放样本批 $\mathcal{B}_s$ 一同输入模型，通过 $Aug(\cdot)$ 模块能够从每张图像中产生两张增强图像，并最后组合为增强数据批 $\tilde{\mathcal{B}}_n$ 输入卷积神经网络。模型通过特征提取器 $Enc(\cdot)$ 获得嵌入特征，并用于计算监督对比损失函数以及原型重构损失函数对网络进行训练。最后，回放内存使用新观察到的数据样本 $\mathcal{B}_t$ 根据更新规则进行更新。此外，原型分类器也根据新输入的嵌入特征进行更新。接下来的章节中将对其他算法模块进行介绍。

5.3.3　卷积神经网络训练

对于用于特征提取的卷积神经网络，数据增量模型使用输入数据流来逐步对网络参数进行更新，从而使其能够提取出具有可分性的嵌入特征。监督对比损失能够加速神经网络训练的收敛速度，使得嵌入特征在特征空间中更为聚集。在数据的特征提取阶段，数据增强（data Augmentation）模块 $Aug(\cdot)$ 和特征提取网络 $Enc(\cdot)$ 被用于对数据进行处理。

在训练阶段 t 中，模型观察到训练数据批 $\mathcal{B}_n = \{(\boldsymbol{x}_i, y_i) | i = 1, \cdots, N\}$，其中包括新输入的训练样本批 $\mathcal{B}_t$ 和回放样本批 $\mathcal{B}_s$，且 $N = N_b + N_s$。数据增强模块对每个图像样本进行反转、裁剪、旋转等一系列随机操作后随机生成两张与原图像大小相同的增强图像。最终生成的增强训练数据批为 $\tilde{\mathcal{B}}_n = \{(\tilde{\boldsymbol{x}}_i, y_i) | i = 1, \cdots, 2N\}$，其中 $\tilde{\boldsymbol{x}}_i$ 和 $\tilde{\boldsymbol{x}}_{2i}$ 是 $\boldsymbol{x}_i$ 的两张增强图像，增强图像也被称为视图（view）。

增强的训练数据批 $\tilde{\mathcal{B}}_n$ 接下来将被特征提取模块 $Enc(\cdot)$ 处理来产生嵌入特征 $\boldsymbol{z}_i = Enc(\tilde{\boldsymbol{x}}_i)$，其中 $\tilde{\boldsymbol{x}}_i \in \tilde{\mathcal{B}}_n$。网络所输出的嵌入特征需要进行标准化处理，

即使得 $\boldsymbol{z}_i = \boldsymbol{z}_i / \|\boldsymbol{z}_i\|_2$，该操作在论文中验证能够提升最终分类的准确性[85]。为使得同类别嵌入特征尽可能靠近，异类嵌入特征尽可能远离，对比损失函数将通过如下方式进行计算：

$$\mathcal{L}_{SC}(\boldsymbol{z}_i) = \sum_{i \in I} \frac{-1}{|C(i)|} \sum_{c \in C(i)} \log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_c / \tau)}{\sum_{a \in A(i)} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_a / \tau)}. \tag{5-6}$$

其中 $I = \{1, \cdots, 2N\}$ 是增强数据批 $\tilde{\mathcal{B}}_n$ 的索引集合，$C(i) = \{c \in A(i) | \tilde{y}_c = \tilde{y}_i\}$ 为属于同一类别的样本索引集合，$A(i) = I \backslash \{i\}$ 为增强数据批的索引 I 中除第 i 个输入样本的索引以外的所有样本的索引集合。

在使用监督对比损失函数的基础上，由于使用原型分类器进行嵌入特征的分类，因此第5.3.2节提出使用原型重构误差 $\mathcal{L}_{PRL}$ 同时进行网络的训练。最终卷积神经网络的训练损失函数 $\mathcal{L}$ 为两种损失函数的总和，即：

$$\mathcal{L} = \mathcal{L}_{SC} + \mathcal{L}_{PRL}. \tag{5-7}$$

5.3.4 回放样本内存

保存部分样本在新数据到来时进行重新训练可以使得网络保持在旧类数据上的性能，这种训练方式被称为回放。在实际训练中，模型将使用限定容量的内存来保存样本以避免随着训练的进行使用过多的储存空间，该内存容量表示为 $|M|$。内存根据当前观察到的类的数量对每个类别的回放样本数量进行平均划分，当新的类出现时，内存将压缩所有类别的样本数量，并删去超出容量的样本。根据上述策略，不同类别的样本数量在内存中是平衡的，因此在数据回放中缓解了数据增量问题下的不平衡问题。在每个训练阶段 t，模型从内存中随机无放回采样出回放样本 $\mathcal{B}_s$，并与输入样本共同进行训练。采样范围包括内存中的所有数据类别，包括当前正在训练的类别。随机方法与采用全部回放样本或选取部分关键样本的技术相比节约了网络训练成本或是样本选择的成本。考虑到在数据增量设置下训练批较小，则在同样数据量的情况下训练阶段 t 会相应增加，则回放的次数也会相应增加，因此节约回放样本的采样成本所提升的效率相比普通类增量学习环境中更为明显。

内存中更新回放样本的策略为：增加回放样本时如果内存容量未满，则可以

算法 5.4 数据增量算法的训练和预测流程

训练阶段:

输入：训练数据流 $\mathcal{S}_{train}$

参数：回放样本内存 M, 原型更新动量 α, 训练数据批大小 N_b, 回放样本批大小 N_s

初始化内存 $\mathcal{M} = \phi$
for 模型接受输入数据批 $\mathcal{B}_t = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_{N_b}, y_{N_b})\} \sim \mathcal{S}$ **do**
    从内存 $\mathcal{M}$ 中随机采样 N_s 个回放样本形成回放样本批 $\mathcal{B}_s$，并与输入数据批合为训练样本批 $\mathcal{B}_n$
    对训练样本批进行数据增强，得到增强数据批 $\tilde{\mathcal{B}}_n$
    使用公式（5-7）来计算模型的损失函数，包括对比训练损失以及原型重构损失，使用梯度回传对网络进行训练
    使用公式 (5-3) 来更新原型
    使用5.3.4中提到的策略更新回放样本内存
end for

输出：神经网络模型 Enc，回放样本内存 $\mathcal{M}$，原型集合 $\mathcal{P}$

测试阶段:

输入：测试数据流 $\mathcal{S}_{test}$, 测试样本批大小 N_{te}

for 模型接受输入测试样本批 $\mathcal{B}_{te} = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_{N_{te}}, y_{N_{te}})\} \sim \mathcal{S}_{test}$ **do**
    初始化预测标签集合 $\hat{\mathcal{Y}}_{te} = \phi$
    for $j = 1, \cdots, N_{te}$ **do**
        使用最近原型分类器预测第 j 个样本对应的标签 $\hat{y}_j$
        $\hat{\mathcal{Y}}_{te} = \hat{\mathcal{Y}}_{te} \cup \{\hat{y}_j\}$
    end for
end for

输出：预测标签集合 $\hat{\mathcal{Y}}_{te}$

直接加入；若容量已满，则将以等概率随机生成一个 $[0, N_{seen} - 1]$ 范围内的整数 r。若 $r < |M_c|$，则替换第 r 个已保存的样本；否则不进行替换。其中 $|M_c|$ 表示类 c 所属的回放样本内存容量，N_{seen} 是属于类 c 的已训练样本的数量。

结合上述所有模块，算法 5.4总结了本章工作的训练和测试过程。

5.4　实验验证

本节中将本章工作与其他可在数据增量设置下工作的增量学习算法在 MNIST、CIFAR10、CIFAR100 以及 MiniImageNet 四个图像数据集上进行了比较。所有模型从训练数据流 $\mathcal{S}_{train}$ 中在每个训练阶段 t 接收批大小为 $N_b = 10$ 的数据流。为了与其他使用线性分类层的增量学习算法进行比较，训练样本仍然按照类别进行划分并根据任务顺序进行排列，即在训练完成当前任务中的所有数据后，再训

练属于新任务类别的数据。实验中的分类结果由 5 次运行平均得到，每次分别采用不同的随机种子并指定 $seed = 1, 2, 3, 4, 5$，该随机种子生成随机的样本序列和神经网络的初始化参数。考虑到每次进行训练的数据量较少，因此每个数据批次仅训练 5 个轮次。模型采用随机梯度下降优化器（SGD），其更适合增量学习的训练[93]。

5.4.1 实验设置

本章对上述四个数据集进行任务划分后进行实验，并分别将其标记为 Split-MNIST（S-MNIST）、Split-CIFAR10（S-CIFAR10）、Split-CIFAR100（S-CIFAR100）以及 Split-MiniImageNet（S-MiniImageNet）表示这些数据集按照类别划分成不同的任务序列，每个任务的类别集合之间没有交集。Split-MNIST （S-MNIST）[167] 由 10 类 60000 张 28×28 的黑白手写数字图像组成，其中包括划分好的 50000 个训练样本和 10000 个测试样本。Split-CIFAR10 （S-CIFAR10）和 Split-CIFAR100（S-CIFAR100）[168] 分别包括 10 个类别和 100 个类别的 32×32 RGB 彩色图像，每个数据集都包含包括 50000 训练图像和 10000 测试图像。Split-MiniImageNet[169] 包含 100 个类别的 84×84RGB 彩色图像。

在平衡实验设置中，S-MNIST 被分成 5 个任务，每个任务包括来自 2 个不相交类的大约 10000 个样本；S-CIFAR10 被分成 5 个任务，每个任务中有来自 2 个类的 10000 个样本；S-CIFAR100 被分成 20 个任务，每个任务中有来自 5 个类的 2500 个样本；S-MiniImageNet 被分为 10 个任务，每个任务中有来自 10 个类的 5000 个样本。在当前任务的训练结束之后，将对所有模型在学习过的类别的测试数据上的分类性能进行评估。

