文章编号:1007-5321(2019)06-0070-06 DOI:10.13190/j.jbupt.2019-125

一种基于 ANTLR 的面向 Scratch3.0 的特征提取和检测系统

刘 派, 孙 岩, 任 玮
(北京邮电大学 北京市智能通信软件与多媒体重点实验室, 北京 100876)

摘要: Scratch 是一种适合少年儿童使用的可视化编程语言,并在全球的编程教育领域中受到广泛地关注. 由于目前各大教育编程平台都开始使用 Scratch3.0 版本,而已有的特征提取和检测系统并不支持新版本,为此,提出了一种基于链表数据结构和一种语言识别工具(ANTLR)的面向 Scratch3.0 的特征提取和检测系统. 实验结果表明,该系统可以有效地从项目中提取编程特征,并为学生和教师提供反馈,其检测性能和检测稳定性比 Scratch2.0 均有所提升.

关 键 词: Scratch; 一种语言识别工具; 特征检测; 特征提取
中图分类号: TP301.6 **文献标志码**: A **OSID 码**:

An ANTLR-Based Feature Extraction and Detection System for Scratch3.0

LIU Pai, SUN Yan, REN Wei
(Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia,
Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract: As a visual programming language for children, Scratch has received wide attention in the programming education. Considering that Scratch has evolved to the latest version 3.0 and its storage structure changes significantly from the previous version, the existing methods cannot be directly applied to project analysis. A new feature extraction and detection system based on linked list data structure and another tool for language recognition (ANTLR) was presented to solve the problem. Experimental results show that the system can effectively extract programming features from the projects and provide feedback to students and teachers. Moreover, its detection performance and stability perform better than the original methods in Scratch2.0.

Key words: Scratch; another tool for language recognition; feature extraction; feature detection

Scratch 是麻省理工学院推出的可视化编程社区,学生可以在其上创建、改编和发布项目. 目前,该编程社区上有超过 300 万注册用户和 300 万个来自全球各地的共享项目. 诸如用户和项目信息等大量的共享数据能够构建分析 Scratch 编程功能的提取和检测模块,基于该系统获得的编程作品检测结果可以帮助学生改进作品质量,进而提升编程能力.

目前各教育编程平台都开始使用 Scratch3.0 版本,而已有的特征提取和检测系统并不支持这一新版本,主要原因是在 Scratch2.0 的存储文件中,编程

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: 刘 派(1993—), 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.

块之间构成了类似数组的高度耦合的结构，而在 Scratch3.0 中每个编程块以特有的 id 编号进行解耦，构成了类似链表的数据结构. 因此，基于之前对 Scratch2.0 版本的特征提取和检测的相关工作，针对 Scratch3.0 版本在 JSON 文件数据结构上的特点，采用解耦的设计思想[1]对特征的提取检测步骤进行优化. 最终，经过与 Scratch2.0 进行的对比实验证明其达到了预期的检测效果.

1　相关工作

Meerbaum-Salant 等[2]将 Scratch 产生的编程习惯大致分为 2 类：从单个 Scratch 块开始，完全自下而上的开发过程；极其细粒度的编程. 由此了解到不良的编程习惯可能会对学生将来学习高级编程语言产生不良的影响，因此有必要对 Scratch 编程作品进行特征检测并对类型加以区分.

Wolz 等[3]开发了一种可以帮助人们理解 Scratch 设计模式的工具. Boe 等[4]开发了一种名为 Hairball 的 Scratch 静态分析框架，它可以帮助理解 Scratch 的诸如嵌套级别、循环次数等模式. 由此可以粗略地区分用户的行为模式，这也为后续的特征挖掘提供了理论基础.

Techapalokul 等[5]采用抽象语法树(AST，abstract syntax tree)的思想对 Scratch 的 JSON 格式文件进行解析. 分析器使用基于 Java 的语言处理框架 JastAdd[6]实现，最终检测到的代码特征由第三方 JavaScript 库在浏览器中呈现给用户. 他们提出了 12 种检测代码块编程特征的方法[7]，并通过 Scratch 数据集验证了其编程特征检测方法的有效性. 但是对于具有未初始化脚本等功能的项目，它们缺乏相应的检测规则. 因此，提出了 9 种新的编程特征检测规则，并增强了检测结果的可靠性.

