面向学生作文的模型改写检测方法

姓名： 卢品仁

学号： 3120220953

学院： 计算机学院

2025 年 6 月

学

面向学生作文的模型改写检测方法

卢品仁

北京理工大学

中图分类号：TQ028.1
UDC 分类号：540

面向学生作文的模型改写检测方法

作　者　姓　名　　卢品仁
学　院　名　称　　计算机学院
指　导　教　师　　李侃教授
答辩委员会主席　　王五教授
申　请　学　位　　工学硕士
一　级　学　科　　计算机科学与技术
学位授予单位　　北京理工大学
论文答辩日期　　2025 年 6 月

A Method for Detecting Model-Rewritten Texts in Student Compositions

Candidate Name:	Pinren Lu
School or Department:	School of Computer Science and Technology
Faculty Mentor:	Prof. Kan Li
Chair, Thesis Committee:	Prof. Wang Wu
Degree Applied:	Master of Engineering
Major:	Computer Science and Technology
Degree by:	Beijing Institute of Technology
The Date of Defence:	June, 2025

研究成果声明

本人郑重声明：所提交的学位论文是我本人在指导教师的指导下独立完成的研究成果。文中所撰写内容符合以下学术规范（请勾选）：

☑ 论文综述遵循“适当引用”的规范，全部引用的内容不超过 50%。

☑ 论文中的研究数据及结果不存在篡改、剽窃、抄袭、伪造等学术不端行为，并愿意承担因学术不端行为所带来的一切后果和法律责任。

☑ 文中依法引用他人的成果，均已做出明确标注或得到许可。

☑ 论文内容未包含法律意义上已属于他人的任何形式的研究成果，也不包含本人已用于其他学位申请的论文或成果。

☑ 与本人一同工作的合作者对此研究工作所做的任何贡献均已在学位论文中作了明确的说明并表示了谢意。

特此声明。

签　　名：　　日 期：

关于学位论文使用权的说明

本人完全了解北京理工大学有关保管、使用学位论文的规定，其中包括：

① 学校有权保管、并向有关部门送交学位论文的原件与复印件；

② 学校可以采用影印、缩印或其它复制手段复制并保存学位论文；

③ 学校可允许学位论文被查阅或借阅；

④ 学校可以学术交流为目的, 复制赠送和交换学位论文；

⑤ 学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

签　　名：　　日 期：

导师签名：　　日 期：

摘要

近年来，生成性大语言模型（LLMs）迅速发展，生成的内容几乎无法与人类撰写的文本区分开来。尽管这一进展在各个领域得到了广泛应用，但也引发了教育工作者对学生提交作品真实性的重大担忧。因此，解决教育领域中对人工智能生成文本（AIGT）的滥用问题已成为当务之急。目前的检测策略主要集中在整篇文档上，这并未完全满足实际需求。由于学生在将 AI 生成的内容纳入自己的论文之前，可能会对其进行一定程度的修改，因此细粒度检测，特别是在句子级别的检测，显得尤为重要。因此，追踪文本来源的任务越来越受到关注。针对这一点，本研究创新性地提出了教育领域内文本来源追踪的任务，并构建了相应的数据集，命名为模型改写文本检测数据集，英文名称为 TOSWT（Tracing the Origins of Students’ Writing Texts）。该数据集包含由五个杰出的语言模型生成的文本，基于学生撰写的论证性论文，共包含 53,328 个文档级和 147,976 个句子级的数据样本。研究通过在文档级和句子级数据上进行实验评估，评估了多种深度学习检测模型。结果表明，文本来源追踪的任务极具挑战性，其中句子级任务尤其困难。

为便于用户操作，本文还实现了一个文本改写检测系统。需求分析、系统架构设计以及各模块和接口的设计均在文中进行了详细阐述。该系统的实现基于 Flask 框架，前端采用 Vue.js 进行开发，后端则使用 Python 编程语言。系统的主要功能包括文本改写和文本改写检测。该系统的设计与实现为后续研究奠定了坚实的基础。。

关键词：大语言模型；文本溯源；检测；教育；追踪来源

Abstract

In recent years, generative large language models (LLMs) have undergone rapid development, producing content that is nearly indistinguishable from human-written text. While this advancement has found widespread application across various fields, it has also raised significant concerns among educators regarding the authenticity of student submissions. Consequently, addressing the misuse of AI-generated text (AIGT) in the educational sector has become an urgent priority. Current detection strategies primarily focus on whole documents, which do not fully satisfy practical requirements. Due to the likelihood that students may modify AI-generated content to some extent before incorporating it into their essays, fine-grained detection, particularly at the sentence level, is of paramount importance. Consequently, the task of tracing text provenance has increasingly garnered attention. In light of this, this study innovatively proposes the task of text provenance tracing within the educational domain and constructs a corresponding dataset named TOSWT (Tracing the Origins of Students' Writing Texts). This dataset, which comprises texts generated by five outstanding large language models, is based on argumentative essays written by students and contains a total of 53,328 document-level and 147,976 sentence-level data samples. The study evaluates multiple deep learning detection models through experimental assessments on both document-level and sentence-level data. The results indicate that the task of text provenance tracing is highly challenging, with the sentence-level task proving particularly difficult.

To facilitate user interaction, this study also implements a text rewriting detection system. The requirements analysis, system architecture design, as well as the design of various modules and interfaces are elaborately described within this document. The implementation of the system is based on the Flask framework, with the front end developed using Vue.js and the back end utilizing the Python programming language. The primary functionalities of the system include text rewriting and text rewriting detection. The design and implementation of this system provide a solid foundation for future research.

Key Words: large language model; text provenance; detect; education; tracing origin

目录

插图

表格

主要符号对照表

AI	人工智能
LLM	大语言模型
NLP	自然语言处理
OpenAI	开发 ChatGPT 的国外公司
ChatGPT	由 OpenAI 开发有历史意义的大语言模型
AIGT	人工智能生成文本
Token	词元
TOSWT	追踪学生写作文本来源数据集

第 1 章 绪论

1.1 研究背景及意义

近年来，基于大型语言模型（Large Language Models，LLMs）的生成式自然语言处理（Generative Natural Language Processing，NLP）系统在技术上取得了显著的进展。这些进步从初期的普通语言模型，如 Transformer[1]、GPT[2]、GPT-2[3]、GPT-3[4]，逐步演化到更为复杂和强大的模型，如 ChatGPT[5]、GPT-4[6] 和 DeepSeek V3[7]。这些先进的语言模型生成的文本内容与人类撰写的文本几乎无法区分，表明了人工智能在自然语言处理领域的巨大潜力。这一技术进步已经在多个领域得到了广泛应用，包括聊天机器人、自动内容生成以及文摘工具等，人工智能生成文本（AI-generated texts, AIGT）几乎渗透到我们生活的每一个角落。

然而，对于教育工作者而言，这些技术进展引发了对学生撰写文档真实性的深切担忧。在教育环境中，确保学生的作业和报告反映其真实能力和思维过程变得愈加重要。因此，如何有效避免人工智能生成文本在教育领域的滥用，成为亟待解决的关键问题[8]。当前的人工智能生成文本检测策略，如基于监督学习的判别器和困惑度测量方法，主要集中于判断整篇文档是否由人工智能生成。然而，现实情况是，用户往往在使用大型语言模型时对部分文本进行修改，而非完全依赖于其生成整个文档[9]。这种现象表明，现有检测方法在应对实际应用中的复杂性时可能显得力不从心。

因此，探索细粒度（例如句子级别）的人工智能生成文本检测显得尤为重要[10]。细粒度检测不仅能够为总体检测器提供基础架构，还能够更准确地揭示文本的生成特征和模式。此外，人工智能生成文本的检测效果也可以用作评估其质量的指标：生成的文本越难以被检测出来，表明其与人类文本的一致性越高。

除了检测人工智能生成文本的能力，人工智能文本溯源的任务也逐渐浮出水面。首先，如果仅能检测出文本是由人工智能生成但无法追溯其具体来源，则该检测的说服力将大幅下降。其次，类似于人类学中的溯源概念，识别 AI 生成文本的来源对于理解文本的生成过程、评估其可信度，以及追踪潜在的偏见和错误至关重要。在教育学领域，为教师提供分析文本来源的工具，不仅能够帮助他们更有效地评估学生的作业和报告，还能识别文本的具体来源，从而判断学生是否过度依赖 AI 工具进行写作[11]。这一过程不仅有助于维护学术标准，还能促进学生对原创性和独立思考的重视。

通过深入了解文本的生成背景，教师能够在课堂上进行更有针对性的指导，鼓励学生发展批判性思维能力和创造性写作技巧。因此，本研究旨在探讨基于学生作文数据的人工智能生成文本分类器，旨在为教育工作者提供有效的工具，帮助他们更好地理解和评估学生的写作能力及其背后的生成机制。

1.2 国内外研究现状及发展趋势

关于使用检测器探测文本是否由人工智能生成的任务，可以从两个主要方面进行深入探讨：文本生成与文本分类。

首先，文本生成涉及到人工智能系统如何利用特定算法和模型生成自然语言文本。这一过程通常依赖于大型语言模型（Large Language Models, LLMs）架构。通过对海量文本数据的学习，这些模型能够捕捉语言的结构和语义特征，从而生成与人类书写风格相似的文本。这一领域的研究不仅关注生成文本的质量和流畅性，还包括生成文本的多样性和上下文适应能力。因此，深入理解文本生成的机制对于评估其潜在的应用和影响至关重要。

另一方面，文本分类是指通过特定的算法和方法对文本进行分析，以确定其是否由人工智能生成。探测文本是否为人工智能生成的任务可以视为一个二分类问题，采用文本分类模型将文本标记为“人类”或“AI”。这一过程通常包括特征提取、模型训练和分类决策等多个步骤。文本分类器可以利用多种技术，包括监督学习和无监督学习方法，这些方法通过分析文本的语言特征、结构特征及其上下文信息来识别文本的生成来源。通过对文本进行分类，研究者能够揭示人工智能生成文本的特征，从而提高检测的准确性和可靠性。

1.2.1 文本生成

国外在文本生成方面的技术暂时领先于国内。截至目前，大多数富有创新性的工作均由国外提出。追溯到 2017 年，Vaswani 等[1] 提出了 Transformer 模型，其核心创新在于自注意力机制，使得模型能够有效处理长距离依赖关系。Transformer 的并行处理能力和高效的特征提取使其成为自然语言处理任务的主流选择。在自然语言处理领域，Transformer 模型的引入标志着一种新的架构设计理念的诞生，其自注意力机制使得模型能够有效捕捉序列中各个元素之间的关系。然而，传统的 Transformer 模型通常依赖于大量标注数据进行任务特定的训练，导致其在不同任务上的适应性和泛化能

力受到限制。

为了解决这一问题，国外的研究人员率先开发出预训练模型。通过在大规模未标注文本上进行预训练，预训练模型不仅学习了语言的基本结构和语义，还能够通过微调快速适应多种下游任务。这种两阶段的训练策略和双向上下文理解能力，使得预训练模型在性能和效率上相较于传统模型具有显著优势，推动了自然语言处理技术的进一步发展。预训练模型在自然语言处理领域引起了广泛关注，尤其是 BERT[12]、BART[13]、GPT-1[2]、GPT-2[3]和 GPT-3[4]等模型的出现，进一步推动了技术的进步。BERT（Bidirectional Encoder Representations from Transformers）通过双向编码器架构，能够同时考虑上下文的前后信息，极大地提升了文本理解能力。BART（Bidirectional and Auto-Regressive Transformers）则结合了 BERT 的编码特性和 GPT 的自回归生成能力，适用于文本生成和序列到序列的任务。GPT-1（Generative Pre-trained Transformer 1）是首个采用自回归生成的预训练模型，奠定了生成模型的基础。随后，GPT-2 在规模和性能上进行了显著提升，展示了强大的文本生成能力和广泛的应用潜力。最新的 GPT-3 则通过 1750 亿个参数，进一步提升了生成质量和上下文理解，展现出更强的通用性和灵活性，成为当前最先进的自然语言处理模型之一。这些模型的共同特征是通过预训练和微调机制，极大地提高了在各种自然语言处理任务中的表现。

与此同时，在文本生成领域，国内对语言模型和预训练模型的研究也取得了显著的进展，展现出丰富的创新成果。例如，王晓峰[14] 基于 Transformer 模型，深入探讨了其在中文文本生成中的应用。他的研究不仅丰富了中文自然语言处理的理论基础，还为实际应用提供了高效的技术支持，推动了该领域的快速发展。这一工作为后续研究者在中文文本生成任务中提供了重要的参考框架，使得相关技术得以在更广泛的场景中应用。

此外，国内还有研究专注于文本生成的基础架构，进一步推动了该领域的创新。李思慧等[15] 开发了一种新型的基于凸损失函数的离散扩散文本生成模型。该模型充分利用了凸函数锐化最优分布的特性，使得生成过程更加专注于高概率输出，从而显著提高了文本生成的质量。为了进一步提升生成文本的整体效果并降低生成词的重复率，研究团队设计了一种混合感知噪音表。该表通过引入非线性变化的噪音标记，增强了模型在生成过程中的灵活性和适应性。在解码过程中，他们还采用了一种高置信度的确定性去噪策略，有效地减少了生成文本中的噪音干扰。这一策略不仅提升了文本的流畅性和可读性，还增强了生成内容的多样性和创新性。这些研究成果不仅为中

文文本生成提供了新的思路和方法，还为未来的研究和应用奠定了坚实的基础，推动了自然语言处理技术在中文语境下的进一步发展。整体而言，国内在文本生成方面的研究正在不断深化，为推动人工智能的进步贡献着重要力量。

国内的研究更多集中于文本生成模型在实际应用领域的探索。胡妍璐[16] 在分析文本生成技术在新闻生产中的应用基础上，探讨了大数据环境下的新闻文本生成，旨在构建一种新型且更具影响力的新闻生产模式。张蔚琪[17] 提出了基于孪生架构与多视图匹配的法律问答模型，利用 BERT 孪生网络架构和注意力池化层生成多个视图的向量表示，通过细粒度的语义交互提升匹配准确性。此外，她将法律问答建模为生成任务，提出了一种基于迁移学习和改进解码算法的生成式法律问答模型，能够在生成过程中识别并替换过于通用的回复，从而提高回答的相关性。进一步地，张蔚琪构建了带有法条知识标注的法律问答数据集，并提出了一种检索知识驱动的生成式模型。该模型结合了知识检索与答案生成，确保生成的回答既具相关性，又能根据最新法律知识进行调整。这些研究成果为法律智能化建设提供了新的思路，推动了法律问答系统的发展。

预训练模型的成功为大语言模型的出现奠定了基础，后者在此基础上进行了多项重要改进。大语言模型通过更大规模的训练数据和参数量，显著提升了模型的表现能力和生成质量。此外，它们采用了更深层次的网络结构和更复杂的训练策略，例如混合精度训练和分布式训练，以提高训练效率和模型的表达能力。同时，大语言模型在上下文理解方面也进行了优化，能够处理更长的文本输入，增强了对上下文的记忆和推理能力。此外，许多大语言模型引入了更先进的自注意力机制和动态计算图，使得模型在生成文本时更加灵活和高效。这些改进使得大语言模型不仅在自然语言生成任务中表现出色，也在理解、翻译和问答等多种应用场景中展现出强大的能力，推动了人工智能技术的进一步发展。在近年来的自然语言处理领域，多个大型语言模型相继被提出，显著推动了文本生成和理解的研究进展。ChatGPT[5] 是 OpenAI 开发的对话生成模型，基于生成预训练变换器（GPT）架构，经过大规模对话数据的微调，旨在实现自然流畅的人机交互。其设计重点在于上下文理解和生成能力，使其能够在多种主题下进行有效交流。GPT-4[6] 是 OpenAI 推出的第四代生成预训练变换器模型，相较于其前身 GPT-3，GPT-4 在模型规模、性能和上下文处理能力上均有显著提升。该模型能够处理更长的文本输入，并在多种复杂任务中表现出更高的灵活性和准确性，尤其在多模态输入（文本和图像）方面展现出新的潜力。Gemini[18] 是由 Google DeepMind

开发的一种新型大型语言模型，旨在与现有的 AI 模型竞争。Gemini 结合了先进的技术和架构，强调在自然语言理解和生成任务中的表现，展现出在多领域应用中的强大能力。LLaMA[19-21]（Large Language Model Meta AI）是 Meta（前 Facebook）推出的一系列开源大型语言模型，旨在提供高效的语言生成和理解能力。LLaMA 关注于可扩展性和性能，支持多种语言和任务，尤其在资源有限的环境中表现出色。这些模型的不断发展为自然语言处理领域的研究和应用提供了新的视角和方法。

近年来，国内在大语言模型的研究领域取得了显著突破。早在 OpenAI 推出 ChatGPT 之前，中国的研究人员便已开始对大语言模型进行探索。由于大型企业拥有丰富的显卡资源，因此其研究进展往往领先于学术界。百度的研究团队率先进入这一领域，提出了文心一言模型[22-24]，该模型具备多模态处理能力，能够有效处理文本、图像等多种数据类型，适应广泛的应用场景。文心一言专门针对中文进行了优化，显著提升了对中文语境的理解和生成效果，展现出卓越的文本生成能力，适合于内容创作和客户服务等多个领域。此外，该模型整合了丰富的知识库，能够在文本生成过程中引用相关知识，从而增强文本的准确性与信息量。其高度的可定制性使得用户能够根据特定需求进行调整，支持实时交互并迅速响应用户输入，从而提升用户体验。这些特性使得文心一言在中文自然语言处理领域具有重要的应用潜力。

在 ChatGPT 震撼发布后，国内研究人员积极追赶，力求超越 OpenAI 的研究成果。华为的研究团队提出了盘古模型[25-27]，该模型采用了大规模的预训练和微调策略，在多种语言理解和生成任务中表现出色。此外，盘古模型在中文处理方面进行了针对性优化，能够更准确地理解和生成符合中文语境的内容，具备强大的上下文理解能力，能够在复杂对话和长文本生成中保持连贯性。华为团队还深入研究了模型的可扩展性和效率，使其能够在多种硬件平台上高效运行。总体而言，盘古模型为中文自然语言处理提供了强有力的技术支持，推动了相关应用的发展。

在 ChatGPT（GPT-3.5）发布的同时，清华大学的研究团队推出了 GLM-130B[28]，在众多流行的英语基准测试中显著超越了 ChatGPT 的前一版本 GPT-3 175B（davinci）。同时，在相关基准测试中，GLM-130B 也始终显著优于当时百度提出的文心一言系列模型中的最大中文语言模型 ERNIE TITAN 3.0 260B。基于 GLM-130B 独特的缩放特性，该模型实现了 INT4 量化，无需后续训练，几乎没有性能损失，成为首个在 100B 规模模型中实现这一技术的模型。更为重要的是，这一特性使其能够在 4×RTX 3090（24G）或 8×RTX 2080 Ti（11G）GPU 上进行高效推理，成为当时使用 100B 规模模型

所需的经济高效的 GPU 选择。

在 OpenAI 发布 GPT-4 之后，国内的大语言模型研究如雨后春笋般蓬勃发展。阿里巴巴的研究团队推出了通义千问系列模型[29]。Qwen 是一个全面的开源语言模型系列，涵盖了不同参数规模的多种模型。其中包括基础的预训练语言模型 Qwen 以及经过人类对齐技术微调的聊天模型 Qwen-Chat。基础语言模型在多项下游任务中始终展现出卓越的性能，而聊天模型，尤其是那些通过强化学习（RLHF）技术结合人类反馈进行训练的模型，竞争极为激烈。这些聊天模型具备创建代理应用程序的高级工具使用与规划能力，甚至在处理复杂任务时，如使用代码解释器的大型模型中，也表现出令人瞩目的性能。此外，团队还开发了专用于编码的模型 Code-Qwen 和 Code-Qwen-Chat，以及基于基础语言模型的数学模型 Math-Qwen-Chat。与当时的其他开源模型相比，通义千问系列模型的性能显著提升，略微落后于专有模型。

紧接着，清华大学的研究团队在 GLM-130B 模型的研究中取得了创新进展，开发了 ChatGLM-RLHF 流程[30]，这是一个基于人类反馈的强化学习（RLHF[31]）系统，旨在增强 ChatGLM 与人类偏好的对齐一致性。ChatGLM-RLHF 主要包括三个核心组成部分：人类偏好数据的收集、奖励模型的训练以及策略的优化。为降低大规模训练中的奖励方差，采用了通过融合梯度下降实现模型并行性的策略，并设计了正则化约束以防止 LLM 中的灾难性遗忘。实验结果显示，与监督微调（SFT）版本的 ChatGLM 相比，ChatGLM-RLHF 在对齐任务中取得了显著的改进。例如，在中文对齐任务中，其平均表现比 ChatGLM-SFT 提高了 15%。这项研究展示了国内在根据人类偏好对齐大语言模型方面的实践，为强化学习实施中的挑战与解决方案提供了宝贵的见解。

随后，研究团队在此基础上提出了 GLM-4 大语言模型家族[32]，其中包括 GLM-4、GLM-4-Air 和 GLM-4-9B。这些模型代表了国内最具实力的开源模型，基于前三代 ChatGLM 的所有经验和洞察进行训练。GLM-4 模型在十万亿个主要为中文和英文的令牌上进行了预训练，同时还使用了来自 24 种语言的一小部分语料库，主要针对中文和英文的使用进行了对齐。高质量的对齐是通过多阶段的训练后过程实现的，包括监督微调和从人类反馈中学习。评估结果表明，GLM-4 在 MMLU[33]、GSM8K[34]、MATH[35]、BBH[36]、GPQA[37] 和 HumEval[38] 等多个通用指标上与 GPT-4 进行了密切竞争，甚至在某些方面优于 GPT-4。在 IFEval[39] 的指令遵循测评中，GLM-4 的表现接近于 GPT-4-Turbo，并在长上下文任务中与 GPT-4 Turbo（128K）和 Claude 3 的表现相当。此外，在对齐 Bench 测量的中文对齐方面，GLM-4 亦优于 GPT-4。在实际应

用中，GLM-4 在通过 Web 浏览获取在线信息和使用 Python 解释器解决数学问题等任务中，表现与 GPT-4 All Tools 相当甚至更佳，仅在 2023 年便吸引了超过 1000 万次的 HuggingFace 下载。

在 GPT-o1[40] 发布之后，国内再度掀起了追赶的热潮。OpenAI 的 o1 模型展示了推理策略（即测试时的计算方法）对大型语言模型推理能力的显著提升，其在多数数据集上均取得了最佳性能。对此，国内的大语言模型研究者们积极响应，迅速跟进。首先，阿里巴巴公司的研究团队在其早期开发的通义千问系列模型基础上，推出了通义千问 2.5（Qwen 2.5）模型[41]。这一系列全面的大语言模型（LLM）旨在满足多样化的需求。与之前的版本相比，Qwen 2.5 在训练前和训练后阶段均取得了显著进展。在预训练阶段，模型所使用的高质量数据集从 7 万亿词元扩展至 18 万亿词元，为常识、专家知识及推理能力的提升奠定了坚实基础。在后训练阶段，研究团队通过 100 万个样本实施了复杂的监督微调和多阶段强化学习，从而增强了人类偏好，显著改善了长文本生成、结构数据分析及指令遵循能力。为了有效应对多样化的应用场景，研究者们推出了不同规模的 Qwen 2.5 LLM 系列，开放权重的产品包括基础模型和指令调整模型，并提供量化版本。此外，托管方案中还包括两种混合专家（MoE）变体：Qwen 2.5-Turbo 和 Qwen 2.5-Plus，均可通过阿里云模型工作室获得。Qwen 2.5 在语言理解、推理、数学、编码以及人类偏好对齐等多个基准测试中展现了卓越性能。尤其是开放权重的旗舰 Qwen 2.5-72B-Instruct 在众多开放和专有模型中表现优异，并与最先进的开放权重模型 Llama-3-405B-Instruct[21] 竞争，后者的参数量约为其五倍。Qwen 2.5-Turbo 和 Qwen 2.5-Plus 在成本效益方面表现出色，分别与 GPT-4o-mini 和 GPT-4o 相抗衡。此外，Qwen 2.5 模型为训练专业模型（如 Qwen 2.5-Math、Qwen 2.5-Coder、QwQ 及多模态模型）提供了重要基础。

在推出这一文本模态的大语言模型之后，阿里巴巴的研究团队为实现与 OpenAI 的 GPT 系列模型相媲美的多模态能力，继续深入探索多模态大语言模型的潜力，提出了通义千问 2.5 VL[42]。该模型作为 Qwen 视觉语言系列的最新旗舰，展现了基础功能与创新特性的显著进步。Qwen2.5-VL 通过增强的视觉识别、精准的对象定位、强大的文档解析能力及对长视频的理解，实现在理解与世界互动方面的重大突破。Qwen2.5-VL 的一个显著特点是其能够通过边界框或点的方式精确定位对象，并具备从图像和表格中提取强大结构化数据的能力。为有效处理复杂输入，该模型引入了动态分辨率处理和绝对时间编码，使其能够通过二级事件定位来处理不同尺寸的图像及持续

时间可达数小时的视频。这一设计使得模型能够在无需依赖传统归一化技术的情况下，局部感知空间尺度和时间动态。通过从头训练原生动态分辨率视觉转换器（Vision Transformer, ViT[43]）并结合窗口注意力机制，Qwen2.5-VL 在保持原生分辨率的同时有效降低了计算开销。因此，Qwen2.5-VL 不仅在静态图像和文档理解方面表现卓越，作为交互式视觉代理，它能够在实际场景中进行推理、工具使用和任务执行，诸如操作计算机和移动设备等。该模型提供三种不同尺寸，以应对从边缘 AI 到高性能计算的多样化应用场景。旗舰型号 Qwen2.5-VL-72B 在文档和图表理解方面与 GPT-4o 及 Claude 3.5 Sonnet[44]等最先进模型相当，尤其在这些领域表现出色。此外，Qwen2.5-VL 在保持强大语言性能的同时，保留了 Qwen2.5 LLM 的核心语言能力。

在刚刚过去的一年中，杭州深度求索人工智能基础技术研究有限公司经历了技术的“爆炸性”发展，迅速获得了与 OpenAI 开发的人工智能相媲美的能力。2024 年初，该公司推出了 DeepSeek 模型[45]，其研究深入探讨了缩放定律，并提出了独特的发现，这些发现为在两种广泛使用的开源配置（7B 和 67B）中扩展大规模模型提供了重要支持。在缩放定律的指导下，研究团队引入了 DeepSeek LLM，这是一个旨在从长远角度推进开源语言模型的项目。为了更有效地支持预训练阶段，研究人员开发了一个目前包含 2 万亿词元的数据集，并持续进行扩展。随后，在 DeepSeek LLM Base 模型的基础上，团队实施了监督微调（SFT）和直接偏好优化（DPO），最终创建了 DeepSeek Chat 模型。评估结果显示，DeepSeek LLM67B 在一系列基准测试中超越了 LLaMA-270B，尤其在代码、数学及推理领域表现优异。此外，开放式评估结果表明，与 GPT-3.5 相比，DeepSeek LLM67B 在聊天任务中展现了卓越的性能。

在 2024 年中，杭州深度求索人工智能基础技术研究有限公司再次取得了技术突破，推出了 DeepSeek-V2[46]，这是一种强大的混合专家（MoE）语言模型，具有经济高效的训练和推理能力。该模型总参数量达到 236B，其中每个词元的激活参数为 21B，并支持 128K 词元的上下文长度。DeepSeek-V2 采用了多项创新架构，包括多头潜在注意力（MLA）和 DeepSeekMoE。MLA 通过显著压缩键值（KV）缓存为潜在向量，从而确保高效的推理过程；而 DeepSeekMoE 则通过稀疏计算以较低的成本训练出强大的模型。与 DeepSeek67B 相比，DeepSeek-V2 在性能上显著提升，同时节省了 42.5% 的训练成本，减少了 93.3% 的 KV 缓存，并将最大生成吞吐量提升至 5.76 倍。该模型在一个由 8.1T 词元构成的高质量多源语料库上进行了预训练，并进一步实施了监督微调（SFT）和强化学习（RL）以最大限度地发挥其潜力。评估结果显示，即便只有

21B 的激活参数，DeepSeek-V2 及其聊天版本仍在开源模型中展现出顶尖的性能。

在 2024 年 12 月，深度求索人工智能基础技术研究有限公司推出了一项重大创新，基于先前开发的 DeepSeek 模型[45-46]，开源了与 GPT-4o 性能相媲美的 DeepSeek-V3 模型[7]。该模型是一种强大的混合专家（MoE）语言模型，具有总参数量达到 671B，其中每个词元激活了 37B 参数。为实现高效的推理和训练，DeepSeek-V3 采用了多头潜在注意力（MLA）和 DeepSeekMoE 架构，这些架构在 DeepSeek-V2 中经过了全面验证。此外，DeepSeek-V3 引入了一种负载平衡的无辅助损失策略，并设定了多令牌预测的训练目标，以进一步提升模型性能。研究团队在一个包含 14.80 万亿多样化且高质量词元的语料库上进行了 DeepSeek-V3 的预训练，随后进行了监督微调和强化学习阶段，以充分发挥其潜力。综合评估结果表明，DeepSeek-V3 的性能超越了其他开源模型，并与领先的闭源模型相当。尽管性能卓越，DeepSeek-V3 的完整训练仅需 2.788M H800 GPU 小时。此外，其训练过程表现出极高的稳定性，研究人员在整个训练期间未遇到任何不可恢复的损失峰值或需进行回滚的情况。

