基于 Scratch 作品相似度的检测研究

张　锦[1,2],胡子达[1],陆玟冰[1],杨定康[1],李　强[1],罗元盛[2]
(1. 湖南师范大学 信息科学与工程学院,湖南 长沙 410006;
2. 长沙理工大学 计算机与通信工程学院,湖南 长沙 410006)

摘　要:Scratch 作为图形化编程中的热门课程吸引了广大中小学生,而对于学生所做的作品与标准作品之间差异性的评定通常是靠教师通过人工对比检查,对于教师不仅工作量大且耗费巨大精力,因此对于 Scratch 作品相似性的识别就可以辅助教师快速检测学生作品,从而提高教学效率。针对该问题,提出 Siamese-BERT 模型对两个 Scratch 作品之间的相似度进行检测。首先,对 Scratch 源文件进行解析提取原始积木块序列,根据积木块逻辑特征提出一种积木块重构算法,将原始积木块序列排序成 Token 序列,将 Token 序列作为 CBOW(Continuous Bag of Words)模型的输入文本进行预训练,从而得到 Scratch 的词向量模型;再使用 Siamese 神经网络框架结合 BERT(Bidirectional Encoder Representation from Transformers)模型组合训练,最终输入到余弦相似度函数进行相似度计算。数据集来自于长沙市 Scratch 培训机构的培训作品和学生的练习作品,在该数据集上,Siamese-BERT 模型准确度能达到 0.82,对比其它的文本相似度模型,Siamese-BERT 模型在 Scratch 作品相似度检测上更加准确。

关键词:Scratch 图形化编程;Siamese-BERT 模型;连续词袋模型;Siamese 神经网络;BERT 模型;余弦相似度
中图分类号:TP399　　**文献标识码**:A　　**文章编号**:1673-629X(2023)10-0143-07
doi:10.3969/j.issn.1673-629X.2023.10.022

Research on Similarity Detection of Project Based on Scratch

ZHANG Jin[1,2],HU Zi-da[1],LU Wen-bing[1],YANG Ding-kang[1],LI Qiang[1],LUO Yuan-sheng[2]
(1. School of Information Science and Engineering,Hunan Normal University,Changsha 410006,China;
2. School of Computer and Communication Engineering,Changsha University of Science & Technology,Changsha 410006,China)

Abstract:As a popular course in graphic programming,Scratch has attracted a large number of primary and secondary school students,and the evaluation of the difference between the projects made by students and the standard projects is usually made by the teacher through manual comparison and inspection,which is not only a heavy workload for teachers,but also a huge energy consumption. Therefore,the recognition of similarities in Scratch projects can assist teachers to quickly detect students´ projects,thus improving teaching efficiency. To solve this problem,the Siamese-BERT model is proposed to detect the similarity between two Scratch projects. Firstly,the Scratch source file is analyzed to extract the sequence of original building blocks,and a building block reconstruction algorithm is proposed according to the logical characteristics of building blocks to sort the sequence of original building blocks into Token sequence. Token sequence is used as input text of CBOW (Continuous Bag of Words) model for pre-training,so as to obtain Scratch word vector model. Then,Siamese neural network framework is used for combined training with BERT(Bidirectional Encoder Representation from Transformers) model,and finally input into cosine similarity function for similarity calculation. The data set comes from the training projects of Scratch training institution in Changsha City and the practice projects of students. On this data set,the accuracy of Siamese-BERT model can reach 0.82. Compared with other text similarity models,the Siamese-BERT model is more accurate in the similarity detection of Scratch projects.

