AgenticTyper：使用智能体人工智能对遗留软件项目进行自动化类型标注

Clemens Pohle 达姆施塔特应用科学大学 达姆施塔特，德国
clemens.pohle@stud.h-da.de
MaibornWolff GmbH 达姆施塔特，德国
clemens.pohle@maibornwolff.de

arXiv:2602.21251v1 [cs.SE] 21 Feb 2026

摘要

遗留JavaScript系统缺乏类型安全，使维护面临风险。虽然TypeScript可以提供帮助，但手动添加类型成本高昂。以往的自动化类型标注研究侧重于类型推断，却很少在仓库规模上处理类型检查配置、定义生成、缺陷识别或行为正确性。我们提出AgenticTyper，一种基于大语言模型（LLM）的智能体系统，通过迭代纠错，并借助转译对比实现行为保持，从而弥补上述不足。对两个专有代码仓库（81K代码行数）的评估表明，AgenticTyper在20分钟内解决了全部633个初始类型错误，将人工工作量从一个工作日大幅减少。

CCS概念

• **软件及其工程** → **软件维护工具**;**自动编程**; •**计算理论** → 类型结构; • **计算方法** → 智能体; •**一般与参考** →实证研究。

关键词

可选类型，遗留软件，智能体人工智能，JavaScript，TypeScript

ACM 引用格式：

Clemens Pohle. 2026. AgenticTyper：使用智能体人工智能对遗留软件项目进行自动化类型标注。发表于*2026*年 *IEEE/ACM* 第*48*届国际软件工程会议（*ICSE-Companion '26*），*2026*年*4*月*12*日至*18*日，巴西里约热内卢。ACM，纽约，美国，3页。https://doi.org/10.1145/ 3774748.3787746

1 问题与动机

业务关键型遗留系统通常使用 JavaScript 等动态类型语言编写 [6, 35], ，积累了巨大的商业价值，却也变得越来越难以修改和理解 [5,25]。缺乏类型安全会使变更风险重重，还可能破坏现有功能 [12]。可选类型系统 [9]（如 TypeScript）能够提供文档 [15], 、明确依赖关系 [22],并避免错误 [13]。然而，手动添加类型安全成本高昂 [13], ，通常需要数年时间 [40]。开发者必须追踪跨文件的依赖关系、设计可复用的类型定义，并解决级联问题，因为每个类型注解都可能暴露依赖代码中的新问题。因此，我们介绍正在进行的 AgenticTyper 工作[1] [30], ： 利用基于大语言模型（LLM）的编码智能体，在不引入回归问题的情况下，自动为遗留 JavaScript 代码仓库添加类型。

本作品采用知识共享署名 4.0 国际许可协议授权。*ICSE-Companion '26*，巴西里约热内卢
© 2026 版权所有者/作者保留所有权利。ACM ISBN 979-8-4007-2296-7/2026/04
https://doi.org/10.1145/3774748.3787746

2 背景与相关工作

自动化类型标注研究侧重于使用静态方法 [1,19],动态方法 [2, 29],概率方法 [33, 37],和混合方法 [27,31]进行类型推断。然而，这些方法面临关键局限 [14]: 静态方法无法处理动态特性 [36],动态方法需要测试套件 [17], 而概率方法会产生必须舍弃的错误预测 [31]。生成类型定义 [39] 以及识别 [32] 和修复 [26] 现有类型问题仍未得到充分研究。

大语言模型已应用于众多软件工程任务 [18], , 包括类型推断 [10, 20, 21], 、修复类型错误 [11, 16, 28], 以及与静态分析结合使用 [7, 34]。然而，仅通过提示大语言模型添加类型，在代码库规模下并不可行：模型无法浏览和编辑多个文件、处理级联错误，也无法验证行为保持。基于大语言模型的智能体 [38]（智能体人工智能 [24]）通过使用工具和进行多步推理来应对这些问题，近期已应用于程序修复 [8, 23],，但尚未用于类型标注任务。

3 方法与独特性

与以往工作不同，AgenticTyper 将智能体人工智能与转译结果比较相结合，以进行变更检测，从而无需测试套件即可在代码库层面添加类型。此外，它还能区分需要人工关注的实际错误，以及若不改变程序行为就无法添加类型的有效模式（例如，类型强制转换）。

