arXiv:2603.04476v1 [cs.SE] 4 Mar 2026

iScript：用于物理设计 Tcl 脚本生成的领域适配大语言模型与基准

Ning Xu 国家EDA技术创新中心 东南大学 中国南京

Zhaoyang Zhang 国家EDA技术创新中心 东南大学 中国南京

Senlin Shu 国家EDA技术创新中心 中国南京

Lei Qi 东南大学 中国南京

Jiaqi Lv 东南大学 中国南京

Wensuo Wang 国家EDA技术创新中心 中国南京

Tianhao Zhao 国家EDA技术创新中心 中国南京

Chao Zhang 国家EDA技术创新中心 中国南京

Zhaoliang Yang 国家EDA技术创新中心 中国南京

Xiangyu Li 国家EDA技术创新中心 中国南京

Zhaorui Su 国家EDA技术创新中心 中国南京

Jingshan Li 国家EDA技术创新中心 中国南京

Xin Geng 东南大学 中国南京

摘要

现代电子设计自动化流程严重依赖 Tcl 脚本编写，但由于数据极度匮乏、领域特定语义以及物理设计对可靠性的高要求，通用大语言模型在这一领域表现不佳。我们提出 iScript，这是一种面向 Innovus Tcl 脚本生成进行领域适配的 Qwen3-8B 模型；同时提出 iScript-Bench，这是一个全面的基准，涵盖五个任务类别和三个难度级别。为弥补训练数据不足，我们引入了一种多阶段数据合成流水线，整合命令提取、静态检查、需求反向推断和思维链生成，产出包含 10K 个（需求、思维链、脚本）三元组的数据集。iScript 通过结合领域自适应预训练与监督微调的两阶段策略进行训练。为高效评估脚本的正确性，我们进一步提出了一个两步验证框架，包括静态语法验证和基于大语言模型的功能评估。在我们的基准上，iScript 表现出更高的平均而言，pass@k分数高于当前最先进的大语言模型。这些结果证明了领域自适应和数据合成对电子设计自动化脚本任务的有效性。

CCS概念

• **硬件** → **集成电路**； • **计算方法** → **人工智能**。

关键词

大语言模型、电子设计自动化、脚本生成

ACM 引用格式：

Ning Xu、Zhaoyang Zhang、Senlin Shu、Lei Qi、Jiaqi Lv、Wensuo Wang、Tian-hao Zhao、Chao Zhang、Zhaoliang Yang、Xiangyu Li、Zhaorui Su、Jingshan Li 和 Xin Geng。2018年。iScript：一种用于物理设计 Tcl 脚本生成的领域适配大语言模型和基准。载于请填写版权确认邮件中的正确会议名称（会议缩写*"XX"*）。美国计算机协会，纽约州纽约市，8页。https://doi.org/XXXXXXX.XXXXXXX

允许免费制作本作品全部或部分内容的数字或纸质副本，供个人或课堂使用，前提是副本不得出于营利或商业利益而制作或分发，且副本须在首页载明本声明及完整引文。对于本作品中由作者以外的他人拥有的组成部分，其版权必须得到尊重。允许在注明出处的情况下进行摘要。以其他方式复制、再版、在服务器上发布或向列表再分发，均须事先获得特别许可和/或支付费用。请通过 permissions@acm.org 申请许可。会议缩写*"XX"*，纽约州伍德斯托克© 2018年版权由所有者/作者持有。出版权已授权给美国计算机协会。美国计算机协会 ISBN 978-1-4503-XXXX-X/2018/06 https://doi.org/XXXXXXX.XXXXXXX

1 引言

随着工艺技术向个位数纳米推进，现代专用集成电路（ASIC）实现中的物理设计（PD）阶段日益复杂。当代布局布线（P&R）流程高度依赖 Tcl 驱动的自动化，以将

布局规划、布局、时钟树综合、布线和设计收敛整合到统一的优化流程中。行业报告和工具文档表明，完整的量产流程通常需要数千至数万行 Tcl 脚本，涵盖工具配置、约束生成、层次化管理、增量优化以及针对签核的调优 [4, 19]。随着设计规模扩大，功耗、性能与面积（PPA）要求不断提高，编写和维护这些脚本已成为物理设计工程师最耗时、最依赖专业知识的任务之一。因此，依赖人工编写 Tcl 脚本不仅造成了效率瓶颈，也成为新工程师进入先进制程物理设计领域的一道门槛。

大语言模型的近期进展激发了人们对自动化半导体工作流程部分环节的兴趣。多项研究探索了大语言模型辅助的 RTL 生成 [2, 15], 设计调试 [9], EDA 文档理解 [18],以及布局分析 [8, 13, 21]。尽管这些工作前景可期，但主要针对前端设计阶段或自然语言推理任务。尽管物理设计专用 Tcl 脚本编写在工业流程中发挥着核心作用，相关研究却寥寥无几。现有大语言模型直接用于物理设计脚本编写时也存在明显局限：（1）工具专用 Tcl 命令及其选项构成了高度专业且稀少的词汇，一般大语言模型很少接触这些内容；（2）物理设计脚本中的命令在语义上紧密耦合（例如，布局区域必须与布局规划几何结构相匹配，时钟树综合设置取决于时序约束），因此，将自然语言需求映射为可执行 Tcl 尤具挑战性；（3）由于工作流程具有专有性，公开可用的物理设计脚本匮乏，造成了严重的数据匮乏问题，阻碍了模型训练和基准测试。这些观察结果凸显了通用大语言模型与物理设计自动化领域的特定需求之间存在根本性错配。