Key words:Scratch graphical programming;Siamese-BERT;CBOW;Siamese network;BERT;cosine similarity

收稿日期:2022-11-24　　**修回日期**:2023-03-25
基金项目:国防科技重点实验室基金项目(2021-KJWPDL-17);国防科工局国防基础科研计划(WDZC20205500119);湖南省自然科学基金(2021JJ30456)
作者简介:张　锦(1979-),男,教授,博导,博士,CCF 长沙分部秘书长(06153D),研究方向为网络与信息安全、人工智能安全;通讯作者:胡子达(1999-),男,硕士研究生,CCF 会员(F6522G),研究方向为自然语言处理。

0 引 言

Scratch 是麻省理工学院针对儿童设计开发的程序编写语言与环境,目的是让儿童在创作体验中学习编程、表达自己的想法[1]。Scratch 改变了使用复杂代码进行程序设计的现状,为学习者提供门槛低且更容易操作练习的编程环境,主要是通过拖拽积木和拼接积木的方式进行学习,让学生像搭积木一样进行编程,且 Scratch 提供生动形象的各式各样的角色人物和背景画面。这样不仅吸引了学生的注意力,同时还能激发了学生对编程的兴趣,对于锻炼学生的动手能力、逻辑思维能力和计算机思维都有一定的帮助。

教师可以把 Scratch 引入教学当中,对教学内容进行调整,转变学生对编程的认识[2]。学生根据教师布置的课堂任务制作作品,作品完成之后通常是由老师进行检查和评测,出现了各式各样的问题,如:作品不符合任务要求、代码逻辑出现错误等。然后教师再根据出现的问题给学生提出修改意见,但这种人工检查和评判的方法只适用于学生数量少的情况,一旦学生数量多,老师的精力和教学效率也会随之下降。

对于 Scratch 的研究工作,目前国外学术界主要是对代码编程能力和计算机思维的评估[3]。文献[4]提出了一种新的静态分析工具 Hairball,同时还提供了一个可扩展的框架来自动分析 Scratch 程序,主要是对 Scratch 作品的四个方面能力进行分析:初始化、广播和接收、语音和声音同步以及动画。基于 Hairball 工具,文献[5]提出对 Scratch 作品进行计算思维(CT)打分,并开发了一个开源的网络工具 Dr. Scratch 将 CT 细化为七个维度,每个维度又分为三个级别,它主要是分析 Scratch 作品以检查其潜在的错误和不良的编程习惯,同时针对于 2019 年新推出的 Scratch3.0 版本也进行了相应的更新,但是 Dr. Scratch 在进行作品评估打分时容易出现错误。主要是由于相较于 C、Python 等文本型编程语言,Scratch 主要以积木块的形成进行编程,其代码是存储在 JSON 文件里,且代码逻辑结构是无序混乱的。这就要求解析算法能够提取作品的 JSON 文档,并对其存储的无序代码进行拆解并重组,然后再在此基础上抽取特征。

对于 Scratch 作品之间的相似度评测,目前国内外的研究甚少,这就需要从传统的文本相似度研究中借鉴参考自然语言处理(NLP)等技术。文献[6-9]提到很多传统的文本相似度检测方法,文献[10-16]提到多种基于深度学习的模型,而对于 Scratch 作品相似性的检测目前还是缺乏方法和模型。因为国内外学术界对于 Scratch 的研究尚处于教学阶段,而对于作品的检测主要是代码编程能力和计算机思维的评估[16],尚未有比较完善的作品之间相似性模型。

该文主要贡献如下:

(1)针对 Scratch3.0 版本的作品进行特征提取,从 Sb3 格式的作品中提取 JSON 文件,并从中提取角色及其积木块,构成 Token 序列,并提出积木块排序算法,将 Token 序列根据作品特征进行重构排序。

(2)针对 Scratch 作品相似度检测,提出基于 Siamese 网络框架上搭建 BERT 模型的 Siamese-BERT 模型,实验结果显示,与 Siamese-LSTM 模型相比,该模型准确度更高。

1 相关工作

Chang Z 等人提出了一种新的 Scratch 程序分析工具,它是基于 ANTLR 设计实现的并定义了 200 多个词法和语法解析器规则,主要是为了解决 Dr. Scratch 存在的一些缺陷(例如高故障率和低效率)。但是该工具只是针对于 Scratch2.0 版本,而对于 2019 年推出的 Scratch3.0 并没有相应的更新版本[17]。

文献[18]采用抽象语法树的思想方法解析 Scratch 源代码保存的 JSON 文件,提出了一种在线程序分析工具 Quality Hound,它将 Scratch 作品作为输入,该工具基于 JastAdd 框架和 Java 语言[19]实现,并向用户呈现检测到的质量问题的可视化表示。