紧接着，在刚刚过去的 2025 年初，深度求索的研究团队向全球发布了其显著成就，开源了与当今全球最强的 OpenAI 公司所推出的最佳模型 GPT-o1 相媲美的 DeepSeek-R1 模型[47]。此次发布包括了第一代推理模型 DeepSeek-R1-Zero 和 DeepSeek-R1。DeepSeek-R1-Zero 是通过大规模强化学习（Reinforcement Learning, RL）训练而成的，且未经过任何初步的有监督微调（Supervised Fine-tuning, SFT），展现出卓越的推理能力。通过强化学习的方式，DeepSeek-R1-Zero 自然地表现出许多强大而引人注目的推理行为。然而，该模型也面临可读性差和语言混合等诸多挑战。为了解决这些问题并进一步提升推理性能，研究团队推出了 DeepSeek-R1，该模型在强化学习之前结合了多阶段训练和冷启动数据。DeepSeek-R1 在推理任务中实现了与 OpenAI-o1-1217 相当的性能。为了支持学术研究社区，DeepSeek-R1-Zero、DeepSeek-R1 以及基于文心一言和 LLaMa 从 DeepSeek-R1 蒸馏而成的六个密集模型（1.5B、7B、8B、14B、32B、70B）均被无私地开源。

截至目前，阿里巴巴的研究团队再次发布了其最新的研究成果，名为 QwQ-32B[48]。大规模强化学习（Reinforcement Learning, RL）展现出超越传统预训练和后训练方法的潜力，能够显著提升模型的整体性能。近期的研究表明，强化学习不仅能够增强模型的推理能力，还能促进其在复杂任务中的表现。例如，DeepSeek-R1 通过整合冷启动数据与多阶段训练，达成了业界领先的性能，使其具备进行深度思考和复杂推理的

能力。在此背景下，研究人员深入探讨了大规模强化学习对大型语言模型智能提升的作用，并开源了阿里巴巴最新的推理模型 QwQ-32B。这一模型拥有 320 亿个参数，其性能与具备 6710 亿参数（其中 370 亿参数被激活）的 DeepSeek-R1 相媲美。这一成果不仅突显了将强化学习有效应用于经过大规模预训练的强大基础模型的潜力，还显示了在推理任务中实现高效性的可能性。此外，QwQ-32B 推理模型中集成了与智能体（Agent）相关的能力，使其在使用工具的同时能够进行批判性思考，并根据环境反馈灵活调整推理过程。这种设计理念表明，强大的基础模型与大规模强化学习的结合，或许为实现通用人工智能提供了一条可行的路径。这一研究成果为未来的人工智能发展指明了方向，展示了大规模强化学习在提升模型智能和推理能力方面的巨大潜力。

随着 DeepSeek 系列模型和 Qwen 系列模型的相继推出，中国在文本生成领域展现出强大的研究实力，显著推动了全球相关技术的发展。这些模型不仅在架构设计和训练方法上进行了创新，还在推理能力和生成质量上达到了国际领先水平。中国研究团队的这些努力，不仅推动了国内外学术界对文本生成技术的深入探索，也促进了相关应用的多样化发展。通过开源这些先进模型，研究者们为全球学术界提供了宝贵的资源，促进了知识共享和技术交流。因此，可以说，中国在文本生成领域的持续创新和实践，已经为全球研究和应用的进步提供了有力的支持，确立了其在国际舞台上的重要地位。

1.2.2 文本分类

在信息爆炸的背景下，文本分类作为自然语言处理（NLP）领域的一项核心任务，发挥着至关重要的作用。其主要目标是将文本数据自动归类到预设的类别中，广泛应用于情感分析、主题识别、垃圾邮件过滤及新闻分类等领域。在本文中，我们将探讨基于文本分类的方法，以实现对人工智能生成文本来源的检测。随着社交媒体和网络内容的迅速扩展，如何高效且准确地对海量文本进行分类已成为当前研究的一个重要焦点。文本分类的研究方法大致可以划分为三个主要阶段：机器学习、深度学习和预训练模型。

在深度学习方法出现之前，机器学习技术在研究界广泛受到欢迎。彼时，中国的经济条件相对不发达，导致计算机及相关研究领域的发展明显滞后于国外。在国际上，研究人员率先开展了文本分类的相关研究。文本分类的定义是从原始文本数据中提取

特征，并基于这些特征进行类别预测。在过去几十年中，针对文本分类的多种模型相继被提出。在传统模型中，朴素贝叶斯（Naive Bayes, NB）[49] 是首个被应用于文本分类任务的模型。此后，K-近邻（KNN）[50]、支持向量机（SVM）[51] 和随机森林（RF）[52] 等通用分类模型相继问世，这些模型统称为分类器，并在文本分类领域得到了广泛应用。

进入千禧年后，随着中国经济的迅速发展，越来越多的个人电脑普及，使得更多人能够参与计算机科学领域的研究。这一变革也促进了中国学者在机器学习理论方法研究方面的逐步深入。陈天奇等人[53] 提出了极端梯度提升（eXtreme Gradient Boosting, XGBoost）的方法，这是一种创新的稀疏感知算法，旨在处理稀疏数据和加权分位数草图，以实现树学习的近似计算。更为重要的是，他们在论文中提供了关于缓存访问模式、数据压缩及分片的深入见解，以构建可扩展的树提升系统。在此基础上，柯国霖等人[54] 提出了轻量级梯度提升机（Light Gradient Boosting Machine, LightGBM），实验结果显示，该方法在多个公共数据集上将传统梯度提升决策树（GBDT）的训练速度提高了超过 20 倍，同时保持了几乎相同的精度，展现出卓越的性能潜力。除了理论研究，中国学者们还重点关注机器学习的应用研究。面对高校“一站式”学生社区建设中亟需对网络平台事件言论进行分类的问题，孔令怡等人[55] 采用了多种机器学习算法，包括 K-近邻、决策树、多项式朴素贝叶斯、逻辑回归、支持向量机、随机森林、轻量级梯度提升及极端梯度提升，构建了针对社区事件言论的分类模型。此外，赵锴等人[56] 为实现高效且准确的矿床描述文本多标签分类，致力于降低从大量文本中提取细粒度知识的难度，构建了有针对性的标注数据集和相应的机器学习模型。这类应用研究在中国逐渐增多，上述研究以供参考。

伴随着并行计算速度的加快，关于深度学习方法的研究蓬勃发展。深度神经网络（DNNs）由模拟人脑的人工神经网络组成，能够自动从数据中学习高级特征，在语音识别、图像处理和文本理解等方面比传统模型取得更好的效果。在对数据进行分类时，需要分析输入数据集，例如单标签、多标签、无监督、不平衡的数据集。根据数据集的特点，将输入的词向量输入到深度神经网络中进行训练，直到达到终止条件。训练模型的性能通过下游任务进行验证，如情感分类、问答和事件预测等。国外由于率先研究出深度学习方法，因此在深度学习研究的方面暂时领先于国内。

在过去几十年里，人们提出了许多用于文本分类的深度学习模型。文本分类这一技术的应用领域包括情感分析（Sentiment Analysis, SA）、主题标注（Topic Labeling,

TL）、新闻分类（News Classification, NC）、问答系统（Question Answering, QA）、对话行为分类（Dialog Act Classfication, DAC）、自然语言推理（Natural Language Inference, NLI）和关系分类（Relation Classification, RC）。多层感知器[57] 和递归神经网络[58] 是最早用于文本分类任务的两种深度学习方法，与传统模型相比，它们提高了相当巨大的性能。随后，卷积神经网络（Convolutional Neural Network, CNNs）、循环神经网络（Recurrent Neural Network, RNNs）和注意力机制被应用于文本分类[59-61]。此外，许多研究人员通过改进卷积神经网络、循环神经网络和注意力机制，或者采用模型融合和多任务方法，来提升不同任务的文本分类性能。

国内应用深度学习技术来文本分类起步较晚，但也取得了较大的成就。在理论研究方面，姚亮等[62] 使用图卷积网络进行文本分类。基于单词共现和文档单词关系为语料库构建单个文本图，然后为语料库学习一个文本图卷积网络（Text GCN）。其文本图卷积网络用单词和文档的独热编码（one hot）表示初始化，然后它联合学习单词和文档的嵌入，由已知的文档类标签监督。李琛等[63] 提出了一种新颖的用于文本分类的半监督框架，即面向文本的基于图的归纳学习（TextGTL），它在理论指导下细化图拓扑结构并在不同的文本图之间共享信息。TextGTL 还基于图中的密集子结构执行属性空间插值，以使用高质量特征节点预测低熵标签进行数据增强。除了利用深度学习技术来实现的文本分类外，国内也做了很多应用方面的工作。黄瑞等[64] 提出基于不变图卷积神经网络的文本分类，用于通过 GCN 进行归纳文本分类。首先为每个文档构建单个图，使用 GCN 根据其局部结构学习细粒度的单词表示，这可以有效地为新文档中没见过的单词生成嵌入进而将单词节点作为文档嵌入合并；然后提取最大限度地保留不变类内信息的期望子图，使用这些子图进行学习不受分布变化的影响；最后通过图分类方法完成文本分类。孙雪松[65] 在深度学习的网络基础上提出情绪文本分类方法研究，使用 TextCNN 模型对全局特征进行局部特征捕捉,BiLSTM 模型负责捕捉时序特征, 并进一步获取文本上下文特征, 最后进行特征融合, 加强了深度学习模型特征提取能力, 从而提高了模型在文本分类数据集上的预测准确率。通过多种实验对比, 其提出的情绪文本分类混合模型在各项指标上相比较目前通用分类模型均得到有效提升。

伴随着深度学习的模型越来越大，预训练方法也伴随着浮出水面。预训练语言模型能有效地学习全局语义表示，并显著提升包括文本分类在内的自然语言处理任务的效果。它通常采用无监督方法自动挖掘语义知识，然后构建预训练目标，以便机器能

够学习理解语义。

国外率先发起了对于预训练模型的研究。ELMo 模型[66]、OpenAI 提出的 GPT 模型[2] 以及 BERT 模型[12] 都是此时国外提出的可应用于文本分类模型。ELMo 模型是一种深度上下文词表示模型，很容易集成到其他模型中。它可以对单词的复杂特征进行建模，并为不同的语言上下文学习不同的表示。它通过双向长短期记忆网络（LSTM）根据上下文单词学习每个单词的嵌入。GPT 模型采用有监督微调与无监督预训练相结合的方式，学习通用表示，这些表示经过有限的适配后可迁移到许多自然语言处理任务中。此外，目标数据集的领域无需与未标记数据集的领域相似。GPT 模型的训练过程通常包括两个阶段。首先，通过对未标记数据集的建模目标来学习神经网络模型的初始参数。然后，我们可以采用相应的有监督目标，针对目标任务调整这些参数。谷歌提出的 BERT 模型通过在每一层中结合左右上下文进行联合调节，对未标记文本进行深度双向表示的预训练，显著提升了包括文本分类在内的自然语言处理任务的性能。BERT 应用双向编码器，旨在通过联合调整所有层的上下文来预训练深度双向表示。它在预测哪些单词被掩码时可以利用上下文信息。只需添加一个额外的输出层进行微调，就可以构建适用于多种自然语言处理任务的模型，如情感分析、问答系统和机器翻译。与这三种模型相比，ELMo 是一种基于 LSTM 模型[67] 的特征方法，而 BERT 和 OpenAI GPT 是基于 Transformer 模型的微调方法。此外，ELMo 和 BERT 是双向训练模型，而 OpenAI GPT 是从左到右进行训练的。因此，BERT 结合了 ELMo 和 OpenAI 提出的 GPT 模型的优点，取得了更好的效果。

此时的国内更多的是使用预训练模型来进行具体领域的文本分类应用。陈晨等[68] 基于油田安全生产的需求及事故隐患特征，提出了一种基于 BERT-BiLSTM 的分类模型，用于油田安全生产隐患文本的主题自动分类，通过 BERT 模型提取输入文本的字符级特征，生成全局文本信息的向量表示，再通过双向长短时记忆网络（bi-directional long short-term memory, BiLSTM）模型对局部关键信息和上下文深层次特征进行特征提取，进而通过 Softmax 激活函数进行概率计算得到分类结果。周荣等[69] 针对目前讽刺文本分类存在一词多义和准确性不高的问题，提出了一种基于 BERT-BiGRU-CNN 的文本分类模型。首先，使用 BERT 预训练模型对短文本进行句子层面的特征向量表示；其次，通过 BiGRU（双向门控循环单元）获取文本全局序列特征；最后，采用 CNN（卷积神经网络）提取局部重点特征，以此提高模型特征提取性能。韩博等[70] 通过结合标签混淆模型 (Label Confusion Model, LCM), 提出一种基于 BERT 和 LCM 的

文本分类模型 (Model Based on BERT and Label Confusion, BLC)，对文本和标签的特征进一步做了处理。充分利用 BERT 每一层的句向量和最后一层的词向量，结合双向长短时记忆网络 (Bi-LSTM) 得到文本表示，来替代 BERT 原始的文本特征表示。标签在进入 LCM 之前，使用自注意力网络和 Bi-LSTM 提高标签之间相互依赖关系，从而提高最终的分类性能。

尽管国内在文本分类研究领域取得了显著的进展，涌现出了一系列具有创新性的理论和应用成果，然而，与国际领先水平相比，仍然存在一定的差距。国内学者在算法设计、模型优化和实际应用等方面展现了卓越的能力，推动了相关技术的快速发展。然而，国外在基础理论、数据集构建以及跨领域应用等方面积累了更为丰富的经验和资源。因此，尽管我们在这一领域已经取得了一定的成就，但要全面赶上国际先进水平，仍需不断努力，加强国际交流与合作，推动基础研究与应用研究的深入结合，以期在未来的研究中实现更大的突破和创新。

1.3 研究内容及主要贡献

在本文中，提出了如下创新点：

- 提出了一个专门针对教育领域的文本溯源新任务旨在应对在人工智能生成内容日益普及的时代，利用 TOSWT 数据集追踪教育材料来源。
- 创建了教育学中首个可用的句子级别的 AIGT 检测数据集，模型改写文本检测数据集 (TOSWT)，其中包括 5 种最新大模型的生成文本。基于学生写作的短文文本，本数据集构造 prompt 引导大语言模型生成，再经过若干处理，获得了 53,406 条文档级数据和 147,976 条句子级数据。
- 在 TOSWT 数据集上微调了几个基准模型以评估其效果，其中 DeBERTa-V3-Large 模型效果最佳。实验结果表明，该模型在文档级数据上的表现较为优异而在句子级数据上表现较差。
- 搭建模型改写文本检测系统，基于 DeBERTa-V3-Large 模型，构建了一个高效的文本检测系统。该系统能够快速、准确地识别和追踪教育材料的来源，为教育工作者和研究人员提供了有力的工具。

1.4 论文组织结构

本文具有以下结构：

1. **第一章**介绍了关于模型改写文本检测这一任务的研究背景及意义、追根溯源其原理文本生成和文本分类的国内外研究现状及主要贡献以及本论文的组织结构。

2. **第二章**介绍了回顾了自然语言处理领域的相关研究，重点关注文本生成技术和预训练语言模型的进展。先分析了 Transformer 架构的优势，再详细讨论了 BERT 模型的结构及其在预训练和微调过程中的创新。最后，探讨了 AI 生成文本的检测技术，介绍了多种现有方法及其在实际应用中的效果与局限性。

3. **第三章**介绍了详细探讨了模型改写文本检测数据集（TOSWT 数据集）的构建方法，旨在为教育工作者提供一个有效的工具，以分析和追踪学生文本的来源。在数据集的构建过程中，我们以 Learning Agency Lab 发布的 Automated Essay Scoring 2.0 (AES2) 数据集为基础使用正则表达式，我们成功替换了非 ASCII 字符，再使用句子分割技术，细粒度构建数据集。引入了多种大型语言模型，包括 ChatGPT、GPT4o、Gemini、Qwen 和 DeepSeek。最终，构建出的数据集包含 53,328 篇短文和 147,976 个句子，为后续的文本检测研究提供了丰富的数据支持。此外，本章还深入探讨了文本来源追踪的数学化表述，将其视为一个文本分类问题。

4. **第四章**介绍了实验结果与分析，包括对比实验等。本章系统地研究了基于预训练语言模型的文本改写检测方法，通过对比实验和消融分析，深入探讨了不同模型架构在文档级和句子级文本检测任务中的表现差异。对比实验结果表明，DeBERTa-V3-Large 在文档级任务中表现最优，准确率达到 85.97%，显著优于其他对比模型。在任务粒度方面，研究发现文档级和句子级任务呈现出不同的性能特征。所有模型在文档级任务上的表现明显优于句子级任务。在探索性数据分析上，本章提出了基于余弦相似度和评分一致性的双重评估框架。

5. **第五章**介绍了模型改写文本检测系统的设计与实现，包括需求分析、系统架构设计、模块和接口设计以及数据库设计。通过对系统的功能进行详细分析，明确了系统的业务需求和功能需求，为后续的系统实现提供了基础。系统采用前后端分离的架构设计，提高了系统的灵活性和可扩展性。通过模块和接口设计，

明确了系统各个模块之间的关系和数据传输方式。数据库设计则为系统的数据存储提供了支持。最后，通过用户注册与登录、文本改写功能、模型改写文本检测功能等展示了系统的实际应用效果。

第 2 章 相关工作

2.1 文本生成技术

在当前的自然语言处理领域，主流的大语言模型普遍采用了 Transformer 框架[1]。这一架构自 2017 年提出以来，因其卓越的并行处理能力和长距离依赖建模能力而迅速成为研究的热点。Transformer 通过自注意力机制有效捕捉输入序列中各个部分之间的关系，使得模型能够在处理大规模文本数据时，显著提高了训练效率和表达能力。诸如 BERT、GPT 系列等知名模型，均基于这一框架，展现出了在多种语言任务上优异的性能。因此，深入探讨 Transformer 的结构与原理，不仅有助于理解当前大语言模型的成功之处，也为未来的研究提供了重要的理论基础。

Transformer 是 Google 于 2017 年提出的一种基于自注意力机制的用于处理 Seq2Seq 任务的神经网络模型，被广泛用于各种 NLP 任务中。Transformer 利用自注意力机制来处理输入序列中的关联性，能够有效地捕捉长距离依赖关系，无需依赖于循环神经网络或卷积神经网络。同时，Transformer 可以并行地处理输入序列中的不同位置，因此在训练和推理时，特别是在处理长序列时具有更高的效率。

2.1.1 编码器-解码器结构

（1）编码器

编码器有 $N = 6$ 个完全一样的 TransformerEncoderLayer 层。在每一个 TransformerEncoderLayer 层中，可以将每一层分成两个子层。第一个子层是多头的自注意力机制，第二个子层仅仅只是朴素的位置全连接前馈网络。这两个子层中，每一个子层后面都要接一个残差连接网络，这之后再把经过残差连接层的变量进行归一化。简言之，每一个子层输出出来的向量实际上是 LayerNormal$(x + \text{Sublayer}(x))$，上式中 Sublayer(x) 表示的是每一个子层自己实现的功能。为了使得残差连接的维度数相等以及方便残差连接计算等一系列原因，上述模型中的所有子层以及嵌入层的输出向量的维度数量都为 $d_{\text{model}} = 512$。

（2）解码器

解码器也有 $N = 6$ 个完全一样的 TransformerDecoderLayer 层。在每一个 TransformerDecoderLayer 层中，包含了三个子层。其中两个子层与编码器层中的子层大体相似，而第三个子层，这个子层输入编码器堆栈的输出且输入解码器层的上一个多头

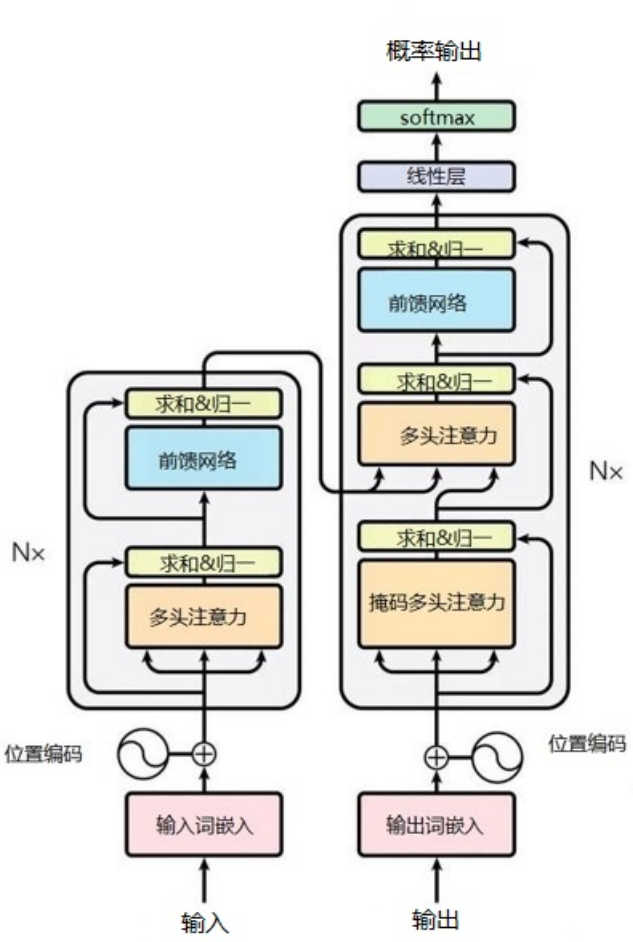


图 2.1　Transformer 模型结构

注意力子层的输出，并在该子层内执行多头注意力操作。与编码器层中的子层的结构类似，把残差连接网络接到每一个解码器层的子层后面，这之后再把经过残差连接层的变量进行层归一化。本课题也修改了解码器结构中的自注意力子层，加入了位置掩码，这样可以防止出现从前面的某个位置获得后面某个位置的信息。这种位置掩码再加上一个位置的偏移，可以使得对于序列中位置 i 的元素的预测内容只能通过序列中小于 i 位置的全部内容得出。

2.1.2 注意力机制

在 Transformer 中，定义注意力函数为输入为查询和一组键值，然后进行输出的函数，在此之中查询（Query, Q）、键（Key, K)、值（Value, V）和输出都是向量。在注意力函数中，简单地讲，输出实质上是值的加权和。此其中对于每个值的权重由权重相应的键以及查询计算得出。

（1）缩放点积注意力

在 Transformer 论文中使用的注意力机制为缩放点积注意力。缩放点积注意力机制的输入包括了查询、键、值。其中，查询和键维度数同为 d_k 以及值的维度数为 d_v。在缩放点积注意力中，我们基于所有键这些向量并以此来计算键与查询的点积，在此之后将所有得到的向量都除以 $\sqrt{d_k}$，在此之后将这些向量通过 softmax 函数来计算出来每个值对应乘起来的权重，具体过程如图 2.2 所示。

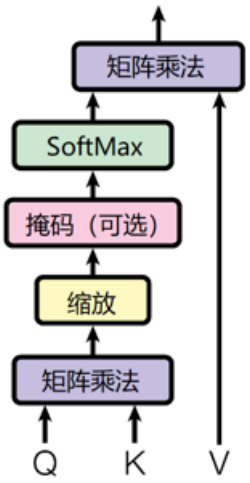


图 2.2　缩放点积注意力

具体落实在实践中，对于一组亟待查询的注意力函数，应当将这些查询同时计算，封装到一个矩阵 Q 中。键和值也封装到矩阵 K 和 V 中。这样在具体计算中，就可以

使用矩阵乘法的方法来予以优化。这样的话，可以把注意力函数中输出的矩阵计算为：

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

在 Transformer 之前的研究中，加法注意和点积的乘法注意是在此之前最常用的两个注意力函数。点积注意力与 Transformer 的缩放点积注意力算法相比，除了 $\frac{1}{\sqrt{d_k}}$ 的比例因子之外，都差不多，没有什么不同。加法注意的方法在计算的时候使用具有单个隐藏层的前馈网络。即使加法注意和点积注意这两个注意力方法在理论上的复杂性相似，但是点积注意力由于使用可以并行化且优化非常棒的矩阵乘法来实现，所以在实践中代码跑的更快且代码更节省空间一些。

在查询和键的维度数 d_k 较小的时候，这两种机制的性能相似。但是在 Transformer 原论文中有提及，查询和键的维度数 d_k 较大时，矩阵相乘后的造成的矩阵中的元素的值会变大，进而使得经过 softmax 函数后的矩阵中的每个元素的梯度反而变得更小，更难以训练且难以出结果。为了抵消这种影响，我们将点积乘上 $\frac{1}{\sqrt{d_k}}$，以缩小其中元素的值。

（2）多头注意力

其实，如果将键、值和查询都使用 d_{model} 维度的话，相比于将查询、键和值直接投影到 d_k、d_k 和 d_v 维度上做不同的学习线性投影实际上是对这个模型反而是会变得更劣。在此之后，在每个查询、键和值的投影向量上，代码并行地执行注意力函数，并产生 d_v 维输出值。上一步输出出来的 d_v 维输出值被连接起来并再次投影，再输出出去，具体过程如图 2.3 所示。

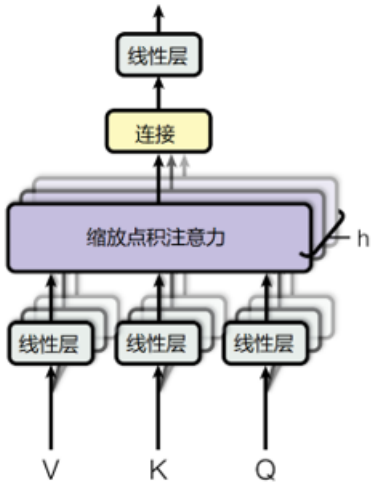


图 2.3　多头注意力

多头注意力的优点就是可以让注意力模型可以同时关注在不同位置上的信息。而如果只有单个注意力头的话，平均化会抑制这一点。

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,...,\text{head}_h)W^O \tag{2.2}$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.3}$$

其中投影是参数矩阵 $W_i^Q \in \mathbb{R}^{d_{\text{model}}}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}, W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$

在 Transformer 中，使用了 $h = 8$ 个并行注意力层或头。对于其中的每一个，使用 $d_k = d_v = d_{\text{model}}/h = 64$。由于每个头的维度减少，总计算成本类似于具有全维度的单头注意力。

2.1.3 位置前馈网络

除了注意力子层之外，编码器和解码器中的每一层都包含一个完全连接的前馈网络，该网络分别且相同地应用于每个位置。这包括两个线性变换，中间有一个线性整流函数（ReLU）激活。

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.4}$$

虽然线性变换在不同位置上是相同的，但它们在层与层之间使用不同的参数。另一种描述方式是内核大小为 1 的两个卷积。输入和输出的维数为 $d_{\text{model}} = 512$，内层的维数为 $d_{ff} = 2048$。

2.1.4 嵌入层和 softmax 函数

与其他序列转导模型类似，Transformer 使用学习嵌入将输入序列中的词汇和输出序列中的词汇转换为维度 d_{model} 的向量。除此之外，还使用了通常的学习线性变换和 softmax 函数将解码器输出转换为预测的下一个词汇的概率。在我们的模型中，我们在两个嵌入层和 pre-softmax 线性变换之间共享相同的权重矩阵。在嵌入层中，我们将这些权重乘以 $\sqrt{d_{\text{model}}}$。也有使用 log-softmax 函数替代 softmax 函数的情况。

2.1.5 位置编码

由于 Transformer 模型不包含递归和卷积，为了让模型利用序列的顺序，必须注入一些关于标记在序列中的相对或绝对位置的信息。为此，我们在编码器和解码器堆栈

底部的输入嵌入中添加“位置编码”。位置编码与嵌入具有相同的维度 d_{model}，因此可以将两者相加。位置编码有很多选择，由模型学习出来的或是公式计算固定出来的。

在 Transformer 中，使用不同频率的正弦和余弦函数：

$$\begin{aligned} \text{PE}(pos, 2i) &= \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \\ \text{PE}(pos, 2i+1) &= \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \end{aligned} \tag{2.5}$$

其中 pos 是位置，i 是维度。也就是说，位置编码的每个维度对应一个正弦曲线。波长形成从 2π 到 $10000 \cdot 2\pi$ 的几何级数。我们选择这个函数是因为我们假设它可以让模型轻松学习通过相对位置来参与运算，因为对于任何固定的偏移量 k，PE_{pos+k} 可以表示为 PE_{pos} 的线性函数。

使用公式固定计算出来的正弦版本好处有，它可以让模型推断出比训练期间遇到的序列长度更长的序列长度。

2.1.6 最后的线性层

数据通过 log-softmax 函数之后，将通过一层线性层将这个维度为 d_{model} 的向量转化为维度为词汇个数的向量，该向量表示下一个预测的词汇是什么。

2.2 预训练语言模型技术

近年来，自然语言处理（(Natural Language Processing，NLP）的发展取得了巨大进步，其中预训练语言模型（PLMs，Pre-trained Language Models）的研究为自然语言处理的进展提供了关键支持。其核心概念是首先在大型数据集上对深度神经网络进行预先训练，以获取模型参数，然后利用这些经过训练的模型来执行各种具体的下游任务，从而避免从头开始训练模型并减少对标注数据的需求量。与预训练语言模型思想相关的早期工作可以追溯到 NNLM[71] 和 word2Vector[72]，2017 年 Vaswani 等人[1] 提出了 Transformer，基于 Transformer 的预训练语言模型大大提高了各种 NLP 任务的性能，由此 Transformer 成为各种预训练语言模型的主干。基于 Transformer 的编码端（Encoder），Radford 等人[2] 提出了 GPT 模型，GPT 真正意义上实现了预训练-微调的框架，不再需要将模型中的词嵌入取出来，而是直接把预训练好的模型在下游任务上微调，对于不同任务采用不同的输入或对输出层改造，让下游任务更贴近上游预训练模型，目前 GPT 模型的应用取得了巨大成功，各种 GPT 大模型层出不穷，给 NLP 等领域

注入了新的活力。Devlin 等人[12] 基于 Transformer 的 Encoder 编码端提出 BERT 模型，也是目前在 NLP 中应用最广泛的预训练模型之一，BERT 采用了一种 Mask Language Model（MLM）的方法，通过随机 mask 掉输入文本中的某些词元，然后利用上下文信息进行预测，实现对数据语义关系的提取。继 GPT 和 BERT 之后，研究人员又提出了 XLNet[73]、RoBERTa[74]、DeBERTa[75] 、ERNIE[22]、T5[76] 和 BART[13] 等预训练语言模型。本节将对本文所使用模型 RoBERTa 和 DeBERTa 的基础模型 BERT 预训练语言模型进行介绍。

