RQ-RAG: APRENDENDO A REFINAR CONSULTAS PARA RECUPERAÇÃO GERAÇÃO AUMENTADA

Chi-Min Chan1 , Chunpu Xu2 Ruibin Yuan1 , Hongyin Luo3 , Wei Xue1, Caramba, Guo1 †, Jie Fu1 †
[1] Universidade de Ciência e Tecnologia de Hong Kong
[2] Universidade Politécnica de Hong Kong 3
Instituto de Tecnologia de Massachusetts
zorowin123@gmail.com

RESUMO

Grandes Modelos de Linguagem (LLMs) exibem capacidades notáveis, mas são propensos a gerar respostas imprecisas ou alucinatórias. Essa limitação decorre de sua dependência de vastos conjuntos de dados de pré-treinamento, tornando-os suscetíveis a erros em cenários inéditos. Para enfrentar esses desafios, a Geração Aumentada de Recuperação (RAG) aborda essa questão incorporando documentos externos relevantes ao processo de geração de respostas, alavancando assim o conhecimento não paramétrico juntamente com as habilidades de aprendizagem em contexto dos LLMs. No entanto, as implementações de RAG existentes concentram-se principalmente na entrada inicial para recuperação de contexto, ignorando as nuances de consultas ambíguas ou complexas que exigem maior esclarecimento ou decomposição para respostas precisas. Para tanto, propomos aprender a Refinar Consulta para Geração Aumentada de Recuperação (RQ-RAG) neste artigo, buscando aprimorar o modelo equipando-o com capacidades para reescrita, decomposição e desambiguação explícitas . Nossos resultados experimentais indicam que nosso método, quando aplicado a um modelo 7B Llama2, supera o estado da arte anterior (SOTA) em uma média de 1,9% em três conjuntos de dados de QA de salto único, e também demonstra desempenho aprimorado no tratamento de conjuntos de dados de QA complexos e de salto múltiplo. Nosso código está disponível em https://github.com/chanchimin/RQ-RAG.

1 Introdução

Avanços recentes em Modelos de Grandes Linguagens (LLMs) (OpenAI, 2023; Ouyang et al., 2022; Touvron et al., 2023) demonstraram capacidades significativas na compreensão de vários conceitos e na resolução de tarefas subsequentes (Brown et al., 2020; Raffel et al., 2020). Apesar da vasta quantidade de dados utilizada durante o treinamento inicial ou nas fases subsequentes de ajuste fino, esses modelos permanecem inerentemente estáticos. Uma vez construídos e atualizados em um ponto específico no tempo, sua base de conhecimento deixa de evoluir, impedindo a incorporação de novas informações em tempo real. Essa limitação os limita a confiar exclusivamente em seu conhecimento paramétrico pré-codificado (Mallen et al., 2022) durante a inferência. Sem acesso a informações atualizadas, os LLMs são propensos a gerar alucinações (Ji et al., 2023) e podem ter dificuldades para fornecer respostas precisas e oportunas a perguntas que exigem as informações mais recentes (Vu et al., 2023).

Para superar esses desafios, a integração de funcionalidades de recuperação em modelos generativos apresenta uma solução promissora (Lewis et al., 2020; Luo et al., 2023). Essa abordagem enriquece modelos de linguagem paramétrica padrão com elementos de recuperação não paramétricos capazes de acessar informações relevantes de bancos de dados externos. Esses bancos de dados variam de repositórios abrangentes de documentos,

† Autores correspondentes.

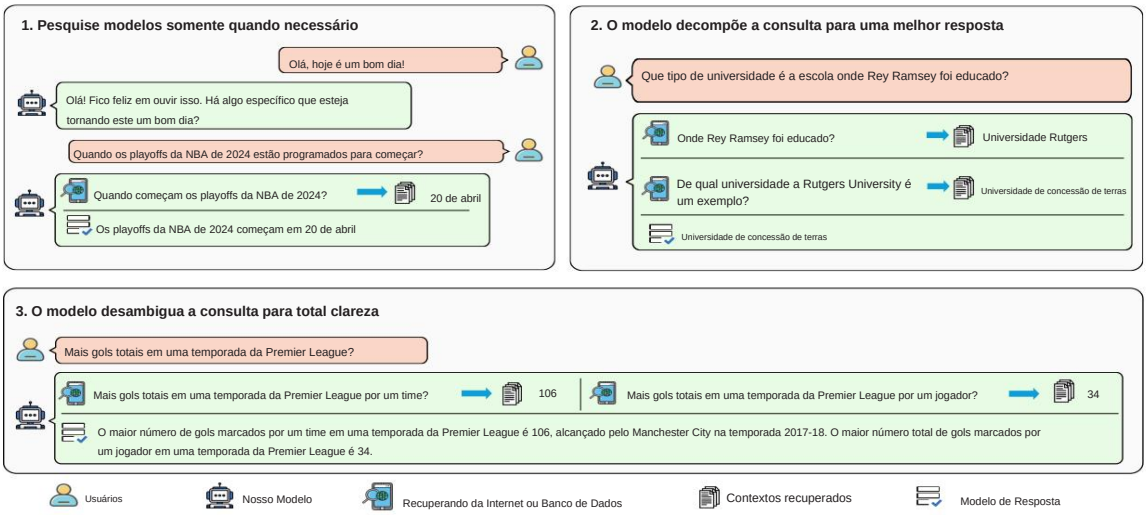

Figura 1: Nosso modelo aprendeu a pesquisar sob demanda, reescrever, decompor e desambiguar uma consulta quando necessário.

como a Wikipédia, até fontes continuamente atualizadas, como mecanismos de busca na internet, incluindo Bing[1] , DuckDuckGo[2] , entre outros.

Embora tais aprimoramentos possam atenuar a tendência a imprecisões e manter a utilidade dos LLMs em domínios em rápida evolução, ainda existem vários problemas na estrutura anterior. Em primeiro lugar, o uso indiscriminado de sistemas de recuperação de informações para contextualizar consultas pode ser contraproducente. Conforme demonstrado em pesquisas anteriores (Shi et al., 2023a), o contexto irrelevante não apenas diminui a qualidade da geração, mas também pode obstruir a capacidade dos LLMs de responder a consultas que, de outra forma, seriam capazes de abordar. Para consultas simples, como saudações diárias, os LLMs devem responder diretamente, em vez de incorporar contexto desnecessário, evitando assim respostas de baixa qualidade que poderiam dissuadir os usuários. Ou seja, o modelo deve aprender a pesquisar sob demanda, veja a Figura 1 (canto superior esquerdo). Em segundo lugar, para consultas complexas, a simples pesquisa com a consulta original muitas vezes não consegue recuperar informações adequadas. É crucial que os LLMs primeiro decomponham essas consultas em subconsultas mais simples e respondíveis e, em seguida, busquem informações relevantes para esses subcomponentes.
Ao integrar as respostas a essas subconsultas, os LLMs podem construir uma resposta abrangente à consulta original complexa. Veja a Figura 1 (canto superior direito). Por fim, para consultas ambíguas com múltiplas respostas possíveis, usar a consulta original para recuperação de informações é insuficiente. Para fornecer respostas completas e diferenciadas, os LLMs devem aprender a esclarecer a consulta, idealmente identificando a intenção do usuário, e então elaborar uma consulta de pesquisa mais direcionada. Após coletar as informações relevantes, os LLMs podem então fornecer uma resposta detalhada e abrangente, veja a Figura 1 (inferior).

