难度、指令需求的复杂程度等方面着手。考察点难度覆盖难、中、易三档；指令复杂程度体现在多约束条件、单 / 多指令、题目长度、文本类型、个性化需求等方面，还涉及单轮 / 多轮的轮数设置。

此外，进行多维度综合考察，涵盖指令理解 / 跟随、医学正确性、回答有效性、可读性、安全风险等。评测集指令丰富，每个能力项约 500 条指令，总计约 3000 条评测样本，每条指令都配有对应的参考答案及回答要点，以提高机评准确率。整体评测方式采用模型自动化评测与人工二次评测相结合。

12. 大语言模型应用开发

大语言模型的广泛应用正在推动技术创新与产业变革。自 2023 年以来，大语言模型在多个领域的应用开发取得了显著进展，包括智能客服、内容生成、教育辅助、医疗咨询、代码生成等场景。大语言模型凭借其强大的语言理解与生成能力，为开发者和企业提供了全新的工具和平台。然而，大语言模型的应用开发也面临诸多挑战，例如如何高效地部署和调用模型、如何定制化以满足特定业务需求以及如何应对生成内容的质量控制和潜在风险等等。因此，构建一套系统化的大语言模型应用开发流程与方法显得尤为重要。

本章将首先介绍大语言模型典型应用场景，并在此基础上根据典型应用介绍开发流程、开发工具与平台，最后介绍大语言模型本地部署实践。

12.1 大语言模型典型应用场景

本节将围绕大语言模型的典型应用场景展开探讨，重点介绍其在内容创作与生成、对话系统与聊天机器人、翻译与多语言处理、信息抽取与知识图谱等领域中的实际应用及其技术创新。同时，还将详细分析大语言模型在代码生成与编程辅助、智能搜索与推荐、教育与培训、企业管理与决策支持，以及法律与合规等行业中的广泛应用。通过对这些场景的全面阐述，旨在介绍大语言模型在推动各行业效率提升、创新发展中的核心作用，并为未来技术与产业的深度融合提供启示。

12.1.1 内容创作与生成

大语言模型在内容创作与生成领域展现出了强大的能力，能够显著提高内容创作的效率与质量。在文章写作方面，大语言模型可以自动生成新闻报道、博客文章和产品描述等内容。例如，OpenAI 的 ChatGPT 已被多家媒体和企业应用于文章初稿的生成，通过输入简单的主题或关键词，即可快速生成结构清晰、语言流畅的文本。这种能力帮助内容创作者节省了大量时间，提高了内容发布的效率，尤其适用于需要高频更新的新闻媒体和电商平台。

在故事创作方面，大语言模型能够根据用户提供的提示或情节大纲生成完整的故事情节，为创意写作提供了全新的方式。许多作家和创意团队使用 GPT-4 等模型，生成故事大纲和角色设定，

从而激发更多灵感。还有一些产品则是根据故事创作领域的核心需求，开发特定的大语言模型产品。例如，Sudowrite 能够根据用户的提示词和需求生成多种形式的文本内容，并提供包括润色、摘要、大纲生成等各类能力。当用户输入“魔法世界的冒险”这一主题时，模型可以生成相关的故事片段、对话场景或完整的大纲。同时，模型还支持内容续写功能，帮助用户延续未完成的小说或故事情节，保持语气一致性。这种辅助创作工具已经成为许多写作者的得力助手。

此外，大语言模型在诗歌与歌词创作中也有出色表现，通过对特定风格和主题的理解，生成具有艺术性和情感表达的诗歌或歌词。例如，谷歌的 Bard 模型能够根据用户输入的主题，创作出风格多样的诗歌，甚至模仿某些文学流派的语言特征。同样，音乐创作领域也开始广泛应用大语言模型，如歌词生成工具 LyricStudio，其通过大语言模型为音乐人提供多种主题和风格的歌词创作建议，具有智能建议与押韵功能，能为特定单词找到押韵，使歌词更流畅有韵律，显著降低了词曲创作的门槛。

总体来看，大语言模型正在重塑内容创作的方式，从新闻稿到文学创作，再到诗歌与歌词创作，赋能创意产业的多个环节。这不仅提高了创作者的生产效率，还为更多非专业人士提供了创作的可能性。

12.1.2 对话系统与聊天机器人

客服机器人是大语言模型最成熟且广泛应用的领域之一，能够为企业提供高效、智能的客户服务解决方案。传统客服机器人在面对复杂、模糊的客户提问时，常常理解偏差，答非所问。大语言模型凭借其强大的自然语言处理能力，可深入剖析客户语句含义，即使是隐喻、口语化表述，也能精准提取关键信息。在电商领域，熟知各类商品参数、使用方法、售后政策；在金融行业，对贷款流程、理财产品细则、金融法规等也能信手拈来。以保险客服场景为例，客户询问“我买的这款重疾险，在国外就医能理赔吗？特殊治疗手段，比如质子重离子治疗报销吗?”，大语言模型客服机器人可依据保险条款细则和过往理赔案例，给出全面且准确的解答。国内外大量厂家的客服系统都通过大语模型极大提升了整体体验。

传统虚拟助手（如 Siri、Google Assistant 和 Alexa）表现出了强大的语音识别和执行能力，但它们的核心架构主要基于任务导向的对话系统，主要功能集中在预定义的任务上，如设置闹钟、播放音乐、查询天气等。这种设计虽然高效，但在处理开放式对话或复杂的上下文理解时，其能力显得不足。但是基于大语言模型，虚拟助手可以更好的理解用户意图，完成更加复杂的能力。例如，荣耀手机 YOYO 助理，通过引入大语言模型，能够更精准地理解用户带有隐喻、口语化等复杂表述的意图，也可以理解类似“用小学生能听懂的方式解释量子力学”这样的问题，并生成趣味解读。还能很好地记住对话的上下文内容，在多轮对话中保持连贯和准确，根据前文内容针对性地回答后续问题。比如用户先询问“附近有哪些川菜馆”，接着问“哪家评价最高”，YOYO 智能体可以关联上下文，准确回答，甚至可以直接帮你电话餐馆直接进行预定。

在心理健康和情感支持领域，大语言模型同样展现了重要价值，尤其是在心理健康应用中充

当情感陪伴和心理疏导的角色。例如，Replika 是由美国 Luka 公司开发的一款人工智能聊天机器人应用，致力于为用户提供个性化的对话和情感支持体验。其功能包括学习用户的语言风格、兴趣爱好和情感反应，提供定制化对话体验；通过倾听和同理心回应，帮助缓解压力和焦虑；增强现实（AR）互动，让用户在现实环境中与虚拟形象进行交流；记忆功能则能记住用户的重要信息和喜好，增加互动的连贯性。此外，Replika 整合了情感管理工具，为用户提供情绪识别和心理健康建议，在娱乐与情感陪伴方面表现出色，同时也为用户提供心理健康支持。这种情感支持类的对话机器人正在为心理健康服务提供一种低成本、高可达性的解决方案。

此外，大语言模型驱动的对话系统在医疗、教育等专业领域也展现了巨大的潜力。例如，微软推出的“Azure AI Health Bot”能够解答用户关于常见疾病的疑问，帮助他们初步判断病情并推荐适当的医疗资源。在教育领域，Duolingo 等语言学习应用通过大语言模型开发的对话功能，为用户提供更自然的互动体验，帮助他们有效提升语言学习能力。