在传统的文本相似度检测方法中,基于字符串匹配和词频特征是比较常见的方式,字符串匹配是分析字符之间的重叠,最长公共子序列法[6]、编辑距离[7]、Jaccard 系数[8]、贪婪式字符串匹配算法等是较常用的方法;词频特征是词语在文档中出现的频率,最常见的是 TF-IDF 方法。Tai 等人采用深度学习的方法提出一种名为 Tree-Lstm[9] 树形结构模型,该模型可以从结构化特征中提取到语义关联信息,从而计算句子的相似度。Tom Kenter[10] 在 2016 年提出的 Siamese CBOW 模型是一种神经网络架构,可以有效地学习为生成句子表示而优化的词嵌入,两个句子向量之间的余弦相似度作为最终的语义相似度得分,能够很好地区别语义相似的句子,该模型优于 word2vec 和 GloVe。文献[11]提出 Siamese-LSTM 网络结构,对于可变长度序列可以通过孪生网络实现单边网络的权重共享实现,从而得到句子之间的语义相似性。

近些年来,随着计算机硬件设备和深度学习等相关技术的快速发展,对于代码相似度方面的研究也开始在深度学习方向上展开,并且取得了不错的成绩[12-14]。例如抽象语法树,文献[15]提出一种基于 AST 的神经网络(ASTNN)模型,把大型的 AST 进行分割处理,变成一个个小语句树序列,并获取该树的每一层句法知识,从而将每个语句树映射成相应的向量。文献[16]提出 Gemini 算法,该模型的输入是两个二

进制函数对,输出是该函数对的相似度,该模型采用 Siamese 神经网络框架并使用梯度下降算法,在速度和精确度方面均优于传统方法。

2　Siamese-BERT 模型

2.1　总体框架

该文设计并实现了 Siamese-BERT 相似度模型,整个模型的总体框架如图 1 所示。该模型能够对两个 Scratch 作品进行相似度检测,具体实现步骤如下:

(1)数据处理。

Scratch 作品的重要信息主要保存在 JSON 文件中,因此首先要获取 JSON 文件。Scratch 作品的 JSON 文件是以对象格式进行保存,如:{"targets":[舞台 1,角色 1]},舞台和角色中又包含多个积木块,所以需要对 JSON 文件进行特征提取,提取出来的特征是一个包含角色和舞台及其积木块的字典,如{舞台 1:[积木块 1,……,积木块 n],角色 1:[积木块 1,……,积木块 n]}。此时的字典中的积木和角色都是无序的,还需要对其进行排序处理,得到最终的 Token 序列。

(2)CBOW 词嵌入模型。

Word2vec 是轻量级的神经网络,其模型包括输入层、隐藏层和输出层共三层,模型种类包括 CBOW 和 Skip-gram 两种。CBOW 模型是通过上下文预测中心词($w(t)$),Skip-gram 则是根据中心词($w(t)$)预测上下文。

(3)BERT 模型。

构建 BERT 模型进行训练,该模型能提取特征词在句子中的关系特征,即在多个不同层次提取关系特征,从而更好地反映作品积木块之间的逻辑关系。整个模型由输入层、编码层和输出层构成,其中输入层是{ $e_1,e_2,\cdots,e_n$ }向量,编码层由多个 Transformer 组成,最终输出向量为{ $T_1,T_2,\cdots,T_n$ }。

(4)余弦相似度算法。

计算两个作品的相似度是用向量空间中两个向量之间夹角的余弦值来表示。余弦值越接近 1,表示越相似,反之越接近 0,则越不相似。

$$\cos\theta=\frac{a\cdot b}{|a|\cdot|b|}=\frac{\sum_{i=1}^{n}(x_i\times y_i)}{\sqrt{\sum_{i=1}^{n}X_i^2\times\sum_{i=1}^{n}Y_i^2}} \tag{1}$$