尽管人们对“电子设计自动化+大语言模型”研究的兴趣日益浓厚，但该领域仍缺少用于评估大语言模型在物理设计 Tcl 生成方面表现的标准化基准。相比之下，自然语言转代码研究的进展得益于 HumanEval [5], MBPP [3], 和 EvalPlus [12], 等全面基准，它们定义了有意义的任务、具有代表性的难度级别以及可复现的指标。物理设计自动化领域尚无类似基准，因此一些根本性问题仍未得到解答：

最先进的大语言模型对物理设计意图的理解能力如何？它们能否可靠地将复杂的设计需求映射为语法正确且具有实际功能的 *Tcl* 命令？领域适配模型与通用模型相比表现如何？

如果缺乏统一的基准，就几乎无法对不同模型进行比较，也难以复现结果。

除了缺乏基准之外，如何评估物理设计 Tcl 生成也尚未形成共识。真正的功能验证需要在商业 EDA 工具中执行脚本，这既成本高昂，又受到诸多限制，且无法复现。相反，纯粹基于语法的检查无法判断脚本是否满足设计意图。以往的 EDA+大语言模型研究 [14] 基本都回避了这一问题，导致评估要么不完整（仅检查语法），要么停留在定性层面（人工审查）。因此，我们需要一种可扩展、可复现且独立于工具的评估框架，以推动研究取得实质性进展。

为弥补这些不足，我们推出了 iScript——一种专用于物理设计 Tcl 生成的领域适配型大语言模型，以及 iScript-Bench——首个专为评估布局布线流程中自然语言到 Tcl 转换能力而设计的基准。我们的工作得益于一个多阶段数据合成流水线，该流水线整合了：（i）从公开来源整理领域语料库；（ii）从 Innovus 手册中提取并规范化物理设计命令；（iii）借助教师大语言模型，通过思维链推理自动生成脚本；（iv）利用轻量级物理设计沙箱进行静态语法验证。借助该流水线，我们构建了一个包含一万条样本的数据集，每条样本由（需求、推理、脚本）组成，涵盖布局规划、布局、时钟设计、布线和设计优化任务，并分为三个难度级别。我们还在 Qwen3-8B 的基础上进行领域自适应持续预训练（CPT）和推理增强监督微调（SFT），从而得到 iScript。

我们还开发了一个两步验证框架，将（1）基于物理一致的 Innovus 风格沙箱进行的语法检查，与（2）配备物理设计命令知识的功能评估大语言模型相结合。尽管该框架无法替代完整的工具执行，但它能够对不同模型进行可扩展、一致且可复现的比较。

我们的贡献如下：

- 我们推出了 iScript，这是一款通过持续预训练+监督微调并结合思维链监督进行训练的领域适配大语言模型，能够生成语法有效且语义明确的物理设计 Tcl 脚本。
- 我们提出了 iScript-Bench，这是首个面向物理设计 Tcl 脚本生成的基准，涵盖五个任务类别和三个难度级别。
- 我们开发了一种可扩展的两步验证框架，无需访问商业 EDA 工具，即可可靠地评估语法和功能。

2 相关工作

大语言模型在电子设计自动化领域的应用主要沿着两个密切相关的方向发展：(1) 通用代码生成与程序合成；(2) 面向设计自动化任务的 EDA 专用适配。

面向代码的大语言模型。 程序合成领域的既有研究确立了“在代码上预训练、针对任务进行微调”的范式，这一范式构成了我们方法的基础。CodeTrans 和 CodeT5 等大规模模型与数据集表明，结合语法和标识符感知的预训练能够提升下游代码生成与检索性能 [7, 20]。HumanEval 和 MBPP 等基准提供了可复现的任务定义和 pass@k 评估流程，塑造了后续代码生成社区中的评估实践 [3, 5]。这些工作促使我们采用这一（需求 → 脚本）监督微调设置。

面向芯片设计的领域适配。近期研究表明，在芯片设计语料上进行领域自适应持续预训练，能显著提升电子设计自动化任务的表现。NVIDIA 的 ChipNeMo 在专有芯片数据上进行持续预训练，并证明其工具脚本生成能力优于通用模型 [13]。类似的领域适配策略已在其他垂直领域成功应用，这促使我们采用持续预训练+监督微调流水线。