在非平衡实验设置中，部分任务减少了样本数量以形成样本数量不平衡的设置。数据集的任务划分按照平衡实验进行，但每个任务中的训练数据量将产生变化。其中一个任务的样本量正常，而其他任务的训练样本数量被缩减，本文实验中将未进行缩减的正常任务表示为 T_i，并且在实验中依序指定不同的任务为该正常任务进行结果对比。对于 S-MNIST，正常任务有 2000 个样本而其他任务有 200 个样本，正常任务分别为 $T_i = \{1, 2, 3, 4, 5\}$；对于 S-CIFAR10，特殊任务有 4000 个样本，而其他任务有 2000 个样本，$T_i = \{1, 2, 3, 4, 5\}$；对于 S-CIFAR100，特殊任务有 2500 个样本，而其他任务有 1000 个样本，$T_i = \{1, 5, 10, 15, 20\}$；

对于 S-MiniImageNet，特殊任务有 5000 个样本，而其他任务有 1000 个样本，$T_i = \{1, 3, 5, 7, 9\}$。

卷积神经网络的骨架为对比其他模块的性能基本与其他论文中的实验设置保持一致。在 S-MNIST 数据集上，$Enc(\cdot)$ 为 MLP 模型，具有 400 个神经元的两个隐藏层用于平衡的 S-MNIST 数据集[163][83]，100 个神经元的用于不平衡的 S-MNIST 数据集[162][83]。S-CIFAR10、S-CIFAR100 以及 S-MiniImageNet 数据集中使用 ResNet18 模型[170]，并在实验中去掉了线性分类层。对于 S-MNIST 数据集，数据增强模块使用 Pytorch 实现的随机裁剪、旋转 (旋转范围为 $[0, 10°]$) 和仿射的组合以及来自[171] 的切割策略。由于数字图像经过大角度旋转，翻转后一部分数字将变成另一个数字甚至成为无法辨识的数字，因此对于该数据集没有采用翻转策略，旋转的角度也比较小，即在 0 度至 20 度范围内随机。在 S-CIFAR10、S-CIFAR100 以及 S-MiniImageNet 数据集上，$Aug(\cdot)$ 模块则采用多种图像操作随机产生增强图像，具体操作方法来自于[171]。

在平衡设置中，回放内存容量在 S-MNIST 数据集上设置为 $|M| = 2000$，在 S-CIFAR10 数据集上设置为 $|M| = 1000$，在 S-CIFAR100 与 S-MiniImageNet 上设置为 $|M| = 5000$。在非平衡设置中，由于每个类别的样本数量被减少了很多，因此将 MNIST 数据集的回放内存减少到 300。对于每个阶段训练时的回放样本数量，S-CIFAR10、S-CIFAR100、S-MiniImageNet 数据集上设置为 $N_s = 100$，S-MNIST 上设置为 $N_s = 300$。

本次实验采用了 11 个增量学习算法与本章工作进行比较，其中部分在 S-MNIST、S-CIFAR10 以及 S-CIFAR100 数据集上的结果引用来自论文[83] 的实验结果，并在本文中保持了与其相同的实验设置。S-MiniImageNet 结果保持为 S-CIFAR100 的设置，结果为自行实验得到。上述引用结果结果表中用 * 对方法进行了注释。其中，iid-offline 模型使用传统交叉熵损失函数进行批量训练的，即模型从头开始就接触了所有类别的数据，这个结果可以看作是增量学习结果的上界。finetune 模型则与其余模型采用了相同的数据增量实验设置进行训练，但是未采用任何对抗遗忘的技术，因此可以视为增量学习结果的下界。iCaRL 和 GEM 是使用回放技术的类增量方法。对于数据增量模型，本次实验中采用了 Reservoir，MIR，GSS 以及 CoPE 方法，它们都使用了样本回放的方式来对抗遗忘。此外，CN-DPM 作为一种参数隔离的方法也能够适用于数据增量环境。连续

表 5-1 13 种方法在数据平衡的数据增量环境下，S-MNIST、S-CIFAR10、S-CIFAR100、S-miniImageNet 四个数据集上的准确率结果。该准确率为所有数据训练完成后在全部任务的测试数据集上的结果，每个结果均为 5 次随机种子进行训练后预测结果的平均。

	S-MNIST	S-CIFAR10	S-CIFAR100	S-MiniImageNet
iid-offline*	98.44 ± 0.02	83.02 ± 0.60	50.28 ± 0.66	51.33 ± 0.20
finetune*	19.75 ± 0.05	18.55 ± 0.34	3.53 ± 0.004	1.22 ± 0.03
GEM*	93.25 ± 0.36	24.13 ± 2.46	11.12 ± 2.48	4.46 ± 2.37
iCaRL*	83.95 ± 0.21	37.32 ± 2.66	10.80 ± 0.37	5.45 ± 0.78
DN-CPM*	93.23 ± 0.09	45.21 ± 0.18	20.10 ± 0.12	13.63 ± 0.46
reservoir*	92.16 ± 0.75	42.48 ± 3.04	19.57 ± 1.79	13.35 ± 2.07
MIR*	93.20 ± 0.36	42.80 ± 2.22	20.00 ± 0.57	15.36 ± 3.73
GSS*	92.47 ± 0.92	38.45 ± 1.41	13.10 ± 0.94	10.05 ± 2.96
CoPE*	93.94 ± 0.20	48.92 ± 1.32	21.62 ± 0.69	16.75 ± 1.08
SCR	96.58 ± 0.22	71.29 ± 1.35	45.64 ± 0.60	31.23 ± 0.77
SCL	96.62 ± 0.45	71.07 ± 1.70	45.43 ± 0.35	27.98 ± 1.09
SCPR	$\mathbf{96.92 \pm 0.37}$	72.13 ± 1.50	46.78 ± 0.57	32.24 ± 0.44
SCPR-r	96.81 ± 0.20	$\mathbf{73.27 \pm 0.56}$	$\mathbf{47.08 \pm 0.56}$	$\mathbf{33.10 \pm 0.56}$

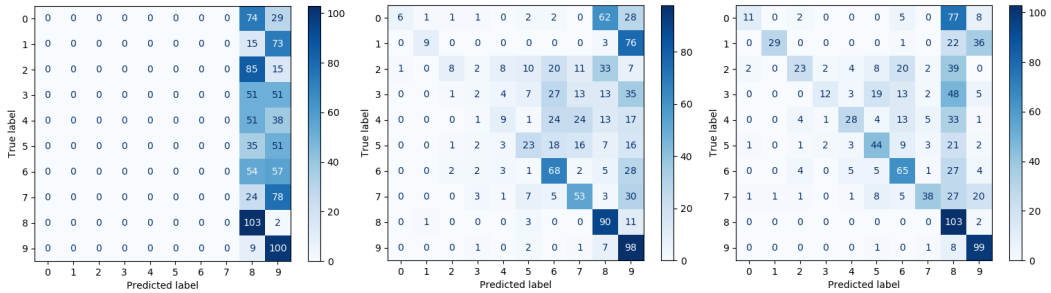
对比学习模型 SCR 和 SCL 也将与本章工作在数据增量的环境下进行对比。具体来说，SCR 使用回放样本内存中的样本来计算嵌入特征的均值作为最近原型分类器，而 SCL 模型则采用与本章算法相同的原型更新规则，同时这两种算法都不计算原型交叉熵损失。在下述实验中，三种对比方法使用相同的参数设置。在数据不平衡的设置下，CoPE，GSS，MIR，Reservoir 以及两个基于对比学习的方法参与了比较。本章工作在实验中标记为 SCPR 与 SCPR-r，其中 SCPR 指训练完成后直接使用更新的原型分类器进行标签预测，而 SCPR-r 则指代在训练完成后使用回放样本内存中的所有样本计算的均值原型分类器对标签进行预测。实验中所使用的指标为预测准确率（%），其计算方式为 $accuracy = \frac{N_{correct}}{N_{test}}$，其中 $N_{correct}$ 和 N_{test} 分别为预测正确的测试样本个数和总体测试样本个数。

5.4.2 结果分析

数据平衡环境下的实验结果记录在表5-1中。首先对于两个平凡的训练方案，即在所有数据上直接进行批量训练的 iid-offline 以及在数据增量环境中不采用任何对抗遗忘手段的 finetune，其结果与前面理论上的分析相同。iid-offline 的结果好于任一数据增量算法，其表现可以作为数据增量算法结果的上限。由于数据增量算法的不断完善，可以看出其结果在逐渐接近批量训练的结果。而不采用任何对抗遗忘手段时，算法的预测表现则下降十分明显，可以作为所有训练方案的下

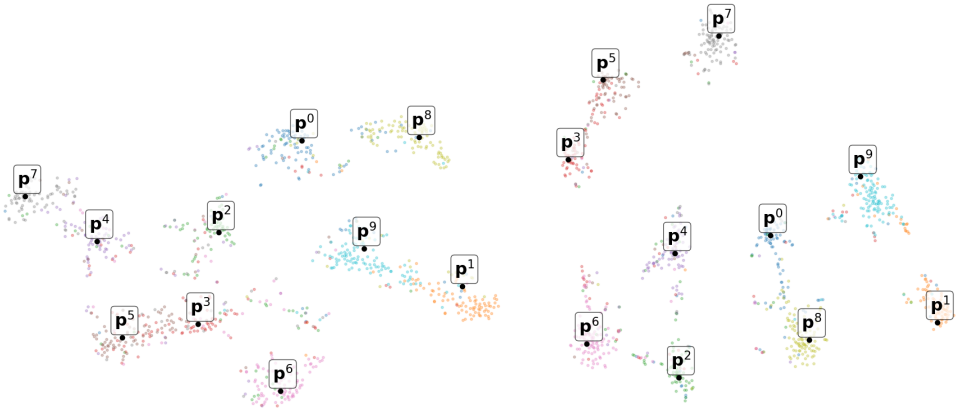
表 5-2　7 种数据增量算法在不平衡设置下的 S-MNIST，S-CIFAR10，S-CIFAR100、S-MiniImageNet 数据集上的结果。其结果为所有数据训练完后在测试数据集上的结果。任务 T_i 表示属于该任务的训练数据数大于其他任务的数据量。均值结果为 5 次任务结果的平均，所有结果都为 5 次随机种子初始化后的运行结果。