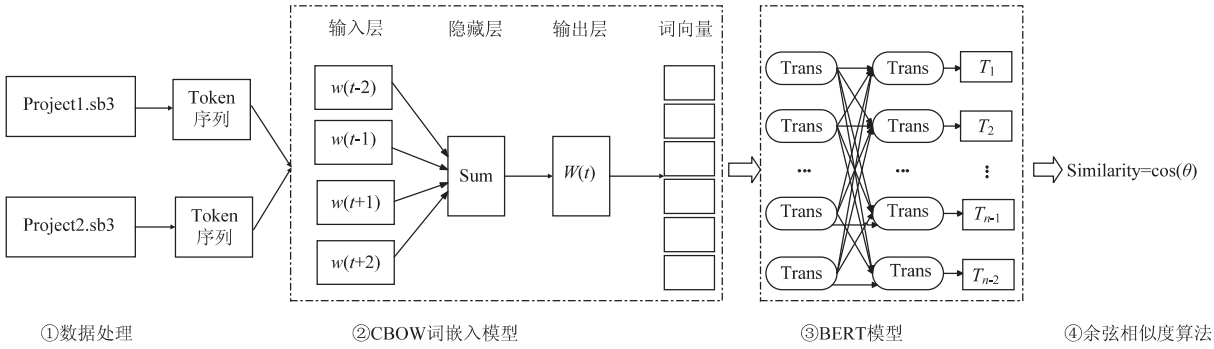


图 1　Siamese-BERT 模型总体框架

2.2　数据处理

由于角色和舞台提取出来的积木块(block)顺序是混乱的,所以需要根据 block 的构造特点进行排序,block 之间根据其父块(parent)、子块(next),以及是否是顶层块(top)等字段联系起来。如图 2(a)所示,event_whenflagclicked 积木块的 next 是 control_repeat_until,且它还是顶层块,因为是该脚本块的起始积木块。

积木块中有种特殊的积木块,即嵌入积木,如:循环积木块、判断积木块,该种积木块的特点是其 next 既不是其条件体的第一个积木块(condition),也不是其语句体内的第一个积木块(substack),而是紧随其后的第一个积木块。

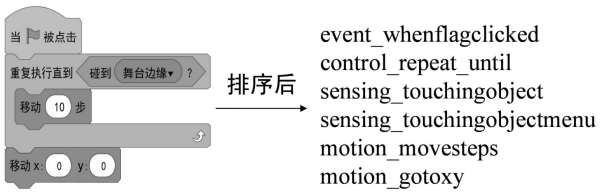
算法 1:积木块排序

输入:无序的积木块列表(blocks)

输出:有序的积木块列表(Order blocks)

```
1. for block in blocks do
2. if block 是顶层积木块 then
3. Order_blocks. append(block) / * 将顶层积木块添加到有序积木块列表中 * /
4. top=block//将顶层快视为下一个积木块的头部
5. for block in blocks do
6. if block. parent= =top and block 是嵌入块 then
7.        将 block 的条件块以及语句块从 blocks 取出放到 Q_block 列表中
8. Order_blocks. append(Q_block)
9. end if
10. elif block. parent= =top then
11. Order_blocks. append(block)
12. top=block
13. end if
14. end for
15. end if
16. end for
17. return Order_blocks
```

由于顶层积木块通常是由事件积木充当,而嵌入积木块都是属于控制积木或运算积木。如算法 1,检索无序积木块列表时首先判断是否为顶层块,将顶层块找到放到有序积木块列表中,再在剩余的积木块中去寻找其子块,在排序过程中需要先判断是否为嵌入积木块,根据其特点将其与 condition 和 substack 进行规整,使其成为一个整体,然后再与普通积木块根据 parent 和 next 特征进行排序,最终得到如图 2(b)所示的结果。



(a) (b)

图 2 Scratch 积木块重构过程

2.3 词嵌入模型

以 CBOW 模型为例,其中输入层是一个形状为 $C \times V$ 的 one-hot 张量, C 表示上下文中词的个数, V 表示词汇表大小。隐藏层是一个形状为 $V \times N$ 的参数张量,一般称为 word-embedding, N 表示每个词的词向量的维度。输出层创建另一个形状为 $N \times V$ 的参数张量,将隐藏层得到的 $1 \times N$ 的向量与该 $N \times V$ 的向量进行矩阵相乘,得到了一个形状为 $1 \times V$ 的向量。将 softmax 作用于输出向量,即得到中心词的预测概率。最终得到一个 $V \times N$ 词向量矩阵,每个词通过 one-hot 查询词向量矩阵就能得到其对应的词向量,CBOW 模型的结构如图 3 所示。而 Skip-gram 模型则与 CBOW 模型相反,它是通过中心词去预测上下文,其模型结构如图 4 所示。