Com base nos requisitos acima, propomos neste artigo **o Learning to Refine Query for Retrieval Augmented Generation (RQ-RAG) .** Treinamos um modelo 7B Llama2 de forma ponta a ponta para que ele possa refinar consultas de pesquisa dinamicamente por meio da reescrita, decomposição e esclarecimento de ambiguidades. Nosso trabalho se inspira no Self-RAG (Asai et al., 2024) e no SAIL (Luo et al., 2023), que são pioneiros na ampliação de conjuntos de dados de ajuste instrucional com resultados de pesquisa e ensinam modelos a filtrar ruídos dos resultados de pesquisa para geração de respostas baseadas em contexto.
Com base nessa base, introduzimos modificações inovadoras no processo de criação desses conjuntos de dados, aprimorando a capacidade do modelo de produzir recuperações de informações mais eficazes.
Especificamente, utilizamos o ChatGPT3 para elaborar consultas de pesquisa personalizadas em vários cenários (reescrita, decomposição, desambiguação) usando modelos de prompt distintos, em vez de depender da consulta original. Além disso, observamos casos em que a saída inicial do conjunto de dados não corresponde à

[1]bing.com
2duckduckgo.com
3Por padrão, ChatGPT se refere a **gpt-3.5-turbo-0125** e GPT-4 se refere a **gpt-4-0125-preview** (OpenAI, 2023) neste artigo.

contexto retornado pelo sistema de recuperação de informações. Nesses casos, empregamos o ChatGPT para gerar novas respostas contextualmente alinhadas, aumentando assim a relevância e a precisão do processo de recuperação de informações. Seguindo as metodologias de pesquisas anteriores (Lu et al., 2022; Keskar et al., 2019; Asai et al., 2024), empregamos tokens de controle — também conhecidos como tokens especiais — para direcionar nosso processo de geração. Com a aplicação de múltiplos tokens de controle, nosso modelo pode navegar por diversas trajetórias em resposta à consulta de um usuário. A qualquer momento, ele tem a flexibilidade de reescrever, decompor, desambiguar a consulta ou encerrar o processo de busca e prosseguir com a geração de respostas.

Para identificar a trajetória ótima como nossa resposta final, desenvolvemos meticulosamente três métodos de seleção distintos que não dependem de LLMs externos para avaliação de trajetórias (Yao et al., 2024). Esses métodos incluem seleção baseada em perplexidade (PPL), confiança e uma abordagem de conjunto, conforme elaborado na Seção 2.3. Além disso, avaliamos o limite superior do desempenho do nosso método, determinando se alguma trajetória gerada contém a resposta correta. Essa análise revela um potencial notavelmente alto para o nosso sistema, ressaltando sua eficácia se conseguirmos selecionar com precisão as trajetórias corretas.

Em suma, o nosso artigo faz várias contribuições importantes

- Primeiro, mostramos que o modelo 7B Llama2 treinado em nosso conjunto de dados elaborado supera o método de última geração anterior (Asai et al., 2024) em três tarefas de QA de salto único que avaliamos e também demonstramos desempenho superior em três tarefas de QA de vários saltos, atribuído à nossa abordagem de refinamento de consulta. •

Segundo, destacamos a eficácia da regeneração de respostas com base nos resultados da pesquisa durante a construção de dados, indo além do simples uso das saídas do conjunto de dados original. Este método se mostra mais eficaz do que aqueles empregados em trabalhos anteriores (Luo et al., 2023; Asai et al., 2024), enfatizando o valor da geração de respostas contextualmente fundamentadas. • Terceiro, mostramos o grande

potencial de nossa estrutura ilustrando seu limite superior consideravelmente alto, bem como sua resiliência a diferentes fontes de dados em comparação com métodos anteriores.

2 RQ-RAG: Aprendendo a Refinar Consultas para Recuperação de Geração Aumentada

Nesta seção, apresentamos o RQ-RAG, incluindo a seguinte parte: Construção do conjunto de dados (Seção 2.1), Treinamento do gerador (Seção 2.2) e Estratégias de amostragem (Seção 2.3).

2.1 Construção do conjunto de dados

Para treinar o modelo para refinar consultas explicitamente, o núcleo do nosso pipeline envolve a coleta de dados de treinamento que espelham o processo no momento da inferência. Isso inclui a geração de consultas refinadas para pesquisas e a elaboração de respostas com base nas informações recuperadas.

Dado um par de entrada-saída (Xorigin, Yorigin) do conjunto de dados original, nosso principal objetivo é construir uma sequência de ações incorporando tokens especiais. Esses tokens especiais especificam o tipo de refinamento SP ECIALtype e são seguidos pela consulta refinada, condicionada ao token especial específico, representado como Qi, tipo, onde 'tipo' se refere à ação de refinamento (reescrever, decompor ou desambiguar) e 'i' se refere à sequência das iterações. Posteriormente, recuperamos os k principais documentos, denotados como [Di1, Di2, ..., Dik]. Na etapa final da iteração, geramos uma nova resposta condicionada aos contextos acima, denotada como Ynew.

Em resumo, a construção do nosso conjunto de dados pode ser vista como a transformação denotada pela Equação 1.

$$(\text{Xorigin}, \text{Yorigin}) \ddot{y} (\text{Xorigin}, \underbrace{\text{SP ECIALtype}, \text{Qi}, \text{type}, [\text{Di1}, \ldots, \text{Dik}]}_{\text{repita por i vezes}}, \ldots, \text{Ynew}) \quad (1)$$

Além disso, é crucial reunir um corpus que abranja uma ampla gama de cenários, conforme descrito na Seção 1. Assim, nossa coleta de dados se concentra em cenários que incluem diálogos multi-turno, consultas que exigem decomposição e consultas que exigem desambiguação. Para uma análise abrangente das estatísticas de dados, consulte o Apêndice A.1.

Após reunir essas tarefas representativas, estabelecemos um conjunto de tarefas e prosseguimos com o processo de transformação detalhado na Equação 1. Idealmente, a execução dessa transformação envolve esforço humano para refinar consultas, conduzir pesquisas por informações relevantes e anotar as respostas refinadas. Embora essa abordagem manual garanta a qualidade, ela exige muitos recursos, tempo e é desafiadora de replicar. Para mitigar essas limitações, utilizamos os recursos avançados do ChatGPT para automatizar o processo de anotação. Para uma descrição detalhada desse processo e dos prompts utilizados, consulte o Apêndice A.2.

Conforme ilustrado na Figura 2, nosso fluxo de trabalho de anotação automatizado compreende vários estágios principais, que detalhamos passo a passo da seguinte forma:

1. Comece classificando as tarefas do conjunto coletado nas três categorias mencionadas anteriormente . Esta etapa é simples, pois cada conjunto de dados corresponde a um tipo de dado específico.
2. Para cada tipo de conjunto de dados, usamos inicialmente o ChatGPT com um modelo de prompt predefinido para gerar uma consulta refinada. Em seguida, utilizamos essa consulta para buscar informações em fontes de dados externas. Usamos principalmente o DuckDuckGo para a maioria dos casos e tratamos o processo de recuperação como uma caixa-preta.
3. Em seguida, solicitamos ao ChatGPT que gere uma resposta renovada com base nas consultas refinadas e seus contextos correspondentes. Repetindo esse processo, acumulamos um total de cerca de 40 mil instâncias.

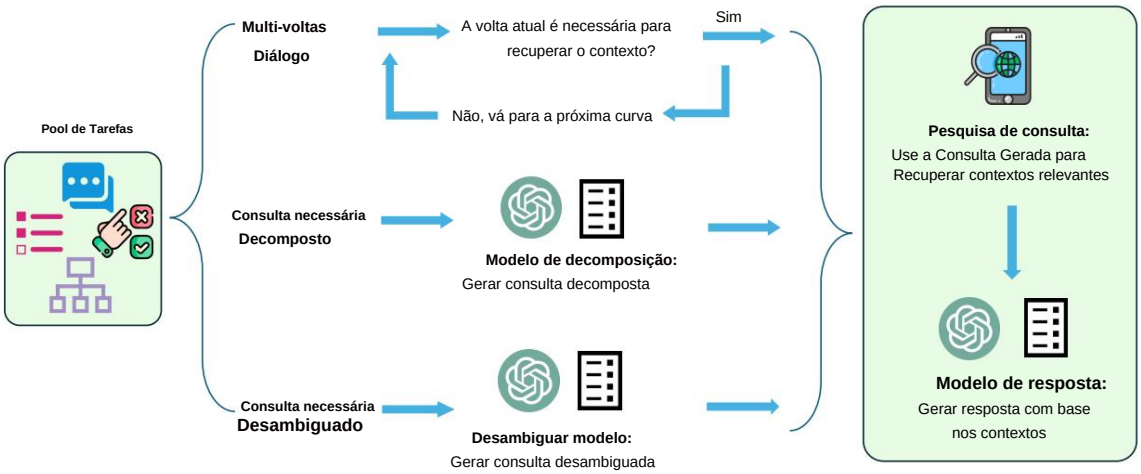

Figura 2: Pipeline de construção do conjunto de dados.