数据集	任务	SCPR-r	SCPR	SCL	SCR	CoPE*	GSS*	MIR*	Reservior*
S-MNIST	T_1	80.2 ± 2.3	79.8 ± 3.0	79.6 ± 3.2	81.6 ± 3.2	83.4 ± 2.0	75.9 ± 3.2	64.8 ± 5.1	64.2 ± 2.3
	T_2	85.1 ± 2.2	84.8 ± 1.8	84.7 ± 0.7	84.5 ± 1.2	84.5 ± 1.6	78.5 ± 2.7	67.4 ± 3.2	65.5 ± 4.6
	T_3	87.4 ± 0.9	85.7 ± 2.2	85.1 ± 1.0	87.6 ± 1.1	85.1 ± 0.6	81.5 ± 2.3	72.4 ± 3.0	72.1 ± 4.0
	T_4	89.9 ± 0.8	89.2 ± 1.0	89.1 ± 0.5	88.7 ± 0.8	84.8 ± 1.0	79.5 ± 0.6	72.6 ± 3.1	73.6 ± 2.4
	T_5	91.0 ± 0.6	90.9 ± 1.7	90.9 ± 0.2	85.3 ± 1.1	84.0 ± 1.3	79.1 ± 0.7	77.2 ± 3.4	73.2 ± 4.0
	平均表现	$\mathbf{86.7 \pm 4.3}$	86.1 ± 4.3	85.9 ± 4.4	85.5 ± 2.8	84.4 ± 0.7	78.9 ± 2.0	70.9 ± 4.9	69.7 ± 4.5
S-CIFAR10	T_1	56.6 ± 3.7	52.9 ± 3.3	40.2 ± 2.2	51.5 ± 3.0	39.0 ± 1.3	32.3 ± 3.0	32.6 ± 3.0	35.5 ± 3.4
	T_2	57.1 ± 2.3	56.1 ± 5.5	36.7 ± 2.2	50.4 ± 4.5	35.3 ± 2.6	28.3 ± 0.4	28.3 ± 0.4	29.3 ± 2.8
	T_3	59.3 ± 1.7	56.1 ± 3.8	42.9 ± 4.7	51.8 ± 2.4	36.2 ± 2.5	29.5 ± 1.5	29.5 ± 1.5	31.4 ± 2.1
	T_4	61.5 ± 3.2	59.7 ± 3.8	47.7 ± 3.0	55.9 ± 4.0	39.1 ± 2.4	34.6 ± 1.3	34.6 ± 1.3	32.1 ± 0.6
	T_5	64.2 ± 1.7	62.8 ± 2.8	56.5 ± 5.5	60.6 ± 2.3	37.3 ± 3.3	28.3 ± 2.4	28.3 ± 2.4	28.8 ± 1.9
	平均表现	$\mathbf{59.7 \pm 3.2}$	57.5 ± 3.8	44.8 ± 7.7	54.0 ± 4.2	37.4 ± 1.7	30.6 ± 2.8	29.6 ± 2.3	31.4 ± 2.7
S-CIFAR100	T_1	39.6 ± 0.8	40.8 ± 0.9	37.3 ± 0.6	35.1 ± 0.7	18.2 ± 0.6	10.2 ± 3.0	18.4 ± 0.9	11.1 ± 0.6
	T_2	39.9 ± 0.8	40.6 ± 1.2	37.5 ± 1.1	35.6 ± 0.8	18.5 ± 1.3	10.7 ± 0.4	17.6 ± 0.9	11.5 ± 1.4
	T_3	40.0 ± 0.8	41.7 ± 0.3	37.6 ± 0.7	35.2 ± 0.8	19.2 ± 0.9	11.1 ± 1.5	17.8 ± 0.7	11.9 ± 0.7
	T_4	40.1 ± 1.2	40.9 ± 1.2	37.0 ± 1.3	35.6 ± 0.8	18.7 ± 0.6	11.1 ± 1.3	17.8 ± 0.9	12.1 ± 0.8
	T_5	40.9 ± 0.9	41.1 ± 0.9	37.6 ± 0.9	36.5 ± 0.6	18.5 ± 1.5	11.1 ± 2.4	17.6 ± 0.4	12.5 ± 1.1
	平均表现	40.1 ± 0.5	$\mathbf{41.0 \pm 0.4}$	37.4 ± 0.3	35.6 ± 0.6	18.6 ± 0.4	10.8 ± 0.4	17.8 ± 0.3	11.8 ± 0.5
S-MiniImageNet	T_1	24.3 ± 1.2	23.8 ± 2.3	21.4 ± 0.4	23.1 ± 1.9	13.7 ± 0.8	8.2 ± 2.5	13.2 ± 0.9	9.2 ± 0.3
	T_2	22.7 ± 2.9	23.3 ± 1.3	20.5 ± 1.7	22.4 ± 1.1	13.9 ± 0.3	9.7 ± 2.1	14.3 ± 1.0	10.4 ± 1.1
	T_3	25.5 ± 0.6	23.6 ± 1.6	22.4 ± 0.8	23.7 ± 1.6	12.8 ± 2.2	10.3 ± 0.7	12.4 ± 1.1	10.1 ± 0.7
	T_4	23.8 ± 1.0	22.9 ± 1.3	22.7 ± 0.6	21.1 ± 0.4	13.0 ± 0.5	9.1 ± 1.2	11.5 ± 1.9	11.5 ± 0.8
	T_5	24.8 ± 0.9	24.0 ± 0.8	23.6 ± 1.0	21.5 ± 0.9	11.5 ± 0.9	10.5 ± 2.0	12.8 ± 1.2	10.9 ± 0.9
	平均表现	$\mathbf{24.2 \pm 1.0}$	23.5 ± 0.4	22.1 ± 1.2	22.3 ± 1.0	13.0 ± 0.9	9.56 ± 0.9	12.8 ± 1.0	10.4 ± 0.9



(a) finetune 方法　　**(b) CoPE 方法**　　**(c) SCPR-r 方法**

图 5-2　使用混淆矩阵展示 $seed = 1$ 时使用 finetune 方法、CoPE 方法以及 SCPR-r 方法在 Split-CIFAR10 上进行 5 个任务的训练后，最终在 10 个类别数据上进行测试时的分类性能。纵轴为测试数据的真实标签，横轴为测试数据的预测标签，颜色越深表示符合当前预测-真实标签对的测试样本数量越多。



(a) S-MNIST 数据集的测试数据与原型的 t-SNE 可视化结果。

(b) S-CIFAR10 数据集的测试数据与原型的 t-SNE 可视化结果。

图 5-3 本章工作在 S-MNIST 数据集与 S-CIFAR10 数据集上的测试数据嵌入特征与原型分类器使用 t-SNE 进行可视化的结果。

限。GEM 和 iCaRL 是较早提出的图像增量学习算法，其对抗遗忘的经典技术在最新的增量学习算法中仍然常常被使用，但遗憾的是其算法在数据增量算法中表现较差。在其余数据增量算法中，参数隔离方案 CN-DPM 与使用原型分类器的 CoPE 算法在三个数据集上都表现良好。CN-DPM 使用扩张型的专家系统进行数据的分类，因此避免了后续任务中数据对已经学习过知识的污染，但是如何将数据分配到对应专家模型可能仍是该算法需要解决的瓶颈；CoPE 算法也使用了拉进同类嵌入特征、远离异类嵌入特征的思想，但其评估表现仍然不如对比损失函数。以 SC 开头的四个对比算法都采用了监督对比损失函数进行网络的训练，从结果可以看出，该损失函数在数据增量问题上表现优于其他损失函数。与除三个对比损失算法外结果最好的方法 CoPE 相比，基于 SCR 的模型在 S-MNIST 数据集上提高了约 2% 的准确率，在 S-CIFAR10 数据集上提高了约 22% 的准确率，在 S-CIFAR100 数据集上提高了至少 24% 的准确率，在 S-MiniImageNet 数据集上提高了至少 11%。可以认为，结合监督对比损失与最近原型分类器在数据增量问题上表现优秀。

表5-2中报告了数据增量算法在不平衡数据集设置下的表现情况，CoPE、GSS、MIR 和 Reservoir 的结果来自于来自[83] 的补充材料。与平衡设置相似，三个基于监督对比损失的方法具有明显的性能优势，具体为在 S-MNIST 数据集上超过了最好方法 CoPE 大约 1% 的准确率，在 S-CIFAR10 数据集上提升了 7% 的准确率，在 S-CIFAR100 数据集上提升了 17% 的准确率，以及在 S-MiniImageNet 上提升

了 9%。但需要注意的是，基于监督对比损失的方法在该设置中相比平衡设置所具有的优势更小，因此数据样本缺乏时对比损失函数对分类性能的提升会降低。

通过比较表5-1和表5-2中 SCR、SCL、SCPR 与 SCPR-r 的结果，可以分析不同实验设置下原型分类器以及原型重构损失函数的作用。从结果上来看，在数据平衡的环境下，使用更新原型结合原型交叉熵损失的 SCPR 算法表现优于其他两种对比学习算法。具体来说，与 SCR 方法相比，SCPR 方法在 S-MNIST 数据集上准确率提高了 0.34%，在 S-CIFAR10 数据集上准确率提高了 0.84%，在 S-CIFAR100 数据集上准确率提高了 1.14%，而在 S-MiniImageNet 数据集上准确率提升了 1.01%；与 SCL 方法相比，SCPR 方法的准确率分别提高了 0.30%、1.06%、1.35%、4.26%。在数据非平衡的实验设置中，除去 S-MNIST 数据集上的 T_1，T_3 任务设置，SCPR 算法在几乎所有的任务中都超越了其他两个对比算法，并且在多个任务的平均准确率上取得了最优的结果。具体来说，SCPR 方法的平均准确率相比 SCR 方法提高了 0.6%、3.5%、5.4%、1.2%，相比 SCL 方法提高了 0.2%、12.7%、3.6%、1.4%。因此，可以发现更新原型配合原型交叉熵损失性能更好，并且该模块在数据不平衡时提升了更多的准确率，特别是在 S-CIFAR10 和 S-CIFAR100 数据集上。此外，可以看到 SCR 和 SCL 在不同数据集上互有优势。

