结合文本与源代码数据预测编程任务标签

Artyom Lobanov，* Egor Bogomolov，* Yaroslav Golubev，* Mikhail Mirzayanov，† Timofey Bryksin*
*JetBrains Research，†Codeforces{artem.lobanov、egor.bogomolov、yaroslav.golubev、timofey.bryksin}@jetbrains.com、mirzayanovmr@gmail.com

arXiv:2301.04597v1 [cs.SE] 11 Jan 2023

摘要—竞赛编程仍是一项非常受欢迎的活动，同时结合了软件工程与教育。为了准备竞赛和进行练习，参赛者会使用竞赛编程平台上提供的历届竞赛题目的大型题库。一种提高这一过程效率的方法，是为编程任务提供自动标签系统。以往的工作通过使用题目描述或其解答的代码来实现这一目标。

在本研究中，我们探讨哪种信息源对于标签预测更有价值。为回答这一问题，我们在同一数据集上，使用相同的标签集合，对比了现有的两类方法。随后，我们提出了一种新方法，即将用于分析解答的门控图神经网络模型与用于处理题目描述的双向变换器编码器表示模型相结合的集成模型。实验表明，我们的方法在PR-AUC指标上比先前提出的模型高出0.175 。

索引词—**神经网络、分类、算法识别、文本分析、竞赛编程**

I. 引言

竞赛编程是一种脑力运动，参赛者需要使用受支持的某种编程语言编写计算机程序，在时间和计算资源受到一定限制的条件下解决算法题。参加此类竞赛能提高学生学习编程、算法和数据结构的积极性，也有助于他们通过 IT 公司的技术面试 [1]。

竞赛编程平台，例如 Codeforces [2]和 CodeChef [3]，，保存着大量历届竞赛的题目，可用于备战竞赛，也可作为算法或数据结构编程课程的练习。不幸的是，这些题目档案往往缺乏良好的系统整理，因此很难找到特定主题的题目。为缓解这一问题，一些竞赛平台采用了标签系统。标签是分配给题目的一种特殊标记，用来表明题目所属的类别。标签可以描述题目的主题（例如，数学、几何、图、字符串），也可以描述解题时可能采用的方法（例如，动态规划、暴力法、二分查找）。一道题目可以同时拥有这两类中的多个标签。标签通常由人工设置，需要耗费大量时间和精力。此外，最终的标注可能带有主观性，且不一致。自动标签系统旨在解决这些问题，目前已有多种创建此类系统的尝试 [4]–[9]。

一种可能的标签预测方法是分析题目描述。由于题目描述通常以自然语言（例如，英语或中文）撰写，一些作者建议将自然语言处理技术应用于标签预测问题 [4]–[6], [9]。这些方法在预测与题目主题相关的标签方面表现出良好的前景；但仅根据题目描述确定用于描述解题方法的标签，则是一项困难的任务，难度与解答题目本身相近。

为了识别描述题目解法的标签，我们可以利用已提交的该题解答中的信息。这项任务需要进行源代码分析，这也是一个快速发展的领域。此前的若干研究遵循了这一思路，将标签预测视为算法分类问题 [7], [8],，旨在根据代码片段所实现的算法对其进行分类。这种方法也有一些缺点。例如，标签通常用于标记题目，而不是解答。此外，有些复杂题目可能有多种解法，因此某些标签可能与特定解答无关。

遗憾的是，已有研究采用了不同的任务定义和评估方法，因此无法直接判断哪种标签预测方式效果更好。鉴于这两类方法各有优缺点，我们提出将两种数据源结合起来，形成一种新方法。我们方法的核心思想是训练两个独立的机器学习模型：第一个模型采用现代自然语言处理方法分析题目描述，第二个模型提取代码语义，以分析提交的解答。随后，我们将这两个模型集成为一个集成模型，以获得更稳健的结果。由于每道题目可能对应多个相关标签，因此这项任务属于多标签分类问题。实验表明，这种方法优于仅使用一种输入数据的现有解答。

我们的主要贡献如下：
1)新颖的方法，用于预测竞赛编程问题的标签。与以往的研究不同，我们的方法利用了题目描述（文本）和提交的解答（源代码）中的信息。2) **对比**现有的标签预测方法。我们复现了五篇论文中的 13 种现有方法，并在同一数据集上比较了它们的效果。实验表明，我们的模型在 PR-AUC 指标上比现有方法高出 0.175。

*这项工作和论文于 2020 年完成。

3) **复现包**，其中包含我们的训练模型以及所有受评估模型的源代码 [10]。该训练模型可用于预测教育类或竞赛编程问题的标签。

本文其余部分安排如下。第二节讨论标签预测方面的相关研究。第三节介绍我们所用方法的架构。第四节描述用于比较各模型的数据集。随后，我们在第五节介绍实验设置并呈现结果，同时进行讨论。第六节列出对本研究有效性的威胁。最后，我们在第七节总结全文。

