arXiv:2304.13717v1 [cs.LG] 26 Apr 2023

使用自编码器的关联规则挖掘

Théophile Berteloot、Richard Khoury、
Audrey Durand拉瓦尔大学 魁北克市

摘要

关联规则挖掘是数据挖掘中研究最广泛的领域之一，应用范围从购物篮问题到可解释的分类系统。经典的关联规则挖掘算法存在若干局限，尤其是执行时间过长、生成的规则数量过多。在过去十年中，神经网络解决方案已被用于解决各种优化问题，例如分类、回归或聚类。然而，目前仍没有利用神经网络高效挖掘关联规则的方法。本文提出一种名为 ARM-AE 的自编码器解决方案，用于挖掘关联规则。我们在三个分类数据集上将该算法与 FP-Growth 和 NSGAII 进行比较，结果表明，该算法能够发现支持度和置信度较高的规则集；在保持所生成规则集质量的同时，其执行时间也优于经典方法。

关键词 关联规则挖掘 · 深度学习 · 自动编码器

1 引言

关联规则挖掘（ARM）最初由 Agrawal 提出，用于解决购物篮问题，此后已在数据库中的知识发现（KDD）问题中得到广泛应用，涵盖金融分析到医学诊断等领域。关联规则（AR）是一种形式为 $A \Rightarrow C$ 的蕴含关系，可以理解为“如果前件 A 为真，则后件 C 必须为真”，其中 A 和 C 是数据库中的不同项的集合（项集）。关联规则由其前件、后件和两个度量指标定义。第一个指标是支持度，即数据集中同时出现前件和后件的行数所占的比例。第二个指标是置信度，即在观察到前件的条件下观测到后件的条件概率。

最广泛使用的挖掘策略 Apriori 和其他穷举策略通常首先挖掘频繁项集，然后组合这些项集以生成关联规则。然而，所有这些算法都面临相同的问题：它们生成的规则数量会随着数据库中项的数量呈指数增长，因此人类不可能逐一筛选返回的规则来挑选出最佳规则。面对大规模数据集时，它们的执行时间也会成为一个问题。最后，这些算法需要支持度和置信度阈值，才能高效地搜索解空间，而这些阈值必须经过谨慎选择：较低的值可能导致执行时间过长以及规则数量过多，而较高的值则会使算法遗漏有趣的规则。

研究人员曾尝试使用顶尖k 算法 [9], 元启发式算法 [10] （例如 NSGAII） [11],并行化 [12], 和 GPU [13]来解决这些问题。与此同时，深度学习已被应用于从图像分析 [14] 到自然语言处理 [15] 等研究领域，并展现出发现输入数据 [16]之间潜在关系的强大能力。自动编码器（AE）是一种神经网络，通常用于发现聚类 [17] 或学习表示[18]。AE 以数据集作为输入，并以相同的数据集作为目标，以监督方式进行训练。
因此，AE 似乎适用于关联规则挖掘（ARM），因为 AE 在尝试重建原始数据集时，会学习数据集各列之间的联系；随后，我们便可以利用这些联系来挖掘关联规则（AR）。
因此，本文的贡献有两个方面：

• 我们提出了一种使用自编码器挖掘关联规则的新方法，并将其称为 ARM-AE。

• 我们根据执行时间、生成的规则数量、所生成规则的平均支持度和置信度，以及覆盖度（即我们的方法发现的 FP-Growth 规则占 FP-Growth 规则总数的百分比），将我们的方法与 FP-Growth（一种穷举式的最先进算法）和 NSGAII（一种常用的元启发式算法）进行了比较。我们使用了三个取自广受欢迎的 UCI 仓库的分类数据集 [19], ，其规模从小型数据集到超大规模数据集不等。在所有情况下，ARM-AE 都能够高效地挖掘出数量有限的高质量规则，而无需指定任何强制性阈值。

本文其余部分的组织结构如下：第 2 节介绍利用神经网络解决关联规则挖掘问题的相关文献；第 3 节介绍我们的 ARM-AE 模型。我们在第 4 节使用该模型并以 FP-Growth 作为基准开展实验，并在第 5 节讨论实验结果。最后，我们在第 6 节总结研究结果并给出结论性意见。