同时，最终分类采用学习的单原型分类器还是均值计算分类器则同样可以进行一定的探讨。在上述比较的基础上，可以发现在除去平衡设置下的 S-MNIST 数据集，以及非平衡设置下的 S-MiniImageNet 数据集的大多数数据集上，最终在分类时重新计算原型分类器能够提升整体的分类性能。在平衡设置下，使用重新计算的均值原型分类器，即 SCPR-r 结果的性能在 S-CIFAR10、S-CIFAR100、S-MiniImageNet 上分别提升了 1.14%、0.30%、0.86%; 在非平衡设置下，SCPR-r 的结果在 S-MNIST、S-CIFAR10、S-CIFAR100 以及 S-MiniImageNet 上分别提升了 0.6%、2.5%、0.7%，其相比前面数值上的提升幅度基本较小。

在表格数据的基础上，图5-2进一步展示了在 CIFAR10 数据集上 Finetune 方法、CoPE 方法以及 SCPR 在进行 5 个任务的训练之后，在 10 个类别的测试数据上的分类结果。可以看出，Finetune 方法对于所有的类别都会预测为新学习的两个类别，无法预测出旧任务中的类别。CoPE 相应的有所改善，但是其预测正确的类别基本只集中在后 5 个类别中，更为早期进行学习的类别则无法进行正

表 5-3 本章工作在 S-CIFAR10 和 S-CIFAR100 数据集上不同参数设置的准确率结果，该结果同样由 5 次随机种子的实验结果平均得到。

数据集	$N_b = 10$（在线）	$N_b = 20$	$N_b = 50$	$N_b = 100$	$N_b = 200$	$N_s = 300$
S-CIFAR10	65.4 ± 3.2	70.5 ± 2.2	64.2 ± 4.4	67.1 ± 2.4	65.0 ± 2.7	73.1 ± 2.2
S-CIFAR100	31.6 ± 2.1	43.1 ± 1.3	35.3 ± 1.0	28.3 ± 1.4	26.6 ± 1.6	46.7 ± 0.7

确雨泽。SCPR 方法则能够做到所有已经学习过的类别数据都能够进行一定程度上的预测，越新学习的类别，其预测成功率基本越高。因此，本章工作确实具有更好的对抗遗忘能力。但也可以看出，该算法仍具有较大的改进空间。许多错误预测的数据都集中在了类别 9 上，说明新学习的原型仍旧占据了旧原型的空间，使得旧类别上的测试数据被错误分类到新类别的原型上。图5-3则展示了 MNIST 以及 CIFAR10 数据集的测试嵌入特征与本章工作所学习的原型分类器使用 t-SNE 进行可视化的结果。可以看出，这两个数据集的嵌入特征都能够较好地被分开，并且原型能够对数据分布进行表示。但是仍然可以看出，还有一些类别的嵌入特征未能够很好地与同类特征聚集在一起，并且原型也不总是位于每个类别数据分布的中心区域，因此导致了在最终分类时产生的谬误。考虑到在平衡的 S-CIFAR10 数据集上重新计算的 SCPR-r 算法分类效果更好，可以推测输入特征在嵌入空间中仍产生了一定的漂移，而原始的分类器综合了旧特征与新特征的分布，因此与最终学习得到的嵌入分布不完全符合。因此，如何在平衡数据增量设置下网路的稳定性-可塑性，使得分类模型更好掌握加入新类别数据后的数据分布，仍然是值得探讨的课题。

5.4.3 参数敏感性分析

在本节中进一步探讨了三个参数的不同取值，包括内存容量 $|M|$，每次模型接受的数据批大小 N_b 以及每次训练样本回放的数量 N_s 对本章工作的性能影响。

理论上讲，回放样本内存容量越大，则其保存的回放样本越多，那么样本的分布就更有可能贴近数据流 $\mathcal{S}$ 的分布。对本章工作设置不同的重放内存大小的分类结果在图5-4中报告。根据该图可以得出结论，内存大小和最终的性能是正相关的，这与理论分析中的假设一致。S-CIFAR10 数据集上的结果受容量的影响很大，尤其是 $M = 100$ 和 $M = 200$ 的设置，这表明这两种设置不适合该数据

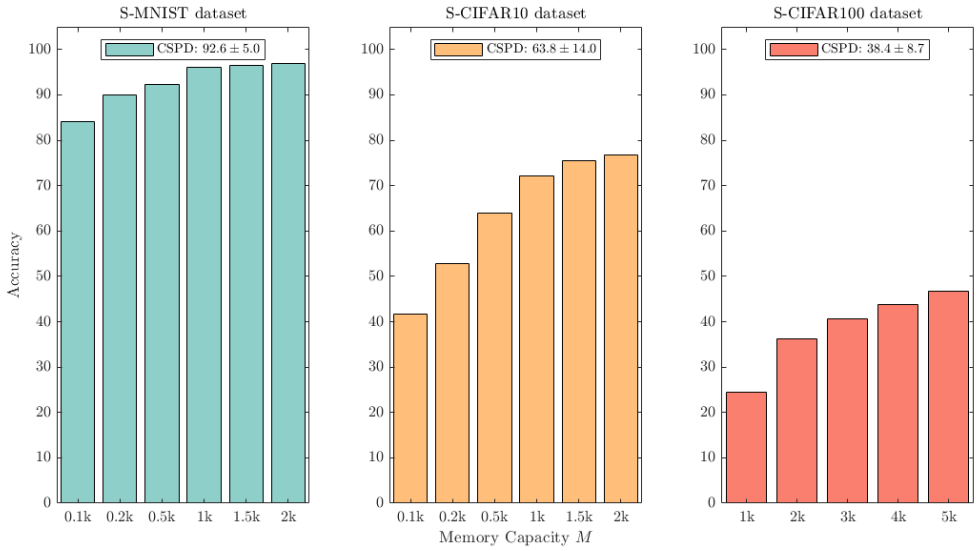


图 5-4　本章工作在 S-MNIST, S-CIFAR10, S-CIFAR100 数据集上使用不同回放样本容量 $|M|$ 参数的结果比较，所有实验结果由 5 次使用不同的随机种子的测试准确率平均后得到。

集。由于类似的原因，$M = 100$ 对于 S-CIFAR100 数据集来说也略小。

已有数据增量工作[83] 中指出准确性随着批量的增加而降低，因此本次实验对数据批量的参数 N_b 进行了不同的设置，其结果如表5-3所示。考虑到在 $N_b = 100$ 和 $N_b = 200$ 的设置中模型的训练周期大幅减少，在实验中将每个训练批次的训练周期增加到 10 以改善结果。表中的 $N_b = 10$ (在线) 表示模型只训练一次。通过实验结果可以发现，在线训练和较大的批量都由于减少了训练次数而降低了数据增量算法的性能。

表5-3的最后一列报告了 $N_s = 300$ 设置下的测试准确率。对于 $N_s = 100$ 设置，S-CIFAR10 上的性能提升，而 S-CIFAR100 上的性能保持不变。因此，更大的重放大小是否会带来更好的结果取决于训练数据集。

5.5　小结

为解决深度神经网络在动态环境中的学习问题，本章采用了单原型学习框架，提出了原型重构误差函数用于网络训练，并在此基础上提出了解决数据增量问题的监督对比与原型重构学习算法。其主要思路是协同原型学习框架对神经网络的特征提取功能提供原型重构损失进行训练，以及使用原型集合对神经网络提取的嵌入特征进行分类。在上述原型学习框架与神经网络结合的基础上，

本章模型还包含监督对比损失函数、样本回放进一步减少神经网络在数据分布产生变化时对旧类别数据的遗忘程度，使得在嵌入空间中同类特征进一步靠近，异类特征远离，从而提升网络的分类性能。在实验中采用多个数据集对本文算法与其他数据增量算法进行对比，展示了本文算法处理数据增量问题的能力；通过消融实验的对比，展示了原型学习框架的有效性；通过在非平衡数据集上的实验对比，展示了本章算法在更为严苛的学习环境中也能够拥有更好的分类性能；通过不同参数的实验结果，对实验中应该如何进行参数设置进行了讨论。

该项工作对数据增量分类性能进行了提升，然而依旧存在改进与理论分析的空间，可以作为未来的研究方向。从传统的交叉熵损失函数的角度来看，原型重构损失函数将原型视为某种标签作为训练目标，对比学习损失函数使用同类的嵌入特征作为训练目标，而传统交叉熵函数则使用标签的 one-hot 编码来作为训练目标。可以看出，原型重构损失与对比损失函数所使用的训练目标并不正交，正交性是否会影响训练的结果还需进一步进行理论分析。此外，与其他数据增量方法例如 CoPE 相比，基于监督对比损失的方法所需的回放样本数量更多。回放样本具有诸多缺点，例如会增加训练的储存和时间成本、具有一定安全泄露的风险等，因此如何减少甚至不使用回放样本还能够不降低分类性能是数据增量乃至增量学习工作的重点。最后，本章工作虽然在非平衡数据集中已经取得了较好的结果，然而仅在管理回放样本时平衡了样本的数量。是否能够在设计网络训练目标时也考虑到数据不平衡的情况，也是日后可以进行思考的方向之一。