BERT(Bidirectional Encoder Representation from Transformers)[12] 是 2018 年 10 月由 Google AI 研究院提出的一种预训练模型，该模型在机器阅读理解顶级水平测试 SQuAD1.1[77] 中表现出惊人的成绩: 全部两个衡量指标上全面超越人类，并且在 11 种不同 NLP 测试中创出 SOTA 表现，包括将 GLUE 基准[78] 推高至 80.4% (绝对改进 7.6%)，MultiNLI 数据集[79] 准确度达到 86.7% (绝对改进 5.6%)，成为 NLP 发展史上的里程碑式的模型成就。BERT 在 Transformer 的基础上引入了预训练和微调的策略，使得模型在预训练阶段学习通用的语言表示，而在特定任务中进行微调，极大地提升了模型的适应性和性能。因此，尽管 BERT 不是第一个预训练模型，但它在预训练模型的发展历程中具有里程碑式的意义。

2.2.1 BERT 结构

BERT 在 Transformer 基础上做的改进不多，采用了 Transformer 编码器的结构作为基础结构。总体结构图如 2.4 所示，多个 Transformer 编码器层一层一层地堆叠起来，就是 BERT 的总体结构了，在原始论文中，作者分别用 12 层和 24 层 Transformer 编码器组装了两套 BERT 模型，110M 和 340M 分别为两套模型的总参数。

2.2.2 嵌入

对于 BERT 而言，一个巨大的改进就是其嵌入（Embedding）上的改进。为了使 BERT 能够处理多种下游任务，BERT 的输入表示能够在一个词元序列中清晰地表示单个句子和一对句子 (例如, 〈问题, 答案〉)。在整个工作中，“句子”可以是一个任意跨度的连续文本，而不是一个实际的在自然语言意义下的句子。“序列（sequence）”是指输入 BERT 的词元序列，它可能是单个句子，也可能是多个句子打包在一起。

在 BERT 中使用带有 30，000 个词元词汇表的 WordPiece 嵌入。每个序列的第一个标记始终是一个特殊的分类标记（[CLS]）。与此标记对应的最终隐藏状态用作分类

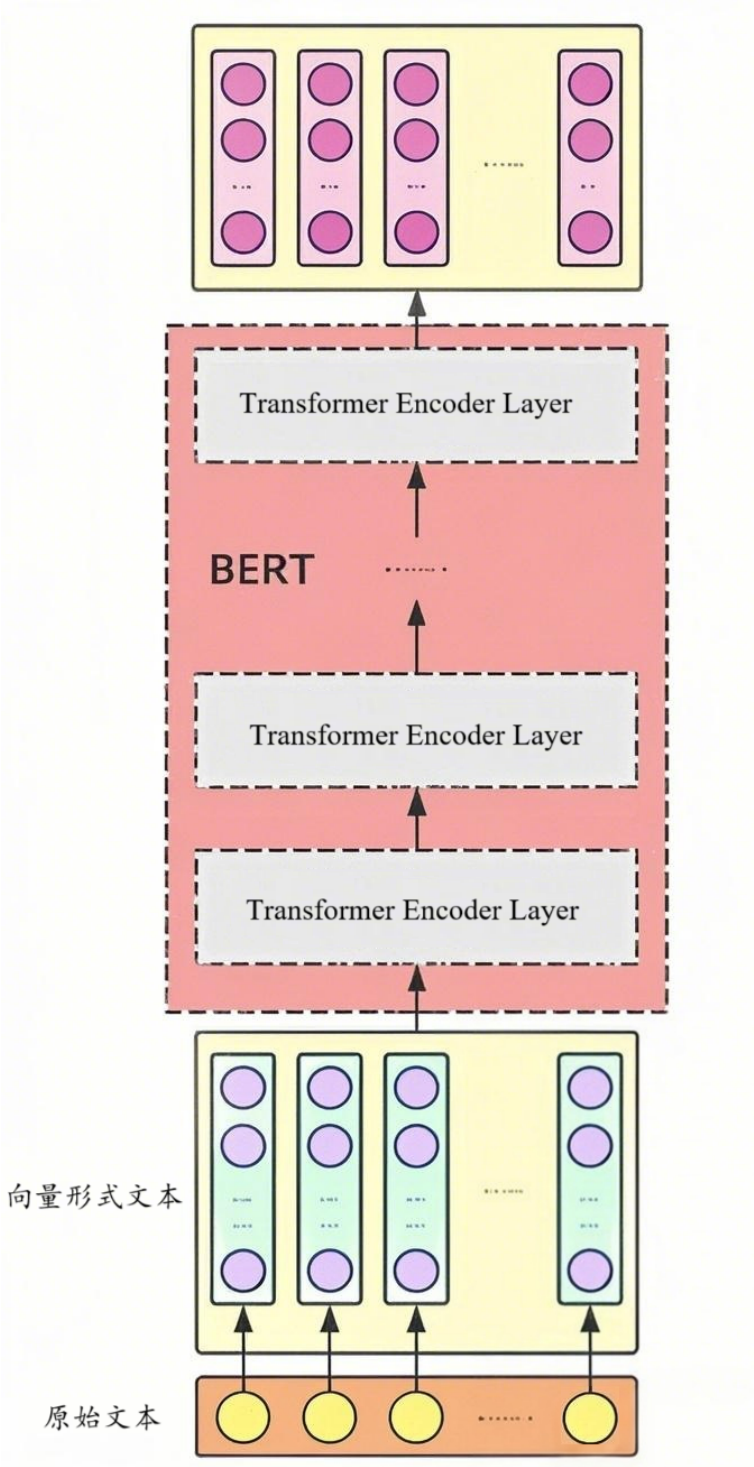

Transformer Encoder Layer
BERT
Transformer Encoder Layer
Transformer Encoder Layer
向量形式文本
原始文本

图 2.4　BERT 架构图

任务的聚合序列表示。最后一层该位对应向量可以作为整句话的语义表示，从而用于下游的分类任务等。因为与文本中已有的其它词相比，这个无明显语义信息的符号会更“公平”地融合文本中各个词的语义信息，从而更好的表示整句话的语义。具体来说，self-attention 是用文本中的其它词来增强目标词的语义表示，但是目标词本身的语义还是会占主要部分的，因此，经过 BERT 的 12 层（BERT-base 为例），每次词的 embedding 融合了所有词的信息，可以去更好的表示自己的语义。而 [CLS] 位本身没有语义，经过 12 层，句子级别的向量，相比其他正常词，可以更好的表征句子语义。在 BERT 执行过程中，句子对被打包到一个单一的序列中。在此情况下，通过两种方式对句子进行区分。首先，用一个特殊的标记（[SEP]）将它们分开。其次，BERT 将学习到的嵌入添加到每个词元中，指示它是属于句子 A 还是句子 B。

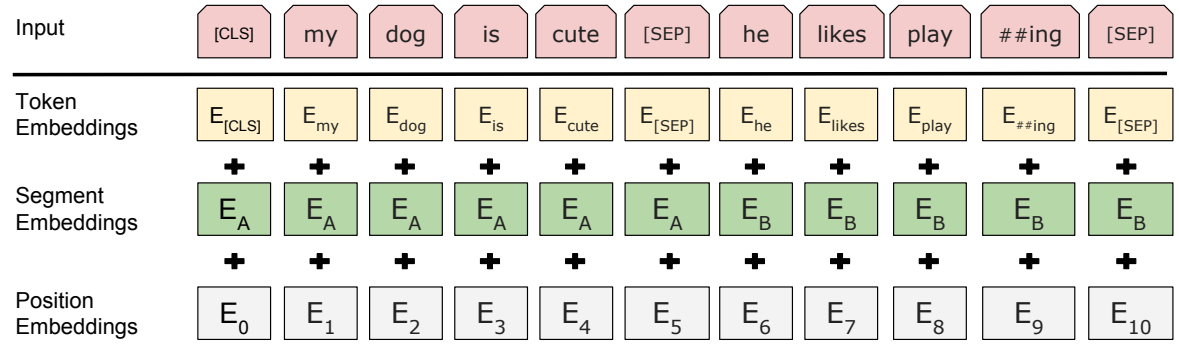


图 2.5　BERT 嵌入（Embedding）示意图

对于给定的词元，其输入表示（input representation）是通过对相应的词元（token）、段（segment）和位置（position）嵌入进行求和来构造的。这个结构的可视化如图 2.5 中所示。

2.2.3 预训练与微调

BERT 这项工作最大的贡献是将二阶段训练过程发扬光大，其训练过程分为预训练过程（pretraining）和微调过程（finetuning），训练过程示意图如图 2.6 所示。预训练阶段旨在通过大规模的无标注文本数据来学习语言的深层次表示，BERT 采用了带掩码的语言模型（Masked Language Model，MLM）和下一句预测（Next Sentence Prediction，NSP）两种任务，使模型能够理解上下文的双向信息。在微调阶段，BERT 可以针对特定任务进行调整，只需在少量标注数据上进行训练，从而能够高效地适应多种自然语言处理任务，如文本分类、问答系统和命名实体识别等。

这种预训练-微调的二阶段训练范式显著提升了模型在各种 NLP 任务上的性

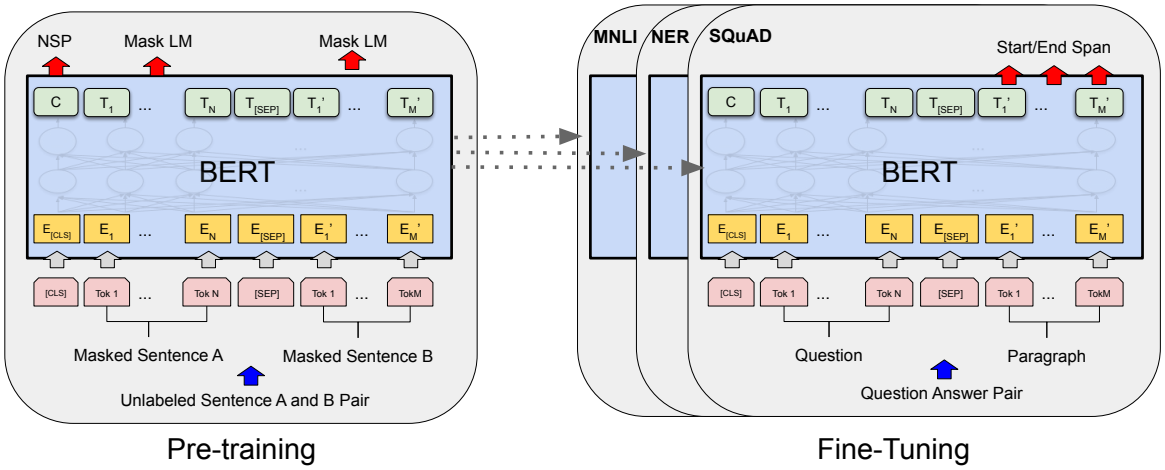


图 2.6　BERT 训练过程示意图

能，标志着自然语言处理领域的一次重大变革。BERT 的成功也激励了后续大量基于 Transformer 架构的预训练模型的研发，例如本文要使用的 RoBERTa 和 DeBERTa，以及其他在 BERT 模型上做出改进的模型，这些模型在不同的任务中展示了更高的效率和准确性。通过这些创新，BERT 不仅推动了学术界对预训练模型的研究，也促进了工业界在实际应用中的广泛采用。

2.3　AI 生成文本探测技术

大型语言模型（LLMs）的快速发展推动了对检测 AI 生成文本方法的深入研究。在 ChatGPT 问世之前，GLTR[80] 作为一种检测工具，通过一系列统计方法识别由不同采样方案生成的文本伪影。GLTR 有效地将人类对生成文本的检测率从 54% 提升至 72%，突显了其在帮助非专家区分人类与机器生成内容方面的实用性。

另一个重要贡献是人类 ChatGPT 比较语料库[8]（HC3），该语料库是在 ChatGPT 出现后开发的。HC3 数据集包含来自人类专家和 ChatGPT 的数万个响应，涵盖金融、医学和法律等多个领域。这一数据集使得对 ChatGPT 的响应与人类输出进行全面评估成为可能，揭示了 LLMs 的能力与局限性。此外，HC3 还促进了检测系统的发展，旨在识别文本是由 ChatGPT 还是人类生成的。

LLM-Detector[9] 则解决了现有人工智能生成文本检测模型面临的挑战，这些模型通常因领域内过拟合而在领域外表现不佳。该研究通过收集人类专家和各种大语言模型的回复，依托于大型语言模型提出了一种新颖的检测方法，利用指令调优来增强文档级和句子级的检测能力。实验结果表明，该方法在性能上显著优于基线方法，展现

了强大的泛化能力。

SeqXGPT[10] 通过合成一个包含大语言模型生成句子和人类写作内容的数据集，引入了句子级检测挑战。该方法利用来自白盒 LLMs 的对数概率列表，结合卷积和自注意力网络来提高检测准确性。结果显示，SeqXGPT 在句子和文档级检测任务中均优于现有方法，进一步强调了有效检测机制的必要性。

针对人工智能生成文本的来源追踪问题[11]，有研究提出了一种新颖的算法，用于检测和追踪人工智能生成的内容。该方法利用对比特征来区分由不同模型生成的文本，能够在白盒和黑盒环境中有效运行。研究结果强调了人工智能部署的伦理影响以及追踪生成文本来源机制的必要性，为有关人工智能系统问责制的持续讨论提供了重要依据。

然而，现有的方法[8-10,80] 未能满足我们对文本来源于特定大型语言模型的检测需求。大多数研究在分类任务上表现较为粗糙，仅能识别文本是由人类还是机器生成，无法进一步确定文本的具体来源。尽管少部分研究[11] 实现了识别文本来源于特定大型语言模型的目标，但其实现方式依赖于白盒测试，必须获取与待检测模型一致或相似的模型生成的特征向量才能进行判断。在当前大多数大型语言模型均为闭源的环境下，这种方法几乎缺乏扩展性和可操作性。

2.4 本章小结

本章回顾了当前自然语言处理领域的相关工作，重点聚焦于文本生成技术和预训练语言模型的进展。首先，我们深入分析了 Transformer 架构的优势，强调了其在大语言模型中的广泛应用。自 2017 年提出以来，Transformer 因其卓越的并行处理能力和长距离依赖建模能力，迅速成为自然语言处理研究的核心。通过自注意力机制，Transformer 能够有效捕捉输入序列中各个部分之间的关系，从而显著提高了模型在处理大规模文本数据时的训练效率和表达能力。

随后，我们详细讨论了 BERT 模型的结构及其在预训练和微调过程中的创新。BERT 不仅在机器阅读理解等多个任务中取得了显著的性能提升，还引入了预训练-微调的二阶段训练范式，使得模型能够在预训练阶段学习通用的语言表示，而在特定任务中进行微调。这一策略极大地提升了模型的适应性和性能，标志着自然语言处理领域的一次重大变革。在预训练语言模型的讨论中，我们提及了如 GPT、RoBERTa 和 DeBERTa 等模型的相继出现，这些模型在 BERT 的基础上进行了不同程度的改进，进

一步推动了预训练模型的研究和应用。

最后，本章探讨了 AI 生成文本的检测技术，介绍了多种现有方法及其在实际应用中的效果与局限性。尽管 GLTR、HC3、LLM-Detector、SeqXGPT 等工具在识别生成文本方面取得了一定的成效，但仍面临特定大型语言模型检测的挑战。尤其是在当前许多大型语言模型为闭源的环境下，现有方法在文本来源追踪和特定模型检测方面的适用性显得尤为重要。

综上所述，本章的讨论为理解当前技术状态及未来研究方向提供了重要视角，为本论文的研究奠定了坚实的基础。通过对 Transformer 架构、BERT 模型及 AI 文本检测技术的全面回顾，我们为后续的研究和实验提供了必要的理论支持和实践指导。

第 3 章 模型改写文本检测数据集构建方法

3.1 引言

在前两章中，我们详细介绍了探测 AI 生成文本的前世今生。在本章节中，我们将构造出属于自己的数据集。我们的数据集名称叫做追踪学生写作文本来源数据集（Tracing the Origins of Students’ Writing Texts, 以下简称 TOSWT 数据集）在前人的基础上，我们首先要在教育学领域，为教师提供分析文本来源的工具数据集，其次要将数据集更加细粒度，例如探索句子级别，使得实验有着更广泛的应用性，可以帮助教师敏锐地指出学生文本中哪一句经过了什么大型语言模型的修改。

在 Learning Agency Lab 发布的 Automated Essay Scoring 2.0[81] (AES2) 数据集的基础上，TOSWT 数据集应运而生。AES2 数据集是一个用于自动写作评价的数据集，包含了 10,584 篇由 9-12 年级学生撰写的短文，并为每篇文章提供了 1-6 分的评分，其中高质量的文本将获得更高的评分。我们在 AES2 数据集的基础上，对原始数据进行了清洗，利用正则表达式替换非 ASCII 字符，并使用句子分割工具将短文划分为句子。此后，我们选用了一种 prompt 进行引导大语言模型改写处理后的短文。

Automated Essay Scoring 2.0 数据集发布于 Kaggle 平台上，为解决自动写作评估问题召开比赛，作为比赛用数据集。短文写作是评估学生学习和表现的重要方法。教育者手工评分也很耗时。自动写作评估（Automated Writing Evaluation, AWE）系统可以对论文进行评分，以补充教育者的其他努力。自动写作评估系统还允许学生定期及时地收到关于他们写作的反馈。但是，由于其成本，该领域的许多进步并没有广泛地提供给学生和教育者。评估学生写作的开源解决方案需要用这些重要的教育工具到达每个社区。

以前开发开源自动写作评估系统的努力受到小型数据集的限制，这些数据集不是广泛性的，也不专注于常见的短文格式。之前在 Kaggle 平台上举办的第一届自动论文评分竞赛对学生写的简短答案进行评分，然而，这是一项课堂上不常使用的写作任务。为了改进以前的努力，需要一个更广泛的数据集，包括高质量、现实的课堂写作样本。此外，为了扩大影响，数据集应该包括跨经济和地点人群的样本，以减轻算法偏见的可能性。

因此 Learning Agency Lab 发布 Automated Essay Scoring 2.0 数据集，这是当时最

大的开放获取写作数据集，且该数据集符合当时学生的评估标准。AES2 数据集示例如表 3.1 所示，其中仅列举 1-3 分数据样例各一个，并省略掉了原文中的换行符“\n”。

在 AES2 数据集中，他们应用如下的评分标准来给学生写作短文打分，分数从低到高是 1 分（最低）到 6 分（最高）。与下面所描述的评分标准一样，每个分数之间的差距（例如 1-2 分、3-4 分、4-5 分）应尽可能相等。

- **6 分**：该类文章展现出清晰且稳定的高水平能力，尽管可能存在一些小错误。典型的文章能就议题有效且深刻地阐述观点，展现出卓越的批判性思维；文章能从原文中选取恰当的例子、理由及其他证据来支撑其观点；文章结构合理、重点突出，展现出清晰的连贯性和流畅的思路推进；文章语言运用娴熟，词汇丰富、准确、恰当，句子结构富有变化；文章在语法、用法和书写规范方面基本没有错误。

- **5 分**：该类文章展现出较为稳定的高水平能力，尽管偶尔会出现错误或质量上的小瑕疵。典型的文章能就议题有效阐述观点，展现出较强的批判性思维；文章通常能选用恰当的例子、理由及其他证据来支撑其观点；文章结构良好、重点明确，展现出连贯性和思路推进；文章语言运用熟练，词汇恰当，句子结构有变化；文章在语法、用法和书写规范方面基本没有错误。

- **4 分**：该类文章展现出足够的能力，尽管在质量上会有起伏。典型的文章能就议题阐述观点，展现出合格的批判性思维；文章能运用足够的例子、理由及其他证据来支撑其观点；文章大致结构合理、重点突出，展现出一定的连贯性和思路推进；文章语言运用能力有所波动，词汇基本恰当，句子结构有一定变化；文章可能存在一些语法、用法和书写规范方面的错误。

- **3 分**：该类文章展现出正在发展的能力，但存在以下一项或多项不足：能就议题阐述观点，有一定批判性思维，但可能表现不稳定，或使用的例子、理由及其他证据不足以支撑观点；文章在结构或重点方面存在局限，或在连贯性和思路推进上有不足；文章语言运用有一定能力，但有时词汇使用较弱、选词不当，和 / 或句子结构缺乏变化或存在问题；文章可能存在语法、用法和书写规范方面的错误累积。

- **2 分**：该类文章展现出的能力较弱，存在以下一项或多项缺陷：对议题的观点模

表 3.1　AES2 数据集数据示例

essay_id	原文	评分
eb8f967	Honestly, who would think there is ”Alien Form” on Mars, I mean some of the people are just plain stupid, and some have facts and details. I think that these so called aliens are just Fallen Angels, because I read the Bible. These are simply just natural lanforms. When you take a look at this structure it does look like a humans face. Then again, I dont think it is an alien monument because, from how far you look at it and how far they took pictures from it. I don’t think aliens could have made a face a couple miles long, it would have to take years to do this. Anyhow, I do believe there was life on the planet, I know there was water, because they have lots of craters, canyons, and mountains. My conclusion is, they were formed by natural occurances. As in the passage they compared the face to some features in the American West such as, Middle Butte, and Snake River Plain of Idaho in the West. These conlusions matched the face up to natural formation. The face formation is known as a messa or a plataeu.	1
7913c3a	My position on driverless cars is very nutral. Like the passage stated there are a lot pros and cons when it comes to this topic, such as numerous types of road blocks like accidents and construction. When it comes to the diverless car getting to an accident people are confused on who it should be to blame the manufacturer, or the passenger. I personaly believe the passenger should be to blame, it doesn’t matter how advanced the technology is something is bound to go wrong so you should always be on alert. When it comes to some of the pros there are some strong ones. Driving will be a lot safer once a majority of people own a self driving car. There will be almost no accidents do to texting, falling asleep at the wheel, and drunk driving. Once we get to the point of almost everyone owning a self diving car they will most likely be eco-friendly.	2
5d147d6	Having the new technolgy would help every teacher and school around the world. By being able to tell how a student feels the teacher can change the way they would teach the classes or the way they would a explain a lesson. The computer would tell the teacher how the students are feeling. It would be valuable to have that type of technolgy in class room becuase it would make teachers jobs less complicated. Using the new technology would hellep teacher understand the students in knowing how they feel whithout having to ask them. ”A classroom computer could recognize when a student is becoming confused or bored”. In this example Dr. Huang explains what the cumputer is able to do it. By telling how a student is feeling this can help teachers be the teacher not having to ask every student how they feel. The computer would do it for the teacher. By having the computer know how the student feels the teaacher can focus more on exlpaing the lesson better and making it easier to understand for students to understand. ”This could modify the lesson,like an effective human instuctor” Dr. Huang elpaians that the computer can change it is teaching the lesson like a teacher would it also states that it would do it as effective as a human instructor would. That means having that technolgy would help improve schools teach improve classes and the way people are being tought.	3

糊或极为有限，批判性思维薄弱；文章提供的例子、理由及其他证据不恰当或不充分，无法支撑观点；文章结构混乱和 / 或重点不明确，或在连贯性和思路推进上存在严重问题；文章语言运用能力极低，词汇量极为有限、选词错误，和 / 或句子结构频繁出现问题；文章存在严重的语法、用法和书写规范错误，在一定程度上影响了意思表达。

- **1 分**：该类文章展现出的能力极低或几乎没有，存在以下一项或多项严重缺陷：无法就议题提出可行的观点，或几乎没有提供证据支撑观点；文章毫无条理、没有重点，导致内容不连贯、逻辑混乱；文章在词汇使用上存在根本性错误，和 / 或句子结构存在严重缺陷；文章存在大量语法、用法或书写规范错误，严重干扰了意思的表达。

3.2 问题阐述

文本来源追踪的任务可以数学化地表述为一个文本分类问题。我们定义一个文本语料库 $T = \{t_1, t_2, \ldots, t_n\}$，其中每个文本 t_i 被表示为一个字符串，表示一段文本内容。我们的目标是将每个文本 t_i 分类到一个预定义的类别集合 $C = \{c_1, c_2, \ldots, c_k\}$。

这一过程可以通过以下步骤进行阐述：

1. **模型训练**：构建一个分类模型 $f : T \to C$，该模型从训练数据集 $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_p, y_p)\}$ 中学习。在这里，$y_i \in C$ 表示文本 t_i 的真实类别，而 $x_i \in T$ 则代表相应的文本内容。模型通过学习这些样本之间的关系，逐渐形成对文本特征和类别之间映射的理解。

2. **分类过程**：对于新的文本 t_{new}，特征提取过程生成 x_{new}。随后，利用训练好的模型进行预测：

$$\hat{y} = f(x_{new})$$

其中，$\hat{y}$ 表示模型预测的类别。这个步骤的核心在于模型能够有效地将新文本映射到其可能的类别中，展现出其在特征空间中的判断能力。

3. **评估**：模型的性能通过计算一系列评估指标来进行评估，这些指标包括准确率、精确率、召回率和 F1 分数等。这些指标不仅帮助我们衡量模型在训练集上的表

现，还能够评估其在未见数据上的泛化能力，从而确保模型在实际应用中的有效性。

通过上述步骤，我们能够系统地处理文本来源追踪的任务，将其转化为一个可操作的文本分类问题。这一方法论不仅为文本分类提供了清晰的框架，也为后续的研究和应用奠定了坚实的基础。

3.3 模型改写文本检测数据集构建方法

在 Learning Agency Lab 发布的 Automated Essay Scoring 2.0 (AES2) 数据集的基础上，TOSWT 数据集应运而生。AES2 数据集是一个用于自动写作评价的数据集，包含了 10,584 篇由 9-12 年级学生撰写的短文，并为每篇文章提供了 1-6 分的评分，其中高质量的文本将获得更高的评分。我们在 AES2 数据集的基础上，对原始数据进行了清洗，利用正则表达式替换非 ASCII 字符，并使用句子分割工具将短文划分为句子。此后，我们选用了一种 prompt 进行引导大语言模型改写处理后的短文。详细过程如图 3.1 所示。

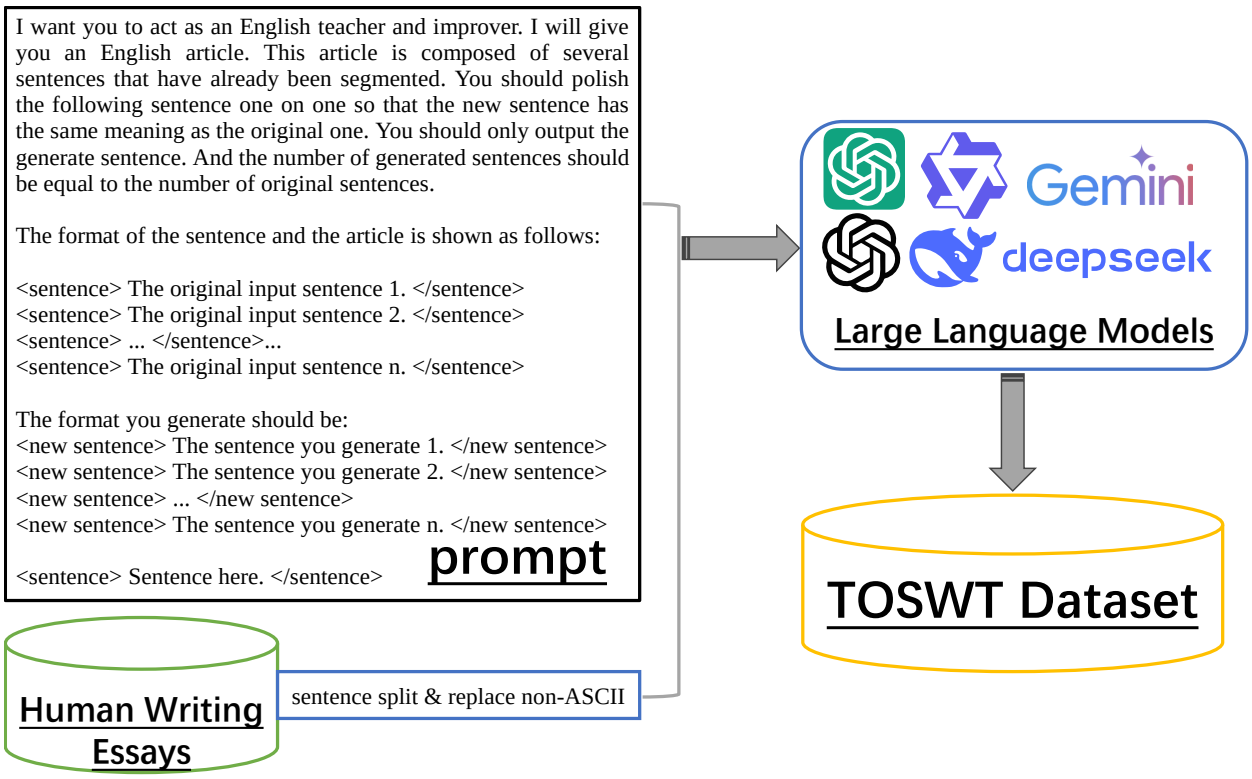


图 3.1　TOSWT 数据集的构造过程图

生成 TOSWT 数据集时使用 prompt 翻译表如表 3.2 所示。整体分成四个段落，第一段为任务的描述，二、三段则给出输入及输出的格式，最后第四段给予大语言模型

输入的句子文本。

这个 prompt 旨在帮助用户提升英语写作能力，具体通过将一篇英语文章进行润色与改进。它设定了用户作为英语教师和改进者的角色，引导用户逐句处理文本，以确保新生成的句子在意义上与原句一致。每个句子都被要求逐一润色，从而增强语言的流畅性和自然度。在执行过程中，用户将接收到一篇由多个已分割句子组成的文章。为了确保生成句子的数量与原句相等，prompt 设定了清晰的输出要求，用户只需专注于生成的新句子而无需担心格式问题。这种简洁的设计使用户能够更高效地进行文本修改。此外，prompt 提供了具体的格式示例，帮助用户理解所需的输出结构。这种结构化的指导不仅提高了工作效率，还确保了生成文本的规范性和一致性，进而为用户提供了系统的学习与实践体验。通过这种方式，用户能够在实际操作中获得更深刻的语言理解与应用能力。

