TeleDoCTR：电信领域专用与上下文故障排除

Mohamed Trabelsi
m
ohamed.trabelsi@nokia-bell-labs.com
Nokia Bell Labs 默里山，新泽西州，美国

Huseyin Uzunalioglu
huseyin.uzunalioglu@nokia-bell-labs.com
Nokia Bell Labs 韦斯特福德，马萨诸塞州，美国

arXiv:2601.00691v1 [cs.LG] 2 Jan 2026

摘要

工单故障排查是指分析并解决通过工单系统上报的问题的过程。对于提供多种服务的大型组织而言，由于提交的工单种类繁多，且需要专门的领域知识，这项任务极为复杂。尤其是在电信领域，故障排查非常耗时，因为专家需要解读工单内容、查阅文档并搜索历史记录，以确定合适的解决方案。这种依赖大量人工的方式不仅会延误问题解决，还会妨碍整体运营效率。为提高电信领域工单故障排查的有效性和效率，我们提出了 TeleDoCTR，这是一种面向电信领域、具有领域专属性并结合上下文的创新故障排查系统，专为电信领域的端到端工单解决而设计。TeleDoCTR 集成了领域专属排序和生成模型，可自动完成故障排查流程中的关键步骤：将工单路由至负责解决工单的相应专家团队（分类任务）、检索上下文和语义上相似的历史工单（检索任务），以及生成详细的故障分析报告，说明问题、根本原因和潜在解决方案（生成任务）。我们在来自电信基础设施的真实世界数据集上评估了 TeleDoCTR，结果表明，其性能优于现有最先进的方法，显著提升了故障排查流程的准确率和效率。

CCS 概念

•**计算方法** → **自然语言生成**；**排序**；**从示范中学习**。

关键词

工单故障排查、大语言模型、检索模型、指令微调

1 引言

服务管理系统广泛应用于各行各业，以确保复杂服务得到可靠交付和运行。电信领域就是一个典型例子，其中移动语音、消息服务、数据和家庭互联网等服务依托复杂基础设施，通过无线、光纤和基于互联网协议（IP）的网络提供。类似的服务管理系统也广泛应用于其他领域，包括信息技术、工业自动化、能源和交通运输。这些服务管理系统通常由两个主要部分组成：（1）监控和日志系统，持续收集服务基础设施的健康状况和性能数据；（2）诊断工具，用于分析这些数据，并据此推断系统健康状况、检测异常并识别潜在故障。从历史上看，解决服务问题的过程，即工单故障排查，很大程度上依赖人工专业知识。创建工单时，通常会包含简短的标题、对问题的自然语言描述，以及注明受影响系统组件的元数据。工程师随后会检索并分析系统诊断信息和日志，以调查问题并确定解决方案。工单故障排查是一项关键工作，因为它会直接影响客户满意度和服务可靠性。然而，这一过程仍然十分耗时，不仅因为解决每个问题本身较为复杂，也因为它会耽误其他工程任务。

现有许多工单故障排查方法 [5, 17, 18, 34,47]侧重于工单故障排查流程中的工单分类任务，例如分配类别、解决类别或解决时间。其他方法 [5, 9] 旨在识别重复或上下文相似的工单，以帮助解决问题。一些技术还会根据历史数据[18, 21]提供解决方案推荐。此外，基于大语言模型（LLM）的解决方案 [1, 26, 45] 已被引入，用于针对常见技术问题提供实时的对话式协助。尽管这些大语言模型在回答常见问题和提供通用支持方面效果良好，但在电信等专业领域往往表现不佳，因为这些领域需要领域专属知识、专业技能和推理能力。此外，现有大多数方法只处理故障排查流程中的单个环节——分类、检索或生成——而没有提供集成统一的故障排查解决方案。这种碎片化的方法给构建端到端系统带来了显著负担，而此类系统需要能够在复杂且具有领域专属性的环境中，全面自动化并支持工单故障排查工作流。

受近年来领域专属大语言模型发展成果的启发 [10, 35, 48, 59, 64], ，我们提出了一种新颖的电信相关领域专用且上下文故障排查（***TeleDoCTR***）系统，专为电信领域的端到端工单解决而设计。TeleDoCTR融合了领域专属排序器和生成模型，可自动完成完整的故障排查流程，包括将工单路由至负责解决该工单的适当专家团队；检索与给定工单在上下文和语义上相似的历史工单，以纳入更广泛的上下文信息；最终生成详细的故障分析报告，概述问题、根本原因和潜在解决方案。TeleDoCTR架构由四个关键组件构成：(1) 用于工单相似度计算和生成的故障分析报告评估的领域专用排序器；(2) 用于预测负责处理该工单的解决团队的分类模块；(3) 使用

微调、多响应生成和多响应排序的故障分析生成模块；以及 (4) 通过示例选择增强的检索增强生成（RAG）对话模块。为实现高效的领域自适应，TeleDoCTR通过QLoRA [12, 25] 对工单分类和故障分析生成采用参数高效微调（PEFT）。生成模型会生成多个候选故障分析报告，经过微调的领域专用排序器则评估这些报告与输入工单之间的语义相关性。随后，排名最高的报告将返回给用户。此外，领域专用排序器还用于为RAG模块选择相关示例，从而实现更准确且更具上下文感知能力的多轮故障分析生成。我们使用大规模电信相关故障排查数据集对TeleDoCTR进行训练和评估，该数据集体现了电信故障排查场景固有的复杂性和多样性。实验结果表明，TeleDoCTR显著优于通用大语言模型和孤立的解决方案，在真实电信环境中实现了更有效、更高效的工单解决。

总之，我们做出了以下贡献：

- 我们提出了一种新的电信领域专用且结合上下文的故障排查系统（TeleDoCTR），该系统将领域适配排序模型与生成模型相结合，以统一、集成的解决方案自动完成工单故障排查的三个关键步骤。
- 我们在故障排查数据集上对专用排序模型进行微调，以衡量工单相似性和故障分析评估中的语义相关性。我们还使用 QLoRA 进行指令微调，以微调一个领域专用大语言模型，用于工单分类和故障分析生成。
- 我们开发了一种响应排序机制：由经过微调的大语言模型生成多个故障分析报告，再使用领域专用排序器对其进行排序，以识别最相关的故障分析报告。
- 我们利用领域专用排序模型选择示例（工单-故障分析对），并将其纳入 RAG 对话模块，以改进多轮故障分析生成。
- 我们使用大规模电信故障排查数据进行训练和评估，并证明我们的新系统在三个关键步骤中的表现均优于基线方法。

2 相关工作

2.1 工单故障排查

工单故障排查是指分析并解决通过工单系统提交的问题。这是电信、信息技术、工业自动化、能源和交通运输等行业中的一项关键流程。现有方法 [1, 5, 17, 18,34, 47]主要侧重于分类任务，例如在更广泛的故障排除流程中，预测负责解决工单的团队、解决类别或预计解决时间。虽然这些方法有助于简化工单路由，但在复杂问题解决所需的可解释性和深度方面仍有所欠缺。我们的系统突破了传统分类方法，通过生成详细的自然语言故障分析报告，实现了更进一步的处理。这些报告会明确说明所识别的问题、其根本原因以及解决方案步骤。与分配离散标签不同，这种转变能够生成易于人类理解的输出，从而提升故障排查的实用性和可信度，并缩小机器生成的响应与人类水平理解之间的差距。

一些现有方法 [5, 9] 通过应用传统信息检索技术和语义相似度模型，识别此前已解决的工单或推荐相关的故障排除案例，从而探索了工单检索任务。这些方法的主要目标是检索在上下文或语义上与新工单相似的历史示例，以加快解决流程，使工程师能够重复利用以往的知识和解决方案。我们的系统将这一范式向前推进了一步，在检索过程中嵌入了解释层。我们的系统并非只是呈现相似工单，而是利用大语言模型主动分析检索到的信息，并结合当前问题对其进行语境化处理。这包括识别相似之处、提取相关诊断模式，以及推断以往的故障分析报告如何为解决当前工单提供参考。通过明确建立历史案例与当前工单之间的联系，我们的系统将检索从被动的参考工具转变为主动的决策支持机制。

大语言模型已被用于故障排查领域 [1, 26,45], ，主要旨在为用户提供实时对话支持，帮助其解决常见技术问题。这些系统通常能够有效处理常见问题，并基于通用知识提供协助。然而，在电信等专业领域，它们的实用性会大幅下降，因为故障排查需要深厚的领域专业知识、复杂的推理能力，以及解读特定情境数据的能力。基于广泛知识库训练的通用大语言模型缺乏必要的上下文意识，难以解决电信基础设施和运营中遇到的各种技术问题。我们的解决方案采用基于真实世界故障排除数据训练的领域微调模型，使系统能够紧密契合故障排查领域的专属知识。

2.2 领域专属大语言模型

深度上下文化语言模型（例如 BERT [14] 和 Ro-BERTa [33],）已被提出，用于解决信息检索（IR） [7, 11, 30, 39, 40, 46, 55, 56] 和自然语言处理（NLP） [53, 54, 57, 58, 60, 62]中的多种任务。近年来，研究人员致力于借助生成式预训练变换器（GPT）模型，提升大语言模型在多种任务中的能力 [20, 37, 50, 63, 68]。为此，人们开发了 ChatGPT、LLaMA 和 Mistral 等多种通用大语言模型，使其能够处理多种任务。尽管通用大语言模型在广泛的任务中表现出色，但在数学、金融、医疗保健和电信等需要深厚专业知识的领域，其效能会显著下降。这一局限源于其训练数据的特性：训练数据通常由大规模异构文本来源构成。因此，这些模型往往缺乏领域特定应用所需的精准性和上下文理解能力。为克服这一挑战，领域专属大语言模型应运而生，成为一种颇具前景的解决方案。通过使用经过筛选的领域特定数据集对通用模型进行微调，这些专用模型能够优化自身，从而掌握

特定领域独有的专业语言和推理模式。这种专门化显著提升了大语言模型在专家级场景中的性能和可靠性。

为了解决通用大语言模型在需要深厚领域专业知识的任务中存在的局限，越来越多的领域专用大语言模型被提出。例如，BloombergGPT[64] 仅使用金融数据进行训练，以增强其对金融领域文本的理解与生成能力。在数学领域，WizardMath [35] 旨在解决通用模型往往难以应对的复杂数学问题。AlphaGeometry [59] 是另一种数学领域专用模型，通过结合监督学习与强化学习（RL），专门解决奥林匹克级几何问题。在医学领域，MedPaLM-2 [48] 已被开发用于处理和解读医学文本、临床记录及健康记录，并具备很高的准确率和相关性。同样，ChatLaw [10] 使用法律文件和法律案件进行微调，以有效执行法律推理。

大语言模型最近在电信行业引起了广泛关注，因为它们有望处理复杂的领域专属任务。它们在故障链追踪方面展现出了令人印象深刻的能力，可用于识别网络中的故障组件 [8], ，以及对技术文档进行分类 [3]。此外，大语言模型也已在6G网络的上下文中得到探索，可支持网络架构的动态选择、部署、适配和创建 [51]。这与新兴的AI互联概念高度契合 [51], ，该概念侧重于优化跨网络组件的人工智能驱动运营。此外，还有研究提出利用大语言模型自动生成网络配置 [38, 61], ，并促进基于FPGA的无线硬件开发 [15]。最后，电信相关的大语言模型已用于处理电信领域的问题回答（QA）和代码生成 [4, 65, 69]。尽管这些任务具有价值，但它们通常具有较强的通用性，通用大语言模型往往已经具备对电信相关概念的广泛理解，因此能够较好地处理这些任务。然而，电信相关工单故障排查需要深厚的领域专属知识，而这些知识在公开数据集中并不存在。