2.2 Treinamento do Gerador

Depois de anotar o corpus de treinamento, podemos usá-lo para treinar um LLM de maneira autorregressiva padrão , cujo objetivo é mostrado na Equação 2.

$$L = \max_{M} E_{(x,y)\ddot{y}D} [\log pM(y|q1, d1, \ldots, qi, , x)] \text{ onde} \quad (2)$$

onde L representa a probabilidade que estamos tentando maximizar. M denota os parâmetros do modelo. A expectativa E(x,y)ÿD é calculada em média sobre nosso conjunto de dados D. pM(y|q1, d1, . . . , qi , , x) é a probabilidade do modelo di M gerar a resposta y, dada a entrada x e a consulta refinada qi com o documento di recuperado na etapa i.

2.3 Estratégias de Amostragem

Nesta seção, apresentamos a estratégia de inferência temporal. Como mostrado na Figura 7 do Apêndice, a cada passo de tempo, nosso modelo pode optar por reescrever, decompor ou desambiguar uma determinada consulta, bem como

como optar por gerar diretamente a resposta. Dada essa natureza interna, projetamos uma estratégia de decodificação de árvore com relação a diferentes métodos de refinamento de consulta. No entanto, usar diferentes consultas para pesquisar produz diferentes contextos recuperados, o que leva a diferentes respostas finais. Ou seja, como amostrar o caminho mais apropriado entre essas trajetórias é uma parte crítica em nosso sistema. Portanto, propusemos três tipos diferentes de estratégias de amostragem, ilustradas a seguir: Usamos pM para denotar um modelo de linguagem com parâmetros M e [R1, R2, . . . , Rn] para denotar n trajetórias, onde cada trajetória

contém uma lista de sequências que é denotada como [X, Y]. Aqui, X é o prompt de entrada e Y é a concatenação dos i passos intermediários de Z1, . . . , Zi (cada Zi é a combinação de consultas e contextos recuperados) e a resposta final Yfinal.

Seleção baseada em PPL: selecionamos a trajetória Rfinal que tem a menor perplexidade (PPL) na saída total gerada, ou seja, Rfinal = arg min PPL(Rj), onde PPL(R) = Rjÿ{R1,...,Rn}

$$\text{1 experiência } \sqrt{\text{eu}} \quad \sum_{t=1}^{L} \log pM(Yt|X, Y<t)$$, aqui L é o comprimento total da saída do modelo.

Seleção baseada em confiança: selecionamos a trajetória Rfinal que tem a maior confiança na resposta final Yfinal (distinguindo-a da seleção baseada em PPL, que avalia a saída total gerada), ou seja, Rfinal = arg max Conf(Rj), onde $\text{Conf}(R) = \sum_{t=l}^{\text{eu}} \log pM(Yt\,|X, Z1, \ldots, Y<t)$, Zi Rjÿ{R1,...,Rn}

aqui, t começa em l, que é a posição inicial da resposta final, Yfinal.

Seleção baseada em conjunto: reunimos os resultados finais selecionando os resultados finais que têm a maior pontuação de confiança cumulativa, ou seja, Yfinal = arg max Conf(Yi). $\text{e}^{\text{eu: sim = sim}}$

Limite superior: também definimos o limite superior do nosso sistema, o que significa que se qualquer trajetória leva à resposta correta, então a consideramos correta.

Nossos métodos de amostragem se inspiram em Wang et al. (2022); Yao et al. (2024), mas diferem significativamente em duas áreas principais. Em primeiro lugar, diferentemente desses estudos, não empregamos um modelo maior para avaliar a qualidade das trajetórias geradas; em vez disso, utilizamos métricas inerentes ao nosso gerador treinado. Em segundo lugar, embora a abordagem de autoconsistência proposta por (Wang et al., 2022) seja limitada a cenários em que a resposta final pertence a um conjunto fixo de opções, nosso método está livre de tais restrições. Veja a Figura 7 para esclarecimento.

3 Experimentos

3.1 Tarefas de Avaliação

Avaliamos a eficácia do nosso método em duas categorias principais de tarefas de Resposta a Perguntas (QA): tarefas de QA de salto único e de salto múltiplo. Consulte o Apêndice B.3 para obter mais detalhes.

O QA de salto único inclui Arc-Challenge (Clark et al., 2018), PopQA (Mallen et al., 2022) e Open- bookQA (Mihaylov et al., 2018).

O controle de qualidade Multi-Hop inclui HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020) e Musique (Trivedi et al., 2022).

3.2 Linhas de base

Comparamos nosso método com um conjunto diversificado de linhas de base, categorizadas em dois grupos principais: **Linhas de Base Sem Recuperação,** que respondem a perguntas sem usar contextos recuperados de bancos de dados externos, e **Linhas de Base Com Recuperação,** que primeiro recuperam contextos relevantes de fontes externas antes de responder com base nesses contextos. Em ambos os cenários, nossas comparações incluem **Llama2-7B** e **Llama2-7B-Chat (Touvron** et al., 2023) em uma configuração zero-shot, bem como esses modelos ajustados em **Conjuntos de Dados Específicos de Tarefas** (por exemplo, ARC_C e OBQA para tarefas de QA de salto único, observando que POPQA não possui um conjunto de treinamento, impedindo assim o treinamento neste conjunto de dados e HOTPOTQA, 2WIKI e MUSIQUE para tarefas de QA de salto múltiplo) e **Nosso Conjunto de Dados Criado** (sem etapas intermediárias) como uma linha de base mais robusta. Além disso, avaliamos **o SAIL-7B (Luo** et al., 2023) e o **Self-RAG-7B** de última geração previamente estabelecido (Asai et al., 2024) em três conjuntos de dados de controle de qualidade de salto único.

Para as tarefas de QA multi-hop, nossas comparações se estendem à **Cadeia de Pensamento (Wei** et al., 2022) e **Cadeia de notas** (Yu et al., 2023), utilizando ChatGPT e GPT-4 como modelos de linguagem subjacentes.

4 Resultados e Análise

4.1 RQ-RAG supera Self-RAG e SAIL em tarefas de QA de salto único

Na Tabela 1, demonstramos que o RQ-RAG (nosso) supera significativamente vários modelos de base tanto em configurações de recuperação quanto de não recuperação. Especificamente, em configurações de recuperação, o RQ-RAG supera LLama2-7B (Zero Shot) em uma média de 33,5%, destacando os desafios que os LLMs enfrentam no processamento contextos recuperados sem ajuste de instrução aumentado pela pesquisa. Além disso, comparamos o RQ-RAG com duas linhas de base robustas: 1) modelos supervisionados em tarefas específicas (ARC_C, OBQA) e 2) modelos supervisionados em nosso conjunto de dados com curadoria, mas sem o intermediário aumentado pela pesquisa passo. Nossos resultados indicam ganhos significativos de desempenho com a supervisão em comparação com abordagens de tiro zero, destacando sua vantagem competitiva. Crucialmente, o RQ-RAG supera ainda mais esses linhas de base, demonstrando o valor agregado da integração de uma etapa aumentada pela pesquisa durante o treinamento.