第六章　结束语

6.1　本文总结

当前机器学习模型对分类、聚类任务已经有大量的解决方案，然而，这些解决方案通常面对独立同分布数据，并且通常为数据仅预测标签集合中的单个二值化标签。而现实世界是动态发展的，具有模糊性的，因此应当提出更为灵活的学习模型用于解决实际生活中的问题。作为一种符合人类直觉和认知模型的经典算法，原型学习具有从大量数据中提取关键信息的作用，因此其能够应用于多种学习场景中。基于重构误差的原型学习框架可以作为一种通用的表征学习方式，将数据转换到更易进行学习的表征空间中；同时原型集合能够进行灵活增删，并且通过控制原型组合的系数使得只有足够相关的输入特征才能够对原型进行激活，使得原型学习框架具有在动态环境中学习的灵活性。但是从实验可以看出，仅使用重构误差在解决具体问题上具有一定的局限性，还需要对基本的原型学习框架进行改进以及结合算法中的其他技术一同形成算法模型。因此，为解决在线增量模糊聚类问题，标签分布学习问题，以及数据增量学习问题，本文分别对应提出了模糊原型网络，非负原型基底，以及均值原型分类器三种改进的原型学习框架。而在后续章节中，本文分别对于这三种问题在三个原型学习框架的基础上提出了对应的解决方案。具体来说，本文所做的工作包含：

- 本文总结了以重构误差为基础的原型学习框架，并采用四种以重构误差为基础的原型学习框架形成的在线降维算法在多个领域的数据集实验上进行结果对比，验证了原型学习框架的有效性。可以看出，该原型学习框架形成的基本具备提取有效表征，并据此完成简单分类任务的能力。
- 针对在线增量模糊聚类问题，本文提出了基于模糊原型竞争学习框架，并结合拓扑结构提出了在线模糊聚类算法。该原型算法具有在动态环境中进行学习的能力，能够根据数据分布表示的需求自主增加或者删除原型神经元。本文对其分别进行了量化和可视化的实验验证，评估了其在在线无监

督环境中的模糊聚类结果以及聚类性能。

- 针对标签分布问题，本文提出了非负原型线性学习框架，并结合最大熵模型提出了标签分布算法。该算法对每个类标所对应的基底向量并将输入特征表示为原型基底的线性组合，同时使用最大熵模型在新的特征空间中完成对标签分布的预测训练。本文采用多个数据集以及指标对该标签分布算法进行实验验证，评估得到该算法的表现相较于已有算法均有提升。
- 针对数据增量问题，本文提出了单原型线性学习框架及原型重构损失函数，并结合卷积神经网络、回放内存提出了数据增量算法。该算法能够通过将输入特征聚集在其对应类别的原型特征附近，缓解深度神经网络在增量学习问题上的遗忘程度。本文采用多个图像数据集对多个数据增量算法的分类性能进行验证，并设置非平衡数据集考察算法对于每个类别的数据量相差较大时的表现情况，最终验证了本文提出算法的有效性。

6.2 未来工作展望

为了解决动态与模糊环境下的学习问题，本文主要关注基于原型学习框架的表征学习能力基础上的改进原型算法。通过分析原型学习算法发展的趋势，可以发现原型学习的研究仍然存在不少机遇与挑战。因此，下面本文将对原型学习中待研究的开放性问题和未来的研究方向进行展望，具体包括：

- **与深度神经网络结合的原型学习算法。**目前对于深度学习的研究热度与日俱增，深度神经网络也证实了自身作为表征学习模型所带来的算法性能提升。一方面来说，深度神经网络在许多研究问题上依旧需要研究者们的继续探索，例如深度神经网络模型在动态数据的学习环境中仍然有提升的空间。本文进行了原型学习和深度神经网络的算法研究，但可以发现其性能相比传统训练模式仍然相距较大，仍待进一步的研究。另一方面来看，例如综述中所提到的包括图像、自然语言处理等多种问题中已有不少算法使用深度神经网络与原型学习的结合提升任务性能。因此，研究原型学习算法如何与深度神经网络相结合是未来的研究趋势。
- **解释模型的原型可视化系统。**随着机器学习模型在实际生活生产场景中的应用，对模型如何进行决策进行解释逐渐成为用户使用安全性保障的一个

手段。本文实验中分别使用原型模型在二维数据上的直接可视化以及在高维度嵌入空间的 t-SNE 可视化展示了模型学习的结果，但该结果仅为实验展示性质，没有更为严谨的评价体系保证展示是否准确地反应了模型的决策过程。如何更为科学地展示、解释模型决策的依据，并将该展示过程应用于解决任务的原型学习算法中，则是原型学习算法研究的一大挑战。

- **面向更为复杂学习环境的原型学习算法。**本文的原型学习算法研究关注了标签模糊性、以及数据增量性等两个研究方向的困难，实际应用中数据环境仍包含更多挑战。例如数据不平衡带来对少量类别关注度不够的问题，本文并未针对该问题进行原型学习算法的优化；对于在线数据中的噪声数据对模型可能产生的影响，本文的模糊聚类算法使用了去噪过程进行处理，但高维空间中数据分布本身就十分稀疏，这使得算法在性能与鲁棒性之间需要平衡。除此之外，数据的标签由于扩展为实数向量，相比 one-hot 类标签更容易产生标记时的错误，以及半监督、弱监督情况下如何使用标签信息等等都是原型学习算法在实际应用时需要考虑的重要问题。

参考文献

[1] Bowman C R, Iwashita T, Zeithamova D. Tracking prototype and exemplar representations in the brain across learning[J]. ELife, 2020, 9: e59360.

[2] Kohonen T. The self-organizing map[J]. Proceedings of the IEEE, 1990, 78(9): 1464-1480.

[3] Hotelling H. Analysis of a complex of statistical variables into principal components.[J]. Journal of educational psychology, 1933, 24(6): 417.

[4] Lee D, Seung H S. Algorithms for non-negative matrix factorization[J]. Advances in neural information processing systems, 2000, 13.

[5] Kim H, Joung S, Kim I, et al. Prototype-Guided Saliency Feature Learning for Person Search[C]//IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021. Computer Vision Foundation / IEEE, 2021: 4865-4874.

[6] Huang L, Dai S, He Z. Few-shot object detection with semantic enhancement and semantic prototype contrastive learning[J]. Knowl. Based Syst., 2022, 252: 109411.

[7] Gupta R, Roy A, Christensen C, et al. Class Prototypes based Contrastive Learning for Classifying Multi-Label and Fine-Grained Educational Videos[C]// IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. IEEE, 2023: 19923-19933.

[8] Liu Z, Zhang H, Zhao C. Prototype-oriented contrastive learning for semi-supervised medical image segmentation[J]. Biomed. Signal Process. Control., 2024, 88(Part A): 105571.

[9] Wu H, Zhang B, Chen C, et al. Federated Semi-Supervised Medical Image Segmentation via Prototype-Based Pseudo-Labeling and Contrastive Learning[J]. IEEE Trans. Medical Imaging, 2024, 43(2): 649-661.

[10] 卢涛, 万永静, 杨威. 基于稀疏主成分分析和自适应阈值选择的图像分割算法[J]. 计算机科学, 2016, 43(7), 95: 95.

[11] Santis E D, Rizzi A. Prototype Theory Meets Word Embedding: A Novel Approach for Text Categorization via Granular Computing[J]. Cogn. Comput., 2023, 15(3): 976-997.

[12] 张幸幸, 朱振峰, 赵亚威, 等. 机器学习中原型学习研究进展[J]. 软件学报, 2022, 33(10): 3732.

[13] Biehl M, Hammer B, Villmann T. Prototype-based models in machine learning[J]. Wiley Interdisciplinary Reviews: Cognitive Science, 2016, 7(2): 92-111.

[14] Meng Y, Shang R, Shang F, et al. Semi-Supervised Graph Regularized Deep NMF With Bi-Orthogonal Constraints for Data Representation[J]. IEEE Trans. Neural Networks Learn. Syst., 2020, 31(9): 3245-3258.

[15] Kohonen T. Improved versions of learning vector quantization[C]//1990 ijcnn international joint conference on Neural networks. 1990: 545-550.

[16] Artac M, Jogan M, Leonardis A. Incremental PCA for on-line visual learning and recognition[C]//2002 International Conference on Pattern Recognition: vol. 3. 2002: 781-784.

[17] Zhu T, Xu Y, Shen F, et al. An online incremental orthogonal component analysis method for dimensionality reduction[J]. Neural Networks, 2017, 85: 33-50.

[18] Havens T C, Bezdek J C, Palaniswami M. Incremental kernel fuzzy c-means[C]//Computational Intelligence: Revised and Selected Papers of the International Joint Conference, IJCCI 2010, Valencia, Spain, October 2010. 2012: 3-18.

[19] Furao S, Hasegawa O. An incremental network for on-line unsupervised classification and topology learning[J]. Neural Networks, 2006, 19(1): 90-106.

[20] Furao S, Ogura T, Hasegawa O. An enhanced self-organizing incremental neural network for online unsupervised learning[J]. Neural Networks, 2007, 20(8): 893-903.

[21] Deng J, Dong W, Socher R, et al. ImageNet: A large-scale hierarchical image database[C]//2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA. IEEE Computer Society, 2009: 248-255.

[22] Lange M D, Aljundi R, Masana M, et al. A Continual Learning Survey: Defying Forgetting in Classification Tasks[J]. IEEE Trans. Pattern Anal. Mach. Intell., 2022, 44(7): 3366-3385.

[23] Mai Z, Li R, Kim H, et al. Supervised Contrastive Replay: Revisiting the Nearest Class Mean Classifier in Online Class-Incremental Continual Learning[C]//IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021. Computer Vision Foundation / IEEE, 2021: 3589-3599.

[24] Zhao H, Zhou T, Long G, et al. Does Continual Learning Equally Forget All Parameters?[J]. ArXiv preprint arXiv:2304.04158, 2023.

[25] Geng X. Label Distribution Learning[J]. IEEE Trans. Knowl. Data Eng., 2016, 28(7): 1734-1748.

[26] Pearson K. LIII. On lines and planes of closest fit to systems of points in space[J]. The London, Edinburgh, and Dublin philosophical magazine and journal of science, 1901, 2(11): 559-572.