12.1.3 翻译与多语言处理

随着大语言模型的崛起，这一领域正在迎来新的变革，大语言模型凭借其强大的语言理解和生成能力，为翻译与多语言处理注入了新的活力，加速了技术和应用的迭代发展。

在机器翻译方面，传统方法主要依赖神经网络、深度学习以及大量语料库的训练来实现文本翻译。然而，大语言模型的出现，为机器翻译带来了质的飞跃。得益于广泛的知识储备和对语言深层语义的理解能力，大语言模型在翻译中表现出更高的准确性和自然性，尤其是在文化背景、隐喻和典故等复杂内容的处理上。例如，在文学翻译场景中，传统机器翻译往往难以还原原文的意境与风格，而大语言模型能够更精准地理解文化元素，并以目标语言重现文本的艺术性。在跨国企业中，大语言模型也广泛应用于产品文档翻译，如苹果、三星等公司利用其快速处理多种专业术语，确保翻译的专业性和一致性，大幅提升了翻译效率，并帮助全球用户更好地理解产品信息。

在跨语言信息检索领域，大语言模型同样展现了强大的能力。以微软学术搜索等平台为例，引入大语言模型后，跨语言检索的精准度和效率显著提升。大语言模型能够深入理解用户提问的语义，即便面对模糊或复杂的问题，也能准确解析，并在多语言数据集中找到相关内容。例如，科研人员在研究人工智能领域时，可通过中文输入问题，模型不仅能理解核心要点，还可以在英文、法文或其他语言撰写的学术论文中精准定位相关信息。这种能力让科研人员能够全面获取全球研究成果，掌握前沿动态，进而推动科研项目的顺利开展。Open AI 2025 年 2 月推出的 Deep Research 则是更一步，基于 o3 模型，专为复杂研究任务设计，能自动搜索、解读、整合海量在线信息，花费 5 到 30 分钟生成专业级研究报告。它具备推理能力，可自主调整研究方向，研究结果附带完整文档、引用来源和逻辑摘要，适用于金融、科学等领域专业人士及有深度调研需求的消费者。

在多语言客户服务方面，大语言模型为企业提供了更智能化的服务解决方案。例如，在线旅游平台 Booking.com 利用大语言模型驱动的智能客服系统，结合实时翻译技术，为全球用户提供个性化、多语言支持。当一位日本游客使用日语在平台上预订法国巴黎的民宿时，提出关于景点、

交通等问题，大语言模型不仅能够准确理解用户需求，还能将答案翻译成日语，提供自然且贴合实际的建议。相比传统客服，这种基于大语言模型的解决方案更具人性化和情景适应性，大幅提升了用户体验，同时增强了用户对平台的信任与忠诚度。

翻译与多语言处理技术正以前所未有的速度融入各个行业，而大语言模型的应用为其带来了新的可能性。未来，随着大语言模型的持续优化，翻译的准确性和多语言处理的效率将进一步提升。这些技术不仅能够在更多领域创造价值，还将拉近不同文化与语言之间的距离，推动全球交流与合作，让世界变得更加紧密相连。

12.1.4 信息抽取与知识图谱

大语言模型在信息抽取领域展现了强大的能力，尤其是在实体识别任务中。借助其深度语言理解能力，大语言模型能够精准地从文本中提取出人名、地名、组织名等关键实体。例如，复旦大学推出的 B^2NE[628] 基于大模型的开放领域信息工具，可以让用户自由的从超过 16 个领域的 400 种类型中，灵活抽取目标实体和关系。在医疗领域，IBM Watson Discovery 广泛用于从医学文献中识别疾病名称、药物名称和治疗方法，从而支持医学研究和临床决策。在金融领域，Bloomberg 使用自研的 GPT 模型 BloombergGPT，帮助从新闻和公告中快速提取公司名称、事件类型（如并购、破产等）和时间节点，为金融分析师提供精准的实时信息。

在关系抽取方面，大语言模型能够识别文本中实体之间的语义关系，并通过语境理解隐含的关联。例如，Google Cloud Natural Language API 提供了强大的关系抽取功能，在法律领域可以从合同中识别合同双方的权利和义务关系；在金融服务中，大语言模型也可以用来从公告和新闻中提取公司并购、股权交易、合作伙伴关系等信息。这些应用帮助企业不仅快速获取结构化信息，还能通过分析实体间的关系，发现隐藏的业务机会或潜在风险。相比传统的基于规则的关系抽取方法，这类大语言模型驱动的产品在处理非结构化文本和复杂语境时表现更精准。

知识图谱构建是信息抽取的重要应用场景，而大语言模型通过其强大的语义理解能力，显著提升了知识图谱构建的效率与规模。例如，Microsoft Azure 的 Knowledge Mining 服务能够利用大语言模型从海量文档中自动提取实体和关系，并更新企业知识图谱。在金融领域，Kensho 使用自然语言处理技术从新闻报道、财务公告中自动提取关键信息，为金融企业构建实时更新的知识图谱。通过这些系统，企业可以轻松追踪市场动态、行业趋势，并快速构建跨领域的知识图谱。

未来，大语言模型在信息抽取与知识图谱领域的应用前景非常广阔。随着技术的进步，这些应用将更加智能化。企业可以通过这些工具快速构建多语言知识图谱，整合全球范围内的资源和信息。此外，像 LinkedIn 公司的 Economic Graph 这样的知识图谱服务，也可能进一步结合大语言模型的能力，帮助企业和个人更高效地管理职业网络和商业生态。

12.1.5 代码生成与编程辅助

大语言模型在辅助编程领域展现了显著的优势，极大地提升了开发效率。例如，GitHub Copilot 是由 OpenAI 和 GitHub 联合推出的一款智能编程助手，能够在开发环境中根据上下文为开发者提

供智能代码补全和建议。当开发者编写函数或算法时，GitHub Copilot 可以预测后续代码，并补全常见的代码片段，如循环、条件语句或函数调用。这种代码补全能力不仅减少了手动输入的工作量，还帮助开发者快速实现复杂的功能，尤其在处理冗长的标准库调用或框架代码时尤为高效。

Cursor 也是一款广受好评的 AI 代码编辑辅助工具，旨在大幅提升开发者的工作效率，深受 Shopify、OpenAI 等众多知名企业工程师的信赖。它能通过自然语言指令编写代码，例如开发者只需简单输入指令，就能快速更新整个类或函数，还能依据代码库提供答案，引用文件或文档内容，一键使用模型生成的代码。Cursor 的智能代码补全功能强大，能根据开发者的操作预测所需代码，约 25% 的情况下可精准预判，开发者按下“Tab”键即可完成输入，仿佛能以思维的速度进行编码。而且它使用起来十分便捷，可一键导入所有扩展、主题和快捷键绑定，让开发者快速上手；若开启隐私模式，代码不会被远程存储，保障了数据安全。

在调试与优化领域，大语言模型为开发者提供了强大的支持，帮助快速发现和修复代码中的错误。例如，Snyk 开发的 DeepCode 利用大模型技术扫描代码库，识别潜在的安全漏洞、性能问题和代码错误，并提供优化建议。类似地，Kite 是一款编程辅助工具，能够实时监测开发者的代码并指出可能的语法错误或逻辑问题，同时给出修复建议。此外，大语言模型还被集成到在线编程教育平台中，如 LeetCode AI 和 HackerRank CodePair，为学生和面试者提供自动化调试支持，生成示例代码并解释代码中的关键逻辑。这种能力不仅在教育领域具有重要意义，还能帮助初学者更快掌握编程技能。

