

杉杉:面向高吞吐低延迟服务的计算机公共课问答系统

杨贇, 刘天扬, 王硕, 苏斌, 蒲鹏, 陆雪松

华东师范大学数据科学与工程学院, 上海 200062

摘要

基于大语言模型的自然语言处理能力在智能问答场景展现出了广泛的应用潜力。为了给高校学生提供个性化的计算机学习支持, 构建了一种基于代码大语言模型的计算机公共课问答系统“杉杉”。该系统采用前后端分离的结构设计, 通过消息队列向大语言模型发送用户指令并监听模型返回结果; 利用大语言模型的语言理解和生成能力, 自动解答学生提出的计算机相关问题, 并采用连续批处理和检索增强生成技术进行优化。性能评估实验表明, 该系统在并发性能、响应时间和多轮对话方面均取得了优于对比方法的表现。

关键词

大语言模型; 智能问答系统; 计算机公共课; 批处理; 检索增强生成

中图分类号: TP391.1

文献标志码: A

doi:10.11959/j.issn.2096-0271.2025058

Shanshan: a question-answering system for computer general courses with high throughput and low latency

YANG Yun, LIU Tianyang, WANG Shuo, SU Bin, PU Peng, LU Xuesong

School of Data Science and Engineering, East China Normal University, Shanghai 200062, China

Abstract

The natural language processing capabilities based on large language models have shown wide application potential in intelligent question-answering scenarios. To provide personalized computer learning support for university students, a computer public course question-answering system named "Shanshan" was developed, which was based on a code-oriented large language model. The system was designed with a front-end and back-end separation architecture, and sends user questions to the large language model through a message queue and listens to the model's return results. Leveraging the language understanding and generation capabilities of the large language model, the system was utilized to automatically answer computer-related questions posed by students, and continuous batch processing and retrieval-augmented generation techniques were employed for optimization. Performance evaluation experiments indicated that the system achieved superior performance in terms of concurrency, response time, and multi-turn dialogue compared to alternative methods.

Key words

large language model, intelligent question-answering system, computer public course, batch processing, retrieval-augmented generation

0 引言

随着全社会数字化进程的推进，高校公共计算机课程（以下简称计算机公共课）已经成为绝大多数高校的公共必修课。其课程目标是培养学生基本的编程思维和利用计算机解决实际问题的能力，从而提升学生的数字素养^[1]。部分学生，进入高校前极少接触专业计算机知识，因此学习计算机课程存在一定困难。特别是课程中的程序设计部分，不仅考验学生对基本概念和语法的理解能力，还要求学生能通过编程实践锻炼提高代码水平。在有限的课堂时间内，教师一般会集中讲解学生经常困惑的共性问题，而面对学生的个性化问题，教师和助教则需要分别花费一定的时间进行解答，对于程序题还需要阅读每个学生编写的代码，并找出其中的错误，极大地增加了教师的教学负担。这种低效率的反馈方式，使学生在遇到的问题时无法及时得到有效的帮助，从而降低了学习的积极性。

为了缓解这一困境，利用智能问答系统辅助教学成为众多高校的一个选择。例如，美国西北大学2018年推出了AI聊天机器人Canvas，并在此后的数年间不断改进，用于回答教师和学生的日常问题。近年来，由于大语言模型出色的语言理解和文本生成能力，基于大语言模型的智能问答系统逐渐成为业界主流，例如OpenAI的ChatGPT、阿里巴巴的通义千问、字节跳动的豆包，都是智能问答系统的标杆。在垂直领域，专用的大语言模型也不断涌现，例如金融领域的BloombergGPT^[2]、医疗领域的ChatDoctor^[3]等。计算机领域也出现了一些优秀的领域大模型，如

DeepSeek^[4]和OpenCodeInterpreter^[5]，这为开发和部署面向高校计算机公共课的智能问答系统带来了契机。然而，在实际部署中，智能问答系统面临并发服务、多轮问答等挑战，导致直接应用模型回答学生问题的性能和效果均欠佳。首先，高校计算机公共课选课人数众多，会出现大量学生同时使用智能问答系统的情形，尤其是在上课期间会出现使用高峰，作为系统基座的大语言模型计算开销巨大，高并发提问会严重降低系统的响应时间，从而极大地影响用户体验。其次，针对某一问题，学生往往需要经过多次提问才能充分理解某个知识点及其相关知识，这就要求智能问答系统具备多轮问答的能力。直接利用大语言模型回答问题会忽略多轮对话和历史对话中的上下文信息，导致系统不能精准回答用户的问题。

为了解决上述问题，本文从系统后端设计和模型推理优化两方面展开研究，构建了基于代码大语言模型的计算机公共课智能问答系统“杉杉”。在系统后端设计上，本文分别采用SignalR和RocketMQ作为实时通信和消息队列中间件，使系统在高并发访问下仍能保持稳定且快速的服务响应。在模型推理优化上，本文使用连续批处理（continuous batching）技术^[6]提升单个图形处理器（graphic processing unit, GPU）运行大语言模型应对高并发提问的能力，进一步缩短了高并发访问下系统的响应时间。此外，为了提高多轮问答的精准度，本文基于检索增强生成（retrieval-augmented generation, RAG）技术，检索历史问答记录扩充上下文，与当前提问一起提示大语言模型回答问题，从而提升回答的质量。实验结果表明，“杉杉”不仅在系统吞吐量和响应时间上的表现优于对比方案，而且在多轮对话中相比直接提示

模型给出了更高质量的回答。目前,“杉杉”已经在华东师范大学“水杉在线”平台完成了部署,成为全校计算机公共课的智能助教。

1 研究现状

1.1 大语言模型用于对话领域

大语言模型是自然语言处理(natural language processing, NLP)领域的最新研究热点。经过海量数据的训练和人类价值的对齐,大语言模型可以生成高质量的自然语言,在一系列NLP任务中取得了优异的表现。大语言模型的基础结构是基于自注意力机制的Transformer架构^[7],该架构最初被用于机器翻译任务,目前已经成为大语言模型的主流架构。通过创新性的伸缩技术,包括混合专家模型(mixture of experts, MoE)^[8]和深度扩展(depth up-scaling, DUS)^[9]等技术,基于Transformer的大语言模型的参数达到数千亿甚至数万亿。现有大语言模型框架主要分为3个类别:自回归模型架构,包括LLaMA (large language model Meta AI)^[9-10]、BLOOM (bigscience large open-science open-access multilingual language model)^[11]和GPT (generative pre-trained transformer)^[12]系列,为单向语言模型;编码器架构,包括ELECTRA (efficiently learning an encoder that classifies token replacements accurately)^[13]和RoBERTa (robustly optimized BERT pretraining approach)^[14]等;编码器-解码器架构,包括BART (bidirectional and auto-regressive transformer)^[15]和T5 (text-to-text transfer transformer)^[16]等。