Fowler[1]揭示了重构过程并提到了面向对象软件开发中常见的 22 种不利于代码重构和复用的代码坏味道，这给特征检测提供了许多借鉴之处.

由于 Scratch 程序是在图形用户界面中开发和运行的，检测 Scratch 程序的各种编程特征有一定的困难，因此从 Scratch 项目的 JSON 文件入手，该 JSON 文件中保存有 Scratch 项目完成后的块的所有静态信息. Quality Hound[5]是一种能够自动检测 Scratch 作品多种编程特征的在线工具. Quality Hound 对 Scratch 的 JSON 文件利用 AST 的思路解析为分析程序使用的内部表示，最终可将 AST 解析为不同的语法子树.

作为一种通过语法描述来自动构造语法解析生成器的框架，语言识别的另一个工具(ANTLR，another tool for language recognition)[8]具有 LL(*)解析策略灵活性的优势，因此选择 ANTLR 来实现该编程特征检测功能. 补充了 9 种新的编程特征检测规则，实验结果表明，新增的特征检测效果也达到了已有的 12 种编程习惯检测的平均检测水平，具有较好的检测效果.

Scratch2.0 已经在少儿编程领域中使用得非常广泛，而 Scratch3.0 已于 2019 年 1 月 2 日正式取代 Scratch2.0. 已知目前有关 Scratch 的检测方法都是基于 2.0 版本的，并没有针对 3.0 版本提出新的检测方法. 但是 3.0 版本在底层的数据存储结构上有了巨大的变化，因此有必要针对 Scratch3.0 的数据结构新特性改进编程特征检测系统.

2　Scratch3.0 编程特征检测系统

2.1　系统描述

图 1 显示了 Scratch 中特征提取和特征检测的系统模型. 首先，从 Scratch 网站获取大量数据；然后，通过 Scratch Feature Extraction 子系统提取有效特征，这些有效功能由功能过滤器过滤并存储在特征数据库中. 用户可以拖放屏幕上的可视化编程块完成 Scratch 作品并提交到服务器，然后作品经过格式提取并将其输入到特征检测中，最后在特定的网页上输出分类结果和检测详情.

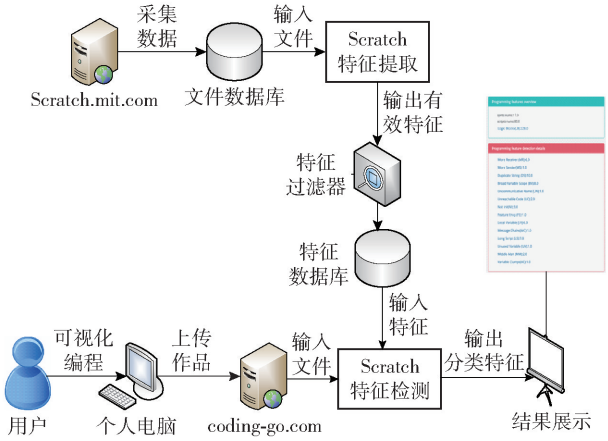


图 1　Scratch 特征提取与检测系统

2.2　特征提取和检测

由于 Scratch2.0 底层的数据结构中各个块堆放在一起，这一设计的不合理之处使得很难从 JSON

文件中抽象出便于解析的数据结构. 因此Scratch2.0的检测更适合采用基于ANTLR的特征检测,但是特征提取和特征检测过程都是基于ANTLR v4,而ANTLR与JSON数据格式耦合度很高,这导致特征检测规则的可读性较差,修改和后期完善的难度较大.

在Scratch3.0中,由于JSON信息数据结构设计得完善,采用了使特征提取和特征检测过程进行解耦的思想. 图2和图3分别展示了采用解耦思想的特征提取模块和特征检测模块.

