文章编号:1007-5321(2019)06-0134-08 **DOI**:10.13190/j.jbupt.2019-126

基于标签关联性的多标签 Scratch 分类算法

彭 聪, 孙 岩, 戚 鹏
(北京邮电大学 计算机学院, 北京 100876)

摘要: 为了实现 Scratch 可视化编程领域的作品分类,提出了一种基于标签关联性的多标签分类算法(MLLR),构建了一个有效的多标签 Scratch 分类模型. 首先提取作品的 Block 使用特征、计算思维技能特征和复杂度特征 3 类特征作为分类特征;然后针对 RAKEL 算法随机选择标签子集,忽略了标签间的关联性,提出了改进的 MLLR 算法,该方法根据多标签之间的关联性来划分标签子集,再训练相应的标签幂集子分类器. 实验结果表明,MLLR 算法在分类性能和时间性能上优于 RAKEL 等多标签分类算法,构建的分类模型对于 Scratch 作品具有较强的适用性,分类的准确率达到 81.3%.

关 键 词: Scratch; 标签关联性; 多标签分类; 分类模型
中图分类号: TP301.6 **文献标志码**: A

Label Relevance Based Multi-Label Scratch Classification Algorithm

PENG Cong, SUN Yan, QI Peng
(School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract: In order to implement the classification of projects in visual programming field of Scratch, a multi-label classification algorithm (MLLR) appears based on label relevance. An effective multi-label classification model for Scratch projects was constructed. Firstly, the block usage features, the computational thinking skill features and the Halstead features of projects are extracted as classification features. Then, the RAKEL algorithm randomly chooses label subsets, ignoring the relevance between labels, thereafter an improved MLLR algorithm was proposed. This method divides label subsets according to the relevance between multiple labels, and then trains the corresponding label power set sub-classifiers. Experiments show that MLLR algorithm is superior to RAKEL and other multi-label classification algorithms in classification performance and time performance, The classification model constructed has a strong applicability for Scratch projects, and the accuracy of classification reaches 81.3%.

Key words: Scratch; label relevance; multi-label classification; classification model

Scratch 是一种基于块的可视化编程语言,由 MIT 实验室开发,已经逐渐在少儿编程领域被广泛使用. 创作者通过拖拉积木块,可以创作出各种 Scratch 作品. Scratch 作品在风格上具有不同的类别,如游戏、动画、故事、音乐、艺术等. 研究 Scratch 作品的自动分类,构建出一个有效的 Scratch 分类模型,可以给创作者提供有效的作品类型提示,还可以进一步探索少儿的编程心理和创作偏好,为个性化

收稿日期: 2019-11-22
基金项目: 国家自然科学基金项目(61672109, 61772085, 61877005)
作者简介: 彭 聪(1995—), 男, 硕士生.
通信作者: 孙 岩(1970—), 女, 教授, 博士生导师, E-mail:sunyan@bupt.edu.cn.

教育提供更多参考. Scratch 作品含有多个类别，如有的作品同时含有动画和故事标签，有的作品含有音乐和艺术标签，因此 Scratch 作品分类问题属于多标签分类问题，需要研究有效的分类特征和适用的多标签分类算法.

为了提高对 Scratch 作品分类的准确性，笔者首先研究提取了 Scratch 作品的 3 种特征，将其作为分类特征；然后针对 RAKEL 算法存在随机选择标签子集，忽略了标签关联性的问题，在 RAKEL 算法的基础上进行改进，提出了一种基于标签关联性的多标签分类（MLLR，a multi-label classification algorithm based on label relevance）算法；最后建立了一个有效的多标签 Scratch 分类模型.