II. 背景

A. 自然语言处理

Bora 等人 [4] 采用长短期记忆网络（LSTM） [11] 神经网络模型，根据题目描述预测标签。作者使用预训练的 word2vec [12]向量和独热编码来表示输入数据。他们从 Codeforces（一种热门的竞赛编程平台）获取题目描述，并且只考虑每道题目的第一个标签。在他们的实验中，只有随机森林 [13] （他们测试的机器学习模型之一）的表现优于虚拟分类器；该分类器被用作基线，始终预测最常见的类别。

Athavale 等人 [5] 采用了几种基于卷积神经网络（CNN） [14] 应用于文本分析的方法。他们将标签预测问题分别视为多类别分类（将每条记录映射到多个类别中的一个）和多标签分类（将每条记录同时映射到多个类别）。他们再次使用了来自 Codeforces 的题目数据集。作者尝试预测出现频率最高的 10 个和 20 个标签，其中 CNN 集成取得了最佳结果。作者还请几位具有一定竞赛编程经验的人预测标签。结果表明，人工预测优于最佳模型的结果，但表现仍不理想：在前 20 个标签的多标签分类任务中，宏平均 F1 分数为 0.43。这表明，即使对人类而言，为题目分配标签也是一项困难的任务。

Iancu 等人 [6]研究了同一项任务。作者从 Codeforces 和 TopCoder [15], 收集题目描述，并将其标签归为 9 个类别。作者采用了以下方法：Doc2Vec [16],、基于 word2vec 的长短期记忆网络，以及独热编码。基于独热编码的长短期记忆网络取得了最高的 F1 分数，但根据加权汉明分数（汉明距离 [17]的加权版本），基于 word2vec 的长短期记忆网络表现更好。

Intisar 等人 [9] 采用了略有不同的方法。首先，他们应用主题建模算法（LDA [18] 和非负矩阵分解 [19]）将文本向量化，然后使用这些向量训练并评估了多种分类算法，例如 k 近邻 [20],、随机森林、多项式朴素贝叶斯（MNB） [21], 和多层感知机 [22]。尽管使用隐式特征影响了各个分类算法的表现（对 k 近邻和 MNB 有正面影响，对 RF 则有负面影响），最终准确率（每种特征类型所对应的最佳方法的结果）与词频-逆文档频率 [23] 基线相比并未提高多少（准确率为 0.86 对 0.88）。

所描述的方法擅长预测一般主题，但不适用于定义与特定算法相关的标签。由于题面通常不会明确说明如何解决题目，确定算法标签本质上需要先解决题目，即使对人类来说也很困难。另一个复杂之处在于，题面的措辞往往具有寓意。题面分析的另一个缺点是，我们通常只有相对较少的题目，这可能导致过拟合——即模型试图记忆训练集中的样本，而不是从这些样本中进行泛化。

B. 源代码分析

人们不仅可以分析用自然语言撰写的题面，还可以分析题目解答的源代码。Shalaby 等人 [7] 采用了一种基于度量的方法进行源代码向量化。对于其数据集中的每个 Codeforces 解答，他们计算了 30 种不同的软件度量指标，例如特定类型的变量数量（例如，int、string）、代码行数、循环数量、嵌套循环数量等。作者评估了这些特征在多大程度上能够将不同类别题目的解答区分开来，以及在特定类别内预测某一类算法的效果。他们的实验表明，传统的分类算法和代码度量指标可以成功应用于解答分类任务。

Sudha 等人 [8] 逐字符应用卷积神经网络，即，将每个解答视为由独立字符组成的序列。作者使用了来自 Codeforces 的题目数据集，并训练了一个基于卷积神经网络的模型，将解答分为四个类别。他们还提出，通过从各个解答的分类结果中选取最常见的类别，来综合所有提交的解答所包含的信息。在实验中，他们成功以 0.61 的准确率将题目区分为四类。

这些方法的一个缺点是，标签通常是分配给整个题目的，因此无法保证每个标签都与每份解答相关，因为同一道题可能有多种不同的解法。

III. 提出的方法

本节介绍所提出的方法。我们的主要思路是合并基于文本的模型和基于代码的模型的结果。为此，我们首先训练了一个门控图神经网络 [24]，以从解答中预测标签；然后微调 BERT [25]，以根据题目描述预测标签；最后将两个模型组合成一个集成模型。

我们将介绍分析题目或其解答的方法。这些方法通常会让我们获得解答或题目的向量表示，并需要利用它来预测该解答或题目的标签概率。为此，我们使用由 Dropout、全连接层和 Sigmoid层组成的序列，并将其称为