2 相关工作

此前已有多项工作在关联规则挖掘的背景下采用了深度学习。例如，研究人员曾结合关联规则和深度神经网络，根据推文预测车祸 [20]。更具体地说，研究人员使用 Apriori 对推文中的词元进行关联规则挖掘，并将其作为深度信念网络的输入，以判断某条推文是否与车祸有关。关联规则和卷积神经网络（CNNs）也曾结合使用，用于对三座西班牙纪念碑的评论进行分类，并提取访客不喜欢的方面，从而帮助纪念碑改善访客体验 [21]。首先，研究人员使用 CNNs 提取人们对纪念碑具有强烈感受的方面。随后，将这些方面聚类为含义相同但由不同词语表示的组。最后，研究人员通过关联规则挖掘总结与各方面相对应的情感。最后一个例子是，Apriori 曾与深度学习结合，用于预测应向个体患者使用哪些抗癌药物 [22]。在这项工作中，研究人员从药物基因组学数据集中挖掘关联规则，并使用提取出的特征训练深度神经网络。

虽然上述引用的先前研究示例分别进行了关联规则挖掘和深度学习，但并未使用深度学习进行关联规则挖掘。很少有研究应对这一挑战。其中一项研究是 [23], ，该研究使用神经网络和遗传算法挖掘关联规则。他们使用自组织映射（SOM）从数据中挖掘聚类模式，然后根据这些模式使用遗传算法生成关联规则。在 [24], 中，研究人员使用卷积神经网络在具有复杂属性的医学数据集上挖掘关联规则，然后通过概率估计方法验证挖掘出的关联规则。

上述系统都没有直接使用深度学习挖掘关联规则：它们要么使用深度学习挖掘频繁项集，要么将其与关联规则挖掘算法结合使用。通过测试项集组合来寻找优质关联规则是一项非常耗时的任务。所提出的 ARM-AE 方法直接挖掘关联规则，因此无需花时间尝试不同的组合。

3 ARM-AE：使用自编码器进行关联规则挖掘

在 ARM-AE 中，项是数据集中的一列的名称，而项集是不含重复项的一组项。为了比较不同的规则，我们使用支持度和置信度。支持度定义为数据集 D 中同时出现 A 和 C 的行所占的比例： $Supp(A,\ C) = |A \cap C|/|D|$，其中 A 为前件， C为后件，二者是没有共同项的两个项集。置信度是给定观察到前件的情况下观察到后件的条件概率： $Conf(A,\ C) = |A \cap C|/|A|$。

我们提出利用自编码器（AE）的表示能力直接提取关联规则。自编码器由两部分组成：第一部分是编码器，由若干尺寸逐渐减小的层组成；第二部分是解码器，由若干尺寸逐渐增大的层组成。编码器的输出是解码器的输入。自编码器的训练目标是让编码器生成输入的压缩表示，并让解码器生成原始输入的转换版本。

ARM-AE 的训练与传统自编码器的训练并无不同。自编码器的输入和预期输出均为数据集中的每一行。因此，自编码器将学习根据相似项集对输入项集进行编码，而解码后的输出将是相似项集，其中各项的取值介于 0 和 1 之间，表示它们属于该项集的可能性。鉴于我们的目标是学习整个数据集的表示，因此无需评估集或测试集，我们使用整个数据集训练模型。在训练过程中，自编码器将学习重建输入的二进制行；因此，训练完成后，如果将一个项集作为输入，自编码器会重建该项集，但不会完全准确。与输入项集在数据集中频繁共同出现的项，其输出值会高于从未与输入项集共同出现的项。

我们的关联规则挖掘方法 ARM-AE 的核心见算法1。我们的算法尝试为每个后件挖掘一组长度不超过 M 的 $N \times M$ 规则或 N 规则，并找出前件项集中最有趣的子集。在 ARM-AE 中，数据集中的每一项都会成为一个后件（第 5 行），不过只需简单修改该行，就能让算法专注于感兴趣的特定项子集。后件仅由一项组成，这是关联规则挖掘中最常见的用例。初始时，后件是自编码器唯一的输入（第 8–11 行）。自编码器