智能对话是大语言模型的杀手级应用。目前基于大语言模型的知名对话应用包括ChatGPT^[17]、ChatGLM^[18]、LLaMA 2^[19]、Ernie Bot (别名“文心一言”)^[20]等,这些应用通过对大量自然语言文本进行预训练和指令微调,在语言理解和生成方面取得了突破性的进展。除了通用领域的对话应用外,目前也有众多面向特定领域的对话应用,例如医疗^[3,21-22]、法律^[23]、金融^[24]等领域,分别用于辅助诊断、法律咨询和投资建议等场景。

1.2 大语言模型用于计算机教育

在计算机教育领域,目前大语言模型主要被应用于支持编程任务理解、生成教学案例与学生编程练习资源等^[25]。Finnie-Ansley等^[26]通过实验证明OpenAI Codex比大部分学生在计算机专业入门系列课程的测试表现要优秀。Liffiton等^[27]认为准确描述算法步骤的提示能够更有效地辅助学习者编写初级代码,他们提出通过CodeHelp工具为学生提供编程学习辅助问答,并且通过渐进问答方式避免直接提供答案。Leinonen等^[28]认为对代码的准确解释可以帮助学习者在编写代码时提高推理能力,于是提出一种方法,即允许学生直接从代码中提出问题,获得交互式的代码解释。Phung等^[29]利用大语言模型为学生代码的语法错误生成高精度的反馈信息。Finnie-Ansley^[30]和Reeves等^[31]利用大语言模型帮助初学者解决入门级的编程问题。Kazemitabaar等^[32]提出的Coding Steps利用大语言模型辅助初学者进行代码生成。上述工作主要针对特定场景挖掘和验证大语言模型的能力。与这些工作不同,本研究聚焦构建面向通用场景的计算机教

育问答系统，并且在系统性能和多轮对话方面进行优化。

2 “杉杉”系统构建

2.1 整体架构

“杉杉”问答系统的技术选型旨在满足高并发、实时响应和灵活扩展的需求。系统采用前后端分离的结构设计，以提升用户体验和系统性能。“杉杉”系统架构如图1所示，系统由3个模块、若干中间件以及1个数据库组成。其中，3个模块分别是网页前端、网页后端和大语言模型端。前端 Web 应用程序负责与用户交互，并通过超文本传输协议（hypertext transfer protocol，HTTP）接口将用户请求提交

给后端应用程序接口（application program interface，API）服务。后端 API 服务将请求提交给大语言模型端进行推理计算，并将结果返回给前端。为保证用户在提问后能及时获得响应，网页前端和网页后端使用了实时通信技术^[33]，系统能够实现网页前后端的双向通信，实时向用户展示大语言模型生成的回答，减少用户的等待时间。网页后端的多个副本之间通过 Redis 实时同步通信数据，保证前端能接收到并且仅接收到一次返回数据。网页后端与大语言模型端之间通过消息队列 RocketMQ 解耦，可以保证高并发下负载的均衡分配，并且可以提高系统的可靠性。大语言模型端基于 vLLM 框架^[34]实现了连续批处理和流式输出等功能，以缩短在高并发访问场景下的用户响应时间。系统还使用 MongoDB^[35]存储用户问答历史。对于

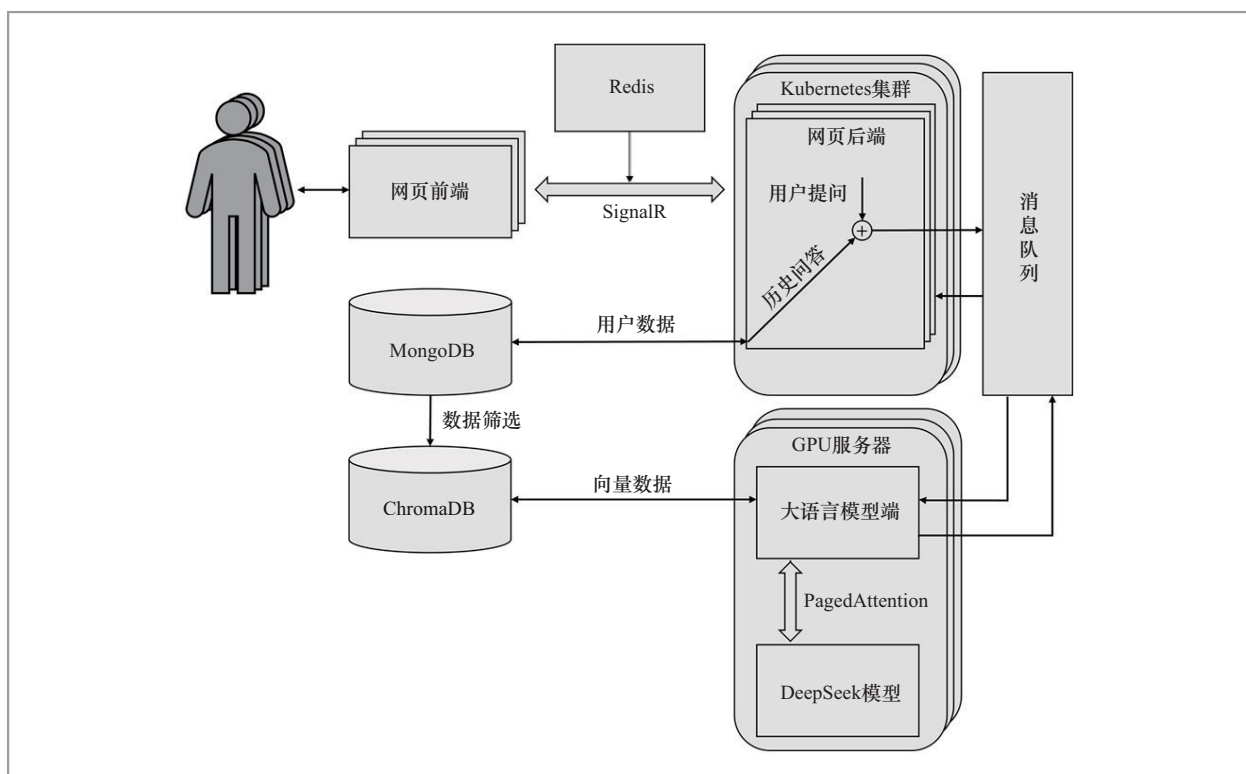


图1 “杉杉”系统架构

MongoDB 中用户评价为高质量的问答数据，将其向量化后存储在 ChromaDB 中。在与用户对话时，采用 RAG 技术提取存储在 ChromaDB 中的相关历史内容以扩充用户提问，提示大语言模型精准完成多轮问答。在部署和管理方面，前后端均采用 Kubernetes 集群进行弹性部署，不仅支持容器化应用的自动化部署、扩展和管理，还能根据实时流量动态调整资源分配，从而提高系统整体的稳定性和可用性。

