DOI：10.7652/xjtuxb201610010

面向大规模在线开放课程的编程题多特征综合自动评分方法

刘月霞，牛志尧，吴宁
（西安交通大学电子与信息工程学院，710049，西安）

摘要：针对大规模在线开放课程环境下 C/C++语言学习者人数众多、自动评阅准确率低的问题，提出一种基于多特征综合分析的编程题自动评分方法。通过对源程序编译预处理剔除提示性信息，用词法分析和抽象语法树(AST)分别抽取学生程序和标准模板程序的多种特征并计算特征相似度，再根据程序编译是否通过，采用不同策略综合分析多种特征相似度进行自动评分。特征相似度包括多项测试用例运行结果的相似度、AST 抽取的各项特征的相似度和源程序代码相似度。如果学生程序编译失败，在计算 AST 特征相似度的同时需进行源程序代码相似度分析。实验结果表明：相对于仅基于测试用例运行结果的动态测试方法和传统静态分析方法，所提方法的平均准确率分别提高了 18.48%和 14.17%，评价结果与人工评分高度相关且无需借助人工辅助分析。该方法适用于大规模在线开放课程教学。

关键词：大规模在线开放课程；自动评阅；多特征分析；抽象语法树；相似度计算
中图分类号：TP311 **文献标志码**：A **文章编号**：0253-987X(2016)10-0064-07

An Automatic Scoring Method of Student Programs Using Multi-Feature Analysis for Massive Open Online Courses

LIU Yuexia, NIU Zhiyao, WU Ning
(School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: A new automatic scoring method based on multi-feature analysis is proposed to focus the problem that there are a great number of C/C++ programming learners on massive open online courses environment, while the existing automatic scoring techniques possess low accuracy. The prompt information in a submitted program is eliminated with a preprocessing compiler. The lexical analysis and abstract syntax tree (AST) methods are used to extract the features of the submitted program and the standard template one, respectively. Then similarities of these features are calculated. According to whether or not the program is compiled successfully, two different strategies are applied to comprehensively analyze the multi-feature similarities and the program is automatically evaluated finally. The multi-feature similarities include the running result similarity of test cases, the AST feature similarity, and the source code similarity. If the program fails to be compiled, both the source code similarity and the AST features similarity need to be analyzed. Experimental results and comparisons with the dynamic test method and the static analysis method show that the average accuracy of the proposed method

收稿日期：2016-05-19。 作者简介：刘月霞(1991—)，女，硕士生；吴宁(通信作者)，女，教授。 基金项目：陕西省科技研究发展计划资助项目(2013K06-05)。
网络出版时间：2016-09-02 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20160902.1630.004.html

increases by 18.38 % and 14.17 %, respectively. The automatically generated scores are highly correlated with manually determined scores and there is no manual assistant to ensure the accuracy of the scoring results. The proposed method can be applied in massive open online courses.

Keywords: massive open online courses; automatic evaluation; multi-feature analysis; abstract syntax tree; similarity computation

自 2012 年起，大规模在线开放课程(massive open online courses，MOOC)开始在全球兴起。对计算机程序设计类 MOOC，编程练习是重要的环节。由于 MOOC 环境下的学习者人数众多，依靠教师人工评阅编程作业不现实。目前，MOOC 平台对编程题的评判大多采用学生互评方式，部分平台通过将学生作业的运行结果与给定答案进行比对的方式评分，以上评分方式并不适合对学生作业和考试程序的评分，难以满足教学需求[1-2]。

学生程序一般具有规模小(代码行数通常在 100 行以内)、在正确或基本正确情况下其代码规模与标准模板程序相差不大、同一道题目可能会有多种解答、易出现语法错误且错误类型较多等特点，这些特点决定了编程题自动评分若要满足教学需求必须解决以下几个问题：①如何证明程序语义、知识的正确性；②如何衡量学生完成编程任务的正确程度；③如何处理有语法错误的程序；④采用什么样的评分模型才能获得与人工阅卷相关性高的评分结果[3]。