Além disso, justapomos RQ-RAG com pesquisa supervisionada e aumentada previamente estabelecida abordagens, nomeadamente Self-RAG e SAIL. Notavelmente, o nosso método supera o SAIL-7B em 20,3% em média em três tarefas de QA. Além disso, mesmo com apenas cerca de 40 mil dados de treinamento, nosso método também supera o antigo estado da arte (Self-RAG, que utiliza 150 mil dados de treinamento supervisionado) por 1,9% em média em três tarefas de controle de qualidade.

No geral, o RQ-RAG demonstra um desempenho robusto em todas as tarefas avaliadas, estabelecendo firmemente sua superioridade sobre as linhas de base acima mencionadas.

Tabela 1: Comparação de desempenho em tarefas de controle de qualidade de salto único

Modelo	ARC_C	POPQA	OBQA	MÉD.
Linha de base sem recuperação				
LLama2-7B (Tiro Zero)	29,8	12,9	34,6	25,8
LLama2-7B-Chat (Tiro Zero)	59,9	14,1	57,6	43,9
LLama2-7B (SFT em QA de salto único)	61,3		49,6	
LLama2-7B (SFT sem conjunto aumentado)	62,0	20,5	64,0	48,8
VELA-7B	47,7	22,8	49,2	39,9
Linha de base com recuperação				
LLama2-7B (Tiro Zero)	28,7	39,8	36,2	34,9
LLama2-7B-Chat (Tiro Zero)	53,8	26,4	40,8	40,3
LLama2-7B (SFT em QA de salto único)	52,1		48,5	
LLama2-7B (SFT sem conjunto aumentado)	51,5	45,0	50,2	48,9
VELA-7B	48,4	44,0	52,0	48.1
Auto-RAG-7B	67,4	55,3	76,4	66,4
RQ-RAG (Nosso)	**68,3 (0,9ÿ)**	**57,1 (1,8ÿ)**	**79,4 (3,0ÿ)**	**68,3 (1.9ÿ)**

4.2 RQ-RAG mostra desempenho superior em conjuntos de dados de QA multi-hop

Na Tabela 2, apresentamos o desempenho do RQ-RAG em três conjuntos de dados de QA multi-hop, refletindo tendências semelhantes aos observados em cenários de QA de salto único. Quando o RQ-RAG é treinado em qualquer um dos cenários específicos conjuntos de dados ou nosso conjunto de dados selecionado (sem etapa de pesquisa aumentada), seu desempenho significativamente supera o de sua contraparte de tiro zero. No entanto, essas linhas de base não dotam o modelo com a capacidade de decompor consultas, uma habilidade crucial em cenários multi-hop onde o uso direto da consulta original para recuperação de informações muitas vezes fica aquém. Em contraste, nosso pipeline permite o modelo para refinar consultas de forma autônoma, gerando um aumento médio de 22,6% em todo o três conjuntos de dados de QA multi-hop. Além disso, quando comparados a linhas de base mais robustas que empregam ChatGPT como modelo de base para responder a perguntas, o RQ-RAG supera significativamente ambos os métodos da Cadeia de Pensamento e da Cadeia de Notas. Esta conquista é particularmente notável dado

Tabela 2: Comparação de desempenho em tarefas de controle de qualidade multi-hop.

Modelo	HOTPOTQA	2WICK	MÚSICA	MÉDIA.
LLM proprietário				
GPT-3.5-TURBO				
+ Cadeia de Pensamento	58,6	43,9	32,3	44,9
+ Cadeia de Notas	52,5	34,1	24,6	37.1
GPT-4				
+ Cadeia de Pensamento	71,4	70,1	50,3	63,9
+ Cadeia de Notas	72,4	58,3	44,1	58,3
Linha de base sem recuperação				
LLama2-7B (Tiro Zero)	6,6	16,0	3,0	8,5
LLama2-7B-Chat (Tiro Zero)	3,6	7,9	1,8	4.4
LLama2-7B (SFT em QA Multi Hop)	34,7	34,2	6,8	25.2
LLama2-7B (SFT sem conjunto aumentado)	35,6	30,8	6,7	24,4
Linha de base com recuperação				
LLama2-7B (Tiro Zero)	16,7	18,7	7,4	14.3
LLama2-7B-Chat (Tiro Zero)	2,8	3,5	1,8	2.7
LLama2-7B (SFT em QA Multi Hop)	37,5	32,3	7,9	25,9
LLama2-7B (SFT sem conjunto aumentado)	43,5	28,8	9,1	27.1
RQ-RAG (Nosso)	**62,6**	**44,8**	**41,7**	**49,7**
	(19,1ÿ)	**(16,0ÿ)**	**(32,6ÿ)**	**(22.6ÿ)**

que nosso modelo de backbone é consideravelmente menor que o ChatGPT, destacando a excepcional capacidade do RQ-RAG eficácia.

4.3 RQ-RAG mostra limite superior alto do sistema

Nas Tabelas 1 e 2, apresentamos os resultados da seleção baseada em conjuntos. Para um estudo abrangente de estratégias de amostragem, mostramos a comparação na Figura 3. Basicamente, descobrimos que para QA de salto único Em tarefas, a estratégia baseada em confiança geralmente se destaca, enquanto para tarefas de QA multi-hop, a estratégia baseada em conjunto apresenta desempenho superior. No geral, a estratégia baseada em PPL demonstra desempenho moderado. desempenho em comparação. Além dessas estratégias, também avaliamos o limite superior do nosso sistema considerando a taxa de sucesso em todas as trajetórias; o sucesso é definido como qualquer trajetória que leve para a resposta correta. Esta avaliação abrangente revela um alto potencial limite superior para nosso sistema. Especificamente, atinge 76,8% em ARC_C, 65,6% em POPQA, 84,0% em OBQA, 80,5% em HOTPOTQA, 60,6% na 2WIKI e 54,5% na MUSIQUE, respectivamente. Esses resultados reforçam a capacidade do sistema de refinar consultas e recuperar contextos variados várias vezes, aumentando assim a probabilidade de chegar à resposta correta.

Uma via para melhorias futuras envolve a concepção de métodos mais eficazes para selecionar entre trajetórias geradas, como alavancar LLMs para pontuar cada uma delas. Além disso, é importante observar que o limite superior atual não representa o limite absoluto do sistema; melhorias também poderiam ser alcançadas por meio da reclassificação de contexto e redução explícita de ruído em contextos recuperados.

4.4 Regenerar respostas baseadas em contextos é importante

Conforme mencionado na Seção 1, uma inovação fundamental do nosso método em comparação com abordagens anteriores é a uso do ChatGPT para regenerar respostas com base em contextos fornecidos, em vez de depender de contextos originais respostas do conjunto de dados. Nesta seção, exploramos o impacto da variação da proporção do resposta original retida do conjunto de dados. Para garantir uma comparação justa, mantemos uma consistência treinando hiperparâmetros em todos os experimentos. Organizamos nosso conjunto de dados ampliado por pesquisa, mantendo 0% (nossa abordagem primária), 25%, 50%, 75% e 100% da resposta original e, em seguida, avaliamos os efeitos no desempenho em três conjuntos de dados de QA multi-hop. A Figura 4 ilustra nossas descobertas sobre HotpotQA, com a configuração de resposta totalmente regenerada (0% de retenção) mostrando o melhor desempenho.