2.2 面向快速用户响应的前后端设计

网页前端的响应速度主要取决于浏览器加载并运行 Web 应用的效率，传统基于纯 JavaScript 的网页前端在加载复杂 Web 应用时速度较慢，影响了用户体验。为此，本文采用 Blazor 作为前端框架^[36]，其装载模型如图 2 所示。首先，Blazor 是基于 WebAssembly 的前端框架，它采用二进制指令格式，为 Web 应用提供高性能的执行环境，能够在浏览器中快速加载并运行，并且与 JavaScript 并行执行，提供接近本地应用的性能。其次，Blazor 的双向数据绑定特性简化了与后端 API 的交

互过程，允许实时更新用户界面，这种机制非常适合智能问答这类高交互性应用，因为它能确保用户在提问时及时获得响应。从开发者角度来看，Blazor 使开发者能够使用多种编程语言（如 C/C++、C# 和 Rust）编写代码，而不需要依赖 JavaScript^[36-37]，并且支持组件化开发，使用户界面的构建更灵活，每个组件都可以独立开发、测试和维护，减少了代码耦合，提高了系统的可维护性和可扩展性^[38]。Blazor 还能够无缝集成现有的 .NET 库和工具，利用这些成熟的技术提升开发效率和系统功能。

前后端整体响应速度主要取决于前后端之间的通信效率。传统 HTTP 请求-响应模型的通信延迟较高，不适合智能问答这类需要实时数据传输的场景。为此，现在的 Web 应用普遍采用 WebSocket 这种在单个传输控制协议（transmission control protocol，TCP）连接上进行全双工通信的协议，通过在客户端和服务端之间建立持久连接，允许双方在任意时刻发送和接收消息^[37]，提供了更低的延迟和更高的效率。但是，WebSocket 存在通信方式单一、功能实现复杂等缺点，面对不同的网络环境缺乏灵活性。为此，文本采用 SignalR 来实现前后端之间即时通信的功能，SignalR 通信示意图如图 3 所示。SignalR 是一个封装了 WebSocket 等底层技术的库，用于构建实时 Web 应用程序，能够使服务器向客户端推送内容。SignalR 提供了简单易用的 API，让开发者可以专注于业务逻辑而不必聚焦于通信方式的实现。在用户网络状态不佳时，SignalR 能够根据浏览器支持情况自动选择最优的通信方式，包括 WebSocket、Server-Sent Events 和长轮询等，确保系统在各种环境下都能正常工作。此外，SignalR 与 ASP.NET

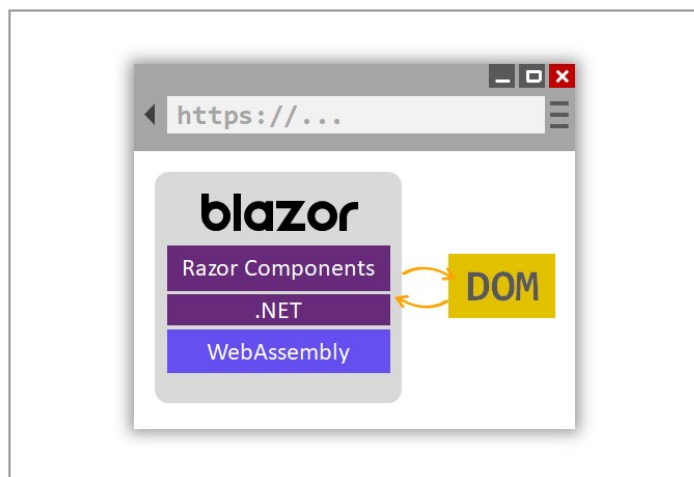


图2 Blazor 装载模型

框架还有着良好的集成，使后端功能实现变得简便，并支持与其他 ASP.NET 组件的无缝协作^[39]。网页后端与大语言模型端通过消息队列进行解耦导致发送问题和接收回答的网页后端可能不是同一个服务。为了确保在这种情况下前端仍能正确接收到后端的回答，系统引入了 Redis 中间件，以实现网页后端之间的 SignalR 相互通信。这种方式使 SignalR 能够有效处理集群环境中的连接请求，确保系统的可用性^[40]。

作为问答系统，除了网页前后端的响应效率以外，还需要考虑网页后端与问答模型之间的通信效率。由于系统采用了大语言模型作为问答模型，其高计算复杂度导致回答单个提问需要消耗 10 s 甚至更长的时间。如果网页后端将提问请求直接发送给大语言模型端，在请求数量稍大时，就有可能造成大语言模型端崩溃。因此，系统采用消息队列作为中间件，将网页后端与大语言模型端解耦。网页后端将用户提问请求发送至消息队列，大语言模型端按照自身的处理能力从消息队列中拉取提问请求进行回答。在使用高峰期，用户提问会被短暂积压在消息队列中，起到削峰作用，使大语言模型端可以利用有限的计

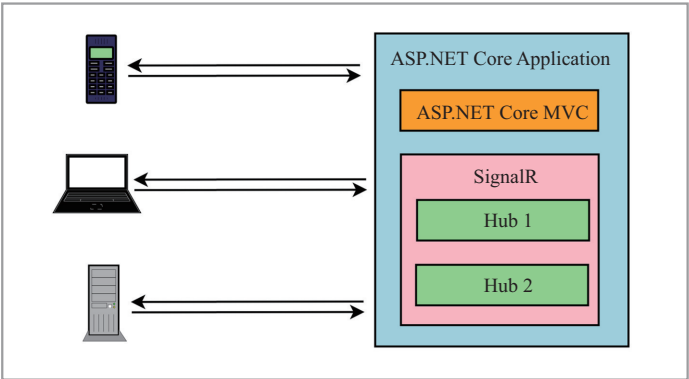


图3 SignalR通信示意图

算资源承载大量提问。特别地，系统采用 RocketMQ^[41]作为消息队列，Rocket MQ 能够提供高可靠性、高性能和可扩展的消息传递服务，以及多种消息存储和确认机制，确保消息在传输过程中的可靠性与持久性，从而避免因故障导致消息丢失。RocketMQ 工作示意图如图 4 所示。在系统中，网页后端将用户提问发送到 RocketMQ 中间件，并从中读取大语言模型处理后的回答；大语言模型端则从 RocketMQ 中间件中读取消息并处理，然后将结果发送回 RocketMQ 中间件。这样的设计提高了网页后端整体的灵活性和可

