增强可追溯性链接恢复未标记数据

朱剑飞*,肖冠平∗∗-, 郑铮≠, 隋玉蕾§
*南京航空航天大学计算机科学与技术学院
†南京大学软件新技术国家重点实验室
≠北京航空航天大学自动化科学与电气工程学院
§澳大利亚悉尼科技大学计算机学院
{zjf、gpxiao}@nuaa.edu.cn, zhengz@buaa.edu.cn , yulei.sui@uts.edu.au

抽象的—可追溯性链接恢复（TLR）是开发值得信赖且可靠的软件系统的一项重要软件工程任务。最近提出的深度学习（DL）模型已经显示出与传统的基于信息检索的方法相比的有效性。深度学习通常严重依赖足够的标记数据来训练模型。然而，手动标记追溯链接既耗时又费力，并且需要领域专家的特定知识。因此，在现实项目中，通常只有一小部分标记数据伴随着大量未标记数据。我们的假设是，如果工件具有相同的链接工件，那么它们在语义上是相似的。

本文介绍了T种族FUN，一种利用未标记数据增强可追溯性链接恢复的新方法。时间种族FUN 首先使用两种相似性预测方法（即向量空间模型和对比学习）来测量未标记和标记工件之间的相似性。然后，基于相似性，在未标记的工件和标记的工件的链接对象之间生成新标记的链接。生成的链接进一步用于 TLR 模型训练。我们评估了T种族在三个 GitHub 项目上享受乐趣，其中包含两个最先进的深度学习模型（即 Trace BERT 和 TraceNN）。结果表明，T种族FUN是有效的，对于 Trace BERT 和 TraceNN，F1 分数最大提高分别高达 21% 和 1,088%。

索引术语—溯源链接恢复、未标记数据、向量空间模型、对比学习

我、我简介

可追溯性链接恢复 (TLR) 是一项软件工程任务，用于恢复不同类型的软件工件（例如需求、源代码、错误报告、测试用例和用户文档）之间的链接。可追溯性在软件开发和维护中发挥着重要作用，为各种软件活动提供有用的支持，例如程序理解[1]、合规性验证[2]、变更影响分析[3]和测试用例回归分析[4]。

手动恢复追溯链接非常耗时、费力且容易出错。在过去的几十年里，人们提出了几种 TLR 方法，通过自动或半自动恢复可追溯性链接来减轻开发人员的负担 [5]-[21]。最近，深度学习（DL）技术已在许多 TLR 方法中得到广泛采用，与基于信息检索（IR）的传统方法相比，性能有了显着的提高[22]-[27]。

*观察和见解。*基于深度学习的方法通常需要大量标记数据。然而，在现实项目中获取此类数据具有挑战性。这是因为从软件存储库手动标记大量真实链接需要领域知识，这是成本效益低的，尤其是对于大型项目。此外，甚至一些启发式规则（例如，正则表达式[28]）也被用来自动检索链接。只能构建一小部分链接，许多链接仍然缺失[16]。例如，我们发现在从 GitHub [27] 收集的 Flask 数据集中，仅建立了与 1,490 个工件相关的 752 个标记链接，但仍然存在 6,233 个未标记工件。

通常，此类未标记的工件将被排除在基于深度学习的模型的训练数据集中。如果我们能够利用丰富的未标记数据源进行模型训练，有望提高性能。众所周知，多个源工件可以链接到相同的目标工件，反之亦然，即多对一关系。由于相同的链接工件，这些源工件通常具有语义相似的关系。例如，三个重复的错误报告可能与一个错误修复提交相关。这些错误报告描述了相同的故障现象，内容相似。如果新提交的报告也有类似的失败描述，则该报告很可能具有相同的错误修复提交。

*我们的解决方案。*受这些见解的启发，本文提出了 T种族有趣，一个痕迹 能力链接恢复F 骨架增强了联合国 标记数据。图1显示了T的概述种族乐趣。首先，为了测量未标记和标记的工件（两者都可以是源工件或目标工件）之间的相似度，我们引入了两种相似度预测方法，即向量空间模型（VSM）和对比学习（CL），如图1所示（A）。VSM是一种常用的预测文本相似度的IR方法[29]，而CL是一种用于在没有标签的情况下对相似和不相似的数据样本进行分类的DL方法[30]。通过这些方法，T种族FUN 计算未标记和标记工件之间的相似性。接下来，根据选择的高度相似的工件对，通过将未标记的工件连接到标记的工件的链接工件来生成新的链接，如图 1（b）所示。最后，将新标记的链接与原始标记数据集成到训练数据集中，以训练基于 DL 的链接

- 通讯作者: 肖冠平.

2022年IEEE第33届软件可靠性工程国际研讨会(ISSRE) | 978-1-6654-5132-1/22/31.00 美元 ©2022 IEEE | DOI：10.1109/ISSRE55969.2022.00050

2332-6549/22/31.00 美元 ©2022 IEEE DOI
10.1109/ISSRE55969.2022.00050

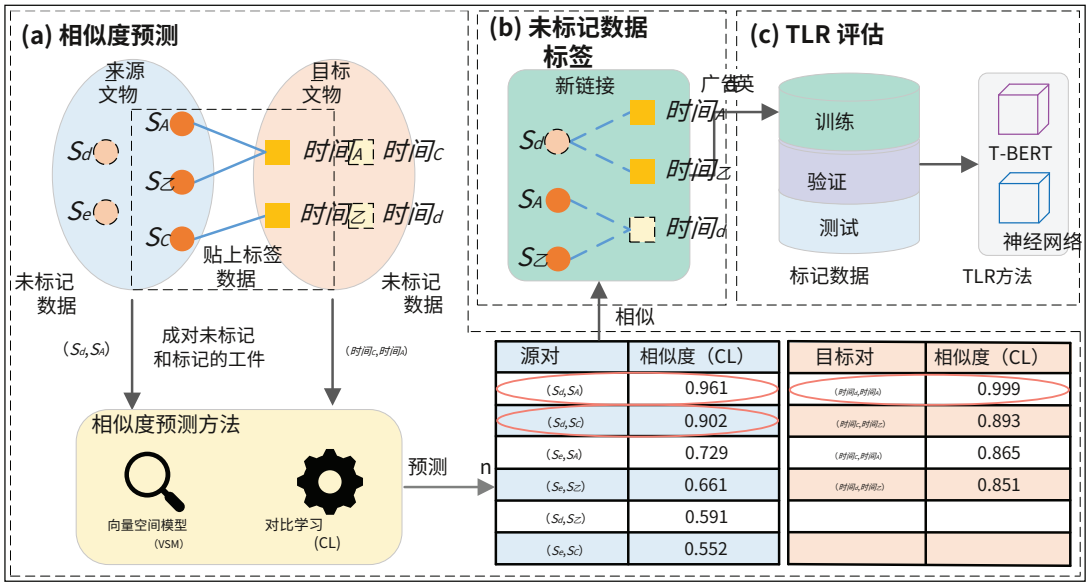

图 1. T 概述种族乐趣。

TLR方法，如图1(c)所示。我们选择了两种最新的方法，即 Trace BERT (T-BERT) [27] 和 TraceNN (TNN) [26]，作为基线 TLR 方法。评估 T种族有趣的是，我们使用从三个开源 GitHub 项目（包括 Flask、Pgcli 和 Keras）收集的未标记数据比较了这两种方法前后的性能。还研究了不同相似性预测方法和新标记数据大小对 TLR 性能的影响。

综上所述，本文有以下主要贡献：

- 据我们所知，本文首次尝试使用未标记数据进行 TLR。
- 时间种族FUN 首次引入 VSM 和 CL 方法来测量未标记和标记工件之间的相似性，以生成新的训练样本。
- 我们评估了T种族通过在三个 GitHub 项目上使用 5 倍交叉验证，将其与两种最先进的方法进行比较，这很有趣。结果表明，T种族FUN 提升了 T-BERT 和 TNN，F1 分数最大提升分别高达 21% 和 1,088%。
- 我们制作了T的源代码种族FUN 可在 https://github.com/TraceFUN 上公开获取。

二. 是奥提瓦汀乙示例

图 2 显示了现实世界中的可追溯性链接示例*易游*项目。四个用例（即UC_1, UC_2, UC_3， 和UC_4) 连接到类*DBBene文化*，执行BeanBeneCulturale列表的增删改查操作。UC_{29}与班级相关联 *标签管理机构*，作为通用标签管理。我们可以发现所有四个用例都描述了对同一对象的操作（即，*文化*），尽管UC_{29}

描述与不同对象相关的操作。

我们使用VSM来计算两个用例之间的相似度，如表I所示。相关对的相似度

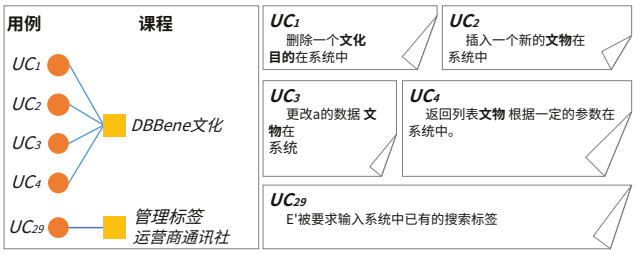

图 2. eTOUR 项目中用例和类之间的可追溯性链接。

表一

S相似之处乙间U东南欧C亚洲电子学会

具有共同的目标神器		没有共同的目标工件	
使用案例	相似	使用案例	相似
(UC_1, UC_3)	0.646	(UC_4, UC_{29})	0.162
(UC_2, UC_3)	0.556	(UC_3, UC_{29})	0.142
(UC_1, UC_2)	0.419	(UC_1, UC_{29})	0.138
(UC_3, UC_4)	0.347	(UC_2, UC_{29})	0.123
(UC_2, UC_4)	0.328		
(UC_1, UC_4)	0.257		