我们提出三个渐进阶段（图 1）。**第一阶段** 建立最低限度的 TypeScript 类型检查。编码智能体分析并处理每个文件中的类型错误（TypeScript 编译器诊断信息），方法是添加或修复 JSDoc 类型注解（例如 @type、@param）。当无法在不改变行为的情况下解决错误时，智能体会插入抑制性注释 （@ts-expect-error）： 将错误标记出来供人工审查，而对有效模式则给出中性说明，

[1]源代码：https://github.com/clemens-mw/agentic-typer

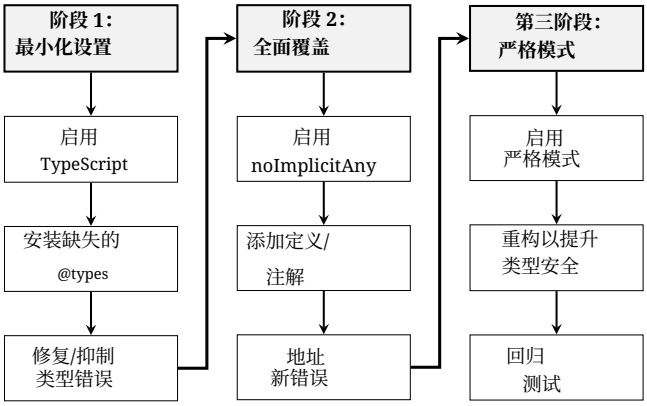

图 1：三阶段类型标注方法。

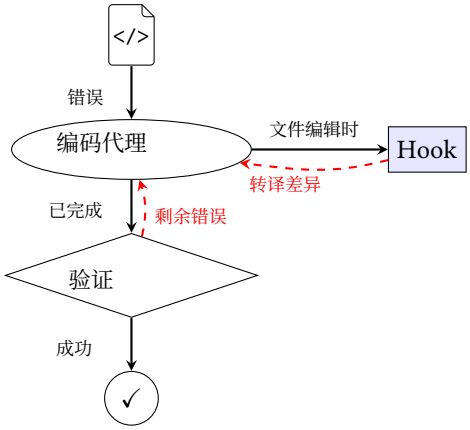

图 2：第一和第二阶段的单智能体架构。防止行为变化的钩子已突出显示。

从而建立干净基线。**第二阶段** 启用 noImplicitAny，要求在各处添加显式注解。智能体添加类型定义和注解，并像第一阶段那样处理新出现的错误。**第三阶段** 启用严格检查，揭示出往往需要重构才能解决的问题（例如空值检查）。由于重构存在风险，此阶段需要通过生成的测试或人工验证进行回归测试。

在第一阶段和第二阶段中，行为保持机制（图 2）无需测试即可防止回归：每次编辑后，钩子都会将修改后的代码转译为 JavaScript，并与原始转译输出进行比较，提醒智能体注意任何运行时变化。智能体完成工作后，静态验证会检查是否仍有错误，再继续后续步骤。多个智能体并行运行，最后由验证智能体处理先前修复引入的跨文件错误。

4 结果与贡献

截至目前，我们已使用 Claude Agent SDK [3] 实现第一阶段，5 [4], 选择 Claude Sonnet 4，因为截至 2025 年 9 月，它拥有领先的编程能力。我们在某工业

表 1：两个专有 JavaScript 代码仓库的第一阶段结果。每项结果为三次运行的平均值。

Repo	LOC	Type 错误	必要抑制	额外抑制	Time	Cost
A	75K	570	327	+26 (+8.0%)	16:51 分钟	$22.85
B	6K	63	56	+0 (+0.0%)	3分06秒	$2.08
总计	**81K**	**633**	**383**	**+26 (+6.8%)**	**19分57秒**	**$24.93**

设施管理平台的两个专有旧版 Node.js 后端代码仓库上评估了 AgenticTyper（表 1）。系统配置为同时运行十个智能体。

AgenticTyper 成功建立了类型检查，并解决了最初的 633 个类型错误，同时未改变运行时行为；这一点已通过转译对比得到验证。此前，这项任务需要一位经验丰富的 TypeScript 开发者花费整整一个工作日，如今只需 20 分钟，成本为 25 美元。虽然 AgenticTyper 能妥善处理大多数跨文件依赖关系，但如果根因追踪更精确，仍有 7% 的抑制操作本可避免。例如，参数遮蔽，以及智能体在多个使用位置而非唯一源头进行抑制的错误。

总而言之，我们的贡献包括：(1) 一种在自动化类型标注过程中保持行为的新型智能体方法；(2) 一种支持智能体并行执行的开源实现；(3) 一项评估，证明该方法在实践中切实可行且具有成本效益。