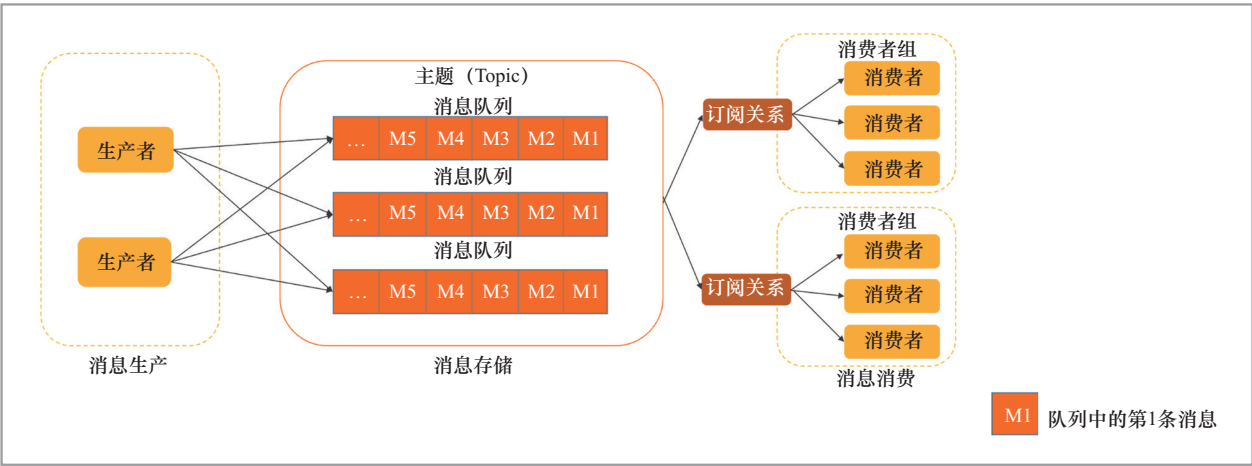


图4 RocketMQ工作示意图

维护性，进而提升了系统整体的可用性。

2.3 面向高并发提问的大语言模型推理服务

大语言模型推理采用迭代计算的方式，即不断基于已经生成的文本内容生成下一个Token。这种计算方式使大语言模型生成完整的回复非常耗时，当面临多个并发提问请求时，后提问的用户将会面临较长的响应时间，在高并发提问场景下更是严重影响用户体验。一个可行的优化方案是使用批处理（batch processing）同时回答多个请求，即大语言模型端从消息队列中一次性取出多个用户提问，利用GPU并行处理的能力，同时生成这些问题的答案，并返回给相应的提问者。这种方法也称为静态批处理。然而，因为不同提问请求及其答案的文本长度可能相差很大，静态批处理不能充分发挥GPU的全部潜力。一方面，同一个批中的请求要等到文本长度最

长的请求和相应答案生成完毕，才能一次性返回给消息队列，造成短文本请求和答案对应的用户的不必要等待；另一方面，由于每个请求会被预先分配固定大小的GPU内存，文本长度较短的请求和答案在当前批中会产生大量空闲内存，造成GPU内存使用率低下。

为了提高大语言模型端应对高并发提问的能力，并且尽可能地提高GPU使用率，采用连续批处理技术^[6]处理用户提问请求。图5模拟了在批大小为4时静态批处理和连续批处理的示意图。T1~T8表示时间戳，S1~S7为大语言模型推理的序列，每个序列的黄色部分代表提示词，蓝色部分代表大语言模型根据提示词生成的Token，红色代表结束。由于大语言模型的推理依赖于之前的Token生成下一个Token，最终序列的生成长度并不相同。在静态批处理中，GPU需要等待一个批次中所有的序列结束生成才会释放资源开始下一个批次的推理，这会导致先完成推理的序列资源被浪费。而在连续批处理中，GPU会在已完成生成的序列之后插入新序列，从而实现比静态批处理更高的GPU利用率，并且不必等待批中所有序列的生成都完成才开始下一个批次的处理。

具体来说，系统采用vLLM大语言模型推理加速框架实现连续批处理，其PagedAttention算法允许将注意力算法中连续的键和值存储在非连续的内存空间中，从而便于批处理的高效内存管理。PagedAttention将每个提问请求序列的键和值缓存划分为块，每个块包含固定数量的键和值。由于这些块不需要在内存中连续，PagedAttention通过块表像操作系统中的虚拟内存那样灵活管理键和值。可以将块类比为分页、Token类比为字节、提问请求类比为进程。一个提问请求的连续

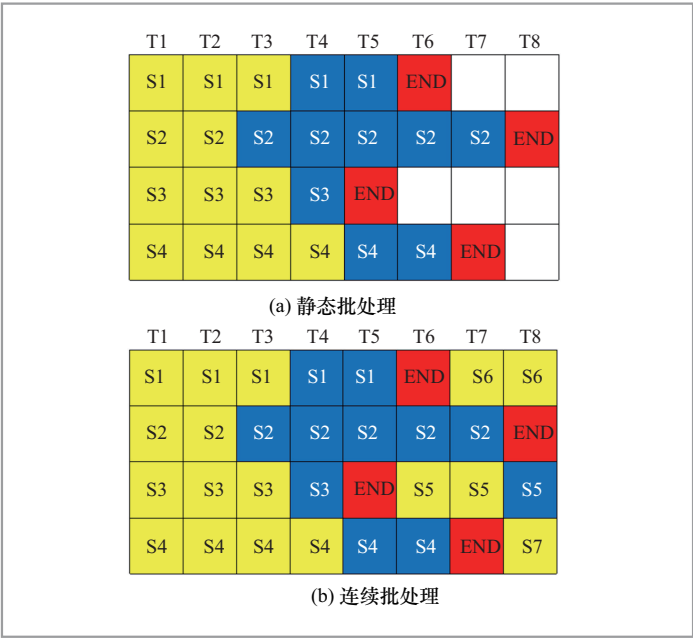


图5 不同批处理技术示意图

逻辑块可以被映射到非连续的物理块，物理块则随着新的 Token 的生成按需分配。每个提问请求只有实际用到的块才被真正放到内存里，这就避免了内部碎片。另外，块相对于一个提问请求的所有缓存来说很小，并且它们的大小相同，这也不会产生很多外部碎片。通过这种方式，一旦当前批中的某个提问请求完成答案生成，就可以将其返回给用户，并且插入一个新的提问请求来代替它，不会存在浪费答案生成之后的内存的情形，从而比传统批处理产生更高的 GPU 利用率，内存浪费只有可能发生在序列的最后一块。在实际使用中，这能够提供接近最优的内存使用，使系统能够批量处理更多的提问请求，显著提高吞吐量。在吞吐量实验中，通过监控 GPU 的实时 KV Cache 使用情况，可以观察到并发峰值下 GPU 的 KV Cache 利用率达到 96% 以上，仅有不到 4% 的浪费。作为 GPU 显存的一部分，KV Cache 的使用率较高说明 GPU 的缓存空间被充分利用。此外，系统基于 vLLM 框架还实现了流式输出，支持逐个返回推理结果中的 Token，避免用户花费较长的时间等待完整的答案。

2.4 基于检索增强生成的多轮对话优化

直接使用用户提问作为提示，大语言模型只会针对当前问题的内容进行回答，并不会关注当前会话的上下文和用户的提问历史，这会导致模型回答脱离上下文，准确性下降，并且因为缺乏对用户已有知识的了解，生成冗长的回答。为了改进这个问题，系统采用RAG技术，从用户历史提问中提取与当前提问相关的内容，与当前提问一起提示大语言模型进行回答。