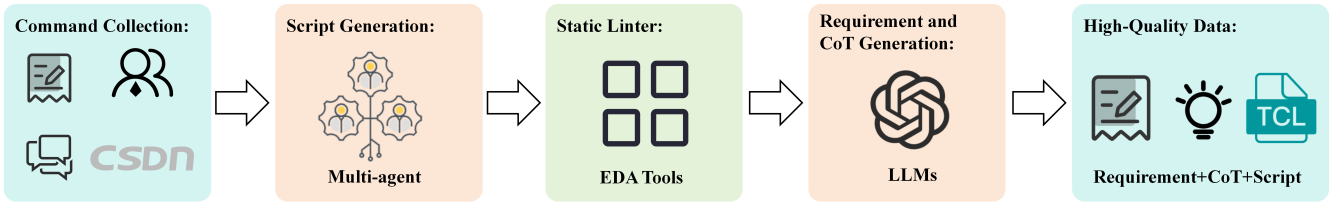

图 1：训练数据合成

用于电子设计自动化工作流和脚本编写的大语言模型。近期有多个系统专门针对电子设计自动化流程和工具控制的自动化。ChatEDA 等自主智能体框架及其变体已在开放工具链中展示了多步骤流程编排能力（脚本生成+ 执行），凸显了智能体自动化的潜力与可靠性挑战 [21]。LayoutCopilot 等相关工作探索了由大语言模型驱动的模拟版图脚本生成和 SKILL 语言任务，强调了领域知识和交互式优化的重要性 [11]。这些工作与我们的主要区别在于研究范围：它们关注多智能体编排或交互式工作流，而我们的工作针对单轮需求 → Tcl 脚本生成，并提出了一个用于系统评估的基准。

数据合成与低资源问题的应对方法。 电子设计自动化大语言模型面临的一个常见障碍是高质量公开训练数据稀缺。为启动模型训练，人们已采用多种方法合成任务数据，例如通过程序化变异、教师模型反向翻译或多智能体生成（如 ChipExpert [16], JARVIS [17]）。我们的数据合成流水线沿用了这一趋势，但着重于：(a) 根据工具手册进行命令级规范化；(b) 通过静态检查确保语法有效；以及 (c) 提供思维链（CoT）监督，教会模型推理布局布线意图——此前尚未有针对 Innovus 风格 Tcl 生成而公开发表的此类组合方法。

评估方法。 最后，基准构建与验证仍是尚待解决的问题。VerilogEval 为硬件描述语言生成引入了基于沙箱的功能评估，相较于词汇指标，更倾向于依据执行结果判断正确性 [14]。在布局布线领域，运行完整工具流程成本高昂且难以复现；因此，先前的研究转而采用仅检查语法的方法或有限的人工评估。我们的贡献是一种两步验证策略：先进行静态语法检查，再使用经大语言模型增强的功能评估器；该策略旨在实现可扩展性与可复现性，同时承认非执行式评估的局限性。

与先前的工作相比，iScript 将领域自适应预训练、结合思维链监督的面向综合的数据流水线，以及针对 Innovus 风格 Tcl 任务定制的基准结合起来。其主要创新在于：将这些要素整合为一个专门面向布局布线 Tcl 生成、可复现的基准和评估流程，并通过实证研究比较领域适配型大语言模型与通用大语言模型在该任务上的表现。

3 iScript

本节将介绍训练数据的收集、iScript 的训练详情以及评估策略。

3.1 训练数据准备

训练面向电子设计自动化的专用大语言模型，主要挑战在于缺乏大规模、高质量的开源数据集。我们的策略是利用多源公共数据、程序化生成和基于大语言模型的优化，构建领域专用的（指令、思维链、代码）数据集。

如图2所示，我们从多种来源收集种子数据：

- EDA工具用户指南：提供常见流程和命令用法示例。
- EDA命令参考手册：详细说明单个命令及参数的语法。
- 中国技术社区（CSDN）：提供教程和用户生成的脚本示例。
- 技术论坛：包含用户问题和专家提供的解决方案。

我们的流水线由三个新颖阶段组成：

阶段1：组合式脚本生成 首先，我们从收集的数据中提取一份全面的核心Tcl 命令及其相关参数列表。然后，我们通过创建这些命令和参数的排列与组合，生成大量Innovus Tcl脚本片段。生成过程基于受 [17]启发的多智能体系统。这一初始语料库规模很大，但质量较低，且常常存在语法错误。

阶段2：静态语法验证 来自阶段1的原始脚本会通过Tcl代码检查器（静态语法检查器）。这一步至关重要，因为它会过滤掉所有语法无效的脚本，确保用于下一阶段的数据至少在结构上是正确的。由此得到一个由语法有效脚本组成的大型语料库。

阶段3：需求反向推断与思维链数据生成 这是我们数据合成贡献的核心。

- 需求反向推断：我们使用强大的教师模型（GPT-4.1）对语法有效的脚本进行“逆向工程”。我们提示模型推断最有可能生成给定脚本的自然语言“需求”或“意图”。
- 思维链生成：我们使用（需求，脚本）对再次提示教师模型，要求其生成思维链（CoT）说明，详细阐述脚本如何实现需求的逐步推理过程。