未来，大语言模型在代码生成与编程辅助中的应用将更加广泛和深入。例如，OpenAI 的 Codex 模型已经为 GitHub Copilot 提供了核心支持，未来有望进一步扩展到更多开发工具中，为企业和个人开发者提供更强大的编程能力。同时，集成大语言模型的 IDE（集成开发环境）如 Visual Studio Code 和 JetBrains IntelliJ IDEA，正在逐步成为智能编程助手的主要载体。通过这些工具，开发者不仅能获得即时的代码生成与优化支持，还可以利用 AI 自动化完成测试、文档生成和代码重构等高难度任务，从而大幅提升软件开发的效率和质量。

12.1.6 智能搜索与推荐

大语言模型与搜索的结合是其最重要的应用之一，覆盖了非常广泛的场景。通过强大的语义理解和上下文分析能力，大语言模型可以帮助搜索引擎精准捕捉用户意图，提供更相关的搜索结果。这种结合在电子商务、知识管理、在线教育、医疗健康等领域展现了巨大潜力，不仅提升了搜索的智能化水平，还显著改善了用户体验，成为大模型应用的核心方向之一。

目前，几乎全部大语言模型公司推出的在线服务都引入了搜索增强功能，以提升问答精准度。2023 年 10 月之暗面推出 Kimi 智能搜索产品，2024 年，OpenAI 推出了 SearchGPT，结合大语言模型的语义理解与实时搜索能力，为用户提供更精确、即时的查询结果。这种结合在知识问答、技术支持和内容生成等场景中表现突出。类似地，微软的 Bing Chat 集成了 OpenAI 的接口，支持实时互联网搜索与智能问答，并已被嵌入到 Edge 浏览器和 Microsoft Office 365 Copilot 中。谷歌

的 Bard 也整合了搜索引擎功能，能够在提供答案的同时引用实时数据来源。此外，电子商务平台如 Amazon 和 eBay 也通过集成大语言模型改进了搜索功能，使其能够理解模糊查询或长尾关键词（如“适合冬季使用的防水登山鞋”），从而为用户提供更精准的商品推荐，提升购物体验。

在个性化推荐方面，大语言模型通过处理用户的历史行为和偏好数据，生成高度相关的内容推荐。例如，Netflix 使用基于深度学习的推荐系统结合大语言模型，分析用户观看历史和兴趣标签，为用户推荐符合其偏好的电影或电视剧。同样，Spotify 通过大语言模型理解用户的音乐播放记录和情绪偏好，生成个性化的歌单（如“每日推荐”或“心情歌单”）。新闻聚合应用如 Flipboard 和 Google News 也使用大语言模型分析用户的阅读习惯，并推荐符合其兴趣领域的新闻文章，例如科技爱好者会收到关于人工智能、机器人等领域的最新动态。这种个性化推荐不仅提高了用户的参与度，还优化了平台的内容分发效率。

大语言模型还被广泛应用于改进搜索与推荐的多模态能力，即结合文本、图像、音频等多种数据类型提供更丰富的结果。例如，YouTube 利用大语言模型结合视频内容的描述信息和用户观看行为，推荐相关视频。用户搜索“如何学习编程”时，模型不仅会推荐编程教学视频，还会根据用户的语言偏好、学习进度推荐对应的教程系列。电商平台淘宝、京东等智能搜索引擎同样集成了图像搜索功能，用户通过上传图片（如衣服样式）即可获得相似商品推荐。

大语言模型在智能搜索与推荐中的应用将进一步扩展。例如，ChatGPT 模型已经被集成到 Notion AI 和 Zapier AI 等工具中，帮助用户快速搜索和推荐相关信息，使知识管理更加高效。此外，企业工具如 LinkedIn 的推荐系统借助大语言模型优化了职位推荐和人脉搜索功能，根据用户的职业背景和兴趣推荐相关的求职机会或潜在合作伙伴。

12.1.7 教育与培训

大语言模型在在线辅导领域有着广泛的应用，通过强大的语义理解和自然语言生成能力，为学生提供个性化的学习支持。例如，Khan Academy 推出的虚拟导师 Khanmigo 能够帮助学生解答各种学科问题，指导他们完成作业，并根据学习进度提供实时建议。这种智能化的辅导方式，不仅提高了学生的学习效率，还缓解了家长和教师在辅导方面的压力。在国内，类似的应用也很普遍，像作业帮和学而思网校利用 AI 技术实现了智能答疑，学生只需拍照或输入问题，系统便能快速分析并生成详细解答，极大地方便了学习过程。

在课程设计方面，大语言模型能够协助教师自动生成教学计划和课程内容，显著减轻了备课负担。例如，微软的 Copilot for Education 可以根据教学目标和学生需求，生成详细的课程大纲、学习资源以及课堂活动建议，帮助教师高效组织教学内容。国内的科大讯飞智慧课堂也整合了类似的 AI 功能，支持教育机构快速设计课程内容，提供多样化的学习路径，并根据学生的反馈动态调整课程结构。这种工具不仅提升了教学效率，还改善了课程的针对性和灵活性，为教育工作者提供了强大的技术支持。

在考试评估场景中，大语言模型显示了极高的自动化能力，尤其是在作业批改和考试反馈方

面。例如，亚马逊的 AWS Educate 平台在编程教育中可以对学生提交的代码作业进行自动评估，提供错误分析和优化建议。在国内，科大讯飞 AI 学习机也广泛应用了 AI 自动批改技术，能够对主观题、作文等复杂题型进行语义分析，生成详细的评分报告，并给出具体的改进建议。这种技术的应用不仅提高了评估效率，也让学生能够更清晰地了解自己的学习薄弱点，从而更有针对性地改进。

此外，大语言模型在教师辅助方面的应用也日益广泛。例如，谷歌的 Google Classroom 利用 AI 帮助教师整理学生学习数据，生成进度报告，并提供个性化的教学策略建议。这种技术使教师可以用更少的时间获取更深刻的学生洞察，从而优化教学方法。在国内，钉钉的智能备课平台也通过 AI 技术支持智能备课，帮助教师快速生成教学材料和课堂内容，并根据不同学生的学习情况调整教学策略。这些功能大大提升了教师的工作效率，使他们能够专注于更有价值的教学活动。

12.1.8 企业管理和决策支持

大语言模型在企业管理和决策支持中表现出了很大的使用前景，能够从大量非结构化文本数据中快速提取关键信息，帮助企业科学制定决策。例如，微软的 Power BI 已结合大语言模型技术，允许用户通过自然语言输入查询，从而快速生成关键业务指标的分析结果。这使得非技术人员也能轻松完成复杂的数据分析任务。类似地，国内的阿里云 Quick BI 通过集成智能分析功能，能够挖掘出隐藏在复杂数据中的趋势、风险点和改进建议，并以图表或文本的形式输出，为企业提供实时的决策支持。这种技术的应用不仅简化了数据分析流程，还提升了分析的效率和精准度，帮助企业更快适应市场变化。

在报告生成方面，大语言模型的应用极大地方便了企业日常运营中的信息处理需求。例如，Tableau GPT 利用自然语言生成功能，能够根据输入的业务数据自动生成可视化的分析报告，包括销售趋势图、客户细分报告等，帮助管理者快速掌握业务状况并制定相应的策略。国内的金蝶云和用友 U8 等企业管理工具也开始引入大语言模型技术，支持财务报表、预算报告等的自动生成，甚至可以根据具体数据生成解释性文字，为用户提供清晰直观的业务洞察。这些工具不仅提高了报告生成的效率，还能够减少人工操作中的错误几率，为企业管理者节省时间和精力。