[27] 申富饶，竺涛，赵健. 快速与增量式数据降维算法研究[M]. 南京: 科学出版社, 2018.

[28] Hui Zou T H, Tibshirani R. Sparse Principal Component Analysis[J]. Journal of Computational and Graphical Statistics, 2006, 15(2): 265-286.

[29] Yang W, Xu H. Streaming Sparse Principal Component Analysis[C]//Bach F R, Blei D M. JMLR Workshop and Conference Proceedings: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015: vol. 37. JMLR.org, 2015: 494-503.

[30] 谭亚芳, 刘娟, 王才华, 等. 一种稀疏可控的主成分分析方法[J]. 计算机科学, 2017, 44(1), 243: 243.

[31] Weng J, Zhang Y, Hwang W. Candid Covariance-Free Incremental Principal Component Analysis[J]. IEEE Trans. Pattern Anal. Mach. Intell., 2003, 25(8): 1034-1040.

[32] Turk M A, Pentland A P. Face recognition using eigenfaces[C]//Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition. 1991: 586-587.

[33] Clausen C, Wechsler H. Color image compression using PCA and backpropagation learning[J]. Pattern recognition, 2000, 33(9): 1555-1560.

[34] Tzeng D Y, Berns R S. A review of principal component analysis and its applications to color technology[J]. Color Research & Application, 2005, 30(2): 84-98.

[35] Yeung K Y, Ruzzo W L. Principal component analysis for clustering gene expression data[J]. Bioinformatics, 2001, 17(9): 763-774.

[36] Jolliffe I T, Cadima J. Principal component analysis: a review and recent developments[J]. Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences, 2016, 374(2065): 20150202.

[37] Abdi H, Williams L J. Principal component analysis[J]. Wiley interdisciplinary reviews: computational statistics, 2010, 2(4): 433-459.

[38] 赵蔷. 主成分分析方法综述[J]. 软件工程, 2016, 19(6): 1-3.

[39] Kim H, Park H. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis[J]. Bioinform., 2007, 23(12): 1495-1502.

[40] Li S Z, Hou X, Zhang H, et al. Learning Spatially Localized, Parts-Based Representation[C]//2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), with CD-ROM, 8-14 December 2001, Kauai, HI, USA. IEEE Computer Society, 2001: 207-212.

[41] Kim Y, Choi S. Weighted nonnegative matrix factorization[C]//Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009, 19-24 April 2009, Taipei, Taiwan. IEEE, 2009: 1541-1544.

[42] Wang Y, Zhang Y. Nonnegative Matrix Factorization: A Comprehensive Review[J]. IEEE Trans. Knowl. Data Eng., 2013, 25(6): 1336-1353.

[43] 李乐, 章毓晋. 非负矩阵分解算法综述[J]. 电子学报, 2008, 36(4): 737-743.

[44] Schölkopf B, Smola A, Müller K R. Nonlinear Component Analysis as a Kernel Eigenvalue Problem[J]. Neural Computation, 1998, 10(5): 1299-1319.

[45] Daoqiang Z, ZhiHua Z, Songcan C. Non-negative Matrix Factorization on Kernels[C]//PRICAI 2006: Trends in Artificial Intelligence. Springer Berlin Heidelberg, 2006: 404-412.

[46] Kambhatla N, Leen T K. Dimension reduction by local principal component analysis[J]. Neural computation, 1997, 9(7): 1493-1516.

[47] Hastie T, Tibshirani R, Friedman J H. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition[M]. Springer, 2009.

[48] Robbins H, Monro S. A stochastic approximation method[J]. The annals of mathematical statistics, 1951: 400-407.

[49] Dittenbach M, Merkl D, Rauber A. The growing hierarchical self-organizing map[C]//Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium: vol. 6. 2000: 15-19.

[50] Rauber A, Merkl D, Dittenbach M. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data[J]. IEEE Transactions on Neural Networks, 2002, 13(6): 1331-1341.

[51] Martinetz T, Berkovich S G, Schulten K. 'Neural-gas' network for vector quantization and its application to time-series prediction[J]. IEEE Trans. Neural Networks, 1993, 4(4): 558-569.

[52] Fritzke B. A Growing Neural Gas Network Learns Topologies[C]//Tesauro G, Touretzky D S, Leen T K. Advances in Neural Information Processing Systems 7, [NIPS Conference, Denver, Colorado, USA, 1994]. MIT Press, 1994: 625-632.

[53] Zhang H, Xiao X, Hasegawa O. A Load-Balancing Self-Organizing Incremental Neural Network[J]. IEEE Trans. Neural Networks Learn. Syst., 2014, 25(6): 1096-1105.

[54] Ouyang Q, Shen F, Zhao J. A Local Distribution Net for Data Clustering[C]// Anthony P, Ishizuka M, Lukose D. Lecture Notes in Computer Science: PRICAI 2012: Trends in Artificial Intelligence - 12th Pacific Rim International Conference on Artificial Intelligence, Kuching, Malaysia, September 3-7, 2012. Proceedings: vol. 7458. Springer, 2012: 411-422.

[55] Xing Y, Cao T, Zhou K, et al. An Incremental Local Distribution Network for Unsupervised Learning[C]//Cao T H, Lim E, Zhou Z, et al. Lecture Notes in Computer Science: Advances in Knowledge Discovery and Data Mining - 19th Pacific-Asia Conference, PAKDD 2015, Ho Chi Minh City, Vietnam, May 19-22, 2015, Proceedings, Part I: vol. 9077. Springer, 2015: 646-658.

[56] Corchado J M, de Paz J F, Rodríguez S, et al. Model of experts for decision support in the diagnosis of leukemia patients[J]. Artif. Intell. Medicine, 2009, 46(3): 179-200.

[57] De Paz J F, Rodríguez S, Bajo J, et al. Case-based reasoning as a decision support system for cancer diagnosis: A case study[J]. Int. J. Hybrid Intell. Syst., 2009, 6(2): 97-110.

[58] Carpine F, Mazzariello C, Sansone C. Online IRC botnet detection using a SOINN classifier[C]//IEEE International Conference on Communications, ICC

2013, Budapest, Hungary, June 9-13, 2013, Workshops Proceedings. IEEE, 2013: 1351-1356.

[59] He X, Ogura T, Satou A, et al. Developmental Word Acquisition and Grammar Learning by Humanoid Robots Through a Self-Organizing Incremental Neural Network[J]. IEEE Trans. Syst. Man Cybern. Part B, 2007, 37(5): 1357-1372.

[60] He X, Kojima R, Hasegawa O. Developmental Word Grounding Through a Growing Neural Network With a Humanoid Robot[J]. IEEE Trans. Syst. Man Cybern. Part B, 2007, 37(2): 451-462.

[61] Kawewong A, Honda Y, Tsuboyama M, et al. Reasoning on the Self-Organizing Incremental Associative Memory for Online Robot Path Planning[J]. IEICE Trans. Inf. Syst., 2010, 93-D(3): 569-582.

[62] 邱天宇, 申富饶, 赵金熙. 自组织增量学习神经网络综述[J]. 软件学报, 2016, 27(9): 2230-2247.

[63] Cover T M, Hart P E. Nearest neighbor pattern classification[J]. IEEE Trans. Inf. Theory, 1967, 13(1): 21-27.

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

[65] Arya S, Mount D M. Approximate nearest neighbor queries in fixed dimensions[C]//SODA: vol. 93. 1993: 271-280.

[66] Bhatia N, et al. Survey of nearest neighbor techniques[J]. ArXiv preprint arXiv:1007.0085, 2010.

[67] Shakhnarovich G, Darrell T, Indyk P. Nearest-neighbor methods in learning and vision[J]. IEEE Trans. Neural Networks, 2008, 19(2): 377.

[68] Chen G H, Shah D, et al. Explaining the success of nearest neighbor methods in prediction[J]. Foundations and Trends® in Machine Learning, 2018, 10(5-6): 337-588.

[69] 李聪, 梁昌勇, 马丽. 基于领域最近邻的协同过滤推荐算法[J]. 计算机研究与发展, 2008, 45(9): 1532-1538.

[70] 钟智, 朱曼龙, 张晨, 等. 最近邻分类方法的研究[J]. 计算机科学与探索, 2011, 5(5): 467-473.

[71] 赵璐璐, 耿国华, 李康, 等. 基于 SURF 和快速近似最近邻搜索的图像匹配算法[J]. 计算机应用研究, 2013, 30(3): 921-923.

[72] Sato A, Yamada K. Generalized Learning Vector Quantization[C]//Touretzky D S, Mozer M, Hasselmo M E. Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, November 27-30, 1995. MIT Press, 1995: 423-429.

[73] Seo S, Bode M, Obermayer K. Soft nearest prototype classification[J]. IEEE Trans. Neural Networks, 2003, 14(2): 390-398.

[74] Seo S, Obermayer K. Soft Learning Vector Quantization[J]. Neural Comput., 2003, 15(7): 1589-1604.

[75] Shamshad F, Khan S, Zamir S W, et al. Transformers in medical imaging: A survey[J]. Medical Image Analysis, 2023: 102802.

[76] Wang H, Li J, Wu H, et al. Pre-trained language models and their applications[J]. Engineering, 2022.

[77] Liu S, Mallol-Ragolta A, Parada-Cabaleiro E, et al. Audio self-supervised learning: A survey[J]. Patterns, 2022, 3(12).

[78] OpenAI. GPT-4 Technical Report[J]. CoRR, 2023, abs/2303.08774.

[79] Zeiler M D, Fergus R. Visualizing and Understanding Convolutional Networks[C]//Fleet D J, Pajdla T, Schiele B, et al. Lecture Notes in Computer Science: Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I: vol. 8689. Springer, 2014: 818-833.

[80] Heaton J. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning: The MIT Press, 2016, 800 pp, ISBN: 0262035618[J]. Genetic programming and evolvable machines, 2018, 19(1-2): 305-307.