1　相关工作

关于 Scratch 可视化编程领域的研究，在计算思维（CT，computational thinking）评测方面，Chang 等[1]研究了一种 Scratch 评测工具，提出了一套 CT 技能评分标准，可以在 CT 技能方面对 Scratch 作品的创作水平进行评估. 在编程习惯检测方面，Techapalokul[2]将 Scratch 作品解析成抽象语法树，然后对 12 种不良的编程特征进行提取和检测，通过检测作品的编程特征来研究学生的不良编程习惯，进而提高编程能力. 在 Scratch 数据集构建方面，Aivaloglou 等[3]从 Scratch 官网爬取了 25 万个 Scratch 作品，通过对作品进行解析，构建了一个包含每个作品的元数据、程序源码及其类别数据的数据集. 在学习路线设计方面，Moreno 等[4]分析了动画、游戏、故事、音乐和艺术 5 类作品在 CT 技能分数上的差异，提出了一种基于数据驱动来制定学习路径的方法. 目前在 Scratch 作品自动分类方面的研究较少，因此需要研究合适的分类特征和适用性较强的多标签分类算法，构建出一个有效的 Scratch 分类模型.

在真实世界里，标签之间存在一定的关联性. 如在 Scratch 作品中，作品的类别之间不是相互独立的，具有一定的关联性，音乐类的作品极有可能带有艺术的标签. Szymański 等[5]通过研究发现，有效地利用标签之间存在的关联性，可以改善多标签的分类性能.

在多标签分类算法[6]方面，目前对标签关联性的考查方式可分为 3 种策略：一阶策略、二阶策略和高阶策略. 二元关联（BR，binary relevance）算法[7]属于一阶策略，不考虑标签之间的相关性，给每个标签训练一个二元分类器. 标签幂集二元关联（LP-BR，label power set binary relevance）算法[8]属于二阶策略，将具有关联性的标签对划分为一个子集. RAKEL 算法[9]属于高阶策略，通过随机选择 m 个大小为 k 的标签子集，然后给每个子集训练标签幂集（LP，label power set）子分类器，共 m 个分类器，预测时组合各分类器的分类结果，再通过投票处理后输出. RAKEL 算法随机选择标签子集，忽略了标签间的关联性[10]，会使标签存在随机关系，进而引入了过多的噪声，影响算法的分类性能.

2　多标签 Scratch 分类模型

2.1　模型描述

自动分类在机器学习领域属于有监督的学习任务，一般分为训练和分类 2 个过程. 多标签 Scratch 分类模型如图 1 所示，分为特征提取、多标签 MLLR 算法 2 个模块.

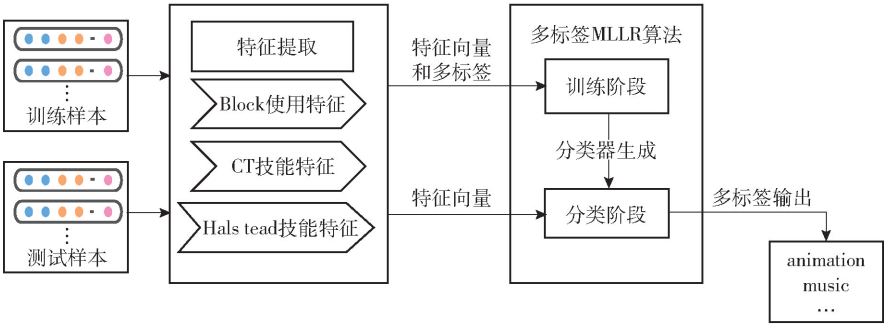

图 1　多标签 Scracth 分类模型

特征提取模块负责构建和提取 Scratch 作品的特征数据，包括 Block 使用特征、CT 技能特征和复杂度特征. 每个 Scratch 作品的 sb2 文件是一种半结构化数据，特征提取模块负责将这些半结构化的原始数据转换为结构化形式的特征向量，使其可以方便地输入到多标签 MLLR 算法中进行训练和分类.

多标签 MLLR 算法分为训练和分类 2 个阶段，对于 Scratch 训练样本，将经过特征提取后得到的特征数据及相应的多标签组成训练集，输入到多标签 MLLR 算法中进行训练，生成 Scratch 分类器. 对于测试样本，先进行特征提取，把得到的特征数据输入到多标签 MLLR 算法中进行分类，由 Scratch 分类器输出多标签的分类结果.

2.2 特征提取

Scratch 作品本质上是一种 sb2 文件，选用开源的解析工具 SAT 对文件进行解析. 通过设计语法解析规则和特征提取规则，得到 Scratch 作品的 3 类特征数据，分别是 Block 使用特征、CT 技能特征和复杂度特征，最后将它们组合成作品的特征向量.