算法1关联规则挖掘自动编码器。

```
1: 函数 ARM-AE(N, M, L)    ▷ N 每个后件的最大长度规则数量， M 每条规则的前件最大数量
   前件 L 相似度阈值
2:     后件 ← 项
3:     AE ←训练好的自动编码器
4:     规则 ← ∅
5:     对于 ∀后件 ∈ 后件执行
6:         后件规则 ← ∅
7:         对于从 I = 1到 N 执行
8:             前件 ← ∅
9:             对于从 J = 1到 M 执行
10:                输入数组 ← 前件 ∪后件
11:                AE输出 ←自编码器.前向传播(输入数组)
12:                AE输出 ←排序(AE输出)
13:                对于 前件 ∈ AE输出执行
14:                    if 前件 ≠ 后件 and 前件 ∉ 前件 and
   计算相似度(后件规则,前件 ∪ 前件) ≤ L 然后
15:                        前件 ← 前件 ∪ 前件
16:                        后件规则 ← 后件规则 ∪ 前件
17:                        退出第 13 行的循环（转到第 9 行）
18:            规则 ← 规则 ∪后件规则
19:    返回 规则
```

对数据集中的每个项输出一个介于 0 和 1 之间的分数；分数越高，该项与自编码器输入一同频繁出现的概率就越高。该输出按降序排序，取其中值最高且不是后件、尚未属于前件，或不会使前件与先前发现的前件的相似度超过阈值 L（如第 3.2 段所述）的项，将其添加到规则的前件中，并将这条新规则添加到该后件对应的关联规则集合中（第 12-16 行）。将前件附加到后件后，作为自编码器的输入（第 10 行），然后重复循环，每次向前件添加一个项，直到其达到最大长度 M 项（第 9 行）。随后，从头开始为该后件生成另一条规则，直到找到 N 条规则（第 7 行），然后转向下一个后件（第 5 行）。ARM-AE 的完整代码可在 GitHub 上获取。[1]

3.1 自编码器训练与超参数

如前文所示，我们的算法有三个主要超参数，即为每个后件生成的最大长度规则数 N、前件中的最大项数 M，以及最大相似度阈值 L。此外，自编码器本身也有相应的超参数，例如层数、每层的神经元数、训练轮数、学习率、激活函数、损失函数和优化器。这些超参数适用于所有自编码器架构。

3.2 相似度

一个重要步骤是在算法1第14行计算新前件与针对某个后件已找到的前件之间的相似度。如果没有这一步，从同一个后件出发，自编码器将始终发现相同的前件。对新前件与已发现前件之间的相似度设置最大值，会迫使自编码器探索解空间，并为每个后件生成各种不同的关联规则。

在 ARM-AE 的实现中，我们采用的相似度量是：对于具有相同后件的规则，候选规则前件中的项与现有规则前件中的项之间的最大重叠度，并以现有规则长度的百分比表示。我们只考虑与候选规则长度相同或更长的规则的相似度，因为否则，较短且一般的现有规则会阻碍发现更长、更具体的候选规则。下面介绍

[1] https://github.com/TheophileBERTELOOT/ARM-A E

表1：为实验选取的 UCI 数据集。

数据集	Rows	列数	支持度	置信度
国际象棋	3,196	75	0.005	0.01
Nursery	12,960	32	0.01	0.01
植物	34,781	70	0.005	0.015

算法 2 中的相似度函数。不过需要指出的是，ARM-AE 并非必须依赖此相似度函数才能运行；也可以改用针对其他特定问题的相似度函数。

算法 2计算前件集合与候选前件之间的相似度

```
1：函数计算相似度(PreviousAntecedents,Candidate)
2:      ▷ PreviousAntecedents 针对特定后件找到的前件列表，Candidate 所对应的前件正在计算相似度
3:      最大相似度 ← 0
4:      对于 ∀前件 ∈ 前件执行
5:          如果 |前件| < |候选项| 那么
6:              跳过前件
7:          相似度 ← 0
8:          对于 ∀Item ∈ 前件执行
9:              如果项 ∈ 候选项 那么 相似度 ← 相似度 + 1
10:         相似度 ← 相似度 / |前件|
11:         最大相似度 ← 最大值(最大相似度,相似度)
12:     返回 最大相似度
```

4 实验

4.1 数据集

我们使用取自知名 UCI 仓库 [19] 的三个分类数据集对 ARM-AE 进行了测试。选择这些数据集是因为它们的行数和项数范围较广。每个数据集都经过预处理，将每个分类属性替换为一组二元属性，每个类别值对应一列。表1列出了这些数据集，以及它们在二值化后的行数和属性数量。