[81] Masana M, Liu X, Twardowski B, et al. Class-incremental learning: survey and performance evaluation on image classification[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 45(5): 5513-5533.

[82] Rebuffi S, Kolesnikov A, Sperl G, et al. ICaRL: Incremental Classifier and Representation Learning[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. IEEE Computer Society, 2017: 5533-5542.

[83] Lange M D, Tuytelaars T. Continual Prototype Evolution: Learning Online from Non-Stationary Data Streams[C]//2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021. IEEE, 2021: 8230-8239.

[84] Yu L, Twardowski B, Liu X, et al. Semantic Drift Compensation for Class-Incremental Learning[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020. Computer Vision Foundation / IEEE, 2020: 6980-6989.

[85] Khosla P, Teterwak P, Wang C, et al. Supervised Contrastive Learning[C]// Larochelle H, Ranzato M, Hadsell R, et al. Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual. 2020.

[86] Jaiswal A, Babu A R, Zadeh M Z, et al. A Survey on Contrastive Self-supervised Learning[J]. CoRR, 2020, abs/2011.00362.

[87] Graf F, Hofer C D, Niethammer M, et al. Dissecting Supervised Constrastive Learning[C]//Meila M, Zhang T. Proceedings of Machine Learning Research: Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event: vol. 139. PMLR, 2021: 3821-3830.

[88] Ruspini E H. A new approach to clustering[J]. Information and control, 1969, 15(1): 22-32.

[89] Badarneh O, Ayyoub M A, Alhindawi N, et al. Fine-Grained Emotion Analysis of Arabic Tweets: A Multi-target Multi-label Approach[C]//ICSC. IEEE Computer Society, 2018: 340-345.

[90] Jain H, Prabhu Y, Varma M. Extreme Multi-label Loss Functions for Recommendation, Tagging, Ranking & Other Missing Label Applications[C]//KDD. ACM, 2016: 935-944.

[91] Zhou Y, Xue H, Geng X. Emotion Distribution Recognition from Facial Expressions[C]//ACM Multimedia. ACM, 2015: 1247-1250.

[92] Parisi G I, Kemker R, Part J L, et al. Continual lifelong learning with neural networks: A review[J]. Neural Networks, 2019, 113: 54-71.

[93] Mirzadeh S I, Farajtabar M, Pascanu R, et al. Understanding the Role of Training Regimes in Continual Learning[C]//Advances in Neural Information Processing Systems: vol. 33. 2020: 7308-7320.

[94] Li W, Zhang Y, Sun Y, et al. Approximate Nearest Neighbor Search on High Dimensional Data - Experiments, Analyses, and Improvement[J]. IEEE Trans. Knowl. Data Eng., 2020, 32(8): 1475-1488.

[95] Cai D, He X. Orthogonal locality preserving indexing[C]//Baeza-Yates R A, Ziviani N, Marchionini G, et al. SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005. ACM, 2005: 3-10.

[96] Yang Z, Oja E. Linear and nonlinear projective nonnegative matrix factorization[J]. IEEE Trans. Neural Networks, 2010, 21(5): 734-749.

[97] Lichman M. UCI Machine Learning Repository[EB/OL]. University of California, Irvine, School of Information. 2013. http://archive.ics.uci.edu/ml.

[98] D. Prokhorov. IJCNN 2001 neural network competition[M]. Ford Research Laboratory, 2001.

[99] J. Y. Wang. Application of support vector machines in bioinformatics[M]. Master's thesis. Department of Computer Science, 2002.

[100] M. Duarte, Y. H. Hu. Vehicle classification in distributed sensor networks[J]. Journal of Parallel and Distributed Computing, 2004, 64(7): 826-838.

[101] J. J. Hull. A database for handwritten text recognition research[J]. IEEE Trans. Pattern Anal. Mach. Intell., 1994, 16(5): 550-554.

[102] Krizhevsky A, Hinton G, et al. Learning multiple layers of features from tiny images[J]., 2009.

[103] Nagy G. State of the art in pattern recognition[J]. Proceedings of the IEEE, 1968, 56(5): 836-863.

[104] Zadeh L. Fuzzy sets[J]. Information and Control, 1965, 8(3): 338-353.

[105] Bezdek J C. Pattern Recognition with Fuzzy Objective Function Algorithms[M]. Springer, 1981.

[106] Cebeci Z, Yildiz F. Comparison of k-means and fuzzy c-means algorithms on different cluster structures[J]. Journal of Agricultural Informatics, 2015, 6(3).

[107] Kanzawa Y, Endo Y, Miyamoto S. Fuzzy c-Means Clustering for Data with Tolerance Using Kernel Functions[C]//IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2006, Vancouver, BC, Canada, July 16-21, 2006. IEEE, 2006: 744-750.

[108] Gustafson D E, Kessel W C. Fuzzy clustering with a fuzzy covariance matrix[C]//1978 IEEE conference on decision and control including the 17th symposium on adaptive processes. 1979: 761-766.

[109] Bezdek J C, Coray C, Gunderson R, et al. Detection and characterization of cluster substructure II. Fuzzy c-varieties and convex combinations thereof[J]. SIAM Journal on Applied Mathematics, 1981, 40(2): 358-372.

[110] Lin P L, Huang P W, Kuo C H, et al. A size-insensitive integrity-based fuzzy c-means method for data clustering[J]. Pattern Recognition, 2014, 47(5): 2042-2056.

[111] Pham D T, Dimov S S, Nguyen C. An incremental K-means algorithm[J]. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 2004, 218(7): 783-795.

[112] Hore P, Hall L, Goldgof D, et al. Online fuzzy c means[C]//NAFIPS 2008-2008 Annual Meeting of the North American Fuzzy Information Processing Society. 2008: 1-5.

[113] Labroche N. Online fuzzy medoid based clustering algorithms[J]. Neurocomputing, 2014, 126: 141-150.

[114] Carpenter G A, Grossberg S, Rosen D B. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system[J]. Neural networks, 1991, 4(6): 759-771.

[115] 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.

[116] Baraldi A, Alpaydin E. Constructive feedforward ART clustering networks. I[J]. IEEE transactions on neural networks, 2002, 13(3): 645-661.

[117] Baraldi A, Parmiggiani F. Novel neural network model combining radial basis function, competitive Hebbian learning rule, and fuzzy simplified adaptive resonance theory[C]//Applications of Soft Computing: vol. 3165. 1997: 98-112.

[118] Baraldi A, Blonda P. A survey of fuzzy clustering algorithms for pattern recognition. I[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 1999, 29(6): 778-785.

[119] HUNTSBERGER T L, AJJIMARANGSEE P. Parallel self-organizing feature maps for unsupervised pattern recognition[J]. International Journal Of General System, 1990, 16(4): 357-372.

[120] Son L H, Thong P H. Some novel hybrid forecast methods based on picture fuzzy clustering for weather nowcasting from satellite image sequences[J]. Applied Intelligence, 2017, 46: 1-15.

[121] Liu L, Li C F, Lei Y M, et al. A new fuzzy clustering method with neighborhood distance constraint for volcanic ash cloud[J]. IEEE Access, 2016, 4: 7005-7013.

[122] Dubey Y K, Mushrif M M, et al. FCM clustering algorithms for segmentation of brain MR images[J]. Advances in Fuzzy Systems, 2016, 2016.

[123] Fu L, Medico E. FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data[J]. BMC bioinformatics, 2007, 8(1): 1-15.

[124] Wang Y F, Yu Z G, Anh V. Fuzzy C-means method with empirical mode decomposition for clustering microarray data[J]. International journal of data mining and bioinformatics, 2013, 7(2): 103-117.

[125] Dua D, Graff C. UCI Machine Learning Repository[EB/OL]. University of California, Irvine, School of Information. 2017. http://archive.ics.uci.edu/ml.

[126] Cai D, He X, Han J. Speed up kernel discriminant analysis[J]. VLDB J., 2011, 20(1): 21-33.

[127] Yang Q, Han G, Gao W, et al. A robust learning membership scaling fuzzy C-means algorithm based on new belief peak[J]. IEEE Transactions on Fuzzy Systems, 2023.

[128] 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.

[129] Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: Machine Learning in Python[J]. Journal of Machine Learning Research, 2011, 12: 2825-2830.

[130] Geng X, Wang Q, Xia Y. Facial Age Estimation by Adaptive Label Distribution Learning[C]//ICPR. IEEE Computer Society, 2014: 4465-4470.

[131] Wang J, Geng X. Label Distribution Learning Machine[C]//Meila M, Zhang T. Proceedings of Machine Learning Research: Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event: vol. 139. PMLR, 2021: 10749-10759.

[132] Geng X, Smith-Miles K, Zhou Z. Facial Age Estimation by Learning from Label Distributions[C]//AAAI. AAAI Press, 2010.

[133] Geng X, Yin C, Zhou Z. Facial Age Estimation by Learning from Label Distributions[J]. IEEE Trans. Pattern Anal. Mach. Intell., 2013, 35(10): 2401-2412.

[134] Jia X, Li W, Liu J, et al. Label Distribution Learning by Exploiting Label Correlations[C]//AAAI, New Orleans, Louisiana, USA, February 2-7, 2018. 2018: 3310-3317.

[135] Zheng X, Jia X, Li W. Label Distribution Learning by Exploiting Sample Correlations Locally[C]//AAAI, New Orleans, Louisiana, USA, February 2-7, 2018. 2018: 4556-4563.

[136] Jia X, Li Z, Zheng X, et al. Label Distribution Learning with Label Correlations on Local Samples[J]. IEEE Trans. Knowl. Data Eng., 2021, 33(4): 1619-1631.

[137] Zhao P, Zhou Z. Label Distribution Learning by Optimal Transport[C]//AAAI, New Orleans, Louisiana, USA, February 2-7, 2018. 2018: 4506-4513.

[138] Zychowski A, Mandziuk J. Duo-LDL method for Label Distribution Learning based on pairwise class dependencies[J]. Appl. Soft Comput., 2021, 110: 107585.