这一三阶段流程将低质量的多源数据转化为适用于监督微调的高质量（需求、思维链、脚本）数据集，我们称之为“高质量问答”数据。这一流水线

生成了我们的最终数据集，其中包含 10,000 组三元组（需求、思维链、脚本）。该数据集主要面向 Cadence Innovus Tcl，涵盖布局规划、电源规划、布局、时钟树综合和布线等任务。

3.2 训练详情

我们选择 Qwen3-8B [23] 作为基础模型。此决定基于该模型在通用编程基准测试中的出色表现、其原生的英语/中文双语能力（与我们的多源数据（CSDN、论坛）十分契合），以及其 8B 参数规模；这些因素使其在高性能与可行的训练成本之间实现了稳健的平衡。我们的训练遵循最先进的“CPT+SFT”方法，该方法已被证明在领域特定的电子设计自动化任务中行之有效，相关工作包括 [13] 和 ChatEDA [21]。

领域自适应持续预训练（持续预训练）这一阶段的主要目标是将 Innovus Tcl 的完整“词汇”和语法模式嵌入模型权重中。在通用预训练阶段，基础模型 Qwen3-8B 很可能几乎没有接触过 Innovus Tcl 数据，甚至完全没有接触过。持续预训练阶段会对模型进行“预热”，使其内部表征适应这种新型的专用语言。当模型进入第二阶段时，它将不再为如何编写 Tcl 而困扰；它只需学习如何根据指令编写相应的 Tcl。经过持续预训练后，模型将熟练掌握 Innovus Tcl 语法。其生成连贯且语法有效的脚本片段（即“流畅度”）的能力将显著提升，为下一阶段奠定坚实基础。

带思维链的监督微调（SFT）在模型已熟练掌握 Tcl 语法之后，本阶段教授其从用户意图（需求）到正确动作（脚本）的高层语义映射。我们在本阶段明确纳入思维链数据，以教会模型不仅知道生成什么，还知道为何如此生成。LLSM [8] 等研究表明，“电子设计自动化引导的思维链”对于电子设计自动化领域（尤其是逻辑综合）中的复杂推理高度有效。此外，EDAid [22] 利用思维链来提升其多智能体系统的可靠性与推理能力。通过训练模型生成这一推理路径，我们增强了其处理复杂多步骤请求并遵循合理布局布线流程的能力。最终的 EDA-Gen 模型既能理解布局布线的自然语言需求，又能生成语法正确、功能恰当的 Innovus Tcl 脚本，并附带逐步推理轨迹。这种两阶段方法预计将显著优于未经过这一专门领域自适应的基线模型。

实现细节 训练使用 LLaMA Factory [24] 框架，在由八块 NVIDIA A800 GPU 组成的集群上进行。训练过程分为两个阶段。在初始的领域自适应持续预训练（CPT）阶段，模型使用原始 Tcl 脚本语料训练 2 个 epoch，学习率为 2×10^{-5}，全局批量大小为 64，权重衰减为 0.1。随后，在监督微调（SFT）阶段，模型训练 1 个 epoch，全局批量大小为 64，学习率为 5×10^{-6}，权重衰减为 0.1。

3.3 评估策略

全面的基准和自动化评估系统是衡量大模型代码生成任务的重要组成部分。然而，在 EDA 脚本生成领域，这类基准和系统尚付阙如。为解决这一问题，我们一方面提出了一个覆盖广泛领域且包含多个难度级别的基准。另一方面，由于完整设计的验证耗时较长，且设计可能无法满足所有需求，因此难以建立能够全自动验证脚本功能的系统。因此，我们提出了一种两步验证策略：首先进行静态语法检测，然后基于大语言模型开展功能评估。

3.3.1 iScript-Bench。为了建立一个真实且全面的基准，我们从多个来源收集并整理了脚本，包括专家编写的脚本、现实需求以及社区资源。完成脚本收集后，我们将对这些脚本进行静态语法检查和人工功能检查。基准的处理流程如图 3 所示。同时，根据需求的不同类型和脚本的实际难度，我们将通过人工功能检查的脚本划分为 5 个主要类别、25 个子类别和 3 个难度级别。难度分类信息见表 1。如表 1 所示，该基准的难度分布均衡。这种设计便于进行双层评估：不仅可以衡量模型对基础命令和参数的理解与应用能力，还可以评估其处理复杂脚本逻辑和复杂命令序列的熟练程度。分类信息如下：

- **设计信息查询与调整（DIQA）**：查询设计中的各种对象，包括时钟路径、数据路径以及具有特定特征的对象。
- **网表信息分析与调整 (NIAA)**：检索网络属性、检查网络连通性并调整网络参数。
- **布局规划调整**：分析并优化宏单元、全局布局和端口。
- **布局分析与优化 (PAO)**：分析并优化标准单元布局约束。
- **时序分析与优化 (TAO)**：调整时钟路径和数据路径的拓扑结构及参数，以实现优化。

这五类任务涵盖了布局规划阶段的主要操作，可全面评估模型理解这些操作并生成语法正确且功能适当代码的能力。

3.3.2 两步验证策略。评估 EDA 脚本的准确性面临独特挑战。完全自动化、基于执行的框架往往难以实现，而纯粹依赖人工评分则无法扩展。因此，我们提出一种两步验证策略，包括静态语法验证和基于大语言模型的功能评估。