到四个用例（即UC_1, UC_2, UC_3， 和UC_4）约为 0.25 至 0.65，明显高于与相关的那些对UC_{29}（约0.15）。结果是预期的，因为UC_1, UC_2, UC_3， 和UC_4连接到同一目标类。如果一个未标记的用例与用例 1 到 4 相似，则该用例可能具有到类的链接*DBBene文化*。因此，我们可以利用这种相似的关系来标记未标记的数据，从而生成更多标记链接来训练 TLR 模型。

三．氧尿道时间种族乐趣A普罗奇

处理 TLR 的未标记数据有两个主要挑战 (C)。

- C1:如何衡量未标记和标记工件之间的相似性？

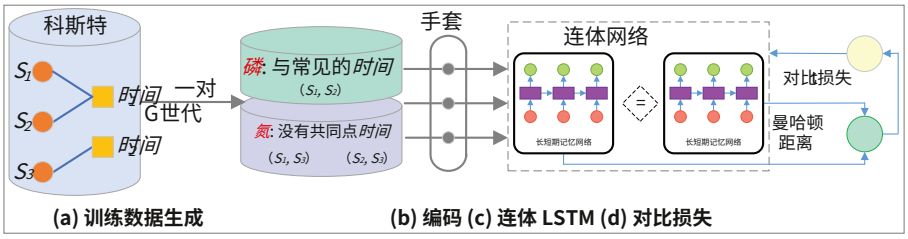

图 3. T 中对比学习的详细结构种族乐趣。

· C2：如何根据计算出的相似度来标记未标记的数据?

为了解决这两个挑战，本节介绍了我们的 T种族FUN，由三部分组成，即相似度预测（C1),未标记数据标记（C2),和TLR评估，如图1所示。

A. 相似度预测

时间种族FUN 测量同一类型软件工件（即源工件或目标工件）中未标记工件和标记工件之间的相似性。为了获得所有可能的对，对未标记和标记的工件执行笛卡尔积计算。两个集合的笛卡尔积*X*和*是*,在集合论中表示为*X*×*是*,是所有可能的有序对的集合：

$$X\times是=\{(x、y)\ |X\varepsilon X\&y\varepsilon是\},\qquad (1)$$

在哪里*X*是一组未标记的工件，并且*是*表示标记工件的集合。例如，在对未标记的源工件执行笛卡尔积计算后 $\{S_d, S_e\}$ 和标记的源工件 $\{S_A, S_Z, S_C\}$，所有可能对的集合 $\{(S_d, S_A), (S_d, S_Z), (S_d, S_C), (S_e, S_A), (S_e, S_Z), (S_e, S_C)\}$ 获得用于进一步的相似性预测，如图1（a）所示。对未标记和标记的目标工件的笛卡尔积计算以相同的方式执行。

由于未标记和标记的工件之间不存在（相似或不同的）标签，T种族FUN引入了两种无监督相似性预测方法：向量空间模型（VSM）和对比学习（CL）。CL专门对未标记的数据进行分类，使得相似的样本彼此接近，而不相似的样本相距很远。为了进行比较，我们使用标准测量VSM，与LDA和LSI等其他传统相似性预测方法相比，它被认为是最好的[27]。请注意，相似性预测方法不限于这两种模型。我们可以在T中添加更多方法种族乐趣。VSM和CL的详细描述如下。

1）向量空间模型： VSM 具有一个文档空间，其中文本内容表示为向量 [5]。文档空间由一个表示*米*×*n*矩阵，其中*米*是项数，并且*n*是文档的数量，也称为“术语-文档矩阵”。一个条目 $d_{我,j}$ 矩阵的权重表示*我*中的第 - 项*j*-th 文档。在最简单的情况下，如果*我*- 第一项出现在*j*-第一个文档，否则为0。

更常用的测量是使用术语频率-逆文档频率（TF-IDF）方法[31]。TF-IDF 结合了词频 (TF) 和逆文档频率 (IDF)。向量元素 $d_{我,j}$ 表示为：

$$d_{我,j}=tf_{我,j}\times以色列国防军_{我},\qquad (2)$$

在哪里 $tf_{我,j}$ 是的频率*我*中的第 - 项*j*第 - 个文档和*以色列国防军*$_{我}$定义为：

$$以色列国防军_{我}=日志\ \frac{|D|}{|\{d:\ t_{我}\varepsilon d\}|},\qquad (3)$$

在哪里*|D|*是文档总数，$|\{d:\ t_{我}\varepsilon d\}|$ 表示包含 term 的文档数量 $t_{我}$。

我们将所有文档视为语料库*V*。一个新的查询*问*以同样的方式表示为向量。然后是相似度(1)查询之间*问*和文件*D*如下：

$$相似(D_j, Q)=\frac{\sum_{我=1}^{V} d_{我,j}*q_{我}}{\sqrt{\sum_{H=1}^{V}(d_{h,j})^2}*\sqrt{\sum_{k=1}^{V}(q_k)^2}},\qquad (4)$$

在哪里 $d_{我,j}$ 是文档的一维向量 D_j,和 $q_{我}$ 是查询的一维向量*问*。在T种族有趣，一个查询*问*代表一个未标记的工件，而 D_j 表示*j*-th 标记的工件。两个都*问*和 D_j 来自相同类型的工件（即源或目标）。

2）对比学习： CL是无监督学习，学习没有标签的表示函数，目的是使相似样本更接近，使不相似样本更远离[30]。

图3显示了CL预测同类型软件工件相似度的过程。首先，由于缺乏标签，数据的一些属性被用来生成伪标签作为训练数据（如图3（a）所示）。目的是让模型知道哪些数据样本是相似的。为了获得工件的这种对比表示，我们使用 CoEST 数据 [22]、[32]-[40]，它提供了来自多个项目的真实可追溯性链接。然后，生成的伪标记样本通过 GloVe [41] 编码为向量，进一步输入 Siamese 长短期记忆（LSTM）神经网络 [42]，如图 3（b）和（c）所示）。最后，对比损失函数用于学习数据的对比表示，如图 3(d) 所示。各部分的详细说明如下。

训练数据生成。 CL 的一个关键问题是生成有意义的训练数据，即一组配对示例

{（$X_{我}$，$X_{我}$）}$_{我=1}$表明$X_{我}$和$X_{+\ \ \ \ \ \ \ \ \ \ 我}$语义相关文物。在T种族FUN，具有相同链接目标工件的源工件被视为相似样本（即正样本）。相比之下，不同的样本（即负样本）是从未连接到相同目标伪影的源伪影生成的。

我们使用一个$s \times s$矩阵来记录源工件之间语义相似的关系，其中s是源工件的总数。元素$r_{我,j}$矩阵中的关系表示*我*第-个源工件和*j*-th 源工件。$r_{我,j}$=1表示这两个工件在语义上相似（即具有共同的目标工件），而$r_{我,j}$=0表示不相似（即，没有共同的目标工件）。为了生成这样的矩阵，我们遍历标记的可追溯性链接。链接到同一目标工件的源工件被分配到同一组。总组数为t，它等于目标工件的总数。如果至少有一组出现两个源伪影，则将它们视为正对样本，即r=1;如果两个源工件没有出现在同一组中，则它们被视为负面情况，即r=0。

接下来，我们使用*itertools.combinations*(),用于笛卡尔积 [43] 的 Python 函数，用于按组合的位置顺序对源工件进行排序，而无需重复元素，例如 elements*{A B C D}*生成组合（*甲*、*乙*),(一、*三*)，（*广告*),(*乙*、*丙*),(*乙*、*丁*），和（*光盘*）。然后，从上述相似关系矩阵中查询两个源工件之间的关系，以生成CL的训练数据。

*编码。*软件工件的文本被视为非结构化特征。文本中的每个单词都分配有一个唯一的索引，并且工件的文本表示为单词索引序列。GloVe [41] 将单词编码为向量表示，然后通过按索引顺序用向量填充矩阵的每一行来形成单词嵌入矩阵。接下来，将单词嵌入矩阵作为嵌入层的权重输入 Siamese 网络，其中句子的每个单词找到其向量表示。

*连体 LSTM。*对比学习需要深度学习模型来捕获相似和不相似数据点的潜在语义关系。在T种族有趣的是，我们构建了一个 Siamese LSTM 网络来预测同一类型的软件工件之间的相似关系。如图 3(c) 所示，该网络由两个相同的 LSTM 组成。

假设一个工件的文本被转换为一系列词嵌入向量A={$w_1, w_2, . . . , w_{氮}$}，在哪里$w_{我}$是的嵌入表示*我*这段文本中的第$_{物间}$一个单词，长度为*氮*$_{时间}$。这$w_{我}$进入LSTM的时间t表示为X_t，如图4所示。输出H_tLSTM 在每个时间步的值取决于输入 X_t在当前时间步，输出t_{t-1}在前一个时间步（即短期记忆单元）和长期记忆C网络（即长期记忆单元）。LSTM 使用一系列“门”控制数据序列中的信息如何进入、存储和离开网络。

