文章编号:2096-1472(2023)10-0032-04
DOI:10.19644/j.cnki.issn2096-1472.2023.010.007

基于预训练模型的代码分类研究

梁 瑶,洪庆成,王 霞,谢春丽

(江苏师范大学计算机科学与技术学院,江苏 徐州 221116)

✉ 1726093250@qq.com; 2396769801@qq.com; 6020040016@jsnu.edu.cn; 6020030132@jsnu.edu.cn

摘 要:代码分类是软件开发与管理的基础工作,有利于代码的重用、理解、查找和维护。现有的有监督学习方法需要大量带标签数据作为训练样本,而数据的标注成本很高,针对这一问题,提出了基于预训练的代码分类方法。首先,对代码进行消除空白、去除低频符号等预处理工作;其次,采用一种基于 BERT 的预训练模型(CodeBERT)在无标注样本上提取代码的语法、语义和上下文语境等相关特征;最后,基于预训练模型在小样本上微调代码分类器。实验结果表明:该方法即使在较小的训练周期也获得了较好的实验结果,其 *F*1 值比文本卷积神经网络(Text-Convolutional Neural Networks,Text-CNN)方法提高了约 12%。

关键词:代码表征;代码分类;预训练模型

中图分类号:TP311 **文献标志码**:A

Research on Code Classification Based on Pre-trained Model

LIANG Yao, HONG Qingcheng, WANG Xia, XIE Chunli

(*School of Computer Science and Technology, Jiangsu Normal University, Xuzhou* 221116, *China*)

✉ 1726093250@qq.com; 2396769801@qq.com; 6020040016@jsnu.edu.cn; 6020030132@jsnu.edu.cn

Abstract: Code classification is a basic task for software development and management, which is conducive to code reuse, code comprehension, code search and code maintenance. Existing supervised approaches for code classification require a large number of labeled data, and the cost of data annotation is high. To solve this problem, this paper proposes a pre-trained code classification method. Firstly, preprocess the code by eliminating whitespace and low-frequency symbols. Secondly, a pre-trained model based on BERT (CodeBERT) is adopted to extract relevant features such as syntax, semantics, and context of the code on unlabeled samples. Finally, the classification task is fine-tuned on the basis of the pre-trained model. The experimental results show that this method achieves good experimental results even in small training cycles, and its *F*1 value is about 12% higher than that of the Text Convolutional Neural Networks (Text-CNN) method.

Key words: code representation; code classification; pre-trained model

0 引言(Introduction)

随着开源代码仓库的出现,网络上存在大量的源代码文件,对代码进行功能分类可帮助开发者快速找到要复用的代码或组件,有利于开发过程中代码的重用、理解、查找和维护,因此代码分类是各种软件开发任务的基础工作。HINDLE 等[1]的研究表明程序语言和自然语言都包含丰富的统计特性,利用自然语言的文本分类技术能够有效地实现源代码分类。基于深度学习的文本分类从浅层表示转变到深层学习表示,分类的准确性与稳定性得到了显著提升。但是,目前的方法多属于有监督学习,需要大量带标注的数据集作为训练样本,而人工标注数据集不仅代价高,而且耗时长。针对这一现状,提出了基于 CodeBERT 的代码预训练模型,首先利用无监督学习技术针对大量无标签的源代码学习其语法、语义、上下文语境等特征信息,其次构建分类器,在少量有监督数据集上进行微调,实现代码分类任务。

收稿日期:2022-10-21

基金项目:江苏省研究生科研与实践创新计划项目(2021XKT1392);江苏省高等学校大学生创新创业训练计划(202010320035Z);江苏省现代教育技术研究课题(2022-R-102067);江苏省教育科学十四五规划立项课题(D/2021/01/139)

1　相关工作(Related work)

代码分类是通过一定的标准、规则将实现不同功能、语义不相似的代码进行区分,最终将语义相似、功能相同的源代码归为同一类别的过程。代码分类首先要从代码不同的抽象层次上提取代码文本中蕴含的特征信息,其次根据分类器映射到不同的类别中,不同的表征方式会影响特征提取的有效性。对于实现不同功能的源代码进行相似度检测的前提,是将源代码按照一定的粒度级别切分为基本代码单元,即检测粒度。目前,代码分类研究根据粒度划分级别主要采用固定切分和自由切分两种方式。固定切分主要包括按行切分、按块切分、按函数切分、按文件切分和按项目切分等,对于固定切分,如果划分的粒度级别太大,则源代码切分后的丢失检查概率会很高,如果划分的粒度级别太小,则检测工作会耗费更多的时间与精力。因此,在具体的应用中,可以采用多种粒度对源代码进行切分操作,例如按照 Token 切分、按字符串切分、按特征向量切分、按树结点切分和按子图结构切分等,不同源代码的表征粒度直接影响相似性计算的精度。