近年来，已有多位学者针对学生编程作业特点开展自动评分方法研究。目前的自动评分方法主要分为两大类：基于测试用例的动态测试和基于程序中间表示形式的静态分析。文献[4-6]研究了动态测试方法，通过随机生成测试用例，然后比对学生程序与标准模板程序的运行结果进行评分。这是评判程序正确性最直接的方法，也是目前使用较多的编程题自动评分方法，例如全国计算机等级考试中的编程题评阅系统等。这种评测方法的前提是程序必须能够运行，其不足是：①当学生程序输出与标准模板程序输出格式不完全相符时会影响评分精度；②如果学生通过数学计算或其他方式直接输出结果会获得满分，这显然与实际情况不符；③若程序编译失败，只会给出零分。这些不足使其难以满足程序设计课程教学的实际需求。国内外众多学者也研究了编程题的静态分析方法。静态分析通常是将源程序转换为抽象语法树或系统依赖图等中间表示形式，从中间形式抽取出程序的各项特征，再根据特征信息的匹配度给出分值[7]。文献[8]通过计算控制流图节点的相似度进行评分；文献[9]通过匹配系统依赖图抽取的程序特征进行评分；文献[10-11]通过分析抽象语法树进行评分。静态分析方法能够在一定程度上反映学生程序和标准模板程序的相似程度，较适合对学生程序的自动评分，但完全依赖中间表示形式也存在不足，主要是当程序因语法错误导致编译失败时，所生成的中间表示形式往往存在偏差，使提取的程序特征信息缺乏足够的可信度，从而影响评分精度。

针对上述存在的问题，本文通过对人工评阅的思维过程进行分析，对 C/C++编程题自动评分中的难点问题进行深入研究，融合动态测试与静态分析方法的优点，提出了一种用于支撑 MOOC 教学的编程题多特征综合分析自动评分方法。

1　编程题多特征综合评分方法

1.1　多特征综合评分模型

人工评阅编程题的一般过程如下。①首先编译学生程序，若编译通过，直接观察运行结果。如果结果正确，再观察程序规模以判断是否是直接输出结果；如果运行结果不正确，会考察其编程思想、关键知识点(例如程序控制结构、表达式、类定义等)等是否符合题目要求，根据符合度进行评分。②若学生程序因存在语法错误导致编译失败，则在扣除一定语法分之后，再考察其编程思想、知识点等进行综合评分。

基于人工评阅思想的自动评分技术中，通过输入测试用例、判断运行结果的方法称为动态测试；通过观察程序编程思想、知识点、程序规模等特征的方法属于静态分析。人工评阅模式下的静态分析依据人的判断，而自动评分系统中的静态分析则需要借助程序的中间表示形式或源程序代码。

作为程序的一种中间表示形式，抽象语法树(abstract syntax tree，AST)是源程序代码的抽象语法结构的树状表现形式，树上的每个节点都表示源代码中的一种结构。之所以称为“抽象”，是因为这

里的语法并不表示出真实语法中出现的每个细节，而只反映其特征。通过将学生程序和标准模板程序转换为以 AST 表示的中间形式，从中抽取各种特征并对比特征相似度，就可以确定学生程序的正确程度。

由于编译失败时生成的 AST 文本与程序实际语义存在较大不一致性，甚至无法生成 AST，所以在此情况下，仅依赖于对 AST 所抽取特征的分析会影响评分精度。为此，需进一步分析程序源代码，计算学生程序与标准模板程序的代码相似度，结合对 AST 所抽取特征的分析结果进行综合评分。

基于上述分析，本文提出一种多特征综合评分模型，如图 1 所示。

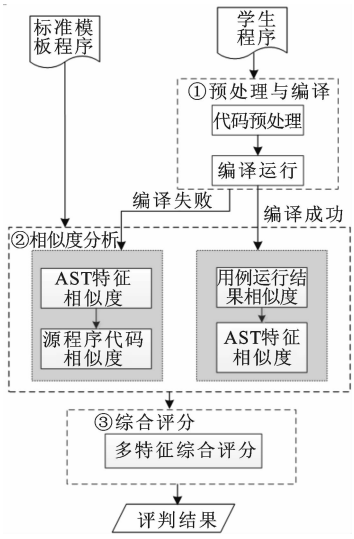


图 1 编程题多特征综合评分模型

图中用虚线框及相应数字编号表示评分过程的 3 个主要阶段。①预处理与编译。首先对学生程序代码进行预处理，去除提示性语句等影响运行和评分准确性的因素。然后调用编译器进行编译，根据编译是否通过决定后续的特征抽取和评分策略。②相似度分析。包括对多项测试用例运行结果的相似度计算(简称用例运行结果)、对从 AST 抽取的各项特征(简称 AST 特征)的相似度计算以及源程序代码的相似度计算。如果学生程序编译通过，则匹配学生程序和标准模板程序用例运行结果相似度，若完全匹配，再分析程序规模，否则进一步分析 AST 特征的相似度；如果学生程序编译失败，在计算 AST 特征相似度的同时，还需进行源程序代码相似度分析。③多特征综合评分。综合相似度分析结果，结合人工评分准则，按照评分策略给出评判结果。