3.3.1 正则表达式替换非 ASCII 字符

正则表达式替换符号如表 3.3 所示。在 AES2 数据集以及大型语言模型改写后的文本中均使用正则表达式将非 ASCII 字符替换为相对应的 ASCII 字符，以保证非 ASCII 字符不会影响数据集的效果。

\u00a0 和 \u2019 均为 Unicode 编码而非 ASCII 字符，从表中可知，常见的非 ASCII 字符包括空格、左右双引号、左右单引号、字母变体、短划线等。

3.3.2 细粒度分割获取句子

在构建 TOSWT 数据集的过程中，为了使用尽可能细粒度的数据进行训练，因此使用到了句子分割的技术。我们使用的句子分割技术主要参考了 GitHub 上 mediacloud 使用的句子分割技术[82]，并添加了一些修改。

这个句子分割方法的主要功能是将输入的文本字符串分割成单独的句子，并返回一个字符串列表。它通过一系列正则表达式和逻辑规则来识别句子的边界，处理复杂的标点符号和特殊情况。以下是该方法的详细解析：

1. **输入验证**：首先，函数检查输入的文本是否为 None 或空字符串。如果是 None，会发出一个警告（SentenceSplitterWarning），并返回一个空列表。如果是空字符串，则直接返回空列表。这些检查确保了函数能够优雅地处理无效输入。

2. **添加句子分隔符**：函数使用多个正则表达式（通过 regex.sub 函数）来识别句子的边界，并在适当的位置插入换行符（\n）作为分隔符：

表 3.2　Prompt 中英文对照表

英文	中文
I want you to act as a English teacher and improver.	我希望你充当一名英语教师和改进者。
I will give you an English article.	我将给你一篇英语文章。
This article is composed of several sentences that have already been segmented.	这篇文章由几个已经被分割的句子组成。
You should polish the following sentence one on one so that the new sentence has the same meaning as the original one.	你应该逐句润色以下句子，使新句子与原句具有相同的意思。
You should only output the generate sentence.	你只需输出生成的句子。
And the number of generated sentences should be equal to the number of original sentences.	生成的句子数量应与原句数量相等。
The format of the sentence and the article is shown as follows:	句子和文章的格式如下所示：
<sentence> The original input sentence 1. </sentence>	<sentence> 原始输入句子 1。</sentence>
<sentence> The original input sentence 2. </sentence>	<sentence> 原始输入句子 2。</sentence>
<sentence> ... </sentence>...	<sentence> ... </sentence>...
<sentence> The original input sentence n. </sentence>	<sentence> 原始输入句子 n。</sentence>
The format you generate should be:	你生成的格式应为：
<new sentence> The sentence you generate 1. </new sentence>	<new sentence> 你生成的句子 1。</new sentence>
<new sentence> The sentence you generate 2. </new sentence>	<new sentence> 你生成的句子 2。</new sentence>
<new sentence> ... </new sentence>	<new sentence> ... </new sentence>
<new sentence> The sentence you generate n. </new sentence>	<new sentence> 你生成的句子 n。</new sentence>
<sentence> Sentence here. </sentence>	<sentence> 句子在这里。</sentence>

表 3.3　正则表达式替换符号表

非 ASCII 字符	替换后字符	含义
\u00a0		空格
“	”	左双引号
”	”	右双引号
‘	’	左单引号
’	’	右单引号
\u2019	’	单引号
á	a	a 变体
ó	o	o 变体
–	-	短划线
—	-	短划线 2
…	...	一个字符表示三个句号
é	e	e 变体

- 非句号的句子结束标记（如? 或!）后面跟随句子开头的标志（如大写字母）。
- 多重句号（如...）后面跟随句子开头的标志。
- 标点符号后紧跟引号或括号的情况。
- 句子结束标点后紧跟句子开头标志的情况。
- 这些规则通过正则表达式的模式匹配来实现，确保了对复杂标点符号的处理。

3. **特殊标点处理**：在处理完主要的句子分隔符后，函数进一步处理特殊标点符号的情况。它将文本拆分为单词列表，并逐一检查每个单词是否符合特定的模式：

- 检查是否是已知的荣誉称谓（如 Dr. 或 Mr.），这些通常不会作为句子结束。
- 检查是否是大写字母缩写（如 U.S.A.），这些也不应被分割。
- 检查是否是数字相关的非分隔符情况。
- 通过这些检查，函数能够避免错误地将某些标点符号识别为句子边界。

4. **清理和返回结果**：在完成所有分隔符的插入后，函数会清理文本中的多余空格和换行符，确保输出的格式整洁。最后，它通过换行符（\n）将文本分割成句子列表，并返回结果。

总结：这个方法的设计非常全面，能够处理多种复杂的句子分隔情况，包括标点符号、引号、括号和缩写等。它适用于需要高精度文本分割的场景，如自然语言处理（NLP）任务或文本分析工具。

3.3.3 大型语言模型改写文本

在经过使用正则表达式替换非 ASCII 字符以及使用 Sentence Splitter 做句子分割后，我们将得到的句子与上表 3.2 提及的 Prompt 组合起来丢给大型语言模型进行重写润色。其中，大型语言模型共计五种，其中包括了：ChatGPT（GPT3.5）、GPT4o、Gemini、Qwen 和 DeepSeek。大型语言模型使用模型版本及价格如表 3.4 所示。均选用了较为先进的模型，且可知国内模型如通义千问和 DeepSeek 模型，其价格远低于国外相同类型的模型。

表 3.4　大型语言模型使用版本及定价（2025 年 4 月 8 日价格）

模型类别	模型版本	输入价格	输出价格
ChatGPT	gpt-3.5-turbo-0613	$1.5/1M token	$2/1M token
GPT4o	gpt-4o-2024-08-06	$2.5/1M token	$10/1M token
Gemini	gemini-exp-1206	$3.5/1M token	$14/1M token
Qwen	qwen-plus-2024-12-20	¥0.8/1M token	¥2/1M token
DeepSeek	deepseek-chat V3 (2024-12-24)	$0.27/1M token	$1.10/1M token

这五种大型语言模型在按照提示生成句子时有一定的失败概率，在第一次生成期间成功率如表 3.5 所示。由于网络波动等相关因素，生成的数据实例总数并不完全一致。Qwen、Gemini 和 DeepSeek V3 展示了遵循 Prompt 生成符合格式内容的强大能力。

3.3.3.1　ChatGPT

最早推出的 ChatGPT[5] 也有别称 GPT-3.5（Generative Pre-trained Transformer 3.5）是 OpenAI 在 GPT-3 基础上优化的大型语言模型（LLM），属于 Transformer 架构的生成式预训练模型家族。该模型在自然语言生成（NLG）、理解（NLU）及多任务处理方面表现卓越，广泛应用于学术写作、代码生成、智能客服等领域。

表 3.5　大语言模型首次生成时遵从 prompt 生成符合格式内容的成功率

模型名称	成功个数	生成总数	成功率
GPT3.5	7642	10745	71.12
GPT4o	8687	10745	80.85
Gemini	9695	10745	90.23
Qwen	9840	10745	**91.58**
Deepseek	9448	10330	87.93

ChatGPT 采用基于 Transformer 的神经网络架构，其核心特征在于自注意力机制（Self-Attention）的应用，该机制能够有效捕捉文本序列中的长距离依赖关系，并支持并行化计算以提升训练效率。研究表明，该模型的部分版本参数规模达到 1750 亿，通过构建深层网络结构并结合海量训练数据（包括互联网公开文本和学术文献等），显著提升了模型的性能表现。在训练方法上，ChatGPT 采用” 预训练 + 微调” 的混合范式：首先通过无监督学习从大规模语料中获取通用语言表征，随后结合有监督微调技术进行任务适配，并创新性地引入人类反馈强化学习[31]（Reinforcement Learning from Human Feedback, RLHF）方法，从而有效优化模型输出与人类价值观的对齐性。

ChatGPT（GPT-3.5）作为 OpenAI 推出的首个面向公众的大规模对话式 AI，在技术发展史上具有里程碑意义。其历史贡献主要体现在三个方面：首先，它通过免费交互形式将 1750 亿参数级大模型的能力普及化，两个月内用户破亿，成为史上增长最快的消费级应用；其次，通过指令微调和人类反馈强化学习（RLHF）的技术创新，显著提升了模型对齐人类意图的能力，奠定了后续对话模型的训练框架；最后，其在代码生成、教育辅助等领域的跨领域应用展示了通用 AI 的潜力，例如通过美国律师考试并催生了 AIGC 产业生态。然而，相较于后续模型（如 GPT-4、GPT-4o），其局限性逐渐显现：上下文窗口仅 3000 词（GPT-4 达 2.5 万词），复杂推理和数学计算准确率低于 GPT-4（如 LSAT 考试中 GPT-4 分数高 15%）；仅支持文本交互而缺乏多模态能力；知识时效性受限于 2021 年训练数据，且幻觉控制较弱，事实错误率较 GPT-4 高约 40%。这些不足推动了模型向更大参数规模、更优对齐性和更低推理成本的方向演进。

从技术迭代视角看，GPT-3.5 的突破性在于首次验证了大规模预训练模型结合人类反馈优化的可行性，但其单模态架构和有限推理能力在专业领域应用中逐渐被超

越。例如，OpenAI 推出的 GPT 新模型 GPT-4 通过引入多模态处理和万亿级参数，在医学诊断、法律分析等任务中展现出更高可靠性；而 GPT-4o 进一步整合音频、视频交互能力，实现了更自然的拟人化交互。

3.3.3.2 GPT4o

GPT-4o[83] 是 OpenAI 于 2024 年 5 月推出的新一代多模态大语言模型，其名称中的”o” 代表”omni”（全能），标志着其在文本、音频和视觉模态上的端到端处理能力实现了重大突破。该模型采用统一的 Transformer 架构，通过单一神经网络直接处理多模态输入（文本、图像、音频的任意组合）并生成文本、音频和图像输出的任何组合的相应输出，消除了传统多模态系统中不同模型间转换带来的延迟问题。它是跨文本、视觉和音频进行端到端训练的，这意味着所有输入和输出都由相同的神经网络处理。技术层面，GPT-4o 通过改进的词元化（tokenization）方法将音频特征和视频帧序列编码为与文本相同的表示形式，利用多头注意力机制实现跨模态语义关联，可以在低至 232 毫秒的时间内响应音频输入，平均为 320 毫秒，这与人类在对话中的响应时间相似。

相较于前代模型，GPT-4o 展现出三方面显著优势：首先，多模态能力实现质的飞跃，可完成图像内容分析生成食谱、通过呼吸声识别情绪状态等复杂任务；其次，非英语语言处理能力显著提升，支持 50 种语言的实时翻译，特别对非拉丁语系文字（如韩语、阿拉伯语）的识别准确率提高 20% 以上；最后，运行效率大幅优化，其 API 成本较 GPT-4 Turbo 降低 50%，推理速度提升 2 倍。

3.3.3.3 Gemini

Gemini[84] 是由 Google DeepMind 公司开发的多模态大语言模型系列，代表了谷歌在通用人工智能领域的重要突破。该系列模型于 2023 年 12 月正式发布，其核心创新在于采用原生多模态架构，能够同时处理文本、图像、音频、视频和代码五种信息类型。与传统的多模态模型不同，Gemini 从预训练阶段就采用统一神经网络处理多模态输入，通过改进的 tokenization 方法将不同模态数据编码为统一表示，利用多头注意力机制实现跨模态语义关联。这种设计使其在复杂推理任务中表现出色，例如能够理解手写物理题解答并指出错误推理步骤，或将视觉信息转化为编程代码。技术层面，Gemini 采用 TPUv4/v5e 加速器训练。其训练数据涵盖网络文档、学术文献和多语言语料，并通过人类反馈强化学习[31] （RLHF）优化输出对齐性。

尽管 Gemini 在多模态理解和复杂推理方面具有优势，但仍存在局限性。早期版本被质疑测试分数夸大和演示视频剪辑问题，且非英语任务性能相对较弱。与后续模型相比，缺乏动态更新机制，在专业领域（如医学、法律）的微调效果有待提升。这些不足推动了谷歌向更大参数规模、更低推理成本和更专业化方向迭代，例如开源模型 Gemma（20 亿/70 亿参数）的发布。总体而言，Gemini 系列通过原生多模态架构和规模化训练基础设施，为通用人工智能的发展提供了重要技术范式。

3.3.3.4 Qwen

通义千问[41]（Qwen）是阿里云自主研发的超大规模语言模型系列，作为中国人工智能领域的重要代表，其技术演进和产业应用体现了国产大模型的发展轨迹。该模型系列于 2023 年 4 月 11 日正式发布，其命名蕴含双重寓意：”通义”象征模型具备跨领域的普适性知识理解能力，”千问”则源自中国古代百科全书《千问》，彰显其应对复杂问题的潜力。技术架构上，通义千问采用改进的 Transformer 结构，通过旋转位置编码[85]（RoPE）增强长序列建模能力，并创新性地结合非绑定式嵌入（un-tied embeddings）与 RMSNorm 层优化，在 72B 参数版本中实现了 2048 个词元的上下文窗口支持。值得注意的是，其开源模型 Qwen1.5-110B 采用稀疏专家混合（MoE）架构，在 2024 年 5 月发布的 2.5 版本中，逻辑推理和代码能力较前代提升 16% 和 10%。

该模型的核心竞争力体现在三方面：多模态融合、产业适配性和开源生态建设。在模态支持上，通义千问 2.0 已实现文本、图像、音频的端到端处理，其多模态理解能力支持从商品设计草图生成营销文案等复杂任务。产业应用方面，阿里云通过”云智一体”战略将模型能力深度嵌入电商、医疗、金融等场景，例如通义灵码智能编程助手可完成 85% 的常规代码生成任务，显著提升开发效率。开源策略上，截至 2025 年，通义千问已发布从 1.8B 到 110B 参数的 7 款开源模型，累计下载量突破 700 万次，其中 Qwen-72B 在 MMLU 基准测试中达到与 Llama3-70B 相当的精度。

与同期国际模型相比，通义千问展现出鲜明的差异化特征。其训练数据包含超过 30% 的高质量中文语料，在古文解析、中文创意写作等任务中优于同等规模的 GPT-3.5 模型。商业落地方面，通过 API 价格策略（2 元/1M tokens）和轻量化部署方案（如联发科天玑 9300 芯片适配），大幅降低企业使用门槛。然而，该模型仍存在在非拉丁语系的多语言任务中准确率较 GPT-4 低约 15% 等问题。这些技术特性优势使通义千问成为中国大模型技术自主创新与全球化竞争的重要案例，其发展路径为学术界研究 AI 技术产业化提供了典型样本。

3.3.3.5 DeepSeek

DeepSeek V3[7] 是由中国深度求索公司（DeepSeek）于 2024 年 12 月推出的新一代多模态大语言模型，代表了国产大模型在技术突破与成本控制方面的重要里程碑。该模型采用混合专家（MoE）架构，总参数量达 6710 亿，其中每个词元激活 370 亿参数，通过动态路由机制实现计算资源的优化分配。技术层面，DeepSeek V3 创新性地结合了多头潜在注意力机制（MLA）和 FP8 混合精度训练，前者通过低秩压缩键值对减少内存占用，后者在矩阵乘法等计算密集型操作中使用 FP8 格式，将训练成本控制在 557.6 万美元，仅为 GPT-4o 等主流模型的 1/10。在 14.8 万亿 token 的预训练基础上，模型通过监督微调和强化学习进一步优化性能，其生成速度达到 60TPS（每秒事务处理量），较前代 V2.5 提升 3 倍，显著改善了交互流畅度。

性能表现上，DeepSeek V3 展现出多领域竞争优势。在知识类任务（MMLU、GPQA）中接近 Claude-3.5-Sonnet 水平，数学竞赛（AIME 2024、CNMO 2024）成绩超越所有开源闭源模型，代码生成能力可媲美 85% 的人类编程选手。中文处理尤为突出，其 30% 高质量中文语料的训练基础使其在古文解析、创意写作等任务中优于同等规模的国际模型。2025 年 3 月发布的 V3-0324 版本进一步强化了前端开发能力，能根据单一提示自动生成包含交互控件和赛博朋克风格界面的完整网站，被开发者评价为” 编码领域的标杆”。模型支持 128K 上下文窗口（开源版），在长文本理解任务（DROP、LongBench v2）中的平均表现领先行业。

商业化与开源策略构成 DeepSeek V3 的另一大特色。其 API 定价极具竞争力，输入与输出词元每百万词元成本分别为 0.27 美元和 1.10 美元，较国际同类产品降低 50% 以上。模型采用 MIT 开源协议，允许商业用途和二次开发，截至 2025 年已催生 700 万次下载量，推动 AI 技术在边缘设备的部署。应用场景覆盖智能编程（如通义灵码助手）、金融风控、教育辅导等领域，其中在电商平台的数据分析场景中，可实时识别欺诈交易并生成可视化报告。

尽管表现卓越，DeepSeek V3 仍存在局限，复杂音频/视觉输入的幻觉率较文本模态高约 20%。而 DeepSeek V3 模型现在已有的这些优势特性使其成为研究 AI 技术普惠化与算力效率优化的典型案例，为中国大模型的全球化竞争提供了重要参考。

3.4 模型改写文本检测数据集

经过上述大型语言模型的润色，原始的人类写作散文与其改进版本共同构成了模型改写文本检测数据集，总计包含 53,328 篇短文和 147,976 个句子。

新数据集中每篇短文的句子数、每篇散文的词元数以及每个句子的词元数如表 3.6 所示。第一行表示一篇短文中的句子个数，接下来分别是短文与句子的词元个数。50% 表示的是中位数，其他以此类推。大语言模型的改写倾向于将文本长度改短，使之更加精炼。

表 3.6　模型改写文本检测数据集数据集句子个数及词元个数

	模型	MIN	10%	25%	50%	75%	90%	MAX	AVG
	句子个数	2	8	11	16	21	26	53	16.65
短文词元	AES2	156	212	267	362	473	588	1338	385.41
	GPT3.5	84	208	259	349	454	558	1257	370.24
	GPT4o	122	199	247	331	428	525	1185	349.90
	Gemini	128	209	261	351	458	565	1268	372.92
	Qwen	111	197	245	325	419	510	1092	343.22
	Deepseek	115	200	249	333	427	517	1193	349.14
单句词元	AES2	4	11	15	20	28	38	305	23.15
	GPT3.5	2	11	15	20	27	35	229	22.24
	GPT4o	2	11	14	19	25	33	257	21.02
	Gemini	4	11	15	20	27	36	308	22.40
	Qwen	4	11	14	19	25	32	189	20.62
	Deepseek	4	11	14	19	25	33	243	20.97

3.5 本章小结

本章详细探讨了模型改写文本检测数据集（TOSWT 数据集）的构建方法，旨在为教育工作者提供一个有效的工具，以分析和追踪学生文本的来源。我们从教育学领域的背景出发，强调了构建该数据集的重要性，尤其是在当前人工智能生成文本日益普及的背景下，教师需要具备识别和分析学生写作中可能受到大型语言模型影响的能

力。

在数据集的构建过程中，我们以 Learning Agency Lab 发布的 Automated Essay Scoring 2.0 (AES2) 数据集为基础，首先对原始数据进行了系统的清洗和处理。通过使用正则表达式，我们成功替换了非 ASCII 字符，确保了数据的统一性和规范性。这一步骤为后续的分析提供了可靠的基础。

接下来，我们采用了句子分割技术，以实现对文本的细粒度处理。这一技术借助了先进的正则表达式和逻辑规则，能够有效识别句子的边界，处理复杂的标点符号和特殊情况。这种细分方法不仅提高了数据的可用性，还为后续的分析和模型训练奠定了基础。

在生成模型改写文本检测数据集的过程中，我们引入了多种大型语言模型，包括 ChatGPT、GPT4o、Gemini、Qwen 和 DeepSeek。这些模型在文本改写过程中展现出强大的能力，能够将原始文本进行润色和改写，同时保持其原意。通过对比不同模型的表现，我们能够评估其在文本生成任务中的有效性和可靠性。最终，构建出的数据集包含 53,328 篇短文和 147,976 个句子，为后续的文本检测研究提供了丰富的数据支持。

此外，本章还深入探讨了文本来源追踪的数学化表述，将其视为一个文本分类问题。我们详细描述了模型训练的过程，包括数据集的构建、特征提取和模型评估。通过构建一个分类模型，我们能够有效地将新文本映射到预定义的类别中，从而实现对文本来源的追踪和分析。这一方法论为后续研究提供了清晰的框架，确保了文本检测任务的系统性和可操作性。

综上所述，本章的工作不仅为学生写作的文本分析提供了重要的数据支持，也为教师在识别和评估学生写作能力方面提供了有力的工具。通过构建模型改写文本检测数据集，我们为教育领域在人工智能生成文本背景下的研究提供了新的视角和方法，推动了文本来源追踪技术的发展。

第 4 章 模型改写文本检测方法

4.1 引言

在第三章中，我们详细介绍了模型改写文本检测数据集（TOSWT 数据集）的构造方法，强调了数据清洗、句子分割以及利用大型语言模型进行文本改写的重要性。这一过程为后续的文本检测任务奠定了坚实的基础。在本章中，我们将深入探讨模型改写文本检测的方法，着重分析基于预训练模型的技术路线，并阐明实验设计及其结果。

随着人工智能技术的飞速发展，尤其是大型语言模型的广泛应用，文本生成的质量和复杂性显著提升。这使得传统的文本检测方法面临新的挑战，尤其是在识别和追踪人工智能生成文本方面。因此，开发有效的模型改写文本检测方法，成为了当前自然语言处理领域的重要研究方向。通过构建有效的检测模型，我们能够更好地识别出文本中可能的人工智能生成部分，从而为教育领域提供支持。

本章的核心目标是介绍两种基于预训练模型的文本检测方法，分别是 RoBERTa 和 DeBERTa 先进的语言模型。这些模型在自然语言理解和生成任务中表现出色，能够通过深层次的语义理解来识别文本的来源。我们将分析这些模型的架构和训练过程，并探讨它们在文本检测任务中的具体应用。

此外，我们将设计一系列实验，以评估所提出的方法在模型改写文本检测中的有效性。这些实验将包括数据集设置、参数配置、评价指标的选择以及对比实验和消融实验的结果分析。通过这些实验，我们不仅能够验证所提出方法的性能，还能为后续研究提供宝贵的经验和数据支持。

本章的结构安排如下：首先，我们将介绍基于预训练的模型改写文本检测方法，详细分析 RoBERTa 和 DeBERTa 模型的特点和应用。接着，我们将阐述实验设计，包括数据集的设置、实验参数的配置以及评价指标的选择。最后，我们将对实验结果进行深入分析，以评估所提出方法的有效性和实用性。

通过本章的研究，我们希望能够为模型改写文本检测领域提供新的思路和方法，推动相关技术的发展与应用。同时，我们也期待这些研究成果能够为教育工作者提供切实可行的解决方案，并启发内容审核人员以及其他相关领域的从业者的应用思路，以应对日益复杂的文本来源问题。

4.2 基于预训练的模型改写文本检测方法

随着自然语言处理（NLP）技术的不断进步，基于预训练模型的方法在文本分析和理解任务中取得了显著的成功。预训练模型通过在大量文本数据上进行训练，学习到丰富的语言特征和语义信息，这使得它们在特定任务中表现出色。在模型改写文本检测领域，利用这些预训练模型能够有效地识别和追踪文本的来源，尤其是在面对人工智能生成的文本时。

预训练模型的核心优势在于其强大的上下文理解能力。与传统的特征工程方法相比，预训练模型能够自动提取文本中的深层次特征，从而提高文本分类和检测的准确性。这一过程通常分为两个阶段：首先是在大规模文本数据上进行无监督预训练，然后在特定任务上进行微调。通过这种方式，模型能够适应不同的文本风格和结构，从而增强其在特定应用场景下的性能。

在模型改写文本检测任务中，预训练模型的使用能够帮助我们识别文本中的细微差异，例如句子结构的变化、用词的替换以及语义的调整。这对于判断文本是否经过人工智能模型的改写尤为重要。通过分析文本的上下文信息，预训练模型能够捕捉到潜在的模式和特征，这些模式可能是人类作者与人工智能生成文本之间的显著区别。

在本节中，我们将介绍两种流行且被本课题应用的预训练模型：RoBERTa[74] 和 DeBERTa[75,86]。这两种模型在文本理解和生成任务中都表现出色，尤其是在文本分类和检测方面，已被广泛应用于多个研究领域。

4.2.1 RoBERTa 模型

RoBERTa[74]（Robustly Optimized BERT Approach）在 BERT[12] 模型的基础上进行了多方面的改进，从而显著提升了模型的性能。

在训练策略方面，原始的 BERT 模型存在训练不足的问题。为了解决这一问题，RoBERTa 通过延长训练步数、使用更大的批次进行训练，并在更广泛的数据集上进行训练。在实验中，使用更大的批次训练 BERT 模型的结果表明，这种方法能够有效改善训练的困惑度（perplexity）和下游任务的准确率。此外，大批次训练更易于通过分布式数据并行训练实现，从而提高了训练效率。

在训练数据的选择上，RoBERTa 使用了包含多种语料库的总计 160GB 未压缩文本，这一数据量远超 BERT 原始训练所用的 16GB 数据。这一结果充分验证了数据量和多样性在预训练过程中所起的重要作用。同时，BERT 原始实现中，掩码是在数据

预处理阶段一次性生成的，形成了静态掩码。尽管通过复制数据来增加掩码的多样性，但这种方法仍存在局限性。相较之下，RoBERTa 采用了动态掩码机制，即每次向模型输入序列时都生成新的掩码模式。这一方法在长时间训练或处理大规模数据集时表现出明显的优势，不仅提升了训练效率，还在实验中显示出相较于静态掩码更优或相当的性能。

在模型输入格式的优化上，原始 BERT 模型在训练时使用了下一个句子预测（NSP）任务，旨在提升下游任务的性能。然而，当时的一些研究[87-89] 对这一任务提出了质疑。经过对比实验，RoBERTa 发现去除 NSP 损失并采用从单个文档中采样完整句子作为输入能够匹配或略微提升下游任务的性能。具体而言，限制序列来自单个文档的 DOC-SENTENCES 格式表现稍好于从多个文档打包的 FULL-SENTENCES 格式，但为了便于与相关研究的比较，最终 RoBERTa 采用了从多个文档打包的 FULL-SENTENCES 格式。此外，RoBERTa 不再使用 BERT 原有的两文档片段拼接并添加 NSP 损失的输入格式，而是采用了从一个或多个文档中连续采样完整句子的方式，使得输入总长度不超过 512 个标记。这一格式有助于模型学习长距离依赖关系，从而增强了文本理解的能力。

在文本编码的改进方面，BERT 原实现使用了一个大小为 30K 的字符级字节对编码[90] （BPE）词汇表，并需要进行启发式分词规则的预处理。RoBERTa 借鉴了其他研究，采用了一个包含 50K 子词单元的更大字节级 BPE 词汇表，这一方法无需额外的预处理或分词。尽管在部分任务上性能略有下降，但由于其通用编码方案的优势，RoBERTa 仍被广泛应用于后续的实验中。

通过上述基于 BERT 的改进，RoBERTa 在多个自然语言处理任务中展现出了卓越的性能，为后续研究提供了强有力的支持。

在本文中，采用 RoBERTa-base 和 RoBERTa-Large 两种大小的 RoBERTa 模型，其参数如表 4.1 所示。表中的参数包括 Transformer 层数、隐藏层维度、注意力头数以及总参数量。这些参数的设计反映了模型的复杂性及其学习能力。较大的模型（RoBERTa-Large）在处理更复杂的语言任务时，能够更好地捕捉深层次的语义关系和上下文信息，而较小的模型（RoBERTa-Base）则在计算效率和资源消耗上具有一定的优势。通过对比这两种模型的性能，我们能够深入探讨不同规模模型在文本理解和生成任务中的表现差异，从而为后续研究提供有价值的参考。

表 4.1　RoBERTa-Base 和 RoBERTa-Large 参数

	RoBERTa-Base	**RoBERTa-Large**
Transformer 层数	12	24
隐藏层维度	768	1024
注意力头数	12	16
总参数量	125M	355M

4.2.2 DeBERTa 模型

目前 DeBERTa 系列模型公开的论文仅有 DeBERTa 模型[75] 和 DeBERTa V3 模型[86]，我们的实验中将会使用 DeBERTa V3 模型，本节将会对 DeBERTa 模型对于 BERT[12] 和 RoBERTa 模型[74] 的改进进行介绍。

4.2.2.1 DeBERTa 模型

DeBERTa 模型[75] 是预训练语言模型不断演进过程中的佼佼者，相较于 RoBERTa 和 BERT 有诸多显著改进。这些改进主要体现在注意力机制、掩码解码器以及训练方法等方面，使其在模型性能和泛化能力上都有大幅提升。

在注意力机制方面，BERT 和 RoBERTa 在输入层将每个单词表示为其词嵌入（内容）与位置嵌入之和的向量，这种表示方式在一定程度上混合了内容和位置信息。而 DeBERTa 创新性地采用解耦注意力机制，每个单词由两个独立向量分别编码内容和位置。在计算单词间注意力权重时，DeBERTa 使用基于内容和相对位置的解耦矩阵分别计算，通过这种方式，能更精细地捕捉单词之间的关系。以句子“The dog chased the cat”为例，当分析“dog”和“cat”的关系时，不仅考虑它们的语义内容，还能依据相对位置更准确地判断其依赖关系，相比之下，BERT 和 RoBERTa 难以如此细致地处理这种关系。

对于序列中位置为 i 的一个词元（token），DeBERTa 用两个向量 $\mathbf{H}_i$ 和 $\mathbf{P}_{i|j}$ 来表示它，其中 $\mathbf{H}_i$ 代表该词元的内容，$\mathbf{P}_{i|j}$ 代表它与位置为 j 的词元的相对位置。词元 i 和 j 之间的交叉注意力分数的计算可以分解为四个部分，即：

$$\begin{aligned}\mathbf{A}_{i,j} &= \{\mathbf{H}_i, \mathbf{P}_{i|j}\} \times \{\mathbf{H}_j, \mathbf{P}_{j|i}\}^{\intercal} \\ &= \mathbf{H}_i\mathbf{H}_j^{\intercal} + \mathbf{H}_i\mathbf{P}_{j|i}^{\intercal} + \mathbf{P}_{i|j}\mathbf{H}_j^{\intercal} + \mathbf{P}_{i|j}\mathbf{P}_{j|i}^{\intercal}.\end{aligned} \tag{4.1}$$