早期的代码分类是借鉴文本分类的方法,20 世纪 50 年代至 60 年代,文本分类领域的资深学者通过积累的工作经验定义一些规则,从而实现分类,这种方法不仅耗时费力,还要求研究人员必须对某一领域有足够深入的了解,才能总结分类规则并进行分类筛选,因此文本分类方法的局限性较大。针对该问题,基于统计学中字词频率的思想,提出了支持向量机(Support Vector Machine,SVM)技术,推动了文本分类的进一步发展。与此同时,深度学习技术在自然语言领域的成功应用给代码分类带来了新的突破,MOU 等[2]通过一个单层神经网络学习抽象语法树(Abstract Syntax Tree,AST)中上层结点的词向量,提出一种基于树的卷积神经网络模型(Tree-Based Convolutional Neural Network,TBCNN)进行代码分类。BEN 等[3]针对代码本身具有的一些结构化特性,提出了连续空间的语句向量(Inst2vec)模型,该模型在代码分类和性能预测等多种任务中获得了很好的效果。ZHANG 等[4]提出了语句级别的向量嵌入,有效捕获了代码的语法和语义信息,学习的代码向量被用于代码克隆检测和代码分类,取得了良好的分类效果。预训练模型能从大量无标签数据中学习代码的特征信息,因此被广泛应用于代码表征任务[5]。例如,自监督预训练模型(InferCode)[6]和基于 BERT 的预训练模型(CodeBERT)[7]。本文将 CodeBERT 预训练模型引入代码分类任务,一方面利用预训练模型强大的特征学习能力提高代码分类的性能,另一方面避免进行昂贵的人工标记,降低了实验成本。

2　代码分类模型(Code classification model)

代码分类模型主要由源码预处理、源代码文本表征以及构建分类器三个部分组成,数据预处理主要是克服源代码文本的篇幅长度、空白字符等消极因素对模型的影响。源代码文本表征指利用 CodeBERT 模型将源代码文本中的每一个词映射为词向量,所有词向量经过拼接处理、特征提取后,可以得到整篇文本的向量化表示,文本表征的输出向量是分类器的输入,将源代码的表征向量送入 Softmax 层进行测试并评估结果,实现代码分类。

2.1　源码预处理

首先将获取的源代码数据集进行预处理,使数据集尽可能地符合预训练模型的输入要求,其次对处理后的源代码数据集进行向量化操作,使得分类的源代码文本向量可以被模型训练学习。目前,常见的文本预处理方法有清除文本杂质、去停用词、类别匹配和文本过滤等[8]。清除中文文本杂质的具体流程一般包括消除难以识别的特殊符号、删除多余的空白字符、将繁体汉字转换为简体汉字三个过程。由于本文数据集是源代码文本,因此本文中去除源代码文本杂质的流程包括清除计算机难以辨识的特殊符号和删除多余的空白字符两项操作,使源代码文本的特征表示只关注源代码中每一个 token 的语义和特征信息,从而提取源代码文本中有价值的特征信息,以获得更加有效的代码表征。由于源代码文本中通常存在输入输出、语句分隔符、括号等反复出现且无重要信息的词,所以本文将这类词加入停用词词库进行筛选,避免无特殊含义的词作为输入词。由于在分类任务中需要少量有监督的样本,因此需要预先知道源代码实现的功能并对其进行标注处理,将实现相同功能的源代码文本整理在同一个文件夹中。为防止源代码文本篇幅长度超过本文模型的输入范围和样本数据分布不一致问题,本文将对源代码文本的篇幅长度进行筛选并对所有类别的文本进行整体过滤。

2.2　源代码文本表征

传统研究方法通过词向量(Word2vec)模型[9]得到源代码文本的词向量,经过此模型训练得到的不同词向量的大小固定,而一篇完整的文本可能由许多不同的字词组成,真正有价值的字词往往只占很小的一部分。此外,由于 Word2vec 模型训练得到的词向量相同,而相同词向量对应的字词在不同的语境中表达的含义不同,因此为符合现实的语义要求,使用 CodeBERT 模型替代 Word2vec 模型训练文本向量。