1）Block 使用特征

Scratch 作为一门基于块的可视化编程语言，含有 9 大类模块共 119 种块（Block）. 如图 2 所示，Scratch 作品的创作是通过组合不同的积木块来实现的. 笔者借鉴了自然语言文本处理领域词嵌入的词频逆文本频率（TF-IDF，term frequency-inverse document frequency）技术[11]，TF-IDF 的主要思想是：如果某个词在一个句子中出现的频率高，并且在其他句子中很少出现，则认为此词对该句子具有良好的类别区分能力，适合用来分类，应赋予较高的 TF-IDF 值. 类似地，笔者定义了一种 Block 的 TF-IDF，用来表示作品的 Block 使用特征. 其主要思想是：当某种 Block 在一个作品中使用的频率高，并且在其他作品中较少出现时，则认为此 Block 对该作品具有较好的类别区分能力，应赋予较高的 TF-IDF 值；反之，当使用此 Block 的作品越多，即该 Block 被普遍使用时，则该 Block 对这个作品的类别区分能力较弱，应赋予较低的 TF-IDF 值.

图 2 Scratch 作品的创作

令频数 J_{Block} 表示一个作品中这种 Block 的使用数量，P 代表一个作品中所有 Block 的数量，S 为作品集中所有作品的数量，Q_{Block} 为使用了这种 Block 的作品数.

该作品中此 Block 的使用频率为

$$F_{\text{Block}} = \frac{J_{\text{Block}}}{P} \tag{1}$$

逆作品频率为

$$I_{\text{Block}} = \text{lb}\,\frac{S}{Q_{\text{Block}}+1} \tag{2}$$

Block 的 TF-IDF 值为

$$N_{\text{Block}} = F_{\text{Block}} I_{\text{Block}} \tag{3}$$

每个作品的 Block 使用特征向量为

$$\boldsymbol{B} = [N_1, N_2, \cdots, N_{119}] \tag{4}$$

其中 N_i 为第 $i(1 \leqslant i \leqslant 119)$ 种 Block 的 TD-IDF 值，共 119 项特征.

通过 Block 使用特征提取实验发现，不同类别的作品使用 Block 的情况不同，如表 1 所示. 表中各 Block 的名称使用英文缩写表示，动画类作品使用造型切换，改变角色大小等 Block 较多；游戏类使用按键侦听，角色点击等 Block 较多.

表 1 各类别作品 TF-IDF 值前 5 的 Block

序号	动画	游戏	故事	音乐	艺术
1	LL	WP	SD	WI	GF
2	SS	WEF	TD	PS	WEF
3	CS	NC	WS	DI	LL
4	WEF	DR	NC	CV	DF
5	ST	WR	LL	RV	HL

2）CT 技能特征

作品的 CT 技能分数是评估 Scratch 作品编程创作水平高低的重要指标. Chang 等[1]提出的 CT 技能评分准则将 CT 技能分为 7 个维度，包括逻辑性、同步、并行、流控制、用户交互、数据表示和抽象化. 将 7 个维度得分所占的比重作为作品的 CT 技能特征，具体定义如下.

T_i 表示作品在第 i 个维度上的得分，Z 表示作品总分，作品第 i 个维度得分的比重为

$$K_i = \frac{T_i}{Z} \tag{5}$$

作品的 CT 技能特征向量为

$$\boldsymbol{C} = [K_1, K_2, \cdots, K_7] \tag{6}$$

共 7 项特征.

如表 2 所示，通过 CT 技能特征提取实验发现，作品在 CT 技能分数上的表现情况和作品的类别有关，游戏类作品在逻辑性、同步、用户交互等维度的平均得分明显高于其他类别作品，动画类作品在并

行、流控制等维度的平均得分较高，艺术类作品的逻辑性得分明显低于其他作品.