也就是说，一对单词的注意力权重可以通过使用基于它们的内容和位置的解耦矩

阵，将四个注意力分数相加来计算，这四个分数分别为内容对内容、内容对位置、位置对内容以及位置对位置的注意力分数。

而在 DeBERTa 之前的相对位置编码方法在计算注意力权重时，使用单独的嵌入矩阵来计算相对位置偏差[91-92]。这相当于仅使用公式 4.1 中的内容对内容和内容对位置项来计算注意力权重。DeBERTa 模型认为位置对内容这一项也很重要，因为一对单词的注意力权重不仅取决于它们的内容，还取决于它们的相对位置，而只有同时使用内容对位置和位置对内容项才能对其进行全面建模。由于 DeBERTa 使用相对位置嵌入，位置对位置项不会提供太多额外信息，因此在我们的实现中从公式 4.1 中去掉了该项。

以单头注意力为例，标准的自注意力操作[1] 可以表示为：

$$\mathbf{Q} = \mathbf{H}\mathbf{W_q}, \mathbf{K} = \mathbf{H}\mathbf{W_k}, V = \mathbf{H}\mathbf{W_v}, \mathbf{A} = \frac{\mathbf{Q}\mathbf{K}^\intercal}{\sqrt{d}} \tag{4.2}$$

$$\mathbf{H_o} = \text{softmax}(\mathbf{A})\mathbf{V} \tag{4.3}$$

其中，$\mathbf{H} \in \mathbb{R}^{N\times d}$ 表示输入的隐藏向量，$\mathbf{H_o} \in \mathbb{R}^{N\times d}$ 是自注意力的输出，$\mathbf{W_q}$、$\mathbf{W_k}$、$\mathbf{W_v} \in \mathbb{R}^{d\times d}$ 是投影矩阵，$\mathbf{A} \in \mathbb{R}^{N\times N}$ 是注意力矩阵，N 是输入序列的长度，d 是隐藏状态的维度。

记 k 为最大相对距离，$\delta(i,j) \in [0, 2k)$ 为从词元 i 到词元 j 的相对距离，其定义为：

$$\delta(i,j) = \begin{cases} 0, & \text{当 } i-j \le -k \\ 2k-1, & \text{当 } i-j \ge k \\ i-j+k, & \text{其他情况} \end{cases} \tag{4.4}$$

DeBERTa 中将带有相对位置偏差的解耦自注意力表示为公式 4.5，其中 $\mathbf{Q}_c$、$\mathbf{K}_c$ 和 $\mathbf{V}_c$ 是分别使用投影矩阵 $\mathbf{W}_{q,c}$、$\mathbf{W}_{k,c}$、$\mathbf{W}_{v,c} \in \mathbb{R}^{d\times d}$ 生成的投影内容向量。$\mathbf{P} \in \mathbb{R}^{2k\times d}$ 表示在所有层中共享的相对位置嵌入向量（即在正向传播过程中保持固定），$\mathbf{Q}_r$ 和 $\mathbf{K}_r$ 是分别使用投影矩阵 $\mathbf{W}_{q,r}$、$\mathbf{W}_{k,r} \in \mathbb{R}^{d\times d}$ 生成的投影相对位置向量。

$$\begin{aligned} \mathbf{Q}_c = \mathbf{H}\mathbf{W}_{q,c}, \mathbf{K}_c = \mathbf{H}\mathbf{W}_{k,c}, \mathbf{V}_c = \mathbf{H}\mathbf{W}_{v,c}, \mathbf{Q}_r = \mathbf{P}\mathbf{W}_{q,r}, \mathbf{K}_r = \mathbf{P}\mathbf{W}_{k,r} \\ \tilde{\mathbf{A}}_{i,j} = \underbrace{\mathbf{Q}_i^c \mathbf{K}_j^{c\intercal}}_{\text{(a) content-to-content}} + \underbrace{\mathbf{Q}_i^c \mathbf{K}_{\delta(i,j)}^{r\intercal}}_{\text{(b) content-to-position}} + \underbrace{\mathbf{K}_j^c \mathbf{Q}_{\delta(j,i)}^{r\intercal}}_{\text{(c) position-to-content}} \\ \mathbf{H_o} = \text{softmax}(\frac{\tilde{\mathbf{A}}}{\sqrt{3d}})\mathbf{V}_c \end{aligned} \tag{4.5}$$

$\tilde{\mathbf{A}}_{i,j}$ 是注意力矩阵 $\tilde{\mathbf{A}}$ 的元素，表示从词元 i 到词元 j 的注意力分数。$\mathbf{Q}_i^c$ 是 $\mathbf{Q}_c$ 的第 i 行，$\mathbf{K}_j^c$ 是 $\mathbf{K}_c$ 的第 j 行，$\mathbf{K}_{\delta(i,j)}^r$ 是 $\mathbf{K}_r$ 中与相对距离 $\delta(i,j)$ 对应的第 $\delta(i,j)$ 行，$\mathbf{Q}_{\delta(j,i)}^r$ 是 $\mathbf{Q}_r$ 中与相对距离 $\delta(j,i)$ 对应的第 $\delta(j,i)$ 行。注意这里我们使用 $\delta(j,i)$ 而不是 $\delta(i,j)$，这是因为对于给定的位置 i，位置对内容计算的是位置 j 处的键内容相对于位置 i 处的查询位置的注意力权重，因此相对距离是 $\delta(j,i)$。位置对内容项的计算为 $\mathbf{K}_j^c\mathbf{Q}_{\delta(j,i)}^{r\intercal}$，内容对位置项的计算方式类似。

最后，我们对 $\tilde{\mathbf{A}}$ 应用一个缩放因子 $\frac{1}{\sqrt{3d}}$。这个因子对于稳定模型训练很重要[1]，特别是对于大规模预训练语言模型。

算法 1 解耦注意力（Disentangled Attention）

Require: 隐藏状态 $\mathbf{H}$, 相对距离嵌入 $\mathbf{P}$, 相对距离矩阵 δ. 内容投影矩阵 $\mathbf{W}_{k,c}$, $\mathbf{W}_{q,c}$, $\mathbf{W}_{v,c}$, 位置投影矩阵 $\mathbf{W}_{k,r}$, $\mathbf{W}_{q,r}$.
Ensure: $\mathbf{H}_o$

$\mathbf{K}_c = \mathbf{H}\mathbf{W}_{k,c}$, $\mathbf{Q}_c = \mathbf{H}\mathbf{W}_{q,c}$, $\mathbf{V}_c = \mathbf{H}\mathbf{W}_{v,c}$, $\mathbf{K}_r = \mathbf{P}\mathbf{W}_{k,r}$, $\mathbf{Q}_r = \mathbf{P}\mathbf{W}_{q,r}$
$\mathbf{A}_{c\to c} = \mathbf{Q}_c\mathbf{K}_c^\intercal$
for $i = 0, ..., N-1$ **do**
    $\tilde{\mathbf{A}}_{c\to p}[i,:] = \mathbf{Q}_c[i,:]\mathbf{K}_r^\intercal$
end for
for $i = 0, ..., N-1$ **do**
    for $j = 0, ..., N-1$ **do**
        $\mathbf{A}_{c\to p}[i,j] = \tilde{\mathbf{A}}_{c\to p}[i, \delta[i,j]]$
    end for
end for
for $j = 0, ..., N-1$ **do**
    $\tilde{\mathbf{A}}_{p\to c}[:,j] = \mathbf{K}_c[j,:]\mathbf{Q}_r^\intercal$
end for
for $j = 0, ..., N-1$ **do**
    for $i = 0, ..., N-1$ **do**
        $\mathbf{A}_{p\to c}[i,j] = \tilde{\mathbf{A}}_{p\to c}[\delta[j,i], j]$
    end for
end for
$\tilde{\mathbf{A}} = \mathbf{A}_{c\to c} + \mathbf{A}_{c\to p} + \mathbf{A}_{p\to c}$
$\mathbf{H}_o = \text{softmax}(\frac{\tilde{\mathbf{A}}}{\sqrt{3d}})\mathbf{V}_c$

DeBERTa 中采取了一种高效实现。在 DeBERTa 之前的研究中对于长度为 N 的输入序列，存储每个词元的相对位置嵌入需要 $O(N^2d)$ 的空间复杂度[91-93]。然而，以内容对位置为例，DeBERTa 中注意到由于 $\delta(i,j) \in [0, 2k)$，并且所有可能相对位置的嵌入始终是 $\mathbf{K}_r \in \mathbb{R}^{2k\times d}$ 的一个子集，因此我们可以在所有查询的注意力计算中重用 $\mathbf{K}_r$。

在 DeBERTa 中，预训练时将最大相对距离 k 设置为 512。解耦注意力权重可以

使用算法 1 高效计算。令 δ 为根据公式 4.4 得到的相对位置矩阵，即 $\delta[i,j] = \delta(i,j)$。我们不是为每个查询分配不同的相对位置嵌入矩阵，而是将每个查询向量 $\mathbf{Q}_c[i,:]$ 乘以 $\mathbf{K}_r^{\intercal} \in R^{d\times 2k}$（如算法 1 的第 3-5 行），然后使用相对位置矩阵 δ 作为索引提取注意力权重（如算法 1 的第 6-10 行）。为了计算位置对内容的注意力分数，我们通过将每个键向量 $\mathbf{K}_c[j,:]$ 乘以 $\mathbf{Q}_r^T$ 来计算 $\tilde{\mathbf{A}}_{p\to c}[:,j]$，即注意力矩阵 $\tilde{\mathbf{A}}_{p\to c}$ 的列向量（如算法 4.4 的第 11-13 行），最后通过相对位置矩阵 δ 作为索引提取相应的注意力分数（如算法 1 的第 14 - 18 行）。通过这种方式，我们不需要为每个查询分配内存来存储相对位置嵌入，从而将空间复杂度降低到 $O(kd)$（用于存储 $\mathbf{K}_r$ 和 $\mathbf{Q}_r$）。

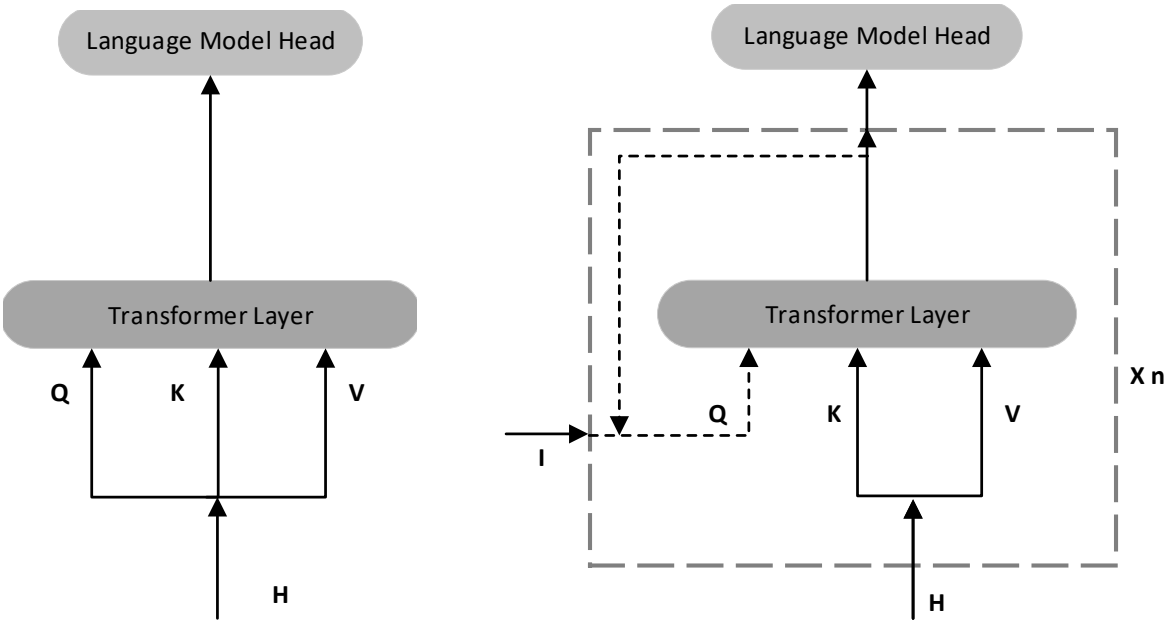
除此之外，DeBERTa 还采用掩码语言建模（Masked Language Modeling，MLM）进行预训练，在该训练过程中，模型学习利用掩码词周围的词来预测被掩码的词。DeBERTa 在 MLM 任务中会利用上下文词的内容和位置信息。解耦注意力机制虽然已经考虑了上下文词的内容和相对位置，但未考虑这些词的绝对位置，而在许多情况下，绝对位置对于预测极为关键。

以句子“a new store opened beside the new mall”为例，若将其中的“store”和“mall”进行掩码并用于预测任务。仅依靠局部上下文（如相对位置和周边词汇），模型难以区分该句中的“store”和“mall”，因为它们相对于“new”的相对位置是相同的。为解决这一局限，模型需要将绝对位置作为相对位置的补充信息纳入考量。例如，该句的主语是“store”而非“mall”，这些句法上的细微差别在很大程度上依赖于单词在句中的绝对位置，所以在语言建模过程中考虑单词的绝对位置至关重要。

将绝对位置信息融入模型有两种方式。BERT 模型是在输入层融入绝对位置信息。而在 DeBERTa 结构中，在所有 Transformer 层之后、但在基于聚合的上下文词内容和位置嵌入来解码掩码词的 Softmax 层之前融入绝对位置信息，如图 4.1 所示。通过这种方式，DeBERTa 在所有 Transformer 层中捕获相对位置信息，并且仅在解码掩码词时将绝对位置信息作为补充信息使用。因此，DeBERTa 将解码组件称为增强掩码解码器（Enhanced Mask Decoder，EMD）。

4.2.2.2 DeBERTa V3 模型

由于 ELECTRA[94] 中的替换词元检测（Replaced Token Detection, RTD）和 DeBERTa 中的解耦注意力机制已被证明在预训练中样本效率较高，DeBERTa V3 在这两种方法的基础上应运而生。它将原始 DeBERTa 中使用的掩码语言模型（Masked Language Modeling, MLM）目标替换为替换词元检测目标，以结合后者的优势。



(a) BERT decoding layer　　(b) Enhanced Mask Decoder

图 4.1　BERT 与 DeBERTa 编码层的比较

基于 Transformer 的预训练语言模型通常在大量文本上进行预训练，通过一种自监督目标来学习上下文单词表示，这种目标被称为掩码语言建模[12]（MLM）。具体来说，给定一个序列 $X = \{x_i\}$，我们通过随机掩码其 15% 的词元将其损坏为 $\tilde{X}$，然后训练一个由 θ 参数化的语言模型，基于 $\tilde{X}$ 预测被掩码的词元 $\tilde{x}$ 来重建 X：

$$\max_{\theta} \log p_{\theta}(X|\tilde{X}) = \max_{\theta} \sum_{i \in \mathcal{C}} \log p_{\theta}(\tilde{x}_i = x_i|\tilde{X}) \tag{4.6}$$

其中，$\mathcal{C}$ 是序列中被掩码词元的索引集。BERT 的作者提议让 10% 的被掩码词元保持不变，另外 10% 用随机选择的词元替换，其余的用 [MASK] 词元替换。

与 BERT 仅使用一个 Transformer 编码器并通过掩码语言模型进行训练不同，ELECTRA 采用生成对抗网络（GAN）的方式，使用两个 Transformer 编码器进行训练。一个是通过掩码语言模型训练的生成器，另一个是通过词元级二元分类器训练的判别器。生成器用于生成模糊的词元来替换输入序列中被掩码的词元。然后，修改后的输入序列被输入到判别器中。判别器中的二元分类器需要判断相应的词元是原始词元还是由生成器替换的词元。我们分别用 θ_G 和 θ_D 表示生成器和判别器的参数。判别器中的训练目标被称为替换词元检测（RTD）。生成器的损失函数可以写为：

$$L_{\text{MLM}} = \mathbb{E}\left(-\sum_{i \in \mathcal{C}} \log p_{\theta_G}(\tilde{x}_{i,G} = x_i|\tilde{\mathbf{X}}_G) \right) \tag{4.7}$$

其中，$\tilde{\mathbf{X}}_G$ 是通过在 $\mathbf{X}$ 中随机掩码 15% 的词元得到的生成器的输入。

判别器的输入序列是通过根据生成器的输出概率采样新的词元来替换被掩码的词元构建的：

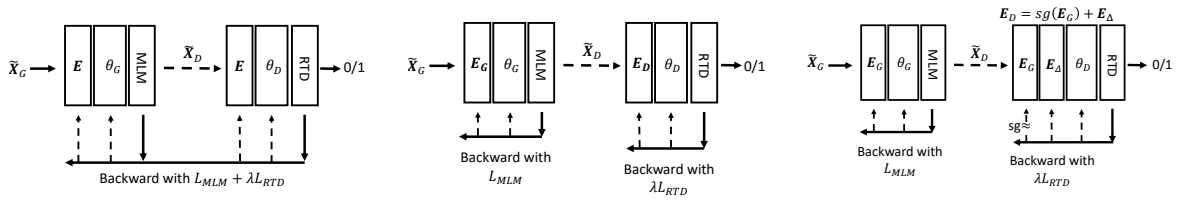
$$\tilde{x}_{i,D} = \begin{cases} \tilde{x}_i \sim p_{\theta_G}(\tilde{x}_{i,G} = x_i | \tilde{\mathbf{X}}_G), & i \in \mathcal{C} \\ x_i, & i \notin \mathcal{C} \end{cases} \tag{4.8}$$

判别器的损失函数可表示为：

$$L_{\text{RTD}} = \mathbb{E}\left(-\sum_i \log p_{\theta_D}\left(\mathbb{1}(\tilde{x}_{i,D} = x_i) | \tilde{X}_D, i \right) \right) \tag{4.9}$$

其中，$\mathbb{1}(\cdot)$ 是指示函数，$\tilde{X}_D$ 是通过公式 3 构建的判别器的输入。在 ELECTRA 中，L_{MLM} 和 L_{RTD} 是联合优化的，$L = L_{\text{MLM}} + \lambda L_{\text{RTD}}$，其中 λ 是判别器损失 L_{RTD} 的权重，在 ELECTRA 中设置为 50。

除此之外，DeBERTa V3 通过用一种新的梯度解耦嵌入共享（Gradient-Disentangled Embedding Sharing, GDES）方法取代 ELECTRA 最初提出的用于替换词元检测的词嵌入共享（Embedding Sharin, ES）方法，使得 DeBERTa V3 的性能得以进一步提升。



(a) 嵌入共享　(b) 无嵌入共享　(c) 梯度解耦嵌入共享

图 4.2　不同嵌入共享方法示意图

DeBERTa V3 提出梯度解耦嵌入共享（Gradient-Disentangled Embedding Sharing, GDES）方法，以克服嵌入共享（Embedding Sharin, ES）和无嵌入共享（No Embedding Sharin, NES）的缺点，同时保留它们的优点。图 4.2 中展示了不同嵌入共享方法示意图。嵌入共享（Embedding Sharin, ES）中，$\mathbf{E}$、θ_G 和 θ_D 将在一次反向传播中依据 $L_{\text{MLM}} + \lambda L_{\text{RTD}}$ 共同更新。无嵌入共享（No Embedding Sharin, NES）中，$\mathbf{E}_G$ 和 θ_G 将首先依据 L_{MLM} 通过反向传播进行更新，然后 $\mathbf{E}_D$ 和 θ_D 将依据 λL_{RTD} 通过反向传播进行更新。而在梯度解耦嵌入共享（Gradient-Disentangled Embedding Sharing, GDES）中，$\mathbf{E}_G$ 和 θ_G 将首先依据 L_{MLM} 在反向传播中进行更新，然后 $\mathbf{E}_\Delta$ 和 θ_D 将依据 λL_{RTD} 和 $\mathbf{E}_G$ 通过反向传播进行更新。sg 是停止梯度操作符，用于防止判别器更新 $\mathbf{E}_G$。

如图 4.2（c）所示，梯度解耦嵌入共享方法在生成器和判别器之间共享词嵌入，

这使得两个模型能够从相同的词汇表中学习，并利用嵌入中编码的丰富语义信息。然而，与嵌入共享方法不同，梯度解耦嵌入共享方法不允许替换词元检测损失影响生成器的梯度，从而避免了由冲突目标导致的干扰和效率低下问题。相反，梯度解耦嵌入共享方法仅使用掩码语言模型损失来更新生成器的嵌入，这确保了生成器输出的一致性和连贯性。结果，梯度解耦嵌入共享方法可以达到与无嵌入共享相同的收敛速度，同时又不牺牲嵌入的质量。

为了实现梯度解耦嵌入共享，DeBERTa V3 将判别器的嵌入重新参数化为 $\mathbf{E}_D = sg(\mathbf{E}_G) + \mathbf{E}_\Delta$，其中停止梯度算子 sg 可防止梯度流经生成器的嵌入 $\mathbf{E}_G$，仅更新残差嵌入 $\mathbf{E}_\Delta$。DeBERTa V3 将 $\mathbf{E}_\Delta$ 初始化为零矩阵，并按照无嵌入共享的训练过程进行模型训练。在每次迭代中，首先使用生成器为判别器生成输入，并利用掩码语言模型损失更新 $\mathbf{E}_G$ 和 $\mathbf{E}_D$。然后，在生成的输入上运行判别器，并利用梯度解耦嵌入共享损失更新 $\mathbf{E}_D$，但仅通过 $\mathbf{E}_\Delta$ 进行更新。训练完成后，将 $\mathbf{E}_\Delta$ 加到 $\mathbf{E}_G$ 上，并将得到的矩阵保存为判别器的 $\mathbf{E}_D$。

DeBERTa V3 进行了广泛的实验，以评估梯度解耦嵌入共享与嵌入共享和无嵌入共享相比的有效性。其结果证实，梯度解耦嵌入共享是一种用于使用掩码语言模型和替换词元检测进行预训练的语言模型的有效权重共享方法。也因此，DeBERTa V3 模型成为了目前 Kaggle 刷榜以及深度学习效果相当强大的模型。我们在本实验中使用的 DeBERTa 系列模型包括 DeBERTa-V3-Base 和 DeBERTa-V3-Large，表 4.2 中展示了其模型参数。

表 4.2　DeBERTa-V3-Base 和 DeBERTa-V3-Large 模型参数

	DeBERTa-V3-Base	**DeBERTa-V3-Large**
Transformer 层数	12	24
隐藏层维度	768	1024
词汇表大小	128K	128K
主干网络参数量	86M	304M
嵌入层参数量	98M	131M

4.3 实验设计

我们设计了如下实验，基于在第 3 章中介绍的模型改写文本检测数据集微调 4.2.1 节中提及的 RoBERTa-Base 和 RoBERTa-Large 以及在 4.2.2 节中提及的 DeBERTa-V3-Base 和 DeBERTa-V3-Large 模型。微调后再调用模型在测试集上输出预测标签，最后使用四种指标准确率（Accuracy）、精确率（Precision）、召回率（Recall）和 F1 分数评估预测标签的水准，进而评定微调后的模型能力。

为满足细粒度的数据要求，分别使用文档级别和句子级别的数据分别微调上文提及的四个模型。这些预训练模型参数来源 URL 如表 4.3 所示。

表 4.3　预训练模型参数来源 URL

模型	URL
RoBERTa-Base	https://huggingface.co/FacebookAI/roberta-base
RoBERTa-Large	https://huggingface.co/FacebookAI/roberta-large
DeBERTa-V3-Base	https://huggingface.co/microsoft/deberta-v3-base
DeBERTa-V3-Large	https://huggingface.co/microsoft/deberta-v3-large

除了该对比实验外，我们还设计了消融实验，在 AES2 数据集评分基础上，进一步探索大语言模型改写文本的能力。有这样两个问题：原始文本对于大语言模型改写文本的影响有多大？大语言模型改写文本后，新文本的文本质量与原始文本质量一致性如何？为深入探索这两个问题，我们同样设计了两个额外的消融实验放置于 4.3.5 中。

4.3.1 数据集设置

本实验中数据集采用在第 3 章中介绍的模型改写文本检测数据集。数据集示例如表 4.4 所示，受单页篇幅所限，这里选取了一个包含两个句子的数据示例，该示例数据评分在其原始 AES2 数据集中被评定为 2 分。

对于文档级数据，我们将模型改写文本检测数据集分割为训练集、验证集与测试集。这三个被分割后的小数据集在大数据集随机打乱后分别按照文档的个数以 7 : 1 : 2 的比例分割出来。待微调的模型在训练集上微调过后，该模型在验证集上得到初步的模型效果，随后选取损失函数最低的模型在测试集上测试，进而得到最终的效果。

而句子级数据中关于数据集的分割步骤也与文档级数据的分割步骤类似。只不过

表 4.4　模型改写文本检测数据集数据示例

模型	文本
AES2	Luke think you should join the seagoing cowboys program because you can travel all around and see some pretty nice things like Luke it said in the passage that you take nine trips to things like Greece,China,you will past the Atlantic Ocean and also you can make the world a better place and you can be a hero for being nice. Also so if you want to you can help out animals in the passage it said, the animals had to drink two or three times a day also a way you can help you could clean the stalls on the ship and they will give you jobs and other stuff and so you won't think it is boring you can have fun onboard to they got a lot of things like baseball,and volleyball and the house it is table-tennis tournaments,fencing,boxing,reading,whittling,and the games make time go fast and that's why I think you should join the seagoing Cowboys program.
GPT3.5	Luke thinks you should join the Seagoing Cowboys program because you can travel all around and see some pretty nice things. Like Luke said in the passage, you can take nine trips to places like Greece and China. You will cross the Atlantic Ocean, and you can make the world a better place by being nice and becoming a hero. Also, if you want to, you can help out animals. In the passage, it said that the animals had to drink two or three times a day. Another way you can help is by cleaning the stalls on the ship. They will give you jobs and other stuff. And so you won't think it's boring, you can have fun onboard too. They have a lot of things like baseball, volleyball, table-tennis tournaments, fencing, boxing, reading, and whittling. The games make time go fast, and that's why I think you should join the Seagoing Cowboys program.
GPT4o	Luke believes you should join the seagoing cowboys program because it allows you to travel extensively and see beautiful places; as mentioned in the passage, you can undertake trips to destinations like Greece and China, cross the Atlantic Ocean, and make the world a better place while becoming a hero by being kind. Additionally, if you wish, you can assist with caring for animals; the passage noted that the animals need to drink two or three times a day, and one of the ways you can contribute is by cleaning the stalls on the ship, where they will provide you with various tasks to keep you engaged, along with fun activities such as baseball and volleyball, as well as other pastimes like table tennis tournaments, fencing, boxing, reading, whittling, which help make the time fly by-these are the reasons I think you should join the seagoing Cowboys program.
Gemini	Luke believes you should join the Seagoing Cowboys program because it offers the opportunity to travel extensively and witness beautiful sights, as he mentioned in the passage that participants take nine trips to destinations like Greece and China, traverse the Atlantic Ocean, and contribute to making the world a better place, potentially becoming a hero through acts of kindness. Additionally, if you're interested, you can assist animals, as the passage mentioned that they need to drink two or three times daily; you could also help by cleaning stalls on the ship, and they'll provide you with various tasks and activities to keep you engaged and prevent boredom, as there are plenty of recreational options onboard, including baseball, volleyball, table tennis tournaments, fencing, boxing, reading, whittling, all of which help pass the time quickly, and that's why I believe you should consider joining the Seagoing Cowboys program.
Qwen	Luke believes you should join the seagoing cowboys program because it offers opportunities to travel extensively and see beautiful places, such as Greece and China, as mentioned in the passage. You will cross the Atlantic Ocean and contribute to making the world a better place, thereby becoming a hero by being kind and helpful. Additionally, if you wish, you can assist animals, as the passage notes they need to drink two or three times a day. You could help by cleaning the ship's stalls and taking on various jobs, ensuring the journey is enjoyable. Onboard activities like baseball, volleyball, table tennis tournaments, fencing, boxing, reading, and whittling make time pass quickly, which is why I think you should join the seagoing cowboys program.
DeepSeek	Luke believes you should join the seagoing cowboys program because it offers the opportunity to travel extensively and witness beautiful sights, such as Greece and China, while crossing the Atlantic Ocean, and it allows you to make the world a better place and become a hero through kindness. Additionally, if you wish, you can assist animals, as mentioned in the passage, which require drinking two or three times daily, and you can contribute by cleaning the stalls on the ship, where they provide jobs and other activities to ensure it's not boring; you can also enjoy onboard entertainment like baseball, volleyball, table-tennis tournaments, fencing, boxing, reading, whittling, and games that make time pass quickly, which is why I recommend joining the seagoing cowboys program.