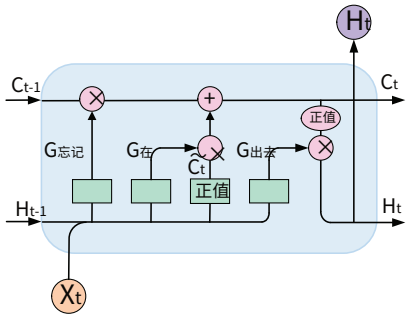

图 4. LSTM 的结构。

首先，通过遗忘门$G_{忘记}$决定忘记不必要的信息：

$$G_{忘记}（t）=\sigma（瓦_F * X_t + U_F * H_{t-1}）\tag{5}$$

然后，一个新的内存更新向量C_t根据短期记忆单元生成H_{t-1}在上一个时间步，以及输入X_t在当前时间步。LSTM 使用输入门$G_{在}$选择要记住的信息并将其添加到最后一个长期记忆单元C_{t-1}菲经过遗忘门过滤，并将其更新为新的长期记忆单元C_t：

$$G_{在}（t）=\sigma（瓦_{我} * X_t + U_{我} * H_{t-1}），\tag{6}$$

$$C_t = 正值（瓦_C * X_t + U_C * H_{t-1}），\tag{7}$$

$$C_t = G_{忘记} * C_{t-1} + G_{在} * C_t。\tag{8}$$

最后，输出门$G_{出去}$在当前时间步选择与当前任务相关的信息来生成输出H_t：

$$G_{出去}（t）=\sigma（瓦_{哦} * X_t + U_{哦} * H_{t-1}），\tag{9}$$

$$H_t = G_{出去} * 正值（C_t）。\tag{10}$$

在方程(5)-(10)中，$瓦$和U是权重矩阵，σ是 sigmoid 函数，并且*正值*是激活函数。

LSTM输出最终的隐藏状态来表示句子的语义信息。因此，非结构化特征表示为n维向量H（n是 LSTM 中隐藏单元的数量）。在我们的 Siamese LSTM 模型中，我们使用曼哈顿距离来衡量两个向量的相似度，定义为：

$$相似（H_1，H_2）= 经验值（-\ \| H_1 - H_2 \|_1），\tag{11}$$

在哪里H_1和H_2是两个软件工件的向量，*经验值*（。）标准化 0 和 1 之间的距离值。

*对比损失。*对比损失是训练目标，使相似样本更接近，不同样本更远[44]。对比损失定义为：

$$L=(1-y)\ *\ *_2+y^*最大限度(0,m-\hat{y})_2, \quad (12)$$

在哪里*米*是一个超参数，定义不同样本之间的下界距离。在T种族乐趣，*米*设置为 1。*y*和*ŷ*是一对两个工件的真实标签和预测标签（曼哈顿距离）$A_{我}$和A_j,分别。如果两个工件不相似（*y*=0），方程(12)最小化它们的预测值*ŷ*; 否则它会最小化*最大限度*(0, *m* − *ŷ*) 2，即最大化其预测值*ŷ*。

B. 未标记数据标记

通过上述每种方法对未标记和标记的工件进行相似性预测后，我们有两个按降序排列的相似性列表，分别针对源工件和目标工件。号码 （*氮*）从未标记数据新生成的链接按比例调整*ρ*（用于训练的原始标记链接总数的20%或50%。

为了标记未标记的工件，T种族FUN 逐一遍历两个相似度列表中的配对项。对于每次迭代， $SS_{我}$（的相似度*我*-源工件列表中的第一项）和$英石_j$（的相似度*j*- 目标工件列表中的第一项）进行比较以选择更相似的对，其中*我*和*j*从索引 0 开始（即从上到下）。例如，如果$SS_{我}$大于$英石_j$,这*我*第-个源工件对（S_p, S_q）选择进行进一步标记，其中S_p是带有 ID 的未标记源工件*p*和S_q是带有 ID 的标记源工件*q*。新链接是从以下位置生成的S_p到链接的目标工件S_q。否则，将从未标记的目标工件生成新链接。在本次迭代中生成新的标记链接后，索引 *我*索引增加一*J*下次比较时保持不变。当新标记的链接之和大于或等于时，循环将终止*氮*。

请注意，由于一个标记的工件可能具有多个链接对象，因此最后一次迭代中生成的链接的累积数量可能大于*氮*。此外，由于两个（或更多）标记的工件可能具有相同的链接工件。一个未标记的工件和两个标记的工件之间的对在排序列表中可能都具有高度相似性，从而生成重复的标记对，需要进一步消除。

这样，T种族FUN 将仅选择高度相似的源或目标工件对来生成未标记工件和标记工件的链接对象之间的链接。

C. TLR 评估

如图1（c）所示，所有新生成的链接与原始标记链接一起添加到训练集中。验证（如果需要）和测试数据样本仅从标记的链接中划分。时间种族FUN 集成了两种最新的基于 DL 的模型，即 T-BERT [27] 和 TNN [26]。T-BERT使用双向语言模型BERT（Bi Direction Encoder Representations from Transformers）[45]，它具有更丰富的上下文信息。T-BERT包括三个实现阶段：预训练和

表二

C收集C氧东部时间的阿塔塞特CLT下雨

项目	源神器		目标神器		# 链接
	类型	# 文物	类型	# 文物	
阿尔贝格特	要求	17 号	代码	55	54
中科院	要求	116	代码	1,064	第587章
CM1	高要求 22		要求低	53	45
电子ANCI	用例	140	代码	55	第567章
易诊所	多类型	160	多类型	160	1,618
EBT	要求	41	代码	50	98
易游	用例	58	代码	116	308
甘特	高要求 17		要求低	69	68
健康保险流通与责任法案	要求	10	技术保障 1,891		243
破冰活动	要求	201	UML 类	73	第457章
输液泵	要求	126	成分	21	131
我相信	要求	131	代码	第367章	第534章
亭	要求	178	流程	178	1,951
SMOS	用例	67	代码	100	1,044
沃克	要求	63	要求	89	136

表三

C收集D的阿塔塞特时间种族趣趣估值

项目	源神器	目标神器	# 链接
烧瓶	3,715	4,008	第752章
PGCLI	1,197	2,189	第529章
喀拉斯	4,811	5,349	第552章

使用不同的训练数据源对 T-BERT 模型进行中间训练，并在实际项目中微调 TLR 任务。TNN 使用结合词嵌入和循环神经网络（RNN）的追踪网络来恢复软件工件的追溯链接。

时间种族FUN 是一个灵活的框架，能够使用未标记的数据来训练任何需要标记数据的监督 TLR 方法（不限于 T-BERT 和 TNN）。

四．乙实验SETUP 和乙估值

A. 数据收集和汇总

*CL 训练数据集。*CL 的训练数据是从 CoEST [46] 提供的 15 个数据集中收集的，如表 II 所示。这些数据集包括不同类型的可追溯性链接，例如需求和代码、用例和类、高级和低级需求。Co-EST 数据有助于我们探索软件工件之间语义相似的关系。

*数据集为*时间种族乐趣 *评估。*用于评估 T 的标记数据集种族FUN 是从[27]收集的，包括三个开源 GitHub 项目，即 Flask、Pgcli 和 Keras，如表 III 所示。他们的工件类型是错误报告和错误修复提交。经过 T-BERT [27] 的评估，跟踪链接是从问题到提交，其粒度是文件级别。此外，还计算未标记和标记问题之间以及未标记和标记提交之间的相似性。

评估 T种族FUN，我们使用了5折分层交叉验证，即将每个项目收集的标记数据集分为5折，其中4折用于训练，剩余1折用于测试。四倍数据样本分割出的验证集的比例为0.2，即20%训练集用于验证。

新生成链接的数量由比例决定*p*超过剩余训练集中标记链接的数量。在我们的实验中，T选择了5个比例，即5%、20%、50%、80%和110%。种族FUN的评价。表IV示出了通过两种相似性预测方法（即VSM和CL）生成的链接的数量。请注意，由于源和目标对的相似性不同，因此从源和目标工件生成的链接数量也不同。例如，对于 Flask 源和目标工件，CL 对前 200 个最相似工件预测的相似性范围分别为 [0.68, 1] 和 [0.99, 1]。

*数据预处理。*收集的数据集通过以下三个步骤进行清理和处理。(1) 单词标记化：从工件中提取的文本首先被划分为单词流；(2)停用词、标点符号去除：进一步去除停用词、数字、标点符号；(3) 单词标准化：单词标记被转换为小写。

B. 实施细节

*VSM 的设置。*评估数据集（表 III）中的所有源工件和目标工件都用作 VSM 的语料库。两个工件之间的相似度是通过它们的向量通过余弦相似度来计算的。

*CL 的设置。*CL的训练参数如下：epoch数为100；批量大小为 64；隐藏单元的数量*n*在LSTM网络中为50；训练与测试分割的比例为0.2，从训练样本中分割出20%的样本作为验证集。

*TLR 方法的设置。*我们评估了两种最先进的 TLR 方法，即 T-BERT 和 TNN，如第 2 节中所述。III-C。T-BERT的训练参数如下：epoch数为400；批量大小为 4；学习率为 0.00004；网络架构是更快的SIMAMESE。另外，TNN的训练参数如下：epoch数为1000；批量大小为1；隐藏单元的数量*n* 是 60；最大序列长度为80；学习率为0.0001；网络模型是GRU。