图6展示了系统RAG部分的数据流。具体来说，采用面向文档存储的NoSQL数据库MongoDB来存储用户提问、回答以及相关的元数据，根据用户对“杉杉”回答的评价，从MongoDB中选择高质量对话数据存储到向量数据库ChromaDB中。当新的用户提问到达时，计算当前提问的嵌入与存储在ChromaDB中该用户历史问答向量的相似度，并提取相似度最高的3条记录对应的文本作为上下文补充，与当前提问一起构成完整的提示，输入大语言模型并获取回答。最后，大语言模型产生的新回答会与当前用户请求组成用户问答数据，存入MongoDB。通过这种方式，大语

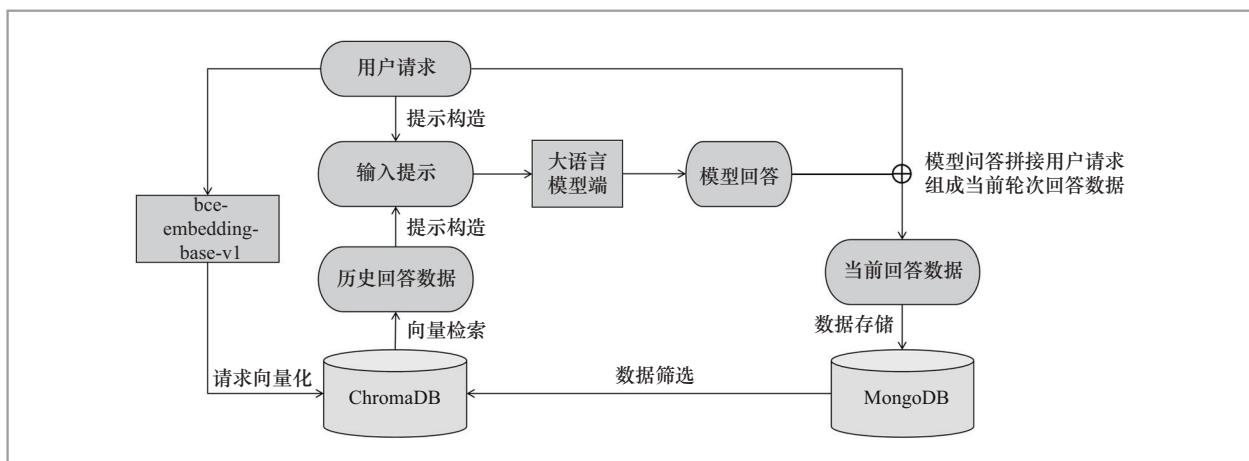


图6 RAG数据流

言模型可以更加精准地理解用户提问的真实意图，从而产生更准确简洁的回答，更有效地解决用户的问题。

3 性能评估

本节内容从响应时间、吞吐量和多轮对话效果评估3个方面对“杉杉”系统进行性能评估。

3.1 大语言模型选型

为了更好地回答与计算机公共课相关的问题，系统采用DeepSeek^[42]开源模型。DeepSeek采用专家混合架构，针对计算

机领域的知识进行了专门训练，支持从自然语言对话到高级代码生成的应用，在MMLU（massive multitask language understanding）、CMMLU（chinese massive multitask language understanding）和C-Eval（chinese evaluation benchmark）等标准基准测试中的表现突出。在代码生成方面，DeepSeek在HumanEval和LiveCodeBench等实时代码生成测试中表现出色，能很好地应对复杂的编程任务。特别地，系统采用了DeepSeek的最新版本DeepSeek-V2，在推理速度和资源利用率方面进行了优化，能够更好地应用于大学计算机公共课的场景。

3.2 系统性能评估

采用GPT4生成一组与计算机相关的问题，包括概念问题和程序问题。随后，利用Locust模拟在线服务用户高并发请求的应用场景，在1 s之内生成并发送所有用户的提问，并在用户收到完整回复后继续提问，用户提问的内容从GPT4生成的问题中随机挑选。测试在不同用户数量和不同批大小的情况下模型完成前1 000个提问请求的响应时间和吞吐量。

首先，比较前后端通信方式对系统性能的影响。固定批大小为128，用户数量按照1、2、4、8、16、32、64、128、256逐步增加，比较前后端通信分别采用SignalR和WebSocket对系统平均响应时间和吞吐量的影响，实验结果如图7所示。由图7可知，随着用户数量不断增加，采用SignalR比传统WebSocket逐步降低了平均响应时间和提高了系统吞吐量。这是因为当系统负载变大时，SignalR能够根据负载情况自动选择最优的通信方式，从而获得最佳的通信性能。

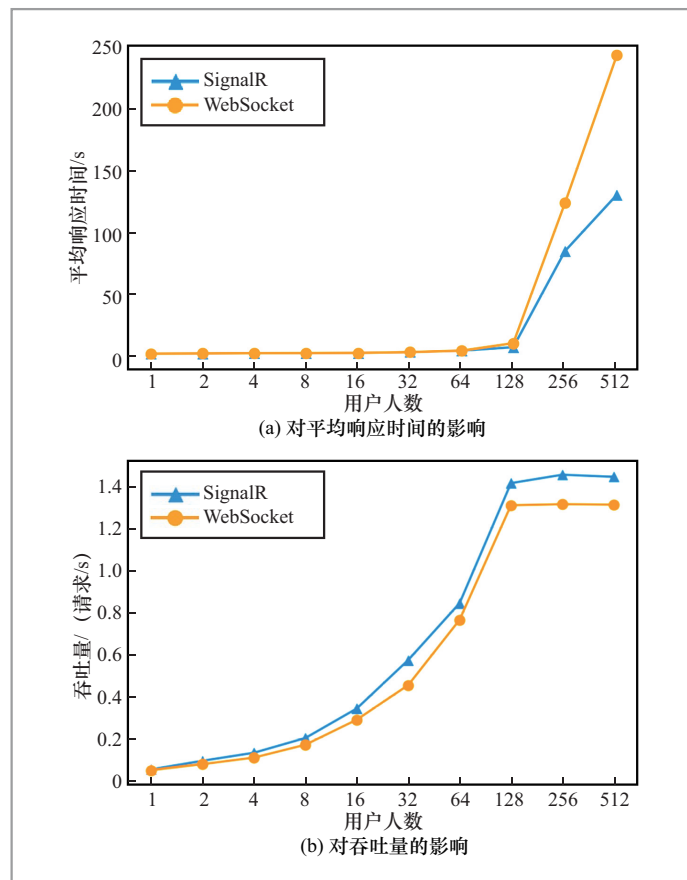


图7 前后端通信方式对系统性能的影响

其次，比较使用和不使用消息队列RocketMQ的情况下系统的性能。固定批大小为128，用户数量按照1、2、4、8、16、32、64、128、256逐步增加，实验结果如图8所示。由图8可知，随着用户数量的不断增加，不使用消息队列会使平均响应时间快速提高、系统吞吐量快速下降（当用户人数超过批大小后）。这是因为大语言模型端无法消化大量同时发出的提问请求，而造成系统堵塞甚至崩溃。在使用RocketMQ作为消息队列后，这一情况大大缓解，平均响应时间随着用户数量的增加缓慢升高。当用