第一步需要先将所有句子分割出来，单独打上标签，按照句子的个数随机打乱顺序后再以 7 : 1 : 2 的比例分割为训练集、验证集和测试集。

对于词元个数过长的数据用例，我们将长度大于 512 后面的部分完全截断。不过，这一步是在词元化（tokenizer）的过程中处理的。

4.3.2 实验参数设置

所有实验均运行于 Ubuntu Linux 22.04 服务器上。详细实验操作系统环境如表 4.5 所示。

表 4.5　实验操作系统环境

字段	值
内核版本	6.8.0-49-generic
构建环境	buildd@lcy02-amd64-103
架构	x86_64 (64 位)
编译器	x86_64-linux-gnu-gcc-12
编译器版本	12.3.0
编译器来源	Ubuntu 12.3.0-1ubuntu1~22.04
链接器 (GNU ld)	GNU Binutils for Ubuntu 2.38
构建编号	#49~22.04.1-Ubuntu
构建类型	SMP (对称多处理) + PREEMPT_DYNAMIC（动态抢占模式）
构建时间	Wed Nov 6 17:42:15 UTC 2024
发行版基础	Ubuntu 22.04.1 LTS

本章节全部实验均在拥有 24GB 显存容量的 NVIDIA TITAN RTX 显卡上完成，该显卡详细环境如表 4.6 所示。

本章节实验在实现上基于 Anaconda 库中搭建的 Conda 虚拟环境运行 Python 代码，Python 环境中部分重要包版本号如表 4.7 所示。

在基础的文档级和句子级的数据实验中，我们使用 DeBERTa 和 RoBERTa 作为实验用的模型并微调了四种模型：RoBERTa-Base、RoBERTa-Large、DeBERTa-Base 和 DeBERTa-Large，可变超参数包括批大小（batch size, BS）、学习率（learning rate, LR）、权重衰减系数（weight decay, WD）。在所有实验中，学习率均采用余弦退火衰减算法，并使用 AdamW 算法作为优化器。超参数详见表 4.8，其中，在句子级实验中，大尺

表 4.6　实验显卡环境

字段	值
GPU 型号	NVIDIA TITAN RTX
GPU 架构	Turning
显存容量	24GB
TDP (功耗)	320W
驱动版本	NVIDIA-SMI 560.35.03
CUDA 版本	12.6

寸模型如 RoBERTa-Large 和 DeBERTa-V3-Large 的超参数均经过广泛地搜索长时间实验才找到可以使模型拟合于数据的一组超参数。DeBERTa-V3-Large 由于其总参数量较大，为防止超出显卡显存，因此批大小相较其他实验的 16 个为一批较小，设置为 4 个为一批。

4.3.2.1　交叉熵损失函数

交叉熵函数[95] 是信息论和机器学习中的核心概念，主要用于衡量两个概率分布之间的差异。其数学定义为：对于真实分布 P 和预测分布 Q，交叉熵 $H(P,Q) = -\sum_x P(x)\log Q(x)$。它表示用 Q 编码来自 P 的事件所需的平均比特数，当 Q 越接近 P 时，交叉熵值越小。

在机器学习中，交叉熵常作为分类任务的损失函数。例如，在本文中，对于 m 个样本的 6 分类问题，损失函数可表示为

$$L = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{6} t_{ij}\log(y_{ij}) \tag{4.10}$$

其中 t_{ij} 是真实标签（one-hot 编码），y_{ij} 是模型预测概率。这种设计能有效惩罚预测错误，尤其配合 Softmax 或 Sigmoid 激活函数时，梯度计算高效且避免均方误差的学习速率下降问题。

交叉熵与信息熵、KL 散度密切相关。信息熵 $H(P)$ 是 P 自身的最佳编码长度，而交叉熵 $H(P,Q)$ 是使用 Q 编码 P 的代价，两者差值即为 KL 散度，反映分布差异。实际应用中需注意数值稳定性，如对 $Q(x)$ 裁剪防止 $\log(0)$ 错误。

将交叉熵函数作为损失函数，其优势在于理论严谨性与实践高效性的结合。例如，

表 4.7　实验 Python 环境部分重要包版本

分类	字段	版本号
基础	conda	24.9.2
	Python	3.12.8
	GCC	11.2.0
	cuda-cudart	11.8.89
	cuda-cupti	11.8.87
	cuda-libraries	11.8.0
	cuda-nvrtc	11.8.89
	cuda-nvtx	11.8.86
	cuda-runtime	11.8.0
	cuda-version	12.6
	ipython	8.30.0
	json5	0.9.25
	matplotlib	3.9.2
	matplotlib-base	3.9.2
	matplotlib-inline	0.1.6
	nltk	3.9.1
	numpy	2.0.1
	openai	1.52.1
	pandas	2.2.3
	pillow	11.0.0
	pip	24.2
	regex	2024.11.6
	sentencepiece	0.2.0
	scikit-learn	1.6.0
	scipy	1.14.1
	tensorboard	2.17.0
	xlsxwriter	3.1.1
	yaml	0.2.5
Torch	pytorch	2.5.1
	pytorch-cuda	11.8
	pytorch-mutex	1.0
	torchaudio	2.5.1
	torchtriton	3.1.0
	torchvision	0.20.1
HuggingFace	accelerate	1.2.1
	datasets	3.2.0
	huggingface-hub	0.27.0
	tokenizers	0.21.0
	transformers	4.47.1

表 4.8　对比试验微调超参数列表

	模型	批大小	初始学习率	权重衰减系数
文档级	RoBERTa-Base	16	10^{-5}	0.01
	RoBERTa-Large	16	10^{-5}	0.01
	DeBERTa-V3-Base	16	10^{-5}	0.01
	DeBERTa-V3-Large	4	10^{-5}	0.01
句子级	RoBERTa-Base	16	10^{-5}	0.01
	RoBERTa-Large	16	10^{-6}	0.0001
	DeBERTa-V3-Base	16	10^{-5}	0.01
	DeBERTa-V3-Large	4	10^{-6}	0.001

图像分类任务中，交叉熵直接优化预测概率与真实标签的匹配度；在本文所处的领域自然语言处理中，它评估生成文本的概率分布质量。

4.3.2.2　AdamW 优化器

AdamW 优化器[96] 是 Adam 优化器[97] 的一个改进版本，算法流程如算法 2 所示。Ilya Loshchilov 和 Frank Hutter 在 2017 年提出该算法，主要解决了传统 Adam 优化器中权重衰减（Weight Decay）处理方式的问题。在原始的 Adam 优化器中，权重衰减是通过将衰减项直接添加到梯度中来实现的，这种方式会导致权重衰减的效果受到自适应学习率的干扰，从而影响正则化的有效性。而 AdamW 通过将权重衰减从梯度更新过程中解耦，使其直接作用于参数更新步骤，从而更准确地实现正则化效果。

AdamW 的核心思想是保留 Adam 优化器的自适应学习率和动量机制，同时改进权重衰减的处理方式。具体来说，AdamW 在参数更新时，将权重衰减项独立于梯度计算之外，直接对参数进行衰减。这种方法使得权重衰减的效果更加稳定，能够有效防止模型过拟合，并提升泛化能力。实验表明，AdamW 在多个深度学习任务中表现优于传统的 Adam，尤其是在需要强正则化的场景下。

AdamW 的优点包括：1) 更有效的正则化，能够更好地控制模型复杂度；2) 在训练深度神经网络时表现出更稳定的收敛性；3) 适用于需要长时间训练的任务，如大语言模型（LLM）的预训练。然而，AdamW 也有一些缺点，例如对学习率和权重衰减系数的选择较为敏感，需要更多的调参工作，这也导致了句子级尺寸较大的模型调参

算法 2 AdamW 优化算法

Require: 学习率 ψ(lr), β_1, β_2 (betas), 初始参数 θ_0 (params), 目标函数 $f(\theta)$ (objective), ϵ, 权重衰减 χ, amsgrad, maximize

Ensure: θ_t

初始化: $m_0 \leftarrow 0$ (一阶矩), $v_0 \leftarrow 0$ (二阶矩), $\widehat{v_0}^{max} \leftarrow 0$
for $t = 1$ to ... **do**
  if maximize **then**
    $g_t \leftarrow -\nabla_\theta f_t(\theta_{t-1})$
  else
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$
  end if
  $\theta_t \leftarrow \theta_{t-1} - \psi\chi\theta_{t-1}$ ▷ 权重衰减
  $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$ ▷ 更新一阶矩
  $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$ ▷ 更新二阶矩
  $\widehat{m_t} \leftarrow m_t / (1-\beta_1^t)$ ▷ 偏差修正
  $\widehat{v_t} \leftarrow v_t / (1-\beta_2^t)$
  if amsgrad **then**
    $\widehat{v_t}^{max} \leftarrow \max(\widehat{v_{t-1}}^{\max}, \widehat{v_t})$
    $\theta_t \leftarrow \theta_t - \psi \widehat{m}_t / (\sqrt{\widehat{v_t}^{\max}} + \epsilon)$
  else
    $\theta_t \leftarrow \theta_t - \psi \widehat{m_t} / (\sqrt{\widehat{v_t}} + \epsilon)$
  end if
end for

工作较为繁琐。

在实际应用中，AdamW 已成为许多深度学习框架（如 PyTorch）的默认优化器之一，特别适合处理高维参数空间和复杂模型结构。它的改进设计使其在保持 Adam 高效性的同时，进一步提升了模型的泛化性能，成为现代深度学习训练中的重要工具。

4.3.2.3 余弦退火调度器

余弦退火是一种在深度学习中广泛使用的学习率调度策略，其核心思想是通过余弦函数动态调整学习率，以优化模型训练过程。该方法最初由 Ilya Loshchilov 和 Frank Hutter 在 2016 年的 SGDR 方法[98] 中提出，随后成为深度学习领域的重要技术。余弦退火的名称来源于其数学实现方式——学习率按照余弦函数曲线从初始值平滑下降到最小值，类似于金属加工中的退火过程，即先高温加热后缓慢冷却。

从数学原理来看，余弦退火的公式为

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})(1 + \cos(\frac{T_{cur}}{T_i}\pi)) \tag{4.11}$$

其中 η_{max} 和 η_{min} 分别表示学习率的上下界，T_{cur} 是当前训练步数，T_i 是总步数。这

种设计使得学习率在训练初期保持较大值以加速收敛，随后逐渐减小以便模型精细调整参数。与传统的阶梯式衰减或线性衰减相比，余弦退火提供了更平滑的过渡，避免了学习率突变导致的训练不稳定。

在实际应用中，余弦退火具有多重优势。首先，其周期性调整特性有助于模型跳出局部最优解，这在处理多峰损失函数时尤为有效。其次，该方法减少了人工调参需求，通过预设的数学规律自动调整学习率，适用于各种复杂模型和大规模数据集。实验表明，在图像分类、自然语言处理等任务中，采用余弦退火策略的模型通常能获得更快收敛速度和更好泛化性能。例如在 ImageNet 竞赛中，ResNet 模型采用该策略后表现出显著提升。

4.3.2.4 权重衰减算法

权重衰减是深度学习中一种常用的正则化技术，主要用于防止模型过拟合。其核心原理是通过在损失函数中添加与权重参数平方和成正比的惩罚项（即 L_2 范数），约束模型参数的幅度。具体公式为

$$L' = L + \lambda \sum \theta_i^2 \tag{4.12}$$

其中 L 是原始损失函数，θ 为模型权重，λ 为正则化系数，控制惩罚强度。这种设计迫使模型在训练过程中倾向于学习较小的权重值，从而降低模型复杂度，避免对训练数据中的噪声过度敏感。

从实现机制来看，权重衰减在梯度下降过程中会直接影响参数更新。以随机梯度下降为例，这种操作相当于在每次迭代时对权重施加线性衰减，因此得名” 权重衰减”。值得注意的是，权重衰减通常不应用于偏置项，因为偏置主要控制输出平移，对模型复杂度影响较小。

权重衰减的作用机理可从多角度解释。首先，较小的权重使模型对输入变化更鲁棒，减少对噪声特征的依赖；其次，通过抑制特征间的共线性，避免模型过度依赖某些特定特征组合；最后，它能有效防止梯度爆炸问题，提升训练稳定性。实验表明，在 ImageNet 等大型数据集上，合理设置 λ 可使测试误差降低 10%-20%。

实际应用中，权重衰减常与其他技术（如 Dropout、批量归一化）结合使用。超参数 λ 的选择需通过多次验证确定，典型取值范围为 10^{-4} 到 10^{-2}，在本文中经多次测试找出相应权重衰减超参数。现代深度学习框架如本文使用的 PyTorch 中内置了权重衰减功能，通过上文提及的 AdamW 优化器的 weight_decay 参数即可启用。需要注

意的是，权重衰减与 L_2 正则化数学形式相似，但实现机制存在差异：前者直接修改优化过程，后者通过损失函数间接作用。

4.3.3 评价指标

本文共使用四种评价指标来评定，分别为**准确率**（Accuracy）、**精准率**（Precision）、**召回率**（Recall）和 **F1 分数**。这四种评价指标将在下面的小节中进行相应的介绍。由于模型改写文本检测本质上是多分类文本分类任务，因此后三者指标需要与二分类指标有相应变化。由于每个生成模型产生相同数量的文本，并且由于数据量对指标的影响最小，因此采用了对数据量敏感的宏观（macro）方法。

以及会介绍一下在后面的章节 4.3.5 中涉及到的 spearman 系数和 pearson 系数。此二者系数用于衡量生成的文本评估的分数与原始 AES2 数据集中数据分数的相关性与一致性。

4.3.3.1 准确率

准确率（Accuracy）是机器学习中最基础且直观的分类评估指标，用于衡量模型整体预测的正确性。其核心定义为模型正确预测的样本数占总样本数的比例，数学表达式为

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.13}$$

其中 TP（真正例）和 TN（真负例）分别表示正负类预测正确的数量，FP（假正例）和 FN（假负例）则为预测错误的数量。例如在疾病诊断场景中，若模型对 100 名患者正确识别了 90 人（含健康与患病），准确率即为 90%，直观反映模型的综合判断能力。

准确率的优势在于计算简单、解释性强，适用于类别分布均衡的场景。当正负样本比例接近时（如 50% 患病和 50% 健康），该指标能有效评估模型性能。但其局限性在于对不平衡数据敏感——若 99% 样本为负类，模型仅预测负类即可获得 99% 的准确率，却完全忽略正类识别。因此，在欺诈检测或罕见病诊断等少数类关键场景中，需结合精确率、召回率等指标综合评估。

实际应用中，准确率常见于多分类任务（如手写数字识别）和快速模型验证阶段。然而，高准确率未必代表模型优越性，例如选择题全选某一选项可能在特定数据分布下获得虚高数值，此时需结合混淆矩阵深入分析。未来随着不平衡学习技术的发展，

准确率将更多作为基础指标与其他评估方法协同使用。

4.3.3.2 精准率

精准率（Precision）是评估分类模型性能的核心指标之一，主要用于衡量模型预测为正类的样本中实际为正类的比例。其数学定义为

$$Precision = \frac{TP}{TP + FP} \tag{4.14}$$

其中 TP（真正例）表示模型正确预测的正类数量，FP（假正例）代表模型错误预测为负类的正类数量。该指标特别关注模型预测为正类的可靠性，例如在医疗诊断中，高精准率意味着较少健康患者被误诊为患病，从而避免不必要的治疗成本。

精准率的应用场景具有鲜明特征。当误报（False Positive）代价较高时，如垃圾邮件过滤或金融风控领域，精准率成为首要考量指标。在这些场景中，将正常邮件误判为垃圾邮件（FP）可能导致重要信息丢失，或将正常交易误判为欺诈（FP）会引发客户投诉，因此需要尽可能降低 FP 值以提高精准率。实验数据显示，卷积神经网络（CNN）等深度学习模型通过调整分类阈值，可在图像分类任务中将精准率提升至 83% 以上。

该指标存在明确的局限性。精准率单独使用时无法反映模型对正类样本的覆盖能力，可能掩盖高漏报（False Negative）问题。例如在癌症筛查中，仅追求高精准率可能导致实际患者未被检出（FN），因此需结合召回率（Recall）形成 F1 分数进行综合评估。研究还发现，当样本分布严重失衡时（如正类仅占 0.4%），仅依赖精准率会产生误导性结论，此时需配合 ROC 曲线等工具分析。

在本文实验中，由于模型改写文本检测本质上是多分类文本分类任务，因此精准率指标需要与二分类指标有相应变化。由于每个生成模型产生相同数量的文本，并且由于数据量对指标的影响最小，因此采用了对数据量敏感的宏观（macro）方法。

4.3.3.3 召回率

召回率（Recall）是机器学习中评估分类模型性能的核心指标之一，主要用于衡量模型对正类样本的识别能力。其数学定义为

$$Recall = \frac{TP}{TP + FN} \tag{4.15}$$

其中 TP 表示真正例（模型正确预测的正类样本数），FN 代表假负例（实际为正类但被误判为负类的样本数）。该指标反映的是所有真实正类样本中被模型成功识别的

比例，因此也被称为" 查全率"。在医疗诊断等高风险场景中，高召回率意味着更少的漏诊风险，例如癌症筛查模型若达到 90% 召回率，说明 90% 的真实患者被正确检出。

召回率的应用场景具有显著特征。当漏检（False Negative）代价较高时，如金融欺诈检测或安全监控领域，该指标成为首要优化目标。实验数据显示，通过调整分类阈值或采用集成学习方法，深度学习模型在图像识别任务中可将召回率提升至 85% 以上。但与精确率（Precision）存在权衡关系——过度追求高召回率可能导致误报增加，例如将正常交易误判为欺诈的比例上升，因此实际应用中常结合 F1 分数综合评估。

该指标存在明确的局限性。在样本分布严重失衡时（如正类仅占 1%），单纯依赖召回率会产生误导性结论。例如垃圾邮件分类器中，若将所有邮件预测为正常邮件（负类），召回率将显示为 0%，但显然这种模型毫无实用价值。因此将召回率与其他三个指标结合起来，可以大幅度弥补其不足之处。

在本文实验中，由于模型改写文本检测本质上是多分类文本分类任务，因此召回率指标需要与二分类指标有相应变化。由于每个生成模型产生相同数量的文本，并且由于数据量对指标的影响最小，因此采用了对数据量敏感的宏观（macro）方法。

4.3.3.4 F1 分数

F1 分数是机器学习中用于评估分类模型性能的核心指标，它通过调和精确率（Precision）和召回率（Recall）来提供综合评估。其计算公式为

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.16}$$

其中精确率衡量模型预测为正类的准确性（$Precision = \frac{TP}{TP+FP}$），召回率评估模型捕捉正类的能力（$Recall = \frac{TP}{TP+FN}$）。这种设计使得 F1 分数对两者赋予同等权重，任何一方的低下都会显著拉低整体得分。

F1 分数的核心价值在于处理类别不平衡的场景。例如在医疗诊断中，若正类样本（患者）仅占 1%，单纯依赖准确率会导致模型因全预测为负类而虚高至 99%，而 F1 分数能有效暴露这种缺陷。实验显示，当癌症筛查模型的召回率为 80%、精确率为 50% 时，F1 分数为 61.5%，比仅关注单一指标更能反映实际性能。其取值范围为 0 到 1，越接近 1 表明模型在精确性和全面性上越均衡。

该指标存在明确的局限性。首先，它完全忽略真负例（TN）的影响，在负类占比极高的场景中可能产生偏差；其次，其默认的等权重设计可能不符合某些业务需求

（如金融风控更重视精确率）。

实际应用中，F1 分数在文本分类、欺诈检测等领域表现突出。例如电商平台检测虚假评论时，模型 A（精确率 90%/召回率 45%）与模型 B（精确率 60%/召回率 90%）的 F1 分数分别为 60% 和 72%，后者因更高的综合性能成为优选方案。对于多分类问题，可通过宏平均（Macro-F1）或微平均（Micro-F1）扩展，前者平等对待各类别，后者更关注大类别样本。

在本文实验中，由于模型改写文本检测本质上是多分类文本分类任务，因此 F1 分数指标需要与二分类指标有相应变化。由于每个生成模型产生相同数量的文本，并且由于数据量对指标的影响最小，因此采用了对数据量敏感的宏观（macro）方法。

4.3.3.5 Spearman 系数

Spearman 相关系数是一种非参数的统计量，用于衡量两个变量之间的单调关系强度。它由心理学家 Charles Spearman 在 1904 年提出[99]，主要基于变量的等级次序而非原始数值进行计算，这使得它特别适用于处理非线性关系或不符合正态分布的数据。与 Pearson 相关系数不同，Spearman 系数不要求变量满足线性假设，对异常值也更具稳健性。

该系数的取值范围在-1 到 1 之间：当系数接近 1 时，表示两个变量的等级顺序完全一致，存在强正相关；接近-1 时则表示完全相反的等级顺序，存在强负相关；若系数接近 0 则表明变量间缺乏单调关系。其经典计算公式为

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{4.17}$$

其中 d_i 表示两个变量对应观测值的秩次差，n 为样本量。当数据存在相同秩次时，需要采用更复杂的计算方法，此时 Spearman 系数实际上等同于对秩次变量计算 Pearson 相关系数。

在实际应用中，Spearman 系数具有广泛适用性。在社会科学领域可用于分析教育程度与收入的关系；医学研究中能评估血压与年龄的相关性；工程领域则适用于分析半导体芯片性能指标间的关联。其优势在于能够处理定序数据，且不受异常值或数据分布形态的过度影响。但需注意，当样本量较小时其统计效力会降低，且对完全相同的秩次处理需要特殊方法。

显著性检验通常通过计算如下两个式子，分别为 $z = \rho\sqrt{n-1}$（近似服从标准正态分布）或 $t = \rho\sqrt{(n-2)/(1-\rho^2)}$（服从 t 分布）来实现。现代统计软件如提供了便

捷的计算函数。这些工具使得研究者能更高效地应用该方法来揭示变量间的潜在关联模式。

4.3.3.6 Pearson 系数

皮尔逊相关系数[100]（Pearson Correlation Coefficient）是统计学中用于衡量两个连续变量之间线性关系强度和方向的指标。该系数由卡尔·皮尔逊（Karl Pearson）在19 世纪末提出，其取值范围介于-1 到 1 之间。当系数为 1 时表示完全正相关，即一个变量增加时另一个变量也严格按比例增加；为-1 时表示完全负相关，即一个变量增加时另一个变量严格按比例减少；系数为 0 则表明变量间不存在线性关系。

从数学定义来看，皮尔逊系数是两个变量协方差与各自标准差乘积的比值，其计算公式为：

$$r = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}} \tag{4.18}$$

其中 $\bar{X}$ 和 $\bar{Y}$ 分别表示变量的样本均值。这个公式本质上是对协方差进行标准化处理，使得结果不受变量量纲的影响。在实际应用中，该系数对数据的正态分布性有要求，且只能反映线性关系，对于非线性关联可能会得出错误结论。

根据系数绝对值的大小，相关性强度可分为五个等级：0.8-1.0 为极强相关，0.6-0.8 为强相关，0.4-0.6 为中等相关，0.2-0.4 为弱相关，0.0-0.2 为极弱或无相关。值得注意的是，强相关性并不意味着因果关系，可能由第三方变量或偶然因素导致。在金融领域，该系数常用于分析证券收益率的联动性；在医学研究中可评估临床指标间的关联；工程领域则用于分析工艺参数与产品性能的关系。

皮尔逊系数的显著性检验通常通过 t 统计量实现，其自由度为 $n-2$（n 为样本量）。当 p 值小于显著性水平（如 0.05）时，才能认为相关性具有统计学意义。此外，该系数对异常值敏感，且当数据存在相同秩次时需要采用特殊处理方法。与 Spearman 秩相关系数相比，皮尔逊系数更适用于连续变量且满足线性假设的场景，而前者能处理更广泛的单调关系。

4.3.4 对比实验结果分析

本节通过两组对照实验系统评估了 RoBERTa 和 DeBERTa 模型在不同粒度文本理解任务中的表现。表 4.9 展示了文档级任务上的微调结果，其中 DeBERTa-V3-Large 以 85.97% 的准确率和 85.84% 的 F1 分数显著领先，验证了大规模预训练模型在处理

长文本依赖关系时的优势。相比之下，表 4.10 揭示的句子级任务结果呈现出截然不同的特征：所有模型的绝对性能明显下降（最佳准确率仅 49.61%），且 DeBERTa-V3-Base 在精准率指标上意外超越其大型版本。这种差异凸显了文本处理粒度对模型性能的关键影响——文档级任务受益于丰富的上下文信息，而句子级任务则更依赖精准的局部语义理解。后续分析将深入探讨造成这种差异的内在机制，以及不同模型架构在两类任务中的适应性表现。

4.3.4.1 文档级数据微调实验

表 4.9 详细展示了 RoBERTa 和 DeBERTa 两个主流预训练语言模型家族在文档级自然语言处理任务上的微调性能对比结果。通过对准确率（Accuracy）、精准率（Precision）、召回率（Recall）和 F1 分数四个核心评估指标的全面分析，我们可以得出若干重要发现。

表 4.9 RoBERTa 与 DeBERTa 模型在文档级数据上微调结果

模型	准确率	精准率	召回率	F1
RoBERTa-Base	72.88	77.26	72.89	72.73
RoBERTa-Large	80.92	81.84	80.92	80.55
DeBERTa-V3-Base	79.85	81.44	79.85	79.81
DeBERTa-V3-Large	**85.97**	**86.31**	**85.96**	**85.84**

从整体性能趋势来看，所有评估指标都呈现出明显的层级结构：DeBERTa-V3-Large 以显著优势领先，其准确率达到 85.97%，精准率 86.31%，召回率 85.96%，F1 分数 85.84%，这四项指标均明显优于其他对比模型。这一结果验证了模型规模和架构改进的双重重要性。值得注意的是，DeBERTa-V3-Base（79.85% 准确率）与 RoBERTa-Large（80.92% 准确率）的性能相当接近，这表明 DeBERTa 系列在基础模型规模下就能达到与更大规模 RoBERTa 模型相近的性能水平，充分体现了其架构设计的先进性。

深入分析模型架构的影响可以发现，DeBERTa 系列采用的分离注意力机制和增强型掩码解码器带来了显著优势。特别是在精准率指标上，DeBERTa-V3-Base（81.44%）已经超过了 RoBERTa-Large（81.84%），而 DeBERTa-V3-Large 更是将这一优势扩大到 86.31%。这表明 DeBERTa 的架构改进特别有利于提升模型预测的精确性，减少假阳性错误的发生。这一特性在需要高精度预测的应用场景（如医疗诊断或金融风险评

估）中尤为重要。

从模型规模的角度观察，RoBERTa 从 Base 到 Large 版本的性能提升幅度为 8.04 个百分点（准确率从 72.88% 到 80.92%），而 DeBERTa 从 Base 到 Large 版本的提升幅度更大，达到 6.12 个百分点（从 79.85% 到 85.97%）。虽然绝对提升值略低，但考虑到 DeBERTa-Base 的起点更高，这一相对提升仍然非常可观。这证实了增加模型参数规模对性能提升的持续有效性，同时也表明在优质架构设计的基础上，扩大模型规模能带来更显著的效果。

特别值得注意的是各模型在精准率和召回率之间的平衡关系。所有模型的精准率都略高于召回率，这种系统性差异可能反映了当前预训练语言模型在文档级任务中普遍存在的保守预测倾向。RoBERTa-Base 的精准率（77.26%）比召回率（72.89%）高出 4.37 个百分点，这一差距在更大规模的模型中有所缩小，DeBERTa-V3-Large 的精准率（86.31%）仅比召回率（85.96%）高出 0.35 个百分点。这表明随着模型规模和架构的优化，模型在保持高精确性的同时，也能更好地覆盖正类样本，实现更均衡的性能表现。

F1 分数的变化趋势进一步验证了上述发现。作为精准率和召回率的调和平均数，F1 分数综合反映了模型的整体分类性能。DeBERTa-V3-Large 的 F1 分数达到 85.84%，比 RoBERTa-Large 的 80.55% 高出 5.29 个百分点，这一提升幅度超过了准确率的提升（5.05 个百分点），说明 DeBERTa 的优势在综合考虑精确性和全面性时更为明显。

从实际应用的角度来看，这些结果对模型选择具有重要指导意义。在计算资源允许的情况下，DeBERTa-V3-Large 显然是首选方案；在资源受限时，DeBERTa-V3-Base 能以较小的参数量达到与 RoBERTa-Large 相当的性能，是更经济高效的选择。此外，精准率普遍高于召回率的现象提示我们，在这些模型的工业部署中可能需要根据具体应用场景的需求，通过调整分类阈值来优化精确性或召回率。

这些实验结果也为未来研究指明了若干方向。首先，DeBERTa 架构的成功验证了分离注意力机制在文档级任务中的有效性，值得进一步研究和扩展。其次，模型规模扩大带来的性能提升尚未达到平台期，暗示继续增大模型规模可能仍有潜力。最后，精准率和召回率之间的不平衡现象值得深入分析，可能需要设计专门的损失函数或训练策略来改善。

综上所述，表 4.9 通过系统的对比实验，不仅展示了不同预训练语言模型在文档级任务上的性能差异，更为模型架构设计、规模选择和实际应用提供了重要的实证依

据。这些发现将有助于研究人员和工程师在文档级自然语言处理任务中做出更明智的模型选择和应用决策。

4.3.4.2 句子级数据微调实验

表 4.10 展示了 RoBERTa 和 DeBERTa 系列预训练语言模型在句子级自然语言处理任务上的微调性能对比。通过分析准确率 (ACC)、精准率 (P.)、召回率 (R.) 和 F1 分数四个关键评估指标，我们可以得出以下深入观察和结论。

表 4.10 RoBERTa 和 DeBERTa 在句子级数据上微调结果

模型	准确率	精准率	召回率	F1
RoBERTa-Base	45.08	50.13	45.08	43.93
RoBERTa-Large	45.69	48.38	45.69	45.44
DeBERTa-V3-Base	47.76	**51.49**	47.76	46.93
DeBERTa-V3-Large	**49.61**	50.25	**49.62**	**49.25**

从整体性能表现来看，所有模型在句子级任务上的表现明显低于文档级任务（对比表 4.9），这反映了句子级理解任务固有的挑战性。DeBERTa-V3-Large 依然保持领先地位，其准确率达到 49.61%，F1 分数 49.25%，但相比文档级任务 85.97% 的准确率有显著下降。这种性能差距凸显了句子级语义理解的复杂性，其中上下文信息有限，模型更难捕捉深层次的语义关系。

值得注意的是，DeBERTa-V3-Base 在精准率指标上以 51.49% 的表现超过所有其他模型，包括其大型版本 DeBERTa-V3-Large（50.25%）。这一反常现象可能表明，在句子级任务中，基础规模的 DeBERTa 模型在避免假阳性预测方面具有特殊优势。同时，RoBERTa 系列的表现相对较弱，RoBERTa-Large 的准确率仅为 45.69%，甚至低于 DeBERTa-V3-Base 的 47.76%，这说明在句子级任务中，模型架构的改进比单纯增加参数规模更为重要。