在预训练阶段,CodeBERT 模型的输入通常设置为自然语言片段与程序语言片段组成的一个句子对,它们之间用一个特殊分隔符[*SEP*]进行分隔,具体的输入形式为[*CLS*],w_1,w_2,…,w_n,[*SEP*],c_1,c_2,…,c_m,[*EOS*],其中第一个片段 w_1,w_2,…,w_n 是自然语言文本,n 为文本长度,第二个片段 c_1,c_2,…,c_m 则是程序语言文本,m 为程序长度。[*CLS*]、[*SEP*]和[*EOS*]是具有特殊作用的标志,[*CLS*]固定位于输入语句对的开头,[*SEP*]标志仅用于分隔输入语句对中不同模态的语言,在后面的训练过程中一般无特殊作用,[*EOS*]则固定位于输入语句对的结尾,它本身不具有任何特殊的含义。

CodeBERT 的输出包括每个 token 的上下文向量表示和[*CLS*]的表示。[*CLS*]作为聚合序列表示蕴含整篇文本的丰富语义信息,在下游任务中一般使用[*CLS*]表征向量,对其参数进行微调可以实现不同的功能应用。由于实验数据集使用的是 C++源码,其本质不含有任何自然语言的单模态数据,所以输入的具体形式为[*CLS*],c_1,c_2,…,c_m,[*EOS*]。通过模型训练可以得到源代码中每一个 token 的向量化表示和蕴含整篇文本信息的[*CLS*]向量,在后期的分类任务中,利用[*CLS*]向量初始化分类器。

2.3 构建分类器

2.3.1 Softmax 回归模型

本文使用 Softmax 回归模型[10]实现代码功能的多分类任务。Softmax 回归模型是逻辑回归模型在多分类问题中的延伸与扩展，一般采用极大似然估计的方法进行参数估计，主要是对已知标签类型的样本进行学习训练从而优化模型，因此经过预处理的数据非常适合有监督学习任务。例如，$\{(x^1,y^1),(x^2,y^2),\cdots,(x^N,y^N)\}$，$x^i(i=1,2,\cdots,N)$表示第 i 篇源代码文本对应的向量表征，N 是训练集中源代码文本的数量，$y^i\in\{1,2,\cdots,K\}$，K 是分类数量，y^i 表示第 i 篇源代码文本对应的标签，即该篇源代码实现的功能，由于本文研究的代码分类属于多分类问题，所以 $K>2$。

在测试集中，输入样本 x^i，Softmax 回归模型会根据分布函数公式计算条件概率，即给定测试集中样本 x^i 属于某一个类别的概率，其中出现概率最大的类别即当前样本 x^i 所属的代码功能类别。

因此，最终分布函数会输出一个 K 维向量，每一维度的数值表示当前样本属于已知类别中某一类别的概率，并且模型将 K 维向量进行求和运算，然后做归一化处理，使得所有向量元素和为 1，Softmax 回归模型的判别函数 $h_\theta(x^i)$ 如公式(1)所示：

$$h_\theta(x^i)=\begin{bmatrix}p(y^i=1\mid x^i;\theta)\\p(y^i=2\mid x^i;\theta)\\\vdots\\p(y^i=k\mid x^i;\theta)\end{bmatrix}=\frac{1}{\sum_{j=1}^{k}e^{\theta_j^T x^i}}\begin{bmatrix}e^{\theta_1^T x^i}\\e^{\theta_2^T x^i}\\\vdots\\e^{\theta_k^T x^i}\end{bmatrix}\tag{1}$$

判别函数中的概率公式 $p(y^i=j\mid x^i;\theta)(j=1,2,\cdots,K)$ 是当前样本属于目标类别 j 的概率，根据网络的前向运算获得 K 个输出概率值，选择具有最大输出值的类别 y^k 作为本文研究的预测值，θ 为网络权值参数。

2.3.2 代码分类模型构建

本文提出基于 CodeBERT 的代码分类模型，其输入为训练集 U 的源代码文本 $U=\{(x^1,y^1),(x^2,y^2),\cdots,(x^N,y^N)\}$，$(x^i,y^i)\in U(i=1,2,\cdots,N)$，$x^i$ 为第 i 篇源代码文本，y^i 为第 i 篇源代码文本对应的标签，即该篇源代码实现的功能，输出是源代码分类模型，具体步骤如下。

(1)训练集 U 经过预处理后得到训练集 U'，源代码文本预处理首先删除文本空白字符，降低输入样本的特征维度，其次去除停用词，消除没有价值的字词对模型训练的消极影响，最后进行长度切分处理，使输入源代码文本的篇幅长度符合模型的输入要求。

(2)使用 CodeBERT 预训练模型在训练集 U 上进行微调，通过 CodeBERT 模型处理之后输出得到训练集 U' 对应的特征表示 $V=\{v^1,v^2,\cdots,v^N\}$，其中 $v^i\in V(i=1,2,\cdots,N)$ 是每条源代码文本 x^i 对应句子级别的特征向量。在具体的实验过程中，输入的源代码文本经过 CodeBERT 模型训练得到的[CLS]向量包含该篇源代码的综合语义信息，提取了每一语句中重要字词的特征信息。