*实验环境。*T的开发环境种族FUN如下：Python 3.8、TensorFlow 2.6.0、Keras 2.6.0。T的所有实验种族FUN在配备Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz、62GB内存和NVIDIA GeForce RTX 2080Ti GPU的云服务器上进行。

*C。*时间种族乐趣*评估*

*研究问题（RQ）。*我们的评估旨在回答以下三个 RQ：

- 问题一：不能种族提高 TLR 性能很有趣吗？
- 问题2：T中使用的不同相似度预测方法有什么影响种族对 TLR 性能感兴趣吗？
- 问题3：T生成的不同大小的新标记链接有什么影响种族对 TLR 性能感兴趣吗？

*评估指标。*评估 T 的指标种族乐趣如下：

表四

氮数量氮埃沃利L阿贝LEDL墨水来源时间种族乐趣

项目	原来的	%	向量空间模型		化学发光	
			来源	目标	来源	目标
烧瓶	第480章	5%	6	18	4	20
		20%	6	90	4	92
		50%	6	234	4	236
		80%	18	第366章	5	第379章
		110%	43	第482章	18	510
PGCLI	第338章	5%	0	16	0	16
		20%	0	67	0	67
		50%	0	169	14	155
		80%	0	270	104	166
		110%	3	第368章	198	173
喀拉斯	第352章	5%	6	11	5	12
		20%	48	22	22	48
		50%	137	39	92	84
		80%	235	46	162	119
		110%	第333章	54	233	154

- F 分数：F 分数是精确率和召回率的调和平均值：

$$F_\beta = \frac{(1+\beta_2)\ *精确*记起}{\beta_2 *精确+ 记起}, \qquad (13)$$

在哪里β是一个积极的真实因素，因此考虑了召回率β倍与精度同样重要。什么时候 β=1（即F1-score），精度和召回率赋予相同的权重。什么时候β=2（即F2-score），召回率比精确率更重要。在我们的实验中，我们使用F1-score和F2-score进行评估。

- 平均精度 (MAP)：平均精度（AveP）是指不同召回率下最大精度值的平均值。对于每个源工件 *问*，AveP是根据所有的位置计算的*n*排名中的相关目标工件。然后通过平均 AveP 值来计算 MAP。我们使用 MAP@3，即仅对工件进行排名（$秩_{我}$）排名前 3 位的贡献 *平均P*：

$$平均P@3 = \frac{\Sigma_n}{n} \frac{我磷}{}, \quad P = \begin{cases} 磷@我 & 如果秩_{我} \ 3 \\ 0, & 否则 \end{cases}, \qquad (14)$$

$$地图@3 = \frac{1\Sigma^{问}}{问_{q=1}} 平均P@3。 \qquad (15)$$

电压结果和A分析

本节介绍并讨论 T 的评估结果种族乐趣。表 V、VI 和 VII 分别显示了 T-BERT 在 F1 分数、F2 分数和 MAP 方面的性能，而 TNN 的结果分别在表 VIII、IX 和 X 中给出。

*A.RQ1：可以*时间种族乐趣*提高 TLR 性能？*

RQ1 旨在调查我们的 T 是否种族FUN可以提高TLR的性能。我们对 5 倍分层交叉验证获得的结果进行平均并计算改进，以比较 T-BERT 和

表五

F1-分数T-BERT T下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)					随机 (50%)
烧瓶	1	0.632	0.648	0.654	0.672	0.684	0.713	0.706	0.789	0.813	0.783	0.783	0.582
	2	0.658	0.699	0.678	0.675	0.687	0.727	0.727	0.762	0.753	0.772	0.771	0.589
	3	0.617	0.613	0.598	0.669	0.642	0.701	0.700	0.699	0.701	0.767	0.728	0.506
	4	0.679	0.713	0.667	0.698	0.679	0.725	0.721	0.789	0.752	0.788	0.751	0.629
	5	0.636	0.660	0.672	0.694	0.664	0.725	0.713	0.756	0.768	0.779	0.783	0.572
	平均。（改进。）	0.644	0.667 (4%)	0.654 (2%)	0.682 (6%)	0.671 (4%)	0.718 (12%)	0.713 (11%)	0.759 (18%)	0.757 (18%)	0.778 (21%)	0.763 (19%)	0.576 (-11%)
PGCLI	1	0.735	0.715	0.720	0.758	0.790	0.760	0.767	0.810	0.751	0.806	0.720	0.620
	2	0.733	0.725	0.714	0.755	0.753	0.747	0.756	0.805	0.687	0.829	0.677	0.629
	3	0.757	0.730	0.794	0.794	0.769	0.761	0.792	0.819	0.736	0.778	0.742	0.651
	4	0.693	0.735	0.681	0.726	0.697	0.751	0.728	0.798	0.685	0.821	0.663	0.640
	5	0.767	0.780	0.759	0.802	0.777	0.808	0.800	0.816	0.709	0.851	0.684	0.648
	平均。（改进。）	0.737	0.737 (0%)	0.734 (0%)	0.767 (4%)	0.757 (3%)	0.765 (4%)	0.769 (4%)	0.810 (10%)	0.714 (-3%)	0.817 (11%)	0.697 (-5%)	0.638 (-13%)
喀拉斯	1	0.938	0.916	0.914	0.940	0.927	0.877	0.914	0.871	0.914	0.853	0.883	0.892
	2	0.950	0.941	0.945	0.900	0.932	0.874	0.922	0.883	0.877	0.878	0.858	0.890
	3	0.953	0.940	0.959	0.943	0.943	0.938	0.935	0.916	0.911	0.841	0.882	0.906
	4	0.926	0.932	0.927	0.943	0.925	0.941	0.897	0.887	0.912	0.859	0.912	0.827
	5	0.960	0.964	0.954	0.955	0.963	0.919	0.933	0.920	0.935	0.900	0.910	0.895
	平均。（改进。）	0.945	0.939 (-1%)	0.940 (-1%)	0.936 (-1%)	0.938 (-1%)	0.910 (-4%)	0.920 (-3%)	0.895 (-5%)	0.910 (-4%)	0.866 (-8%)	0.889 (-6%)	0.882 (-7%)

表六

F2-分数T-BERT T下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)					随机 (50%)
烧瓶	1	0.650	0.612	0.638	0.639	0.659	0.668	0.690	0.760	0.787	0.781	0.774	0.562
	2	0.683	0.675	0.680	0.702	0.700	0.744	0.746	0.776	0.768	0.779	0.788	0.615
	3	0.604	0.603	0.580	0.638	0.623	0.684	0.675	0.707	0.711	0.795	0.740	0.488
	4	0.665	0.696	0.669	0.709	0.698	0.719	0.721	0.805	0.751	0.797	0.771	0.628
	5	0.663	0.683	0.694	0.705	0.681	0.724	0.722	0.797	0.791	0.791	0.777	0.557
	平均。（改进。）	0.653	0.654 (0%)	0.652 (0%)	0.679 (4%)	0.672 (3%)	0.708 (8%)	0.711 (9%)	0.769 (18%)	0.762 (17%)	0.789 (21%)	0.770 (18%)	0.570 (-13%)
PGCLI	1	0.747	0.749	0.750	0.769	0.790	0.782	0.777	0.826	0.730	0.822	0.695	0.627
	2	0.757	0.755	0.742	0.756	0.743	0.782	0.783	0.839	0.680	0.850	0.677	0.641
	3	0.808	0.780	0.821	0.814	0.810	0.806	0.807	0.838	0.721	0.810	0.744	0.672
	4	0.701	0.724	0.711	0.748	0.719	0.758	0.749	0.788	0.712	0.842	0.661	0.620
	5	0.777	0.812	0.784	0.810	0.802	0.809	0.807	0.822	0.729	0.856	0.702	0.668
	平均。（改进。）	0.758	0.764 (1%)	0.762 (0%)	0.779 (3%)	0.773 (2%)	0.787 (4%)	0.785 (4%)	0.823 (9%)	0.714 (-6%)	0.836 (10%)	0.696 (-8%)	0.646 (-15%)
喀拉斯	1	0.943	0.927	0.938	0.936	0.941	0.906	0.924	0.895	0.884	0.886	0.868	0.897
	2	0.944	0.946	0.942	0.892	0.924	0.874	0.930	0.886	0.892	0.889	0.893	0.880
	3	0.937	0.932	0.951	0.927	0.938	0.915	0.925	0.919	0.897	0.855	0.870	0.887
	4	0.948	0.950	0.954	0.961	0.946	0.939	0.918	0.897	0.905	0.871	0.903	0.833
	5	0.967	0.973	0.959	0.957	0.966	0.919	0.950	0.940	0.939	0.916	0.927	0.893
	平均。（改进。）	0.948	0.946 (0%)	0.949 (0%)	0.935 (-1%)	0.943 (-1%)	0.911 (-4%)	0.929 (-2%)	0.907 (-4%)	0.903 (-5%)	0.883 (-7%)	0.892 (-6%)	0.878 (-7%)