深入分析各指标之间的关系可以发现，所有模型的精准率都高于召回率，这与文档级任务中的观察一致，但差距更为显著。RoBERTa-Base 的精准率（50.13%）比召回率（45.08%）高出 5.05 个百分点，DeBERTa-V3-Base 的差距更是达到 3.73 个百分点（51.49% vs 47.76%）。这种系统性偏差表明，当前预训练语言模型在句子级任务中普遍倾向于保守预测，可能为了避免错误而牺牲了部分正类样本的识别能力。

从模型规模的影响来看，RoBERTa 从 Base 到 Large 版本的性能提升非常有限，准确率仅增加 0.61 个百分点（45.08% 到 45.69%），F1 分数增加 1.51 个百分点（43.93% 到 45.44%）。相比之下，DeBERTa 从 Base 到 Large 版本的提升更为明显，准确率增加 1.85 个百分点（47.76% 到 49.61%），F1 分数增加 2.32 个百分点（46.93% 到 49.25%）。这一对比强烈表明，在句子级任务中，DeBERTa 的架构设计能够更有效地利用增加的模型容量。

特别值得注意的是各模型在 F1 分数上的表现。作为精准率和召回率的调和平均，F1 分数更好地反映了模型的整体分类能力。DeBERTa-V3-Large 以 49.25% 的 F1 分数领先，其次是 DeBERTa-V3-Base 的 46.93%，两者都明显优于 RoBERTa 对应版本。这一差距说明 DeBERTa 的分离注意力机制和增强的位置解码设计特别适合处理句子级的语义关系建模。

从实际应用角度考虑，这些结果提供了重要启示。在句子级任务中，DeBERTa-V3-Base 可能是性价比最高的选择，它在多个指标上接近甚至超过更大规模的模型，同时计算成本更低。此外，所有模型相对较低的绝对性能分数（均低于 50%）表明，当前的预训练语言模型在句子级理解任务上仍有很大改进空间，可能需要开发专门的架构改进或训练策略。

这些实验结果也提出了若干值得深入研究的问题。首先，为什么 DeBERTa-V3-Base 在精准率上表现最佳？这是否与其基础规模的参数配置和特定注意力模式的交互有关？其次，为什么模型规模扩大带来的性能提升在句子级任务中不如文档级任务显著？这可能反映了句子级和文档级任务对模型能力的不同需求。最后，如何解释所有模型在句子级任务中表现出的更大的精准率-召回率差距？这可能与句子级数据的特性或评估方式有关。

综上所述，表 4.10 通过详实的实验数据，揭示了预训练语言模型在句子级任务中的性能特点和局限。这些发现不仅对模型选择有直接指导意义，也为未来研究指明了改进方向，包括开发更适合句子级理解的架构变体、设计更有效的微调策略，以及深入理解不同粒度文本处理任务的本质差异。这些见解将有助于推动自然语言处理技术在句子级理解任务上的进一步发展。

4.3.5 探索性数据分析实验

对于目前提出的数据集，此时产生了两个新问题。如何评估原文对用大语言模型改写后的文本的影响？如何评估大语言模型改写后的文本与原文的质量一致性？为解决这两个问题，因此本文设计了如下两个实验来探索数据分析。

4.3.5.1 被改写文本与原文文本相似性

为解决第一个问题，我们可以使用 DeBERTa-V3-Large 模型分别提取文本特征后，再使用余弦相似度的方式来评估大语言模型改写后的文本与原文的质量一致性。被改写文本与原始文本经过原始 DeBERTa-V3-Large 模型提取特征后的余弦相似度如表 4.11 所示，展示了不同文本生成模型在语义保持性方面的量化评估结果。需要特别说明的是，本实验中的余弦相似度指标仅用于衡量改写文本与原始文本的语义相似程度，数值越高表明模型对原文的改动越小，但这并不直接等同于模型质量的优劣。AES2 作为原始文本的基准参照，其 100% 的相似度结果验证了实验设置的有效性，为其他模型的评估提供了理想的上界参考。表中加粗表示该数值为最大值，下划线表示该数值为最小值。

表 4.11 被改写文本与原始文本经过原始 DeBERTa-V3-Large 模型提取特征后的余弦相似度。

	余弦相似度	
模型	训练改写集	测试改写集
AES2	100.00	100.00
GPT3.5	**96.71**	**96.77**
GPT4o	96.39	96.39
Gemini	96.08	95.99
Qwen	<u>95.55</u>	<u>95.53</u>
Deepseek	95.71	95.68

从实验结果来看，所有测试模型在两个改写数据集上的表现呈现出高度一致性。GPT3.5 在训练集和测试集上分别达到 96.71% 和 96.77% 的相似度，这一微小差异（仅 0.06 个百分点）证实了实验结果的可靠性。值得注意的是，虽然标注为” 训练改写集” 和” 测试改写集”，但实际上这些数据并未用于任何模型的训练过程。GPT4o 以 96.39% 的稳定表现紧随其后，而 Gemini、Deepseek 和 Qwen 等模型的表现则相对接近，集中

在 95.5%-96.1% 的区间内。

这些数据揭示了当前大语言模型在文本改写任务中的一些共性特征。首先，各模型普遍保持了较高的语义相似度（均超过 95%），这说明主流大语言模型都具备较强的语义理解与保持能力。其次，约 4-5 个百分点的性能差距反映了不同模型在改写策略上的差异：相似度较高的模型（如 GPT3.5）可能更倾向于保守的改写策略，而相似度稍低的模型（如 Qwen）可能在改写过程中进行了更大胆的语义重组或表达创新。需要强调的是，这种差异并不必然代表模型质量的优劣，而是体现了不同的设计取向——较高的相似度适合需要严格保持原意的场景，而较低的相似度可能带来更具创造性的改写结果。

从技术角度来看，这些结果引发了一些值得深入探讨的问题。为什么不同模型之间会存在这种系统性差异？可能的影响因素包括：模型预训练阶段使用的语料特性、微调策略的差异、解码过程中的温度参数设置等。特别是 GPT3.5 与其后续版本 GPT4o 之间的表现差异，可能反映了 OpenAI 在不同版本迭代过程中对” 忠实度” 与” 创造性” 这两个维度的权衡调整。此外，所有模型与理想值 100% 之间存在的差距也表明，即使是当前最先进的大语言模型，在文本改写过程中也难以完全避免语义信息的细微变化。

在实际应用层面，这些发现为模型选择提供了重要参考。在需要严格保持原意的场景（如法律文书、学术论文的改写）中，相似度更高的 GPT3.5 可能是更稳妥的选择；而在允许一定语义灵活度的场景（如创意写作、广告文案生成）中，相似度稍低但可能更具创造性的其他模型也值得考虑。同时，这些结果也提示我们，在评估文本改写模型时，应该根据具体应用场景的需求，在” 忠实度” 与” 创造性” 之间寻找合适的平衡点，而非简单地追求单一指标的最大化。

4.3.5.2 被改写文本与原文质量一致性

如何评估大语言模型改写后的文本与原文的质量一致性呢？解决第二个问题的方法为，我们可以将原始的 AES2 数据集中的评分作为衡量文本质量的依据，将 DeBERTa-V3-large 模型在原始 AES2 数据集上微调后，再生成所有文本的评分。最后依靠 Spearman 系数和 Pearson 系数来评价大语言模型改写后的文本与原文的质量一致性。表 4.12 详细列出了本实验所采用的关键参数配置和模型设置。在模型架构方面，我们选择了 DeBERTa-V3-Large 作为基础模型，该模型通过引入分离注意力机制和增强的位置编码，在多项自然语言处理任务中表现出色。实验数据集采用 AES2，这

是一个经过专业标注的语义评估基准数据集，特别适合用于文本语义相似性分析任务。

表 4.12　评分一致性实验设置

字段	值
模型	DeBERTa-V3-Large
数据集	AES2
初始学习率	10^{-5}
权重衰减权重	0.01
批大小	4
调度器	余弦退火
损失函数	交叉熵函数
优化器	AdamW

在训练参数设置方面，我们采用了较为保守的初始学习率 10^{-5}，这一选择基于预实验的结果，既能保证模型参数的有效更新，又能避免训练过程中的不稳定性。权重衰减系数设为 0.01，这一正则化设置有助于防止模型过拟合。考虑到模型规模和显存限制，我们将批大小设置为 4，这一较小的批处理规模是由于 DeBERTa-V3-Large 需要占用较大显存导致不得不设置得很小。这样虽然会降低训练速度，但也能提高梯度更新的稳定性。

训练过程中采用了余弦退火学习率调度策略，该策略能够实现学习率的平滑衰减，在训练初期保持较大的学习率以加快收敛，在后期自动降低学习率以获得更精细的参数调整。损失函数选用标准的交叉熵函数，这是分类任务中最常用的目标函数。优化器采用 AdamW，这是 Adam 优化器的改进版本，通过正确处理权重衰减，在保持自适应学习率优势的同时，能获得更稳定的训练效果。

这些超参数的选择经过了多次实验验证，在保证模型性能的同时，也考虑了训练效率和稳定性。DeBERTa-V3-Large 模型与 AES2 数据集的组合，以及上述训练参数的配置，共同构成了一个平衡而高效的实验设置，为后续的语义相似性分析提供了可靠的技术基础。这些参数设置既参考了相关领域的主流做法，也针对具体任务需求进行了适当调整，体现了实验设计的科学性和严谨性。

表4.13展示了基于 AES2 数据集微调后的 DeBERTa-V3-Large 模型在不同改写文

本检测数据集上的评估结果，分别通过 Spearman 相关系数和 Pearson 相关系数来衡量模型评分与 AES2 数据集人工评分之间的一致性。实验结果表明，不同改写模型生成的文本在语义保持性方面存在显著差异，这些差异在训练集和测试集上呈现出不同的分布模式。由于微调模型是在 AES2 模型的训练集上进行训练，因此基于训练集文本被大语言模型改写后的文本可能在某种程度上被微调模型见过，故而我们将数据集分为了微调模型可能见过的训练改写集和一定没见过的测试改写集。

表 4.13　使用 AES2 数据集微调后 DeBERTa-V3-Large 模型在模型改写文本检测数据集上文本评分与 AES2 数据集评分的 spearman 和 pearson 系数

	训练改写集		测试改写集	
模型	**Spearman**	**Pearson**	**Spearman**	**Pearson**
AES2	92.70	92.38	78.26	78.10
GPT3.5	84.42	84.40	76.97	76.77
GPT4o	**84.83**	**84.54**	77.47	77.17
Gemini	83.72	83.46	77.18	76.44
Qwen	83.01	82.68	**78.20**	**77.20**
Deepseek	83.63	83.35	77.12	76.65

从整体表现来看，模型在 AES2 数据集上的原始文本在训练改写集与测试改写集上取得了最优异的成绩，Spearman 和 Pearson 系数分别达到 92.70% 和 92.38%，这一数据实际上是在说明微调后 DeBERTa-V3-Large 模型的微调程度较为优异。然而，该模型在测试改写集上的性能出现明显下降（Spearman 78.26%，Pearson 78.10%），这表明评估模型的泛化能力仍存在提升空间。在商业大模型比较中，GPT4o 模型生成的文本在训练改写集上表现最为突出（Spearman 84.83%，Pearson 84.54%），而在开源模型中，Qwen 则在测试改写集上展现出相对优势（Spearman 78.20%，Pearson 77.20%），这种差异可能反映了不同模型在文本改写策略上的本质区别。

深入分析各模型的跨数据集表现可以发现几个重要现象。首先，所有模型在测试集上的相关系数均低于训练集，这体现了基于 AES2 训练集生成的文本确实在某种程度上被微调模型 DeBERTa-V3-Large 见过。其次，GPT 系列模型（GPT3.5 和 GPT4o）在两个数据改写集上的质量都保持了相对稳定的表现，这可能得益于其强大的预训练基础和稳健的文本生成能力。相比之下，Gemini 和 Deepseek 等模型虽然训练集表现

尚可，但在测试集上的下降幅度更为明显，特别是在 Pearson 系数方面（Gemini 测试集 Pearson 76.44%）。

从统计角度来看，Spearman 和 Pearson 系数之间普遍存在 0.2-0.3 个百分点的差异，这种微小差距表明模型评分与人工评分之间既保持了较好的线性关系（Pearson），也维持了稳定的单调相关性（Spearman）。特别值得注意的是，Qwen 模型在测试集上的 Spearman 系数（78.20%）甚至十分接近于 AES2 数据集中的原始文本（78.26%）。

然而，表中也存在着一些问题。例如，Qwen 模型在测试改写集上拥有与 AES2 最高的评分一致性（Spearman 78.20 和 Pearson 77.20），但是其在训练改写集上却拥有最低的评分一致性（Spearman 83.01 和 Pearson 82.68）。这一对比说明本实验仍存在某些潜在问题，可能是评估过程中存在未被发现的偏差因素，需要进一步分析模型在不同数据子集上的具体表现差异才能得出确切结论。

4.4 本章小结

本章系统地研究了基于预训练语言模型的文本改写检测方法，通过对比实验和消融分析，深入探讨了不同模型架构在文档级和句子级文本检测任务中的表现差异。主要研究内容和结论总结如下：

在模型架构方面，本章详细分析了 RoBERTa 和 DeBERTa 系列模型的技术特点。实验结果表明，DeBERTa-V3-Large 在文档级任务中表现最优，准确率达到 85.97%，显著优于其他对比模型。这验证了其解耦注意力机制和增强掩码解码器在处理长文本依赖关系时的优势。值得注意的是，DeBERTa-V3-Base 以较小的模型规模达到了接近 RoBERTa-Large 的性能水平，体现了架构创新的重要性。

在任务粒度方面，研究发现文档级和句子级任务呈现出不同的性能特征。所有模型在文档级任务上的表现明显优于句子级任务，这反映了丰富的上下文信息对模型性能的积极影响。特别地，在句子级任务中，模型规模扩大带来的性能提升相对有限，而架构差异的影响更为显著，这为不同应用场景下的模型选择提供了重要参考。

在探索性数据分析上，本章提出了基于余弦相似度和评分一致性的双重评估框架。实验发现，在文本相似性上，Qwen 具有最大的修改文本的倾向，而 OpenAI 的 GPT3.5 模型和 GPT4o 模型则在原始文本上仅作小修改。而在以评分作为文本质量代表的一致性实验中，各个大语言模型生成文本的评分均具有与原始文本评分较高的一致性，表明原始文本的文本质量会极大地影响大语言模型改写文本的文本质量。

然而，研究也揭示了当前方法的一些局限性。首先，模型在测试集上的性能下降表明泛化能力仍需提升；其次，句子级任务的整体性能偏低（最佳准确率仅 49.61%）说明现有方法在细粒度语义理解方面仍有改进空间；最后，在探索性数据分析实验中，被改写文本与原文质量一致性实验仍需改进。

这些发现为后续研究指明了多个方向：在文本质量一致性实验中开发更鲁棒的评估模型以提高跨数据集一致性；设计专门针对句子级任务的改进架构；探索更精细的评估指标来捕捉文本改写的多维特性。同时，实验结果也为实际应用中的模型选择提供了实证依据，特别是在需要权衡计算成本和性能表现的场景中。

总的来说，本章通过系统的实验设计和深入的分析，为模型改写文本检测领域提供了重要的方法学参考和技术基础，以推动相关技术在教育质量评估等实际应用中的发展。

第 5 章 模型改写文本检测系统设计与实现

本章主要介绍基于模型改写文本检测系统的设计与实现，第三、四章从理论和实验方面验证了本文提出方法的有效性，为了进一步验证方法在实际应用场景中的有效性，本文以提出的模型改写文本检测方法为基础，遵循软件开发过程的基本原则，设计并实现了模型改写文本检测系统，涵盖文本改写和检测模型改写两大文本生成功能。通过该系统，可以进一步提升文本改写的流畅性与多样性、检测模型改写的准确性。本章以需求分析、系统设计、系统实现以及系统功能展示四个部分介绍模型改写文本检测系统。

5.1 需求分析

本文研究的模型改写文本检测系统给自然语言处理领域带来提供了一种新的技术方案，能够有效提升对于学生写作文本是否由 AI 改写的检测能力，本文构建的模型改写文本检测系统旨在给用户提供一个功能完善的、性能优越的模型改写文本检测系统，该系统用户可以为用户提供多样的文本改写服务以及精准的模型文本改写检测服务。接下来将从业务需求和功能需求角度进行详细分析。

5.1.1 业务需求分析

随着人工智能技术的快速发展，文本改写和检测模型改写的需求日益增长。尤其是在教育领域，教师需要对学生的写作进行评估和反馈，而学生也希望能够通过改写工具提升自己的写作能力。因此，本文提出的模型改写文本检测系统应具备以下业务需求：

- **文本改写服务**：系统应能够提供高质量的文本改写服务，帮助用户提升文本的流畅性和多样性。
- **模型改写检测服务**：系统应能够准确检测出文本是否经过模型改写，帮助教师评估学生的写作水平。
- **用户友好的界面**：系统应具备简洁易用的用户界面，方便用户进行操作。
- **高性能和稳定性**：系统应具备良好的性能和稳定性，能够处理大量的文本数据。

- **安全性和隐私保护**：系统应确保用户数据的安全性和隐私保护，防止数据泄露。
- **可扩展性**：系统应具备良好的可扩展性，能够根据需求进行功能扩展和升级。
- **多平台支持**：系统应能够在不同的平台上运行，包括桌面和移动设备。
- **实时反馈**：系统应能够提供实时的文本改写和检测反馈，帮助用户及时了解文本的质量和改写效果。
- **用户管理**：系统应具备用户管理功能，支持用户注册、登录、权限管理等操作。

5.1.2 功能需求分析

本文从用户和管理员两个角度进行了系统的功能分析，以全面了解系统在不同用户身份下的需求和功能要求。用户功能需求分析从系统提供的服务和功能出发，如用户注册、登录、包括文本改写和检测文本改写功能以及用户历史信息管理等，以确保用户能够顺利地使用系统并获得良好的体验。管理员功能需求分析侧重于系统的管理和维护功能，如用户管理、功能管理、模型管理以及文本改写信息管理等，以确保系统能够高效稳定地运行，并及时处理用户反馈和问题。

模型改写文本检测系统的总体用例图如图 5.1 所示。

本系统用户功能需求分析如下：

1. **用户注册**：用户可以自行在系统上进行用户注册。
2. **用户登录**：用户可以使用已注册的账号密码登录系统。
3. **文本改写服务**：系统应当为用户提供文本改写服务，根据用户输入的文本，润色改写为流畅、通顺的文本。
4. **检测文本改写服务**：系统应当为用户提供检测某段文本是否被改写或由 AI 生成的服务，根据输入的源文本，得出检测结果。
5. **查看历史信息**：用户可以通过历史信息查询接口查看历史改写信息以及历史检测信息。
6. **文本生成信息导出与下载**：支持用户对文本信息进行导出并下载到本地。

本系统管理员功能需求分析如下：

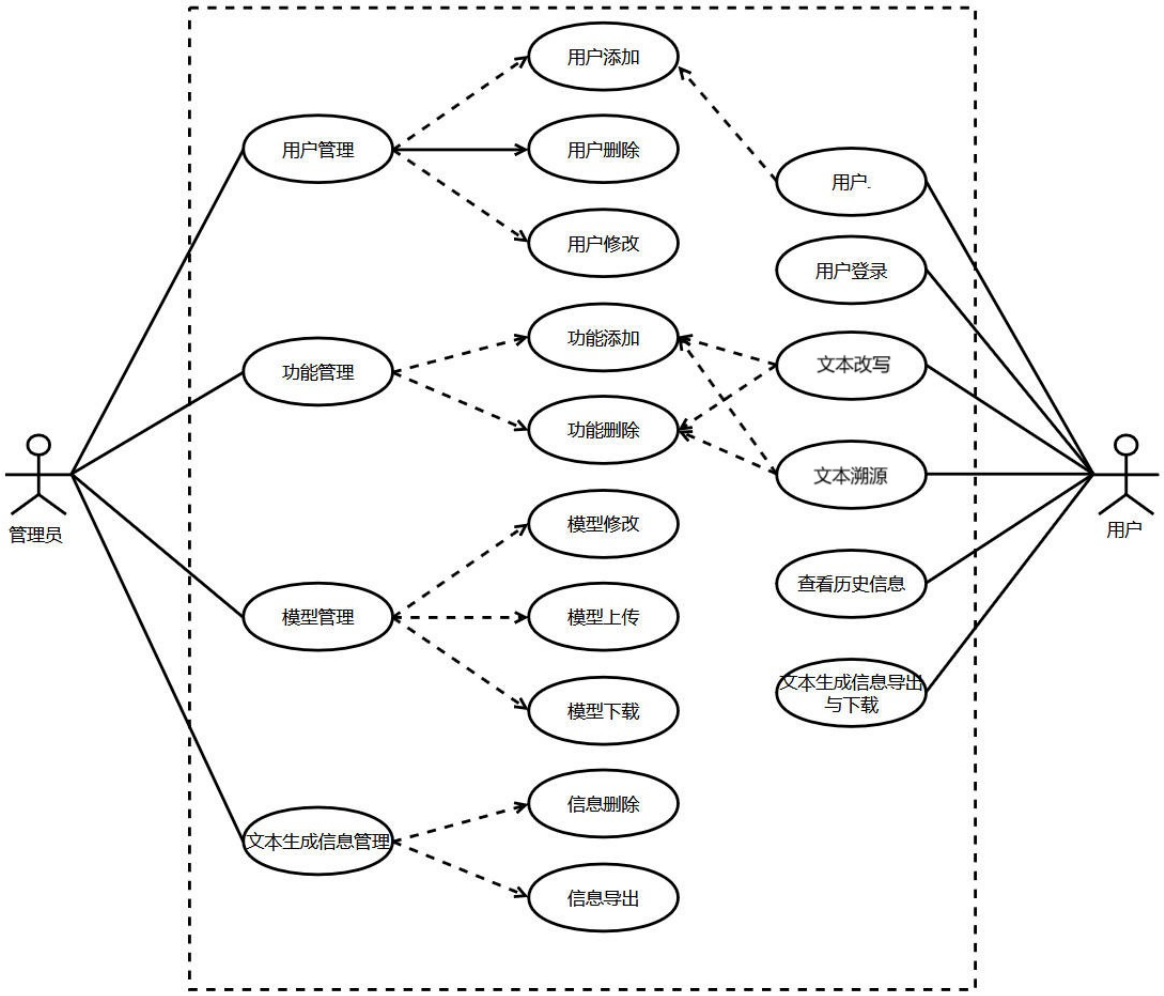


图 5.1　模型改写文本检测系统用例图

1. **用户管理**：管理员可以对用户进行添加、删除以及信息修改等操作。

2. **功能管理**：支持系统功能扩展，可以添加功能或者删除不需要的功能。

3. **模型管理**：管理员可以通过模型管理接口进行模型修改、上传和下载。

4. **文本生成信息管理**：对系统内冗余的文本信息进行删除或导出。

5.2 系统设计

本章主要介绍模型改写文本检测系统的架构设计、模块和接口设计以及相关信息的数据库存储设计。

5.2.1 系统架构设计

为了提高系统的灵活性、可扩展性和可维护性，本系统采用前后端分离的系统架构。前端只负责接收用户数据以及发送系统响应，后端提供具体的服务和处理逻辑，并将处理结果返回给前端。系统架构如图 5.2 所示，包括表现层、API 接口层、应用层、持久层以及基础层。

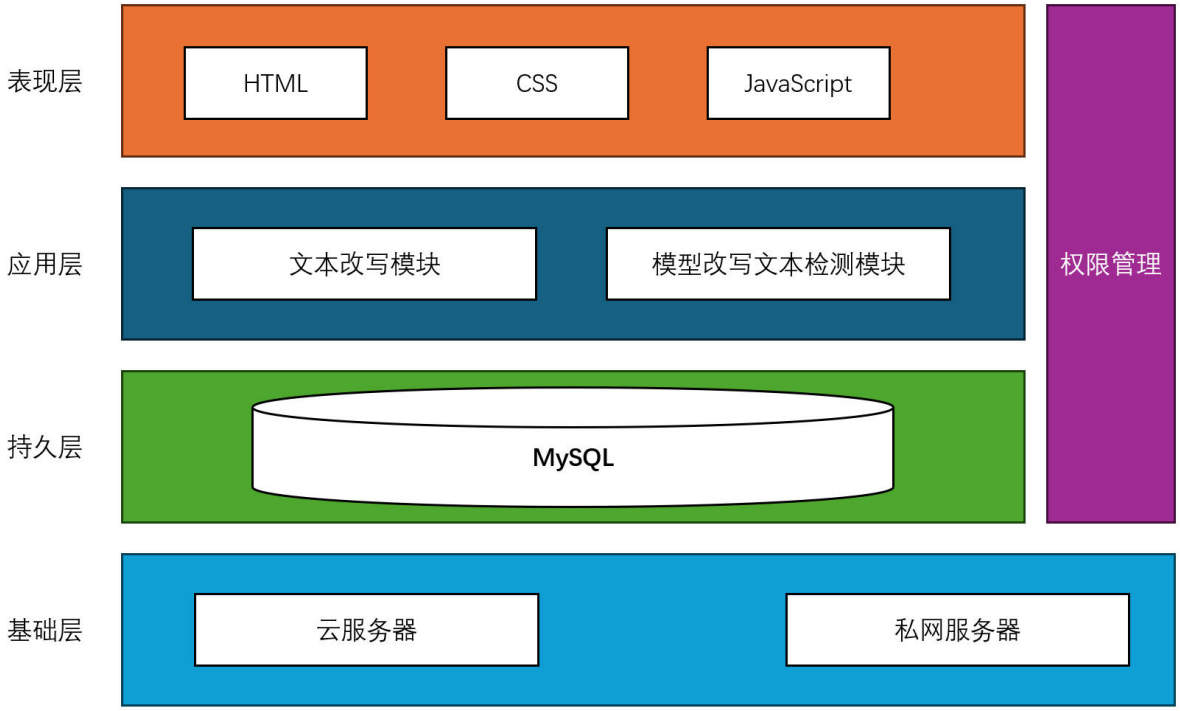


图 5.2　模型改写文本检测系统架构设计

1. **表现层**：负责处理用户界面逻辑和交互控制，管理界面逻辑和权限控制，将系统数据以用户友好的方式展示，并实现用户交互。包括用户界面设计、页面布局和组件设计、样式设计和美化。使用 HTML 定义页面结构，CSS 美化页面样式，JavaScript 处理页面交互和动态效果。

2. **API 接口层**：负责接收前端的请求并返回相应结果，使用 Nginx 进行反向代理，将外部请求转发给内部的服务器处理，并可以进行负载均衡，使系统能够处理大量请求并提供高性能的服务，Flask 应用程序接收到转发的请求后，执行相应的业务逻辑。通过独立的 API 接口层，前端和后端可以进行松耦合的交互，前端可以更灵活地调用后端的功能，而后端可以更容易地扩展和修改接口。

3. **应用层**：包含系统的核心业务逻辑，即对话生成模块以及机器翻译模块，另外

提供用户管理服务、管理员管理服务等。将业务逻辑独立成应用层，可以使得不同功能模块的开发、测试和部署更加独立和灵活，同时也方便了后续的功能扩展和升级。

4. **数据层**：负责管理系统的数据存储和访问，使用 MySQL 存储用户账户信息、个人资料、权限和角色信息以及用户活动记录等结构化信息。使用 MongoDB 存储生成的文本内容等非结构化信息。通过独立的数据层，可以使得系统的数据访问更加统一和安全，同时也方便了数据的管理和维护。

5. **基础层**：使用云服务器和私网服务器部署系统，提供基础运行环境。

通过采用前后端分离的系统架构，本系统能够实现前后端的松耦合、功能模块化和独立部署，提高了系统的灵活性、可扩展性和可维护性，同时也提高了开发效率和团队合作效率。

5.2.2 模块和接口设计

模型改写文本检测系统的模块和接口设计主要包括用户模块、管理员模块、文本改写模块和检测模型改写模块。每个模块都包含了相应的功能和接口设计，以满足系统的需求。

模型改写文本检测系统主要由用户模块、系统管理模块、和系统功能模块组成，模块间关系如图 5.3 所示。用户模块主要负责用户的注册与登录，允许用户创建账户、密码管理并提供身份验证授权功能，保证用户能够正常访问和使用系统服务。系统管理模块主要负责功能管理、模型管理和数据管理，允许对系统功能进行灵活扩展，管理员可以添加新功能或者移除不再需要的功能，通过模型管理接口进行系统模型的修改、上传和下载，数据管理功能主要是对文本生成信息的管理，可以通过系统内的接口删除冗余的文本信息或将其导出下载到本地。系统功能模块是系统的核心，提供了各种具体的系统服务，用于处理前端接收的请求。系统功能包括对话生成和机器翻译，通该模块，用户可以与系统进行交互，请求特定或执行特定的任务，系统会根据接收到的请求，调用相应的功能模块来进行处理，并生成对应的响应。