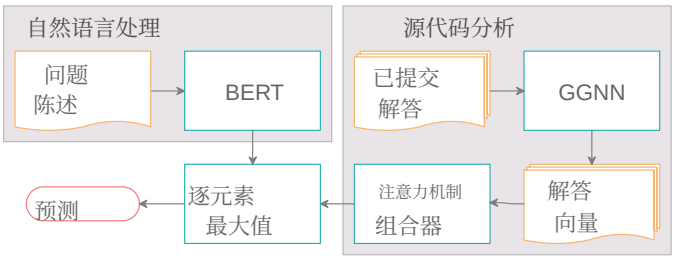


图 1. 所提方法的流程。

TAGPREDICTOR。Sigmoid层会将每个标签的输出映射到 $[0, 1]$ 范围，该值可解释为独立相关性的概率。Dropout层用于正则化，以应对过拟合。整体流程如图 1 所示。现在让我们更详细地介绍每个阶段。

1) 源代码分析：图神经网络 [26] 是一种处理图结构数据的深度学习方法。其工作方式如下：我们为图中的每个节点关联一个数值状态向量，然后在相邻节点之间进行多轮消息交换。这些消息根据节点类型、节点当前状态以及节点之间边的类型来更新节点的状态向量。门控图神经网络（GGNN）是 GNN 的一种特殊变体，它使用门控循环单元 [27]来更新状态向量。近期与语义代码分析相关的研究表明，GGNN取得了良好效果 [24],[28], [29], ，因此我们选择它来处理本研究中的解答。我们的模型基于Allamanis等人提出的 GGNN 版本 [24]。

代码的图表示基于抽象语法树（AST）表示，并在此基础上添加了额外的边。为了构建 AST，我们使用了 Tree-sitter [30], 这一解析器生成工具和一个增量解析库。在本研究中，我们使用了与 Allamanis等人相同的边类型集合 [24]。为了获得整个图而非单个节点的表示，我们添加了一个额外的汇节点，该节点通过一种特殊类型的入边与图中的所有其他节点相连。汇节点的最终状态向量可解释为整个图的向量表示。我们将该向量输入 TAGPREDICTOR，以获得该解答对应的各个标签的概率。模型使用 Adam [31]优化算法通过反向传播进行训练。在实验中，我们从 Allamanis等人描述的超参数 [24], 开始，对其稍作调整并比较结果。最终，我们确定了以下模型超参数：隐藏节点状态和消息的维度为 128，词元嵌入的维度为 32，变量类型嵌入的维度为 8，消息交换轮数为 5。

2）合并解答：为了汇总各个解答中的信息，并预测整个问题的标签，我们冻结了在单个解答上训练的门控图神经网络模型的权重。接着，我们使用该模型计算每个提交解答的向量表示，并将其合并通过注意力机制 [32] 合成为一个向量，再将其输入 TAGPREDICTOR。我们使用 Adam 优化算法拟合合并模型的参数。为防止过拟合，训练时我们最多只使用 50 个随机抽取的提交解答。

3) 题面分析：双向变换器表示（BERT）[25]是一种现代自然语言处理方法，近期已在多项自然语言处理任务 [33]–[35]中展现出最先进的结果。BERT 背后的重要思想之一是，先在大型数据集上对模型进行一次预训练，然后针对每项新任务对其进行微调。由于利用了关于语言的先验知识，即使数据集较小，也能取得良好的性能。因此，BERT 似乎很适合我们的任务。我们采用 WordPieces [36] 技术将词元拆分为多个片段，这有助于处理词汇表外词语。我们在词元序列开头添加了一个特殊的初始词元，并将其最终对应的向量用作文本向量表示。随后，我们将所得向量输入 TAGPREDICTOR，以将其应用于我们的任务。

4) 集成模型：最后一步，我们将 GGNN 和 BERT 结合成一个集成模型，以进行最终预测。这些模型为每个对象提供了两个概率向量。我们本可以基于这种表示训练一个新模型，但由于这些模型最初是在同一个数据集上训练的，它们在训练集上的表现会好得多，因此这种数据表示并不一致。因此，我们将概率向量按元素取最大值，作为自然的池化方法。

IV. 数据集

在本研究中，我们使用了来自 Codeforces 竞赛编程平台的题目数据集。该数据集包含 1,065 场竞赛、5,981 道题目和 14,257,576 个解答。对于每道题目，我们都有一组人工标注的相关标签、一份题面，以及一组通过所有现有测试的提交解答。在这项工作中，我们只分析了以 C++ 编写的解答，因为基于代码的方法依赖于编程语言，而 C++ 是竞赛编程中最流行的语言。解答数量因题目而异：简单且热门的题目有数千个解答，而最难的题目只有少数几个。数据集中共有 35 种不同的标签，但其中 5 种出现频率过低，因此我们将其略去。

A. 去重

数据集中的解答并非唯一。我们使用 SHA-256 [37] 算法计算每个解答的哈希码，然后移除哈希码相同的解答组，但保留每组中最早提交的解答。

然而，不仅解答可能重复，题目描述也可能重复。几乎相同的题目描述通常对应同一道题目，但难度要求不同（输入规模、执行时间和内存限制不同）。简化版题目可能接受使用暴力法的简单解答，而同一道题目的更高难度版本则可能要求使用高级算法的复杂解答。为了