5 未来工作

我们将针对相同的代码库，以及一个包含 300,000 代码行、无需第一阶段处理的 TypeScript 代码库，实现并评估第二和第三阶段。评估将衡量类型覆盖率（已添加类型的元素所占比例）、通过类型检查和人工审查评估的注解正确性，以及以小时数和变更行数衡量的清理工作量。我们还将研究不同的大语言模型，并将该方法扩展到 Python 等其他可选类型语言。

参考文献

[1] Alex Aiken 和 Brian Murphy。1991 年。动态类型语言中的静态类型推断。载于 第*18*届 *ACM SIGPLAN-SIGACT* 编程语言原理研讨会论文集（美国佛罗里达州奥兰多）（*POPL '91*）。ACM，美国纽约州纽约，279 290 10 1145 99583 99621 –。doi: 。/。

[2] Jong-hoon D. An、Avik Chaudhuri、Jeffrey S. Foster 和 Michael Hicks。2011 年。Ruby 静态类型的动态推断。*ACM SIGPLAN Notices* 46，1（2011 459 472 10 1145 1925844 1926437

[3] 2025 年1月），–。doi: 。/。Anthropic。使用 *Claude Agent SDK* 构建智能体。Anthropic。检索于 2025年11月6日，网址：https://www.anthropic.com/engineering/building-agents-with-the-claude-agent-sdk [4] Anthropic 2025。*Introducing Claude Sonnet 4.5*。Anthropic。检索于 6, 2025 4 5 年11月，网

[5] 址：https://www.anthropic.com/news/claude-sonnet- - Wesley K. G. Assunção、Luciano Marchezan、Lawrence Arkoh、Alexander Egyed 和 Rudolf Ramler。2025 年。当代软件现代化：策略、驱动因素与研究机遇。*ACM* 软件工程与方法学汇刊 34，5，文章142（2025年5月），35页。doi:10.1145/3708527[6] Olga Bedrina。2025 年。*2025* 年开发者生态系统现状：*AI* 时代的编码、新生产力指标与不断变化的现实。JetBrains。检索于2025年11月6日，网址：https://blog.jetbrains.com/research/2025/10/state-of-developer- ecosystem-2025/#languages-and-tools [7] Varun Bharti、Shashwat Jha、Dhruv Kumar 和 Pankaj Jalote。2025 年。使用大语言模型自动为 *Python* 添加类型注解。arXiv: 2508 00422 [ ]。cs.PL Islem

[8] 2025 Bouzenia、Premkumar Devanbu 和 Michael Pradel。程序修复智能体：一种基于大语言模型的自主程序修复智能体。载于 *IEEE/ACM* 第*47*届国际软件工程大会论文集（加拿大安大略省渥太华）（*ICSE '25*）。IEEE，美国纽约州纽约，2188–2200。doi:10.1109/ICSE55347。