通过 T 添加新标记数据之前和之后的 TNN种族- 有趣（即VSM和CL）。此外，还研究了通过随机生成标记链接训练的性能。请注意，对于随机选择，我们将源工件和目标工件的配对列表组合在一起，然后随机地从其中随机洗牌一对，而不考虑相似性。新链接的生成方式与第 2 节中描述的相同。III-B。此过程停止，直到新标记的链接数量大于或等于给定大小（即用于训练的原始标记数据的 50%）。

从表V到表X可以看出，对于Flask和Pgcli数据集，T-BERT和TNN的所有三个评估指标（即F1-score、F2-score和MAP）的分数在经过之后均显着提高。通过 VSM 和 CL 添加新标记的链接（即 T种族乐趣）。对于 T-BERT，Flask 数据集的 F1-score、F2-score 和 MAP 的最大改进分别为 21%、21% 和 19%，而 Pgcli 数据集的结果分别为 11%、10% 、和 11% ，分别。同样，在添加 T 生成的新标记链接后，TNN 在这两个数据集上的性能也有了显着的提高种族乐趣。例如，F1-score、F2-score 和 MAP 方面的最大改进是 Flask 和 Pgcli 数据集分别为 1,088%、555%、1,091% 和 719%、322%、733%。

另外，对于Keras数据集，T-BERT和TNN在添加新链接后的性能提升种族乐趣是不同的。对于T-BERT，与原始训练集获得的结果相比，评估分数略有下降。例如，Keras 使用原始标记数据集的 F1 分数为 0.945，而 T 的最佳结果种族乐趣是 0.940。然而，对于 TNN，性能有所提高，F1-score、F2-score 和 MAP 的最佳改进分别为 109%、41% 和 146%，尽管这种改进低于 Flask 和 Pgcli 数据集的改进，如表VIII、IX和X所示。注意，与[27]中报告的结果类似，使用原始标记数据的TNN性能极低，例如，TNN在Flask、Pgcli和Keras 数据集分别为 0.033、0.036 和 0.032。这是因为 TNN 需要大量的标记数据才能获得良好的性能[27]。三个数据集的原始标记数据非常小，因此导致性能较差。

此外，对于随机选择，除了 Pgcli 数据集上关于 F1-score 和 MAP 的 TNN 外，所有的分数

表七

地图的T-BERT T下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)					随机 (50%)
烧瓶	1	0.713	0.731	0.720	0.732	0.772	0.798	0.779	0.836	0.841	0.863	0.860	0.630
	2	0.733	0.756	0.743	0.768	0.786	0.800	0.832	0.845	0.849	0.848	0.859	0.657
	3	0.688	0.684	0.672	0.737	0.693	0.758	0.773	0.787	0.812	0.861	0.849	0.558
	4	0.741	0.769	0.768	0.784	0.797	0.800	0.820	0.862	0.852	0.880	0.854	0.687
	5	0.756	0.760	0.790	0.790	0.767	0.818	0.814	0.846	0.869	0.883	0.863	0.639
	平均。（改进。）	0.726	0.740 (2%)	0.739 (2%)	0.762 (5%)	0.763 (5%)	0.795 (9%)	0.804 (11%)	0.835 (15%)	0.845 (16%)	0.867 (19%)	0.857 (18%)	0.634 (-13%)
PGCLI	1	0.803	0.819	0.797	0.833	0.836	0.847	0.825	0.869	0.797	0.890	0.770	0.690
	2	0.803	0.846	0.813	0.810	0.846	0.863	0.860	0.895	0.770	0.918	0.775	0.695
	3	0.849	0.847	0.862	0.863	0.869	0.865	0.866	0.898	0.792	0.869	0.846	0.734
	4	0.737	0.767	0.789	0.775	0.802	0.830	0.803	0.851	0.781	0.885	0.745	0.656
	5	0.865	0.876	0.873	0.894	0.894	0.884	0.887	0.905	0.873	0.938	0.821	0.748
	平均。（改进。）	0.811	0.831 (2%)	0.827 (2%)	0.835 (3%)	0.849 (5%)	0.858 (6%)	0.848 (5%)	0.884 (9%)	0.803 (-1%)	0.900 (11%)	0.791 (-2%)	0.705 (-13%)
喀拉斯	1	0.973	0.962	0.977	0.965	0.949	0.973	0.935	0.955	0.946	0.940	0.902	0.940
	2	0.967	0.964	0.973	0.935	0.973	0.958	0.968	0.946	0.946	0.934	0.967	0.917
	3	0.948	0.973	0.961	0.955	0.973	0.945	0.948	0.959	0.933	0.930	0.895	0.950
	4	0.986	0.973	0.964	0.977	0.959	0.973	0.950	0.959	0.955	0.939	0.944	0.905
	5	0.977	0.977	0.977	0.977	0.982	0.962	0.986	0.982	0.971	0.971	0.973	0.964
	平均。（改进。）	0.970	0.970 (0%)	0.970 (0%)	0.962 (-1%)	0.967 (0%)	0.926 (-1%)	0.957 (-1%)	0.960 (-1%)	0.950 (-3%)	0.943 (-3%)	0.936 (-3%)	0.935 (-4%)

表八

F1-分数田纳西州下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)					随机 (50%)
烧瓶	1	0.031	0.021	0.037	0.060	0.061	0.130	0.157	0.232	0.310	0.287	0.398	0.027
	2	0.025	0.051	0.025	0.071	0.074	0.182	0.181	0.211	0.347	0.331	0.384	0.020
	3	0.043	0.025	0.023	0.048	0.046	0.115	0.102	0.153	0.292	0.244	0.346	0.015
	4	0.028	0.026	0.037	0.090	0.088	0.187	0.189	0.253	0.355	0.357	0.423	0.026
	5	0.036	0.032	0.054	0.046	0.052	0.130	0.190	0.202	0.349	0.313	0.410	0.020
	平均。（改进。）	0.033	0.031 (-6%)	0.035 (7%)	0.063 (91%)	0.064 (95%)	0.149 (351%)	0.164 (396%)	0.210 (537%)	0.331 (902%)	0.306 (828%)	0.392 (1,088%)	0.022 (-35%)
PGCLI	1	0.036	0.042	0.038	0.087	0.081	0.142	0.231	0.255	0.260	0.380	0.252	0.031
	2	0.035	0.030	0.033	0.058	0.048	0.168	0.154	0.245	0.125	0.288	0.178	0.038
	3	0.039	0.047	0.049	0.100	0.083	0.164	0.169	0.240	0.176	0.292	0.215	0.050
	4	0.031	0.032	0.033	0.053	0.055	0.091	0.154	0.184	0.165	0.238	0.146	0.035
	5	0.040	0.083	0.068	0.110	0.105	0.174	0.198	0.241	0.186	0.277	0.222	0.040
	平均。（改进。）	0.036	0.047 (30%)	0.044 (23%)	0.082 (127%)	0.074 (107%)	0.148 (311%)	0.181 (403%)	0.233 (547%)	0.182 (407%)	0.295 (719%)	0.203 (463%)	0.039 (8%)
喀拉斯	1	0.029	0.033	0.041	0.064	0.040	0.088	0.043	0.090	0.041	0.091	0.039	0.029
	2	0.041	0.026	0.048	0.030	0.029	0.037	0.039	0.031	0.022	0.041	0.036	0.033
	3	0.027	0.078	0.026	0.085	0.031	0.108	0.037	0.109	0.030	0.104	0.045	0.034
	4	0.029	0.035	0.043	0.030	0.062	0.045	0.059	0.052	0.052	0.053	0.058	0.023
	5	0.033	0.031	0.031	0.051	0.028	0.052	0.034	0.025	0.021	0.046	0.027	0.021
	平均。（改进。）	0.032	0.041 (27%)	0.038 (18%)	0.052 (63%)	0.038 (19%)	0.066 (106%)	0.042 (33%)	0.061 (92%)	0.033 (4%)	0.067 (109%)	0.041 (28%)	0.028 (-13%)

T-BERT 和 TNN 的评估指标都比原始标记数据训练的结果差。例如，T-BERT 在 Flask 数据集上的 F1-score、F2-score 和 MAP 分别下降了 11%、13% 和 13%。该结果是预期的，因为随机选择更有可能引入错误标记的链接，这可能会对模型训练产生负面影响。

对 RQ1 的回答：时间种族FUN可以显着提高TLR性能。时间种族FUN 能够捕获未标记和标记工件之间的语义相似关系，从而为 TLR 模型训练生成有效的新标记链接。

*B. RQ2：不同相似度预测方法的影响是什么*时间种族乐趣*关于 TLR 性能？*

在此 RQ 中，我们评估了 T 中使用的 VSM 和 CL 的影响种族TLR 性能很有趣。对于 T-BERT，如表 V、VI 和 VII 所示，VSM 比 CL 更能测量未标记和标记伪影之间的相似性。例如，VSM 在 Flask 数据集上的 F1-score 的最佳改进为 21%，而 CL 的最佳改进为 19%。在Pgcli数据集上，VSM和CL的性能提升差异非常大，例如：

VSM 和 CL 的 F1 分数的最佳改进分别为 11% 和 4%。

然而，对于表 VIII、IX 和 X 中所示的 TNN，在 Flask 数据集上使用 CL 比使用 VSM 更好。CL 的 F1 分数的最佳性能提升为 1,088%，而 VSM 只实现了 828% 的提升。这样的情况在Pgcli数据集上得到了改变。VSM 获得了 F1 分数 719% 的最佳改进，而 CL 在最佳情况下仅将指标改进了 463%。