表 I 去重和子采样后的最终数据集。

数据集	竞赛	题目	解答
训练集	716	3,837	2,493,745
验证集	169	1,020	664,578
Test	159	1,046	734,612
总计	1,044	5,903	3,892,935

避免为单个解答添加额外且无关的标签，我们决定既不删除也不合并这类题目。我们只确保测试集中的题目没有在训练阶段或超参数优化过程中使用。如果题目描述的词元集合相似性根据杰卡德指数 [38] 至少达到 0.9，我们就将其标记为几乎相同。总之，在这一阶段，我们删除了 18 道题目和 2.5% 的解答。

B. 数据划分

我们数据集中的第一场竞赛于 2010 年举行，最后一场竞赛则在 2019 年举办。这些年间发生了许多变化（例如，编程语言中引入了新的语法结构，平台的受众不断扩大），因此数据集并不均一，这可能会影响模型的性能。Shalaby 等人 [7] 的研究对此进行了示例说明。作者比较了在随机划分和按时间顺序划分的数据集上使用相同方法得到的结果，并发现后者的性能指标显著下降。

因此，为了获得可靠的评估结果，我们按时间顺序划分竞赛数据集。为确保划分正确，我们从训练集移除了所有提交时间晚于验证集中首次提交时间的解答。验证集和测试集也采用了相同的处理方式。最后，为了缓解数据不平衡，我们将每道题的解答数量上限设为 1,000；如果某道题的解答超过 1,000 个，就随机将其数量减少至 1,000 个。表 I 展示了去重和划分后数据集各部分的情况。

C. 预处理

由于 C++ 语言的语法丰富，对源代码进行分析是一项复杂的任务。为了稍微简化这一过程，我们移除了所有 *include* 宏，以避免引入标准库中的头文件；运行编译器的预处理器以替换用户定义宏，并移除了注释。

题目描述也需要预处理。最初，题目描述以 LaTeX 格式存储，无法直接用于自然语言处理分析。我们使用 Pan- doc 工具 [39]将题目描述转换为纯文本，然后应用预训练的 WordPieces [36] 分词器，得到 BERT [25] 接收的文本表示作为输入。

V. 评估

本节比较不同的信息来源和标签预测方法。我们将讨论评估指标，概述基线方法，介绍评估实验，并呈现其结果。

A. 评估指标

精确率和召回率 [40] 是普遍采用的分类指标，它们从两个方面衡量预测质量：正确性和完整性。然而，由于分类器通常会提供某个对象属于特定类别的概率，我们可以通过改变分类阈值来影响这些指标。提高阈值通常会提高精确率、降低召回率（因为我们选择的对象更少，但置信度更高），反之亦然。选择阈值始终需要在这些指标之间取得平衡，具体取值取决于模型的应用条件。因此，这些指标通常不会直接用于比较模型的质量。

F1 分数 [40], 是精确率和召回率的调和平均数，更适合用于比较，但仍取决于所选阈值。避免这一问题的一种方法是分析精确率-召回率曲线，该曲线描绘了分类阈值变化时精确率随召回率的变化情况。精确率-召回率曲线下面积（PR-AUC） [41] 是一种常用的分类指标，对高度偏斜的数据集具有较强的稳健性。

我们在实验中选择 PR-AUC 作为主要指标。我们还提供 F1 分数、精确率和召回率；这些指标的阈值是在验证集上拟合的，以最大化 F1 分数。

B. 文本分析

在第一个实验中，我们评估了题目描述对于预测标签的实用性。为此，我们从训练集（见第 IV 节）中提取了所有题目的描述，并训练了多个基于文本的模型来预测其标签。所使用模型的完整列表如下：

1) Bora 等人 [4]：基于独热编码（OHE）的长短期记忆网络（LSTM）、基于 word2vec 的长短期记忆网络（LSTM）、基于词袋（BoW）的逻辑回归（LR）。按照原论文的做法，我们没有采用任何自然语言处理预处理技术（例如，词干提取、移除停用词）。长短期记忆网络模型在预测题目难度的任务上进行了预训练。不过，Codeforces 上并没有严格的难度类别，因此我们使用竞赛组别（有些竞赛面向初学者，另一些则面向更有经验的参赛者）代替难度类别。2) Athavale 等人 [5]：基于可训练词嵌入（TWE + CNN）的卷积神经网络（CNN）、基于预训练 GloVe [42]词嵌入（GloVe + CNN）的卷积神经网络，以及卷积神经网络模型的集成模型。作者公开了源代码，因此我们使用了他们的实现。3) Iancu 等人 [6]：基于独热编码的长短期记忆网络、基于 word2vec 的长短期记忆网络、基于词频-逆文档频率的决策树。我们使用自然语言工具包 [43] 移除了停用词。4) 我们的工作：BERT [25] 模型。我们使用了一个预训练模型，并针对我们的任务对其进行了微调。更多详情请参见第 III-3 节。

