背景调查

200007779

监督员：罗莎-费尔盖拉

2023年3月27日

1 简介

代码相似性是计算机科学中的一个重要课题。它可以用来检测抄袭，寻找类似的代码片段，寻找错误修复，检测恶意软件，等等。

在这次调查中，我们将讨论比较跨语言或单语言代码相似性的不同方法，这些方法基于源代码的各种特征，如它们的AST表示、调用图、控制流图、代码em- beddings，以及基于不同架构的训练模型，如RNN、LSTM和转化器。

2 背景介绍

2.1 克隆的类型

正如[8]中提出的，我们通常将克隆的类型分为四类：

- 第一类：相同的代码片段，除了空白处的变化（也可能是布局上的变化）和注释。
- 第二类：结构/句法上完全相同的片段，除了在标识符、字词、类型、布局和注释方面有差异。
- 第三类：复制的片段，并作进一步修改。除了标识符、字词、类型、布局和注释的变化外，还可以改变、添加或删除语句。
- 类型四：两个或多个代码片段，执行相同的计算，但通过不同的语法变体实现。

在这个项目中，我们的目标是检测第四类克隆。因此，训练数据将包含语义上相似但句法上不同的代码对。

2.2 抽象语法树

抽象语法树（Abstract Syntax Tree），简称AST，是用一种形式语言编写的文本（通常是源代码）的抽象语法结构的树状表示。树上的每个节点表示文本中出现的一个结构[12]。它包含了程序的语义和句法信息，因此通常被用来分析源代码之间的相似性。

2.3 控制流图

控制流图或CFG是一种表示方法，使用图的符号，表示在程序执行过程中可能被穿越的所有路径[14]。它能捕捉到更全面的语义信息，如代码中的分支、循环和调用关系链（由调用图表示），但与AST相比，语法信息要粗略得多[3]。它可以和AST结合起来，更好地表示源代码。

2.4 嵌入

在机器学习的背景下，嵌入是将高维数据映射到低维向量的过程，如果低维向量在某种程度上是相似的，那么它们就会彼此接近。在自然语言处理中，嵌入技术通常用于在向量空间中表示单词，单词之间的语义相似性可以通过其向量之间的距离来衡量。编程语言在很多方面与自然语言相似，因此嵌入技术也可以用来衡量代码的相似性。

2.5 递归神经网络

循环神经网络或简称RNN是一种神经网络，节点之间的连接可以形成一个循环，允许一些节点的输出影响到同一节点的后续输入[15]。RNN擅长处理顺序数据或时间序列数据，常用于自然语言处理任务，如机器翻译、语音识别等。

RNN有两个主要的缺点：

- 由于反向传播的性质，它存在着梯度消失/膨胀的问题。虽然有一些解决这个问题的方法，如梯度剪裁和LSTM模型，但它仍然不善于处理长序列的数据。
- 由于RNN在训练时依靠以前的输出来计算下一个输出，它不能利用并行计算的优势，这使得训练过程很慢。