表 2　各类别作品的 CT 技能分数

维度	动画	游戏	故事	音乐	艺术
逻辑性	0.921	2.839	0.875	0.920	0.501
同步	1.765	3.334	2.221	1.952	1.436
并行	2.534	2.376	2.245	1.603	1.340
流控制	2.107	2.874	1.654	2.051	1.723
用户交互	1.846	2.158	1.910	1.745	1.476
数据表示	1.439	2.716	1.513	1.640	1.214
抽象化	2.225	2.382	2.207	2.289	1.924

3）复杂度特征

Moreno-León 等[12]提出了一种程序代码复杂度的度量方法，通过操作符和操作数的数量来计算程序代码的复杂度. D 表示程序代码的圈复杂度，E 表示程序代码编写需要的努力度，作品的复杂度特征向量为

$$\boldsymbol{H}=[D,E] \tag{7}$$

共 2 项特征. 复杂度特征反映了 Scratch 作品的复杂程度等特征.

通过复杂度特征提取实验发现，不同类别的作品复杂度有所不同. 如表 3 所示，游戏类作品的平均复杂度 D 远高于其他作品，音乐和艺术类的较低.

表 3　各类别作品的复杂度特征

复杂度	动画	艺术	音乐	游戏	故事
D	156.3	103.2	110.4	231.4	124.7
E	656.4	423.1	496.8	1 157.0	531.9

将作品的 Block 使用特征、CT 技能特征和复杂度特征这 3 类特征作为 Scratch 作品的分类特征. 作品经过特征提取模块处理后，输出作品的分类特征向量，共 128 项特征.

3　MLLR 算法

3.1　算法思路

RAKEL 算法在标签子集划分时随机选择标签子集，忽略了标签间的关联性. 为此，笔者在 RAKEL 算法的基础上进行改进，提出了一种 MLLR 算法，通过挖掘强关联规则来确定关联标签组，按照标签间的关联性进行标签子集的划分，以此来改善分类效果.

MLLR 算法的基本思路是：首先通过 FP-Growth 算法[13]挖掘多标签之间的强关联规则，强关联规则包含的项集具有强关联性，将具有强关联性的多个标签划分为一个标签子集，将不和其他任何标签关联的标签单独划分为一个标签子集；待标签子集划分完毕后，对每个标签子集训练相应的 LP 子分类器，训练完后得到 LP 分类器集合；当预测一个新的实例时，利用 LP 分类器集合中的每个分类器对新的实例进行预测，统计所有分类器的分类结果，通过投票策略来判断标签是否存在，最后输出实例的多标签分类结果.

3.2　算法描述

MLLR 算法分为标签子集划分、训练阶段和分类阶段 3 个步骤.

算法 1 描述了根据多标签间的关联性来划分标签子集的过程. 每个标签都是一个项 item，数据集中每个实例所含有的标签集合形成一个事务 transaction（步骤 4 ~ 6）. 通过 FP-Growth 算法挖掘出多标签间的关联规则 RS（步骤 7），从 RS 中选取出置信度大于阈值的关联规则，得到强关联规则 MR（步骤 8）. MR 所包含的标签具有强关联性，将具有强关联性的多个标签划分为一个子集（步骤 9 ~ 15），将不和任何标签关联的标签单独划分为一个标签子集（步骤 16 ~ 18），最后输出标签子集的集合 R.

算法 1　MLLR 算法的标签子集划分过程

输入：标签集 L，训练集 D

输出：标签子集的集合 R

```
1   T ← Label set per instance (transactions);
2     for j←1 to |L| do
3       Unique_j←0;
4     for each instance X_i in D do
5       L_i←labels of X_i;
6       T←T∪L_i;
7     RS←FPGrowth(T);
8     MR←Sub(RS);
9     for i←1 to |MR| do
10      C_i←a rule randomly selected from MR;
11      L_i←labels of C_i;
12      R←R∪L_i;
13      MR←MR\{C_i};
14      forall labels λ_j ∈ L_i do
15        Unique_j←1;
16  for j←1 to |L| do
```

17 if $\text{Unique}_j < 1$ then
18 $R \leftarrow R \cup \{\lambda_j\}$;

算法2描述了MLLR算法的训练阶段. 首先通过算法1完成标签子集划分,得到划分后的标签子集的集合R(步骤1);然后针对每个标签子集Y_i训练相应的LP子分类器h_i(步骤4),共训练$|R|$个LP子分类器,直到所有分类器训练完毕,最后得到LP分类器集合.