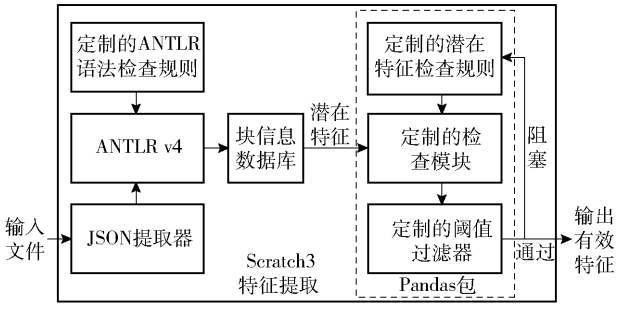


图2 Scratch3.0特征提取模块

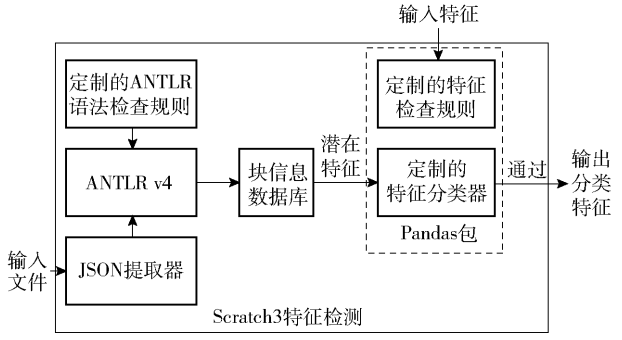


图3 Scratch3.0特征检测模块

在特征提取过程的预处理阶段,在遍历JSON的键值对的过程中ANTLR只负责对结构特征的数据提取,并将其转换为与JSON格式无关的数据表格式.

特征检测过程中则使用了基于Python语言的Pandas统计数据分析包,Pandas提供了很多类似SQL的查询统计功能,且相比Python,其性能更好,操作更简单.

3 实现方案

3.1 块信息数据表

在Scratch2.0的JSON文件中,相连接的块之间构成了脚本结构(类似于函数),并在JSON文件中耦合在一起. 而在Scratch3.0中每个编程块之间是独立的且都有一个唯一id,通过父节点id和子节点id确定块之间的先后位置,构成了类似链表的数据结构. 其好处是各个编程块之间是解耦的,缺点是存储的数据结构冗余度较大,比如某个作品中的Scratch2.0版本的JSON文件大小有17 KB,键值对总数有700个,而该作品的Scratch3.0版本中的JSON文件大小有41 KB,键值对总数则有2 647个. 因此,根据JSON数据结构[9]设计了可以存储Scratch3.0的JSON信息的数据表结构.

如表1所示,这里分为共有字段和各类型的特有字段,JSON中的任何信息块均属于以下5种类型之一,每个信息块包含共有字段和自身所属的数据类型的特有字段.

表1 Scratch块信息表

数据类型		具体字段		
共有字段		fileName	createTime	isStage
		name	data_type	
		detection_type	deteetion_sub_type	
各类型特有字段	variables	id	value	
	lists	id	value	
	broadcasts	id	value	
	blocks	id	opcode	next
		parent	inputs	fields
	comments	id	blockId	text

表1中的共有字段主要表明块所属角色的信息:fileName字段是指文件名称;isStage用于区分是否为舞台或是角色;name表明了该角色的名字(该名字不可重复);data_type指该块的数据类型(变量型、列表型、块型、注释型);detection_type和detection_sub_type存储的分别是块的检测信息和子类别信息,两者都在检测结束后由检测模块进行填充.

根据共有字段中的data_type类型,数据块信息中的特有字段各不相同. 根据对Scratch3.0的文件结构进行解析,将其归纳为5种块类型. 以blocks类型的块为例,它包含如fileName等所有的7个共有字段和id、opcode、next等blocks类型的6个特有字段.

3.2 基于Pandas的特征提取规则

基于常见的编程味道以及块信息的统计难度将特征检测分为如表2所示的3类.

表2中,“块信息基本统计”的统计方式最简单,主要是基于Pandas包进行计数统计;“块信息复杂统计”需要基于blocks类别块的inputs和fields结

合不同的数据表字段综合统计；而“脚本信息复杂统计”类则最复杂，需要构建新脚本结构并对序列进行对比.

这里构建脚本信息的方法是先筛选出所有能作为起始位置的编程块，然后从上述起始块结合 next 字段递归构建完整的脚本链表.