4.2 超参数

对于我们的 AE 架构，编码器和解码器均由三个全连接层组成。每层的神经元数量等于数据集中的不同项数。该架构在我们的实验中取得了最佳结果。神经元使用公式1所示的 tanh 函数作为激活函数，因为 ARM-AE 旨在近似规则的支持度，而支持度定义在 0 和 1 之间，因此我们选择了一个定义域在 0 和 1 之间的常用激活函数。学习率为 10^{-3}。公式2所示的 MSELoss 是我们的损失函数（其中 x 是输入， y 是目标），我们使用 Adam 优化器。

$$Tanh(x) = \frac{\exp x - \exp -x}{\exp x + \exp -x} \tag{1}$$

$$MSELoss(x, y) = (x - y)^2 \tag{2}$$

我们将算法设置为寻找前件中最多包含两个项的规则（$M = 2$），并为每个后件寻找两条规则（$N = 2$）。相似度阈值为 $L = 0.5$，因此两组规则的前件之间最多会有一个重叠项。为了设置训练轮数，我们使用两个训练周期之间的损失差值；当该差值低于某个阈值时，就将前一个训练周期作为最终模型。我们测试了该阈值的多个取值。图1展示了在三个数据集上，阈值取不同值时 ARM-AE 所发现规则的平均支持度；对于每个阈值，我们进行了10次实验，并且每次训练100个训练周期，图中展示的是平均结果。阈值为1意味着没有进行训练

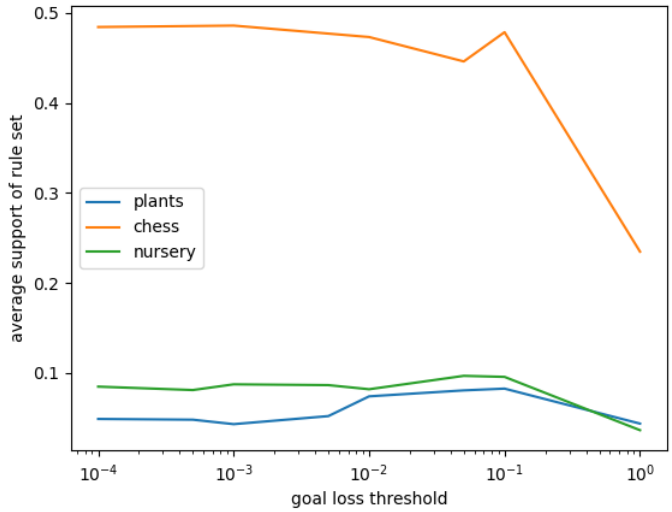


图1：在目标损失采用不同阈值时，三个数据集的 ARM-AE 规则集的平均支持度。

，结果是网络的随机初始化。图中显示，最佳阈值为0.1，实际对应1到5个训练周期。超过该阈值后，性能显著下降。因此，在其余实验中，我们选择使用0.1作为阈值。

4.3 基准算法

我们使用 FPGrowth [5] 作为两个基准算法之一，因为它是目前最快且最流行的详尽型关联规则挖掘算法之一。我们使用了 Mlxtend 库 [25]中的实现。我们为每个数据集选择支持度和置信度阈值，使得挖掘出的规则数量与我们的算法生成的规则数量几乎相同；这些规则的前件包含一个或两个项，后件包含一个项。相关阈值列于表1中。我们的第二个基准算法是 NSGAII [26] 算法，之所以选择它，是因为遗传算法和支配概念是关联规则挖掘中的主流趋势，而 NSGAII 是元启发式算法中常用的基线算法 [10] 。我们使用 DEAP 库[27] 实现了 NSGAII。对于基准算法 NSGAII，我们采用包含 100 个个体的种群、0.9 的交叉率和 0.01 的变异概率。我们将规则中的前件数量上限设为 2，后件数量上限设为 1。对于植物和国际象棋数据集，我们保留所有代中找到的 300 条最佳规则；对于托儿所数据集，则保留 150 条最佳规则，以便结果能够与 ARM-AE 和 FP-Growth 的输出进行比较。为设置代数，我们检查截至目前找到的最佳规则的平均支持度和平均置信度是否超过某个阈值而提高。如果二者均下降，或增幅小于该阈值，则 NSGAII 算法停止。我们使用的阈值为 0.01。