表 II 基于陈述的方法在预测题目标签方面的评估。最后一行代表本研究。

方法	PR-AUC	F1	P	R
OHE + 长短期记忆网络 [4]	0.177	0.236	0.188	0.385
词袋 + 逻辑回归 [4]	0.227	0.252	0.252	0.306
word2vec + 长短期记忆网络 [4]	0.185	0.255	0.252	0.362
TWE +卷积神经网络 [5]	0.268	0.303	0.295	0.372
GloVe +卷积神经网络 [5]	0.276	0.292	0.263	0.388
卷积神经网络集成 [5]	**0.278**	0.289	0.296	0.338
OHE + 长短期记忆网络 [6]	0.203	0.259	0.212	0.409
word2vec + 长短期记忆网络 [6]	0.210	0.280	0.252	0.400
词频-逆文档频率 + 决策树 [6]	0.250	0.207	0.219	0.203
BERT	0.273	**0.308**	0.268	0.417

这些实验的结果见表 II。值得注意的是，决策树等传统分类算法表现相当不错，这与 Bora 等人 [4] 报告的发现一致。可能的原因是，这类算法的参数量相对较少，因此比深度学习模型更能抵御过拟合。所有基于长短期记忆网络的模型表现都稍差一些。长短期记忆网络在分类任务中的一个常见问题是梯度消失 [44], ，这是深度网络中反向传播带来的负面影响。相比之下，基于卷积神经网络的模型没有这个问题。它们的集成模型更加稳健，因此取得最佳的 PR-AUC 0。278 并不令人意外。BERT 模型的 PR-AUC 略低（0。273 对比 0。278），但 F1 分数更高（0。308 对比 0。289）。BERT 的优势在于，它在大型数据集上进行了预训练。

这项实验表明，在根据题目描述预测标签时，我们的模型能够达到与其他现有模型相当的表现。

C. 源代码分析

题目的另一个信息来源是提交的解答列表。我们分两个阶段检验这些信息的效用。第一阶段的任务是预测数据集中各个解答的标签。遗憾的是，我们没有每个具体解答对应的标签，因此假设每道题目的标签适用于它的所有解答。第二阶段的任务是根据题目的所有提交解答来预测题目的标签。我们比较了以下模型：

1) Shalaby 等人 [7]：基于人工设计的软件度量指标集，使用自适应提升、随机森林（RF）和 sklearn [45] 中的支持向量机（SVM）。2) Sudha 等人 [8]：字符级卷积神经网络。3) 我们的工作：使用 GGNN 分析单个解答，并使用带注意力机制的 GGNN + 预测题目的标签。更多详情请参见第 III-1 节。

实验第一阶段的结果如表 III 所示。与 Bora 等人 [4], 的研究类似，随机森林取得了 0.254 的 PR-AUC，在所有基于指标的方法中表现最佳。字符级卷积神经网络是一种更灵活的方法，但其表现仍不如随机森林。它也完全忽略了代码结构信息。因此，在单个解分类任务中，门控图神经网络模型的表现远胜所有基线。

表 III 基于代码的方法对解答标签的预测评估。最后一行代表本研究的工作。

方法	PR-AUC	F1	P	R
指标 +自适应提升 [7]	0.225	0.270	0.208	0.474
指标 +随机森林 [7]	0.254	0.303	0.282	0.376
指标 +支持向量机 [7]	0.225	0.272	0.245	0.495
字符级卷积神经网络 [8]	0.249	0.289	0.306	0.323
GGNN	**0.393**	**0.430**	0.467	0.446

表IV 基于代码的方法在预测题目标签方面的结果。最后一行代表本研究。

方法	PR-AUC	F1	P	R
指标 +自适应提升 [7]	0.325	0.356	0.341	0.452
指标 +随机森林 [7]	0.388	0.389	0.386	0.467
指标 +支持向量机 [7]	0.363	0.339	0.297	0.564
字符级卷积神经网络 [8]	0.372	0.360	0.377	0.414
门控图神经网络 +注意力机制	**0.514**	**0.517**	0.487	0.594

在实验的第二阶段，我们比较了给定某道题目的所有解答时，预测该题标签的质量。Sudha 等人 [8] 的研究探讨了类似的任务。他们提出，在对单个解答进行分类的任务上训练一个模型（与我们评估的第一阶段类似），为提交到该题目的每个解答预测一个类别，并将最常见的结果作为该题目的类别。我们采用这一思路，将第一阶段训练的基于指标的方法和字符级卷积神经网络用于预测题目的标签。与原研究不同，我们处理的是多标签任务，因此会针对每个单独的标签应用该方法。我们还提出了另一种聚合解答的方式。我们使用第一阶段的门控图神经网络模型将所有解答转换为向量，然后训练一个基于注意力机制的模型，利用得到的解答向量表示来预测整道题目的标签。