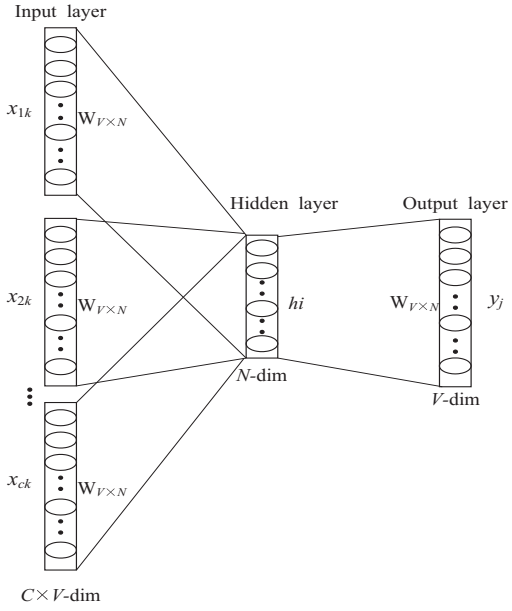


图 3 CBOW 模型结构

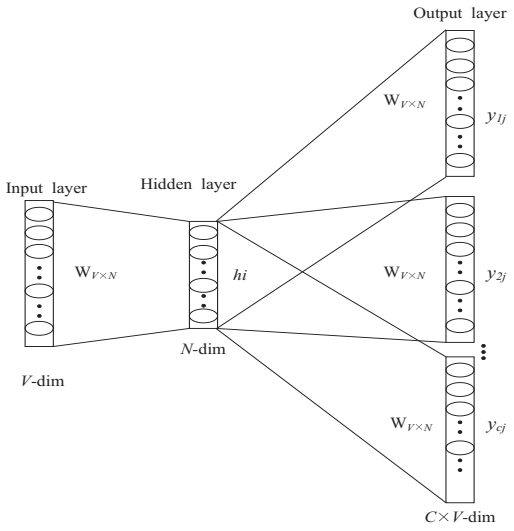


图 4 Skip-gram 模型结构

相对于 Skip-gram 模型,CBOW 模型更适合小型语料库,因此该文采用的是 CBOW 模型,单词上下文大小 C 采用默认参数 5,词向量维度设置为 128。通过实验表明,所训练的 CBOW 模型不仅能够识别 Scratch 八大类别积木块,同时还能够将具有相似功能的同类别积木块转换成接近的向量,从而帮助相似性度模型更好地检测 Scratch 作品之间的相似性。

2.4 BERT 模型

BERT 是一种预训练语义表征模型,该模型通过融合文本表征能力强大的迁移学习(Transformer)模型实现,预训练能获得更好的向量表达。向量的表示可以随机初始化,也可以采用预训练的向量,该文采用 CBOW 对文本进行预训练,从而得到词向量作为 BERT 模型的输入值;BERT 网络结构是由多层 Transformer Encoder 叠加;输出是文本中各个字融合了全文语义信息后的向量表示,如图 5 所示。

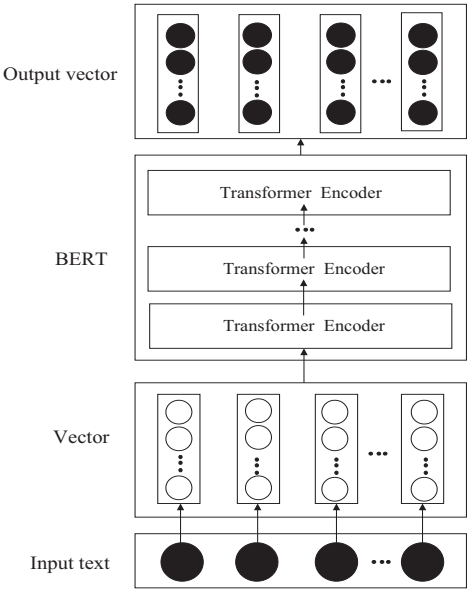


图 5 BERT 模型

2.5　孪生神经网络

孪生神经网络(Siamese)有两个输入(Input1 and Input2),将两个输入传入两个神经网络(Network)中,这两个网络共享权重并将输入转换为向量,然后通过 Distance 计算出两个输入的相似度,如图 6 所示。