户人数超过批大小后，系统吞吐量能够维持相对稳定的状态而不会快速下降。

接着，比较采用连续批处理和静态批处理对系统性能的影响。固定用户人数为128，批大小按照1、2、4、8、16、32、64、128逐步增加。实验结果如图9所示。由图9可知，连续批处理机制能够在高并发场景中显著降低平均响应时间和提高系统吞吐量。在GPU内存允许的前提下，响应时间和吞吐量随着批大小增大而快速下降和提高。例如，在设置批大小为128时，即128名用户并发提问的情形下，平均响

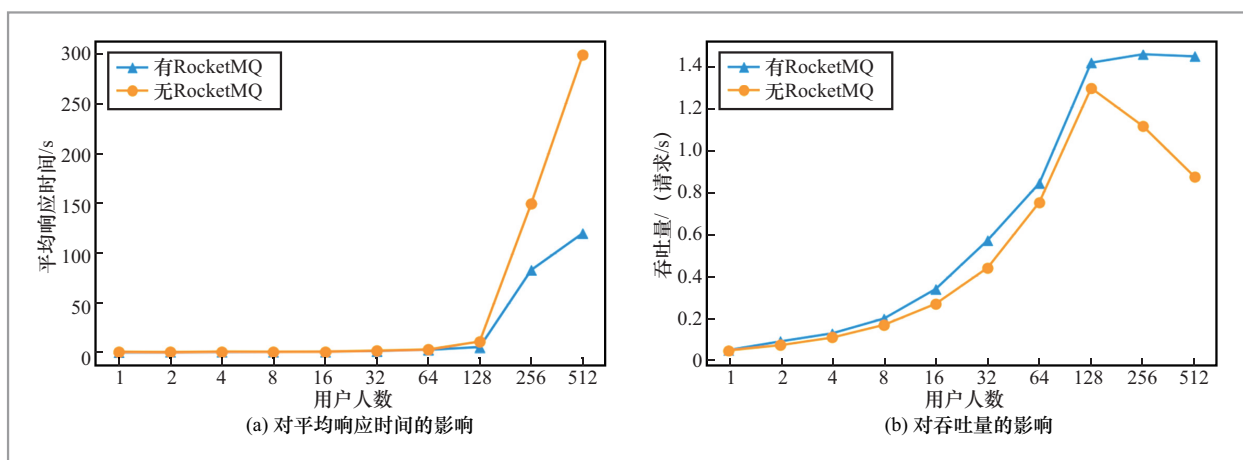


图8 使用和不使用消息队列对系统性能的影响

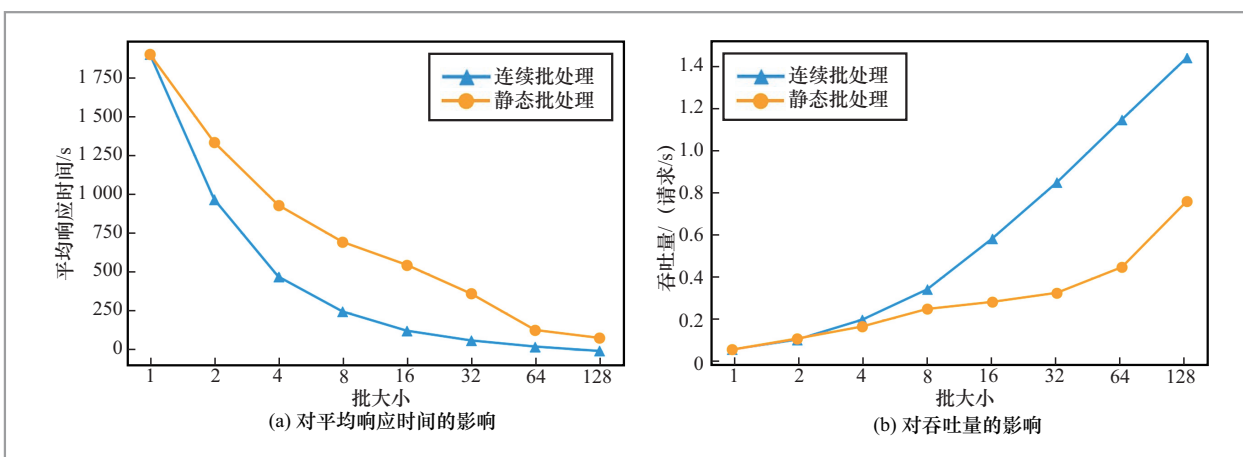


图9 批处理方式对系统性能的影响

应时间不到 1 s，几乎没有响应延迟。

最后，评估向量数据库 ChromaDB 中存储用户历史问答数据量变化对检索耗时的影响。控制 ChromaDB 中存储用户历史问答数据量按照 1、2、4、8、16、32、64、128、256、512、1 024 逐步增加，计算从中检索与当前用户提问最相关的 3 条历史问答数据的耗时。实验结果如图 10 所示。由图 10 可知，在实验使用的数据量下，检索最相关历史问答数据的耗时在 14 ms 左右波动，基本不受 ChromaDB 内数据量变化的影响。这是由于 ChromaDB 内部针对向量数据的存储和检索进行了高度优化，同时应用了高效的向量索引技术，在检索相关文档部分，时间主要消耗在对用户当前提问进行向量化的操作上。

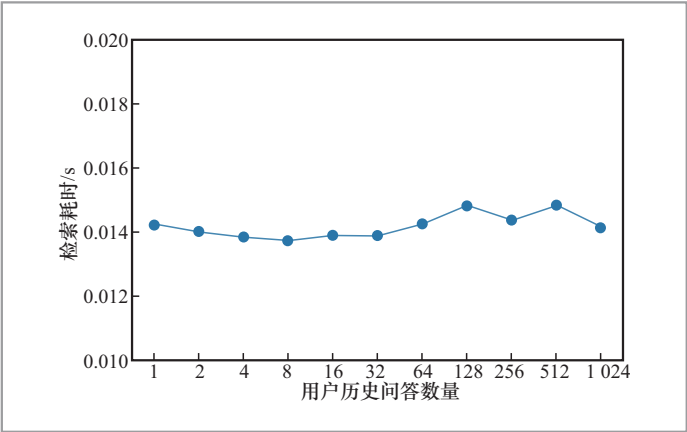


图10 用户历史问答数量对检索耗时的影响

3.3 多轮对话评估

从学生提问记录中抽取 126 条多轮对话数据作为实验样本，通过人工评估比较不使用 RAG 技术和使用 RAG 技术加入对话历史的模型回答的质量。实验邀请了 5 名计算机专业的硕士研究生，对每条多轮对话数据，从模型响应是否通顺（通顺度）、是否联系上下文信息（关联度）和是否解决用户问题（准确度）3 个维度评价模型的回答质量，分值为 0~3 分，分数越高，代表质量越好。人工评价的指标见表 1。