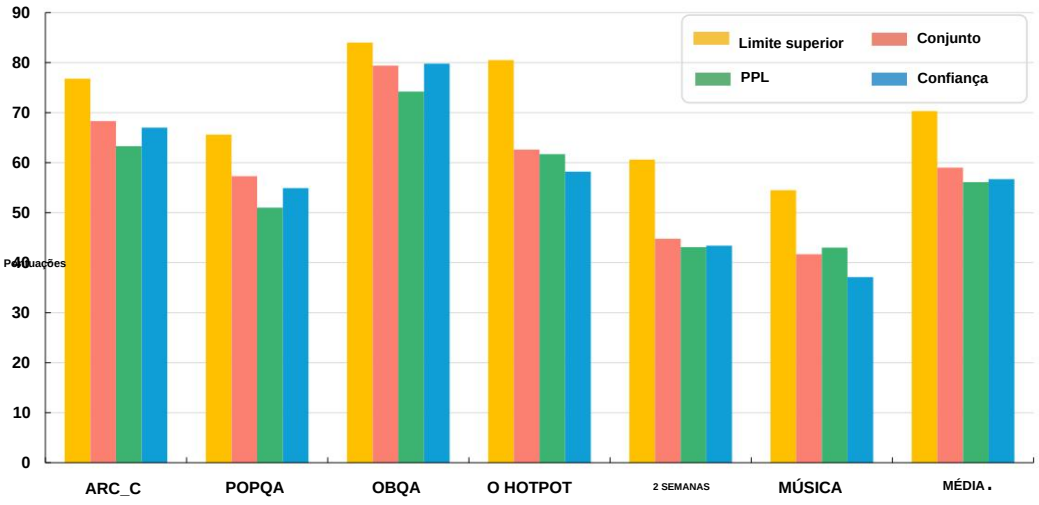

Figura 3: Desempenho de diferentes estratégias de amostragem em seis tarefas.

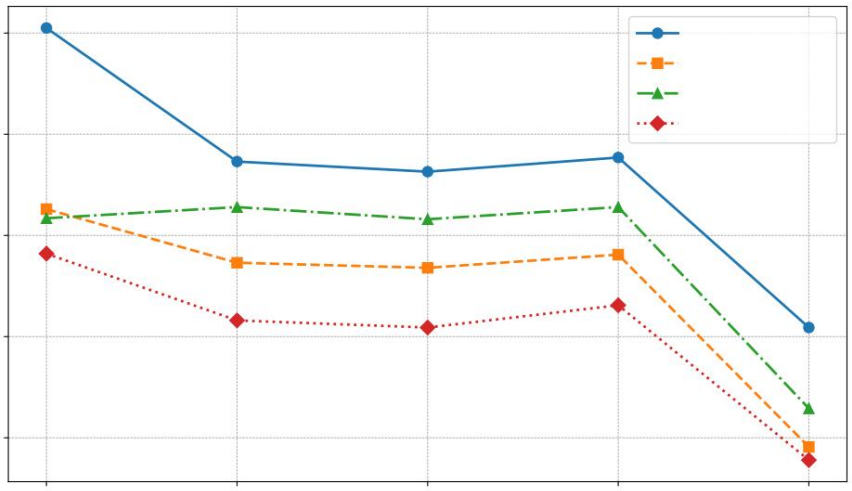

Figura 4: O impacto da taxa de retenção no HotpotQA.

Além disso, há um declínio perceptível na eficácia à medida que a proporção da resposta original retida aumenta, indicando que nossa estratégia de regenerar respostas com base nos contextos recuperados é benéfica. Apresentamos os resultados dos outros dois conjuntos de dados no Apêndice C.

4.5 Nosso sistema é resiliente a diferentes recursos de dados

Nesta seção, investigamos o impacto da utilização de diversas fontes de dados. Nosso conjunto de dados utiliza principalmente o DuckDuckGo para recuperação de dados durante a curadoria do conjunto de dados, mas nossas descobertas sugerem que a escolha das fontes de dados durante a inferência tem um efeito mínimo no desempenho do nosso sistema. Este experimento avalia nosso sistema em três tarefas de QA de salto único, utilizando DuckDuckGo, Wikipedia e Bing Search como fontes de dados durante a inferência. A Tabela 3 demonstra que nosso sistema apresenta maior resiliência a mudanças nas fontes de dados em comparação com o Self-RAG, que utiliza principalmente a Wikipedia para recuperação de dados durante a curadoria de conjuntos de dados. Notavelmente, o uso do Bing Search no momento da inferência resulta em uma queda de desempenho de 3 a 5% para o Self-RAG em todas as tarefas. Em contraste, nosso método,

curado usando DuckDuckGo, mostra impacto insignificante no desempenho (uma variação de 0,7 em comparação para seu 1.8) quando diferentes fontes de dados são usadas durante a inferência.

Método	ARC_C	POPQA	OBQA	AVG. ÿ	VAR. ÿ
Auto-RAG					
+ DuckDuckGo 67.4 + WIKI 67.3	55,3	76,4			
+ BingSearch 64.6 Nosso	54,9	78,0		65,5	1.8
	49,0	76,8			
+ DuckDuckGo 68,3 57,1 + WIKI 67,8 52,6 +		79,8			
BingSearch 67,9 55,6		80,6		**67,6**	**0,7**
		78,8			

Tabela 3: Comparação de desempenho de diferentes fontes de recuperação.

5 Trabalhos Relacionados

A geração aumentada de recuperação (RAG) representa uma estratégia eficaz para melhorar a funcionalidade de um modelo capacidade de alavancar conhecimento não paramétrico recuperando recursos de dados externos, em vez do que confiar apenas em seu conhecimento paramétrico intrínseco durante a geração. Este paradigma tem atraiu ampla atenção tanto na indústria quanto na academia, comprovando sua eficácia em uma variedade de cenários como resposta a perguntas (Lewis et al., 2020), geração de código (Zhou et al., 2022), alinhamento com valores humanos (Xu et al., 2023a) e redução de alucinações (Shuster et al., 2021). Os avanços recentes nos sistemas RAG podem geralmente ser categorizados em duas áreas: melhorias no componente de recuperação ou no componente de geração do sistema.

Retriever na Geração Aumentada de Recuperação

Estudos anteriores destacaram o papel crítico do componente de recuperação em sistemas RAG, ilustrando como os LLMs podem ser suscetíveis a contextos irrelevantes (Shi et al., 2023a). Sistemas que recuperar contextos semanticamente mais relevantes (Karpukhin et al., 2020) superam significativamente aqueles com base no BM25 (Robertson et al., 2009), demonstrando uma melhoria substancial. Mais recentemente, certos trabalhos destacaram que os próprios LLMs podem atuar como um sinal de supervisão para a formação o componente de recuperação (Shi et al., 2023b), ou mesmo servir como componente de recuperação (Sun et al., 2022), graças ao seu amplo conhecimento e capacidades robustas. Outra direção de pesquisa envolve eliminando ruído de contexto irrelevante empregando modelos adicionais para filtrar explicitamente (Yoran et al., 2023) ou comprimir (Xu et al., 2023c) tais contextos. Em nosso trabalho, abordamos o componente de recuperação como uma caixa preta, usando diretamente os contextos retornados das APIs dos mecanismos de busca ou dados pré-fornecidos corpus. Este método é compatível com técnicas de redução de ruído, o que pode melhorar ainda mais nossa desempenho do sistema; explorar essas técnicas será um caminho para pesquisas futuras.

Gerador em Recuperação Geração Aumentada

Em contraste com os esforços destinados a melhorar o componente de recuperação dos sistemas RAG, outra pesquisa concentra-se na otimização da parte do gerador. SAIL (Luo et al., 2023) treina o LLM para diferenciar contextos irrelevantes durante o processo de geração. O Self-RAG (Asai et al., 2024) treina o LLM para autorreflexivo sobre os contextos recuperados. Reescrever-Recuperar-Ler (Ma et al., 2023) treina um pequeno modelo para reescrever consultas para um leitor de caixa preta enquanto treinamos o LLM para refinar consultas por si só neste papel.

6 Conclusão

Neste artigo, apresentamos o RQ-RAG, uma estrutura que aprimora os LLMs treinando-os em um conjunto de dados meticulosamente selecionado. Este treinamento permite que os LLMs refinem as consultas por meio de um processo de reescrevendo, decompondo e desambiguando. Nossos resultados experimentais demonstram que RQ-RAG não só supera o método SOTA previamente estabelecido em três tarefas de controle de qualidade de salto único, mas também apresenta desempenho superior em cenários complexos de QA multi-hop, mesmo quando comparado ao ChatGPT excepcional, ressaltando a grande eficácia da nossa abordagem.