2025.00157[9] Gilad Bracha。2004 年。可插拔类型系统。载于 OOPSLA04 动态语言复兴研讨会（加拿大不列颠哥伦比亚省温哥华）。检索于2025年11月6日，网址：https://bracha.org/pluggableTypesPosition.pdf[10] Federico Cassano、Ming-Ho Yee、Noah Shinn、Arjun Guha 和 Steven Holtzen。2023 年。通过程序分解和填空式类型训练进行类型预测。arXiv: 2305 17145 [ ] 。cs.SE Yiu Wai Chow、Luca Di Grazia 和 Michael Pradel[11]。2024 年。PyTy：修复 Python 静态类型错误。载于 IEEE/ACM 第46届国际软件工程大会论文集（葡萄牙里斯本）（ICSE '24）。ACM，美国纽约州纽约，文章 87, 13 10 1145 3597503 3639184 页。doi：。/。Michael C. Feathers 和 Robert C. Martin[12]。2004。有效处理遗留代码。Prentice Hall Professional Technical Reference，美国新泽西州上萨德尔河。[13] Zheng Gao、Christian Bird 和 Earl T. Barr。2017 年。要不要使用类型：量化 JavaScript 中可检测的错误。载于第39届国际软件工程大会论文集（阿根廷布宜诺斯艾利斯）（ICSE '17）。IEEE，美国纽约州纽约，758–769。doi:10.1109/ICSE.2017.75[14] Yimeng Guo、Zhifei Chen、Lin Chen、Wenjie Xu、Yanhui Li、Yuming Zhou 和 Baowen Xu。2024 年。从类型推断生成 Python 类型注解：我们能走多远？ACM 软件工程与方法学汇刊 33，5，文章123（ 2024年6月），38页。doi:10.1145/3652153[15] Stefan Hanenberg、Sebastian Kleinschmager、Romain Robbes、Éric Tanter 和 Andreas Stefik。2014 年。静态类型对软件可维护性影响的实证研究。实证软件工程 19，5（2014年10月），1335–1382。doi:10.1007/s10664-013-9289-1[16] Sichong Hao、Xianjun Shi 和 Hongwei Liu。2024 年。RetypeR：面向 Python 类型错误的集成式检索增强自动程序修复。载于2024 IEEE 软件维护与演化国际会议（美国亚利桑那州弗拉格斯塔夫）（ICSME ' *24* 199 210 10 1109 58944 2024 00028 ）。IEEE，美国纽约州纽约，–。doi：。/ICSME。。Joshua Hoeflich、Robert Bruce Findler 和 Manuel Serrano[17]。2022。高度不合逻辑，Kirk：尽管存在失效契约、不健全类型和过多代码检查工具，仍能发现大量类型不匹配。ACM 编程语言论文集 6，OOPSLA2，文章142（2022年10月），26页。doi:10.1145/3563305[18] Xinyi Hou、Yanjie Zhao、Yue Liu、Zhou Yang、Kailong Wang、Li Li、Xiapu Luo、David Lo、John Grundy 和 Haoyu Wang。2024 年。用于软件工程的大语言模型：系统性文献综述。ACM 软件工程与方法学汇刊 33，8，文章 220（2024年12月），79页。doi:10.1145/3695988[19] Simon Holm Jensen、Anders Møller 和 Peter Thiemann。2009 年。JavaScript 的类型分析。载于静态分析（美国加利福尼亚州洛杉矶）（SAS '09）。Springer，德国柏林、海德堡， 238 255 10 1007 978 3 642 03237 0_17 –。doi：。/---- Kevin Jesse、Premkumar T. Devanbu 和 Toufique Ahmed[20]。2021。学习类型注解：大数据是否足够？载于第29届 ACM 欧洲软件工程大会与软件工程基础研讨会联合会议论文集（希腊雅典）（ESEC/FSE '21）。ACM，美国纽约州纽约， 1483 1486 10 1145 3468264 3473135 –。doi：。/。Kevin Jesse、Premkumar T. Devanbu 和 Anand Sawant[21]。2023 年。学习预测用户定义类型。IEEE 软件工程汇刊 49，4（ 2023 1508 1522 10 1109 2022 3178945 年4月），–。doi：。/TSE。。Wuxia Jin、Dinghong Zhong、Yuanfang Cai、Rick Kazman 和 Ting Liu[22]。2023 年。评估潜在依赖关系对架构级软件可维护性的影响。IEEE 软件工程汇刊 49，3（2023年3月），1064–1085。doi:10.1109/TSE.2022.3171288[23] Pascal Joos、Islem Bouzenia 和 Michael Pradel。2025 年。CodeCureAgent：静态分析警告的自动分类与修复。arXiv: 2509 11787 [ ] 。cs.SE Sayash Kapoor、Benedikt Stroebl、Zachary S. Siegel、Nitya Nadgir 和 Arvind Narayanan[24]。2025 年。具有影响力的人工智能智能体。机器学习研究汇刊（2025年5月），38页。检索于2025年11月6日，网址：https://openreview.net/forum?id=Zy4uFzMviZ[25] Ravi Khadka、Belfrit V. Batlajery、Amir M. Saeidi、Slinger Jansen 和 Jurri- aan Hage。2014 年。专业人士如何看待遗留系统与软件现代化？载于第36届国际软件工程（印度海得拉巴）（ICSE '14）。