2.6 转换器

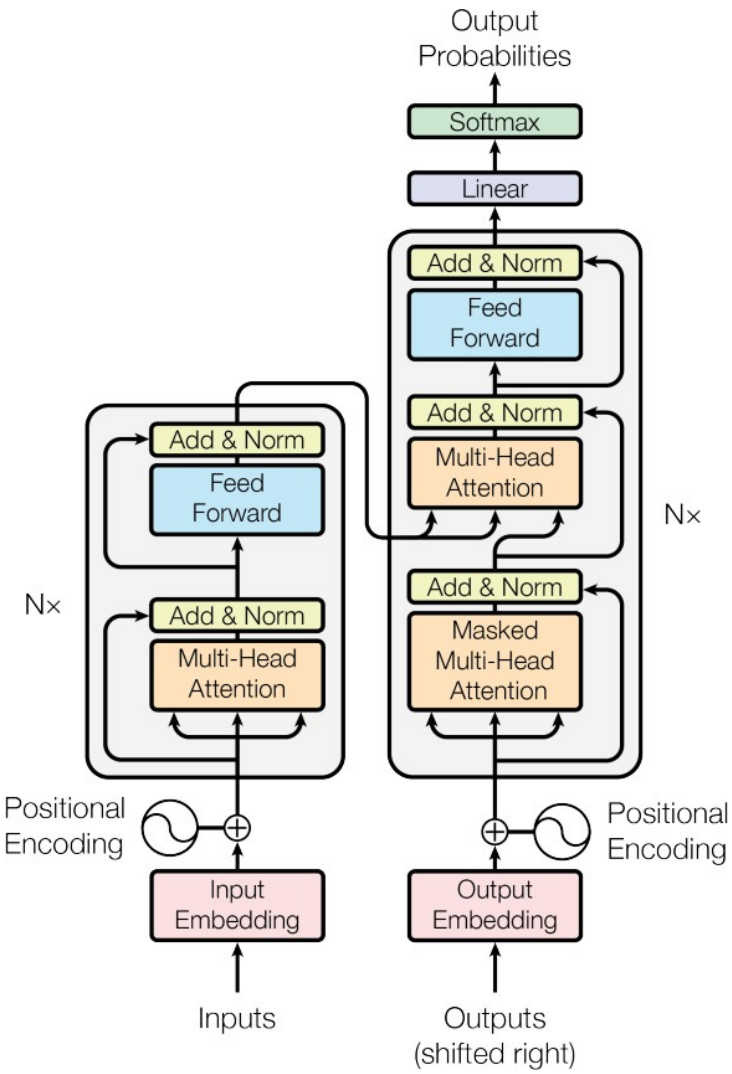

图1：变压器结构[9]

变换器是一种深度学习模型，它采用了自我关注的机制，对输入数据的每一部分的重要性进行不同的加权[16]。它也被设计用来处理顺序数据，但不依赖递归，因此它能够并行处理输入数据，这利用了现代GPU的并行计算能力，使训练过程更加高效。

在训练RNN时，输入序列是按顺序送入模型的，与输入序列是平行送入的变压器不同。在编程语言中，每条语句的顺序决定了该语言的功能。

代码。因此，保持原始语句的顺序对我们的目标至关重要。原始转化器用来保存顺序信息的方式是使用位置编码。在将输入映射到嵌入后，每个嵌入都被添加了一个代表其位置信息的向量，因此，所产生的嵌入是有顺序意识的。

除了位置编码机制使转化器在不丢失输入顺序信息的情况下更有效地进行并行训练外，另一个非常重要的机制称为自我注意，它使转化器能够理解输入数据的每一部分的重要性，以及为这些输入增加上下文信息。它还大大降低了每层的计算复杂性，并增加了可并行化的计算量，如[9]中所述，使其更有效地进行训练。

在转化器被引入之前，大多数最先进的NLP系统重新依赖于门控RNNs[16]。现在，转化器越来越成为NLP问题的首选模型，像BERT和GPT这样的预训练转化器是用真正的大数据集训练的[16]。它们可以针对具体任务进行微调，并取得最先进的结果。

3 方法

3.1 与AST的培训

在许多提议的方法中[7][18][19], RNN如LSTM通常被用作编码器。编码器被训练来转换AST为向量（嵌入）。为了预测相似性，加入了一个分类模型，从嵌入中学习相似性。

本文[7]训练了一个基于树的跳格模型来将AST编码为向量。跳格模型通常用于自然语言处理，将单词变成向量，但它们也可用于编程语言。然后训练一个基于LSTM的神经网络来进一步编码输入数据并预测其相似性。

在另一篇论文[18]中，作者使用了一个Tree-LSTM网络来学习一个函数在AST中的语义表示。树-LSTM网络将两个AST编码为两个向量，然后将输出传给一个连体网络来计算相似度。

上述方法通常使用标记的数据来训练分类器，而编码器是以无监督的方式训练的。也有一些基于无监督聚类方法的方法，如[17]。在这种方法中，从函数AST中提取关键的in-formation并封装成一个自定义对象。每个程序被表示为这些对象的列表，两个程序之间的相似性由它们的列表表示的相似性来计算。最后，相似性矩阵被聚类并使用无监督的方法进行分析。

3.2 用调用图和控制流图进行训练

像[3]这样的方法利用了这样一个事实：具有相同功能的函数通常具有类似的函数调用图和控制流图结构。然而，由于控制流图的抽象性，节点差异

诸如控制流图中运算符的使用可能无法被清楚地识别。因此，在本文中，CFG和AST都被用来作为彼此的补充，并结合在一起，以更好地表示代码的功能。与AST方法类似，嵌入技术被用来对AST和CFG进行编码，以提取句法和语义特征。然后，它们被结合起来训练一个深度神经网络分类器来预测代码片段的相似性。