静态语法验证：虽然为 EDA脚本 建立一个完全自动化的功能验证系统在计算上难以承受，但确保语法正确性是可行的，且

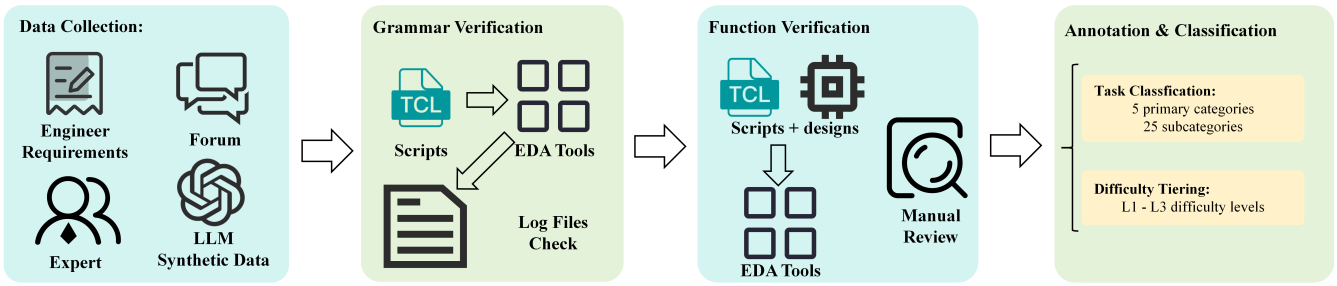

图 2：基准收集

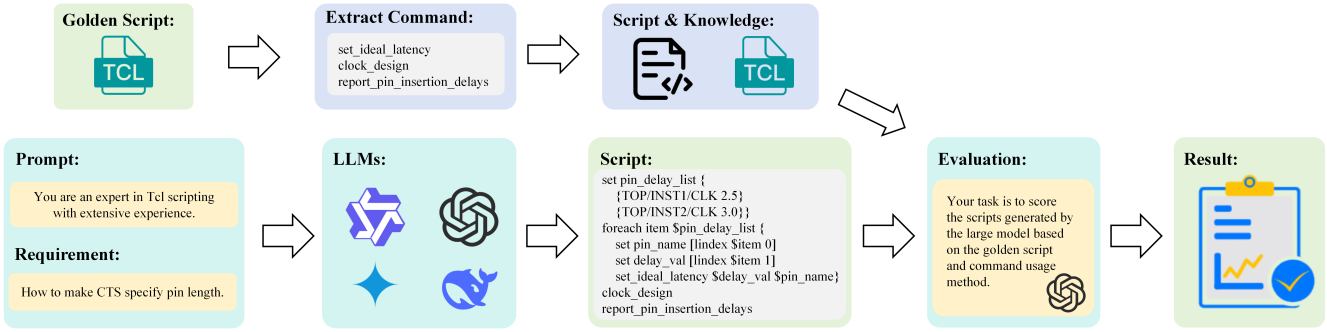

图 3：评估流程。

表 1：难度分级

级别	编号	描述
L1	54	单行代码或单条命令的简单用法。
L2	36	多条命令及其参数的组合
L3	26	不仅需要组合多条命令及其参数，还需要具备基础的代码逻辑

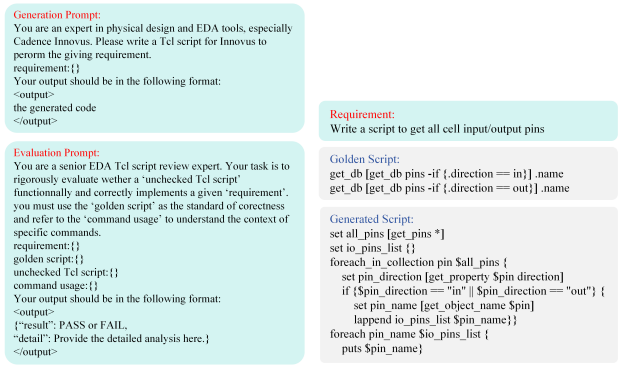

图4：（a）提示模板和（b）评估示例。

关键的第一步。我们设计了一个轻量级沙盒环境

基于最小化的物理设计布局执行静态语法检查。在此过程中，生成的脚本会在沙箱中执行，并解析生成的工具日志。具体而言，如果日志中包含模式“** ERROR:”，则将该脚本标记为语法无效；否则，视为语法正确。

基于大语言模型的功能评估：通过静态语法检查的脚本将进入基于大语言模型的功能评估。此阶段的主要挑战是确保大语言模型评判的可靠性。为此，我们采用知识增强提示策略。我们不仅向评估用大语言模型提供原始设计需求和“黄金”（参考）脚本，还提供从官方用户手册中提取的精确命令定义和对象属性描述。这一增强上下文使模型能够准确理解编程逻辑，并验证生成的脚本是否实现预期功能。为进一步验证该指标的可靠性，我们分析大语言模型评判与人类专家评审之间的一致性，详见第4节。