第二阶段的结果见表 IV。不出所料，对整个题目进行分类的结果优于仅对单个解答进行分类的结果，因为现在我们可以考虑解决题目的不同方式，并降低噪声解答的影响。由于我们使用的是实验第一阶段训练的模型，因此这些结果与表 III 中呈现的结果相关并不意外。所有方法的指标值都有所提高，但我们的方法仍取得了最佳的 PR-AUC 得分。

D. 标签预测

在前几节中，我们从两个角度探讨了标签预测问题。乍看之下，基于代码的模型似乎肯定优于

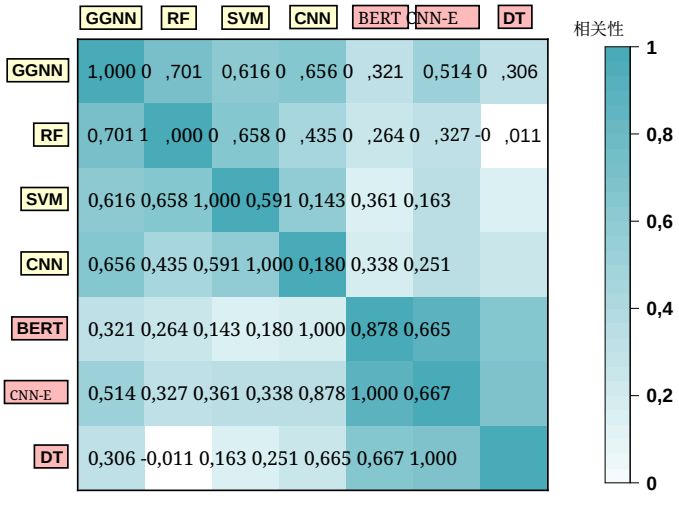


图2. 几种方法的 PR-AUC 值列表之间的相关性。

表V 最终结果。最后几行代表本研究。

方法	PR-AUC	F1	P	R
词袋 + 决策树 [4]	0.257	0.236	0.233	0.249
卷积神经网络集成 [5]	0.278	0.268	0.271	0.327
指标 + 随机森林 [7]	0.388	0.389	0.386	0.467
代码 + 卷积神经网络 [8]	0.367	0.392	0.429	0.419
BERT	0.273	0.308	0.268	0.417
门控图神经网络 + 注意力机制	0.514	0.517	0.487	0.594
BERT + 门控图神经网络	**0.542**	**0.532**	0.489	0.618

基于语句的模型。然而，这种印象的产生是因为表格只展示了所有标签之间的平均指标。实际情况更加复杂。为了说明这一点，我们选取了多个模型，计算它们针对各个标签的 PR-AUC 值列表，然后测量这些列表的皮尔逊相关系数 [46]，即如果某个模型对某个标签的预测效果优于平均水平，那么预计第二个模型对该标签的预测效果相对而言也会更好。结果如图2所示。基于代码的模型以黄色突出显示，基于文本的模型则以红色突出显示。正如图中所示，同类型模型之间的相关性高于不同类型模型之间的相关性，这证明不同类型的模型往往更适合不同的标签集合。因此，同时使用这两类信息非常重要。

如第III-4节所述，我们将门控图神经网络和 BERT 模型结合成一个集成模型，作为不同类型方法的代表。该集成模型与其他方法的比较如表V所示。我们的模型优于所有其他模型，包括单独使用的门控图神经网络和 BERT。

E. 动机示例

该模型让我们有机会通过检查题目来验证现有标签是否正确：对于模型非常确信答案、但答案与当前标签不一致的题目，我们可以进行检查。例如，我们的模型建议为题目添加标签 字符串 和 哈希，题目 *1073/G* [47]。目前该题目没有这些标签，但它显然涉及字符串处理，也可以像提交否*44992266* [48]中那样使用哈希来查找最长公共前缀。

VI. 有效性威胁

内部效度： 相关论文中只有一篇提供了公开可用的实现，因此我们不得不自行复现其他方法，这一过程中可能会出现错误。不过，由于所采用的大多数模型都是该研究领域的典型模型，且相应论文描述了最重要的实现细节，我们认为这一威胁不会对结果产生显著影响。另一个可能的问题是，我们使用的数据集在近十年的时间里由许多不同的人进行标注，因此部分标签可能存在噪声或不一致。然而，由于只有题目作者和解决过这些题目的高经验用户才被允许提供标签，我们预计这些标签大多是正确的。

外部效度： 我们使用了来自单一竞赛编程平台的数据集，因此对于其他题目数据库，结果可能会有所不同。不过，我们认为该数据集仍具有代表性，因为 Codeforces 是最受欢迎的平台之一，且我们没有进行任何特定的题目筛选。我们也无法保证对于使用其他编程语言的解答，能够达到相同的预测质量。不过，C++是竞赛编程中最流行的语言，因此我们相信我们的结果仍然具有价值。