算法2 MLLR算法的训练阶段

输入: 标签集L,训练集D

输出: LP分类器h_i集合和其对应的标签集Y_i

1 $R \leftarrow \text{LabelSetPartition}(D, L)$;
2 for $i \leftarrow 1$ to $|R|$ do
3 $Y_i \leftarrow$ a labelset randomly selected from R;
4 train an LP classifier $h_i: X \rightarrow P(Y_i)$ on D;
5 $R \leftarrow R \backslash \{Y_i\}$;

算法3描述了MLLR算法的分类阶段,对于任意标签$\lambda_j \in L$,Result_j为1表示该标签存在,Result_j为0表示该标签不存在.

算法3 MLLR算法的分类阶段

输入: 测试实例x,LP分类器h_i集合以及相应的标签集Y_i,标签集L

输出: 多标签LP分类器集合分类结果Result

1 for $j \leftarrow 1$ to $|L|$ do
2 $\text{Sum}_j \leftarrow 0$;
3 $\text{Votes}_j \leftarrow 0$;
4 for $i \leftarrow 1$ to $|R|$ do
5 forall labels $\lambda_j \in Y_i$ do
6 $\text{Sum}_j \leftarrow \text{Sum}_j + h_i(x, \lambda_j)$;
7 $\text{Votes}_j \leftarrow \text{Votes}_j + 1$;
8 for $j \leftarrow 1$ to $|L|$ do
9 $\text{Avg}_j \leftarrow \text{Sum}_j / \text{Votes}_j$;
10 if $\text{Avg}_j > t$ then
11 $\text{Result}_j \leftarrow 1$;
12 else $\text{Result}_j \leftarrow 0$;

对于一个新的实例x,使用LP分类器集合中的每个LP子分类器h_i对其进行预测,将每个分类器的分类结果进行投票处理. 对于LP子分类器h_i,如果标签λ_j($\lambda_j \in Y_i$)预测存在,则$h_i(x, \lambda_j)$为1;否则为0. 统计完所有分类器h_i的分类结果后,计算每个标签λ_j($\lambda_j \in L$)的得票率,如果大于阈值,则该标签存在,Result_j输出为1;否则该标签不存在,Result_j输出为0.

4 实验及结果分析

4.1 实验数据集

笔者以Scratch官网上标注类别的作品作为实验的数据集,共爬取了4万多个作品的sb2文件及其对应的类别标签. 将数据集随机抽取70%作为训练集,剩下30%作为测试集进行实验. 实验所用Scratch数据集的相关统计信息如表4所示.

表4 Scratch数据集的基本统计信息

Instances	Labels	Cardinality	Density
48 445	5	2.201	0.440 2

实验数据集共有48 445个作品实例,每个作品特征向量的维度为128,多标签总数为5种,其中含有game标签的作品数为22 380,含有animation标签的作品数为24 187,含有story标签的作品数为18 525,含有music标签的作品数为21 287,含有art标签的作品数为20 246,总体上各类别的作品数量比较均衡.

4.2 评价指标

多标签分类任务的评价指标[14]分为两类:一类是基于样本的评价指标;另一类是基于标签的评价指标. 笔者选用了HammingLoss、SubsetAccuracy、M_{macro}、M_{micro}这4个常见的评价指标来评测分类算法性能.

1) 基于样本的评价指标

D表示测试数据集,L表示标签集合,Y_i表示实例X_i实际的标签集合,Z_i表示实例X_i预测的标签集合.

汉明损失定义如下:

$$\text{HammingLoss}(h, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|Y_i \Delta Z_i|}{|L|} \quad (8)$$

其中Δ代表2个集合的对称差分. 该指标用来评测算法的预测标签与真实标签之间的匹配失准率,HammingLoss值越小,说明算法的性能越好.

子集准确度定义如下:

$$\text{SubsetAccuracy}(h, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} I(Y_i = Z_i) \quad (9)$$

其中:$I(\text{true}) = 1$,$I(\text{false}) = 0$. 该指标用来评测算法的预测标签与真实标签之间的匹配率,Subset Accuracy值越大,表示算法的性能越好.

2) 基于标签的评价指标

对于标签集合L中的第j个标签,公式中符号

所代表的意义如表 5 所示.