会议记录与摘要是大语言模型在企业管理中的另一重要应用场景。例如，Otter.AI 结合语音识别和自然语言处理技术，能够实时记录会议内容并生成简洁的摘要，方便参会者快速回顾会议要点，或者让未参会人员轻松了解关键内容。在国内，腾讯会议和飞书会议等工具也集成了类似功能，支持会议内容的自动转录和要点提取，并且可以生成后续任务清单或行动计划。这种技术不仅降低了手动记录的时间成本，还保证了记录内容的完整性和准确性，同时提高了会议的整体效率和后续工作的执行力。

此外，大语言模型在战略规划和管理优化方面也提供了强有力的支持。例如，IBM Watson 可以通过分析企业的运营数据和行业趋势，生成优化建议并协助制定未来的策略规划。在国内，华为云 EI 企业智能提供了从运营监控到战略规划的全流程支持，帮助企业识别潜在的市场机会、优化

资源配置，并发现运营中的瓶颈。这些技术的应用让企业能够在激烈的市场竞争中快速调整方向，占据市场优势，同时也为管理层提供了数据驱动的决策依据，显著提升了管理效率和执行效果。

12.1.9 法律与合规

大语言模型在合同审查方面展现了极大的应用潜力，能够自动识别合同条款中的潜在法律风险，显著提升审查效率。例如，Kira Systems 是一款基于大语言模型的合同审查工具，能够快速分析合同内容，标记关键条款，并指出可能存在的问题。这款工具已被众多律师事务所和企业采用，用于高效处理大量复杂的商业合同。在国内，类似的工具如“法大大”合同助手，通过大语言模型技术，支持对合同条款进行逐条审查，自动识别潜在的法律风险点，如不平等条款或隐藏的违约责任，从而帮助律师和企业快速发现问题并优化合同内容。

在法律业务管理和协作领域，大语言模型通过优化工具帮助企业确保其政策和流程符合法律法规。例如，HighQ 是 Thomson Reuters 推出的一款先进的法律业务管理和协作软件，具备案件管理、合同生命周期管理、法务工作受理、文档自动化以及安全云端协作等全面功能。它可以集中管理案件文档，自动分配任务并跟踪案件进度，从而确保案件处理的高效性和有序性。同时，利用人工智能技术对合同进行智能起草、审核和风险分析，实现从合同生成到续约的全流程自动化管理。其标准化的法务受理系统简化了需求提交和处理流程，而文档自动化功能则通过智能模板和填充技术快速生成法律文书，不仅减少了重复性劳动，还显著提升了文档的准确性和一致性。

在法律文书生成和审查方面，大语言模型的自然语言生成能力已经被广泛应用。例如，LawGeex 是一款智能合同审查平台，通过高效、精准的技术支持合同管理全流程。其核心功能包括合同自动化审查分析，利用深度学习算法快速扫描合同全文，精准识别遗漏、错误、歧义及潜在风险，实现审查精度接近或超越人类专家；个性化审查方案则根据用户需求定制审查策略，有针对性地优化合同质量。通过智能识别复杂条款中的隐性风险，显著降低合同纠纷可能；同时，通过节省高达 90% 的审查成本，实现高投资回报率。

大语言模型还在法律研究和案情分析方面发挥了重要作用。例如，Casetext 是一款法律研究工具，结合 AI 和语义搜索功能，可以快速从庞大的判例库中找到相关案例，并生成简洁的法律分析摘要。在国内，MetaLaw 等平台也通过大语言模型技术，为律师提供快速的案例检索和法律依据分析服务。这些工具不仅加快了研究速度，还为法律从业者提供了更系统、更全面的支持，使他们能够更高效地准备案件材料并提供法律咨询服务。

12.2 大语言模型应用开发案例

大语言模型的价值只有在具体场景中才能得到充分体现。无论是智能客服、内容创作、代码生成，还是医疗诊断和科研辅助，大语言模型的能力都需要与实际需求和应用场景相结合，才能真正为人们提供有效的支持。通过针对不同领域的任务进行定制化开发、优化，甚至专门的模型训练，可以为企业和个人带来高效、智能的解决方案。

本节将以浏览器智能插件和个人智能助理的开发场景为例，展示大语言模型在实际应用中的开发案例。

12.2.1 浏览器智能插件

在日常浏览网页的过程中，常常面对信息量过大、语言不通或多媒体内容难以理解的情况，因此自动摘要、网页翻译和视频翻译等功能显得尤为重要。自动摘要可以帮助我们快速提取网页的核心内容，避免浪费时间在冗长的信息中；网页翻译能够打破语言障碍，让我们轻松访问不同语言的内容资源；而视频翻译则能帮助我们理解非母语的视频信息，提升学习和获取知识的效率。针对这些痛点，可以将大语言模型与浏览器插件相结合，满足人们在高效获取、多语言理解和多媒体学习上的实际需求，使浏览体验更加便捷和智能。

FisherAI 开源项目[①]提供一款专为提升学习效率而设计的智能 Chrome 插件，它结合了大语言模型和多功能工具，为用户提供了高效便捷的使用体验。通过一键操作，FisherAI 支持多种实用功能，包括自动摘要、网页翻译、视频翻译、多轮对话以及工具箱等。这些功能帮助用户快速提取信息、跨越语言障碍，并高效处理复杂的学习和工作任务。

FisherAI 它支持多种大语言模型，包括 ChatGPT、Gemini、DeepSeek、Qwen、Mistral、Groq 等主流模型，也可以通过 Ollama 调用本地模型，让用户能够根据需求选择最适合的工具。同时，FisherAI 允许用户自定义模型配置、API 密钥和代理地址，从而满足个性化和多样化的使用场景。FisherAI 还内置丰富的快捷工具。例如，它支持划词翻译、以及通过输入“/”触发快捷功能，包括翻译、摘要、等操作。如图12.1所示，通过插件可以对网页内容进行总结和翻译。

① 可以通过 GitHub 搜索 FisherAI 查找项目，并获取源代码

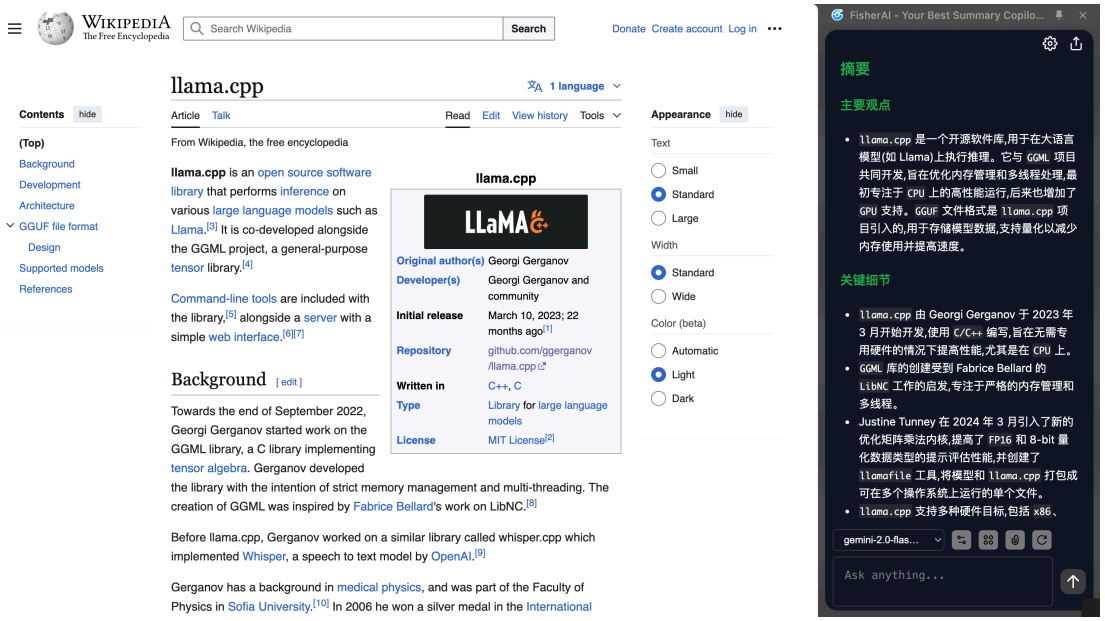

图 12.1　FisherAI 网页全文摘要展示

针对网页中包含的各种语言内容，FisherAI 提供了便捷的划词翻译功能。用户只需选中网页中的任意文字或段落，即可快速获得翻译结果，无需额外复制粘贴或切换页面。这种实时翻译的能力，不仅适用于简单的单词或短语，还能高效处理较长的句子和复杂语境下的文本，极大地提高了多语言网页浏览的效率，如图12.2所示。