对 RQ2 的回答：T 中使用 VSM 和 CL 的性能改进种族对于不同的 TLR 方法和数据集，FUN 是不同的。需要根据具体的TLR方法和数据集选择合适的相似度预测方法，以更好地提高性能。对于问题和提交的 TLR 任务，建议用户使用 VSM 和 CL。

*C. RQ3：生成的不同大小的新标记链接有何影响*时间种族乐趣*关于 TLR 性能？*

在这个实验中，我们生成不同大小的新标记链接，即用于训练的原始标记链接的5%、20%、50%、80%和110%，以评估它们对TLR性能的影响。

表九

F2-分数田纳西州下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)	随机 (50%)
烧瓶	1	0.050	0.040	0.059	0.071	0.066	0.125	0.162	0.198	0.250	0.234	0.334	0.038
	2	0.048	0.048	0.049	0.072	0.077	0.152	0.175	0.197	0.282	0.291	0.297	0.040
	3	0.046	0.038	0.042	0.057	0.063	0.120	0.108	0.156	0.216	0.220	0.252	0.036
	4	0.043	0.043	0.045	0.069	0.072	0.161	0.161	0.217	0.298	0.313	0.353	0.048
	5	0.051	0.051	0.054	0.075	0.068	0.125	0.166	0.206	0.279	0.278	0.335	0.043
	平均。（改进。）	0.048	0.044 (-8%)	0.050 (4%)	0.069 (43%)	0.069 (44%)	0.137（185%）	0.154（222%）	0.195（306%）	0.265（452%）	0.267 (457%)	0.314（555%）	0.041 (-15%)
PGCLI	1	0.059	0.071	0.066	0.110	0.100	0.140	0.195	0.220	0.215	0.295	0.204	0.055
	2	0.067	0.055	0.065	0.059	0.066	0.138	0.126	0.194	0.131	0.255	0.145	0.053
	3	0.050	0.067	0.062	0.077	0.097	0.149	0.152	0.198	0.193	0.232	0.169	0.063
	4	0.054	0.060	0.053	0.070	0.073	0.119	0.123	0.186	0.142	0.231	0.129	0.052
	5	0.064	0.064	0.079	0.107	0.099	0.166	0.178	0.192	0.176	0.232	0.171	0.056
	平均。（改进。）	0.059	0.063 (7%)	0.065（10%）	0.085 (43%)	0.087 (47%)	0.142（141%）	0.155（162%）	0.198（236%）	0.171（191%）	0.249 (322%)	0.164（177%）	0.056（-5%）
喀拉斯	1	0.047	0.062	0.052	0.061	0.074	0.072	0.068	0.077	0.058	0.087	0.066	0.054
	2	0.065	0.054	0.071	0.059	0.053	0.054	0.047	0.066	0.049	0.068	0.054	0.061
	3	0.054	0.073	0.055	0.081	0.051	0.104	0.053	0.078	0.054	0.080	0.050	0.055
	4	0.050	0.050	0.061	0.053	0.091	0.074	0.050	0.061	0.051	0.073	0.052	0.046
	5	0.048	0.056	0.047	0.049	0.050	0.054	0.046	0.047	0.044	0.065	0.044	0.049
	平均。（改进。）	0.053	0.059 (11%)	0.057 (8%)	0.061 (14%)	0.064 (20%)	0.072 (35%)	0.053 (0%)	0.066 (24%)	0.051 (-3%)	0.075 (41%)	0.053 (0%)	0.053 (0%)

表十

地图的田纳西州下雨了氮埃沃利L阿贝LEDDATAG能源由时间种族有趣（VSM和CL),和右安多姆S选举

项目	折叠	原来的	VSM (5%) CL (5%)	VSM (20%) CL (20%)	VSM (50%) CL (50%)	VSM (80%) CL (80%)	VSM (110%) CL (110%)	随机 (50%)
烧瓶	1	0.031	0.020	0.032	0.063	0.049	0.118	0.155	0.198	0.257	0.243	0.326	0.025
	2	0.010	0.034	0.017	0.052	0.085	0.147	0.163	0.178	0.290	0.286	0.321	0.009
	3	0.040	0.022	0.026	0.048	0.053	0.114	0.099	0.149	0.238	0.220	0.270	0.013
	4	0.024	0.013	0.039	0.069	0.072	0.160	0.167	0.204	0.302	0.318	0.354	0.026
	5	0.030	0.029	0.058	0.047	0.053	0.131	0.171	0.216	0.301	0.281	0.337	0.018
	平均。（改进。）	0.027	0.024 (-13%)	0.034 (27%)	0.056 (107%)	0.062（131%）	0.134（396%）	0.151 (459%)	0.189（600%）	0.278 (928%)	0.270 (899%)	0.322 (1,091%)	0.018 (-33%)
PGCLI	1	0.038	0.042	0.031	0.082	0.099	0.137	0.222	0.225	0.201	0.310	0.208	0.041
	2	0.041	0.014	0.027	0.042	0.044	0.135	0.121	0.186	0.123	0.236	0.149	0.050
	3	0.022	0.052	0.038	0.075	0.085	0.140	0.149	0.215	0.181	0.250	0.167	0.028
	4	0.017	0.014	0.041	0.047	0.058	0.090	0.112	0.168	0.134	0.206	0.132	0.039
	5	0.032	0.062	0.070	0.097	0.097	0.156	0.195	0.210	0.170	0.248	0.183	0.032
	平均。（改进。）	0.030	0.037 (23%)	0.041 (38%)	0.069 (129%)	0.077 (155%)	0.132 (339%)	0.160 (433%)	0.201（569%）	0.162 (439%)	0.250（733%）	0.168（459%）	0.038 (27%)
喀拉斯	1	0.027	0.014	0.026	0.038	0.038	0.063	0.050	0.050	0.035	0.071	0.039	0.027
	2	0.021	0.017	0.038	0.035	0.029	0.033	0.038	0.030	0.018	0.044	0.045	0.026
	3	0.023	0.067	0.026	0.062	0.036	0.085	0.032	0.086	0.023	0.076	0.036	0.021
	4	0.021	0.039	0.039	0.033	0.080	0.041	0.036	0.058	0.056	0.052	0.048	0.015
	5	0.026	0.017	0.024	0.036	0.023	0.062	0.044	0.023	0.003	0.052	0.027	0.009
	平均。（改进。）	0.024	0.031 (28%)	0.031 (28%)	0.041 (70%)	0.041 (72%)	0.057 (137%)	0.040 (67%)	0.049 (106%)	0.027 (13%)	0.059 (146%)	0.039 (63%)	0.020 (-18%)

对于T-BERT，如表五、六、七所示，对于Flask数据集，T-BERT的性能随着VSM和CL新标记链接的比例增加至110%而上升，即链接越多使用的性能越好。对于Pgcli数据集，VSM呈现出相同的趋势，即可以通过添加更多新标记的链接来提高性能。然而，随着新链路规模的增加，CL 实现的性能先增加后减少。Pgcli 数据集上 CL 的最佳比例是 50%。对于 TNN，如表 VIII、IX 和 X 所示，Flask 和 Pgcli 数据集上的性能均随着新标记链接数量的增加而增加。

另外，通过T添加新标记的链接后，性能在某些比例上变差种族比原来的更有趣。例如，T-BERT 在 CL 获得的 Pgcli 数据集上的性能下降了 80% 和 110%。对于Keras数据集，无论使用何种相似度预测方法，性能都会随着新标记链接比例的增加而下降。VSM和CL实现的最佳性能为5%，性能比原始性能稍低。Keras数据集上出现的这种性能下降可能是由以下原因引起的。原始标记数据足以训练一个好的T-BERT模型，即F1-score、F2-score和MAP已经分别达到0.945、0.948和0.970。改进的空间很小。

对 RQ3 的回答：一般来说，通过 T 添加更多标记链接可以提高 TLR 性能种族乐趣。然而，对于不同的 TLR 方法和数据集，新标记的链接的大小会极大地影响性能。因此，需要微调T标记的新链接的大小种族根据特定的 TLR 方法和数据集获得更好的结果很有趣。

*D. 局限性*时间种族乐趣

时间种族FUN 有两个限制。首先，新增链路的最佳数量无法一次性确定。例如，随着 Pgcli 数据集新链接大小的增加，CL 实现的性能先增加后减少。调整新增链接的比例是必要的。

二、T种族FUN 可能会导致性能下降。对于一些数据标签相对较好的项目，T种族- FUN 可能会导致性能下降，因为噪声数据将与来自未标记数据（例如 Keras 数据集）的新标记链接一起引入。

六．时间威胁到V活力

*对内部有效性的威胁。*T中使用的相似度预测方法种族乐趣是对内部有效性的主要威胁。为了减少这种威胁，我们引入了两种不同的方法，即 VSM 和 CL，来测量未标记和标记工件之间的相似性。为了训练 CL 模型，我们仅使用从源工件中提取的自然语言文本作为由类似 Word2Vec 的技术（例如 GloVe [41]）嵌入的特征来捕获语义相似的关系。可以通过使用代码嵌入技术[47]（例如，code2seq[48]、code2vec[49]和Flow2Vec[50]）作为主要包含源代码的工件的特征来改进相似性预测。结果，T种族FUN 的性能可以进一步提高，因为可以捕获更多相似的工件对来生成训练 TLR 模型的新链接。