Referências

Asai, A., Wu, Z., Wang, Y., Sil, A. e Hajishirzi, H. (2024). Auto-reflexão: aprendendo a recuperar, gerar e criticar por meio da autorreflexão. Na Décima Segunda Conferência Internacional sobre Representações de Aprendizagem.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, JD, Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Modelos de linguagem são aprendizes de poucas tentativas. Avanços em sistemas de processamento de informações neurais, 33:1877–1901.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C. e Tafjord, O. (2018). Você acha que conseguiu resolver a questão da resposta? Experimente o arc, o desafio de raciocínio do ai2. Pré-impressão do arXiv arXiv:1803.05457.

Ho, X., Nguyen, A.-KD, Sugawara, S. e Aizawa, A. (2020). Construindo um conjunto de dados de controle de qualidade multi-hop para avaliação abrangente das etapas de raciocínio. arXiv pré-impressão arXiv:2011.01060.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, YJ, Madotto, A., e Fung, P. (2023). Pesquisa sobre alucinação na geração de linguagem natural. ACM Computing Surveys, 55(12):1–38.

Karpukhin, V., Oÿguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D. e Yih, W.-t. (2020). Recuperação de passagens densas para resposta a perguntas de domínio aberto. arXiv pré-impressão arXiv:2004.04906.

Keskar, NS, McCann, B., Varshney, LR, Xiong, C. e Socher, R. (2019). Ctrl: Um modelo de linguagem de transformador condicional para geração controlável. arXiv pré-impressão arXiv:1909.05858.

Köpf, A., Kilcher, Y., von Rütte, D., Anagnostidis, S., Tam, ZR, Stevens, K., Barhoum, A., Nguyen, D., Stanley, O., Nagyfi, R. et al. (2024). Conversas OpenAssistant — democratizando o alinhamento de modelos de linguagem em larga escala. Avanços em Sistemas de Processamento de Informação Neural, 36.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Geração aumentada por recuperação para tarefas de PNL intensivas em conhecimento. Avanços em sistemas de processamento de informações neurais, 33:9459–9474.

Lu, X., Welleck, S., Hessel, J., Jiang, L., Qin, L., West, P., Ammanabrolu, P. e Choi, Y. (2022). Quark: Geração de texto controlável com desaprendizagem reforçada. Avanços em sistemas de processamento de informações neurais , 35:27591–27609.

Luo, H., Zhang, T., Chuang, Y.-S., Gong, Y., Kim, Y., Wu, X., Meng, HM e Glass, JR (2023). Pesquise por aprendizagem por instrução aumentada. Na Conferência de 2023 sobre Métodos Empíricos em Processamento de Linguagem Natural.

Ma, X., Gong, Y., He, P., Zhao, H. e Duan, N. (2023). Reescrita de consultas para recuperação aumentada grandes modelos de linguagem. arXiv pré-impressão arXiv:2305.14283.

Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D. e Hajishirzi, H. (2022). Quando não confiar em modelos de linguagem: Investigando a eficácia de memórias paramétricas e não paramétricas. arXiv pré-impressão arXiv:2212.10511.

Mihaylov, T., Clark, P., Khot, T. e Sabharwal, A. (2018). Uma armadura pode conduzir eletricidade? novo conjunto de dados para respostas a perguntas de livros abertos. pré-impressão arXiv arXiv:1809.02789.

Mukherjee, S., Mitra, A., Jawahar, G., Agarwal, S., Palangi, H., e Awadallah, A. (2023). Orca: Aprendizado progressivo a partir de traços explicativos complexos de gpt-4. arXiv pré-impressão arXiv:2306.02707.

OpenAI (2023). Relatório técnico Gpt-4. Pré-impressão arXiv arXiv:2303.08774.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Treinamento de modelos de linguagem para seguir instruções com feedback humano. Avanços em sistemas de processamento de informações neurais, 35:27730–27744.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., e Liu, P.J. (2020). Explorando os limites da aprendizagem por transferência com um transformador unificado de texto para texto. Journal of machine learning research, 21(140):1–67.

Robertson, S., Zaragoza, H., et al. (2009). A estrutura de relevância probabilística: Bm25 e além. Fundamentos e Tendências® em Recuperação de Informação, 3(4):333–389.

Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, EH, Schärli, N. e Zhou, D. (2023a). Grandes modelos de linguagem podem ser facilmente distraídos por contextos irrelevantes. Em Conferência Internacional sobre Aprendizado de Máquina, páginas 31210–31227. PMLR.

Shi, W., Min, S., Yasunaga, M., Seo, M., James, R., Lewis, M., Zettlemoyer, L. e Yih, W.-t. (2023b). Replug: Modelos de linguagem de caixa preta com recuperação aumentada. arXiv pré-impressão arXiv:2301.12652.

Shuster, K., Poff, S., Chen, M., Kiela, D. e Weston, J. (2021). Aumento de recuperação reduz alucinações em conversas. arXiv pré-impressão arXiv:2104.07567.

Stelmakh, I., Luan, Y., Dhingra, B. e Chang, M.-W. (2022). Asqa: Perguntas factuais atendem a formato longo respostas. arXiv pré-impressão arXiv:2204.06092.

Sun, Z., Wang, X., Tay, Y., Yang, Y. e Zhou, D. (2022). Modelos de linguagem ampliados por recitação. arXiv pré-impressão arXiv:2210.01296.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P. e Hashimoto, TB (2023). Alpaca de Stanford: um modelo de lhama que segue instruções.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Base aberta e modelos de bate-papo aprimorados. arXiv pré-impressão arXiv:2307.09288.

Trivedi, H., Balasubramanian, N., Khot, T. e Sabharwal, A. (2022). Música: Perguntas multi-salto via composição de perguntas de salto único. Transactions of the Association for Computational Linguistics, 10:539–554.

Trivedi, H., Balasubramanian, N., Khot, T. e Sabharwal, A. (2023). Recuperação intercalada com raciocínio em cadeia de pensamento para questões multietapas com uso intensivo de conhecimento. 61ª Reunião Anual da Associação de Linguística Computacional.

Vu, T., Iyyer, M., Wang, X., Constant, N., Wei, J., Wei, J., Tar, C., Sung, Y.-H., Zhou, D., Le, Q., e outros. (2023). Freshllms: Atualizando grandes modelos de linguagem com aumento de mecanismos de busca. arXiv pré-impressão arXiv:2310.03214.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., e Zhou, D. (2022). A autoconsistência melhora o raciocínio da cadeia de pensamento em modelos de linguagem. arXiv pré-impressão arXiv:2203.11171.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, QV, Zhou, D., et al. (2022). A estimulação da cadeia de pensamento estimula o raciocínio em modelos de linguagem abrangentes. Avanços em sistemas de processamento de informações neurais, 35:24824–24837.

Xu, C., Chern, S., Chern, E., Zhang, G., Wang, Z., Liu, R., Li, J., Fu, J. e Liu, P. (2023a). Alinhamento em tempo real: Adaptando o comportamento do chatbot às normas estabelecidas. arXiv pré-impressão arXiv:2312.15907.

Xu, C., Sun, Q., Zheng, K., Geng, X., Zhao, P., Feng, J., Tao, C. e Jiang, D. (2023b). Wizardlm: Capacitando grandes modelos de linguagem para seguir instruções complexas. arXiv pré-impressão arXiv:2304.12244.

Xu, F., Shi, W. e Choi, E. (2023c). Recomp: Aprimorando filmes com recuperação aumentada por compressão e aumento seletivo. arXiv pré-impressão arXiv:2310.04408.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, WW, Salakhutdinov, R. e Manning, CD (2018). Hotpotqa: Um conjunto de dados para respostas a perguntas multi-salto diversas e explicáveis. arXiv pré-impressão arXiv:1809.09600.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y. e Narasimhan, K. (2024). Árvore de pensamentos: Resolução deliberada de problemas com grandes modelos de linguagem. Avanços em Sistemas de Processamento de Informação Neural , 36.