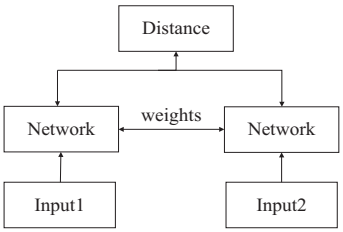


图 6　孪生神经网络

该文所提出的 Siamese-BERT 模型采用 Siamese 神经网络框架,其中神经网络层使用 CBOW+BERT 模型组合,相似度距离采用余弦相似度进行计算,Siamese-BERT 模型框架如图 7 所示。

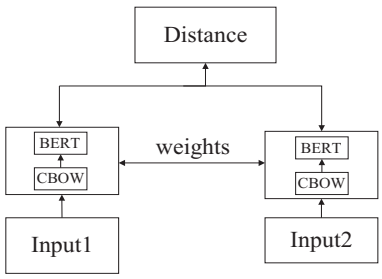


图 7　Siamese-BERT 模型框架

3　实验结果及分析

3.1　Scratch 作品组成

Scratch 作品是以后缀为. sb3 的格式进行保存,该文件只能由 Scratch 客户端打开,通过将其后缀改为. zip格式,再进行解压就可以得到一个作品源文件。如图 8 所示,该文件包含图像、音频等多媒体资源以及一个源代码为 project. JSON 格式文件。JSON 文件中主要包含角色和舞台两大部分,而其中角色又是最为重要的部分,选定好角色就需要对角色设计积木块代码来实现其指定的功能;舞台则是角色显示区,在舞台区中可以通过选择不同的背景来为角色更换场景。

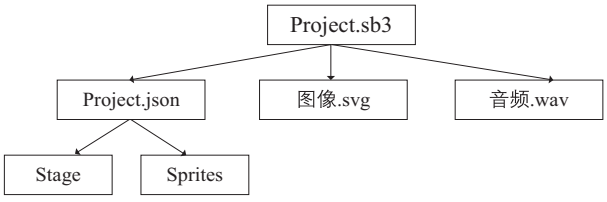


图 8　Scratch 作品组成部分

Scratch 作品主要是通过为角色和舞台搭建积木块进行创作,每一个角色和舞台都包含一个或多个脚本块,脚本块相当于实现角色的功能,可以理解成编程语言中函数的概念,而一个脚本块又包含多个连续的积木块。Scratch 积木块总共有 8 大类以及自定义积木类别,如表 1 所示。

表 1　积木块类别

积木块名称	作用
运动	使角色动起来,如移动、转向、位置坐标等
外观	设计作品的外观,如背景、颜色、角色造型等
声音	为作品添加声音,如背景声音、音乐等
事件	作品中发生的时间,如触发积木块,更换背景等
控制	控制程序流程,如等待几秒、重复执行等
侦测	判断事件,如侦测鼠标是否碰到角色等
运算	与数学有关,如加减乘除四则运算等
变量	记录发生变化的事物,如移动的步数
自定义积木	根据作品的需求自定义设计积木

3.2　数据集构造

3.2.1　数据来源

目前国内外对于 Scratch 作品数据集的研究工作很少,因此没有一个合适的公开数据集。文中的数据集来自长沙市 Scratch 图形化编程培训机构所提供的作品。共包含四个 Scratch 等级(分别对应四个学期),每个等级中包含 16 个课程,其中每个课程含有一个教师所制作的标准作品以及 5 个学生作品,如表 2 所示。