3 问题陈述

完整的工单故障排查流程由以下环节组成：将工单路由至负责解决该工单的相应专家团队（分类任务）；检索上下文和语义上相似的历史工单（检索任务）；以及生成一份详细的故障分析报告，概述问题、根因和潜在解决方案（生成任务）。在训练 TeleDoCTR 的工单故障排查模型时，给定了两个数据集：

- 第一个数据集 $D_t = \{(T_1, t_1), (T_2, t_2) \ldots, (T_{|D_t|}, t_{|D_t|})\}$ 包含 $|D_t|$ 个工单-团队对，其中 T_i 是工单，$t_i \in \mathbf{C}$ 是标签集 $\mathbf{C}$ 中的团队标签。
- 第二个数据集 $D_f = \{(T_1, f_1), (T_2, f_2) \ldots, (T_{|D_f|}, f_{|D_f|})\}$ 包含 $|D_f|$ 个工单-故障分析对，其中 T_i 是工单，f_i 是故障分析。

每个工单 T_i 由以下内容组成：(1) 若干属性 A_i，例如产品名称、软硬件构建版本、标题和问题描述；以及 (2) 从附加日志文件中选取并关联到工单的日志片段 L_i 日志文件。软件日志广泛用于各种诊断和自动化任务，因为它们通常是记录软件运行时信息的主要来源。收集到的日志用于多种日志挖掘任务，例如异常检测 [22], 、故障预测 [6], 和故障诊断 [28]。然而，每个日志文件中通常都包含大量无关信息，这些信息会掩盖可用于日志挖掘下游任务的相关信号。人工辅助的日志行选择工具用于处理收集到的日志文件，并输出经过筛选的相关日志片段 L_i。形式上，每个工单 T_i 定义如下：

$$T_i = A_i \oplus L_i \tag{1}$$

其中，$\oplus$ 表示字符串拼接操作。

每份故障分析 f_i 分为三个部分：识别I_i，包括故障的调试过程；根本原因RC_i，即从给定根本原因集合中进行的分类；以及解决方案 R_i，由故障的可能解决方案组成。形式化地说，每份故障分析 f_i 定义如下：

$$f_i = I_i \oplus RC_i \oplus R_i \tag{2}$$

训练完成后，TeleDoCTR 模型通过预测应负责解决工单的团队、检索相似的历史工单，以及生成详细说明问题应如何解决的故障分析报告，来处理给定的未见工单 T_u 。

4 TELEDOCTR：工单故障排查系统

本节介绍我们的工单故障排查系统 TeleDoCTR。该系统以工单为输入，并自动执行故障排查工作流中的关键步骤，包括：将工单路由至适当的专家团队（分类任务）、检索在上下文和语义上相似的历史工单（检索任务），以及生成详细的故障分析报告，概述问题、根本原因和潜在解决方案（生成任务）。如图 1 所示，TeleDoCTR 由四个关键组件构成：(1) 领域专用排序器；(2) 用于工单路由的微调生成模型；(3) 支持多种故障分析生成方式并进行排序的微调生成模型；以及 (4) 结合示例选择的增强型基于 RAG 的故障分析生成模块。我们的领域专用故障排查生成模型基于多 LoRA 方法构建，为每项任务配置一个专用适配器，同时使用相同的基础大语言模型，记为 M。采用具有多个任务专用适配器的共享模型，可以实现单一部署，并根据所选任务动态激活适当的适配器。因此，TeleDoCTR 能够高效地训练和推理完整的故障排查流程。在接下来的章节中，我们将介绍如何在电信相关故障排查的上下文中训练这四个组件。

4.1 领域专用排序器

评估工单之间的相似度是工单故障排查的重要组成部分，有助于问题解决。历史工单包含有关以往工单如何解决的宝贵信息，可显著简化新工单的整个故障排查流程。工单之间的相似度可以

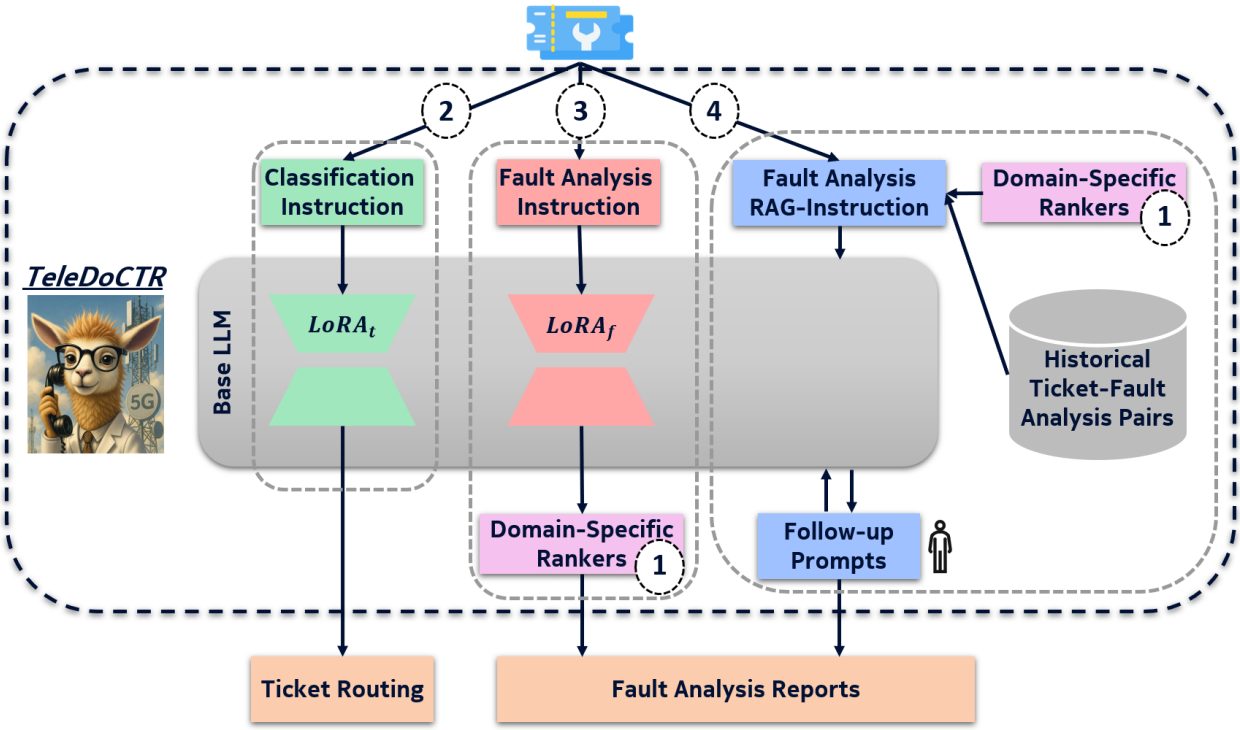


图 1：TeleDoCTR 概览，包含四个关键组件：(1) 领域专用排序器（排序）；(2) 用于工单路由的微调生成模型（分类）；(3) 具备多项故障分析生成与排序功能的微调生成模型（生成）；以及 (4) 通过示例选择增强的基于 RAG 的故障分析生成（生成）。

通过多种方式定义，例如是否包含一些共同的关键词、某些工单属性是否相似等。在我们的模型中，我们捕捉了与故障排查更相关的相似度信号，并将其定义为故障分析相似度。换言之，按照我们的相似度定义，如果两个工单具有相似的故障分析，就将它们视为相似。为此，我们从数据集 D_f 中构建两种类型的配对，用于训练领域专用排序器，记为 R。一项故障分析 f_i 可以关联多个工单 $\mathrm{T}_i = \{T_i^j,\ j = 1,\ 2,\ \ldots\ldots,\ n_i\}$，因为这些工单具有相似的症状，因此领域专家会将它们归入同一项故障分析。第一种配对与隐式工单相似度有关，由工单-故障分析对构成，取自 T_i 和 f_i，即 $IS_i = \{(T_i^j,\ f_i)\ j = 1,\ 2,\ \ldots\ldots,\ n_i\}$。由所有唯一故障分析构成的完整工单-故障分析数据等价于 D_f（$\bigcup_{i=1}^{n} IS_i = D_f$，其中 n 是唯一故障分析的总数。第二种配对与显式工单相似度有关，由工单-工单对构成，取自 T_i，即 $ES_i = \{(T_i^j,\ T_i^k);\ j < k;\ j,\ k = 1,\ 2,\ \ldots\ldots,\ n_i\}$。$ES_i$ 中的配对数量为 $|ES_i| = {}^{n_i}(n_i - 1)\ 2$。由所有唯一故障分析构成的完整工单-工单数据定义为 $\mathrm{ES} = \bigcup_{i=1}^{n} ES_i$。用于训练排序模型 R 的最终领域特定数据定义如下：

$$\mathcal{S} = D_f \cup \mathcal{ES} \tag{3}$$

工单和故障分析可视为不同的基于文本的模态。通常，模态对齐已在相关文献中得到充分研究，并受益于基于 Transformer 的架构，例如 CLIP [42], Florence [66], ALIGN [27], MPNet [49], TSLM [52], 等。在我们的领域专用排序器中，训练目标是将每种模态的嵌入（工单模态记为 T_m，故障分析模态记为 FA_m）投影到联合嵌入空间，以加深对模态间关系的理解，并确保表示连贯一致。这样，我们便能够评估投影到共享嵌入空间中的模态之间的相似度（工单与工单之间，以及工单与故障分析之间的相似度）。

工单 T 的嵌入表示为 $\boldsymbol{T} = R(T) \in \mathbb{R}^d$，故障分析 f 的嵌入表示为 $\boldsymbol{f} = R(f) \in \mathbb{R}^d$。我们的目标是创建一个嵌入空间，使相关的工单-工单对和工单-故障分析对的相似度高于不相关的对。为了使用 S 训练基于文本的跨模态稠密检索模型 R，我们需要创建正样本对和负样本对。对于给定的对 $(T,\ O)$，其中 $O \in \{\mathrm{T}_m,\ \mathrm{F\ A}_m\}$，我们将负实例表示为 $O_1^-,\ O_2^-$、……、O_{ng}^-，其中 $O_j^- \in \{\mathrm{T}_m,\ \mathrm{F\ A}_m\}$，$ng$ 为负例的数量。我们通过最小化相关（正样本）实例 O 的对数似然损失，来更新稠密检索模型 R 的参数：

$$L\left(T, O, O_1^-, \cdots, O_{ng}^-\right) = -\log \frac{e^{\frac{\mathrm{sim}(T,O)}{Temp}}}{e^{\frac{\mathrm{sim}(T,O)}{Temp}} + \sum_{j=1}^{ng} e^{\frac{\mathrm{sim}\left(T,O_j^-\right)}{Temp}}} \tag{4}$$

其中，$Temp$为温度值，sim (T, O) 表示工单 T与实例 $O \in \{T_m, FA_m\}$之间的相似度，该相似度通过它们向量的余弦相似度计算：

$$\text{sim}(T, O) = \frac{\boldsymbol{T} \cdot \boldsymbol{O}}{\|T\| \, \|O\|} \tag{5}$$

给定一个由来自 S 的 B 对组成的批次，我们使用批次内负样本 [29] 来训练跨模态稠密检索模型。设 $\boldsymbol{T}_B$ 和 $\boldsymbol{O}_B$ 分别为该批次中工单和实例的嵌入，维度为（$B \times d$），批次大小为 B。我们计算相似度分数矩阵 $\boldsymbol{SIM} = \boldsymbol{T}_B \boldsymbol{O}_B^T \in \mathbb{R}^{B \times B}$，其中每一行对应一个工单及其来自$\{T_m、FA_m\}$的 B 个候选实例。对角线位置上的实例表示真实实例（工单或故障分析），其余$B-1$ 个实例均为负样本。