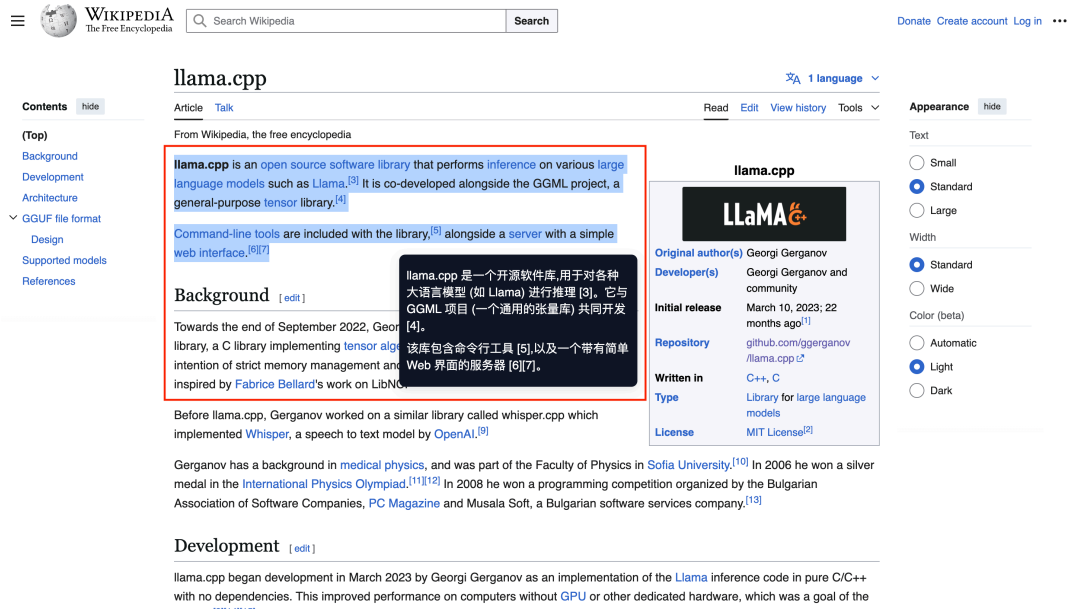

图 12.2 FisherAI 网页划词翻译展示

通过如下 JavaScript 脚本可以获取选中区域文本内容，并开启翻译。主要核心逻辑是监听鼠标点击事件，mouseup 事件监听：当用户释放鼠标按钮时，检查是否有文本被选中。如果有选中，显示按钮并定位到选中文本的位置。

```
// 监听选中事件
 document.addEventListener('mouseup', function (event) {
   const selection = window.getSelection();
   const selectedText = selection.toString().trim();

   // 当用户选中了文本
   if (selectedText) {
     const rects = selection.getRangeAt(0).getClientRects();
     if (rects.length > 0) {
       const rect = rects[0];
       button.style.top = `${rect.bottom + window.scrollY + 10}px`;
       button.style.left = `${rect.left + window.scrollX + 10}px`;
       button.style.display = 'block';
     }
   } else {
     // 没有选中文本，隐藏按钮和弹窗
     button.style.display = 'none';
     translationPopup.style.display = 'none';
   }
 });
```

当用户点击翻译按钮（button）时，会获取用户选中的文本内容，并通过大语言模型的接口（chatWithLLM）将选中的文本翻译成中文，然后将翻译结果显示在页面上的一个弹出框（translationPopup）中。

```
// 监听按钮点击事件
    button.addEventListener('click', function () {
      chrome.storage.sync.get([QUICK_TRANS], async function(config) {
        translationPopup.innerHTML = '';
        const selection = window.getSelection();
        const range = selection.getRangeAt(0);
        const rects = range.getClientRects();

        ....

        // 设置翻译结果弹出框的位置和显示状态
        translationPopup.style.top = `${topY}px`;
        translationPopup.style.left = `${middleX + window.scrollX}px`;

        translationPopup.style.display = 'block';
        button.style.display = 'none';

        const selectedText = window.getSelection().toString().trim();
        if (selectedText == '') {
          return;
        }

        try {
          let model = config[QUICK_TRANS].selectedModel;
          if (!model) {
            return;
          }
          const baseUrl, apiKey = await getBaseUrlAndApiKey(model);
          if(model.includes(FISHERAI_MODEL) || model.includes(OLLAMA_MODEL)) {
            chatWithLLM(model, TRANSLATE2CHN_PROMPT + selectedText, null, HUACI_TRANS_TYPE);
          }else if(baseUrl && apiKey) {
            chatWithLLM(model, TRANSLATE2CHN_PROMPT + selectedText, null, HUACI_TRANS_TYPE);
          } else {
            translationPopup.innerHTML = DEFAULT_TIPS;
          }
        } catch (error) {
          console.error('Error retrieving model or API information:', error);
          translationPopup.innerHTML = DEFAULT_TIPS;
        }

        translationPopup.style.display = 'block';
        button.style.display = 'none';
      });
```

完整代码内容可以参考“FisherAI/scripts/content.js”，大语言模型调用代码参考“FisherAI/

scripts/llm.js”。

12.2.2 论文搜索助理

学术研究的基石在于文献检索，这一过程极为复杂且富有挑战性。研究者不仅需要掌握各领域的专业知识，还需通晓各类综述性文章，并具备处理高精密度检索任务的能力。例如，关于“基于 UCB 算法的非平稳强化学习中价值导向研究”，这类专业性极强的检索需求，普通搜索引擎如谷歌学术往往难以完全满足[629]。

学术研究者们在进行文献调研时往往会面临巨大的工作量压力。当前，大语言模型技术为科研工作者提供了新的解决方案，尤其是在优化检索效果方面展现出独特优势。但学术研究远不止于机械的信息获取，更需要研究者深入理解各篇文献的核心观点并建立完整的知识体系。鉴于此，打造一款兼具深度分析与智能辅助功能的研究助手显得尤为必要。这不仅能大大节省研究者的时间投入，还能确保学术检索过程的专业性与可靠性。

PaSa（Paper Search）[629] 就是一款由大语言模型驱动的高级论文查找助理，旨在为复杂的学术问题提供全面且准确的结果。PaSa 能够自主完成一系列决策，包括调用搜索工具、阅读论文以及选择相关引用，从而高效地满足用户的学术需求。通过使用包含 35,000 个精细学术查询和对应论文的合成数据集 AutoScholarQuery，PaSa 应用强化学习进行了优化。此外，团队还构建了 RealScholarQuery，一个基于真实学术查询的基准数据集，用于评估 PaSa 在现实场景中的表现。尽管 PaSa 主要基于合成数据训练，但其在 RealScholarQuery 基准测试上的表现显著优于现有方法，包括 Google、Google Scholar、使用 GPT-4 改写查询的 Google、支持搜索功能的 ChatGPT 和 GPT-o1。