1.2 代码预处理

目前的程序设计类课程教学基本上仅涉及控制台程序，因此学生程序会涉及大量的控制台输入输出语句。部分提示性语句、暂停性语句等会对程序的自动运行产生影响。对于初学者，学生程序中可能会缺乏提示性语句或输出的提示性语句与标准模板程序输出的提示性语句在空格、字符内容、大小写等方面存在一定的不一致。另外，学生程序中若存在暂停性语句会导致其无法自动结束。这些因素都会影响动态测试的评分准确性。

为了有效地进行动态测试，在编译之前需对学生程序进行以下预处理：①去掉程序中所有提示性信息，防止输入输出语句中的提示性信息对运行结果匹配造成干扰；②去掉程序中的空行，避免空行对程序规模特征匹配的准确度造成影响；③为防止因输入格式问题而导致程序无法正常运行，在不影响正常输入的情况下去掉输入中的各种符号，包括空格、换行符、分号等；④在不改变程序语义的情况下，去掉程序中的暂停性语句，例如 system("pause")语句，主函数中位于输出语句和 return 语句之间且靠近 return 语句的 getchar()语句等。

1.3 用例运行结果相似度计算

判断程序是否正确的最直接方法就是运行程序、观察结果。如果结果正确，则完全匹配，相似度为 1，否则相似度为 0。一般情况下，学生程序的运行结果通常包含的几种形式见表 1。

表 1 学生程序常见运行结果类型

运行结果类型	特点
数值型	整型数，单精度浮点数或双精度浮点数
数值序列	有序或无序数组
字符序列	有序或无序字符型数组
字符串序列	每个字符串之间可能含空格或其他分隔符
数值、字符、字符串混合序列	各元素间可能含空格或其他分隔符
字符串	可能含空格、标点符号等
算术表达式	多种运算符

根据学生程序运行结果类型，匹配用例运行结果相似度的算法如下。

输入 学生程序的用例运行结果，标准模板程序的用例运行结果。

输出 用例运行结果相似度。

步骤 1　设置 $i=1,n,s=0$，其中 n 为用例运行结果的组数；i 为当前用例运行结果的组数编号，$i=1,2,\cdots,n$；s 为从第 1 组到第 i 组的用例运行结果相似度之和。

步骤 2　判断 i 是否小于等于 n，如果是，判断学生程序的第 i 组用例运行结果的类型；否则，转到步骤 10。

步骤 3　若结果是整型数，直接匹配运行结果，如果相等，则相似度为 1，否则为 0。

步骤 4　若结果是浮点数，设定误差 $\varepsilon=0.000\ 01$，匹配学生程序和标准模板程序运行结果，如果误差 $\leqslant\varepsilon$，则相似度为 1，否则为 0。

步骤 5　若结果是数值序列，则依次匹配每个数值。对整型数组，仅当所有数值都依序完全匹配时，相似度为 1，否则为 0；对于浮点型数组，则依照步骤 4 所述方法，计算每对数值的匹配度。

步骤 6　若结果是字符，利用 ASCII 码进行匹配。若完全匹配，则相似度为 1，否则为 0。

步骤 7　若结果是字符串、字符串序列、混合序列等，先剔除不相关元素，例如空格、标点符等，然后进行字符匹配。

步骤 8　若结果是算术表达式，将其转换为逆波兰表达式，再进行字符匹配。

步骤 9　计算前 i 组用例运行结果相似度之和 s，设置 $i=i+1$，转到步骤 2。

步骤 10　计算 n 组用例运行结果的平均相似度值，即 s/n。

步骤 11　结束。

1.4　源程序代码相似度分析

利用有限自动机对预处理后的源程序进行词法分析，可以生成属性字序列。单词的属性，例如分界符、标识符、运算符、关键字等是单词的特征。属性字是单词及其特性的二元组，可表示为＜关键字，'if'＞、＜运算符，'＋'＞、＜分界符，'；'＞、＜标识符，'book'＞等。例如 C 程序代码：while(i＞＝j) i－－；经词法分析后，可生成属性字序列＜关键字，'while'＞＜分界符，'('＞＜标识符，'i'＞＜运算符，'＞＝'＞＜标识符，'j'＞＜分界符，')'＞＜标识符，'i'＞＜运算符，'－'＞＜运算符，'－'＞＜分界符，'；'＞。可以看出，属性字序列反映了程序实现的过程顺序和包含的单词及其特性。通过计算两段程序属性字序列的相似度，就可判断两段程序的相似程度。