表 5　各符号所代表的信息

第 j 个标签		实际的标签	
		含有标签 j	不含标签 j
预测的标签	含有标签 j	a_j	w_j
	不含标签 j	x_j	y_j

宏观平均值定义如下：

$$M_{\text{macro}}=\frac{1}{|L|}\sum_{j=1}^{|L|}M(a_j,w_j,x_j,y_j) \tag{10}$$

其中 $M(a_j,w_j,x_j,y_j)=\dfrac{2a_j}{2a_j+w_j+x_j}$. 该指标也称为宏观综合分类率，是精确率和召回率结合得到的评价指标，M_{macro} 值越大，表示分类算法的性能越好.

微观平均值定义如下：

$$M_{\text{micro}}=M\left(\sum_{j=1}^{|L|}a_j,\sum_{j=1}^{|L|}w_j,\sum_{j=1}^{|L|}x_j,\sum_{j=1}^{|L|}y_j\right) \tag{11}$$

4.3　实验结果分析

将所提出的 MLLR 算法与多标签邻近算法（ML-KNN，multi label k nearest neighbor）、RAKEL 算法、BR 算法、LPBR 算法 4 种多标签分类算法进行实验比较. 使用梯度提升决策树算法作为 BR 算法中的子分类器算法，LPBR 算法中的子分类器算法，RAKEL 算法中的 LP 子分类器算法以及 MLLR 算法中的 LP 子分类器算法. 对于 RAKEL 算法，将参数 k 设置为 3，参数 m 设置为 $2|L|$（标签总数的 2 倍）；对于 ML-KNN 算法，设置参数 k 为 10；对于 MLLR 算法，设置 FP-Growth 的参数支持度为 0.025，设置置信度的阈值为 0.5，设置分类阶段投票的阈值 t 为 0.5. 通过改变数据集的样本数量 m 进行多次实验，每次实验使用留出法，训练和测试的样本比例为 7∶3. 图 3 给出了各算法在不同规模数据集上的实验结果.

由实验结果可以发现，随着数据集样本数量的不断增大，各算法的分类性能都有提升，最后趋于稳定. MLLR 算法在各项性能评价指标上都优于 RAKEL 算法，MLLR 算法根据多标签之间的关联性来划分标签子集，而 RAKEL 算法是随机选择标签子集，引入了过多的噪声. 与其他 3 种考虑标签之间不同阶相关性的多标签分类算法相比，MLLR 算法在大部分情况下，都能达到最好的性能；BR 算法和 ML-KNN 算法将每个标签独立看待，没有考虑标签间的关联性，分类性能不佳；LPBR 算法在数据集