鉴于训练 R时批次中负实例具有随机性，我们可以训练多个排序器 R_1、R_2、……、R_{rn}，其中 rn 为排序器总数。首先，这些排序器能够通过使用不同的负实例训练模型，捕捉不同的排序信号，因此由多个排序器计算出的聚合相关性分数涵盖了多个相似度方面。其次，假设对于检索集 S_i，给定排序 K 的 Recall@k 在所有排序器 R_i中均为 1。这意味着，对于所有排序器 R_i，真实相似工单都位于 S_i 中。因此，真实工单位于交集 $\bigcap_{i=1}^{rn} S_i$中。于是，交集 $\bigcap_{i=1}^{rn} S_i$的 Recall@k 也为 1，但其排序 K' 小于或等于 K（$K' = |\bigcap_{i=1}^{rn} S_i| \le |S_i| = K$；$i = 1$、2、$\ldots rn$）。因此，多个排序器的交集能够剔除并非所有排序器都检索到的条目，同时保持 Recall@k 为 1。

使用领域特定数据 S 训练基于文本的跨模态稠密检索模型后，将 R_1，R_2，…… R_{rn} 纳入微调模型和基于检索增强生成的模型，以提高故障分析生成的准确性并改进工单排序。

4.2 工单路由

传统上，工单路由是一项分类任务：使用多标签交叉熵损失函数训练机器学习模型，然后预测每张工单所属的团队。为高效使用带有多个任务专用适配器的共享大语言模型，我们将分类任务转化为生成任务，构建指令微调数据集，以微调大语言模型来执行工单路由或团队标签生成。对于每个 $I_j = (T_j, t_j) \in D_t$，我们采用图 2 所示的指令微调模板，生成用于团队标签生成的指令微调数据，记作 $INST_{D_t}$。预训练基础大语言模型记作 M，使用团队标签指令微调监督数据进行微调；对于 $INST_{D_t}$ 中的每个实例，在真实标注团队标签 t_j上采用教师强制执行下一个词元预测任务。为提高训练效率、节省时间和内存，我们向 M 中添加基于工单路由的 LoRA 层，记作 $LoRA_t$。训练期间，仅更新 $LoRA_t$参数，且损失仅根据团队标签部分计算交叉熵；基础模型 M的参数保持冻结。在工单路由的推理阶段，激活 $LoRA_t$ 层，使其与冻结的基础模型 M协同工作，从而生成团队标签，无需承担全模型微调的开销。

系统提示：你是一名工单解决与故障排除助手专家。

用户提示：预测负责解决给定工单的团队。

工单 $= T_i$

助手回复： t_i

图 2：用于微调团队标签生成模型的指令微调模板。

系统提示：你是一名专业的工单解决与故障排除助手。

用户提示：通过识别和解释问题及症状，分析给定工单，然后生成可能的根本原因和解决方案。

工单 $= T_i$

助手回复： f_i

图3：用于微调故障分析生成模型的指令微调模板。

4.3 多故障分析生成与排序

我们展示了如何在响应排序机制中利用排序模型 R_1，R_2、…… R_{rn}：由微调后的大语言模型生成多份故障分析报告，再使用特定领域的排序器对其进行排序，以识别最相关的报告。

4.3.1 微调用于故障分析生成的领域专用大语言模型。与工单路由类似，我们构建指令微调数据集，以微调大语言模型来生成故障分析。对于每个 $I_j = (T_j, f_j) \in D_f$，我们应用图 3 所示的指令微调模板，获得用于故障分析生成的指令微调数据，记为 $INST_{D_f}$。我们使用指令微调故障分析监督数据，对预训练基础大语言模型M进行微调；对于 $INST_{D_f}$ 中的每个实例，针对真实故障分析 f_j，采用教师强制执行下一个词元预测任务。与工单路由类似，我们将用于故障分析的 LoRA 层记为 $LoRA_f$，并将其添加到 M 中，在训练期间更新这些层，同时保持基础模型 M 的参数冻结。在故障分析生成的推理阶段，我们启用$LoRA_f$层，并将其与基础模型 M 结合使用，以生成详细且特定领域的故障分析。这种模块化设计只需替换相应的 LoRA 适配器，即可支持多个任务的可扩展部署。

4.3.2 故障分析报告的响应排序方法。使用领域特定故障排除数据进行微调，可以让基础大语言模型 M 掌握领域专业知识，并有效适应电信相关故障分析生成的独特特点。通过这一过程，我们的微调模型能够生成详细且考虑上下文的故障分析报告，并针对电信领域的复杂特点进行定制。然而，由于电信故障排查通常固有地复杂且存在歧义，微调模型可能会生成多个看似合理的故障分析候选项；每个候选项都从不同角度描述故障，确定性也各不相同。为应对这一挑战并确保生成的故障分析报告可靠，我们采用了第 4.1 节所述的领域专用排序器 R_1，R_2、……、R_{rn}。这些排序器经过训练，可评估每个故障分析候选项与给定工单之间的语义相关性和上下文对齐程度。通常，与生成相比，评分引入的不确定性更少。这是因为在评分过程中，完整的故障分析和相应工单都已预先提供，模型可以直接评估二者的语义相关性。相比之下，生成本质上是自回归的，即大语言模型逐词元生成故障分析，每一步都仅以先前生成的词元为条件。这种顺序生成的特性会带来不确定性，因为在生成过程中，模型无法获取最终输出的完整上下文。因此，通过计算相似度分数，排序器能够自动筛选并优先选出相关性最高、准确性最高的响应。这种排序机制可减少输出中的噪声，确保用户只收到可信的故障分析报告。

具体而言，对于未见工单 T_u，我们会生成多份故障分析报告 $FR = f_1, f_2, \ldots\ldots, f_P$，其中 P 是报告总数。对于每个排序器 R_i，工单 T_u 与故障分析报告 $f_j \in FR$ 之间的排序分数 RS_j^i 定义如下：

$$RS_j^i = \mathrm{sim}_i(T_u, f_j) \tag{6}$$

其中，sim_i 是按照方程 (5) 使用从排序器 R_i 提取的嵌入计算得出的相似度。排序分数 RS_j^i 用于评估工单与故障分析之间的相似度，这得益于 R_i 的共享嵌入空间，其中包含工单和故障分析的嵌入。未见工单 T_u 与 $f_j \in FR$ 之间的最终排序分数 RS_j 定义如下：

$$RS_j = \frac{\sum_{i=1}^{rn} RS_j^i}{rn} \tag{7}$$

最终排序分数 RS_j 是一个多排序器分数，用于捕捉通过训练多个排序器所获得的多种相似度信号。根据分数 RS_1，RS_2，……，RS_P，我们返回分数最高的故障分析报告。

4.3.3 使用基于自动反馈的强化学习进行模型对齐。先前的响应排序方法利用领域专用排序器 R_1、R_2、……、R_{rn}，旨在提升微调模型在推理阶段的故障分析生成效果。我们还研究了在训练阶段通过额外的基于强化学习的微调，使微调后的故障分析模型与领域专用排序器的偏好保持对齐的可能性。为此，我们提出 RLRF（基于排序器反馈的强化学习，Rankers Feedback），利用根据排序器得分构建的偏好数据集，进一步训练微调后的故障分析生成模型。我们获得了另一组工单-故障分析对，记作 D_f^{RL}，与 D_f 类似，但规模更小。对于每个工单 $T_i \in D_f^{RL}$，我们生成多份故障分析报告 $FR_i = f_1^i$、f_2^i、……、f_Z^i，其中 Z 为报告总数。然后，我们使用公式（7）计算每份故障分析的相关性得分，得到得分 RS_1^i、RS_2^i、……、RS_Z^i。如果两个候选响应的得分差超过阈值 τ，我们就构建一个偏好对，并将得分较低的响应选作拒绝示例。对于选定示例，我们以概率 p 使用真实故障分析 FA_{gt}，或以概率 $1-p$ 使用得分较高的生成响应。这样，我们构建了基于排序器的偏好数据集 P_{RL}，其中每个实例都是一个三元组（T_i、$chosen_i$、$rejected_i$）。

微调后的故障分析生成模型不仅必须学习复杂的故障排查任务，还必须适应其不规则的半结构化输出格式。这一双重挑战会导致模型出现一定程度的不稳定性，常常产生病态输出，例如退化案例 [31]。为解决这一问题，我们提出一种基于直方图的方法，用于检测退化模型输出。我们观察到，病态响应（例如字符、词元或数字过度重复的响应）呈现出异常偏斜的频率分布。我们的方法会比较生成的故障分析与其对应真实标注的归一化字符直方图；如果字符级频率差异超过阈值 Th，就会将该响应标记为病态。我们将检测到的病态示例作为负样本，扩充自动生成的偏好数据集 P_{RL}，用于 RLRF 微调。这些示例充当反例，引导模型生成结构良好且高质量的输出。我们还使用直接偏好优化（DPO）[43] 进一步微调 $LoRA_f$，同时冻结基础模型 M，以获得 $LoRA_{f+RLRF}$。

4.4 基于增强 RAG 的故障分析生成与示例选择

我们展示了如何利用排序模型 R_1、R_2、…… R_{rn} 来选择示例（工单-故障分析对），并将其整合到 M 的 RAG 对话模块中，以改进多轮故障分析生成。

4.4.1 示例选择。检索模型是基于 RAG 的架构中最重要的组成部分。检索模型以查询和已索引文档作为输入，并返回一组与查询相关的文档。查询分为两类：基于关键词的查询和基于内容的查询。与传统 RAG 不同，传统 RAG 使用通用的基于关键词的搜索，从大型文档语料库中提取相关段落；我们的任务具有很强的领域专用性，需要专门设计的查询、语料库和检索模型。我们采用领域专用的基于内容的搜索，以便为基于 RAG 的模型检索准确上下文。首先，查询是一个新的未见工单 T_u，由若干属性 A_u 和日志片段 L_u 组成，如

方程 (1) 所示。其次，已索引文档是来自 D_f 的领域专用工单-故障分析对。第三，检索模型是领域专用排序器 R_1，R_2，……，R_{rn}。

对于每个排序器 R_i，未见工单 T_u 与 j^{th} 历史工单-故障分析对 $I_j = (T_j, f_j) \in D_f$ 之间的排序分数 RS_j^i 定义如下：

$$RS_j^i = \text{sim}_i(T_u, T_j) + \text{sim}_i(T_u, f_j) \tag{8}$$

其中，sim_i 是使用从排序器 R_i 提取的嵌入，按照方程 (5) 计算得出的相似度。排序分数 RS_j^i 评估工单-工单相似度和工单-故障分析相似度，以获得更准确的检索分数。这种双层分数计算依托于 R_i 的共享嵌入空间，该空间同时包含工单和故障分析模态。T_u 与 j^{th} 实例 $I_j = (T_j, f_j) \in D_f$ 之间的最终排序分数 RS_j 定义如下：

$$RS_j = \frac{\sum_{i=1}^{rn} RS_j^i}{2 \times rn} \tag{9}$$

最终排序分数 RS_j 是一个多排序器、多模态分数，其中包含通过训练多个排序器和多种模态所捕获的多个相似度信号。根据分数 $RS_1, RS_2, \ldots, RS_{|D_f|}$，我们返回相似度最高的前 K 个实例集合 $SD = \{(T_r, f_r)\}_{r=1}^{K}$，其中 K 是检索实例的数量。SD 是示例集合（工单-故障分析对），会被纳入基于RAG的对话模块，以改进多轮故障分析生成。