最长公共子序列（longest common subsequence，LCS）是将两个或多个给定字符串分别删除任意多个字符后得到的顺序不变且长度最长的相同字符序列。LCS 越长，序列越相似。因此，通过计算学生程序和标准模板程序的属性字序列的最长公共子序列，根据最长公共属性字子序列的长度以及学生程序和标准模板程序的属性字序列长度，就可以计算出属性字序列的相似度，也就是两段程序的相似度。基于动态规划 LCS 算法的源程序代码相似度 S_1 的计算方法为

$$S_1=2L(T_1,T_2)/(t_1+t_2) \tag{1}$$

式中：T_1 和 T_2 分别表示学生程序和标准模板程序的属性字序列；t_1、t_2 表示 T_1 和 T_2 的长度；$L(T_1,T_2)$ 为 T_1 和 T_2 最长公共子序列的长度。

1.5　AST 特征相似度分析

定义从 AST 中提取的能够反映学生程序和标准模板程序相似性的各项特征如下。

定义 1　规模特征。程序长度、程序词汇量所组成的单元。程序长度为程序中的所有操作符和操作数的数量之和；程序词汇量为程序中的所有操作符及操作数的种类数量之和。

定义 2　变量特征。程序中全局变量和局部变量的类型和个数所组成的序列。

定义 3　类特征。可以最大程度衡量类的语义结构的特征向量，包括变量、调用、继承、多态、方法个数等信息。

定义 4　表达式特征。程序中运算符种类和个数所组成的序列。

定义 5　结构特征。程序控制结构序列。控制结构包括循环、选择和顺序结构。为便于处理，用关键字 Loop 统一标识循环结构（while，do-while，for），用 Branch 统一标识分支结构（if-else，switch），用每个关键字后边紧跟的数字表示关键字的嵌套深度。

如果将特征视为一个文档，将特征分量视为文档中的词项，那么计算学生程序和标准模板程序某项特征的相似度就相当于计算两个文档之间的相似度。因此，可以利用向量空间模型来计算 AST 特征的相似度。

向量相似度[12]计算方法很多，例如夹角余弦法、广义 Jaccard 系数法等。广义 Jaccard 系数采用乘积方式，分母中减去两个向量相同的部分，放大了向量的差异，增大了向量相似度的辨识度，常用来计算不完全相同的两个向量的相似度，适用于如 AST 特征这样的离散型数据。设向量 $\boldsymbol{u}$、$\boldsymbol{v}$，这两个向量的相似度使用广义 Jaccard 系数计算的公式为

$$J(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{uv}/(\|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2 - \boldsymbol{uv}) \quad (2)$$

其中 $J(\boldsymbol{u},\boldsymbol{v})$ 表示了向量 $\boldsymbol{u}$ 和向量 $\boldsymbol{v}$ 共同具有的特征项在两向量全部特征项中的比重。在不考虑特征项顺序的情况下，式(2)反映了两个向量的相似程度，可以利用向量空间模型计算从 AST 抽取的变量特征、表达式特征、类特征和规模特征。若用下标 1 表示学生程序，下标 2 表示标准模板程序，各特征的相似度计算方法为

$$S_i = \left(\sum_{v\in V_i} d_{1i}d_{2i}\right)/\left(\sum_{v\in V_i} d_{1i}^2 + \sum_{v\in V_i} d_{2i}^2 - \sum_{v\in V_i} d_{1i}d_{2i}\right) \quad (3)$$

式中：S_i 表示 AST 特征中特征 i 的相似度；V_i 表示学生程度和标准模板程序中特征 i 的集合；d_{1i} 和 d_{2i} 分别表示特征分量 v 在学生程序和标准模板程序中出现的频数。

利用向量空间模型计算文档相似度时，并没有考虑检索词出现的顺序。由于结构特征是程序控制流程的反映，具有一定的上下位顺序关系，因此，对结构特征的相似度计算采取与 1.4 节所述源程序代码同样的处理方法。

