计算机系统结构项目文档

题目：部署引入检索的大型语言模型

摘要：

本文介绍了大型语言模型，它们通过利用深度学习和自然语言处理技术来理解和生成人类语言，并广泛应用于机器翻译、情感分析、文本摘要和问答系统等领域。这些模型通过训练大量文本数据来学习词汇、语法和句子结构等语言特征。但这些模型需要大量的计算资源和数据，可能导致高成本、环境影响和偏见问题。为了解决这些问题，本文探讨了将检索与生成相结合的RAG方法和其他检索方法如知识图谱。在开放领域对话生成中，已证明RAG等检索方法有效。本文介绍了使用FAISS和DPR等检索方法进行本地部署的实验，并讨论了云端部署的选择，指出引入RAG检索和WebGLM在线检索都有不同特点，因此应根据需求和资源选择合适的模型和部署方式。

关键字：检索增强，生成式语言模型，自然语言处理

1. 简介

大型语言模型是一种人工智能技术，它使用深度学习和自然语言处理技术来理解和生成人类语言。这种模型通常由数百万甚至数十亿个参数组成，可以处理大量的文本数据，从而学习语言的模式和结构。通过对大量文本数据的训练，这些模型可以学习到词汇、语法、句子结构等语言特征，从而能够生成与输入文本相似的新文本。

大型语言模型的训练通常包括两个阶段：预训练和微调。在预训练阶段，模型会在大量的文本数据上进行训练，学习语言的基本特征和模式。在微调阶段，模型会在特定的任务或领域的数据上进行训练，以适应特定的应用场景。例如，一个模型可以在医学文献上进行微调，以便更好地理解和生成医学相关的文本。

大型语言模型在许多应用中都有广泛的应用。例如，它们可以用于机器翻译、情

感分析、文本摘要、问答系统等。这些模型可以理解和生成多种语言，包括英语、中文、西班牙语等。此外，这些模型还可以用于生成创意性的文本，如小说、诗歌等。

然而，大型语言模型也存在一些挑战和问题。首先，这些模型需要大量的计算资源和数据来进行训练，这可能导致高昂的成本和环境影响。其次，这些模型可能会学习到训练数据中的偏见和歧视，从而在生成的文本中体现出这些偏见。此外，这些模型可能会生成不准确或不真实的信息，这可能对用户造成误导。

因此，近年来，研究人员已经开始探索一些新的方法来解决这些问题。其中之一就是Retrieval-Augmented Generation（RAG）方法。RAG是一种将检索和生成结合起来的方法，它可以在生成文本之前先从大型数据库中检索相关信息。通过这种方式，模型可以利用外部知识来生成更准确和真实的文本。

RAG方法的工作原理是将输入的文本与数据库中的文档进行匹配，找到与输入文本相关的文档。然后，模型会根据这些文档生成文本。这种方法可以有效降低大型语言模型的幻觉，因为模型可以利用外部知识来生成文本，而不是完全依赖于训练数据中的信息。

除了RAG方法之外，还有一些其他的检索方法也可以有效降低大型语言模型的幻觉。例如，一些模型使用了知识图谱来辅助生成文本。知识图谱是一种结构化的知识表示方法，它可以将知识以图的形式表示出来。通过使用知识图谱，模型可以更好地理解输入文本中的信息，并生成更准确和真实的文本。

Retrieval-Augmented Generation方法和其他类似的检索方法都可以有效降低大型语言模型的幻觉。这些方法可以利用外部知识来生成文本，从而提高文本的准确性和真实性。然而，这些方法也有一些局限性，例如它们可能需要大量的计算资源和数据来进行检索。因此，研究人员仍然在探索更加高效和准确的方法来解决大型语言模型的幻觉问题。

2. 背景和相关工作

Retrieval方法，如RAG，已在QA任务中证明有效，但尚未应用于开放领域对话生成。在图1中，Parametric memory是指在神经网络中隐式存储在权值中的知识。Non-Parametric memory是指从Wikipedia corpus中抽取的2100万组100个单词的