通过评估模型在不同对话轮次情况下的表现，计算其得分并求平均值，最终得到多轮对话评估结果，见表 2。从表 2 中可以看到，在使用 RAG 技术加入当前会话的对话历史后，模型在多轮对话的 3 个指标上均有不同程度的提升，尤其在关联度，即回答是否联系上下文信息上，分数提升最明显。这验证了加入对话历史提升多轮对话的效果。同时，使用 RAG 技术之后，模型在准确度，即是否解决用户问题上，也取得了巨大的提升。实验结果表明，系统通过引入 RAG 技术，可以在多轮对话中更好地解决学生的疑问。

4 应用案例

“杉杉”目前已部署在华东师范大学

表1 人工评价指标

指标	分值			
	3	2	1	0
通顺度	回答语句通顺,不冗余,没有无关信息	回答语句较通顺,可读性较好	回答语句具有一定可读性	回答语句出现乱码,没有可读性
关联度	回答内容与上下文高度关联	回答内容存在与上下文的关联	回答内容几乎没有与上下文的关联	回答内容完全没有与上下文的关联
准确度	很好地解决了用户的问题	一定程度上解决了用户的问题	不能解决用户的问题	没有回答用户问题,甚至出现幻觉等误导内容

“水杉在线”平台上，为计算机公共课学生提供实时答疑。对于学生的第一个问题“哈希是什么”，“杉杉”介绍了哈希算法的相关知识，包括哈希函数的特性、哈希函数的应用以及具体的哈希函数实例，如图 11 所示。

在学生继续提问“可以用 Python 实现一种吗”的时候，“杉杉”通过上下文正确地理解了学生的问题，给出了一种用 Python 语言实现的哈希算法的代码，如图 12 所示。上述示例展示了系统可以正确理解会话历史，并给出较通顺、可读性较好的回答。

5 结束语

本文构建了基于代码大语言模型的计算机公共课问答系统“杉杉”，通过 Web 端向用户提供交互式页面，并基于 vLLM 推理服务为用户生成回答结果，利用文档数据库记录用户历史问答数据，以支持多轮对话。在应用中发现，智能回答系统能够提高教师的工作效率，使他们能够把更多的时间和精力投入提高教学质量和专业指导上；其次，通过即时反馈和个性化学习建议，学生能够快速理解概念和纠正错误，增强学习的连续性和提高效率。希望通过部署这一问答系统，提升学生独立解决问题的能力，同时提高学生的编程思维和代码质量，培养出更加适应数字化时代需求的个性化人才。

参考文献：

[1] 李旭宏, 潘文文, 赵猛猛. MOOC 环境下计算机网络安全教学的思考[J]. 中国多媒体与网络教学学报(上旬刊), 2018(9): 129–130.

LI X H, PAN W W, ZHAO M M. Reflec -

表2 多轮对话评估结果

对话轮次		指标		
		通顺度	关联度	准确度
2	无RAG	2.873	1.087	1.325
	有RAG	2.905	2.841	2.762
3	无RAG	2.825	1.056	1.048
	有RAG	2.857	2.802	2.730
4	无RAG	2.778	1.048	1.032
	有RAG	2.794	2.643	2.460
5	无RAG	2.675	1.024	1.000
	有RAG	2.761	2.087	2.087

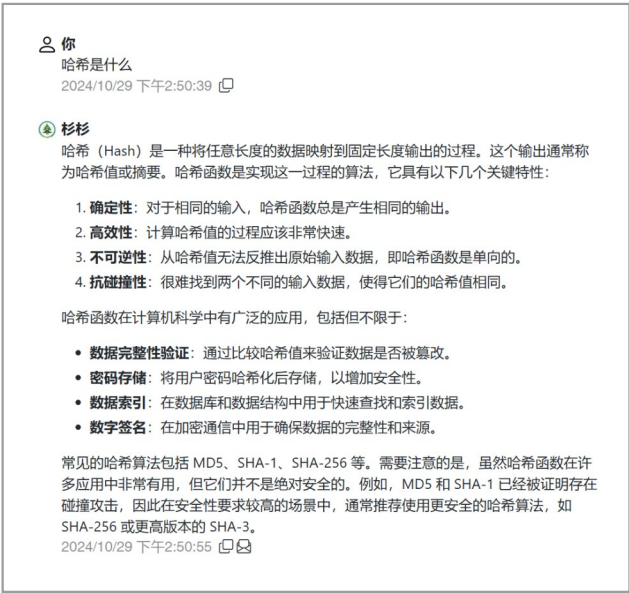


图 11 多轮对话案例一



图 12 多轮对话案例二

- tions on computer network security teaching in MOOC environment[J]. China Journal of Multimedia & Network Teaching, 2018(9): 129-130.
- [2] WU S J, IRSOY O, LU S, et al. BloombergGPT: a large language model for finance[EB]. arXiv preprint, 2023, arXiv: 2303.17564.
- [3] LI Y X, LI Z H, ZHANG K, et al. ChatDoctor: a medical chat model fine-tuned on a large language model meta-AI (LLaMA) using medical domain knowledge[J]. Cureus, 2023, 15(6): e40895.
- [4] GUO D Y, ZHU Q H, YANG D J, et al. DeepSeek-coder: when the large language model meets programming: the rise of code intelligence[EB]. arXiv preprint, 2024, arXiv: 2401.14196.
- [5] ZHENG T Y, ZHANG G, SHEN T H, et al. OpenCodeInterpreter: integrating code generation with execution and refinement[EB]. arXiv preprint, 2024, arXiv: 2402.14658.
- [6] YU G I, JEONG J S, KIM G W, KIM G W et al. Orca: a distributed serving system for transformer-based generative models[C]//Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2022). [S.l.:s.n.], 2022: 521-538.
- [7] VASWANI A, SHAZEER N, PARMAR N. Attention is all you need[C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. New York: Curran Associates Inc., 2017: 6000-6010.
- [8] CAI W L, JIANG J Y, QIN L, et al. Shortcut-connected expert parallelism for accelerating mixture-of-experts[EB]. arXiv preprint, 2024, arXiv: 2404.05019.
- [9] KIM D, PARK C, KIM S, et al. Solar 10.7 b: scaling large language models with simple yet effective depth up-scaling[EB]. arXiv preprint, 2023, arXiv: 2312.15166.
- [10] TOUVRON H, LAVRIL T, IZACARD G, et al. LLaMA: open and efficient foundation language models[EB]. arXiv preprint, 2023, arXiv: 2302.13971.
- [11] LE SCAO T, FAN A, AKIKI C, et al. Bloom: a 176b-parameter open-access multilingual language model[EB]. arXiv preprint, 2022, arXiv: 2211.05100.
- [12] ACHIAM J, ADLER S, AGARWAL S, AHMAD L, et al. Gpt-4 technical report [EB]. arXiv preprint, 2023, arXiv: 2303.08774.
- [13] CLARK K, LUONG M T, LE Q V, et al. ELECTRA: pre-training text encoders as discriminators rather than generators [EB]. arXiv preprint, 2020, arXiv: 2003.10555.
- [14] LIU Y H, OTT M, GOYAL N, et al. RoBERTa: a robustly optimized BERT pretraining approach[EB]. arXiv preprint, 2019, arXiv: 1907.11692.
- [15] LEWIS M, LIU Y H, GOYAL N, et al. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension [C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Dublin: Association for Computational Linguistics, 2020: 7871-7880.
- [16] RAFFEL C, SHAZEER N, ROBERTS A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. Journal of Machine Learning Research, 2020, 21(1): 5485-5551.
- [17] ROUMELIOTIS K I, TSELIKAS N D. ChatGPT and open-ai models: a preliminary review[J]. Future Internet, 2023, 15(6): 192.
- [18] DU Z X, QIAN Y J, LIU X, et al. GLM: general language model pretraining with autoregressive blank infilling[C]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dub-