Yoran, O., Wolfson, T., Ram, O. e Berant, J. (2023). Criando modelos de linguagem com recuperação aumentada robusto a contexto irrelevante. arXiv pré-impressão arXiv:2310.01558.

Yu, W., Zhang, H., Pan, X., Ma, K., Wang, H. e Yu, D. (2023). Cadeia de notas: aumentando a robustez em modelos de linguagem de recuperação aumentada. arXiv pré-impressão arXiv:2311.09210.

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., Ma, X., Efrat, A., Yu, P., Yu, L., et al. (2024). Lima: Menos é mais para alinhamento. Avanços em Sistemas de Processamento de Informação Neural, 36.

Zhou, S., Alon, U., Xu, FF, Wang, Z., Jiang, Z. e Neubig, G. (2022). Docprompting: Gerando código recuperando os documentos. arXiv pré-impressão arXiv:2207.05987.

Uma coleta de dados

A.1 Estatísticas de dados

As Figuras A mostram as categorias que incluímos em nosso processo de curadoria de dados e o conjunto de dados específico que usamos. Nossos dados contêm principalmente tarefas de QA de salto único, incluindo Arc-Easy/Arc-Challenge (Clark et al., 2018), OpenbookQA (Mihaylov et al., 2018), tarefas de QA de salto múltiplo, incluindo HotpotQA (Yang et al., 2018), Musique (Trivedi et al., 2022), tarefas ambíguas, incluindo ASQA (Stelmakh et al., 2022). Para equipar nosso modelo com recursos gerais, também adicionamos tarefas de acompanhamento de instruções, incluindo LIMA (Zhou et al., 2024), WizardLM (Xu et al., 2023b), Open-Orca (Mukherjee et al., 2023), OpenAssistant (Köpf et al., 2024) e GPT4-Alpaca (Taori et al., 2023). No total, coletamos 42.810 instâncias.

Além disso, como ilustrado na Figura 6, a distribuição de tokens por densidade de probabilidade é significativamente alterada pelo aumento da busca. Nosso conjunto de dados aumentado exibe um aumento considerável na contagem de tokens, com a maioria das instâncias abrangendo de 150 a 2.000 tokens. Em contraste, o conjunto de dados original apresenta um comprimento notavelmente menor, com a maioria das entradas abaixo de 200 tokens.

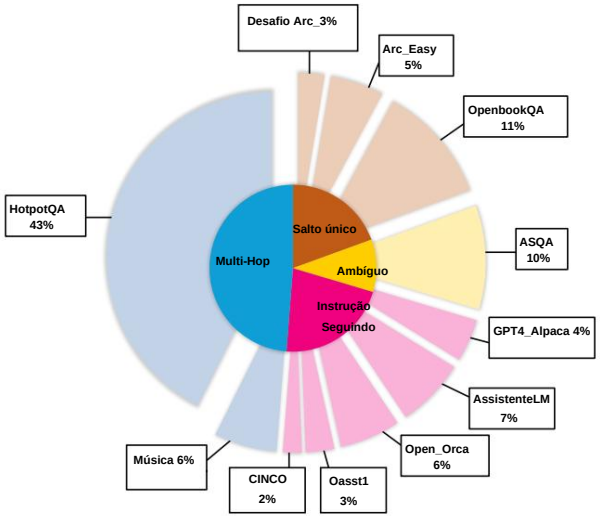

Figura 5: Categorias de dados.

A.2 Anotação de Dados

Em nosso estudo, utilizamos o ChatGPT, especificamente a versão **gpt-3.5-turbo-0125 ,** para anotação de dados, definindo o parâmetro de temperatura como 0 para melhorar a reprodutibilidade. Ao longo do processo de anotação, encontramos certas limitações com o ChatGPT, incluindo sua recusa ocasional em responder e a falha em aderir ao formato de saída especificado. As instâncias que apresentaram esses problemas foram posteriormente excluídas.

As Tabelas 4, 5 e 6 exibem os modelos de prompt que usamos para interagir com o ChatGPT em diferentes fases do processo de anotação, conforme detalhado na Seção 2. Nesses modelos, o texto azul serve como um espaço reservado para uma entrada específica.

B Detalhes Experimentais

B.1 Hiperparâmetros de treinamento

Neste artigo, todos os modelos foram treinados em 8 placas NVIDIA H800 com 80 GB de memória. Todos os modelos foram treinados em 1 época, usando uma taxa de aprendizado de 2e-5 com etapas de aquecimento de 3%. Dado o comprimento de contexto estendido do nosso conjunto de dados, definimos o comprimento máximo de entrada em 4096.

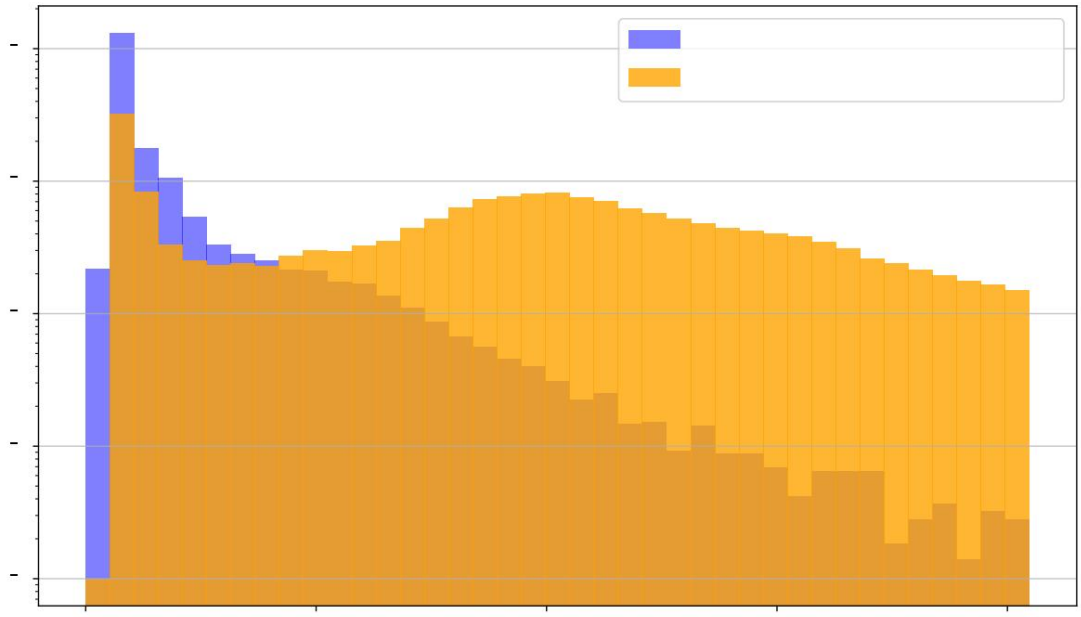

Figura 6: Distribuição de tokens.