序列。我们使用Document Encoder对长度为100个字的序列进行编码。

当我们提出一个query（末端有一个mask的x序列）时，对该query进行编码，并使用MIPS（maximum inner product search implemented with face library）来找到在Non-Parametric memory中已编码的最相似的文档。当query和document被编码后，它们作为孪生Sentence-Bert模型的输入。BERT模型可以将两个不同序列作为输入，通过计算两个序列的交叉注意力，对两个不同序列进行语义相似性自然语言推理比较。Generator将前面检索得到的这些文档作为输入，并将其附加到上下文X中。

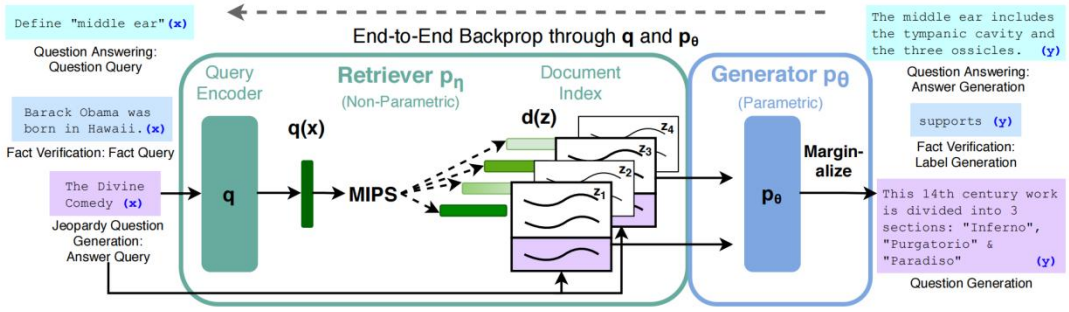


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

图1：RAG方法的框架。

对于检索，作者同时采用了两种方法：FAISS（Facebook AI Similarity Search）和DPR（Dense Passage Retrieval）。

FAISS方法为稠密向量提供高效相似度搜索和聚类，支持十亿级别向量的搜索。给定向量集合Wikipedia corpus，通过FAISS建立索引。FAISS通过倒排索引（Inverted File System）和乘积量化（Product Quantization）两个技术，旨在使用聚类缩短向量集合空间中的距离和量化加快检索速度。

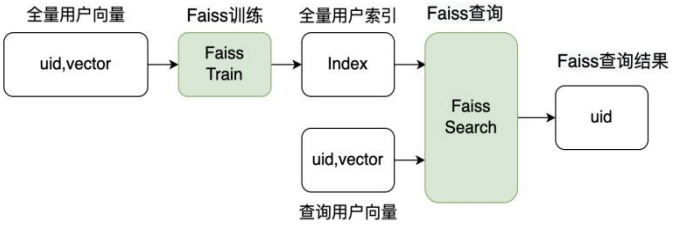


图2：FAISS工作流程图

Dense Passage Retrieval（DPR）是一种开放领域的问答检索方法，它使用预先训练的密集编码器和高效的相似性搜索来从大型语料库中找到相关段落。与传统的稀疏检索方法相比，DPR的主要优势在于它能够捕捉问题和段落之间的语义关系，从而获得更高质量的检索结果。通过使用密集嵌入表示问题和段落，DPR可以识别相关段落，即使问题和段落之间没有精确的术语匹配。如图3所示，将向量集合中的每一个向量输入至一个单独的编码器。文档编码器和检索编码器是两个独立的具有1.1亿个参数的BERT模型，不共享参数。DPR不训练文档编码器，并对文档进行一次编码，构建出便于向量相似度搜索的文档索引。