PaSa 的系统框架如图12.3所示。PaSa 系统由两个大语言模型 Agent 组成：爬取器（Crawler）和选择器（Selector），协同工作以实现高效的学术论文检索与筛选。系统在接获检索请求后，将激活 Crawler 模块。该模块可自主调用检索系统或从原文文献中获取引用信息，继而动态获取并纳入待处理文献库。随后，Crawler 模块将对文献库中的每一篇内容进行循环处理，通过追踪引文关系链，持续发现更为契合检索要求的学术资料，最终构建起一个内容丰富的文献体系。Selector 负责对论文队列中的每篇论文进行仔细阅读和评估，以判断其是否符合用户查询的具体需求。PaSa 框架采用了强化学习框架 AGILE[630] 进行优化，从而提升了大语言模型 Agent 在复杂任务中的决策能力。通过 Crawler 和 Selector 的高效协作，PaSa 不仅能够自动化地完成复杂的文献检索，还能确保结果的精准性和全面性，为学术研究者提供强有力的支持。

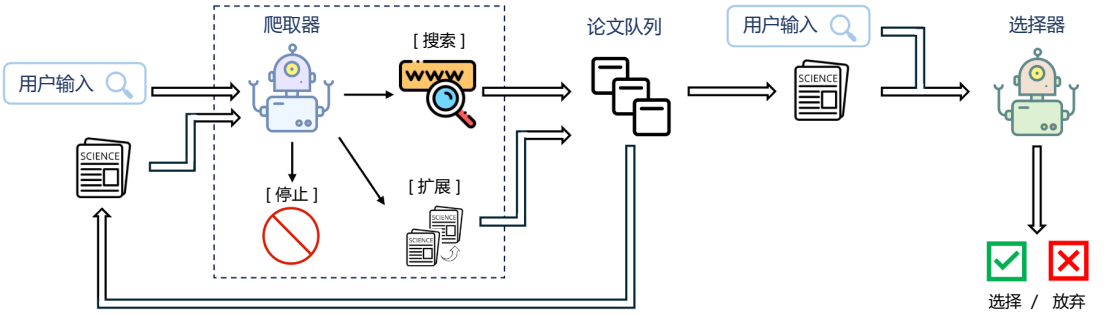

图 12.3 PaSa 系统框架[629]

Github 上搜索 bytedance PaSa 可以获得代码、数据、模型等。数据可以通过 Hugging Face 平台获取 PaSa-dataset 并存入 data 文件夹下。下载模型 PaSa-7b-crawler 和 PaSa-7b-selector 并保存到 checkpoints 文件夹下。可以通过以下命令开启 PaSa：

```
git clone git@github.com:hyc2026/transformers.git
cd transformers
pip install -e .
cd ..
pip install -r requirements.txt
python run_paper_agent.py
```

运行时需要首先获取 Google Search API 的访问凭证，并在在 utils.py 文件设置。Crawler 首先分析用户提交的提问，继而筛选出论文中的主要分支进行选择扩展。随后 Selector 基于文章的概况对信息进行量化打分，衡量其与提问的契合程度。系统会通过调用 Google 搜索引擎和 arxiv/ar5iv 搜索 API，完成信息的检索与完整论文的获取。

12.3 大语言模型本地部署实践

本地部署大语言模型的实践具有重要意义，不仅能够提升数据隐私和安全性，避免敏感信息在云端传输的风险，还能降低对网络连接的依赖，实现离线环境下的高效应用，同时在成本控制和定制化部署方面具备显著优势。

大语言模型的推理过程通常需要大量计算资源，因此依赖于硬件加速设备，例如 GPU 和 NPU 等。为了适配多种硬件环境，需要构建能够高效运行的大型语言模型框架。llama.cpp 是一个用纯 C/C++ 实现的大语言模型推理项目，其主要功能是为用户提供跨硬件的高效推理能力。与此同时，近年来涌现了大量开源的大语言模型，为了方便普通用户使用，还需要提供更友好的管理工具，例

如 Ollama，它基于 llama.cpp，具备简洁的安装和使用流程。此外，考虑到普通用户通常不会直接操作控制台界面，还需要开发支持 Web 界面和应用界面的解决方案。Open Webui 就是一个旨在提供类似 ChatGPT 界面的工具，方便用户与模型交互。本地部署大语言模型的整体架构如图 12.4 所示。

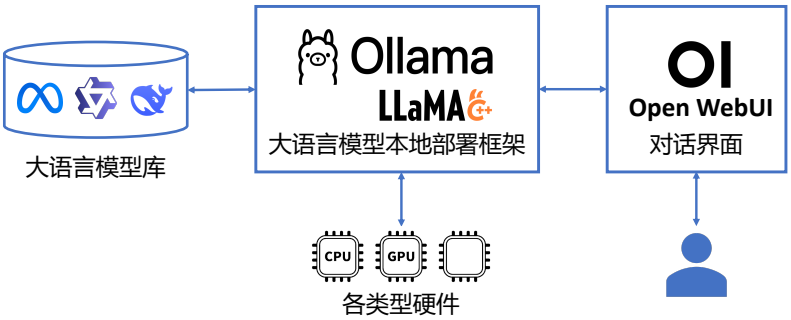

图 12.4　大语言模型本地部署系统结构图

本节将首先介绍大语言模型本地部署的核心工具 llama.cpp，在此基础上介绍本地部署工具 Ollama，最后介绍大语言模型网页交互工具 Open WebUI。

12.3.1 llama.cpp

llama.cpp 是一个用纯 C/C++ 实现的大语言模型（LLM）推理项目，旨在以最小的设置和高性能支持 LLaMA 及其他模型的本地运行。该项目的目标是让用户能够在各种硬件（包括本地设备和云端）上高效运行大型语言模型，同时优化对资源的使用。llama.cpp 支持多种硬件架构，包括 Apple Silicon（通过 ARM NEON、Accelerate 和 Metal 框架优化）、x86 架构（支持 AVX、AVX2、AVX512 和 AMX 指令集）以及 NVIDIA 和 AMD GPU（通过 CUDA 和 HIP 实现）。此外，它还提供多种量化技术（例如 1.5-bit 到 8-bit），以减少内存使用并加快推理速度。

llama.cpp 的主要优势在于其跨平台兼容性和灵活性。它不仅支持在 CPU 和 GPU 之间的混合推理，使得即使在显存不足的情况下也能运行大型模型，还提供了广泛的后端支持（如 Vulkan、SYCL 和 Metal）。用户可以通过工具将其他模型的权重转换为 llama.cpp 支持的 GGUF 文件格式，从而运行多种模型，包括 LLaMA、LLaMA 2、Falcon、BERT 等。此外，llama.cpp 提供了大量命令行工具，支持交互式聊天、文本生成、语法约束输出等功能，同时兼容 OpenAI API，方便用户构建和部署自定义应用。

llama.cpp 提供了多种部署选项，用户可以通过构建源码、本地安装包（如 Homebrew）、Docker 镜像或直接使用预构建的二进制文件来快速上手。它支持在边缘设备和离线环境中运行模型，非常适合需要高隐私性和低延迟的场景，如企业内部部署、嵌入式设备运行和个人研究用途。此外，llama.cpp 还支持多种编程语言和开发框架的绑定（如 Python、Rust、Node.js 等），以及大量的社