(3)将每一篇源代码对应的特征表示 V 输入 Softmax 回归模型进行分类处理，Softmax 回归模型会根据分布函数的条件概率公式计算出该篇源代码最有可能对应的功能类别。

(4)源代码文本分类模型输出每一篇源代码的标签。

3 实验与分析(Experiment and analysis)

本文主要进行了两个实验，即基于 CodeBERT 和基于 Text-CNN 的代码分类，本节主要介绍实验中使用的实验数据、实验过程及实验结果。基于 CodeBERT 的代码分类如图 1 所示。

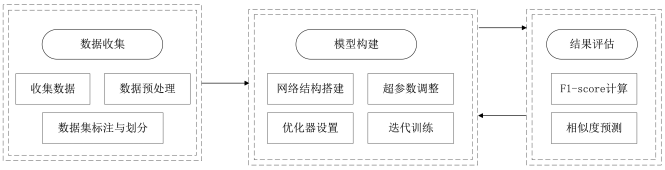

图 1 基于 CodeBERT 的代码分类

Fig. 1 Code classification based on CodeBERT

3.1 数据集

本文收集了江苏师范大学教学科研辅助平台中学生提交的 C++源代码课程作业作为数据集，其中包括 35 个种类，共 905 篇源代码，每个文件夹种类都代表一种功能的实现，包括但不限于进制转换、欧几里得算法、求逆序数、迪杰斯特拉算法及判断闰年。虽然每个种类的文件夹内又包含了数量不等的源代码文本，但是不同种类的源代码文本的数量符合相同的数学分布，即具有大致相同的均值与方差，所以将这些数据进行训练学习可以避免实验误差。本文设置的两组实验均采用相同的数据集，模型评价指标主要为精确率(Precision)、准确率(Accuracy)、召回率(Recall)和 $F1$ 值。

3.2 实验

3.2.1 基于 CodeBERT 的实验

本文实验主要使用 CodeBERT 预训练模型对预处理后带标签的源代码数据进行训练与验证，最终得到每一篇源代码的特征向量，该特征向量不仅含有源代码的语法与语义信息，还包含丰富的上下文语境信息，最终将每一篇源代码的特征向量送入全连接层进行拼接，连接 Softmax 层实现分类。本文实验将预处理后生成的带有标签的数据集划分为训练集、验证集，便于后续 CodeBERT 模型读取和处理，数据文件的每一行都表示为源代码标签。训练集和验证集均经过充分的打乱，服从同样的数据分布，便于模型训练的调参和结果评估。

由于本文实验设备存在内存限制与运行效率低的局限性，因此对本文实验的数据进行划分，按批次输送到模型中。本文模型在验证集上取得的精确率最高为 0.983，$F1$ 值最高为 0.96，具体的实验结果如表 1 所示。

表 1 CodeBERT 模型实验结果

Tab.1 Experimental results based on CodeBERT

Epoch	*Precision*	*Recall*	*Accuracy*	*F1*
Epoch1	0.939	0.939	0.939	0.87
Epoch2	0.983	0.983	0.983	0.96

3.2.2 基于 Text-CNN 的实验

本文实验对数据集进行预处理，匹配并标记源码，按 8∶1∶1

的比例将其划分为训练集、测试集和验证集。由于 Text-CNN 模型数据最终的输入形式为源码的路径信息,因此需要将样本数据集转化为文本文件,该文本文件的具体内容为组合后源代码的不同路径信息及它们的标签。本文实验同样划分数据,将其按批次输送到模型中计算余弦相似度衡量代码语义相似度。

本文模型经过反复训练,相似度阈值范围取[0,1],$F1$ 值的最好结果为 0.846(如图 2 所示),其对应的精确率为 0.853,召回率为 0.840。

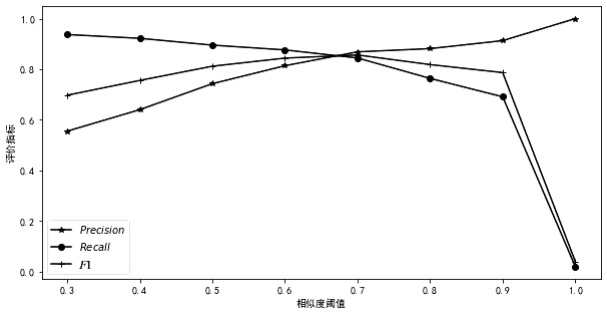

图 2　基于 Text-CNN 实验结果

Fig. 2 Experimental results based on Text-CNN model

3.3　实验结果与分析

本文的两个实验使用了相同的 C++源码数据集,并采用了两种不同的方式从本质上实现了代码分类。因此,本文实验对比分析具有可信度且从对比结果来看,本文实验为代码分类研究提供了新方法、新工具。基于 CodeBERT 的模型训练和基于 Text-CNN 的模型训练对比结果如表 2 所示。