3.3 变压器方法

BERT 来自变形器的双向编码器表示法，简称BERT，是一种基于变形器的机器学习技术，用于自然语言处理（NLP）的预训练，由谷歌开发[13]。它有两个预训练目标：屏蔽语言模型和下句预测。掩蔽语言模型随机掩蔽输入序列中的一些词，模型被训练来预测掩蔽的词。下一句预测任务是预测第二句是否是第一句的延续。这两个任务都可以以无监督的方式进行训练，这使得BERT可以在大型数据集上进行预训练，如BooksCorpus和英语维基百科[2]。预训练的BERT模型能够对语言有一定程度的理解，并且可以针对具体的下游任务进行微调。

CodeBERT CodeBert[4]是一个编程语言的预训练模型，它是在6种编程语言（Python, Java, JavaScript, PHP, Ruby, Go）的NL-PL对上预训练的多编程语言模型。它采用了与RoBERTa[6](A Robustly Optimized BERT Pretraining Approach)相同的架构，后者是BERT的变体之一。它是第一个针对多种编程语言的大型NL-PL预训练模型。在预训练阶段，它用双模数据（指自然语言-代码对）和单模数据（仅指自然语言或代码）进行训练。它的两个目标是屏蔽语言建模（如上一段所述）和替换标记检测。在替换令牌检测目标中，一些输入令牌被替换成由生成器生成的合理的替代物，目标是训练一个模型，以确定损坏的输入中的每个令牌是否被生成器样本所替换[1]。经过预训练和微调，CodeBERT在自然语言代码搜索和代码文档生成方面都能有最先进的表现。

GraphCodeBERT 与CodeBERT类似，GraphCodeBERT是第一个预训练的模型，它利用代码结构来学习代码表示和改善代码理解[5]。它遵循BERT模型的结构，利用带有配对注释的源代码来预训练模型。它有两个训练目标，即屏蔽语言建模和边缘预测。在边缘预测目标中，该模型通过训练预测数据流中的屏蔽变量的边缘，从数据流中学习表示。当对几个下游任务进行微调和评估时，GraphCodeBERT在自然语言代码搜索方面显示出比CodeBERT稍好的性能，并且能够在克隆检测任务中取得最佳结果。在微调阶段，任务是一个二进制分类，每个输入包括两个源代码和它们的

对应的数据流。输出是两个输入是否是克隆的概率。

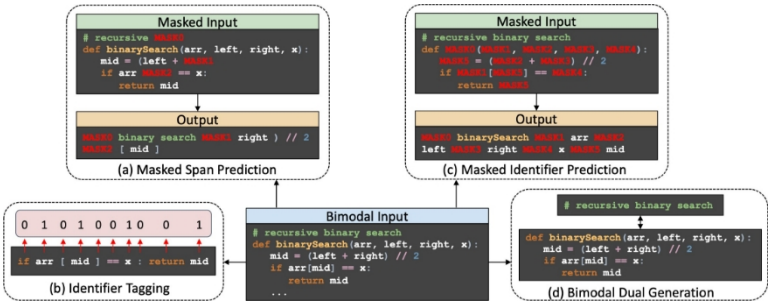

图2：CodeT5的学习目标[11]。

CodeT5 与上述模型和其他许多模型不同，CodeT5[11]既不依赖类似BERT的纯编码器模型，也不依赖GPT的纯解码器模型。当应用于代码归纳任务时，像CodeBERT 这样的纯编码器模型需要一个额外的解码器，它不能从预训练中获益[11]。CodeT5是一个建立在T5架构上的预训练的编码器-解码器模型。与GraphCodeBERT和其他一些利用编程语言结构方面的模型相比，CodeT5专注于利用标识符，其名称是由开发人员指定的，因为它们包含了丰富的代码语义信息。它的目标包括**标识符感知去噪预训练**，要求解码器从随机屏蔽的版本中重新覆盖原始源代码；**标识符标记**，要求模型识别输入中的标识符；**屏蔽标识符预测**，专门屏蔽标识符并要求模型从上下文中恢复它们；以及**双模态双重生成**，旨在改善NL和PL对应物之间的一致性。结果，当用[10]中提供的Java数据对代码克隆检测任务进行评估时，CodeT5的F1得分是97.2，比GraphCode-BERT的97.1和CodeBERT的96.5略好。此外，它在其他下游任务中也表现良好，如代码总结、代码生成等。