4.4.2 基于 *RAG* 的故障分析生成。我们停用两个 LoRA 层 $LoRA_t$ 和 $LoRA_f$，并使用基础大语言模型 M 进行基于 RAG 的故障分析生成，以避免在特定任务上微调导致部分关键通用技能（摘要、推理等）发生灾难性遗忘 [36]。我们将减少灾难性遗忘、并使微调模型能够用于基于 RAG 的生成的技术留待未来研究。我们通过手动编写第一轮对话，引入检索步骤中选出的示例，引导基于 RAG 的对话模块为新的未见工单生成故障分析。第一轮对话中的 system、user 和 assistant 角色定义如图 4 所示。我们将 system 设定为工单解决与故障排除助手。对于未见工单 T_u，用户提示首先包含选出的示例 S_i，随后指示基础大语言模型 M 结合所提供的示例，为该工单 T_u 生成详细的故障分析。对于 assistant 的回复，我们对回复的开头词元进行硬编码，迫使模型在考察该工单 T_u 与选出的示例 SD 在症状和解决方案方面的相似性时，采用思维链（CoT）推理。随后，assistant M 会生成实际回复，如图 4 所示。完成第一轮引导式对话后，用户可以在包含完整第一轮对话的同一上下文中向 assistant 提出后续问题。部分后续问题包括要求深入分析与示例之间发现的相似之处、进一步说明建议的解决方案等。由于电信领域复杂，且需要深厚的技术专业知识，排查电信相关工单面临重大挑战。在许多情况下，可能存在多种解决方案，因此很难生成一个

系统提示：你是一位工单解决与故障排除助手专家。

用户提示：以下是一些示例，每个示例都由一张故障工单及其准确的故障分析组成：

Demonstration 1:
Ticket = T_1
Fault Analysis = f_1
Demonstration 2:
Ticket = T_2
Fault Analysis = f_2
⋮
Demonstration K:
Ticket = T_K
Fault Analysis = f_K

请结合所提供的示例，对这张新的故障工单进行非常详细的故障分析：

Ticket = T_u

请逐步思考这张新故障工单的故障分析！

助手回复：作为一名工单解决方案专家，以下是我结合所提供示例，对这张新工单进行的逐步详细故障分析：

[故障分析]

我的详细思维链从考察症状和解决方案方面与所提供示例的相似之处开始，然后我应借鉴相似示例中的故障分析部分，生成对新工单最合理的故障分析。

相似之处：**实际响应**

图 4：基于检索增强生成的模块第一轮故障分析生成提示。

确定的一次性解决方案。因此，仅依靠与大语言模型进行单次交互，往往无法解决问题。为此，我们的领域专用基于检索增强生成的模块支持迭代交互，使用户能够根据后续查询对大语言模型生成的回复进行完善、澄清和扩展。

基于微调模型和检索增强生成的故障分析报告分别呈现互补的视角，分别捕捉故障排查的细粒度和粗粒度方面。微调模型擅长生成精准的解决方案计划，因为它直接学习了从工单信息到故障分析报告的映射关系。

然而，这些细粒度报告可能缺乏更广泛的上下文理解，尤其是从历史相似工单中获得的见解。这正是基于检索增强生成的模块发挥价值之处：它通过检索语义相似的历史工单来丰富分析，并通过用户与大语言模型之间的多轮对话，以连贯且友好的方式呈现这些粗粒度上下文。通过将微调故障分析生成模块的针对性与基于检索增强生成的模块的上下文深度相结合，TeleDoCTR 提供了更全面、更有依据的故障分析，不仅解决当前问题，还将其置于更广阔的历史背景中。

5 评估

5.1 电信故障排除数据集

5.1.1 故障排查工单。当电信组件出现问题时，会创建工单 T_i 来详细描述问题。工单 T_i 由以下字段组成：

- 产品名称：有多种与电信相关的产品，例如*5G* 无线电、*WCDMA* 基站、*LTE* 基站等。此属性表示问题发生的位置。
- 硬件单元：它是产品中的一个物理实体，在报告的问题中被确定为问题来源或受影响的组件。
- 软件构建版本：它是部署在电信设备上的指定系统软件版本，包含决定设备功能行为的代码和配置。在工单故障排查中，软件构建版本用于识别与特定版本相关的潜在错误、兼容性问题或已知故障。
- 标题：这是一个简短的自然语言句子，用于概述问题。
- 问题描述：这是一段基于模板的长文本，包含多个字段，例如详细的测试步骤、测试步骤的预期结果、实际观察到的结果、测试人员的分析、故障描述及发生率、客户影响分析、所用标志等。
- 日志片段：工单中附有多种类型的日志，例如系统日志、运行时日志和事后分析日志。借助人工辅助的日志行选择工具处理收集到的日志文件，并输出一组相关日志片段 L_i。

5.1.2 故障分析报告。在解决工单的过程中，会编写故障分析报告，以记录解决问题的各个步骤。故障分析 f_i 由以下字段组成：

- 识别：这是故障分析报告中首先编写的部分，是一段较长的模板化文本，包含问题摘要、问题的技术描述、配置依赖，以及故障组件和版本信息。
- 根本原因：在识别部分之后确定，由实现错误、编译器错误、配置错误等类别的粗粒度分类组成。总共有 88 种可能的根本原因标签。
- 解决方案：这是故障分析报告的最后一部分，它根据输入工单以及已确定的识别和根本原因部分生成。它是一段中等长度的模板文本，包含用于工单解决的可能变通方案、修正说明和测试要求。

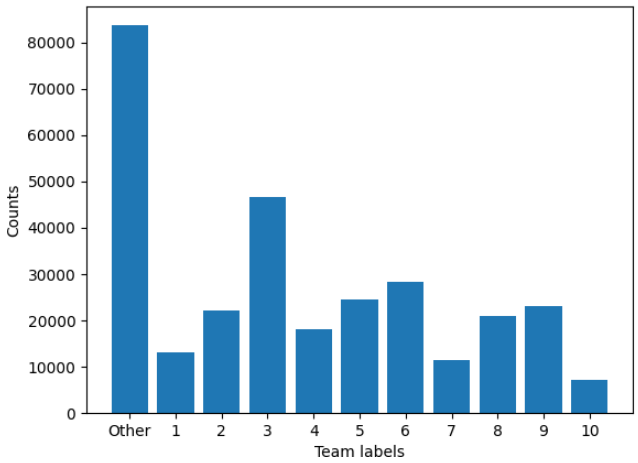


图5：团队标签分布。

为了训练有效的领域专用模型，用于排序和故障分析生成，确保训练数据包含信息充分、内容丰富的故障分析报告至关重要。为此，我们采用数据筛选步骤，过滤掉信息量低的样本。具体而言，给定一组故障分析报告 f_i，我们首先计算数据集中每个词元的逆文档频率（逆文档频率），以衡量词元的稀有程度和信息量。然后，对于每份报告 f_i，我们通过对报告中词元的逆文档频率最高的前 k 个值取平均，计算其信息量得分 IS_i。其基本思路是，主要由常见词元或模板化词元组成的报告会得到较低的 IS_i 值，而包含与故障排查细节相关的领域特定术语或罕见术语的报告则会得到较高的 IS_i 分数。接下来，我们应用严格的阈值 Th_f，仅保留超过最低信息量水平的报告；在筛选后的数据集中，我们优先考虑精确率而非召回率。在我们的实现中，前 k 等于 15， Th_f 等于 5，从而确保只选取信息量最高的样本。通过这一过滤流程，我们得到一个高质量数据集 D_f，其中包含跨越多年的 130,781 对数据。因此，用于微调 $LoRA_f$ 的基于故障分析的指令数据规模为 130,781。基于信息量的自动过滤对于减少噪声、提高学习效率至关重要，尤其是在电信故障排查这类复杂且具有领域特性的任务中。对于与多个工单相关联的 $f_i \in D_f$，我们构建工单-工单对数据集，记为 ES。ES 的规模为 71,570，因此用于训练领域专用排序器 R_1的 R_2、..... R_{rn}的领域特定数据 S $= D_f \cup$ ES 的规模为 202,351。

5.1.3 团队标签。 根据工单信息，每张工单都会分配给一个团队进行处理。团队标签集 C 由 11 个标签组成：10 个主要团队，以及一个额外的标签其他 ，该标签包含来自其余团队的工单。出于团队保密的考虑，我们无法提供团队名称。

数据集 D_t 由 299,495 个工单-团队对组成，用于微调工单路由层 $LoRA_t$。如图5所示，工单的分布并不均衡。

5.2 实验设置

5.2.1 领域专用排序器。我们将排序器总数 rn 设为 6。我们使用维度d 为 768 的 MPNet[1] 初始化每个排序器 R_i。该预训练模型适用于我们的场景，因为它已经在大量用于语义相似度的句对上完成训练。此外，MPNet 是一种轻量级模型，在训练和推理阶段都能兼顾效率与内存占用。较大的批量大小是训练有效排序器的关键，这样排序器才能捕捉相似性和差异性的多个方面。因此，我们将批量大小 B 设为 1024。计算排序损失时，我们汇总$\boldsymbol{SIM} = \boldsymbol{T_B O_B}^T \in \mathbb{R}^{B\times B}$和 $\boldsymbol{SIM}_{sym} = \boldsymbol{O_B T_B}^T \in \mathbb{R}^{B\times B}$的交叉熵，以获得能够捕捉两个方向上的正样本对和负样本对的对称损失。鉴于批量大小较大，无法通过单次前向传播计算嵌入 $\boldsymbol{O_B}$和 $\boldsymbol{T_B}$。GradCache [19]为训练期间大批量大小带来的内存瓶颈提供了高效的解决方案。它将计算分解为两个阶段——嵌入阶段和损失计算阶段，并可通过小批量进行扩展，从而减少内存占用。因此，固定大小的内存（在我们的场景中为小批量大小 = 100 ）现在可以处理大得多的批次（B = 1024）。对于排序器温度 $Temp$，我们将其设为 0.1，使方程 (4) 中的指数部分缩放 10 倍。训练集、验证集和测试集的大小分别为 145,693、16,188 和 40,470。我们使用加速器库在 3 × H100 GPU 上通过分布式数据并行（DDP）训练排序器。模型训练 15 个训练轮次，并保存验证集上损失最低的轮次。

5.2.2 领域专用生成模型。 为实现高效微调，我们将基础预训练大语言模型 M 设为 LLaMA-3-8B-Instruct[2] 。将其他开源轻量级大语言模型在电信相关故障排除数据上进行微调，将作为未来工作。表 1 列出了用于工单路由和故障分析生成的 QLoRA 指令微调超参数。两个生成模型均使用加速器库在 5 × H100 GPU 上通过分布式数据并行进行微调；考虑表 1 中的梯度累积步数后，实际批量大小为 40。

工单路由任务的训练集、验证集和测试集大小分别为 191,669、47,922 和 59,904。我们对模型训练 10 个轮次，并保存验证集损失最低的轮次所对应的模型。进行工单路由推理时，我们采用束宽为 2 的贪心搜索策略，以一致地生成排名前 2 的词元。这些词元对应于最有可能的前 2 个团队标签，其概率分布由微调模型根据 LLaMA 词表确定。

为对领域专用大语言模型进行故障分析生成微调，我们首先留出 1,000 张工单作为测试集，用于报告评估指标并与基线方法进行比较。其余工单-故障分析对用于微调，其中训练集和验证集的规模分别为 116,803 和 12,978。我们对模型训练 10 个轮次，并保存验证集损失最低的轮次

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[2] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

表1：工单路由和故障分析生成的指令微调超参数。

超参数	值
最大序列长度	10,000
学习率	5e-05
训练轮次	10
每个 GPU 的批量大小	1
梯度累积步数	8
学习率调度器	余弦
优化器	AdamW
损失函数	交叉熵损失
训练精度	bf16
预热比例	0.1
量化位数	4
LoRA Alpha	16
LoRA 目标模块	All
LoRA 秩	8