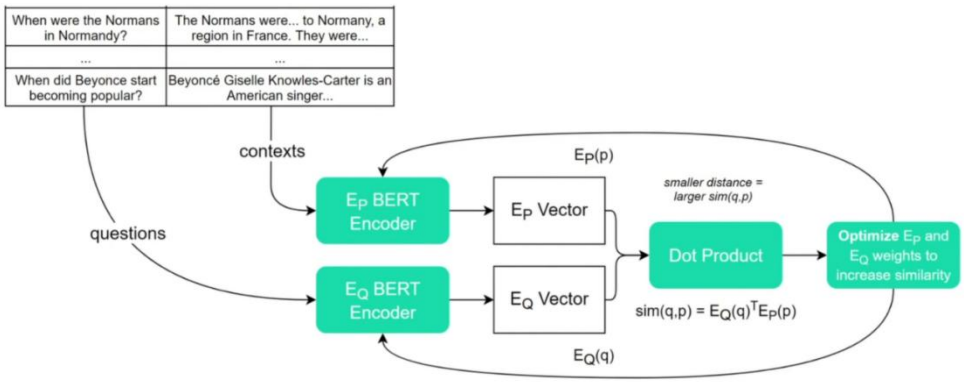


图3：DPR工作流程图

以下是将 FAISS 与 DPR 结合使用的抽象概述：

训练神经网络模型（如 BERT 或 RoBERTa），将文档和查询嵌入高维向量空间。使用训练好的模型将文档嵌入数据集中，并创建文档向量的 FAISS 索引。当收到查询时，使用训练有素的模型将查询嵌入相同的向量空间。使用 FAISS 索引快速检索与查询向量最相似的文档。根据相似度得分对检索到的文档进行排序，并将排序靠前的结果返回给用户。

3. 实验部署

选择云算力提供商：

在这个项目中，我选择了 AutoDL，因为它具备弹性租用的设定，提供多种不同环境与镜像。此外，他们专门针对部署环境提供了加速服务，加快环境依赖安装

和克隆的速度。

实验设备：

GPU：RTX 3090(24GB) * 1

CPU：14 vCPU Intel(R) Xeon(R) Gold 6330 CPU @ 2.00GHz

内存：80GB

4. 问题和困难

由于云服务提供商不提供额外的VPN服务，RAG实验项目中所使用Generator为ChatGPT，其模型不对内地提供服务，因此导致无法访问模型接口API，不能将该模型部署至内地云服务器。对于RAG检索模型的实验在本地部署，查看检索效果。对于部署至云端的模型更改为WebGLM，WebGLM 旨在使用 10 亿参数的通用语言模型（GLM）提供一种高效且低成本的网络增强问答系统。它旨在通过将网络搜索和召回功能集成到预训练的语言模型中以进行实际应用的部署。

5. 实验内容

本地部署实验

对于建立索引，引用了haystack库。定义了两个函数：preprocess_docs 和 vector_stores。preprocess_docs 函数将目录路径作为输入，将目录中的文件转换为文档，然后使用 haystack.nodes 模块中的 PreProcessor 类处理这些文档。然后返回处理后的文档。vector_stores 函数将处理过的文档作为输入，使用指定的数据库文件路径创建 SQLite 引擎，并尝试从数据库中删除名为 "document "的表。然后，它使用指定参数创建 FAISSDocumentStore 实例，并将处理过的文档写入文档存储区。然后返回文档存储。用于预处理文档并将其存储到 FAISS 文档存储库中，这是一个基于向量的文档存储库，可以高效地进行文档的相似性搜索和检索。

对于检索文档，定义了一个使用 Haystack 库创建问题解答管道的函数。该管道从文档存储中检索相关文档，读取这些文档的内容，并从文档中提取问题的答案。其中，使用RoBerta读取文档内容，使用sentence BERT作为检索器的编码器并从文档索引中提取出确切的答案。