4.4 指标

我们计算了多个指标，以比较 ARM-AE 与基准算法。第一个指标是 ARM-AE 找到的支持度大于 0 的规则数量。支持度大于 0 的规则，表示该规则确实存在于数据集之中。但由于 ARM-AE 在搜索过程中不会计算规则的支持度，不同于 FP-Growth 及其他穷举算法，因此无法对此提供任何保证，这也正是该指标重要的原因。第二个指标是 FP-Growth 找到的规则中，同时也被我们的算法和 NSGAII 找到的规则所占的百分比。由于 FP-Growth 会对所有最佳规则执行穷举搜索，因此该指标可以反映 ARM-AE 和 NSGAII 的规则集与完整规则集的接近程度。不过需要注意的是，我们设置 ARM-AE 的超参数，使其针对每个后件生成四条关联规则，且前件中仅包含一个或两个项。为了进行公平比较，我们因此设置 FP-Growth 的阈值，以提取此前所述长度下每个后件的最佳四条关联规则。接下来，我们比较三个算法的执行时间。最后，我们比较三个算法生成规则的平均支持度和置信度。

表2：支持度大于 0 的规则百分比

数据集	规则
国际象棋	0.99
Nursery	0.98
植物	1.00

表3：各算法和数据集的覆盖度、支持度和置信度结果。

数据集	ARM-AE			NSGAII			FP-Growth	
	覆盖度	支持度	置信度	覆盖度	支持度	置信度	支持度	置信度
国际象棋	0.20	0.48	0.50	0.02	0.33	0.71	0.51	0.54
Nursery	0.40	0.1	0.33	0.08	0.06	0.38	0.12	0.37
植物	0.44	0.08	0.75	0.02	0.05	0.78	0.09	0.7

5 结果与分析

我们使用每个数据集分别独立训练和测试自编码器与 NSGAII 10 次，以考虑网络随机初始化的差异以及元启发式算法的随机性。我们报告 10 次运行的平均结果。由于 FPGrowth 算法不涉及随机初始化，因此仅运行一次。

首先要考虑的指标是 ARM-AE 生成的规则中支持度大于 0 的规则所占的百分比。表2所示结果表明，生成的规则中最多有 2% 实际上并不存在于数据集中。由于 ARM-AE 需要为每个后件生成一定数量的规则，因此当不存在具有高支持度的后件规则时，规则集中会包含一些支持度极低或零支持度规则。如果不计算支持度，ARM-AE 就无法检测或过滤掉这些情况。这是我们的方法存在的一项局限性。不过，由于此类规则占提取规则集的比例不到 2%，因此所引入的噪声量并不是一个严重问题。
接下来的指标将比较 ARM-AE、NSGAII 和 FP-Growth 挖掘出的关联规则集。首先，我们考察 ARM-AE 和 NSGAII 挖掘出的规则占 FP-Growth 规则集的比例。FP-Growth 会穷举挖掘支持度和置信度均高于设定阈值的所有规则，这些规则包含一个后件以及一个或两个前件。相比之下，ARM-AE 挖掘出的规则占该规则集的 20% 至 44%，如表3所示。至于 NSGAII，它挖掘出的规则最多占 FP-Growth 所挖掘规则集的 8%。这可以归因于遗传算法高度随机的特性，也因为 NSGAII 专注于非支配规则，而 FP-Growth 专注于高支持度和高置信度规则。在非支配挖掘中，如果一条规则的支持度很低但置信度很高，且不存在支持度更高、置信度至少与之相同的其他规则，那么这条规则可能不会被保留。表3还展示了这三种算法挖掘出的规则的平均支持度和置信度。结果表明，ARM-AE 找到的规则，其支持度和置信度几乎与 FP-Growth 挖掘出的规则一样高；事实上，在植物数据集中，ARM-AE 的置信度更高。尽管如此，ARM-AE 的结果中仍包含了多达 2% 的零支持度规则，正如前文所述。这表明，虽然 ARM-AE 挖掘出的规则不到 FP-Growth 所发现规则的一半，但它挖掘出的规则子集质量很高，且这些规则都存在于数据库中。NSGAII 挖掘出的规则集置信度很高，但支持度低于另外两种算法。与 NSGAII 相比，ARM-AE 挖掘出的规则集与 FP-Growth 找到的规则集更为相似。