1.6 综合评分策略

综合各项特征的相似度分析结果，设计如下综合评分模型

$$F = \left(\sum_{i=1}^{N} S_i w_i\right)\times 100 \quad (4)$$

式中：$F\in[0,100]$ 为学生程序得分；N 为特征个数，由图 1 可知，用于评判学生程序的特征信息总体包括用例运行结果、AST 抽取的 5 项特征和源程序代码共 7 个特征，即 $N=7$；$S_i\in[0,1]$ 为特征 i 的相似度，S_i 为 1 表示学生程序与标准模板程序的特征 i 完全匹配；w_i 为特征 i 的权重值，取值范围为[0，1]。不同情况下各特征的 w_i 值设置不同。按照图 1 所示评分模型，当学生程序编译通过时，源程序代码特征的权重为 0；当学生程序编译失败时，用例运行结果特征的权重为 0。

根据学生程序是否编译通过，设计综合评分策略如下。

(1)学生程序编译通过。①如果全部测试用例运行结果正确且规模相似度达到设定阈值，输出学生程序为满分；②若部分测试用例运行正确，依据式(4)进行评分，此时结果特征和 AST 各项特征的权重均为 1/6；③对全部测试用例运行结果正确但规模相似度小于设定阈值、全部测试用例运行均不正确、编译成功但不可运行等 3 种情况，依据人工评分原则需扣除一定的错误分，修改评分模型为

$$F = \left(\sum_{i=1}^{N} S_i w_i\right)\times 100\times 80\% \quad (5)$$

由于此时运行结果对评分已经没有意义，故设置用例运行结果特征的权重时将 AST 各项特征的权重值调整为 1/5。

(2)学生程序编译失败。①如果 AST 生成成功，依据人工评分原则编译失败需扣除一定语法分，然后再综合考虑 AST 特征和源程序代码相似度进行评分，修改评分模型为

$$F = \left(\sum_{i=1}^{N} S_i w_i\right)\times 100\times 70\% \quad (6)$$

此时源程序代码相似度和 AST 各项特征的权重值均为 1/6；②如果 AST 生成失败，即无法获取 AST 特征，说明学生程序存在较为严重的语法错误，依据人工经验取源程序代码相似度和 0.1 之间的较小值进行评分。

2 实验结果与分析

2015 年，我们将多特征综合分析自动评分方法(multi-feature analysis，MFA)应用于“大学计算机”MOOC 在线编程作业和期末考试的自动评判中，并以人工评分为标准，对期末考试的 367 份答卷、10 道编程题进行了评分准确率测试。评分准确率的计算公式为

$$P = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{|M_1 - M_2|}{M_0}\times 100\% \quad (7)$$

式中：P 是评分准确率；n 是考生答卷份数；M_1 是教师严格按照评分准则给出的分数；M_2 是指采用自动评分方法给出的分数；M_0 是指该题的满分分值。

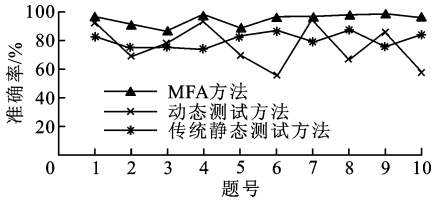
实验在 Visual Studio 2013 环境下进行，每道题目提供 1 份模板程序和 3 个测试用例。

2.1 评分准确率分析

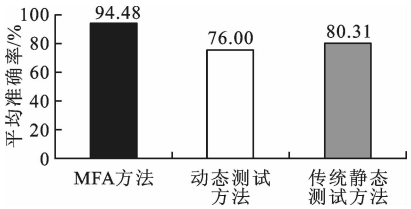
我们对学生答卷中的题目分别用 3 种方法进行评分测试：①动态测试，采用 1.3 节所述的直接匹配用例运行结果的方法；②传统静态测试，仅依据对程序中间表示形式抽取的各项特征的相似度分析结果；③本文提出的 MFA 方法，按照式(7)分别计算 3 种方法的评分准确率，图 2 给出了 10 道题目 3 种方法评分准确率的对比。