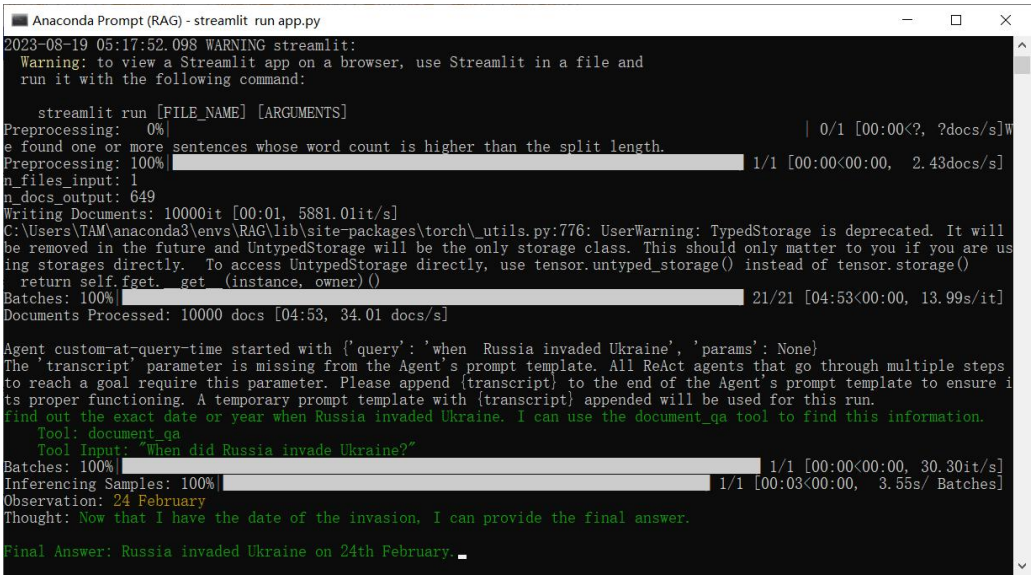
创建一个使用 GPT-4 语言模型生成问题回复的代理。该代理配备了一个工具，可使用 document_qa 管道搜索与问题有关的答案。该代理使用提示模板来格式化回复，并提供如何回答问题的说明。

如图4所示，这是截取自数据库用于检索的文档，文档开头第一句写明关于最近俄乌冲突发生的具体日期。如果没有任何模板以及检索增强的情况下，GPT4生成的回复是2014年（图5），不能紧贴目前时事动向。因此，如图6所示，加入RAG方法后，GPT利用数据库中的文档，对问题采取了更为精确地回复。

On 24 February 2022, Russia invaded Ukraine in an escalation of the Russo-Ukrainian War which began in 2014. The invasion has killed tens of thousands on both sides. Russian forces have been responsible for mass civilian casualties and for torturing captured Ukrainian soldiers. By June 2022, about 8 million Ukrainians had been internally displaced. More than 8.2 million had fled the country by May 2023, becoming Europe's largest refugee crisis since World War II. Extensive environmental damage, widely described as ecocide, contributed to food crises worldwide.