最后，我们在表4中从执行时间和生成规则总数两个方面比较了三种算法的性能。我们首先考察整个算法的执行时间，其中对于ARM-AE，还包括训练自编码器的时间。在每种关联规则挖掘算法中，最耗时的步骤都是计算规则的支持度和置信度。FP-Growth使用FP树来加速频繁项集的挖掘，但随后必须计算所有项集对的支持度和置信度。NSGAII在每一代都会为种群中的很大一部分个体计算支持度和置信度，这解释了其为何比FP-Growth更慢。相比之下，ARM-AE只对挖掘出的规则子集计算支持度和置信度。对于三个数据集，就执行时间而言，ARM-AE的效率都远高于FP-Growth和NSGAII。就规则数量而言，ARM-AE和NSGAII具有生成较小规则集的优势，其规模由算法的超参数控制。相比之下，FP-Growth的穷举搜索会生成数万至数十万条规则。如前所述，这是此类算法的主要缺点之一 [8]：人类无法阅读和理解如此庞大的规则集，因此在规则适合人类使用之前，还必须在后处理中应用额外的过滤或排序算法。ARM-AE和NSGAII不存在这一问题。

表4：算法的性能

数据集	ARM-AE 执行时间（秒）	规则	NSGAII 执行时间（秒）	规则	FP-Growth 执行时间（秒）	规则
国际象棋	1.2	296	95.0	241	35.2	278,823
Nursery	0.8	124	99.0	150	1.4	18,140
植物	1.8	280	979.0	261	114	269,586

6 结论

在本文中，我们介绍了一种用于挖掘关联规则的新型深度学习架构，即关联规则挖掘自编码器（ARM-AE）算法。我们在三个分类数据集上，将这一新算法与广受欢迎的先进FP-Growth算法以及先进遗传算法NSGAII进行了比较。结果表明，ARM-AE生成的规则集具有与FP-Growth相近的平均支持度和置信度，且支持度高于NSGAII，同时执行时间显著更短，生成的规则数量也远少于FP-Growth，因此其结果对人类而言更加易于使用。此外，ARM-AE还具有用于控制每个后件生成的规则数量、规则前件中的最大项数以及规则之间相似度的超参数，这进一步提升了算法的用户友好性。ARM-AE的主要缺点源于它不会计算其规则的支持度和置信度，这意味着它无法过滤掉零支持度规则，也无法为挖掘出的规则提供重要性排序；但这些问题将在未来的研究中得到解决。

致谢

本研究得以完成 得益于 INSPQ 的支持，以及加拿大研究资助机构 CIHR 的慷慨支持。

参考文献

[1] Rakesh Agrawal、Ramakrishnan Srikant 等。用于挖掘关联规则的快速算法。载于第*20*届超大型数据库国际会议论文集，*VLDB*，第1215卷，第487–499页。Citeseer，1994年。[2] Binoy B Nair、VP Mohandas、Nikhil Nayanar、ESR Teja、S Vigneshwari 和 KVNS Teja。基于时序关联规则挖掘的股票交易推荐系统。*SAGE Open*，5(2):2158244015579941，2015年。[3] Azhar Hussein Alkeshuosh、Mariam Zomorodi Moghadam、Inas Al Mansoori 和 Moloud Abdar。使用粒子群优化算法生成用于心脏病诊断的最优规则。载于*2017*年国际计算与应用会议（*ICCA*），第306–311页。IEEE，2017年。[4] Liqiang Geng 和 Howard J Hamilton。选择合适的视角：在数据挖掘中发现有趣之处。载于数据挖掘中的质量度量，第3–24页。Springer，2007年。[5] Christian Borgelt。化繁为简：通过递归消除寻找频繁项集。载于第一届开源数据挖掘国际研讨会论文集：频繁模式挖掘实现，第66–70页，2005年。[6] Borgelt Christian。FP-Growth算法的一种实现。载于第一届开源数据挖掘国际研讨会论文集：频繁模式挖掘实现，第1–5页，2005年。[7] Jian Pei、Jiawei Han、Hongjun Lu、Shojiro Nishio、Shiwei Tang 和 Dongqing Yang。H-mine：大型数据库中快速且节省空间的频繁模式挖掘。*IIE Transactions*，39(6):593–605，2007年。[8] Ian Fund。医疗数据上的关联规则与深度神经网络比较。博士论文，休斯敦大学，2019年。[9] Philippe Fournier-Viger、Cheng-Wei Wu 和 Vincent S Tseng。挖掘top-k关联规则。载于加拿大人工智能会议，第61–73页。Springer，2012年。[10] Akbar Telikani、Amir H Gandomi 和 Asadollah Shahbahrami。关联规则挖掘的进化计算综述。*Information Sciences*，524:318–352，2020年。[11] D Martín、Alejandro Rosete、Jesús Alcalá-Fdez 和 Francisco Herrera。QAR-CIP-NSGA-II：一种用于挖掘定量关联规则的新型多目标进化算法。*Information Sciences*，258:1–28，2014年。