表2：不同类别的 Pass@1（%）。

任务	DIQA（35）		NIAA（10）		FA（22）		PAO（21）		TAO（28）		总计（116）	
	语法	函数	语法	函数	语法	函数	语法	函数	语法	函数	语法	函数
GPT	8.57	8.57	0.00	0.00	0.00	0.00	4.76	0.00	17.86	0.00	7.76	2.59
Gemini	31.43	17.14	**60.00**	**30.00**	22.73	18.18	28.57	9.52	28.57	7.14	31.03	14.66
Claude	31.43	17.14	50.00	**30.00**	0.00	0.00	9.52	4.76	14.29	7.14	18.97	10.34
DeepSeek	11.43	2.86	50.00	20.00	4.55	0.00	9.52	0.00	3.57	0.00	11.21	2.59
iScript	**71.43**	**25.71**	30.00	10.00	**77.27**	**22.73**	**66.67**	**9.52**	**35.71**	**17.86**	**59.48**	**18.97**

表3：不同类别的 Pass@5（%）。

任务	DIQA（35）		NIAA（10）		FA（22）		PAO（21）		TAO（28）		总计（116）	
	语法	函数	语法	函数	语法	函数	语法	函数	语法	函数	语法	函数
GPT	45.71	25.71	30.00	0.00	0.00	0.00	19.05	9.52	28.57	0.00	26.72	9.48
Gemini	85.71	57.14	**80.00**	40.00	40.91	27.27	66.67	28.57	**85.71**	35.71	73.28	39.66
Claude	60.00	45.71	60.00	**50.00**	9.09	0.00	28.57	14.29	21.43	10.71	35.34	23.28
DeepSeek	40.00	17.14	60.00	30.00	4.55	0.00	23.81	4.76	28.57	0.00	29.31	8.62
iScript	**100.00**	**71.43**	**80.00**	20.00	**95.45**	**40.91**	**100.00**	**33.33**	75.00	**39.29**	**91.38**	**46.55**

表 4：不同难度下的通过率（%）。

难度	L1				L2				L3			
	语法		函数		语法		函数		语法		函数	
指标	Pass@1	Pass@5	Pass@1	Pass@5	Pass@1	Pass@5	Pass@1	Pass@5	Pass@1	Pass@5	Pass@1	Pass@5
GPT	9.26	29.63	5.56	16.67	11.11	36.11	0.00	5.56	0.00	15.38	0.00	0.00
Gemini	25.93	74.07	12.96	59.26	41.67	72.22	**22.22**	27.78	26.92	73.08	7.69	15.38
Claude	27.78	37.04	20.37	33.33	16.67	44.44	2.78	22.22	3.85	19.23	0.00	3.85
DeepSeek	12.96	38.89	3.70	12.96	13.89	27.78	2.78	8.33	3.85	11.54	0.00	0.00
iScript	**66.67**	**94.44**	**27.78**	**68.52**	**47.22**	**88.89**	11.11	**33.33**	**61.54**	**88.46**	**11.54**	**19.23**

4 实验

4.1 实验设置

模型选择：在EDA脚本生成中，我们将所提出的iScript与四种最先进的大语言模型进行比较，包括GPT- 4. 1 [1], 2 5 [6], 4 5, 3 1 Gemini- . - [10] 2 5 pro、Claude-sonnet- . 和DeepseekV . 。在基于大语言模型的功能评估中，我们选择综合能力较强的Gemini- . - pro作为评判模型。对于这些模型，我们在脚本生成中将温度设为1，并使用最新版本的API。

提示模板：对于提示词，我们定义了角色、任务，并指定了输出格式。详细信息请参见图4（a）。

通过率：我们采用常用的 pass@k 作为评估指标。我们使用 pass@1 和 pass@5 来评估这些大语言模型和我们的 iScript 的性能。

4.2 实验结果

不同类别的结果：为了评估各种模型在不同类型设计任务上的能力，我们针对不同类别的任务进行了实验。结果如表2和表3所示。如表2和表3所示，iScript 在几乎所有类别中始终优于其他通用大语言模型（GPT、Gemini、Claude 和 DeepSeek）。在 Pass@1 指标方面（表2），iScript 的语法总分最高，为 59.48%；功能正确性总分为 18.97%。这显著领先于排名第二的 Gemini，后者相应得分为 31.03% 和 14.66%。iScript 在 FA 和 DIQA 类别中表现尤为突出，而 GPT 和 DeepSeek 等其他模型往往难以取得任何功能正确性得分（0.00%）。从 Pass@5 指标（表3）来看，性能差距进一步扩大。iScript 的语法总分高达 91.38%，功能正确性得分为 46.55%。