。对于用于故障分析生成的响应排序方法，我们针对测试集中的每张工单 T_i，在以下温度值（0.1、0.3、0.5、0.7、0.9）下各生成 5 份报告，因此报告总数 P 为 25。每次生成时，top-p [16] 和 top-k [24] 采样分别固定为 0.95 和 50。对于通过基于排序器反馈的强化学习进行故障分析模型对齐，共生成 Z 6 份报告，其中温度值为 0.1、0.5 和 0.9 时，分别生成 1、2 和 3 份报告。这样便能高效地构建基于排序器的偏好数据集 P_{RL}。我们将选定的故障分析被选中的概率 p 设为 0.5。我们为 P_{RL}生成两个版本的数据集。第一个版本 P_{RL}^1包含 5,399 个样本，且不将检测到的退化大语言模型响应用作拒绝样本。构建 P_{RL}^1时，我们将排序器分数差异阈值 τ 设为 0.3。第二个版本 P_{RL}^2 将病态示例作为负样本，并收集了 24,574 个实例用于 DPO 微调。构建 P_{RL}^2时，我们将病态样本的频率差异阈值 Th 设为 0.07。

对于增强型基于RAG的故障分析生成，我们通过固定示例的词元预算（在我们的案例中为 5,000 个词元），动态调整前 K 个相似工单，并持续添加示例，直到达到词元预算。我们将用于基于RAG的生成的基础大语言模型 M 的温度设置为 0.3。

5.3 实验结果

5.3.1 领域专用排序器。 在本部分中，我们研究 TeleDoCTR 中领域专用排序器 R_1、R_2、……、R_6 的性能。我们将领域专用排序器与 BM25 [44] 和原始 MPNet [49], 分别捕获的词汇匹配信号和语义匹配信号进行比较。我们使用 Recall@k评估领域专用排序器和基线方法的有效性，其中 $k \in$ {1、10、50、100、500、1000}。对于每个测试实例，Recall@k衡量真实工单是否出现在排名前k 的检索候选项中。如果真实工单位于排名前k 的候选项中，则记为 1 分；否则记为 0 分。最终的 Recall@k 通过对整个测试集中的这些分数取平均值计算得出。在我们的设置中，检索基于一个包含 130,781 张工单的索引进行

表 2：故障排除数据测试集上的工单检索评估结果。

模型	Recall@1	Recall@10	Recall@50	Recall@100	Recall@500	Recall@1000
BM25	0.072	0.113	0.234	0.275	0.371	0.412
原始 MPNet	0.000	0.000	0.000	0.001	0.004	0.008
单个领域专用排序器	0.200±0.012	0.400±0.017	0.571±0.021	0.649±0.012	0.821±0.008	0.880±0.005
单个领域专用排序器 + BM25	0.113±0.015	0.296±0.019	0.540±0.018	0.627±0.014	0.810±0.010	0.870±0.005
TeleDoCTR（多个领域专用排序器）	**0.251**	**0.461**	**0.627**	**0.700**	**0.853**	**0.901**

，这意味着 Recall@1 表示从 130,781 个候选项中正确检索出排名第一的结果。报告的结果是对包含 40,470 张工单的留出测试集取平均后得出的。表 2 展示了不同方法在故障排除数据测试集上的表现。结果清楚地表明，原始 MPNet 模型的语义匹配信号不适用于电信相关故障排查中的领域特定检索。预训练嵌入模型缺少特定领域故障排查所需的专门词汇和术语，因此检索结果不准确。我们计算了全部 6 个领域专用排序器的 Recall@k 分数，并报告平均分以及标准差。与基于语义的原始 MPNet 模型和基于词法的 BM25 模型相比，每个排序器 R_1、R_2、.....、R_6 的 Recall@k 分数都显著更高。我们在“单个领域专用排序器 +BM25”这一变体中结合语义和词法匹配信号，但与仅使用单个领域专用排序器相比，这并未提升检索分数。这意味着，对于我们的领域特定数据，传统的词法匹配信号无法补充学习到的领域特定语义匹配信号。然而，当我们按顺序取所有

6 K 2,500 2 500 = 1 R_1 R_2 ... R_6

排序器的交集，其顺序为 (Recall@ ，所有排序器的 、 、 、), 并使用方程 (9) 合并交集中的工单排序分数（记为 $\bigcap_{i=1}^{6} S_i$（S_i 是排序器 R_i 返回的集合，包含 2,500 张工单）），便能显著提升排序表现，优于每个单独的领域专用排序器。值得注意的是，Recall@K' ($K' = |\bigcap_{i=1}^{6} S_i|$) 仍为 1，这表明交集 $\bigcap_{i=1}^{6} S_i$ 保留了所有相关工单，同时剔除了未被所有排序器 R_1、R_2、.....、R_6 选中的工单。这一步筛选降低了噪声，提高了候选集的精确率。随后，我们进一步通过合并不同排序器的相关性分数来优化交集，其中每个排序器都能捕获语义相似度的不同方面。这种共识驱动排序确保排名最高的工单在多个排序器之间获得高度一致的评价，从而使所有评估层级上的 Recall@k 都得到持续提升。

5.3.2 工单路由。 本部分研究 TeleDoCTR 中工单路由模块的性能。我们比较了团队标签预测的三类方法：（1）利用 MPNet 模型将团队标签预测作为检索任务；（2）利用 BERT 模型将团队标签预测作为分类任务；（3）使用我们的 PEFT 微调后的 $LoRA_t$ 模型，将团队标签预测作为生成任务。我们使用准确率、宏平均精确率（P）、召回率（R）和 F1 分数来评估 TeleDoCTR 和基线方法在工单路由任务上的表现，评估数据为测试集，共包含59,904个工单。表3展示了不同方法在故障排除数据测试集上的工单路由表现。

对于检索类别，我们采用基础版 MPNet、使用批内负样本损失微调的单个领域专用排序器，以及与工单检索相同的多个领域专用排序器。此外，我们还采用了三种变体：其中两种使用团队标注信息和批内全三元组损失进行微调 [23]；另一种则联合使用批内负样本损失和批内全三元组损失进行微调。三元组损失旨在确保锚点与正样本之间的距离至少比锚点与负样本之间的距离小一个预定义的间隔。批内全三元组损失会考虑批次中满足标签约束的所有锚点、正样本和负样本组合。在我们的设置中，标签约束指团队标注。变体“使用团队标签微调的 MPNet（仅工单）”使用由工单-团队对组成的 D_t 数据集进行训练。除了 D_t之外，变体“使用团队标签微调的 MPNet（工单与故障分析）”还纳入了故障分析-团队对，这些数据对可通过连接 D_f和 D_t推导得出。对于所有检索基线方法，给定测试集中的一张工单，我们首先对历史工单中的工单部分和故障分析部分建立索引；其次，检索前10个相似工单；然后，计算返回的前10张工单中的团队标签分布；最后，将出现频率最高的团队标签作为该工单的预测标签。与仅使用工单信息和批内全三元组损失微调 MPNet，以及使用故障分析信息和批内负样本损失微调 MPNet 相比，使用工单与故障分析信息以及团队标签微调的 MPNet 变体取得了更好的评估指标。与使用故障分析信息的批内负样本损失相比，结合团队标签的批内全三元组损失能够捕捉到更多团队相似性信号。这意味着，从团队标注信息中能够学习到的粗粒度相似性信号，比 TeleDoCTR 的领域专用排序器通过故障分析相似度捕捉到的细粒度相似性信号更多。此外，在团队标签预测任务中，与单个领域专用排序器相比，结合多个领域专用排序器并不能提升性能，因为不同的细粒度信号仍缺少工单路由所必需的粗粒度相似性信号。最后，我们同时使用以下两种方式对 MPNet 进行微调：(1) 使用批内全三元组损失和团队标签（工单与故障分析）；(2) 使用批内负样本损失和工单-故障分析对。在此变体中，我们同时捕捉粗粒度和细粒度相似性，因此在检索类别中取得了最佳团队路由结果。

表 3：故障排除数据测试集上的工单路由评估结果。

类别	方法	准确率	宏平均精确率	宏平均召回率	宏平均F1
检索	原始 MPNet	67.12	64.50	61.67	62.55
	单个领域专用排序器	75.57	73.84	71.85	72.51
	多个领域专用排序器	75.57	73.84	71.85	72.51
	使用团队标签微调的MPNet（仅工单）	74.82	72.54	70.23	70.85
	使用团队标签微调的MPNet（工单与故障分析）	76.33	74.27	71.79	72.59
	MPNet联合微调（团队与故障分析标签）	78.59	76.44	74.51	75.18
分类	微调后的BERT	77.49	75.09	74.12	74.29
	预训练的 + 微调后的BERT	78.94	77.31	75.27	75.94
	自定义分词器 +预训练 + 微调 BERT	79.60	77.32	76.32	76.65
生成	TeleDoCTR（$LoRA_t$）	**80.31**	**78.43**	**77.13**	**77.47**

对于分类类别，我们采用三种变体。第一种变体基于使用 D_t微调仅编码器 BERT[3] 模型。我们将第一个词元 [CLS] 的最终隐藏状态 h_θ汇聚为整个输入工单的表示，其中 θ表示 BERT 的参数。随后，在 BERT 顶部添加一个参数为W的 softmax 层，用于预测给定团队标签 l的概率：$p(l \mid h_\theta) = \text{softmax}(Wh_\theta)$。为了将电信相关知识注入仅编码器 BERT，第二种变体在工单路由任务的微调阶段之前加入预训练阶段，使用掩码语言模型（MLM）任务 [13]对 BERT 模型进行预训练，所用数据为大量未标注工单（489,890 张）。为进一步将电信领域专用术语融入 BERT，第三种变体使用根据故障排除数据构建的领域专用自定义分词器。使用标准 BERT 分词器对工单语料库进行分词后，我们识别出 1,932,532 个唯一词元。词元频率遵循齐夫分布，即少数词元频繁出现，而大多数词元很少出现。为兼顾覆盖范围与效率，我们仅保留排名前 30,000 的电信相关故障排查词元，这些词元不在原始 BERT 词汇表中；原始词汇表包含 30,522 个词元。经过扩充后，新分词器总共包含 60,522 个词元。接下来，我们使用更新后的分词器预训练 BERT，使模型能够通过掩码语言模型目标为新增的领域专用词元学习上下文嵌入。此预训练步骤确保新增的电信相关词汇能够有意义地融入表示空间，而非被视为词汇表外词元。预训练阶段结束时，模型的验证困惑度达到 1.88，表明模型已有效适应电信相关故障排除数据。最后，我们使用自定义分词器对预训练后的 BERT 进行微调，以完成工单路由这一项下游任务。表 3 显示，变体“自定义分词器+ 预训练 + 微调 BERT”在分类类别中取得了最佳性能。

TeleDoCTR ($LoRA_t$) 在所有评估指标上均略胜最强的基于检索和分类的基线方法，且这一提升稳定一致。这一优势可归因于它处理更高模型复杂度和更长工单输入的能力（如表 1 所示，最多可处理 10,000 个词元）。除了性能优势外，TeleDoCTR ($LoRA_t$) 在实际部署方面也具有显著优势。其设计整合了