由图 2 可以看出，MFA 方法的平均评分准确率达 94.48%，较动态测试方法和传统静态分析方法分别提高了 18.48%和 14.17%。

虽然 MFA 方法的最高评分准确率可以达到

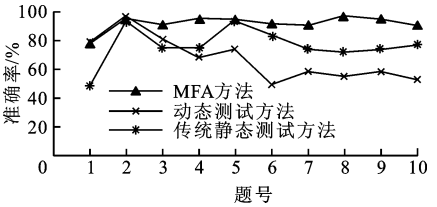


(a)10 道题目的平均评分准确率比较

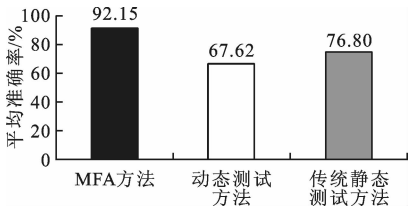


(b)平均评分准确率比较

图 2　3 种方法评分准确率比较



(a)10 道题目的平均评分准确率比较



(b)平均评分准确率比较

图 3　编译失败时的 3 种方法评分准确率比较

98.58%，但从图 2 也可以看出，第 3 题和第 5 题的评分准确率未能达到 90%（分别为 86.16% 和 88.86%）。主要原因是：①在用例运行结果全部错误、AST 各项特征相似性在 50% 以上时，MFA 方法的评分会高于人工评分（如第 3 题）；②MFA 方法采用了平均权重设计，各项特征的权重相同，而人工评分会针对具体题目人为提高某项特征的权重，如在第 5 题中部分学生程序的类特征相似度在 50% 左右，但其他特征相似度在 20% 以下，教师对该题更关注类知识点，由此导致人工评分高于 MFA 评分。图 2 中，第 6 和第 10 题的动态评分准确率较低，是因为有 80% 的学生程序的测试用例仅部分通过或全部不通过，这说明了仅使用动态测试方法会有很大局限性。第 4 题的传统静态方法评分效果明显低于其他两种方法，是因为同一道编程题目可能会存在不同的解答，例如，学生程序中定义的变量类型、使用的表达式等可能和标准模板程序不完全一致，虽然并不影响输出结果的正确性，但却会影响 AST 特征相似度的计算结果，使得仅依赖中间形式进行评分的效果不佳。

2.2　编译失败时的评分准确率分析

专门针对学生程序编译失败情况下 MFA 方法的评分准确率进行了分析，结果如图 3 所示。可以看出，此时 MFA 方法的平均准确率达 92.15% 以上，明显高于动态测试方法 67.62% 和仅基于 AST 的传统静态评分方法 76.80% 的平均准确率。但是第 1 题和第 2 题使用 MFA 方法评分的准确率低于动态测试的准确率，导致这一点的主要原因是学生程序的多个特征与标准模板程序具有 20% 以上的相似度，尤其变量相似度高达 90%，但程序不符合编程思想且有较多错误语句（例如 cin≫"N";），人工评分几乎为 0 分。MFA 方法因考虑了源程序代码相似度和 AST 各项特征的相似度，没有给出 0 分的评价，而动态测试方法在程序编译失败时评分为 0 分，因此其结果表现得更接近人工评分。另外，在图 3a 中 MFA 方法的准确率均高于传统静态评分方法的准确率，这也说明 MFA 方法在编译失败时考虑源程序代码相似度的做法是比较合理的。

3　结　论

随着 MOOC 在全球的兴起，对编程题的在线自动评判方法及其准确率的研究成为了新的热点。与现有研究相比，本文提出的 MFA 方法分别考虑了编译成功和编译失败两种不同情况。对编译成功且运行结果全部正确的程序引入规模特征相似度计算，防止了因直接输出结果对评分准确率产生的影响。对编译成功但运行结果不正确的程序，借助了 AST 特征相似度分析；在编译失败时，综合考虑了 AST 特征相似度和源程序代码相似度，有效防止了编译失败情况下 AST 文本与程序实际语义存在较大差异，甚至无法生成 AST 的情况。实验结果表明，相对于仅依靠动态测试或传统静态分析的评分方法，本文方法表现出更好的健壮性。

虽然实际测试结果已表明本文方法总体上优于仅依靠动态测试或传统静态分析的评分方法，但受各种因素限制，还无法做到对其他各类程序语言所编程序代码的准确评判。另外，由于静态分析依赖于标准模板程序，若能通过系统筛选将人工评分为满分但程序等价性分析相似度较低的学生程序自动添加到标准程序库，将会进一步提高评分的准确性。

参考文献：

[1] STAUBITZ T, KLEMENT H, RENZ J, et al. Towards practical programming exercises and automated assessment in massive open online courses [C]// Proceedings of the 2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering. Piscataway, NJ, USA: IEEE, 2015: 23-30.