*对外部有效性的威胁。*T 的推广种族- 乐趣是外部有效性的主要威胁。一、T中集成的TLR方法种族FUN 和可追溯性链接的评估数据集对外部有效性构成威胁。为了减少这种威胁，我们选择了两种最先进的基于深度学习的 TLR 方法，即 T-BERT [27] 和 TNN [26]，其中 T-BERT 是最新的方法，在小规模上表现出了优异的性能。数据集与 TNN 的比较。此外，为了评估T-BERT和TNN，我们使用[27]提供的相同数据集，即Flask、Pgcli和Keras进行评估。时间种族FUN 适用于任何需要标记数据的监督 TLR 方法。未来我们将集成更多的TLR方法，使用更多类型的溯源链接数据集来评估T种族乐趣。

*对建构有效性的威胁。*预测性能评估指标的选择可能会对构建有效性构成威胁。为了减少这种威胁，我们使用 F 分数和 MAP，它们与 T-BERT [27] 中使用的评估指标相同。

七．右兴高采烈瓦奥克

*TLR 深度学习。*在过去的几十年里，研究人员提出了各种 TLR 方法 [5]-[21]。最近，深度学习技术因其良好的性能而被应用于 TLR [22]-[27]。米尔斯 *等人。*[[22]提出TRAIL方法，该方法使用可追溯性链接来训练机器学习模型，以验证新软件工件之间的链接是否正确。阮 *等人。*[24]提出 DeepLink 使用词嵌入和 RNN 来恢复问题和提交之间的链接。学习问题和提交之间的语义关系。郭 *等人。*[26]提出 TraceNN（TNN），使用词嵌入来解决词汇不匹配的问题。词之间的语义关系通过词嵌入向量之间的线性关系来表示。然后，工件向量被输入 RNN 网络，以预测软件工件之间的可追溯性链接。林 *等人。*介绍Trace BERT（T-BERT）[27]，一个与BERT模型相结合的TLR框架。T-BERT 使用预训练的语言模型和迁移学习，有效地将从代码搜索问题中学到的知识迁移到 TLR 中。缓解了预训练数据不足的问题，提高了恢复链接的准确性。据我们所知，我们的工作首先研究了通过生成新标记的链接来利用现成的未标记数据的方法，以进一步增强 TLR 模型训练。

*对比学习。*对比学习（CL）是一种制定寻找相似和不相似数据样本任务的方法。它可以在没有标签的情况下将数据分类为相似和不同。CL已成功应用于计算机视觉[51]-[53]。陈 *等人。*[[30]提出了 SimCLR 方法，通过潜在空间中的对比损失来最大化同一数据示例的不同增强视图之间的一致性来学习表示，大大优于以前的自监督和半监督学习方法。最近，CL在软件工程中也有一些应用。布伊 *等人。*[[54]提出 Corder方法，通过CL识别相似和不同的代码片段，并使用大量未标记的源代码数据训练神经网络来识别语义等效的代码片段。在代码检索任务中，Corder 显着优于没有 CL 的模型。程 *等人。*[[55]提出了 ContraFlow，一种选择性但精确的对比价值流嵌入方法，用于静态检测软件漏洞。我们的论文首次将 CL 引入到 TLR 领域。我们使用 CL 来捕获未标记和标记工件之间的语义相似关系，旨在通过生成新标记的链接，用丰富的未标记数据来补充有限的训练数据。

八．C结论

本文介绍了T种族FUN，一种利用未标记数据增强的新型 TLR 框架。在T种族有趣的是，未标记和标记工件之间的语义相似关系首先由 VSM 和 CL 预测。然后，选择高度相似的工件对来生成未标记工件和标记工件的链接工件之间的链接。这些新标记的链接稍后会与原始标记数据一起添加到训练集中，用于 TLR 模型训练。我们对比评价了T种族FUN 在三个 GitHub 项目（包括 Flask、Pgcli 和 Keras）上拥有两个最先进的 TLR 模型（即 T-BERT 和 TNN）。我们还研究了 T 中使用的不同相似性预测方法的影响种族新生成的标记链接的 FUN 和大小对 TLR 性能的影响。我们的结果表明，T种族 FUN 通过利用未标记的数据有效地提高了 TLR 性能。我们的源代码和数据可在 https://github.com/TraceFUN 获取。

A致谢

感谢匿名审稿人提出的宝贵意见和建议。这项工作得到了国家自然科学基金项目62002163和61772055、江苏省自然科学基金项目BK20200441、新型软件技术国家重点实验室（南京大学）开放研究基金KFKT2020B20的部分支持，以及澳大利亚研究委员会授予 DP210101348。

右参考文献

[1] M. Grechanik、KS McKinley 和 DE Perry，“恢复和使用用例图到源代码的可追溯性链接”，载于*欧洲软件工程大会第六届联席会议暨 ACM SIGSOFT 软件工程基础研讨会 (ESEC/FSE) 论文集*，2007 年，第 95–104 页。

[2] B. Ramesh 和 M. Jarke，“走向需求可追溯性的参考模型”，*IEEE 软件工程汇刊*，卷。27、没有。1，第 58-93 页，2001 年。

[3] A. Tang、Y. Jin 和 J. Han，“用于设计可追溯性和推理的基于基本原理的架构模型”*系统与软件杂志*，卷。80，没有。6，第 918–934 页，2007 年。

[4] L. Naslavsky 和 DJ Richardson，“使用可追溯性支持基于模型的回归测试”，载于*第 22 届 IEEE/ACM 国际自动化软件工程 (ASE) 会议论文集*，2007 年，第 567–570 页。

[5] G. Antoniol、G. Canfora、G. Casazza、A. De Lucia 和 E. Merlo，“恢复代码和文档之间的可追溯性链接”*IEEE 软件工程汇刊*，卷。28、没有。10，第 970–983 页，2002 年。

[6] A. Marcus 和 JI Maletic，“使用潜在语义索引恢复文档到源代码的可追溯性链接”，载于*第 25 届国际软件工程会议 (ICSE) 论文集*。IEEE，2003 年，第 125–135 页。

[7] AD Lucia、F. Fasano、R. Oliveto 和 G. Tortora，“使用信息检索方法恢复软件工件管理系统中的可追溯性链接”，*ACM 软件工程和方法学汇刊*，卷。16、没有。4，第 13 页 - es，2007 年。

[8] X. Chen，“文档和源代码之间的追溯关系的提取和可视化”，载于*IEEE/ACM 国际自动化软件工程会议论文集 (ASE)*，2010 年，第 505–510 页。

[9] X. Chen 和 J. Grundy，“通过结合检索技术改进自动化文档以实现代码可追溯性”，载于*第 26 届 IEEE/ACM 国际自动化软件工程 (ASE) 会议论文集*。IEEE，2011 年，第 223-232 页。

[10] B. Dagenais 和 MP Robillard，“恢复 api 及其学习资源之间的可追溯性链接”，载于*第 34 届国际软件工程会议 (ICSE) 论文集*。IEEE，2012 年，第 47-57 页。

[11] N.阿里，Y.-G。Guéhéneuc 和 G. Antoniol，“Trustrace：挖掘软件存储库以提高需求可追溯性链接的准确性”
IEEE 软件工程汇刊，卷。39，没有。5，第 725–741 页，2012 年。

[12] N. Ali、F. Jaafar 和 AE Hassan，“利用历史共同变更信息进行需求追踪”，载于*第20届逆向工程工作会议（WCRE）论文集*。IEEE，2013 年，第 361–370 页。

[13] J.Guo、N.Monaikul、C.Plepel 和 J.Cleland-Huang，“迈向智能的特定领域可追溯性解决方案”，载于*第 29 届 ACM/IEEE 自动软件工程 (ASE) 国际会议论文集*，2014 年，第 755–766 页。

[14] A. Panichella、C. McMillan、E. Moritz、D. Palmieri、R. Oliveto、D. Poshyvanyk 和 A. De Lucia，“何时以及如何使用结构信息来改进基于 IR 的可追溯性恢复”，载于*第 17 届欧洲软件维护与再工程会议 (CSMR) 论文集*。IEEE，2013 年，第 199-208 页。

[15] A. Panichella、A. De Lucia 和 A. Zaidman，“基于 IR 的可追溯性恢复的自适应用户反馈”，载于*第八届 IEEE/ACM 软件和系统可追溯性 (SST) 国际研讨会论文集*。IEEE，2015 年，第 15-21 页。

[16] A. Bachmann、C. Bird、F. Rahman、P. Devanbu 和 A. Bernstein，“缺失的链接：错误和错误修复提交”，载于*第 18 届 ACM SIGSOFT 软件工程基础国际研讨会 (FSE) 论文集*，2010 年，第 97–106 页。

[17] R. Wu、H. Zhang、S. Kim 和 S.-C。Cheung，“重新链接：恢复错误和更改之间的链接”，载于*第 19 届 ACM SIGSOFT 研讨会和第 13 届欧洲软件工程基础会议 (ESEC/FSE) 论文集*，2011 年，第 15-25 页。

[18] A. Sureka、S. Lal 和 L. Agarwal，“应用 follegi-sunter (fs) 模型进行错误数据库和版本档案之间的可追溯性链接恢复”，载于*第十八届亚太软件工程会议（APSEC）论文集*。IEEE，2011 年，第 146–153 页。