表 2　检测特征分类表

特征分类	特征名称		
块信息基本统计	全局变量数	无意义命名数量	自定义函数个数
	数据、事件、控制、运算块总数	注释个数	
块信息复杂统计	使用的字符串数	重复字符串数	死代码块数
	未初始化编程块数	未被定义的块个数	未被使用的变量数
	仅存在于一个角色的全局变量数	存在于 2 个角色的全局变量数	3 个及以上的角色的全局变量数
	1 发送器→n 接收器	n 发送器→1 接收器	
脚本信息复杂统计	无操作的脚本数	过长的脚本数	接口类型脚本数
	调用变量过多的脚本数	重复代码数	

3.3　检测算法

详细的 Scratch 检测算法如算法 1 所示.

整个过程大致分为 3 部分，即从 JSON 中提取 Scratch 块信息(2 ~ 9 行)、根据 Pandas 检测规则对 Scratch 块信息进行检测(10 ~ 15 行)、对检测结果进行统计和分类(17 ~ 20 行).

在第 1 部分中，采用了基于 ANTLR 的 JSON 解析技术[10]，根据设计的 Scratch 块信息表以及块信息提取规则[8]对 JSON 进行顺序遍历并根据提取规则将块信息加入到块结构数据库中.

在第 2 部分中，根据数据库中存储的块信息以及设计的 Pandas 检测规则对数据表进行高效的规则检测，然后将结果写入到预留在数据表中的 detection_type 和 detection_sub_type 字段中.

在第 3 部分中，基于预设的分类，系统将统计结果进行分类并输出分类统计结果.

算法 1　Scratch 特征检测

输入：Scratch 文件(file)；块结构信息(BSM)；Pandas 特征检测规则(PFDR)；特征分类规则(FCR)

输出：检测特征(DF)

```
1  for all file such that file ∈ JSON do
2  extract JSON from file
3  generate AST from JSON by ANTLR Parser
4  while ParseTreeWalker ∈ AST do
5    if enter a JSON rule then
6      collecting block message from BSM
7      save block message to BSM Database
8    end if
9  end while
10  while block message ∈ BSM Database do
11    if enter a PFDR then
12      collecting detection message from BSM Database
13      save detection message to detection Database
14    end if
15  end while
16  end for
17  for all feature such that feature.class ∈ PFDR do
18  count detection message from detection Database
19  add feature to FCR.class
20  end for
21  return DF
```

相比 Scratch2.0 中的检测方法，Scratch3.0 检测的最大优势在于将各部分检测规则拆分开并将过程结果进行保存. 同时，考虑到算法的扩展性单元测试的方便，调试时可以基于 Pandas 在每个步骤之间输出 csv 格式的过程结果，并且采用了和 Scratch2.0 检测算法一致的数据输出格式，增强了算法的可读性和扩展性.

4　实验与结果分析

4.1　实验准备

4.1.1　实验环境

为了尽可能与实际的使用环境一致，采用了与阿里云服务器相同配置的环境进行实验，硬件配置为 8 核 CPU、16 GB 内存，软件环境 Ubuntu 16.04 64 位、Python3.6，使用的 Chrome 浏览器版本为 59.0.3071.115.

4.1.2　实验数据集

本实验采用的数据集如表 3 所示.

Scratch 官网数据集：在 Scratch 官方编程网站上从不同题材的作品中随机下载了 500 个 Scratch2.0 作品和 500 个 Scratch3.0 作品. 根据 Scratch 官网的统计信息，用户的年龄分布主要集中在 9 ~ 18 周岁，并且各个年龄段的用户均有参与，用

户遍布美国、中国、西班牙、巴西等全球各大洲,因此基本可以代表全世界学生编程作品样本.

金华市中小学生计算机现场制作比赛数据集:与当地教育部门合作承办了金华市中小学生计算机现场制作比赛的活动并获取了金华市第一届中小学生计算机现场制作比赛的比赛作品. 本次比赛共有263人参加,共获取有效样本252份. 参赛选手全部使用Scratch2.0语言,需要在2 h内完成指定题材作品的制作. 与Scratch官网数据集相比,该数据集最大的特点是在当地教育局的监督下,经过严谨的比赛流程设计与实施,因此可以代表国内当前学生编程作品样本.