图4：输入的文档的一部分，作为输入。

图5：由GPT4生成的回复。



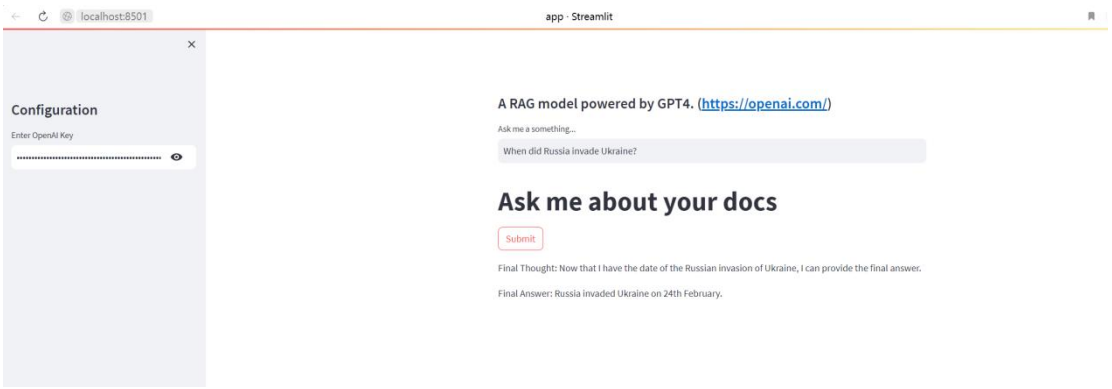
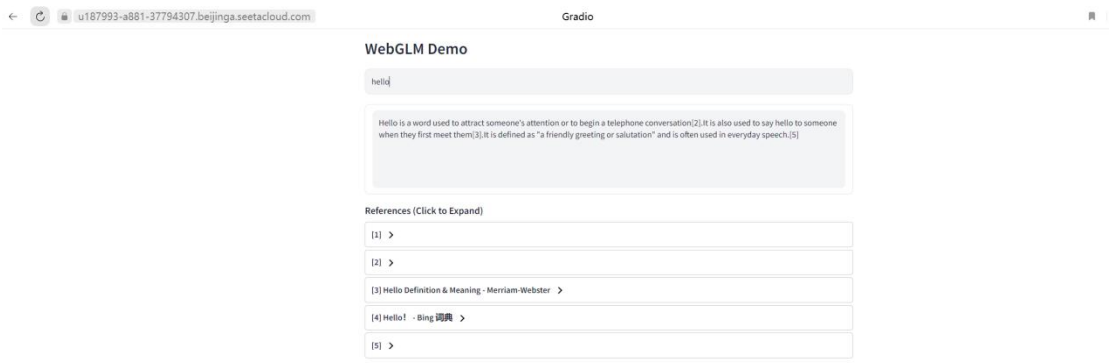


图6：经过RAG生成的回复，模型找到Observation和Thought并产生最后回复。

云端部署

在云端我选择了与检索相关的模型WebGLM，基于General Language Model (GLM)架构，部署的模型为WebGLM-2B参数的模型。考虑到计算显存有限，10B语言模型需要超过24GB的显存，因此使用较为轻量的模型，导致模型生成的内容较为生硬。但由于网络信息未经筛选，检索的错误信息可能被生成器赋予较大的注意力从而作为输出内容，这与自己建立FAISS索引不同。建立索引所提供的语料文档可以清洗，而在线检索难以筛选。在针对需要低容错率的QA任务中，推荐使用RAG模型，确保模型生成内容准确率和一致性。如若是针对紧贴时事，较为大查询量的部署时，应当选用WebGLM这类模型以减少更新索引带来的开销。