[19] AT Nguyen、TT Nguyen、HA Nguyen 和 TN Nguyen，“用于恢复错误报告和修复之间链接的多层方法”
在*ACM SIGSOFT 第 20 届软件工程基础国际研讨会 (FSE) 论文集*，2012 年，第 1-11 页。

[20] T.-DB Le、M. Linares-Vásquez、D. Lo 和 D. Poshyvanyk，“Rclinker：利用丰富的上下文信息自动链接问题报告和承诺”，载于*IEEE 第 23 届程序理解国际会议 (ICPC) 论文集*。IEEE，2015 年，第 36-47 页。

[21] Y. Sun、Q. Wang 和 Y. Yang，“Frlink：通过重新审视文件相关性来改善丢失问题提交链接的恢复”，*信息与软件技术*，卷。84，第 33-47 页，2017 年。

[22] C. Mills、J. Escobar-Avila 和 S. Haiduc，“通过机器学习分类进行自动追溯维护”，载于*IEEE 软件维护和演进国际会议 (ICSME) 论文集*。IEEE，2018 年，第 369–380 页。

[23] C. Mills、J. Escobar-Avila、A. Bhattacharya、G. Kondyukov、S. Chakraborty 和 S. Haiduc，“用更少的数据进行跟踪：基于分类的可追溯性链接恢复的主动学习”，载于*IEEE 软件维护和演进国际会议 (ICSME) 论文集*。IEEE，2019 年，第 103-113 页。

[24] H. Ruan、B. Chen、X. Peng 和 W. Zhu，“Deeplink：基于深度学习恢复问题提交链接”，*系统与软件杂志*，卷。158，p。110406, 2019。

[25] AC Marcén、R. Lapeña、O. Pastor 和 C. Cetina，“使用由学习排序算法引导的进化算法实现需求和模型之间的可追溯性链接恢复：列车控制和管理案例”*系统与软件杂志*，卷。163，p。110519, 2020。

[26] J.Guo、J.Cheng 和 J.Cleland-Huang，“使用深度学习技术在语义上增强软件可追溯性”，载于*第 39 届 IEEE/ACM 国际软件工程会议 (ICSE) 论文集*。IEEE，2017 年，第 3-14 页。

[27] J. Lin、Y. Liu、Q. Zeng、M. Jiang 和 J. Cleland-Huang，“可追溯性转型：使用预训练的 bert 模型生成更准确的链接”，载于*第 43 届 IEEE/ACM 国际软件工程会议 (ICSE) 论文集*。IEEE，2021 年，第 324–335 页。

[28] A. Bachmann 和 A. Bernstein，“软件过程数据质量和特征：开源和闭源项目的历史观点”，载于 *软件演进原则 (IWPSE) 和软件演进 (Evol) 研讨会国际联合年度 ERCIM 研讨会论文集*，2009 年，第 119–128 页。

[29] D. Harman，“排名算法”，载于*信息检索：数据结构和算法*。Prentice-Hall, Inc.，1992 年，第 363–392 页。

[30] T. Chen、S. Kornblith、M. Norouzi 和 G. Hinton，“视觉表征对比学习的简单框架”，载于*第 37 届国际机器学习会议 (ICML) 论文集*。PMLR，2020 年，第 1597–1607 页。

[31] G. Salton 和 C. Buckley，“自动文本检索中的术语加权方法”，*信息处理与管理*，卷。24、没有。5，第 513-523 页，1988 年。

[32] T.Zhao、Q.Cao、Q.Sun，“一种基于词嵌入的追溯恢复改进方法”，载于*第24届亚太软件工程会议（APSEC）论文集*。IEEE，2017 年，第 81-89 页。

[33] M. Gethers、R. Oliveto、D. Poshyvanyk 和 A. De Lucia，“关于整合正交信息检索方法以提高可追溯性恢复”，载于*第 27 届 IEEE 软件维护国际会议 (ICSM) 论文集*。IEEE，2011 年，第 133-142 页。

[34] A. Mahmoud、N. Niu 和 S. Xu，“用于追溯链接恢复的语义相关性方法”，载于*第 20 届 IEEE 国际程序理解会议 (ICPC) 论文集*。IEEE，2012 年，第 183-192 页。

[35] D. Diaz、G. Bavota、A. Marcus、R. Oliveto、S. Takahashi 和 A. De Lucia，“使用代码所有权改进基于 ir 的可追溯性链接恢复”，载于*第 21 届国际程序理解会议 (ICPC) 论文集*。IEEE，2013 年，第 123–132 页。

[36] T. Dasgupta、M. Grechanik、E. Moritz、B. Dit 和 D. Poshyvanyk，“通过使用相关文档自动扩展语料库来增强软件可追溯性”，载于*IEEE 国际软件维护会议 (ICSM) 会议记录*。IEEE，2013 年，第 320-329 页。

[37] S. Lohar、S. Amornborvornwong、A. Zisman 和 J. Cleland-Huang，“通过数据驱动的配置和跟踪特征组合提高跟踪准确性”，载于*第九届软件工程基础联席会议(FSE)论文集*，2013 年，第 378–388 页。
[38] A. De Lucia、M. Di Penta、R. Oliveto、A. Panichella 和 S. Panichella，“应用平滑滤波器改进基于 ir 的可追溯性恢复流程：一项实证研究”，*信息与软件技术*，卷。55，没有。4，第 741–754 页，2013 年。
[39] A. Mahmoud 和 N. Niu，“论语义在自动需求跟踪中的作用”，*需求工程*，卷。20、没有。3，第 281-300 页，2015 年。

[40] K. Moran、DN Palacio、C. Bernal-Cárdenas、D. McCrystal、D. Poshyvanyk、C. Shenefiel 和 J. Johnson，“使用分层贝叶斯网络提高追溯链接恢复的有效性”，载于*第 42 届 ACM/IEEE 软件工程国际会议 (ICSE) 论文集*，2020 年，第 873–885 页。

[41] J. Pennington、R. Socher 和 CD Manning，“Glove：单词表示的全局向量”，载于*2014 年自然语言处理经验方法会议 (EMNLP) 会议论文集*，2014 年，第 1532–1543 页。

[42] J. Mueller 和 A. Thyagarajan，“用于学习句子相似性的连体循环架构”，载于*AAAI 人工智能会议 (AAAI) 论文集*，卷。30，没有。2016 年 1 月。
[43]“itertools — 创建迭代器以实现高效循环的函数”，https://docs.python.org/3/library/itertools.html，2022。
[44] R. Hadsell、S. Chopra 和 Y. LeCun，“通过学习不变映射进行降维”，载于*IEEE 计算机学会计算机视觉和模式识别会议 (CVPR) 会议论文集*，卷。2. IEEE，2006 年，第 1735–1742 页。

[45] J.德夫林，M.-W。Chang、K. Lee 和 K. Toutanova，“Bert：用于语言理解的深度双向转换器的预训练”*ArXiv 预印本 ArXiv：1810.04805*，2018。
[46]“Coest — 软件和系统可追溯性卓越中心”，http://coest.org，2022 年。
[47] Y. Wan，W. Zhao，H. Zhang，Y. Sui，G. Xu，and H. Jin，“他们做什么？捕获？——对源代码的预训练语言模型的结构分析”，*第 44 届国际软件工程会议 (ICSE) 论文集*。IEEE，2022。

[48] U. Alon、S. Brody、O. Levy 和 E. Yahav，“code2seq：从代码的结构化表示生成序列”，*arXiv 预印本 arXiv:1808.01400*，2018。

[49] U. Alon、M. Zilberstein、O. Levy 和 E. Yahav，“code2vec：学习代码的分布式表示”，载于*ACM 编程语言会议录*，卷。3、没有。POPL。ACM 美国纽约州，2019 年，第 1-29 页。

[50] Y. Sui、X. Cheng、G. Zhang 和 H. Wang，“Flow2vec：基于价值流的精确代码嵌入”，载于*ACM 编程语言会议录*，卷。4、没有。面向对象的SLA。ACM 美国纽约州，2020 年，第 1-27 页。

[51] K. He、H. Fan、Y. Wu、S. Xie 和 R. Girshick，“无监督视觉表示学习的动量对比”，载于*IEEE/CVF 计算机视觉和模式识别会议 (CVPR) 会议论文集*，2020 年，第 9729–9738 页。

[52] M. Caron、I. Misra、J. Mairal、P. Goyal、P. Bojanowski 和 A. Joulin，“通过对比聚类分配对视觉特征进行无监督学习”*神经信息处理系统的进展*，卷。33，第 9912–9924 页，2020 年。

[53] P. Khosla、P. Teterwak、C. Wang、A. Sarna、Y. Tian、P. Isola，A. Maschinot、C. Liu 和 D. Krishnan，“监督对比学习”，*神经信息处理系统的进展*，卷。33，第 18 661–18 673 页，2020 年。

[54] ND Bui、Y. Yu 和 L. Jiang，“通过语义保留转换进行代码检索和摘要的自监督对比学习”，载于*第 44 届国际 ACM SIGIR 信息检索研究与发展会议 (SIGIR) 论文集*，2021 年，第 511–521 页。

[55] X. Cheng、G. Zhang、H. Wang 和 Y. Sui，“通过对比学习进行路径敏感代码嵌入用于软件漏洞检测”，载于 *ACM SIGSOFT 国际软件测试与分析研讨会 (ISSTA) 论文集*，2022 年，第 519–531 页。