表 2　数据来源

Scratch 等级	课程个数	学生作品合计
等级一	16	80
等级二	16	80
等级三	16	80
等级四	16	80
总计	64	320

3.2.2　数据集构建

除了每个课程标准作品和该课程对应的 5 个学生作品进行相似度计算之外,该文还将每个学生作品与其他的作品进行相似度计算,从而扩充数据集。

采用 TF-IDF 算法和余弦相似度对 Scratch 作品进行相似度计算:

(1)将作品中的连续积木块分为独立的积木块合集。

(2)求出两个积木块集合的并集。

(3)计算各个积木块集的词频并把词频向量化。

(4)计算两个向量的余弦相似度,值越大就表示越相似,即可求出两个作品的相似度。

在构建两个作品的相似度标签(Label)时,由于 Label 是 1 和 0,该文在得到两个作品的相似度(sim)之后,采用以下方法对其进行 Label 标注:

$$\text{Label} = \begin{cases} 1, \text{sim} \geqslant 0.6 \\ 0, \text{sim} < 0.6 \end{cases} \quad (2)$$

除了同一个课程里的学生作品相似度高一点之外,大多的学生作品之间的相似度都不是太高,因此标签为 0。为了保证数据的准确性,该文去除了大部分标签为 0 和部分相似度为 1 的数据,最终保证每个作品与之标签为 1 和为 0 的数据数接近。最终数据集由 1 520 条训练集和 150 条验证集构成。

3.3 相似度算法对比实验

该文选用了几个常用的相似度计算方法进行对比实验,其中包括欧氏距离、曼哈顿距离、余弦相似度。欧氏距离是通常采用的距离定义,指在 m 维空间中两个点之间的真实距离;曼哈顿距离又被称作计程车几何,也就是欧几里得空间的固定直角坐标系上两点所形成的线段对轴产生的投影的距离总和。该文采用精准度、召回率作为模型的衡量标准,计算公式如下:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (3)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4)$$

其中,TP 为真正样本,TN 为真负样本,FP 为假正样本,FN 为假正样本。

如表 3 所示,在准确率和召回率两个指标上,余弦相似度最高,曼哈顿距离其次,而欧氏距离最低。因此,该文最终采用了余弦相似度作为模型的相似度计算法。

表 3 不同的相似度算法实验对比

相似度算法	Accuracy	Recall
欧氏距离	0.76	0.72
曼哈顿距离	0.79	0.79
余弦相似度	0.82	0.80

3.4 CBOW 词嵌入模型实验结果

采用 Word2vec 的两种模型 CBOW 和 Skip-gram 进行训练对比,在积木块的相似判断上,CBOW 表现效果明显优于 Skip-gram。举例如下:以事件积木块“event _ whenflagclicked”、运动积木块“motion _ turnleft”、控制积木块“control_repeat”三个积木块为例,分别返回相应词向量得到的最接近的 3 个积木块,实验结果如表 4 所示。

表 4 CBOW 和 Skip-gram 结果对比

积木块	CBOW 模型		Skip-gram 模型	
	相似积木块	相似度	相似积木块	相似度
event_whenflagclicked	event_whenkeypressed	0.52	operator_random	0.53
	event_whenthisspriteclicked	0.52	looks_setsizeto	0.52
	event_broadcast	0.47	event_whenkeypressed	0.52
motion_turnleft	motion_movesteps	0.53	event_whenkeypressed	0.58
	motion_turnright	0.52	pen_setPenColorToColor	0.53
	motion_goto	0.50	looks_sayforsecs	0.53
control_repeat	control_forever	0.52	procedures_definition	0.59
	control_if	0.36	looks_changeeffectby	0.55
	control_if_else	0.33	looks_setsizeto	0.52

从 CBOW 和 Skip-gram 的结果看来,与积木块相似度最高的积木块都属于同一种模块,通过表 4 可以看出同类别的积木块在功能上是最相近的,从这一点可以看出 CBOW 比 Skip-gram 在识别同类型的积木块上效果更好。

3.5 相似度模型实验结果

该文采用基于孪生神经网络结构的 LSTM 模型[11]以及训练好的词向量模型与该文提出的 Siamese-BERT 模型进行对比。采用了精准度、召回率作为模型的衡量标准,实验结果如表 5 所示。