ACM，美国纽约州纽约，36–47。doi:10.1145/2568225.2568318[26] Wonseok Oh与Hakjoo Oh。2022年。PyTER：针对Python类型错误的有效程序修复。载于第30届ACM欧洲软件工程联合会议暨软件工程基础研讨会论文集（新加坡）（ESEC/FSE ' *22* 922 934 10 1145 3540250 3549130 ）。ACM，美国纽约州纽约，–。doi: . / . Irene Vlassi Pandi、Earl T. Barr、Andrew D. Gordon与Charles Sutton[27]。2021。OptTyper：通过优化逻辑约束与自然约束进行概率类型推断。arXiv: 2004 00348 [ ] . cs.PL Yun Peng、Shuzheng Gao、Cuiyun Gao、Yintong Huo与Michael Lyu[28]。2024年。领域知识至关重要：利用修复模板改进提示以修复Python类型错误。载于IEEE/ACM第46届国际软件工程大会论文集（葡萄牙里斯本）（ICSE '24）。ACM，美国纽约州纽约，文章编号 4, 13 10 1145 3597503 3608132 页。doi: . / . Juan A. Pizzorno与Emery D. Berger[29]。2025。RightTyper：面向Python的有效且高效的类型注解。arXiv: 2507 16051 [ ] . cs.PL Clemens Pohle[30]。2025。AgenticTyper：使用智能体人工智能对遗留软件项目进行自动化类型标注。doi: 10 5281 17954447 . /zenodo. Michael Pradel、Georgios Gousios、Jason Liu与Satish Chandra[31]。2020年。TypeWriter：结合基于搜索的验证的神经类型预测。载于第28届ACM欧洲软件工程会议暨软件工程基础研讨会联合会议论文集（线上活动，美国）（ESEC/FSE ' *20* 209 220 10 1145 3368089 3409715 ）。ACM，美国纽约州纽约，–。doi: . / . Michael Pradel、Parker Schuh与Koushik Sen[32]。2015。TypeDevil：面向JavaScript的动态类型不一致性分析。载于第37届国际软件工程会议论文集（意大利佛罗伦萨）（ICSE '15，第1卷）。IEEE，美国纽约州纽约， 314 324 10 1109 2015 51 –。doi: . /ICSE. . Veselin Raychev、Martin Vechev与Andreas Krause[33]。2015年。从“大规模代码”预测程序属性。载于第42届ACM SIGPLAN-SIGACT编程语言原理年度研讨会论文集（印度孟买）（POPL '15）。ACM，美国纽约州纽约，111–124。doi:10.1145/2676726.2677009[34] Lukas Seidel、Sedick David Baker Effendi、Xavier Pinho、Konrad Rieck、Brink van der Merwe与Fabian Yamaguchi。2023年。学习类型推断以增强数据流分析。载于《计算机安全》（荷兰海牙）（ESORICS '23）。Springer，瑞士沙姆，184–203。doi:10.1007/978-3-031-51482-1 10[35] Stack Exchange 2025年。2025年Stack Overflow开发者调查。Stack Exchange。于2025年11月6日检索，来源 https://survey.stackoverflow.co/2025/technology#1-programming-scripting-and-markup-languages[36] Ke Sun、Yifan Zhao、Dan Hao与Lu Zhang。2023年。面向Python的静态类型推荐。载于第37届IEEE/ACM自动化软件工程国际会议论文集（美国密歇根州罗切斯特）（ASE '22）。ACM，美国纽约州纽约，文章编号 98, 13 10 1145 3551349 3561150 页。doi: . / . Jiayi Wei、Maruth Goyal、Greg Durrett与Isil Dillig[37]。2019。LambdaNet：使用图神经网络的概率类型推断。载于第7届学习表征国际会议论文集（美国路易斯安那州新奥尔良）（ICLR '19）。Curran Associates, Inc.，美国纽约州雷德胡克，11页。于2025年11月6日检索，来源 https://openreview.net/forum?id=Hkx6hANtwH[38] Zhiheng Xi、Wenxiang Chen、Xin Guo、Wei He、Yiwen Ding、Boyang Hong、Ming Zhang、Junzhe Wang、Senjie Jin、Enyu Zhou、Rui Zheng、Xiaoran Fan、Xiao Wang、Limao Xiong、Yuhao Zhou、Weiran Wang、Changhao Jiang、Yicheng Zou、Xiangyang Liu、Zhangyue Yin、Shihan Dou、Rongxiang Weng、Wenjuan Qin、Yongyan Zheng、Xipeng Qiu、Xuanjing Huang、Qi Zhang与Tao Gui。2025年。基于大语言模型的智能体的兴起与潜力：综述。《中国科学：信息科学》68, 2，文章编号121101（2025年1月），44页。doi:10.1007/s11432-024-4222-0[39] Ming-Ho Yee。2024年。利用机器学习预测TypeScript类型注解与定义。博士学位论文。东北大学。doi:10.17760/ D20653005[40] Ming-Ho Yee与Arjun Guha。2023年。机器学习模型生成的TypeScript类型能否通过类型检查？载于第37届欧洲面向对象编程会议（美国华盛顿州西雅图）（ECOOP '23）。Schloss Dagstuhl，德国瓦登，文章编号37，28页。doi:10.4230/LIPIcs.ECOOP.2023.37