为了实现系统对用户的服务以及系统内不同模块之间数据传输，本章定义了系统核心接口，如表 5.1 所示。

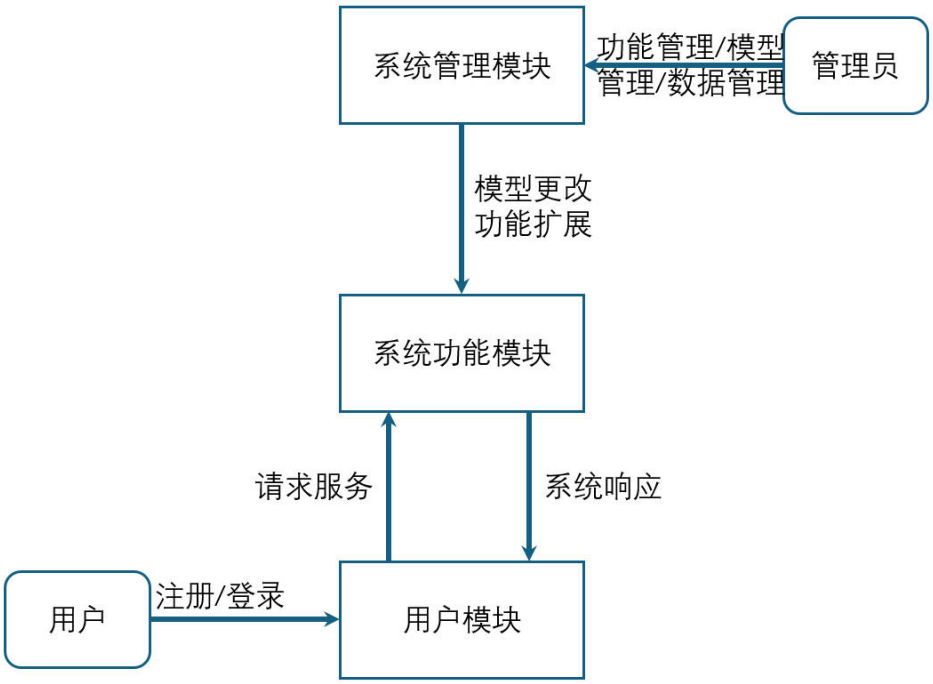


图 5.3　模型改写文本检测系统模块设计

5.2.3 数据库设计

根据系统存储需求，本文采用 MySQL 数据库存储结构化数据，包括用户表、功能表、模型表、文本数据表等。

MySQL 数据库设计主要包括用户表、功能表、模型表和文本数据表。每个表都包含了相应的字段和数据类型，以满足系统的需求。

用户表：记录了用户 ID、用户名、密码、邮箱、权限、注册时间和最近登录时间信息，如表 5.2 所示。

功能表：记录了系统中相关功能的信息，包括功能 ID、功能名称以及功能接口，如表 5.3 所示。

模型表：记录系统中模型的相关信息，包括模型 ID、模型名称、模型类型以及模型上传时间，如表 5.4 所示。

文本数据表：记录用户改写文本和模型改写文本信息，包括用户 ID、文本 ID、操作类型、操作时间、文本内容，如表 5.5 所示。

表 5.1　模型改写文本检测系统核心接口说明

接口名	接口描述
userRegister	接收用户提交注册信息，并进行用户注册。
userLogin	接收用户提交登录信息，并进行身份验证和授权。
getModelInfo	查询系统提供的模型，返回相关模型信息。
upLoadModel	用于上传已训练好的模型到系统中，以供后续的服务需求。
downLoadModel	用于从系统中导入已有模型。
funcDelete	用于删除系统中不需要的功能。
funcCreate	用于扩展系统功能，提供更好的服务。
dataManage	用于管理系统中的文本生成信息，支持删除或下载文本数据。
dialogService	接收用户输入，使用模型进行对话生成，并返回结果。
translationService	接收用户输入，使用模型进行机器翻译，并返回结果。

5.3 系统实现

本节主要介绍模型改写文本检测系统的使用到的工具和系统服务，并分模块对系统功能进行讲解。

5.3.1 开发环境与工具

本文采用前后端分离架构进行开发，前端采用 Vue 来实现用户界面，提供良好的交互体验。后端开发采用 Pytorch、Transformers 等技术，用于构建文本生成服务。在编程语言方面，使用 HTML、CSS、JavaScript 和 Python，分别用于前端和后端的开发。为了提高开发效率，选择 Vscode 作为开发工具。在数据存储方面，系统使用了 MySQL 作为数据库，用于存储用户信息和生成的文本数据。为方便起见，使用 Flask 框架处理后端业务逻辑。为了保证系统安全, 采用 JWT 进行权限管理、验证用户身份。本文使用到的所有开发环境和工具如表 5.6 所示。

5.4 系统展示

由于管理员仅有在系统后台调用脚本进行管理操作，本文主要展示用户端的系统功能。用户端主要包括用户注册与登录、文本改写功能、检测模型改写功能和历史信息查询功能。用户注册与登录功能主要用于用户注册和登录系统，文本改写功能主要

表 5.2　用户表

序号	字段	数据类型	说明	是否主键
1	user_id	int	用户 ID	是
2	user_name	varchar(30)	用户名	否
3	password	varchar(128)	密码	否
4	email	varchar(30)	邮箱	否
5	authority	enum	权限	否
6	register_time	datetime	注册时间	否
7	lat_login_time	datetime	最近登录时间	否

表 5.3　功能表

序号	字段	数据类型	说明	是否主键
1	fuc_id	int	功能 ID	是
2	fuc_name	varchar(30)	功能名称	否
3	fuc_interface	varchar(30)	功能接口	否

用于对输入的文本进行改写，检测模型改写功能主要用于检测输入的文本是否经过模型改写，历史信息查询功能主要用于查询用户的历史操作记录。

5.4.1 用户注册与登录

用户可以使用已注册的账号密码登录系统，如图 5.4a 所示。用户注册时需要填写用户名、密码和邮箱等信息，系统会对用户输入的信息进行验证，确保信息的合法性和完整性。如果未有注册账号，用户可以通过注册页面进行注册，如图 5.4b 所示。

5.4.2 文本改写功能

用户登录后，可以在系统中进行文本改写操作，如图 5.5 所示。用户输入需要改写的文本，系统会使用模型进行改写，并返回改写后的文本。用户可以对改写后的文本进行查看和下载。用户可以选择不同的改写模型进行文本改写，系统会根据用户选择的模型进行相应的处理。用户还可以对改写后的文本进行编辑和修改，以满足自己的需求。

表 5.4　模型表

序号	字段	数据类型	说明	是否主键
1	model_id	int	模型 ID	是
2	model_name	varchar(20)	模型名称	否
3	model_type	varchar(20)	模型类型	否
4	model_size	varchar(20)	模型大小	否
5	upload_time	datetime	上传时间	否

表 5.5　文本数据表

序号	字段	数据类型	说明	是否主键
1	user_id	int	用户 ID	是
2	text_id	int	文本 ID	是
3	text_type	varchar(20)	操作类型	否
4	generated_time	datetime	操作时间	否
5	text_content	text	文本内容	否

5.4.3 模型改写文本检测功能

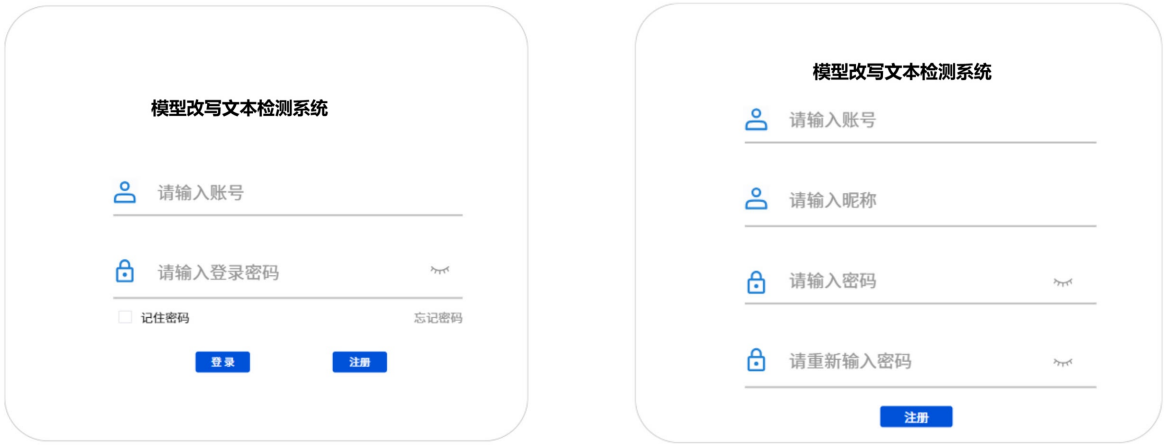
用户登录后，可以在系统中进行模型改写文本检测操作，如图 5.6 所示。用户输入需要检测的文本，系统会使用模型进行检测，并返回检测结果。用户可以对检测结果进行查看和下载。用户还可以选择不同的检测模型进行文本检测，系统会根据用户选择的模型进行相应的处理。

5.5 本章小结

本章主要介绍了模型改写文本检测系统的设计与实现，包括需求分析、系统架构设计、模块和接口设计以及数据库设计。通过对系统的功能进行详细分析，明确了系统的业务需求和功能需求，为后续的系统实现提供了基础。系统采用前后端分离的架构设计，提高了系统的灵活性和可扩展性。通过模块和接口设计，明确了系统各个模块之间的关系和数据传输方式。数据库设计则为系统的数据存储提供了支持。最后，通过用户注册与登录、文本改写功能、模型改写文本检测功能等展示了系统的实际应

表 5.6　模型改写文本检测系统开发环境与工具

描述	配置
编程语言	HTML、CSS、JavaScript、Python
前端框架	Vue
服务端开发	Pytorch1.8、Transformers
开发工具	Vscode
数据库	MySQL
权限管理	JWT
服务器端	Flask1.1.2、Docker



(a) 用户登录　　(b) 用户注册

图 5.4　模型改写文本检测系统用户注册与登录

用效果。

本章为模型改写文本检测系统的设计与实现奠定了基础，为后续的系统测试和优化提供了依据。通过对系统的设计与实现，可以看出模型改写文本检测系统在实际应用中的可行性和有效性，为后续的研究和应用提供了参考。

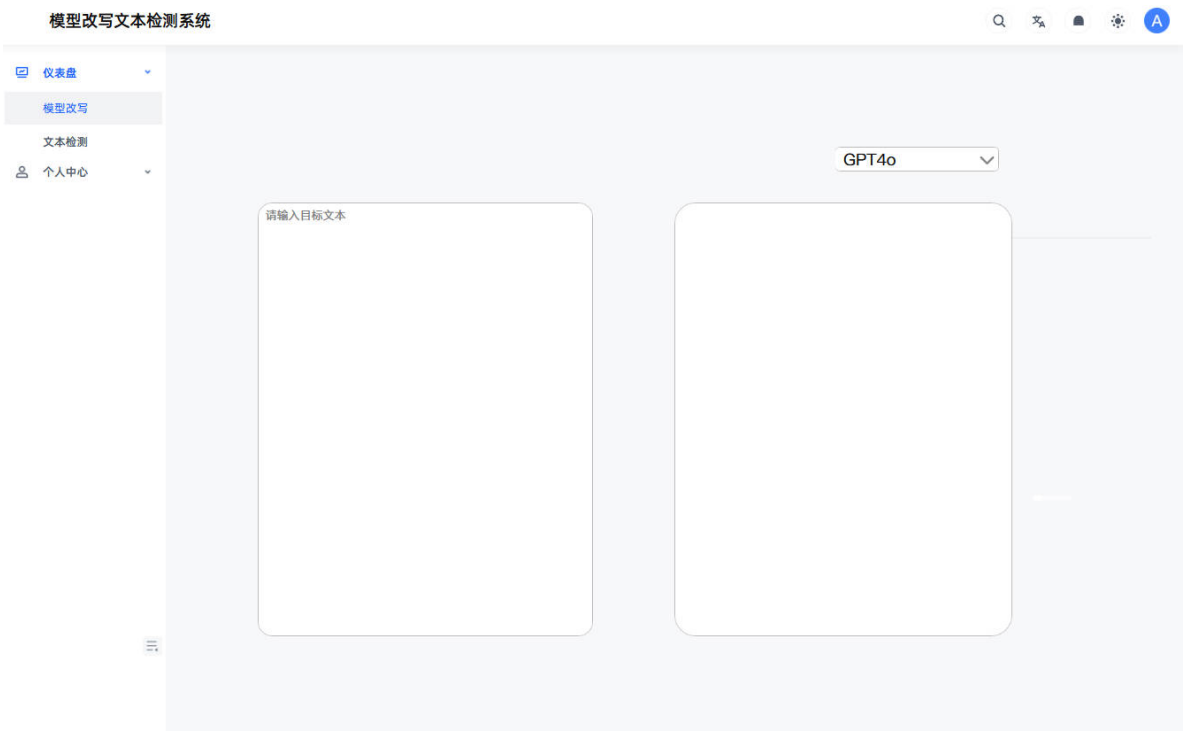


图 5.5　模型改写文本检测系统——文本改写功能

图 5.6　模型改写文本检测系统——模型改写文本检测功能

结论

本研究针对教育领域的文本溯源任务，提出了新的研究框架并创建了模型改写文本检测数据集。我们的实验结果表明，经过微调的 DeBERTa-V3-Large 模型在文档级任务上表现优异，显示出其在处理复杂文本结构方面的潜力。

然而，在句子级任务中，该模型的表现相对较差，揭示了这一领域仍然存在的挑战。本研究的贡献在于为文本溯源提供了新的数据集和基准，推动了对 AI 生成文本检测技术的理解。尽管如此，研究也存在一定的局限性，例如数据集规模和多样性可能影响模型的泛化能力。

本文搭建的模型改写文本检测系统为教育工作者和研究人员提供了一个有价值的工具，帮助他们识别和分析学生的写作风格和文本来源。通过对模型的进一步优化和改进，我们相信可以在未来实现更高效的文本溯源。

未来的研究可以集中在改进句子级文本溯源的算法，以及探索更大规模和多样化的数据集，以提升模型的整体性能和适应性。此外，结合其他技术（如图神经网络或迁移学习）可能为这一领域带来新的突破。本文搭建的系统目前仍存在一些不足之处，如功能过少、管理员仍需要从命令行操作用户等。未来的开发可以集中在改进系统的用户体验和功能扩展上，以满足更广泛的需求。

参考文献

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]. NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, California, USA: Curran Associates Inc., 2017: 6000-6010.

[2] Radford A, Narasimhan K. Improving Language Understanding by Generative Pre-Training[C]. 2018.

[3] Radford A, Wu J, Child R, et al. Language Models are Unsupervised Multitask Learners[C]. 2019.

[4] Brown T B, Mann B, Ryder N, et al. Language models are few-shot learners[C]. NIPS '20: Proceedings of the 34th International Conference on Neural Information Processing Systems. Vancouver, BC, Canada: Curran Associates Inc., 2020.

[5] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback [EB/OL]. 2022. https://arxiv.org/abs/2203.02155.

[6] OpenAI. GPT-4 Technical Report[EB/OL]. 2024. https://arxiv.org/abs/2303.08774.

[7] DeepSeek-AI. DeepSeek-V3 Technical Report[EB/OL]. 2024. https://arxiv.org/abs/2412.19437.

[8] Guo B, Zhang X, Wang Z, et al. How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection[EB/OL]. 2023. https://arxiv.org/abs/2301.07597.

[9] Wang R, Chen H, Zhou R, et al. LLM-Detector: Improving AI-Generated Chinese Text Detection with Open-Source LLM Instruction Tuning[EB/OL]. 2024. https://arxiv.org/abs/2402.01158.

[10] Wang P, Li L, Ren K, et al. SeqXGPT: Sentence-Level AI-Generated Text Detection[C]. Bouamor H, Pino J, Bali K. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Singapore: Association for Computational Linguistics, 2023: 1144-1156.

[11] Li L, Wang P, Ren K, et al. Origin Tracing and Detecting of LLMs[EB/OL]. 2023. https://arxiv.org/abs/2304.14072.

[12] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[C]. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics, 2019: 4171-4186.

[13] Lewis M, Liu Y, Goyal N, et al. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension[C]. Jurafsky D, Chai J, Schluter N, et al. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020: 7871-7880.

[14] 王晓峰. 基于 Transformer 模型的中文文本生成方法研究[J]. 无线互联科技, 2024, 21(20): 44-46.

[15] 李思慧, 蔡国永, 蒋航, 等. 一种新的基于凸损失函数的离散扩散文本生成模型[J]. 计算机科学, 1-12.

[16] 胡妍璐. 新闻生产的大数据技术应用与文本生成[J]. 新闻采编, 2024(06): 103-104.

[17] 张蔚琪. 基于文本匹配与文本生成的法律问答模型研究[D]. 四川大学, 2023.

[18] Google G T. Gemini: A Family of Highly Capable Multimodal Models[EB/OL]. 2024. https://arxiv.org/abs/2312.11805.

[19] Touvron H, Lavril T, Izacard G, et al. LLaMA: Open and Efficient Foundation Language Models [EB/OL]. 2023. https://arxiv.org/abs/2302.13971.

[20] Touvron H, Martin L, Stone K, et al. Llama 2: Open Foundation and Fine-Tuned Chat Models [EB/OL]. 2023. https://arxiv.org/abs/2307.09288.

[21] Grattafiori A, Dubey A, Jauhri A, et al. The Llama 3 Herd of Models[EB/OL]. 2024. https://arxiv.org/abs/2407.21783.

[22] Sun Y, Wang S, Li Y, et al. ERNIE: Enhanced Representation through Knowledge Integration [EB/OL]. 2019. https://arxiv.org/abs/1904.09223.

[23] Sun Y, Wang S, Li Y, et al. ERNIE 2.0: A Continual Pre-training Framework for Language Understanding[EB/OL]. 2019. https://arxiv.org/abs/1907.12412.

[24] Sun Y, Wang S, Feng S, et al. ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation[EB/OL]. 2021. https://arxiv.org/abs/2107.02137.

[25] Zeng W, Ren X, Su T, et al. PanGu-α: Large-scale Autoregressive Pretrained Chinese Language Models with Auto-parallel Computation[EB/OL]. 2021. https://arxiv.org/abs/2104.12369.

[26] Wang Y, Chen H, Tang Y, et al. PanGu-π: Enhancing Language Model Architectures via Nonlinearity Compensation[EB/OL]. 2023. https://arxiv.org/abs/2312.17276.

[27] Ren X, Zhou P, Meng X, et al. PanGu-Σ: Towards Trillion Parameter Language Model with Sparse Heterogeneous Computing[EB/OL]. 2023. https://arxiv.org/abs/2303.10845.

[28] Zeng A, Liu X, Du Z, et al. GLM-130B: An Open Bilingual Pre-trained Model[EB/OL]. 2023. https://arxiv.org/abs/2210.02414.

[29] Bai J, Bai S, Chu Y, et al. Qwen Technical Report[EB/OL]. 2023. https://arxiv.org/abs/2309.16609.

[30] Hou Z, Niu Y, Du Z, et al. ChatGLM-RLHF: Practices of Aligning Large Language Models with Human Feedback[EB/OL]. 2024. https://arxiv.org/abs/2404.00934.

[31] Kaufmann T, Weng P, Bengs V, et al. A Survey of Reinforcement Learning from Human Feedback [EB/OL]. 2024. https://arxiv.org/abs/2312.14925.

[32] GLM T, : Zeng A, et al. ChatGLM: A Family of Large Language Models from GLM-130B to GLM-4 All Tools[EB/OL]. 2024. https://arxiv.org/abs/2406.12793.

[33] Hendrycks D, Burns C, Basart S, et al. Measuring Massive Multitask Language Understanding

[EB/OL]. 2021. https://arxiv.org/abs/2009.03300.

[34] Cobbe K, Kosaraju V, Bavarian M, et al. Training Verifiers to Solve Math Word Problems[EB/OL]. 2021. https://arxiv.org/abs/2110.14168.

[35] Hendrycks D, Burns C, Kadavath S, et al. Measuring Mathematical Problem Solving With the MATH Dataset[EB/OL]. 2021. https://arxiv.org/abs/2103.03874.

[36] Suzgun M, Scales N, Schärli N, et al. Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them[EB/OL]. 2022. https://arxiv.org/abs/2210.09261.

[37] Rein D, Hou B L, Stickland A C, et al. GPQA: A Graduate-Level Google-Proof QA Benchmark [EB/OL]. 2023. https://arxiv.org/abs/2311.12022.

[38] Peng Q, Chai Y, Li X. HumanEval-XL: A Multilingual Code Generation Benchmark for Cross-lingual Natural Language Generalization[EB/OL]. 2024. https://arxiv.org/abs/2402.16694.

[39] Zhou J, Lu T, Mishra S, et al. Instruction-Following Evaluation for Large Language Models[EB/OL]. 2023. https://arxiv.org/abs/2311.07911.

[40] Wu S, Peng Z, Du X, et al. A Comparative Study on Reasoning Patterns of OpenAI's o1 Model [EB/OL]. 2024. https://arxiv.org/abs/2410.13639.

[41] Qwen, : Yang A, et al. Qwen2.5 Technical Report[EB/OL]. 2025. https://arxiv.org/abs/2412.15115.

[42] Bai S, Chen K, Liu X, et al. Qwen2.5-VL Technical Report[EB/OL]. 2025. https://arxiv.org/abs/2502.13923.

[43] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale[EB/OL]. 2021. https://arxiv.org/abs/2010.11929.

[44] Anthropic S. Model Card Addendum: Claude 3.5 Haiku and Upgraded Claude 3.5 Sonnet[C].

[45] DeepSeek-AI, : Bi X, et al. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism[EB/OL]. 2024. https://arxiv.org/abs/2401.02954.

[46] DeepSeek-AI, Liu A, Feng B, et al. DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model[EB/OL]. 2024. https://arxiv.org/abs/2405.04434.

[47] DeepSeek-AI, Guo D, Yang D, et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning[EB/OL]. 2025. https://arxiv.org/abs/2501.12948.

[48] Team Q. QwQ-32B: Embracing the Power of Reinforcement Learning[EB/OL]. 2025. https://qwenlm.github.io/blog/qwq-32b/.

[49] Maron M E. Automatic Indexing: An Experimental Inquiry[J]. J. ACM, 1961, 8: 404-417.

[50] Cover T M, Hart P E. Nearest neighbor pattern classification[J]. IEEE Trans. Inf. Theory, 1967, 13: 21-27.

[51] Joachims T. Text Categorization with Support Vector Machines: Learning with Many Relevant Features[C]. European Conference on Machine Learning. 1999.

[52] Breiman L. Random Forests[J]. Machine Learning, 2001, 45: 5-32.

[53] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System[J]. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[54] Ke G, Meng Q, Finley T, et al. LightGBM: A Highly Efficient Gradient Boosting Decision Tree[C]. Neural Information Processing Systems. 2017.

[55] 孔令怡, 杨钰, 孙敏, 等. 基于机器学习的学生社区网络平台事件言论分类研究[J]. 现代计算机, 2024, 30(21): 99-105.

[56] 赵锴, 叶丹. 基于机器学习的矿床描述文本多标签分类[J]. 中国矿业, 2024, 33(10): 153-161.

[57] Alsmadi M K, Omar K B, Noah S A M, et al. Performance Comparison of Multi-layer Perceptron (Back Propagation, Delta Rule and Perceptron) algorithms in Neural Networks[J]. 2009 IEEE International Advance Computing Conference, 2009: 296-299.

[58] Pouyanfar S, Sadiq S, Yan Y, et al. A Survey on Deep Learning: Algorithms, Techniques, and Applications[J]. ACM Comput. Surv., 2018, 51(5).

[59] Zhao W, Ye J, Yang M, et al. Investigating Capsule Networks with Dynamic Routing for Text Classification[J]. ArXiv, 2018, abs/1804.00538.

[60] Qin L, Che W, Li Y, et al. DCR-Net: A Deep Co-Interactive Relation Network for Joint Dialog Act Recognition and Sentiment Classification[J]. ArXiv, 2020, abs/2008.06914.

[61] Deng Z, Peng H, He D, et al. HTCInfoMax: A Global Model for Hierarchical Text Classification via Information Maximization[C]. North American Chapter of the Association for Computational Linguistics. 2021.

[62] Yao L, Mao C, Luo Y. Graph Convolutional Networks for Text Classification[J]. ArXiv, 2018, abs/1809.05679.

[63] Li C, Peng X, Peng H, et al. TextGTL: Graph-based Transductive Learning for Semi-supervised Text Classification via Structure-Sensitive Interpolation[C]. International Joint Conference on Artificial Intelligence. 2021.

[64] 黄瑞, 徐计. 基于不变图卷积神经网络的文本分类[J]. 计算机科学, 2024, 51(S1): 120-124.

[65] 孙雪松. 基于深度学习的网络情绪文本分类方法研究[D]. 西安理工大学, 2024.

[66] Peters M E, Neumann M, Iyyer M, et al. Deep Contextualized Word Representations[J]. ArXiv, 2018, abs/1802.05365.

[67] Greff K, Srivastava R K, Koutník J, et al. LSTM: A Search Space Odyssey[J]. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(10): 2222-2232.

[68] 陈晨, 石赫, 徐悦, 等. 基于 BERT-BiLSTM 的油田安全生产隐患文本分类[J]. 科学技术与工程, 2024, 24(29): 12650-12657.

[69] 周荣, 钱钢. 基于 BERT-BiGRU-CNN 的讽刺文本分类研究[J]. 智能计算机与应用, 1-6.

[70] 韩博, 成卫青. 基于 BERT 和标签混淆的文本分类模型[J]. 南京邮电大学学报 (自然科学版), 2024, 44(03): 100-108.

[71] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. J. Mach. Learn. Res., 2003, 3(null): 1137-1155.

[72] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space [EB/OL]. 2013. https://arxiv.org/abs/1301.3781.

[73] Yang Z, Dai Z, Yang Y, et al. XLNet: generalized autoregressive pretraining for language understanding[M]. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2019.

[74] Liu Y, Ott M, Goyal N, et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach [EB/OL]. 2019. https://arxiv.org/abs/1907.11692.

[75] He P, Gao J, Chen W. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing[EB/OL]. 2023. https://arxiv.org/abs/2111.09543.

[76] Raffel C, Shazeer N, Roberts A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. J. Mach. Learn. Res., 2020, 21(1).

[77] Rajpurkar P, Zhang J, Lopyrev K, et al. SQuAD: 100,000+ Questions for Machine Comprehension of Text[EB/OL]. 2016. https://arxiv.org/abs/1606.05250.

[78] Wang A, Singh A, Michael J, et al. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding[EB/OL]. 2019. https://arxiv.org/abs/1804.07461.

[79] Williams A, Nangia N, Bowman S R. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference[EB/OL]. 2018. https://arxiv.org/abs/1704.05426.

[80] Gehrmann S, Strobelt H, Rush A M. GLTR: Statistical Detection and Visualization of Generated Text [EB/OL]. 2019. https://arxiv.org/abs/1906.04043.

[81] Crossley S, Baffour P, King J, et al. Learning Agency Lab - Automated Essay Scoring 2.0[Z]. https://kaggle.com/competitions/learning-agency-lab-automated-essay-scoring-2. Kaggle. 2024.

[82] Cloud M. Sentence Splitter[EB/OL]. GitHub. 2017. https://github.com/mediacloud/sentence-splitter.

[83] OpenAI. GPT-4o System Card[EB/OL]. 2024. https://arxiv.org/abs/2410.21276.

[84] Team G. Gemini: A Family of Highly Capable Multimodal Models[EB/OL]. 2024. https://arxiv.org/abs/2312.11805.

[85] Su J, Lu Y, Pan S, et al. RoFormer: Enhanced Transformer with Rotary Position Embedding[EB/OL]. 2023. https://arxiv.org/abs/2104.09864.

[86] He P, Gao J, Chen W. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing[EB/OL]. 2023. https://arxiv.org/abs/2111.09543.

[87] Lample G, Conneau A. Cross-lingual Language Model Pretraining[J]. ArXiv, 2019, abs/1901.07291.

[88] Yang Z, Dai Z, Yang Y, et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding[C]. Neural Information Processing Systems. 2019.

[89] Joshi M, Chen D, Liu Y, et al. SpanBERT: Improving Pre-training by Representing and Predicting Spans[J]. Transactions of the Association for Computational Linguistics, 2019, 8: 64-77.

[90] Shibata Y, Kida T, Fukamachi S, et al. Byte Pair Encoding: A Text Compression Scheme That Accelerates Pattern Matching[J]. 1999.

[91] Shaw P, Uszkoreit J, Vaswani A. Self-Attention with Relative Position Representations[C]. North American Chapter of the Association for Computational Linguistics. 2018.

[92] Huang C Z A, Vaswani A, Uszkoreit J, et al. Music Transformer: Generating Music with Long-Term Structure[C]. International Conference on Learning Representations. 2018.

[93] Dai Z, Yang Z, Yang Y, et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context[C]. Annual Meeting of the Association for Computational Linguistics. 2019.

[94] Clark K, Luong M T, Le Q V, et al. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators[EB/OL]. 2020. https://arxiv.org/abs/2003.10555.

[95] Botev Z I, Kroese D P, Rubinstein R Y, et al. Chapter 3 - The Cross-Entropy Method for Optimization [G]. Rao C, Govindaraju V. Handbook of Statistics: Handbook of Statistics: vol. 31. Elsevier, 2013: 35-59.

[96] Loshchilov I, Hutter F. Decoupled Weight Decay Regularization[EB/OL]. 2019. https://arxiv.org/abs/1711.05101.

[97] Kingma D P, Ba J. Adam: A Method for Stochastic Optimization[EB/OL]. 2017. https://arxiv.org/abs/1412.6980.

[98] Loshchilov I, Hutter F. SGDR: Stochastic Gradient Descent with Warm Restarts[EB/OL]. 2017. https://arxiv.org/abs/1608.03983.

[99] Revelle W. Spearman, Charles (1863–1945)[M]. 2015.

[100] Benesty J, Chen J, Huang Y, et al. Pearson Correlation Coefficient[M]. Noise Reduction in Speech Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009: 1-4.

攻读学位期间发表论文与研究成果清单

[1] Yin H, **Lu P**, Li Z, Sun B, Li K. MODE: a multimodal open-domain dialogue dataset with explanation [J]. Appl. Intell., 2024, 54(7): 5891-5906.

致谢

恍恍惚惚，硕士研究生三年时光飞逝，我要毕业迈向工作了。从刚进入北京理工大学的战战兢兢，到如今即将离校的依依不舍，心中感慨万千。感谢母校的培养，感谢老师的教导，感谢同学的陪伴，感谢朋友的支持。

在此，我要特别感谢导师李侃老师和网络信息中心的屈少杰老师，是您们悉心的指导和无私的帮助，让我在研究中不断成长。李老师的严谨治学和屈老师的热情支持让我在科研道路上走得更加坚定。

感谢我的同门师兄弟姐妹们，感谢你们在我研究生生活中给予的支持和帮助。我们一起度过了无数个难忘的时光，分享了彼此的快乐与烦恼。你们的陪伴让我在这段旅程中不再孤单。

感谢我的家人，感谢你们一直以来的支持和理解。无论我身处何地，你们始终是我最坚实的后盾。你们的爱让我在追求梦想的道路上充满力量。