[12] Kamel Eddine Heraguemi、Nadjet Kamel 和 Habiba Drias。采用多种协作策略进行关联规则挖掘的多群蝙蝠算法。*Applied Intelligence*，45(4):1021–1033，2016年。[13] Youcef Djenouri、Ahcene Bendjoudi、Djamel Djenouri 和 Marco Comuzzi。用于解决关联规则挖掘问题的基于GPU的仿生模型。载于*2017*年第*25*届*Euromicro*并行、分布式与基于网络的处理国际会议（*PDP*），第262–269页。IEEE，2017年。[14] Kin Gwn Lore、Adedotun Akintayo 和 Soumik Sarkar。LLNet：一种用于自然低光照图像增强的深度自编码器方法。*Pattern Recognition*，61:650–662，2017年。[15] Alan Ramponi 和 Barbara Plank。自然语言处理中的神经无监督域适应：综述。*arXiv*预印本 *arXiv:2006.00632*，2020年。[16] Yann LeCun、Yoshua Bengio 和 Geoffrey Hinton。深度学习。*Nature*，521(7553):436–444，2015年。[17] Erxue Min、Xifeng Guo、Qiang Liu、Gen Zhang、Jianjing Cui 和 Jun Long。深度学习聚类综述：网络架构视角。*IEEE Access*，6:39501–39514，2018年。[18] Fuzhen Zhuang、Xiaohu Cheng、Ping Luo、Sinno Jialin Pan 和 Qing He。监督式表示学习：利用深度自编码器进行迁移学习。载于第二十四届国际人工智能联合会议，2015年。[19] Dheeru Dua 和 Casey Graff。UCI机器学习知识库，2017年。[20] Zhenhua Zhang、Qing He、Jing Gao 和 Ming Ni。利用深度学习从社交媒体数据中检测交通事故的方法。*Transportation Research Part C: Emerging Technologies*，86:580–596，2018年。[21] Ana Valdivia、Eugenio Martínez-Cámara、Iti Chaturvedi、M Luzon、Erik Cambria、Yew-Soon Ong 和 Francisco Herrera。人们如何看待这座纪念碑？通过深度学习、聚类和描述性规则理解负面评论。*Journal of Ambient Intelligence and Humanized Computing*，11(1):39–52，2020年。[22] Konstantinos Vougas、Magdalena Krochmal、Thomas Jackson、Alexander Polyzos、Archimides Aggelopoulos、Ioannis S Pateras、Michael Liontos、Anastasia Varvarigou、Elizabeth O Johnson、Vassilis Georgoulias 等。利用深度学习和关联规则挖掘预测癌症药物反应：一种个性化医疗方法。*BioRxiv*，第070490页，2017年。[23] Peddi Kishor 和 Porika Sammulal。利用优化遗传算法和无监督神经网络进行关联规则挖掘。载于*2018*年通信与网络物理工程国际会议，第657–669页。Springer，2018年。[24] Xiaofeng Li、Dong Li、Yuanbei Deng 和 Jinming Xing。基于深度学习的复杂医疗数据智能挖掘算法。*Journal of Ambient Intelligence and Humanized Computing*，12(2):1667–1678，2021年。[25] Sebastian Raschka。Mlxtend：为Python科学计算技术栈提供机器学习与数据科学实用工具和扩展。*The Journal of Open SourceSoftware*，3(24)，2018年4月。[26] Kalyanmoy Deb、Amrit Pratap、Sameer Agarwal 和 TAMT Meyarivan。一种快速且精英保留的多目标遗传算法：NSGA-II。*IEEE Transactions on Evolutionary Computation*，6(2):182–197，2002年。[27] Félix-Antoine Fortin、François-Michel De Rainville、Marc-André Gardner Gardner、Marc Parizeau 和 Christian Gagné。DEAP：让进化算法变得简单。*The Journalof Machine Learning Research*，13(1):2171–2175，2012年。