[3]https://huggingface.co/distilbert/distilbert-base-uncased

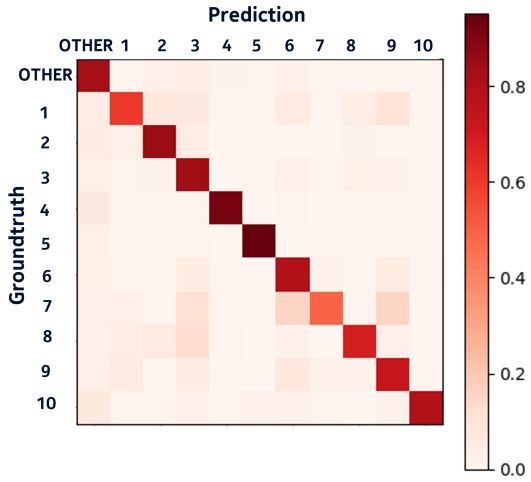


图 6：TeleDoCTR（$LoRA_t$）在工单路由测试集上的混淆矩阵。

LoRA 适配器，使其能够在单一、统一的故障排除流程中高效切换任务，而无需依赖多个碎片化模型。这种架构简化了系统维护，并确保随着新增故障排查任务，系统能够更顺畅地扩展。综合而言，这些特性使 TeleDoCTR ($LoRA_t$) 成为一种更具可持续性、适于部署的端到端电信故障排除解决方案。

在图 6 中，我们展示了 TeleDoCTR（$LoRA_t$）在工单路由测试集上的混淆矩阵。总体而言，混淆矩阵的对角线部分清晰，反映出准确的分类结果。根据对角线颜色，标签的性能排序从高到低依次为：5、4、OTHER、10、3、2、6、8、9、1 和 7。团队标签 3 尤其值得关注，因为它代表了最难准确预测的案例之一。因此，我们工作的一个关键目标是提升该标签上的 F1 分数，以彰显模型处理困难路由场景的能力。TeleDoCTR（$LoRA_t$）在团队标签 3 上取得了 80.0 的 F1 分数，相较于所有测试的基线方法，在这一特定路由标签上有显著提升。这一结果证明了我们的方法在应对困难工单路由场景方面的有效性，也凸显了 TeleDoCTR 是迈向更准确、可靠的自动化工单路由的重要里程碑。

表 4：故障排除数据测试集上的故障分析生成评估结果。

类别	方法	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore
基础版	LLaMA3-8B	0.216	0.030	0.106	0.800
	LLaMA3-70B	0.219	0.030	0.108	0.800
	LLaMA3.1-8B	0.190	0.021	0.084	0.794
	LLaMA3.1-70B	0.182	0.020	0.082	0.793
	LLaMA3.2-3B	0.179	0.017	0.079	0.792
	LLaMA3.3-70B	0.204	0.026	0.094	0.797
RAG	原始 MPNet +通用电信文档	0.218	0.032	0.112	0.806
	TeleDoCTR（单轮示例选择）	0.266	0.086	0.168	0.813
	TeleDoCTR（多轮示例选择）	0.371	0.235	0.284	0.845
微调	$LoRA_f$	0.325	0.191	0.246	0.832
	$LoRA_f$ +使用基础版 MPNet 进行响应排序	0.351	0.212	0.270	0.840
	TeleDoCTR（$LoRA_f$ + 响应排序，使用 R_1，R_2，.....，R_6）	**0.392**	**0.252**	**0.309**	**0.848**

5.3.3 故障分析生成。在这一部分，我们研究 TeleDoCTR 中故障分析生成模块的性能。

故障分析生成的定量结果：我们比较了三类故障分析生成方法：基础版、检索增强生成和微调。我们使用三个 ROUGE [32] 分数（ROUGE-1、ROUGE-2 和 ROUGE-L）以及 BERTScore [67]评估不同方法的性能。表 4 展示了不同方法在故障排除数据测试集上的故障分析生成效果。对于基础版类别，我们报告了多个不同规模（从 3B 到 70B）的 LLa MA 模型的结果。这些 LLaMA 模型都是经过指令微调的变体，针对响应用户指令进行了优化。对于检索增强生成类别，第一个基线方法（原始 MPNet +通用电信文档）是一种标准的检索增强生成方法，它使用基础版 MPNet 作为检索模型，并以与电信相关的通用内部文档作为知识语料库。对于基于检索增强生成的 TeleDoCTR 模块，我们评估了单轮和多轮故障分析生成。多轮版本模拟用户交互：除图 4 所示的主要故障分析生成提示外，还加入 3 个后续提示。这些额外的后续提示为：列出与新工单最相似的 *3* 个示例、根据这些示例的故障分析报告，说明这些已识别的相似示例是如何解决的，以及基于所有分析，严格按照以下故障分析 *JSON* 模板，在指定部分填入答案，提供新工单的最终故障分析报告。这些后续提示进一步促使大语言模型重新核查所提供的示例，并明确地对其重新排序，从而返回最相似的 3 个示例，并利用其中的故障分析部分，按照指定模板为给定工单生成最终故障分析。对于微调类别，我们比较了使用微调模型 ($LoRA_f$) 生成单个故障分析，与生成多个响应并进行排序后返回最相关响应这两种方式。对于响应排序机制，我们比较了使用基础版 MPNet 模型对响应进行排序（使用基础版 MPNet 的响应排序 $LoRA_f$ +），与使用我们的领域专用排序器进行排序（使用 、, 、 的响应排序 $LoRA_f$ + R_1 R_2 ... R_6 4 ）。表明，TeleDoCTR（单轮示例选择）显著优于基础版类别中的所有基线方法，此外还优于检索增强生成类别中的变体“原始 MPNet + 通用电信文档”。这表明，在检索增强生成中纳入相似的历史工单作为相关上下文，并使用我们的领域专用排序器生成准确的故障分析报告，至关重要；这优于大语言模型采用的标准检索增强生成方法。随后，如 TeleDoCTR（多轮示例选择）的结果所示，通过增加用户交互轮次，可以进一步改进基于检索增强生成的 TeleDoCTR 模块，从而更明确地优化故障分析报告。对于微调类别，表 4 显示，如果增加测试时计算量，让经过微调的大语言模型为排序机制生成多个报告（使用 MPNet 以及 R_1、R_2、.....、R_6模型），与微调后仅生成一份报告 ($LoRA_f$) 相比，定量指标会进一步提升。当我们在 TeleDoCTR 中使用领域专用排序器对微调模型生成的响应进行排序时，定量指标达到最佳 ($LoRA_f$ + 使用 R_1、R_2、..... 、R_6 进行响应排序)。

RLRF 结果：我们使用多种评估指标比较 $LoRA_f$ 和 $LoRA_{f+RLRF}$，以了解 RLRF 的贡献。对于由 1,000 个工单组成的测试数据集，我们计算了 3 类指标。第一类是生成的故障分析与真实故障分析之间的文本相似度，使用 BLUE[41]（通过衡量生成文本与参考文本之间 n 元语法的重叠度来关注精确率）、ROUGE-L [32]（通过衡量生成文本覆盖参考文本的程度来关注召回率）和 METEOR [2]（在考虑同义词和词干提取的同时结合精确率与召回率）进行评估。总体而言，这些指标从多个角度呈现词汇和结构相似度，从而能更细致地评估 $LoRA_f$ 和 $LoRA_{f+RLRF}$生成的故障分析报告。第二类是领域专用排序器相似度，即对测试集中每个工单-故障分析对使用公式（7）计算相似度，再取其在整个测试集上的平均值。该指标衡量生成的故障分析与领域专用排序器之间的一致性。第三类是病理响应比例，用于反映退化模型输出所导致的病理响应。表5展示了在用于故障分析生成的故障排除数据测试集上，$LoRA_f$ 和 $LoRA_{f+RLRF}$ 的评估指标。为了估算领域专用排序器相似度方面的上界性能，我们

表5：RLRF 在故障排除数据测试集上的评估结果。

数据集	优化步骤	排序器分数↑	病理比率↓	BLUE↑	ROUGE-L↑	METEOR↑
真实标注测试集	N/A	**0.60**	-	-	-	-
D_f及工单-故障分析对	$LoRA_f$	0.48	1.4%	**0.29**	**0.31**	**0.21**
偏好数据集 P_{RL}^1	$LoRA_{f+RLRF}$	0.53	3.7%	0.25	0.25	0.17
偏好数据集 P_{RL}^2	$LoRA_{f+RLRF}$	0.52	**0.7%**	**0.29**	0.30	0.20

计算测试集中真实工单-故障分析对的排序器得分，如表5所示，该得分为 0.60。与 $LoRA_f$相比，使用偏好数据集 P_{RL}^1进行微调的 $LoRA_{f+RLRF}$有助于提高排序器得分，使其更接近上界。然而，病理响应比例上升以及 BLUE、ROUGE-L 和 METEOR 下降，反映出了一些负面影响。使用包含被拒绝的病理响应的偏好数据集P_{RL}^2进行微调的 $LoRA_{f+RLRF}$，能减轻所有这些负面影响。使用 P_{RL}^2 进行微调的 $LoRA_{f+RLRF}$，其排序器得分也与使用 P_{RL}^1 微调的模型非常接近。这些结果表明，RLRF 能够：（1）进一步使微调后的故障分析生成模型与领域专用排序器反馈保持一致；（2）减少退化模型输出所导致的病理响应。未来的工作将进一步改进 RLRF，以提高 BLEU、ROUGE 和 METEOR 等基于文本匹配的指标。

故障分析生成的定性结果：我们对表 4 中排名前两位的方法所生成的故障分析报告进行了定性评估，它们分别是：（1）增强型 RAG 模块 TeleDoCTR（多轮示例选择）；（2）采用响应排序机制的微调模块 TeleDoCTR（$LoRA_f$ + 使用 R_1、R_2、……、R_6进行响应排序）。我们使用大语言模型评审器评估以下四项标准：准确率（预测的故障分析在多大程度上与真实故障分析语义一致？）、完整性（预测的故障分析是否涵盖了真实故障分析的所有关键方面（主要原因、促成因素、解决提示）？）、相关性（预测的故障分析是否紧扣所提供工单的主题并避免无关细节？）以及清晰度（预测的故障分析中的推理是否清晰且结构良好？）。大语言模型以工单信息、真实故障分析和预测的故障分析为输入，并针对每项标准预测一个从 0（差）到 5（优）的评分，同时为每项评分提供理由。我们使用 LLaMA3.3-70B[4] 作为大语言模型评审器，对由 1,000 个工单组成的测试数据进行预测故障分析评估。定性评估表明，TeleDoCTR（多轮示例选择）和 TeleDoCTR（$LoRA_f$ + 使用 R_1、R_2、……、R_6进行响应排序）在相关性和清晰度两项标准上的平均分均高于 4。具体而言，TeleDoCTR（多轮示例选择）的相关性得分为 4.87 ± 0.36，清晰度得分为 4.32 ± 0.54；而 TeleDoCTR（$LoRA_f$ + 使用 R_1、R_2、……、R_6进行响应排序）的相关性得分为 4.06 ± 0.56，清晰度得分为 4.03 ± 0.63。然而，两者在准确率和完整性方面存在差异。TeleDoCTR（多轮示例选择）的准确率为 3.33 ± 0.76，完整性为 3.32 ± 0.84，优于 TeleDoCTR（$LoRA_f$ + 使用 R_1、R_2、……、R_6进行响应排序），后者的准确率为 2.56 ± 0.78，完整性为 2.58 ± 0.87。这种差异主要归因于大语言模型评审器对根因预测的严格评估。由于根本原因以简短的类别标签表示，即使与真实标注略有偏差，也会导致准确率和完整性大幅扣分。在故障分析生成模型的微调过程中，这种类别式根本原因对整体指令微调损失的贡献很小，因为根本原因文本明显短于识别与解决字段。因此，尽管微调模型生成的报告在其他方面连贯一致，它仍更容易生成不匹配的标签。这与 TeleDoCTR（多轮示例选择）不同，后者能够从通过检索语义最相似工单构建的上下文中，选出最相关的确切根本原因。相比之下，ROUGE 分数和 BERTScore 等定量指标侧重于衡量整个生成的故障分析报告与真实报告之间的总体语义相似度。根据这些指标，TeleDoCTR（$LoRA_f$ + 使用 R_1、R_2、……、R_6进行响应排序）的表现更优，如表 4 所示，这体现了其生成全局一致且结构良好的输出的优势。总而言之，定性评估表明，根因预测应被视为一项独立任务，与团队标签预测类似，即引入新的 LoRA 层，专门用于根据工单和故障分析的识别部分生成根本原因。