表2　实验对比结果

Tab.2 Comparison of experimental results

模型	*Precision*	*Recall*	*Accuracy*	*F1*
CodeBERT	0.983	0.983	0.983	0.960
Text-CNN	0.853	0.840	0.882	0.846

实验结果表明,基于 CodeBERT 预训练模型的方法可以更全面、准确地反映代码之间的结构关系、语义关系,具有更好的性能指标。但是,该实验存在模型整体结构单一、在识别粒度更小的结构上存在不足等问题。对于相似度计算实验,后期将进一步研究关键词、语义、语法、结构等方面对代码分类结果的影响,设计更全面的检测方法。

4　结论(Conclusion)

在数据量日益激增的信息化时代,如何高效地处理与提取文本数据的有效信息具有非常重要的研究价值。本文在解决源代码的分类问题时,使用 CodeBERT 预训练模型替代传统的 Word2vec 模型作词嵌入操作,利用学习得到的文本向量实现分类。实验通过更新参数为模型挑选适合的优化器,其性能比一般基于文本的词向量方法优越。

通过实验对比发现,本文实验中源代码的表征方法可以充分利用文本的结构与上下文信息,但对代码文本的局部特征信息进行学习时,其效果不如基于 CNN 模型表征学习得到的局部信息丰富。因此,在充分抽取文本局部特征信息的同时,如何利用源代码文本的结构和上下文信息提高分类效率,是未来需要进一步研究的内容。

参考文献(References)

[1] HINDLE A,BARR E T,SU Z D,et al. On the naturalness of software[C]//Institute of Electrical and Electronics Engineers. Proceedings of the 2012 International Conference on Software Engineering. Piscataway:IEEE,2012:837-847.

[2] MOU L L,LI G,ZHANG L,et al. Convolutional neural networks over tree structures for programming language processing[C]//Association for the Advance of Artificial Intelligence. Proceedings of the AAAI Conference on Artificial Intelligence. Menlo Park: AAAI,2016,30(1):1287-1293.

[3] BEN T,JAKOBOVITS A S,HOEFLER T. Neural code comprehension:A learnable representation of code semantics[C]//MIT. Proceedings of Advances in Neural Information Processing Systems. Cambridge,MA:MIT Press,2018:3589-3601.

[4] ZHANG J,WANG X,ZHANG H Y,et al. A novel neural source code representation based on abstract syntax tree [C]//Institute of Electrical and Electronics Engineers. Proceedings of the 2019 International Conference on Software Engineering. Piscataway:IEEE,2019:783-794.

[5] 冷林珊,刘爽,田承霖,等. 预训练增强的代码克隆检测技术[J]. 软件学报,2022,33(5):1758-1773.

[6] BUI N D Q,YU Y J,JIANG L X. InferCode:self-supervised learning of code representations by predicting subtrees[C]//Institute of Electrical and Electronics Engineers. Proceedings of the 2021 International Conference on Software Engineering. Piscataway:IEEE,2021:1186-1197.

[7] FENG Z Y,GUO D Y,TANG D Y,et al. CodeBERT:a pre-trained model for programming and natural languages [C]//Association for Computational Linguistics. Findings of the Association for Computational Linguistics:EMNLP 2020. Stroudsburg:ACL,2020:1536-1547.

[8] 代寒静,涂新辉. 基于 Pre-RoBERTa-MTL 的中文机器阅读理解模型[J]. 计算机应用,2020,40(S2):12-18.

[9] 周顺先,蒋励,林霜巧,等. 基于 Word2vector 的文本特征化表示方法[J]. 重庆邮电大学学报(自然科学版),2018,30(2):272-279.

[10] GAO F,LI B,CHEN L,et al. A softmax classifier for high-precision classification of ultrasonic similar signals[J]. Ultrasonics,2021,112:106344.

作者简介:

梁　瑶(1997-),女,硕士生。研究领域:代码分析。

洪庆成(2000-),男,本科生。研究领域:代码分类。

王　霞(1978-),女,硕士,讲师。研究领域:程序分析。本文通信作者。

谢春丽(1979-),女,博士,副教授。研究领域:软件可靠性分析,深度学习。