总的来说，尽管这些威胁值得注意，但我们认为已妥善缓解了它们，它们不会使我们的研究结果失去效力。

VII. 结论

本文提出了一种用于竞赛编程问题自动标签预测的新方法。该方法基于将源代码分析与自然语言处理相结合的思路。我们训练了一个采用注意力机制的门控图神经网络，用于分析提交的解答；微调了 BERT 模型，用于处理题目描述；并将这些模型组合成一个集成模型。

我们实现了五篇相关论文中提出的多种现有方法，并在一个包含来自 Codeforces 的题目的数据集上对它们进行了评估。对模型预测结果相关性的分析表明，基于文本的模型和基于代码的模型擅长预测不同的标签，因此将二者结合应能提升预测的整体质量。事实上，我们的模型在 PR-AUC 指标上分别比最佳基于文本的模型高 0.264，比最佳基于代码的模型高 0.175。该项目的源代码和训练模型可在线获取 [10]。

总的来说，虽然现在断言竞赛编程题目的人工标注已不再必要还为时过早，但我们的模型能够显著简化这一流程，并发现现有标签集中的错误。我们相信，本研究能为竞赛编程平台的开发者和参赛者带来益处，也希望能对该领域的其他研究人员有所帮助。

参考文献

[1] A. Bloomfield 和 B. Sotomayor，《编程竞赛策略指南》，载于《第47届 ACM 计算机科学教育技术研讨会论文集》，2016年，第609–614页。 [2] CodeForces.（访问日期：2023年1月1日）一个竞赛编程平台。 [在线]。网址：https://codeforces.com/ [3] CodeChef.（访问日期：2023年1月1日）一个竞赛编程平台。[在线]。网址：https://www.codechef.com/ [4] A. Bora 和 A. Sinha，《根据自然语言问题描述预测编程问题的算法方法》，2016年。[5] V. Athavale、A. Naik、R. Vanjape 和 M. Shrivastava，《预测编程文字题的算法类别》，载于《第五届嘈杂的用户生成文本研讨会（W-NUT 2019）论文集》，2019年，第84–93页。[6] B. Iancu、G. Mazzola、K. Psarakis 和 P. Soilis，《编程挑战题背景下自动标签预测的多标签分类》，arXiv 预印本 arXiv:1911.12224，2019年。[7] M. Shalaby、T. Mehrez、A. El Mougy、K. Abdulnasser 和 A. Al-Safty，《使用机器学习自动识别源代码中的算法》，载于《2017年第16届 IEEE 国际机器学习与应用会议（ICMLA）》，2017年，第170–177页。[8] S. Sudha、A. A. Kumar、M. M. Nagappan 和 R. Suresh，《使用卷积神经网络对竞赛编程问题进行分类与推荐》，载于《智能信息技术国际会议》，2017年，第262–272页。[9] C. M. Intisar、Y. Watanobe、M. Poudel 和 S. Bhalla，《基于主题建模的编程问题分类》，载于《2019年第七届信息与教育技术国际会议论文集》，2019年，第275–283页。[10] R. Package.（访问日期：2023年1月1日）预测编程任务的标签。[在线]。网址：https://github.com/JetBrains-Research/tag-prediction [11] S. Hochreiter 和 J. Schmidhuber，《长短期记忆网络》，《神经计算》，第9卷，第8期，第1735–1780页，1997年。[12] T. Mikolov、I. Sutskever、K. Chen、G. S. Corrado 和 J. Dean，《词语和短语的分布式表示及其组合性》，载于《神经信息处理系统进展》，2013年，第3111–3119页。[13] T. K. Ho，《随机决策森林》，载于《第三届文档分析与识别国际会议论文集》，第1卷，1995年，第278–282页。[14] Y. Kim，《用于句子分类的卷积神经网络》，载于《2014年自然语言处理实证方法会议（EMNLP）论文集》，2014年，第1746–1751页。[15] TopCoder.（访问日期：2023年1月1日）一个竞赛编程平台。[在线]。网址：https://www.topcoder.com/ [16] Q. V. Le 和 T. Mikolov，《句子和文档的分布式表示》，载于《国际机器学习会议》，2014年。[17] R. W. Hamming，《检错与纠错码》，《贝尔系统技术期刊》，第29卷，第2期，第147–160页，1950年。[18] D. M. Blei、A. Y. Ng 和 M. I. Jordan，《潜在狄利克雷分配》，《机器学习研究期刊》，第3卷，第993–1022页，2003年。[19] D. D. Lee 和 H. S. Seung，《非负矩阵分解算法》，载于《神经信息处理系统进展》，2001年，第556–562页。[20] D. T. Larose 和 C. D. Larose，《从数据中发现知识：数据挖掘导论》，2014年。[21] J. D. Rennie、L. Shih、J. Teevan 和 D. R. Karger，《解决朴素贝叶斯文本分类器的不合理假设》，载于《第20届国际机器学习会议（ICML-03）论文集》，2003年，第616–623页。[22] T. Hastie、R. Tibshirani 和 J. Friedman，《统计学习基础：数据挖掘、推断与预测》，2009年。[23] J. Ramos 等，《使用词频-逆文档频率确定文档查询中的词语相关性》，载于《第一届机器学习教学会议论文集》，第242卷，2003年，第133–142页。[24] M. Allamanis、M. Brockschmidt 和 M. Khademi，《学习用图表示程序》，ArXiv，第abs/1711.00740卷，2018年。