表 5 Siamese-LSTM 和 Siamese-BERT 实验对比

模型	Accuracy	Recall
Siamese-LSTM	0.73	0.75
Siamese-BERT	0.82	0.80

从实验结果中分析可得,Siamese-BERT 无论是在准确度上还是召回率上都优于 Siamese-LSTM,这也证明提出的 Siamese-BERT 在检测 Scratch 作品相似度上有着不错的表现。

4 结束语

该文提出的 Scratch 解析、重构是充分考虑了 Scratch 作品的特征以及源代码 JSON 文件等方面,这对 Scratch 的解析和评测工作具有一定的参考作用。同时提出的 Siamese-BERT 模型在检测 Scratch 作品相似性上具有良好的表现,这也为 Scratch 图形编程教师在教学上提供了一定的辅助作用。但仍然存在一些不足,比如准确度上还有待提升;角色和背景之间的相似度尚未进行检测,未来也将朝着这些方向继续研究。

参考文献:

[1] 刘　波. 小学信息技术课堂中 Scratch 的应用探索[J]. 信息记录材料,2019,20(12):108-109.

[2] 朱丽彬,金炳尧. Scratch 程序设计课教学实践研究——基于体验学习圈的视角[J]. 现代教育技术,2013,23(7):30-33.

[3] FAGERLUND J, HÄKKINEN P, VESISENAHO M, et al. Computational thinking in programming with scratch in primary schools: a systematic review[J]. Computer Applications in Engineering Education,2021,29(1):12-28.

[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. Denver: ACM,2013:215-220.

[5] MORENO-LEÓN J, ROBLES G, ROMÁN-GONZÁLEZ M. Dr. scratch: automatic analysis of scratch projects to assess and foster computational thinking[J]. RED. Revista de Educación a Distancia,2015(46):1-23.

[6] IRVING R W, FRASER C B. Two algorithms for the longest common subsequence of three (or more) strings[C]//Combinatorial pattern matching. Berlin: Springer,1992:214-229.

[7] DAMERAU F J. A technique for computer detection and correction of spelling errors[J]. Communications of the ACM, 1964,7(3):171-176.

[8] JACCARD P. The distribution of the flora in the alpine zone. 1[J]. New Phytologist,1912,11(2):37-50.

[9] TAI K S, SOCHER R, MANNING C D. Improved semantic representations from tree-structured long short-term memory networks[J]. arXiv:1503.00075,2015.

[10] KENTER T, BORISOV A, DE RIJKE M. Siamese cbow: optimizing word embeddings for sentence representations[J]. arXiv:1606.04640,2016.

[11] MUELLER J, THYAGARAJAN A. Siamese recurrent architectures for learning sentence similarity[C]//Proceedings of the AAAI conference on artificial intelligence. Phoenix: AAAI,2016.

[12] YU H, LAM W, CHEN L, et al. Neural detection of semantic code clones via tree-based convolution[C]//2019 IEEE/ACM 27th international conference on program comprehension (ICPC). Montreal: IEEE,2019:70-80.

[13] VAN BLADEL B, DEMEYER S. A novel approach for detecting type-iv clones in test code[C]//2019 IEEE 13th international workshop on software clones (IWSC). Hangzhou: IEEE,2019:8-12.

[14] RAGKHITWETSAGUL C, KRINKE J, CLARK D. A comparison of code similarity analysers[J]. Empirical Software Engineering,2018,23(4):2464-2519.

[15] ZHANG J, WANG X, ZHANG H, et al. A novel neural source code representation based on abstract syntax tree [C]//2019 IEEE/ACM 41st international conference on software engineering (ICSE). Montréal: IEEE, 2019: 783-794.

[16] XU X, LIU C, FENG Q, et al. Neural network-based graph embedding for cross-platform binary code similarity detection[C]//Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. Dallas: ACM,2017:363-376.

[17] CHANG Z, SUN Y, WU T Y, 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, 2018: 950-955.

[18] 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). Raleigh: IEEE,2017:337-338.

[19] HEDIN G, MAGNUSSON E. JastAdd—an aspect-oriented compiler construction system[J]. Science of Computer Programming,2003,47(1):37-58.