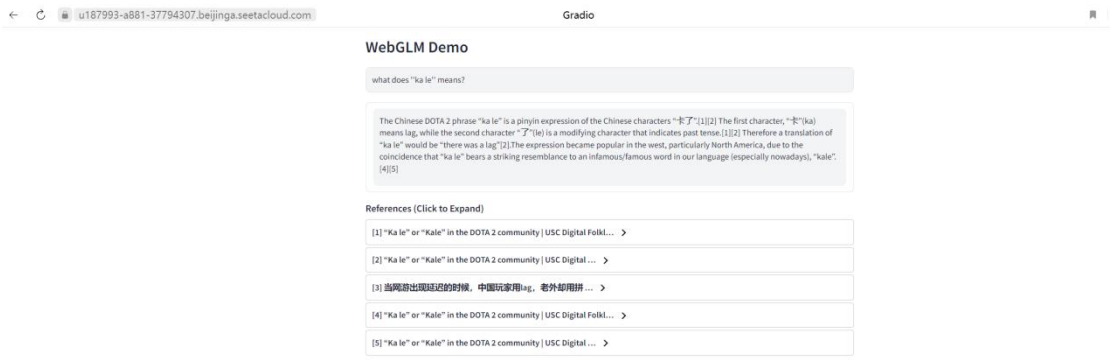
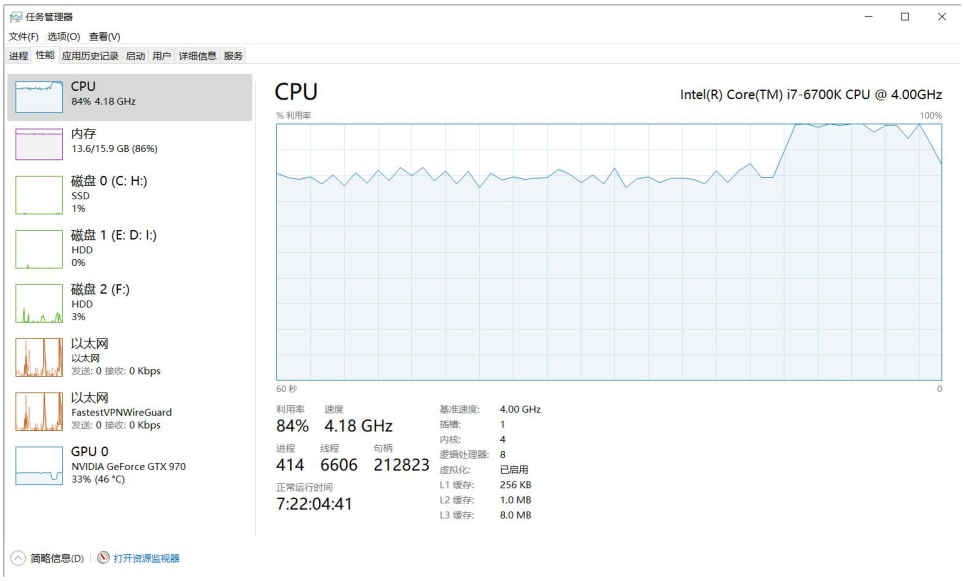


图7：WebGLM的回复。通过在线检索并提取有用的信息，再进行更为精确地回复。

6. 资源消耗

建立FAISS的索引对处理器和内存有较高要求，处理器速度影响处理文件的时间，数量以及内容较多的文档在建立索引时需要占用更多的计算机内存资源。



WebGLM-2B模型在推理时占用大概6GB左右的显存。

7. 总结

在这项工作中，分别部署测试了两种检索增强的语言模型。本文介绍了本地部署实验，包括建立索引和文档检索。建立索引时使用haystack库处理和存储文档到FAISS文档库中，检索文档阶段采用了Haystack库、RoBerta和sentence BERT来创建问题解答管道并提取确切答案。在引入RAG方法后，GPT能够更精确地回答问题。

在云端，选择了WebGLM-2B模型，虽然生成内容较生硬，但适合大查询量的部署。在线检索难以筛选，对于需要低容错率的QA任务，推荐使用RAG模型。总结，建立FAISS索引对资源要求较高，而WebGLM-2B模型在推理时占用约6GB显存。因此，应根据需求和资源选择合适的模型和部署方式。

参考文献

[1] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.

[2] 智谱清言. Available at: https://chatglm.cn/blog (Accessed: 20 August 2023).

[3] Russian invasion of Ukraine (2023) Wikipedia. Available at: https://en.wikipedia.org/wiki/Russian_invasion_of_Ukraine (Accessed: 20 August 2023).

[4] THUDM/WebGLM: WebGLM: An efficient web-enhanced question answering system (KDD 2023), GitHub. Available at: https://github.com/THUDM/WebGLM/tree/main (Accessed: 20 August 2023).

[5] AutoDL 算力云. Available at: https://www.autodl.com/ (Accessed: 20 August 2023).

[6] OpenAI. Available at: https://openai.com/ (Accessed: 20 August 2023).

[7] Haystack. Available at: https://haystack.deepset.ai/ (Accessed: 20 August 2023).

[8] Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., ... & Yih, W. T. (2020). Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

[9] Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, *7*(3), 535-547.