[25] J. Devlin, M.-W. Chang, K. Lee, 和 K. Toutanova，《BERT：用于语言理解的深度双向变换器预训练》，载于 NAACL-HLT 会议论文集，2019 年，第 4171–4186 页。[26] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, 和 M. Sun，《图神经网络：方法与应用综述》，AI Open，第 1 卷，第 57–81 页，2020 年。[27] K. Cho, B. van Merrienboer, C.. G¨ulc.ehre, D. Bahdanau, F. Bougares, H. Schwenk, 和 Y. Bengio，《使用 RNN 编码器-解码器学习用于统计机器翻译的短语表示》，载于 EMNLP，2014 年。[28] M. Allamanis, E. T. Barr, S. Ducousso, 和 Z. Gao，《Typilus：神经类型提示》，载于第 41 届 ACM SIGPLAN 程序设计语言设计与实现会议论文集，2020 年，第 91–105 页。[29] Y. Li, R. Zemel, M. Brockschmidt, 和 D. Tarlow，《门控图序列神经网络》，载于 ICLR'16 会议论文集，2016 年。[30] Tree-sitter。（访问日期：2023年1月1日）解析器生成工具。 [Online]。网址：https://tree-sitter.github.io/ [31] D. P. Kingma 和 J. Ba，《Adam：一种随机优化方法》，arXiv 预印本 arXiv:1412.6980，2014 年。[32] D. Bahdanau, K. H. Cho, 和 Y. Bengio，《通过联合学习对齐与翻译实现神经机器翻译》，载于第三届国际学习表征会议（ICLR 2015），2015 年。[33] C. Sun, L. Huang, 和 X. Qiu，《通过构建辅助句利用 BERT 进行基于方面的情感分析》，载于 NAACL-HLT 会议论文集，2019 年，第 380–385 页。[34] Y. Liu 和 M. Lapata，《使用预训练编码器进行文本摘要》，载于 2019年自然语言处理实证方法会议与第九届自然语言处理国际联合会议（EMNLP-IJCNLP）论文集，2019 年，第 3730–3740 页。[35] M. E. Peters, S. Ruder, 和 N. A. Smith，《要不要调优? 将预训练表示适配到不同任务》，载于第四届自然语言处理表示学习研讨会（RepL4NLP-2019）论文集，2019 年，第 7–14 页。[36] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey 等，《谷歌神经机器翻译系统：弥合人工翻译与机器翻译之间的差距》，arXiv 预印本 arXiv:1609.08144，2016 年。[37] H. Gilbert 和 H. Handschuh，《SHA-256 及相关算法的安全性分析》，载于密码学选定领域国际研讨会，2003 年，第 175–193 页。[38] J. C. Gower 和 M. J. Warrens，《相似性、相异性与距离度量》，Wiley StatsRef：在线统计学参考资料，第 1–11 页，2014 年。[39] M. Dominici，《Pandoc 概述》，TUGboat，第 35 卷，第 1 期，第 44–50 页，2014 年。[40] D. M. Powers，《评估：从精确率、召回率和 F值到 ROC、知情度、标记度与相关性》，2011 年。[41] J. Davis 和 M. Goadrich，《精确率-召回率与 ROC 曲线之间的关系》，载于第二十三届国际机器学习会议论文集，2006 年，第 233–240 页。[42] J. Pennington, R. Socher, 和 C. D. Manning，《GloVe：用于词表示的全局向量》，载于 2014年自然语言处理实证方法会议（EMNLP）论文集，2014 年，第 1532–1543 页。[43] NLTK。（访问日期：2023年1月1日）自然语言工具包。 [Online]。网址：https://www.nltk.org/ [44] R. Pascanu, T. Mikolov, 和 Y. Bengio，《循环神经网络训练的难点》，载于国际机器学习会议，2013 年，第 1310–1318 页。[45] SKLearn。（访问日期：2023年1月1日）scikit-learn。 [Online]。网址：https://scikit-learn.org/ [46] J. Benesty, J. Chen, Y. Huang, 和 I. Cohen，《皮尔逊相关系数》，载于《语音处理中的噪声抑制》，2009 年，第 1–4 页。[47] P. 1073/G。（访问日期：2023年1月1日）一个关于字符串的问题。 [Online]。网址：https://codeforces.com/contest/1073/problem/G [48] S. No44992266。（访问日期：2023年1月1日）提交记录 No44992266。[Online]。网址：https://codeforces.com/contest/1073/submission/ 44992266