区工具和用户界面，从而使其成为开发大语言模型应用的理想选择。

使用 llama.cpp 前，首先需要下载模型参数文件。Hugging Face 等平台上有大量的适配 llama.cpp 的模型。llama.cpp 要求模型以 GGUF 文件格式存储，对于其他数据格式的模型，可以使用仓库中的 convert_*.py 脚本进行转换。Hugging Face 平台提供多种在线工具来支持与 llama.cpp 的集成，包括 GGUF-my-repo 用于将模型转换为 GGUF 格式并量化权重以减小模型大小，GGUF-my-LoRA 用于将 LoRA 适配器转换为 GGUF 格式，GGUF-editor 支持在浏览器中编辑 GGUF 元数据，以及 Inference Endpoints 功能可直接在云端托管 Llama.cpp 模型。这些工具显著简化了模型格式转换和部署过程。

llama.cpp 提供了多种命令行工具，包括 llama-cli、llama-server、llama-perplexity、llama-bench、llama-run 以及 llama-simple。接下来，分别介绍上述命令的使用。

llama-cli 是用于访问和实验 llama.cpp 大多数功能的命令行工具。主要包含如下几种使用模式：

- 对话模式：具有内置聊天模板的模型会自动激活对话模式。也可以通过添加 -cnv 并使用 –chat-template NAME 指定合适的聊天模板。

```
llama-cli -m model.gguf

# > hi, who are you?
# Hi there! I'm your helpful assistant! I'm an AI-powered chatbot designed to assist
and provide information to users like you. I'm here to help answer your questions,
provide guidance, and offer support on a wide range of topics. I'm a friendly and
knowledgeable AI, and I'm always happy to help with anything you need. What's  on
your mind, and how can I assist you today?
#
# > what is 1+1?
# Easy peasy! The answer to 1+1 is... 2!
```

- 自定义聊天模板的对话模式：

```
# 使用 "chatml" 模板 (使用 -h 查看模板列表)
llama-cli -m model.gguf -cnv --chat-template chatml

# 使用自定义模板
llama-cli -m model.gguf -cnv --in-prefix 'User: ' --reverse-prompt 'User:'
```

- 文本补全模式：使用 -no-cnv 禁用对话模式

```
llama-cli -m model.gguf -p "I believe the meaning of life is" -n 128 -no-cnv

# I believe the meaning of life is to find your own truth and to live in accordance with
it. For me, this means being true to myself and following my passions, even if they
don't align with societal expectations. I think that's what I love about yoga – it's
not just a physical practice, but a spiritual one too. It's about connecting with
yourself, listening to your inner voice, and honoring your own unique journey.
```

- 自定义语法约束模式：

```
llama-cli -m model.gguf -n 256 --grammar-file grammars/json.gbnf -p
'Request: schedule a call at 8pm; Command:'

# "appointmentTime": "8pm", "appointmentDetails": "schedule a a call"
```

grammars/ 文件夹包含示例语法。如果需要编写定制的语法，参阅 GBNF 指南。

llama-server 是一个轻量级的、提供与 OpenAI API 兼容的 HTTP 服务器，为对大语言模型 API 调用提供服务。主要包含如下几种使用模式：

- 在默认配置下，使用端口 8080 启动本地 HTTP 服务器：

```
llama-server -m model.gguf --port 8080

# 基础 Web UI 界面可以通过 http://localhost:8080 访问
# API 调用节点: http://localhost:8080/v1/chat/completions
```

- 多用户并行解码：

```
# 支持最多 4 个并发访问，每个最长 4096 词元上下文
llama-server -m model.gguf -c 16384 -np 4
```

- 推测解码支持：

```
#  draft.gguf 模型是目标模型 model.gguf 的精简版本
llama-server -m model.gguf -md draft.gguf
```

- 嵌入模型服务：

```
# 使用 /embedding 作为访问点
llama-server -m model.gguf --embedding --pooling cls -ub 8192
```

- 重排模型服务：

```
# 使用 /reranking 作为访问点
llama-server -m model.gguf --reranking
```

- 使用语法约束所有输出：

```

# 定义语法
llama-server -m model.gguf --grammar-file grammar.gbnf

# 使用 JSON
llama-server -m model.gguf --grammar-file grammars/json.gbnf
```

llama-perplexity 是一个用于测量模型在给定文本上困惑度（以及其他质量指标）的工具。可以通过如下命令判定给定文本文件的困惑度：

```
llama-perplexity -m model.gguf -f file.txt

# [1]15.2701,[2]5.4007,[3]5.3073,[4]6.2965,[5]5.8940,[6]5.6096,[7]5.7942,[8]4.9297, ...
# Final estimate: PPL = 5.4007 +/- 0.67339
```

llama-bench 用于评测模型推理性能基准评测。

```
llama-bench -m model.gguf

# Output:
# | model          |       size | params | backend    | threads |   test |              t/s |
# | -------------  | ---------: | -----: | ---------- | ------: | -----: | ---------------: |
# | qwen2 1.5B Q4_0 | 885.97 MiB | 1.54 B | Metal,BLAS |      16 |  pp512 | 5765.41 ± 20.55 |
# | qwen2 1.5B Q4_0 | 885.97 MiB | 1.54 B | Metal,BLAS |      16 |  tg128 |  197.71 ± 0.81  |
#
# build: 3e0ba0e60 (4229)
```

12.3.2 Ollama

Ollama 在基于 llama.cpp 开发是一款本地大语言模型运行工具，支持 macOS、Windows 和 Linux 系统，具备简洁的安装和使用流程。用户无需复杂配置，只需通过简单命令（如“ollama run [模型名]”）即可快速启动和运行模型。Ollama 提供丰富的模型库，包括 Llama2、Mistral、DolphinPhi、Code Llama 等，用户还可以通过 modelfile 自定义和微调模型，以满足特定任务需求。此外，Ollama 针对性能进行了优化，即使在普通电脑上也能高效运行小型模型，而在配备高性能 GPU 的设备上则能充分发挥模型的推理能力。

Ollama 还支持多种交互方式，用户既可以通过命令行快速运行模型，也可以选择使用图形用户界面（如 Ollama WebUI 和 macOS 原生应用 Ollamac）进行操作。在数据隐私方面，Ollama 将模型完全本地化运行，数据保留在用户设备上，避免了云端运行可能导致的数据泄露风险，非常适合对隐私要求较高的用户和场景。

Ollama 使用非常简单，在安装完成后，如果想在本地启动 Llama 3.2，可以直接使用如下命令：

```
ollama run llama3.2
```

如果想将大语言模型作为后端服务进行使用，在不启动桌面应用的情况下，可以 ollama serve 命令来启动。用如下命令：

```
# 启动 Ollama 服务
./ollama serve

# 运行模型
./ollama run llama3.2
```

服务启动后，Ollama 提供 REST API 对模型进行调用：

```
# 生成回复
curl http://localhost:11434/api/generate -d '{
  "model": "llama3.2",
  "prompt":"Why is the sky blue?"
}'

# 对话模式
curl http://localhost:11434/api/chat -d '{
  "model": "llama3.2",
  "messages": [
     "role": "user", "content": "why is the sky blue?"
  ]
}'
```

也可以通过界面或者命令行非常方便的拉取、删除或者复制模型：

```
# 创建 Modelfile 文件模型
ollama create mymodel -f ./Modelfile

# 拉取模型
ollama pull llama3.2

# 删除模型
ollama rm llama3.2

# 复制模型
ollama cp llama3.2 my-model
```