表3 实验数据集详情表

数据集	Scratch官网	Scratch官网	金华市中小学生Scratch竞赛	金华市中小学生Scratch竞赛
数据类型	scratch2	scratch3	scratch2	scratch3
数量	500	500	252	252
作者年龄	全年龄段	全年龄段	大部分为五年级和六年级,还有少量四年级	大部分为五年级和六年级,还有少量四年级
作品题材	非指定题材,包括游戏、音乐、故事和艺术等	非指定题材,包括游戏、音乐、故事和艺术等	指定题材	指定题材
限时完成	否	否	是,2 h	是,2 h
获得途径	各类别随机选取	各类别随机选取	由平台获取scratch2版本作品	经过官网的格式转换得到

为了进行对比,采用了将Scratch2.0作品上传到Scratch官网并转换为Scratch3.0然后保存下来的方法来获取对应的Scratch3.0版本作品. 经过人工检查确认,经过转换后的Scratch3.0保留了所有的Scratch2.0信息,实现了无损转换.

4.2 算法性能

为了在对比实验中有效控制唯一变量,采用了金华比赛数据集的Scratch2.0和Scratch3.0进行性能对比,根据同一个作品的2.0和3.0版本进行耗时对比,对比结果如图4所示.

基于对Scratch2.0和Scratch3.0的检测算法原理进行分析,可知两者的检测算法时间复杂度均为$O(n)$,为线性变化. 从图4中可以看到,当单个作品的规模增大时,Scratch2.0的检测算法明显要比

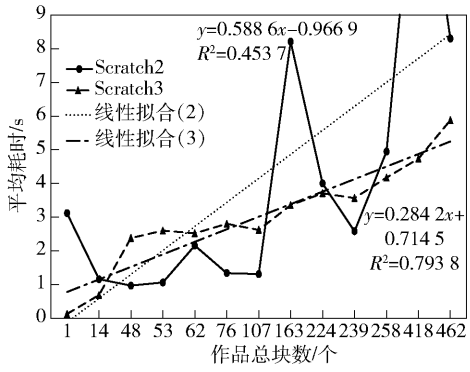


图4 算法性能对比

Scratch3.0增长得快. 经过线性拟合大致得到Scratch2.0的性能拟合曲线为$y=0.588\,6x-0.966\,9$,其中$R^2=0.453\,7$;Scratch3.0的性能拟合曲线为$y=0.284\,2x+0.714\,5$,其中$R^2=0.793\,8$. 因此,可知Scratch3.0检测算法的增长速度约为Scratch2.0的一半,在大规模作品的检测上优势明显.

4.3 算法稳定性

为了在对比实验中有效控制唯一变量,采用了金华比赛数据集的Scratch2.0和Scratch3.0进行性能对比,根据同一个作品的2.0和3.0版本进行耗时分布对比,对比结果如图5所示.

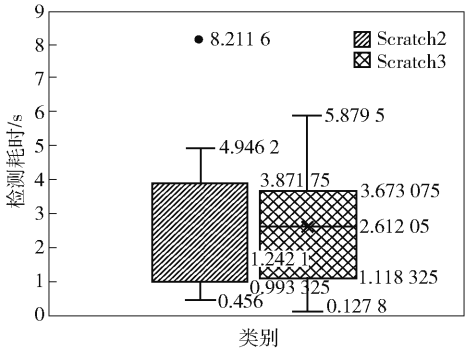


图5 算法稳定性对比

由图5可知,整体来看,Scratch2.0的耗时分布与Scratch3.0基本一致,中位数Scratch2.0比Scratch3.0要低,但细节上Scratch2.0会出现波动较大的异常值点且分布更分散,而Scratch3.0整体分布更集中,性能更稳定.

4.4 算法检测成功率

检测成功率需要采用更广泛的数据集,因此采用了从Scratch官网的数据集. 检测结果如图6所示,整体来看两者的识别率稳定在95%上下且相差不大并均已满足识别预期的效果.