[139] Gao B, Zhou H, Wu J, et al. Age Estimation Using Expectation of Label Distribution Learning[C]//Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden. 2018: 712-718.

[140] Zhang H, Zhang Y, Geng X. Practical age estimation using deep label distribution learning[J]. Frontiers Comput. Sci., 2021, 15(3): 153318.

[141] Wang K, Geng X. Binary Coding based Label Distribution Learning[C]// Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden. 2018: 2783-2789.

[142] Zhang Z, Lai C, Liu H, et al. Infrared facial expression recognition via Gaussian-based label distribution learning in the dark illumination environment for human emotion detection[J]. Neurocomputing, 2020, 409: 341-350.

[143] Liu Z, Chen Z, Bai J, et al. Facial Pose Estimation by Deep Learning from Label Distributions[C]//2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019. IEEE, 2019: 1232-1240.

[144] Liu T, Wang J, Yang B, et al. NGDNet: Nonuniform Gaussian-label distribution learning for infrared head pose estimation and on-task behavior understanding in the classroom[J]. Neurocomputing, 2021, 436: 210-220.

[145] Ling M, Geng X. Soft video parsing by label distribution learning[J]. Frontiers Comput. Sci., 2019, 13(2): 302-317.

[146] Lawson C L, Hanson R J. Solving Least Squares Problems[M]. Society for Industrial, 1995.

[147] Kim J, Park H. Toward Faster Nonnegative Matrix Factorization: A New Algorithm and Comparisons[C]//ICDM. IEEE Computer Society, 2008: 353-362.

[148] Yuan Y x. A modified BFGS algorithm for unconstrained optimization[J]. IMA Journal of Numerical Analysis, 1991, 11(3): 325-332.

[149] Tütüncü R H, Toh K, Todd M J. Solving semidefinite-quadratic-linear programs using SDPT3[J]. Math. Program., 2003, 95(2): 189-217.

[150] CVX Research I. CVX: Matlab Software for Disciplined Convex Programming, version 2.0[Z]. http://cvxr.com/cvx. 2012.

[151] Grant M, Boyd S. Graph implementations for nonsmooth convex programs[G]//Blondel V, Boyd S, Kimura H. Lecture Notes in Control and Information Sciences: Recent Advances in Learning and Control. Springer-Verlag Limited, 2008: 95-110.

[152] M.B.Eisen, P.T.Spellman, P.O.Brown, et al. Cluster analysis and display of genome-wide expression patterns[C]//Proc. Nat. Acad. Sci. USA. 1998: 14863-14868.

[153] Lyons M J, Akamatsu S, Kamachi M, et al. Coding Facial Expressions with Gabor Wavelets[C]//3rd International Conference on Face & Gesture Recognition (FG '98), April 14-16, 1998, Nara, Japan. 1998: 200-205.

[154] Ahonen T, Hadid A, Pietikäinen M. Face Description with Local Binary Patterns: Application to Face Recognition[J]. IEEE Trans. Pattern Anal. Mach. Intell., 2006, 28(12): 2037-2041.

[155] Masana M, Liu X, Twardowski B, et al. Class-incremental learning: survey and performance evaluation[J]. CoRR, 2020, abs/2010.15277.

[156] Zhao H, Zhou T, Long G, et al. Does Continual Learning Equally Forget All Parameters?[C]//Krause A, Brunskill E, Cho K, et al. Proceedings of Machine Learning Research: International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA: vol. 202. PMLR, 2023: 42280-42303.

[157] Hu X, Tang K, Miao C, et al. Distilling Causal Effect of Data in Class-Incremental Learning[C]//IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021. Computer Vision Foundation / IEEE, 2021: 3957-3966.

[158] Lopez-Paz D, Ranzato M. Gradient Episodic Memory for Continual Learning[C]//Guyon I, von Luxburg U, Bengio S, et al. Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. 2017: 6467-6476.

[159] Vitter J S. Random Sampling with a Reservoir[J]. ACM Trans. Math. Softw., 1985, 11(1): 37-57.

[160] Chaudhry A, Rohrbach M, Elhoseiny M, et al. Continual Learning with Tiny Episodic Memories[J]. CoRR, 2019, abs/1902.10486.

[161] Aljundi R, Belilovsky E, Tuytelaars T, et al. Online Continual Learning with Maximal Interfered Retrieval[C]//Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems

2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. 2019: 11849-11860.

[162] Aljundi R, Lin M, Goujaud B, et al. Gradient based sample selection for online continual learning[C]//Wallach H M, Larochelle H, Beygelzimer A, et al. Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. 2019: 11816-11825.

[163] Lee S, Ha J, Zhang D, et al. A Neural Dirichlet Process Mixture Model for Task-Free Continual Learning[C]//8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. 2020.

[164] Cha H, Lee J, Shin J. Co^{2}L: Contrastive Continual Learning[C]//2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021. IEEE, 2021: 9496-9505.

[165] Guo Y, Liu B, Zhao D. Online Continual Learning through Mutual Information Maximization[C]//Proceedings of Machine Learning Research: International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA: vol. 162. PMLR, 2022: 8109-8126.

[166] Khosla P, Teterwak P, Wang C, et al. Supervised contrastive learning[J]. Advances in Neural Information Processing Systems, 2020, 33: 18661-18673.

[167] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proc. IEEE, 1998, 86(11): 2278-2324.

[168] Krizhevsky A, Hinton G, et al. Learning multiple layers of features from tiny images[J]., 2009.

[169] Vinyals O, Blundell C, Lillicrap T, et al. Matching Networks for One Shot Learning[C]//Lee D D, Sugiyama M, von Luxburg U, et al. Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain. 2016: 3630-3638.

[170] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[C]//2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. IEEE Computer Society, 2016: 770-778.

[171] Zhou D, Wang F, Ye H, et al. Forward Compatible Few-Shot Class-Incremental Learning[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022. IEEE, 2022: 9036-9046.

致　　谢

一转眼间在仙林校区已经待了十年有余，期间设想过离开校园的生活，但在真正告别时却只有不舍。回首博士期间的这七年，有过对自己能力的质疑、失败的痛苦，也有过发现新想法时的欣喜、发表论文时的成就感，但这一切都随着时间的流逝消磨了情感的波澜，最终沉淀为记忆中闪闪发亮的一部分。匆匆时光带走了我的稚嫩，带走了我的发量，也见证了我的成长，使我成为了今天的我。幸运的是，在这条孤独的求学道路上，我却并不孤单。因为有带领我一路走来，陪伴我一起走下去的人们，我才能够成功走到今天。

首先需要感谢的是申老师对我学业上的帮助与教诲。犹记得本科期间因为大创项目联系到了申老师，使得我满怀期待地向您申请了直博资格。正是您温和且充满耐心的教导，才使得我有勇气一路解决困难，成功获得完成学业的资格。回首这些年的经历，我并不是一个让人省心的学生，但是最终您还是带领我走到了今天，为我指引了未来的道路。感谢您的指导，能够成为您的学生是一件十分幸运的事情。

其次需要感谢同门们与朋友们对我的支持与鼓励。感谢竺涛师兄、徐百乐师兄对我科研问题上的帮助；感谢我的室友王涵，那间阴冷的一楼宿舍见证了我们从陌生到熟络再到长时间的陪伴，在我不成熟的岁月里有你的陪伴与支持令我感到十分的幸福；感谢我同门的兄弟姐妹们，在我需要帮助之时你们热情地伸出援手，我们相互帮助从而走到了今天。此外还要感谢我的朋友们，一路走来我们相互倾诉工作生活的苦恼，同时积极为对方排忧解难。感谢一路走来支持我的前辈与朋友们，你们为我的生活添加了色彩，让我知道前进的道路上我并不孤单。

最后需要感谢我的家人们，你们是我最温暖的港湾，最坚强的后盾。感谢我的父母，为我提供生活上的支持，精神上的支撑以及工作上的激励，却不求回报；感谢我的长辈们，每次见面都能收获暖心的关怀；感谢我的姐妹们，在过去、现在以及未来的人生中我们将相互陪伴。感谢支持我到今天的家人们，正是有你们才有今天的我。

攻读博士学位期间的学术成果和获奖情况

已发表学术论文

Zhang T, Xu B, Shen F. Fuzzy Self-Organizing Incremental Neural Network for Fuzzy Clustering[C]. ICONIP. Springer, Cham, 2017: 24-32. *(CCF-C 类会议)*

Zhang T, Shen F, Zhu T, Zhao J. An Evolutionary Orthogonal Component Analysis Method for Incremental Dimensionality Reduction[J]. IEEE Transactions on Neural Networks and Learning Systems, 2020. *(CCF-B 类期刊，中科院计算机学科分区 1 区，top)*

Zhang T, Mao Y, Shen F, Zhao J. Label distribution learning through exploring nonnegative components[J]. Neurocomputing, 2022. *(CCF-C 类期刊，中科院计算机学科分区 2 区，top)*

Yang S, Li J, **Zhang T**, Zhao J, Furao S. AdvMask: A sparse adversarial attack-based data augmentation method for image classification[J]. Pattern Recognition, 2023. *(CCF-B 类期刊)*

评审中的学术论文

Zhang T, Yang S, Shen F, Zhao J. Supervised Contrastive Learning with Prototype Distillation for Data Incremental Learning.

申请专利

申富饶; 张天玥; 时晓峰; 杨锁荣; 赵健. “一种基于自组织增量图的半监督多媒体数据流分类方法”. CN202211315078.3

获奖情况

新生校长特别奖学金，计算机科学与技术系，南京大学，2017

科研项目参与情况

2023.1-至今：国家自然基金面上项目，面向增量式无监督学习的新型神经网络研究，62276127

2019.2-2020.12：重点实验室项目, 基于增量神经网络的感知融合海洋目标识

别，6142106180301

2019.1-2022.12：国家自然基金面上项目，基于深度感知增量式联想记忆神经网络的信息融合系统研究，61876076