综合来看，这些发现表明，TeleDoCTR（$LoRA_f$ + R_1 R_2 … R_6 响应排序，结合„„）和 TeleDoCTR（多轮示例选择）各有互补优势。TeleDoCTR（$LoRA_f$ + 响应排序，结合 $R_1, R_2, \ldots, R_6$）擅长将工单信息精准映射为详细的解决方案计划，这一点体现在其出色的量化表现上。然而，这些细粒度报告往往缺乏更广泛的上下文洞见，而这些洞见对于将问题置于历史故障排查知识中加以理解至关重要。TeleDoCTR（多轮示例选择）通过检索语义相似的历史工单，并借助多轮大语言模型与用户交互提供上下文，弥补了这一不足。通过结合微调故障分析的具体性与基于 RAG 的报告所具备的上下文感知能力，TeleDoCTR 提供了一个全面的故障排查框架，不仅能解决当前故障，还能将其置于电信系统的整体背景中加以理解。

[4]https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

6 结论

本文提出了一种新的电信相关、领域专用且具备上下文感知能力的故障排查系统，称为 TeleDoCTR。该系统结合领域适配排序与生成模型，实现电信相关工单解决流程的自动化。我们的系统采用多 LoRA 方法，对领域专用大语言模型进行微调，以完成工单分类和故障分析生成。此外，我们还在故障排查数据集上对专用排序模型进行微调，以衡量语义相关性。我们引入了一种响应排序机制：先由微调后的大语言模型生成多份故障分析报告，再通过领域专用排序器进行排序，从而识别出最相关的故障分析报告。为增强基于 RAG 的生成，我们利用排序模型选择示例（工单-故障分析对），并将其纳入基于 RAG 的对话模块，以改进多轮故障分析生成。为了训练和评估 TeleDoCTR，我们使用了大规模电信故障排查数据，并证明新系统在排序、分类和生成方面均优于最先进的基线方法。

未来的工作包括：(1) 将 TeleDoCTR 集成到基于代理的工作流中，以实现故障排除数据的持续收集、整理和适配自动化；(2) 通过减轻微调所导致的灾难性遗忘，并引入自我反思能力，使 TeleDoCTR 能够评估和完善自身输出，从而增强系统的稳健性。

致谢

我们感谢 Nokia Mobile Networks 的 Gabriel Górski、Jakub Kozerski 和 Bartlomiej Ruszaj 收集电信相关故障排查工单和日志，并部署 TeleDoCTR 系统的各个模块。我们感谢 Nokia Bell Labs 的 Ahmet Akyamac 和 Jin Cao 提供宝贵的反馈。我们感谢实习生 Kalliopi Basioti 和 Ziyi Liu 对故障排除数据进行深入且有价值的分析。

参考文献

[1] Nicola Arici、Luca Putelli、Alfonso Emilio Gerevini、Luca Sigalini、Ivan Serina 等。2023年。《基于大语言模型的自动工单分配方法：一项意大利真实应用》。载于 *NL4AI@ AI* IA*。[2] Satanjeev Banerjee 和 Alon Lavie。2005年。《METEOR：一种与人工判断相关性更高的机器翻译评估自动指标》。载于 机器翻译和／或摘要内在与外在评估指标 *ACL*研讨会论文集。65–72。[3] Lina Bariah、Hang Zou、Qiyang Zhao、Belkacem Mouhouche、Faouzi Bader 和 Mérouane Debbah。2023 年。《通过大语言模型理解电信语言》。载于 *IEEE* 全球通信大会，*GLOBECOM 2023*，吉隆坡，马来西亚，*2023*年*12*月*4*日至*8*日。IEEE，6542–6547。[4] Andrei-Laurentiu Bornea、Fadhel Ayed、Antonio De Domenico、Nicola Piovesan 和 Ali Maatouk。2024年。《Telco-RAG：应对电信领域检索增强语言模型的挑战》。载于 *2024*年 *IEEE* 全球通信大会，*GLOBECOM 2024*，开普敦，南非，*2024*年*12*月*8*日至*12*日。IEEE，2359–2364。[5] Nathan Bosch、Serveh Shalmashi、Forough Yaghoubi、Henrik Holm、Fitsum Gaim 和 Amir H Payberah。2022年。《微调基于 BERT 的语言模型以检索重复故障报告》。载于 *2022*年 *IEEE*国际大数据会议（大数据）。IEEE，4737–4745。[6] Yujun Chen、Xian Yang、Qingwei Lin、Hongyu Zhang、Feng Gao、Zhangwei Xu、Yingnong Dang、Dongmei Zhang、Hang Dong、Yong Xu、Hao Li 和 Yu Kang。2019年。《云服务系统的中断预测与诊断》。载于 万维网大会，*WWW 2019*。ACM，2659–2665。[7] Zhiyu Chen、Mohamed Trabelsi、Jeff Heflin、Yinan Xu 和 Brian D. Davison。2020年。《使用深度上下文化语言模型进行表格搜索》。载于 论文集第43届ACM SIGIR国际信息检索研究与发展会议（线上活动，中国）。计算机协会，纽约州纽约市，美国，589–598。[8] Zhuo Chen、Wen Zhang、Yufeng Huang、Mingyang Chen、Yuxia Geng、Hongtao Yu、Zhen Bi、Yichi Zhang、Zhen Yao、Wenting Song、Xinliang Wu、Yi Yang、Mingyi Chen、Zhaoyang Lian、Yingying Li、Lei Cheng 和 Huajun Chen。2023年。面向故障分析的电信知识预训练。载于第39届IEEE国际数据工程会议 ICDE 2023，美国加利福尼亚州阿纳海姆，2023年4月3日至7日。IEEE，3453–3466。[9] Matei Cristian、Săcărea Christian 和 Tolciu Dumitru-Tudor。2019年。利用自然语言处理自动识别服务工单的研究。载于2019年软件、电信与计算机网络国际会议（SoftCOM）。IEEE，1–6。[10] Jiaxi Cui、Zongjian Li、Yang Yan、Bohua Chen 和 Li Yuan。2023年。ChatLaw：集成外部知识库的开源法律大语言模型。CoRR abs/ 2306 16092 2023 10 48550 2306 16092 。[11] 2019 （）。https://doi.org/。/arXiv。。Zhuyun Dai 和 Jamie Callan。。借助上下文神经语言建模加深对信息检索中文本的理解。载于第42届ACM SIGIR国际信息检索研究与发展会议论文集。4页。[12] Tim Dettmers、Artidoro Pagnoni、Ari Holtzman 和 Luke Zettlemoyer。2023年。QLoRA：高效微调量化大语言模型。载于《神经信息处理系统进展》第36卷：2023年神经信息处理系统年会，NeurIPS 2023。[13] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。2019年。BERT：用于语言理解的深度双向Transformer预训练。载于2019年计算语言学协会北美分会年会：人类语言技术会议论文集（NAACL-HLT 2019）。计算语言学协会，4171–4186。[14] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。2019年。BERT：用于语言理解的深度双向Transformer预训练。载于NAACL-HLT。[15] Yuyang Du、Hongyu Deng、Soung Chang Liew、Kexin Chen、Yulin Shao 和 He Chen。2024年。大语言模型在无线通信系统开发中的作用：FPGA平台案例研究。arXiv: 2307 07319 [ ] 2307 07319 。eess.SP [16] 2018 https://arxiv.org/abs/。Angela Fan、Mike Lewis 和 Yann N. Dauphin。。层次化神经故事生成。载于计算语言学协会第56届年会论文集，ACL 2018。计算语言学协会，889–898。[17] Leon Feng、Jnana Senapati 和 Bill Liu。2022年。Ta Daa：面向客户支持、服务台和问题工单系统的实时工单分配深度学习自动顾问。arXiv预印本 arXiv:2207.11187（2022年）。[18] Nicolas Ferland、Wenting Sun、Xuancheng Fan、Lule Yu 和 Jieneng Yang。2020年。利用混合自然语言处理自动解决故障工单。载于2020年IEEE计算智能研讨会系列。IEEE，1334–1340。[19] Luyu Gao、Yunyi Zhang、Jiawei Han 和 Jamie Callan。2021年。在内存受限的设置下扩展深度对比学习的批量大小。载于第6届自然语言处理表示学习研讨会论文集，RepL4NLP@ACL-IJCNLP 2021。计算语言学协会，316–321。[20] Tanya Goyal、Junyi Jessy Li 和 Greg Durrett。2022年。GPT-3时代的新闻摘要与评估。CoRR abs/2209.12356（2022年）。[21] Nuria Marzo I Grimalt、Serveh Shalmashi、Forough Yaghoubi、Leif Jonsson 和 Amir Hossein Payberah。2022年。BERTicsson：故障排查推荐系统。载于SDU@ AAAI。[22] Xiao Han、Shuhan Yuan 和 Mohamed Trabelsi。2023年。LogGPT：通过GPT进行日志异常检测。载于IEEE国际大数据会议，BigData 2023，意大利索伦托，2023年12月15日至18日。IEEE，1117–1122。[23] Alexander Hermans、Lucas Beyer 和 Bastian Leibe。2017年。为行人重识别中的三元组损失辩护。CoRR abs/1703.07737（2017年）。http://arxiv.org/abs/ 1703 07737 。Ari Holtzman、Jan Buys、Li Du、Maxwell [24] Forbes 和 Yejin Choi。2020年。神经文本退化的奇特案例。载于国际学习表征会议。[25] Edward J Hu、Yelong Shen、Phillip Wallis、Zeyuan Allen-Zhu、Yuanzhi Li、Shean Wang、Lu Wang、Weizhu Chen 等。2022年。LoRA：大型语言模型的低秩适配。ICLR 1, 2（2022年），3。[26] Shubham Jain、Amit Gupta 和 Kumari Neha。2024年。用于优化支持服务的AI增强型工单管理系统。载于第4届AI-ML Systems国际会议论文集，AIMLSystems 2024，美国路易斯安那州巴吞鲁日，2024年10月8日至11日。ACM，21:1–21:7。[27] Chao Jia、Yinfei Yang、Ye Xia、Yi-Ting Chen、Zarana Parekh、Hieu Pham、Quoc V. Le、Yun-Hsuan Sung、Zhen Li 和 Tom Duerig。2021年。利用噪声文本监督扩展视觉与视觉语言表示学习。载于第38届国际机器学习会议论文集，ICML 2021（《机器学习研究论文集》第139卷）。PMLR，4904–4916。[28] Tong Jia、Pengfei Chen、Lin Yang、Ying Li、Fanjing Meng 和 Jingmin Xu。2017年。一种基于混合图模型与日志的分布式服务异常诊断方法