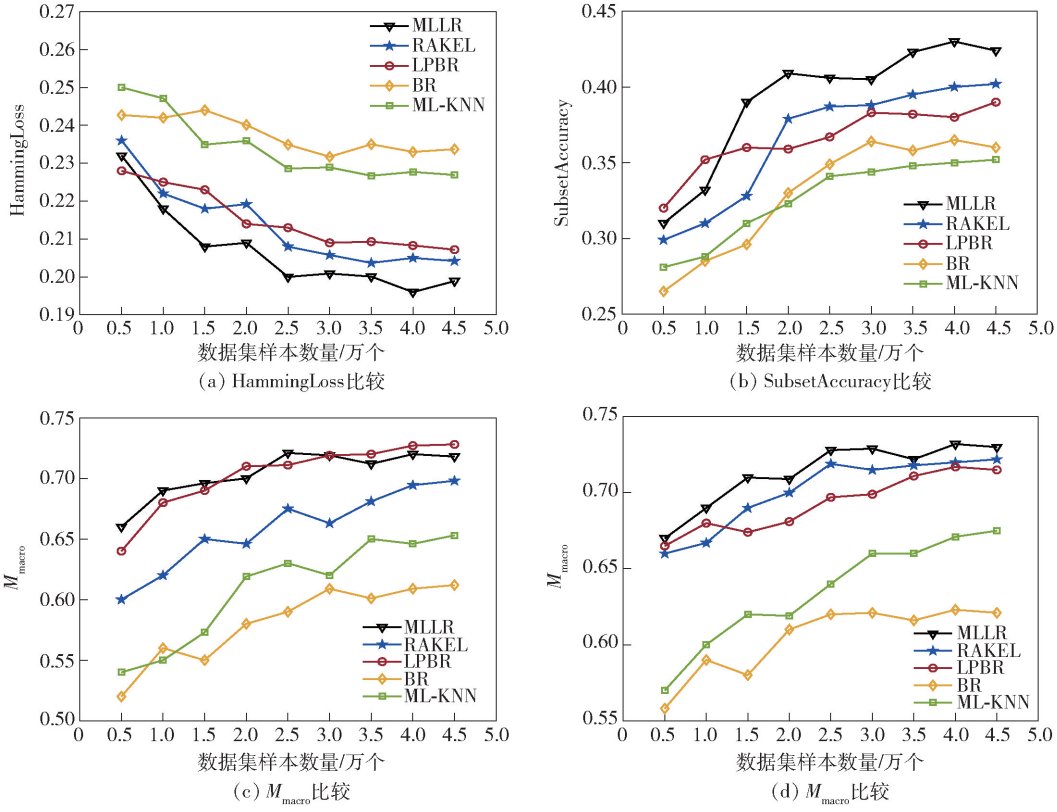

图 3　各算法的分类性能对比

规模较小时,具有较好的分类性能,随着数据集规模的增大,只考虑两两标签,不能充分利用多标签之间的关联性,分类性能没有较大提高. 从整体上看,MLLR 算法有效地利用了多个标签之间的关联性,从而很好地提高了算法的分类性能.

时间性能上,由图 3 可得,当样本数量 m 为 4 万时,MLLR 算法在各项评价指标上具有最优的分类性能. 图 4 给出了此时各算法在训练和分类 2 个阶段的时间开销对比. 在训练阶段 BR 和 ML-KNN 算法需要的时间较少,LPBR、RAKEL 和 MLLR 算法考虑了标签间的关联性,需要划分标签子集,再训练多个 LP 分类器,因此训练阶段所需的时间较长. MLLR 算法需要挖掘强关联规则,时间复杂度略高于 RAKEL 算法. 在分类阶段,BR 算法速度最快,但其分类效果不佳;ML-KNN 算法最慢,它需要计算距离最近的 k 个邻点的信息;MLLR 算法将多个具有关联性的标签划分为一个标签子集,生成的 LP 子分类器数量比 LPBR 和 REKEL 算法少,因此 MLLR 算法的分类速度比 LPBR 和 RAKEL 算法快,在分类阶段的时间性能上表现较好.

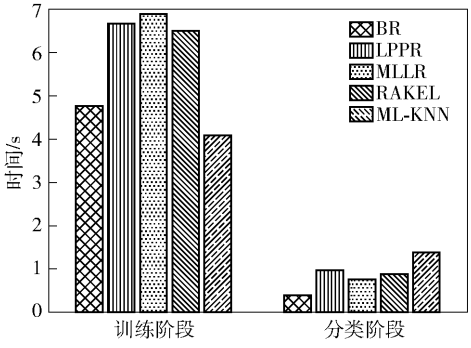

图 4 各算法的时间性能对比

当数据集样本数量 m 为 4 万时,由 MLLR 构建的 Scratch 分类模型具有最好的性能. 此时如果独立地看待每个标签,单独计算每类标签的分类准确率,动画和游戏类标签的分类准确率较高,如表 6 所示. 总体上看,分类模型的平均准确率达到了 81.3%.

表 6 MLLR 分类模型在 5 类标签上的分类准确率

标签	分类准确率	标签	分类准确率
动画	0.863	音乐	0.793
游戏	0.872	艺术	0.756
故事	0.784	平均	0.813

5 结束语

为了提高对 Scratch 作品分类的准确性,笔者通过研究 Scratch 作品的特点,提取了 Scratch 作品的 Block 使用特征、CT 技能特征和复杂度特征 3 类特征作为作品的分类特征. 在分类算法方面,针对 RAKEL 算法随机选择标签子集,忽略了标签间的关联性. 笔者在 RAKEL 算法的基础上进行改进,提出了一种 MLLR 算法,通过挖掘多标签间的强关联规则来确定关联标签组,根据多标签之间的关联性来划分标签子集,再训练相应的 LP 子分类器. 实验结果表明,MLLR 算法在分类性能和时间性能上优于 RAKEL 等多标签分类算法,所构建的 Scratch 分类模型对于 Scratch 作品具有较强的适用性,分类的准确率达到 81.3%.