此外，Ollama 也支持多模态模型，可以通过参数中加入文件地址完成模型图片输入：

```
ollama run llava "What's in this image? /Users/jmorgan/Desktop/smile.png"
```

12.3.3 Open WebUI

Open WebUI 是一个可扩展、功能丰富且用户友好的自托管大语言模型平台，设计完全离线运行。它支持多种大语言模型运行工具，包括 Ollama 以及所有兼容 OpenAI 的 API，并内置用于 RAG 推理引擎，可以快速构建大语言模型部署解决方案。

如何 Ollama 已经安装于本机，可以使用如下命令，非常方面的通过 Docker 部署 Open WebUI：

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway
-v open-webui:/app/backend/data --name open-webui
--restart always ghcr.io/open-webui/open-webui:main
```

安装完成后可以通过 http://localhost:3000 访问 Open WebUI，如图12.5所示。

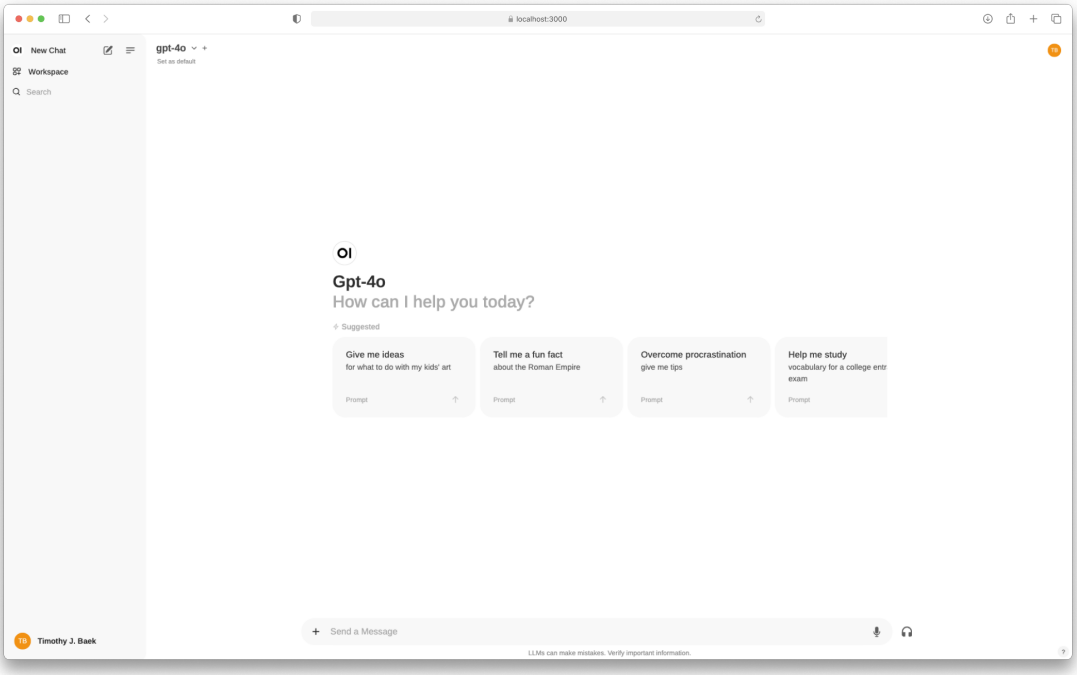

图 12.5　Open WebUI 界面

在安装完成后，也可以通过 Open WebUI 的管理员设置对 OpenAI API 接口进行设置，也可以对本地 Ollama 进行管理，如图12.6和图12.7所示。

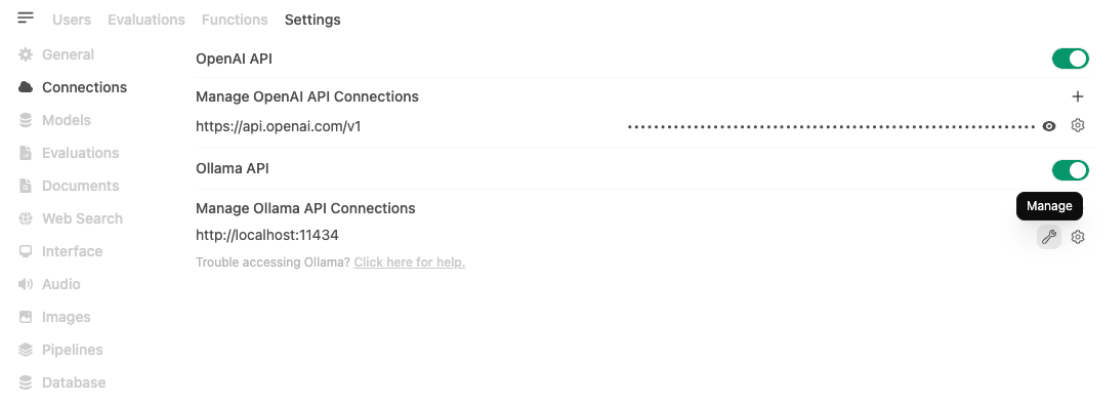

图 12.6　Open WebUI 管理员界面

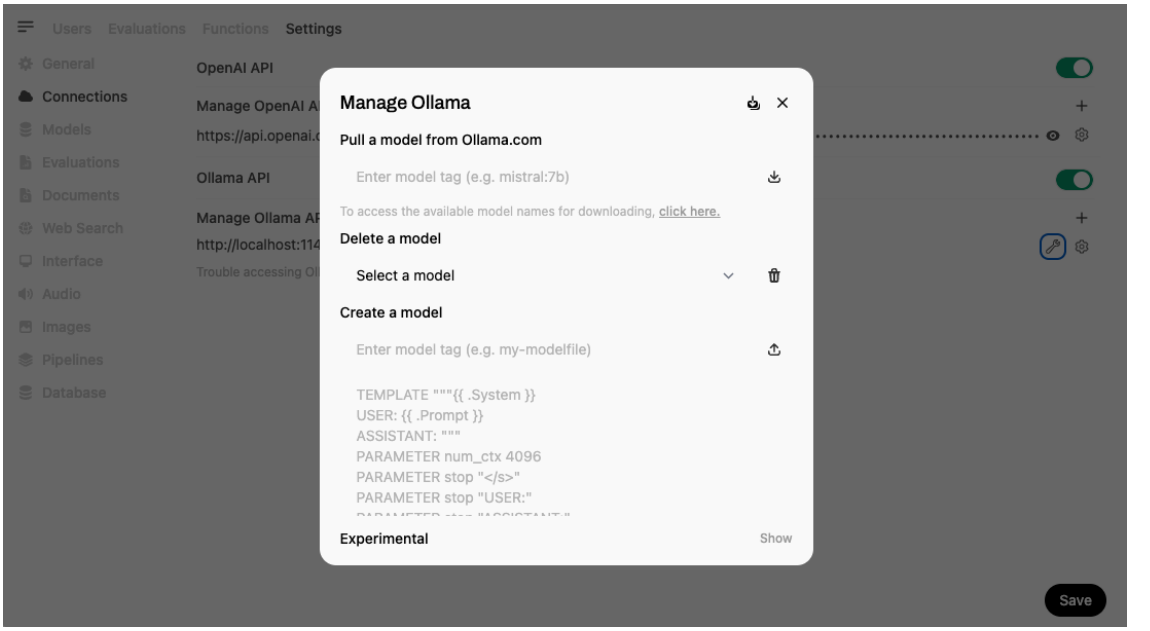

图 12.7　Open WebUI Ollama 管理界面