考虑到codinggo-Scratch3.0检测采用解耦的设

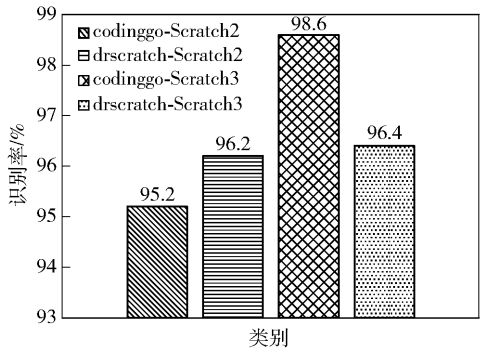


图 6　算法功能检测对比

计思路,因此算法有良好的扩展性和可读性,进一步提升识别率更容易,因此 codinggo-Scratch3.0 的检测算法潜力更大.

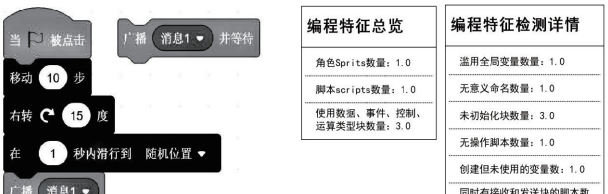
同时,也横向对比了 drscratch 的检测成功率,codinggo-Scratch3.0 的检测成功率高于 drscratch 的成功率,并且 20 种特征检测多于 drscratch 的 10 种检测规则.

4.5　算法输出

随机选取 Scratch3.0 实例来验证 Scratch3.0 的作品检测功能是否正常,能否按照预期的检测规则输出正确的结果.

如图 7(a)所示,Scratch3.0 实例作品中存在一个角色和一个有效脚本,其中存在一个名为 my variable 的滥用的全局变量,存在一个名为 Sprite1 的无意义的角色命名,存在一个没有起始块的死代码块以及 2 个使用了字符串信息的脚本.

最终的输出结果如图 7(b)所示,与预期的检测结果一致,功能正常.



(a) Scratch3.0实例详情　　(b) Scratch3.0实例检测结果

图 7　Scratch3.0 项目实例及检测结果

5　结束语

为了解决已有的特征提取和检测系统不支持 Scratch3.0 版本的问题,提出了一种基于链表数据结构和 ANTLR 的面向 Scratch3.0 的特征提取和检测系统. 此外,对其与 Scratch2.0 版本的检测方法进行了性能、稳定性、功能上的对比实验,结果表明,其检测性能和检测稳定性比 Scratch2.0 均有所提升,达到了预期的功能目标. 同时,由于检测采用解耦的算法设计思想使算法有良好的扩展性和可读性,所以 Scratch3.0 检测算法的改进和提升空间更大.

参考文献:

[1] Fowler M. Refactoring: improving the design of existing code[M]. [S.l.]: Addison-Wesley Professional, 2018.

[2] Meerbaum-Salant O, Armoni M, Ben-Ari M. Habits of programming in scratch[C]// Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education- ITiCSE'11. New York: ACM Press, 2011: 168-172.

[3] Wolz U, Hallberg C, Taylor B. Scrape: a tool for visualizing the code of Scratch programs[C]// Poster Presented at the 42nd ACM Technical Symposium on Computer Science Education. Dallas: [s.n.], 2011: 100-110.

[4] Boe B, Hill C, Len M, et al. Hairball: lint-inspired static analysis of scratch projects[C]// Proceeding of the 44th ACM Technical Symposium on Computer Science Education. [S.l.]: ACM, 2013: 215-220.

[5] Techapalokul P, Tilevich E. Quality hound: an online code smell analyzer for scratch programs[C]// 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). New York: IEEE Press, 2017: 337-338.

[6] Hermans F, Stolee K T, Hoepelman D. Smells in block-based programming languages[C]//2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). New York: IEEE Press, 2016: 68-72.

[7] Techapalokul P, Tilevich E. Understanding recurring quality problems and their impact on code sharing in block-based software[C]// 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). New York: IEEE Press, 2017: 43-51.

[8] Parr T. The definitive ANTLR 4 reference[M]. [S.l.]: Pragmatic Bookshelf, 2013.

[9] Wang Xiaolei, Huang Xinmin, Yan Chang, et al. Analysis of scratch project with process data[C]//2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW). New York: IEEE Press, 2018: 1-2.

[10] 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). New York: IEEE Press, 2018: 950-955.