参考文献:

[1] Chang Zhong, Sun Yan, Wu Tinyu, et al. Scratch analysis tool (SAT): a modern scratch project analysis tool based on ANTLR to assess computational thinking skills [C]// 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC). Limassol: IEEE Press, 2018: 950-955.

[2] Techapalokul P. Sniffing through millions of blocks for bad smells[C]// Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education-SIGCSE'17. Washington: ACM Press, 2017: 781-782.

[3] Aivaloglou E, Hermans F, Moreno-Leon J, et al. A dataset of scratch programs: scraped, shaped and scored[C]// 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR). Buenos Aires: IEEE Press, 2017: 511-514.

[4] Moreno-Leon J, Robles G, Roman-Gonzalez M. Towards data-driven learning paths to develop computational thinking withscratch[J]. IEEE Transactions on Emerging Topics in Computing, 2017, 14(6): 185-197.

[5] Szymański P, Kajdanowicz T, Kersting K. How is a data-driven approach better than random choice in label space division for multi-label classification? [J]. Entropy, 2016, 18(8): 282-305.

[6] Zhang Minling, Zhou Zhihua. A review on multi-label learning algorithms[J]. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(8): 1819-1837.

[7] Zhang Yahong, Li Yujian, Cai Zhi. Correlation-based pruning of dependent binary relevance models for multi-label classification[C]// 2015 IEEE 14th International

Conference on Cognitive Informatics & Cognitive Computing (ICCI * CC). Beijing, China: IEEE Press, 2015: 399-404.

[8] Lena T, Lior R, Bracha S. Identification of label dependencies for multilabel classification[C]// Proceedings of the 2nd International Workshop on Learning from Multi-Label Data. Dublin, Ireland: IEEE, 2010: 53-60.

[9] Tsoumakas G, Katakis I, Vlahavas I. Random *k*-labelsets for multilabel classification[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(7): 1079-1089.

[10] Charte F, Rivera A, del Jesus M J, et al. Improving multi-label classifiers via label reduction with association rules[M]// Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012: 188-199.

[11] Liu Caizhi, Sheng Yanxiu, Wei Zhiqiang, et al. Research of text classification based on improved TF-IDF algorithm[C]// 2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE). Lanzhou, China: IEEE Press, 2018: 218-222.

[12] Moreno-Leon J, Robles G, Roman-Gonzalez M. Comparing computational thinking development assessment scores with software complexity metrics[C]//2016 IEEE Global Engineering Education Conference (EDUCON). Abu Dhabi, UAE: IEEE Press, 2016: 1040-1045.

[13] Wang Zezhong, Cao Shuo. A power load association rules mining method based on improved FP-growth algorithm[C]// 2018 China International Conference on Electricity Distribution (CICED). Tianjin, China: IEEE Press, 2018: 2833-2837.

[14] Gibaja E, Ventura S. Atutorial on multilabel learning [J]. ACM Computing Surveys, 2015, 47(3): 1-38.

(上接第 133 页)

[10] Liu Xiaodong, Shen Yelong, Duh K, et al. Stochastic answer networks for machine reading comprehension[C]// Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg, PA, USA: Association for Computational Linguistics, 2018: 1694-1704.

[11] Levy O, Seo M, Choi E, et al. Zero-shot relation extraction via reading comprehension[C]// Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017). Vancouver, USA: Association for Computational Linguistics, 2017: 333-342.

[12] Pennington J, Socher R, Manning C. Glove:global vectors for word representation[C]// Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha: Association for Computational Linguistics, 2014: 1532-1543.

[13] McCann B, Bradbury J, Xiong C M, et al. Learned in translation: contextualized word vectors[C]// Advances Inneural Information Processing Systems. New York: Curran Associates, 2017: 6294-6305.

[14] Huang H Y, Zhu C G, Shen Y L, et al. Fusionnet: fusing via fully-aware attention with application to machine comprehension[EB/OL]. 2017 (2018-02-04) [2019-11-18]. https://arxiv.org/abs/1711.0734-1v2.