不同难度级别的结果：为了评估不同模型处理不同难度级别设计任务的能力，我们针对不同难度级别的任务进行了实验。结果如表 4 所示。如表 4 所示，随着难度从 L1 提高到 L3，整体表现通常会下降，但与其他模型相比，iScript 仍表现出更强的鲁棒性。在 L1（最简单）级别，iScript 表现突出，Pass@1 语法率为 66.67%，Pass@5 率为 94.44%。难度升至 L3（最难）时，大多数通用模型的表现大幅下滑；例如，Claude 和 DeepSeek 的 Pass@1 语法率降至约 3.85%。相比之下，iScript 在 L3 级别仍保持着 61.54% 的高 Pass@1 语法得分。尽管在 L3 级别，功能正确性对所有模型而言仍是一项挑战（大多数模型的得分为 0.00%

Pass@1 功能准确率），iScript 仍以 11.54% 的成绩领先。这表明，尽管复杂逻辑对所有模型而言都很困难，但与竞争模型相比，iScript 在处理高难度设计任务方面的优化效果显著更好。

人类与大语言模型之间的比较：为验证基于大语言模型的功能评估的可靠性，我们将其与人工验证进行了对比研究。我们从784个语法有效的输出中随机抽取了100个生成脚本，并提交给资深工程师审查。结果显示，大语言模型和人类专家分别判定其中39个和42个脚本在功能上正确。关键的是，大语言模型认可的39个脚本全部包含在人工验证认可的42个脚本之中。这种包含关系表明，我们的自动评估器具有近乎完美的精确率（没有假阳性），可作为严谨且可靠的模型性能下界估计器。尽管大语言模型评估器较为保守，但其高可靠性确保报告的通过率不会被夸大。在分析差异（即大语言模型拒绝、但专家认可的3个脚本）后，我们发现，这些脚本使用了不同于黄金参考的替代命令，实现了所需功能。图4(b)展示了一个具有代表性的假阴性案例：模型严格遵循参考逻辑，却因此惩罚了一种有效的变体。

5 结论

对于 EDA脚本生成而言，主要面临三项挑战：数据极度匮乏、缺少专有模型以及缺少高效的评估方法。为应对这些挑战，我们提出了 iScript，这是一种通过两阶段训练构建的专用于 EDA脚本生成的大语言模型。具体而言，我们首先采用基于多智能体的数据合成框架，生成大量高质量的（需求+思维链+脚本）配对数据。随后，我们以基础模型 Qwen3-8B 为基础，通过两阶段训练流程训练了 iScript。此外，为了有效且准确地评估生成性能，我们提出了两步验证策略：静态语法验证和基于大语言模型的功能评估。现有实验结果表明，iScript 效果显著。

局限性：总体而言，我们的工作存在两项主要局限。首先，现有训练数据量仍略显不足，导致 iScript 对命令代码之间组合关系的理解不够充分，难以处理复杂任务。其次，与人工评估相比，基于大语言模型的功能评估已显示出有效性，但自动评估系统才是最佳选择，因为它能够展示脚本实际执行的结果。因此，我们未来的工作是收集更多高质量数据，并建立自动评估系统。