。载于*2017*年*IEEE*国际*Web*服务会议，*ICWS 2017*。IEEE，25–32。[29] Vladimir Karpukhin、Barlas Oguz、Sewon Min、Patrick S. H. Lewis、Ledell Wu、Sergey Edunov、Danqi Chen 和 Wen-tau Yih。2020年。用于开放域问答的稠密段落检索。载于*2020*年自然语言处理实证方法会议论文集，*EMNLP 2020*。计算语言学协会，6769–6781。[30] Omar Khattab 和 Matei Zaharia。2020年。ColBERT：通过BERT上的上下文化晚期交互实现高效且有效的段落搜索。载于第*43*届*ACM SIGIR*国际信息检索研究与发展会议论文集。ACM，39–48。[31] Huayang Li、Tian Lan、Zihao Fu、Deng Cai、Lemao Liu、Nigel Collier、Taro Watanabe 和 Yixuan Su。2023年。重复输入，重复输出：从数据视角理解神经文本退化。载于《神经信息处理系统进展》第*36*卷：*2023*年神经信息处理系统年会，*NeurIPS 2023*。[32] Chin-Yew Lin。2004年。ROUGE：用于摘要自动评估的工具包。载于文本摘要的拓展。74–81。[33] Yinhan Liu、Myle Ott、Naman Goyal、Jingfei Du、Mandar Joshi、Danqi Chen、Omer Levy、Mike Lewis、Luke Zettlemoyer 和 Veselin Stoyanov。2019年。RoBERTa：一种稳健优化的BERT预训练方法。*ArXiv* abs/1907.11692（2019年）。[34] Zhexiong Liu、Cris Benge 和 Siduo Jiang。2023年。Ticket-bert：利用语言模型标注事件管理工单。*arXiv*预印本 *arXiv:2307.00108*（2023年）。[35] Haipeng Luo、Qingfeng Sun、Can Xu、Pu Zhao、Jian-Guang Lou、Chongyang Tao、Xiubo Geng、Qingwei Lin、Shifeng Chen、Yansong Tang 和 Dongmei Zhang。2025年。WizardMath：通过强化Evol-Instruct增强大语言模型的数学推理能力。载于第十三届国际学习表征会议，*ICLR 2025*，新加坡，*2025*年*4*月*24*日至*28*日。OpenReview.net。[36] Yun Luo、Zhen Yang、Fandong Meng、Yafu Li、Jie Zhou 和 Yue Zhang。2023年。大语言模型持续微调过程中灾难性遗忘的实证研究。*CoRR* abs/2308.08747（2023年）。[37] Bonan Min、Hayley Ross、Elior Sulem、Amir Pouran Ben Veyseh、Thien Huu Nguyen、Oscar Sainz、Eneko Agirre、Ilana Heintz 和 Dan Roth。2024年。大型预训练语言模型推动自然语言处理的最新进展：综述。*ACM Comput. Surv.* 56, 2（2024年），30:1–30:40。[38] Rajdeep Mondal、Alan Tang、Ryan Beckett、Todd D. Millstein 和 George Varghese。2023年。大语言模型需要什么才能合成正确的路由器配置？载于第*22*届*ACM*网络热点研讨会论文集，*HotNets 2023*。ACM，189–195。[39] Rodrigo Frassetto Nogueira 和 Kyunghyun Cho。2019年。使用BERT进行段落重排序。*CoRR* abs/1901.04085（2019年）。[40] Rodrigo Frassetto Nogueira、Wei Yang、Kyunghyun Cho 和 Jimmy Lin。2019年。使用BERT进行多阶段文档排序。*CoRR* abs/1910.14424（2019年）。[41] Kishore Papineni、Salim Roukos、Todd Ward 和 Wei-Jing Zhu。2002年。BLEU：一种用于机器翻译自动评估的方法。载于计算语言学协会第*40*届年会论文集。311–318。[42] Alec Radford、Jong Wook Kim、Chris Hallacy、Aditya Ramesh、Gabriel Goh、Sandhini Agarwal、Girish Sastry、Amanda Askell、Pamela Mishkin、Jack Clark、Gretchen Krueger 和 Ilya Sutskever。2021年。通过自然语言监督学习可迁移视觉模型。载于第*38*届国际机器学习会议论文集，*ICML 2021*（《机器学习研究论文集》第*139*卷）。PMLR，8748–8763。[43] Rafael Rafailov、Archit Sharma、Eric Mitchell、Christopher D Manning、Stefano Ermon 和 Chelsea Finn。2023年。直接偏好优化：你的语言模型其实是一个奖励模型。神经信息处理系统进展 36（2023年），53728–53741。[44] Stephen E. Robertson、Steve Walker、Susan Jones、Micheline Hancock-Beaulieu 和 Mike Gatford。1994年。TREC-3中的Okapi。载于第三届文本检索会议论文集，*TREC 1994*，第500-225卷。109–126。[45] Devjeet Roy、Xuchao Zhang、Rashi Bhave、Chetan Bansal、Pedro Henrique B. Las-Casas、Rodrigo Fonseca 和 Saravan Rajmohan。2024年。探索用于根因分析的大语言模型代理。载于第*32*届*ACM*国际软件工程基础会议附属论文集，*FSE 2024*，巴西加利尼亚斯港，*2024*年*7*月*15*日至*19*日，Marcelo d'Amorim（编）。ACM，208–219。[46] Wataru Sakata、Tomohide Shibata、Ribeka Tanaka 和 Sadao Kurohashi。2019年。利用查询与问题相似度以及基于BERT的查询-答案相关性进行常见问题检索。载于第*42*届*ACM SIGIR*国际信息检索研究与发展会议论文集。计算机协会，1113–1116。[47] Kenneth R Sample、Alan C Lin、Brett J Borghetti 和 Gilbert L Peterson。2018年。预测故障工单解决方案。*Restoration* 4, 358（2018年），2。[48] Karan Singhal、Tao Tu、Juraj Gottweis、Rory Sayres、Ellery Wulczyn、Le Hou、Kevin Clark、Stephen Pfohl、Heather Cole-Lewis、Darlene Neal、Mike Schaekermann、Amy Wang、Mohamed Amin、Sami Lachgar、Philip Andrew Mansfield、Sushant Prakash、Bradley Green、Ewa Dominowska、Blaise Agüera y Arcas，

Nenad Tomasev、Yun Liu、Renee Wong、Christopher Semturs、S. Sara Mahdavi、Joelle K. Barral、Dale R. Webster、Gregory S. Corrado、Yossi Matias、Shekoofeh Azizi、Alan Karthikesalingam 和 Vivek Natarajan。2023年。借助大语言模型实现专家级医疗问答。CoRR abs/2305.09617 (2023)。[49] Kaitao Song、Xu Tan、Tao Qin、Jianfeng Lu 和 Tie-Yan Liu。2020年。MPNet：面向语言理解的掩码与置换预训练。载于《神经信息处理系统进展》第33卷：神经信息处理系统年度会议 2020（NeurIPS 2020），2020年12月6日至12日，线上，Hugo Larochelle、Marc'Aurelio Ranzato、Raia Hadsell、Maria-Florina Balcan 和 Hsuan-Tien Lin（编）。[50] Xiaofei Sun、Xiaoya Li、Jiwei Li、Fei Wu、Shangwei Guo、Tianwei Zhang 和 Guoyin Wang。2023年。通过大语言模型进行文本分类。载于《计算语言学协会论文集：EMNLP 2023》论文集。计算语言学协会，8990–9005。[51] Sasu Tarkoma、Roberto Morabito 和 Jaakko J. Sauvola。2023年。用于在 6G 系统中集成大语言模型技术的 AI 原生互联框架。CoRR abs/2311.05842 (2023)。[52] Mohamed Trabelsi、Aidan Boyd、Jin Cao 和 Hüseyin Uzunalioglu。2025年。用于描述性标题生成的时间序列语言模型。CoRR abs/2501.01832 (2025)。[53] Mohamed Trabelsi、Jin Cao 和 Jeff Heflin。2020年。使用深度上下文化语言模型进行语义标注。CoRR abs/2010.16037 (2020)。[54] Mohamed Trabelsi、Jin Cao 和 Jeff Heflin。2021年。SeLaB：使用 BERT 进行语义标注。载于 2021年国际神经网络联合会议（IJCNN）。1–8。https://doi.org/

10 1109 52387 2021 9534408 。/IJCNN。 。Mohamed Trabelsi、[55] Zhiyu Chen、Brian D. Davison 和 Jeff Heflin。2021年。用于文档检索的神经排序模型。《信息检索期刊》24, 6 (2021), 400–444。[56] Mohamed Trabelsi、Zhiyu Chen、Shuo Zhang、Brian D. Davison 和 Jeff Heflin。2022年。StruBERT：用于表格搜索与匹配的结构感知 BERT。载于万维网大会论文集（WWW 2022）。[57] Mohamed Trabelsi、Jeff Heflin 和 Jin Cao。2022年。DAME：用于实体匹配的领域自适应。载于第 15 届 ACM 国际网页搜索与数据挖掘会议论文集（WSDM 2022）。[58] Mohamed Trabelsi 和 Hüseyin Uzunalioglu。2023年。Absformer：用于无监督多文档抽象式摘要的 Transformer 模型。载于《文档分析与识别——ICDAR 2023研讨会论文集》第二部分（《计算机科学讲义》第14194卷）。Springer，151–166。[59] Trieu H. Trinh、Yuhuai Wu、Quoc V. Le、He He 和 Thang Luong。2024年。无需人工示范解决奥林匹克几何问题。《自然》625, 7995 (2024), 476–482。[60] David Wadden、Ulme Wennberg、Yi Luan 和 Hannaneh Hajishirzi。2019年。利用上下文化跨度表示进行实体、关系和事件抽取。载于《2019年自然语言处理实证方法会议论文集暨第九届自然语言处理国际联合会议论文集》（EMNLP-IJCNLP 2019）。计算语言学协会，5783–5788。[61] Changjie Wang、Mariano Scazzariello、Alireza Farshin、Dejan Kostic 和 Marco Chiesa。2023年。让网络配置更便于人类使用。CoRR abs/2309.06342 (2023)。[62] Difeng Wang、Wei Hu、Ermei Cao 和 Weijian Sun。2020年。用于文档级关系抽取的全局到局部神经网络。载于《2020年自然语言处理实证方法会议论文集》（EMNLP 2020）。计算语言学协会，3711–3721。[63] Liang Wang、Nan Yang 和 Furu Wei。2024年。学习为大语言模型检索上下文示例。载于第 18 届计算语言学协会欧洲分会会议论文集。计算语言学协会，1752–1767。[64] Shijie Wu、Ozan Irsoy、Steven Lu、Vadim Dabravolski、Mark Dredze、Sebastian Gehrmann、Prabhanjan Kambadur、David S. Rosenberg 和 Gideon Mann。2023年。BloombergGPT：面向金融的大语言模型。CoRR abs/2303.17564 (2023)。[65] Girma M. Yilma、Jose A. Ayala-Romero、Andres Garcia-Saavedra 和 Xavier Costa-Pérez。2024年。TelecomRAG：借助检索增强生成和大语言模型驾驭电信标准。《计算机通信评论》54, 3 (2024), 18–23。[66] Lu Yuan、Dongdong Chen、Yi-Ling Chen、Noel Codella、Xiyang Dai、Jianfeng Gao、Houdong Hu、Xuedong Huang、Boxin Li、Chunyuan Li、Ce Liu、Mengchen Liu、Zicheng Liu、Yumao Lu、Yu Shi、Lijuan Wang、Jianfeng Wang、Bin Xiao、Zhen Xiao、Jianwei Yang、Michael Zeng、Luowei Zhou 和 Pengchuan Zhang。2021年。Florence：面向计算机视觉的新型基础模型。CoRR (2021)。[67] Tianyi Zhang、Varsha Kishore、Felix Wu、Kilian Q. Weinberger 和 Yoav Artzi。2020年。BERTScore：使用 BERT 评估文本生成。载于第八届国际学习表征会议（ICLR 2020）。OpenReview.net。[68] Wenxuan Zhang、Yue Deng、Bing Liu、Sinno Jialin Pan 和 Lidong Bing。2023年。大语言模型时代的情感分析：现实检验。CoRR abs/2305.15005 (2023)。[69] Hang Zou、Qiyang Zhao、Yu Tian、Lina Bariah、Faouzi Bader、Thierry Lestable 和 Mérouane Debbah。2024年。TelecomGPT：构建电信领域专用大语言模型的框架。CoRR abs/2407.09424 (2024)。