- lin: Association for Computational Linguistics, 2022: 320–335.
- [19] TOUVRON H, MARTIN L, STONE K, et al. Llama 2: Open foundation and fine-tuned chat models[EB]. arXiv preprint, 2023, arXiv: 2307.09288.
- [20] MOTLAGHNY, KHAJAVIM, SHARIFIA, et al. The impact of artificial intelligence on the evolution of digital education: a comparative study of OpenAI text generation tools including ChatGPT, Bing chat, bard, and ernie[EB]. arXiv preprint, 2023, arXiv: 2309.02029.
- [21] 张芊, 陈攀峰, 冯林坤, 等. PeMeBench: 中文儿科医疗问答基准测试方法[J]. 大数据, 2024, 10(5): 28–44.
- ZHANG Q, CHEN P F, FENG L K, et al. PeMeBench: Chinese pediatric medical Q & A benchmark testing method[J]. Big Data Research, 2024, 10(5): 28–44.
- [22] ZHANG H B, CHEN J Y, JIANG F, et al. HuatuoGPT, towards taming language model to be a doctor[C]//Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023. Singapore: Association for Computational Linguistics, 2023: 10859 – 10885.
- [23] CUI J, LI Z, YAN Y, et al. Chatlaw: Open-source legal large language model with integrated external knowledge bases[EB]. arXiv preprint, 2023, arXiv: 2306.16092.
- [24] YANG H Y, LIU X Y, DAN WANG C. FinGPT: open-source financial large language models[J]. arXiv preprint, 2023, arXiv:2306.06031.
- [25] LEINONEN J, HELLAS A, SARSA S, et al. Using large language models to enhance programming error messages[C]//Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1. New York: ACM, 2023: 563–569.
- [26] FINNIE-ANSLEY J, DENNY P, LUXTON-REILLY A, et al. My AI wants to know if this will be on the exam: testing OpenAI’s codex on CS2 programming exercises[C]//Proceedings of the 25th Australasian Computing Education Conference. New York: ACM, 2023: 97–104.
- [27] LIFFITON M, SHEESE BE, SAVELKA J, et al. CodeHelp: using large language models with guardrails for scalable support in programming classes[C]//Proceedings of the 23rd Koli Calling International Conference on Computing Education Research. New York: ACM, 2024: 1–11.
- [28] DENNY P, KUMAR V, GIACAMAN N, et al. Conversing with copilot: exploring prompt engineering for solving CS1 problems using natural language[C]//Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1. New York: ACM, 2023: 1136–1142.
- [29] PHUNG T, CAMBRONERO J, GULWANI S, et al. Generating high-precision feedback for programming syntax errors using large language models[C]//Proceedings of the 16th International Conference on Educational Data Mining. International Educational Data Mining Society, 2023: 370–377.
- [30] FINNIE-ANSLEY J, DENNY P, BECKER B A, et al. The robots are coming: exploring the implications of OpenAI codex on introductory programming[C]//Proceedings of the 24th Australasian Computing Education Conference. New York: ACM, 2022: 10–19.
- [31] REEVES B, SARSA S, PRATHER J, et al. Evaluating the performance of code generation models for solving Parsons problems with small prompt variations [C]//Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1. New York: ACM, 2023: 299–305.
- [32] KAZEMITABAARM, CHOW J, MACK T, et al. Studying the effect of AI code

generators on supporting novice learners in introductory programming[C]//Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. New York: ACM, 2023: 1–23.

[33] JENNINGS R B, NAHUM E M, OLSHEFSKI D P, et al. A study of Internet instant messaging and chat protocols[J]. IEEE Network, 2006, 20(4): 16–21.

[34] KWON W, LI Z H, ZHUANG S Y, et al. Efficient memory management for large language model serving with PagedAttention[C]//Proceedings of the 29th Symposium on Operating Systems Principles. New York: ACM, 2023: 611–626.

[35] CHAUHAN D, BANSAL K L. Using the advantages of NOSQL: a case study on MongoDB[J]. International Journal on Recent and Innovation Trends in Computing and Communication, 2017, 5(2): 90–93.

[36] HIMSCHOOT P. Microsoft Blazor: building Web applications in .NET 6 and beyond [M], California: Apress Berkeley, 2021: 1–49.

[37] TUFEGDŽIĆ J, DODOVIĆ M, OGRIZOVIĆ M, et al. Application of WebAssembly technology in high-performance web applications[C]//Proceedings of the 2024 11th International Conference on Electrical, Electronic and Computing Engineering (IcETRAN). Piscataway: IEEE Press, 2024: 1–6.

[38] SANDBERG E. Evaluating Blazor: a comparative examination of a web framework[M]. 2021.

[39] SHARMA N, AGARWAL R. HTTP, WebSocket, and SignalR: a comparison of real-time online communication protocols[C]//Proceedings of the 9th International Conference(MIKE 2023). Cham: Springer, 2023: 128–135.

[40] KYLMÄKÄ V. Horizontal scalability and high availability of a hospital information system[D]. Oulu: University of Oulu, 2023.

[41] MA Y, YAN R Y, SUN J W, et al. A MQTT protocol message push server based on RocketMQ[C]//Proceedings of the 2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA). Piscataway: IEEE Press, 2017: 295–298.

[42] BI X, CHEN D L, CHEN G T, et al. DeepSeek LLM: scaling open-source language models with longtermism[EB]. arXiv preprint, 2024: 2401.02954.

作者简介



杨贇（1999–），男，华东师范大学数据科学与工程学院硕士生，主要研究方向为教育领域的大模型应用。



刘天扬（2001-），男，华东师范大学数据科学与工程学院硕士生，主要研究方向为教育领域的大模型应用。



王硕（2000-），男，华东师范大学数据科学与工程学院硕士生，主要研究方向为教育领域的大模型应用。



苏斌（1997-），男，华东师范大学数据科学与工程学院工程师，主要研究方向为数据驱动的计算教育学。



蒲鹏（1978-），男，华东师范大学数据科学与工程学院高级工程师，主要研究方向为数据驱动的计算教育学。



陆雪松（1985-），男，华东师范大学数据科学与工程学院副教授，主要研究方向为数据驱动的计算教育学。

收稿日期: 2024-11-18

通信作者: 苏斌, bsu@dase.ecnu.edu.cn

基金项目: 国家自然科学基金项目(No.62277017)

Foundation Item: The National Natural Science Foundation of China (No.62277017)