[2] ALBER S, DEBIASI L. Automated assessment in massive open online courses [EB/OL]. (2013-07-16) [2015-12-12]. http://uni-salzburg.at/fileadmin/multimedia/SRC/docs/teaching/SS13/SaI/Paper_Alber_Debiasi.pdf.

[3] PIETERSE V. Automated assessment of programming assignments [C]// Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research. New York, USA: ACM, 2013: 45-56.

[4] ALA M, KIRSTI M. A survey of automated assessment approaches for programming assignments [J]. Computer Science Education, 2005, 15(2): 83-102.

[5] POZENEL M, FURST L, MAHNICC V. Introduction of the automated assessment of homework assignments in a university-level programming course [C]// Proceedings of the 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics. Piscataway, NJ, USA: IEEE, 2015: 761-766.

[6] RUBIO M, KINNUNEN P, PAREJA C, et al. Student perception and usage of an automated programming assessment tool [J]. Computers in Human Behavior, 2014, 31(2): 453-460.

[7] GUPTA S, DUBEY S K. Automatic assessment of programming assignment [J]. Computer Science & Engineering, 2012, 2(1): 67-74.

[8] VUJOSEVIC M, NIKOLIC M, TOSIC D, et al. Software verification and graph similarity for automated evaluation of students' assignments [J]. Information & Software Technology, 2013, 55(6): 1004-1016.

[9] 马培军，王甜甜，苏小红. 基于程序理解的编程题自动评分方法 [J]. 计算机研究与发展，2009，46(7)：1136-1142.
MA Peijun, WANG Tiantian, SU Xiaohong. Automatic grading of student programs based on program understanding [J]. Journal of Computer Research and Development, 2009, 46(7): 1136-1142.

[10] 王倩，苏小红，马培军. 有语法错误的编程题自动评分方法研究：用局部语法分析和采分点匹配实现 [J]. 计算机工程与应用，2010，46(17)：239-242.
WANG Qian, SU Xiaohong, MA Peijun. Automatic grading method for programs with syntax error: via local syntax analysis and key point matching [J]. Computer Engineering and Applications, 2010, 46(17): 239-242.

[11] CUI B, LI J, GUO T, et al. Code comparison system based on abstract syntax tree [C]// Proceedings of the 2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology. Piscataway, NJ, USA: IEEE, 2010: 668-673.

[12] 张宇，刘雨东，计钊. 向量相似度测度方法 [J]. 声学技术，2009，28(4)：532-536.
ZHANG Yu, LIU Yudong, JI Zhao. Vector similarity measurement method [J]. Technical Acoustics, 2009, 28(4): 532-536.

[本刊相关文献链接]

李扬，潘泉，杨涛. 基于短文本情感分析的敏感信息识别. 2016，50(9)：80-84. [doi：10.7652/xjtuxb201609013]

杨迎辉，李建华，沈迪，等. 多重边融合复杂网络动态演化模型. 2016，50(9)：132-139. [doi：10.7652/xjtuxb201609021]

戴启华，刘勤让，沈剑良，等. 采用集簇方法的片上网络动态映射算法. 2016，50(8)：52-58. [doi：10.7652/xjtuxb201608009]

魏倩，蔡远利. 一种基于神经网络的中制导改进算法. 2016，50(7)：125-130. [doi：10.7652/xjtuxb201607019]

张小栋，郭晋，李睿，等. 表情驱动下脑电信号的建模仿真及分类识别. 2016，50(6)：1-8. [doi：10.7652/xjtuxb201606001]

姜涛，黄伟，王安麟. 多路阀阀芯节流槽拓扑结构组合的神经网络模型. 2016，50(6)：36-41. [doi：10.7652/xjtuxb201606006]

宋青松，田正鑫，孙文磊，等. 用于孤立数字语音识别的一种组合降维方法. 2016，50(6)：42-46. [doi：10.7652/xjtuxb201606007]

陈江城，张小栋. 人体下肢行走关节连续运动表面肌电解码方法. 2016，50(6)：61-67. [doi：10.7652/xjtuxb201606010]

刘兆丽，秦涛，管晓宏，等. 采用用户名相似度传播模型的线上用户身份属性关联方法. 2016，50(4)：1-6. [doi：10.7652/xjtuxb201604001]

（编辑　武红江　苗凌）