4 总结

在这项调查中，我们介绍了代码克隆检测领域的各种最先进的方法。随着代码理解模型列表的增加，越来越明显的是，由于其训练效率和出色的性能，变压器架构在该领域越来越受欢迎。在最近流行的由GPT-3模型驱动的应用程序ChatGPT中也可以看到变压器架构。

因此，在这个项目中，我将使用预训练的转化器，因为它们在更大的数据集上表现更好，而且微调预训练模型的复杂性比从头开始训练要低很多。

参考文献

[1] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning.Electra: 预先训练文本编码器作为判别器而不是生成器, 2020.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova.伯特：用于语言理解的深度双向变换器的预训练，2018。

[3] 方春荣，刘子熙，史艳阳，黄杰夫，史庆凯。语法和语义融合学习的功能代码克隆检测.在*第29届ACM SIGSOFT国际软件测试与分析研讨会论文集*，ISSTA 2020，第516-527页，美国纽约，2020。计算机械协会。

[4] 冯占银，郭大亚，唐德宇，段楠，冯晓成，龚明，寿林军，秦兵，刘婷，姜大新，周明。Codebert: 一个用于编程和自然语言的预训练模型, 2020.

[5] Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, and Ming Zhou.Graphcodebert：用数据流预先训练代码代表，2020。

[6] 刘银汉、迈尔-奥特、纳曼-戈亚尔、杜京飞、曼达尔-乔希、陈丹琪、奥马尔-利维、迈克-刘易斯、卢克-泽特莫耶和维塞林-斯托扬诺夫。Roberta：一个稳健优化的伯特预训练方法，2019年。

[7] Daniel Perez and Shigeru Chiba.通过对抽象语法树的学习-ing进行跨语言克隆检测。在*2019年IEEE/ACM第16届挖掘软件库国际会议（MSR）上*，第518-528页，2019。

[8] Chanchal Roy and James Cordy.关于软件克隆检测研究的调查。*计算机学院TR 2007-541*, 01 2007.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin.注意是你所需要的，2017。

[10] 王文汉，李革，马波，夏昕，和金志.用图神经网络和流动增强的抽象语法树检测代码克隆.在*2020年IEEE第27届软件分析、进化和再设计国际会议（SANER）上*，第261-271页，2020。

[11] Yue Wang, Weishi Wang, Shafiq Joty, and Steven C. H. Hoi.Codet5：用于代码理解和生成的识别器感知的统一预训练的编码器-解码器模型，2021年。

[12] 维基百科的贡献者。https://en.wikipedia.org/w/index.php?title=Abstract_syntax_tree&oldid=1103626323, 2022年，抽象语法树 - 维基百科，自由的百科全书。[Online; accessed 26-October- 2022].

[13] 维基百科的贡献者。Bert (language model) - 维基百科，自由的百科全书。https://en.wikipedia.org/w/index.php?title=BERT_(language_model)&oldid=1107671243, 2022.[Online; accessed 31-October-2022].

[14] 维基百科的贡献者。Control-flow graph - Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Control-flow_graph&oldid=1115609094, 2022.[Online; accessed 25-October-2022].

[15] 维基百科的贡献者。Recurrent neural network - Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1117752117, 2022.[Online; accessed 28-October-2022].

[16] 维基百科的贡献者。Transformer (machine learning model) - Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Transformer_(machine_learning_model)&oldid=1118251522, 2022.[Online; accessed 28-October-2022].

[17] 谢一凡，周文安，胡华淼，吕志成，吴梦媛.基于星形无监督聚类方法的代码相似性检测技术.In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1590-1595, 2020.

[18] 杨守国，程龙，曾益成，郎哲，朱红松，史志强。Asteria：基于深度学习的AST编码的跨平台二进制代码相似性检测。在*2021年第51届IEEE/IFIP可靠系统和网络国际会议（DSN）上*。IEEE, jun 2021.

[19] 张健，王旭，张宏宇，孙海龙，王凯旋，和刘旭东。一种基于抽象语法树的新型神经源代码表示法。In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 783-794, 2019.