参考文献

[1] Josh Achiam、Steven Adler、Sandhini Agarwal、Lama Ahmad、Ilge Akkaya、Florencia Leoni Aleman、Diogo Almeida、Janko Altenschmidt、Sam Altman、Shyamal Anadkat 等。2023年。GPT-4 技术报告。arXiv 预印本 arXiv:2303.08774（2023年）。 [2] Mohammad Akyash、Kimia Azar 和 Hadi Kamali。2025年。RTL++：用于 RTL 代码生成的图增强型大语言模型。arXiv 预印本 arXiv:2505.13479（2025年）。[3]Jacob Austin、Augustus Odena、Maxwell Nye、Maarten Bosma、Henryk Michalewski、David Dohan、Ellen Jiang、Carrie Cai、Michael Terry、Quoc Le 等。2021年。利用大语言模型进行程序合成。arXiv 预印本 arXiv:2108.07732（2021年）。 [4] Cadence Design Systems, Inc. 2017年。Innovus 实现系统命令参考手册。Cadence Design Systems, Inc. Cadence 专有文档。 [5] Mark Chen。2021年。评估基于代码训练的大语言模型。arXiv 预印本 arXiv:2107.03374（2021年）。[6] Gheorghe Comanici、Eric Bieber、Mike Schaekermann、Ice Pasupat、Noveen Sachdeva、Inderjit Dhillon、Marcel Blistein、Ori Ram、Dan Zhang、Evan Rosen 等。2025年。Gemini 2.5：借助先进推理、多模态、长上下文和新一代智能体能力推动前沿发展。arXiv 预印本 arXiv:2507.06261（2025年）。[7] Ahmed Elnaggar、Wei Ding、Llion Jones、Tom Gibbs、Tamas Feher、Christoph Angerer、Silvia Severini、Florian Matthes 和 Burkhard Rost。2021年。CodeTrans：通过自监督深度学习和高性能计算，探索硅芯片代码的语言。arXiv 预印本 arXiv:2104.02443（2021年）。 [8] Shan Huang、Jinhao Li、Zhen Yu、Jiancai Ye、Jiaming Xu、Ningyi Xu 和 Guo-hao Dai。2025年。LLSM：采用 EDA 引导的思维链提示、混合嵌入和针对 AIG 的加速的逻辑综合增强型大语言模型。在第30届亚洲及南太平洋设计自动化会议论文集中。974–980。 [9] Junyan Li、Sam-Zaak Wong、Gwok-Waa Wan、Xi Wang 和 Jun Yang。2025年。EDA-Debugger：一种用于自动解决 EDA 运行时问题的基于大语言模型的框架。在2025年第26届国际高质量电子设计研讨会（ISQED）上。IEEE，1–7。[10] Aixin Liu、Bei Feng、Bing Xue、Bingxuan Wang、Bochao Wu、Chengda Lu、Chenggang Zhao、Chengqi Deng、Chenyu Zhang、Chong Ruan 等。2024年。DeepSeek-V3 技术报告。arXiv 预印本 arXiv:2412.19437（2024年）。[11] Bingyang Liu、Haoyi Zhang、Xiaohan Gao、Zichen Kong、Xiyuan Tang、Yibo Lin、Runsheng Wang 和 Ru Huang。2025年。LayoutCopilot：用于交互式模拟版图设计的基于大语言模型的多智能体协作框架。IEEE集成电路与系统计算机辅助设计汇刊（2025年）。 [12] Jiawei Liu、Chunqiu Steven Xia、Yuyao Wang 和 Lingming Zhang。2023年。你的代码真的是 ChatGPT 生成的吗？对代码生成大语言模型的严格评估。《神经信息处理系统进展》第36卷（2023年），21558–21572。 [13] Mingjie Liu、Teodor-Dumitru Ene、Robert Kirby、Chris Cheng、Nathaniel Pinckney、Rongjian Liang、Jonah Alben、Himyanshu Anand、Sanmitra Banerjee、Ismet Bayraktaroglu 等。2023年。ChipNeMo：面向芯片设计的领域适配型大语言模型。arXiv 预印本 arXiv:2311.00176（2023年）。 [14] Mingjie Liu、Nathaniel Pinckney、Brucek Khailany 和 Haoxing Ren。2023年。VerilogEval：评估用于 Verilog 代码生成的大语言模型。在2023年 IEEE/ACM国际计算机辅助设计会议（ICCAD）上。IEEE，1–8。[15] Shang Liu、Wenji Fang、Yao Lu、Jing Wang、Qijun Zhang、Hongce Zhang 和 Zhiyao Xie。2024年。RTLCoder：完全开源且高效的 LLM 辅助 RTL 代码生成技术。IEEE集成电路与系统计算机辅助设计汇刊（2024年）。[16] ZhaoyangZhang 等。Ning Xu。2024 年。ChipExpert：首个开源的集成电路设计专用大语言模型。arXiv 预印本 arXiv:2408.00804v1（2024年）。[17] Ghasem Pasandi、Kishor Kunal、Varun Tej、Kunjal Shah、Hanfei Sun、Sumit Jain、Chunhui Li、Chenhui Deng、Teodor-Dumitru Ene、Haoxing Ren 等。2025年。JARVIS：用于高质量 EDA 脚本生成的多智能体代码助手。arXiv 预印本 arXiv:2505.14978（2025年）。 [18] Siyu Qiu、Benjamin Tan 和 Hammond Pearce。2024年。借助大语言模型解释 EDA 综合错误。在2024年 IEEE 大语言模型辅助设计研讨会（LAD）上。IEEE，1–6。[19] Synopsys, Inc. 2016年。IC Compiler II用户指南。Synopsys, Inc. Synopsys 专有文档。 [20] Yue Wang、Weishi Wang、Shafiq Joty 和 Steven CH Hoi。2021年。CodeT5：面向代码理解与生成的标识符感知统一预训练编码器—解码器模型。arXiv 预印本 arXiv:2109.00859（2021年）。[21] Haoyuan Wu、Zhuolun He、Xinyun Zhang、Xufeng Yao、Su Zheng、Haisheng Zheng 和 Bei Yu。2024年。ChatEDA：由大语言模型驱动的 EDA 自主智能体。IEEE集成电路与系统计算机辅助设计汇刊 43卷第10期（2024年），3184–3197。[22] Haoyuan Wu、Haisheng Zheng、Zhuolun He 和 Bei Yu。2025年。《殊途同归：基于大语言模型的多智能体协作系统，用于

电子设计自动化。*arXiv* 预印本 *arXiv:2502.10857*（2025年）。[23] An Yang、Anfeng Li、Baosong Yang、Beichen Zhang、Binyuan Hui、Bo Zheng、Bowen Yu、Chang Gao、Chengen Huang、Chenxu Lv 等。2025年。Qwen3 技术报告。*arXiv* 预印本 *arXiv:2505.09388*（2025年）。

[24] Yaowei Zheng、Richong Zhang、Junhao Zhang、Yanhan Ye、Zheyan Luo、Zhangchi Feng 和 Yongqiang Ma。2024年。LlamaFactory：统一高效的100+ 语言模型微调。载于计算语言学协会第*62*届年会论文集。计算语言学协会，泰国曼谷。http://arxiv.org/abs/2403.13372