Em um cenário de diálogo multi-turno, sua tarefa é determinar se é necessário usar um mecanismo de busca para responder à consulta de um usuário e fornecer uma lista de possíveis consultas de pesquisa.
Considere os dois cenários a seguir: Respostas
não informativas: Às vezes, os usuários podem responder com declarações ou expressões que não exigem recuperação de informações, como "obrigado" ou "ok". Nesses casos, avalie se uma consulta em um mecanismo de busca é necessária.
Consultas ambíguas ou pouco claras: Às vezes, a consulta de um usuário pode não ser clara ou carecer de detalhes específicos. Sua função é reconhecer a intenção do usuário e reescrever a consulta para torná-la mais clara e precisa, facilitando uma consulta eficaz nos mecanismos de busca.
Perguntas respondidas anteriormente: Verifique se a pergunta atual ou uma semelhante já foi feita e respondida no histórico da conversa. Se informações ou evidências relevantes já tiverem sido fornecidas, confirme e evite repetir a busca.
Com base nos 3 cenários acima, responda estritamente no seguinte formato: Para o caso de não ser necessário consultar o mecanismo de busca, a saída deve ser a seguinte: {Exemplos em contexto}

—

Conforme descrito, é necessário gerar primeiro a Necessidade de Recuperação, e a saída deve ser "sim" e "não", e a consulta ao mecanismo de busca deve ser dividida por uma quebra de linha. Na maioria dos casos, o processo de recuperação pode ajudar a responder melhor à pergunta; pule o processo de recuperação somente quando tiver certeza de que não fará isso.
Agora, por favor, responda:
Histórico de conversas:
{Histórico de conversas}
Consulta do usuário atual:
{Consulta do usuário atual}
Resposta para necessidade de recuperação:

Tabela 4: Modelo de prompt de diálogo Multi-Turn durante a construção de dados.

Sua tarefa é decompor com eficácia perguntas complexas e multissaltos em subperguntas ou tarefas mais simples e gerenciáveis. Esse processo envolve dividir uma pergunta que requer informações de várias fontes ou etapas em perguntas menores e mais diretas que podem ser respondidas individualmente.

Veja como você deve abordar isso: Analise a pergunta: Leia atentamente a pergunta multi-salto para entender seus diferentes componentes. Identifique quais informações específicas são necessárias para responder à pergunta principal.

Aqui está um exemplo de como você deve resolver a tarefa:

{Exemplos em contexto}

Conforme descrito, formate sua resposta em várias linhas de texto. Certifique-se de que cada pergunta subsequente seja uma continuação da anterior, seja independente e possa ser respondida por si só. Certifique-se de que haja exatamente uma quebra de linha entre cada linha.

Agora, por favor, responda: Contextos fornecidos: {Contextos fornecidos}

Pergunta Multihop:

{Pergunta Multihop}

Consultas decompostas:

Tabela 5: Modelo de prompt de decomposição de consulta durante a construção de dados.

Sua tarefa é identificar e resolver ambiguidades em perguntas complexas, garantindo que sejam claras e inequívocas. Para isso, é preciso identificar elementos da pergunta que podem ser interpretados de mais de uma maneira e refinar a pergunta para garantir uma interpretação única e clara.

Aborde esta tarefa da seguinte forma:

Analise a pergunta: Leia a pergunta atentamente para identificar as partes ambíguas. Considere as diferentes maneiras pelas quais a pergunta poderia ser interpretada com base em sua formulação atual.

Esclareça a pergunta: reformule a pergunta para eliminar ambiguidades. Isso pode envolver especificar detalhes, restringir termos gerais ou fornecer contexto adicional para orientar a interpretação.

Aqui está um exemplo de como concluir a tarefa: Por exemplo:

{Exemplos em contexto}

Conforme descrito, formate sua resposta como várias linhas de texto. Certifique-se de que haja exatamente uma quebra de linha entre cada linha.

Agora, por favor, responda:

Pergunta original:

{Pergunta original}

Consulta desambiguada:

Tabela 6: Modelo de prompt de desambiguação de consulta durante a construção de dados.

B.2 Estratégias de Amostragem

A Figura 7 ilustra nossa estratégia de decodificação em árvore por meio de um fluxo estruturado. Essa estratégia se desdobra controlando caminhos de expansão por meio de tokens especiais, gerando e recuperando consultas iterativamente: um processo de gerar ÿ recuperar ÿ gerar ÿ recuperar ÿ... ÿ responder. A cada iteração, o modelo decodifica diversas consultas de pesquisa adaptadas a necessidades específicas — seja para reescrever, decompor ou desambiguar.
Essas consultas, por sua vez, buscam contextos distintos, levando a caminhos de expansão variados. Assim, dentro de uma amplitude e profundidade de exploração predeterminadas, nossa abordagem facilita a geração de múltiplas trajetórias. Dada essa diversidade, torna-se imperativo adotar um método para amostrar com precisão a trajetória ótima, conforme discutido na Seção 2. Experimentalmente, definimos a profundidade de exploração como 2 para três tarefas de QA de salto único e a ajustamos para 2 para HotpotQA e 4 para 2WikiMultihopQA e MuSiQue, respectivamente.

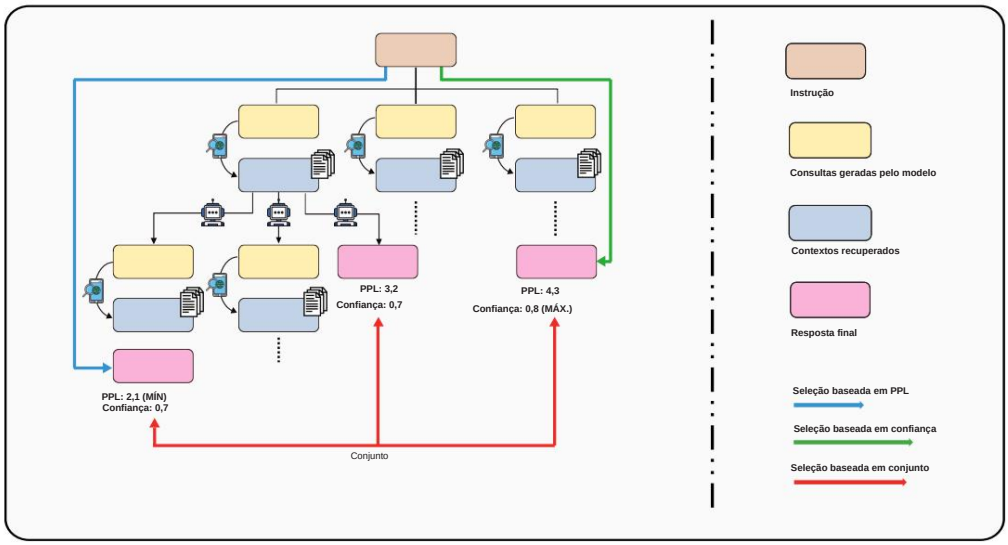

Figura 7: Considerando diferentes caminhos, desenvolvemos três estratégias diferentes: seleção baseada em pessoas, seleção baseada em confiança e seleção baseada em conjunto.

B.3 Conjuntos de dados de avaliação e métricas

QA de Salto Único:

Para o Arc-Challenge (Mallen et al., 2022), que compreende 1.172 instâncias de QA de quatro opções, medimos o desempenho do modelo usando a precisão. No caso do PopQA (Mihaylov et al., 2018), focamos em um subconjunto de cauda longa de 1.399 instâncias e adotamos uma métrica de pontuação de correspondência para avaliar se a saída do modelo inclui as verdades básicas. Para o OpenbookQA (Mihaylov et al., 2018), que compreende 500 instâncias de QA de quatro opções , medimos o desempenho do modelo usando a precisão.

QA Multi-Hop:

Avaliamos a QA multi-hop em uma amostra de 500 instâncias seguindo Trivedi et al. (2023). Nossos experimentos utilizam conjuntos de dados do HotpotQA-distractor (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020) e MuSiQue-Ans (Trivedi et al., 2022) em um ambiente de compreensão de leitura, onde os documentos candidatos são de seus conjuntos de dados originais. Para HotpotQA e 2WikiMultihopQA, cada questão está vinculada a 10 passagens, das quais apenas algumas (2 para HotpotQA e 2 ou 4 para 2WikiMultihopQA) são relevantes. O MuSiQue-Ans apresenta um desafio maior, oferecendo 20 documentos candidatos e apresentando questões que exigem 2, 3 ou 4 saltos para serem respondidas. Empregamos a pontuação F1 como nossa principal métrica de desempenho.

B.4 Configuração do Retriever

Durante a curadoria de dados, utilizamos principalmente o DuckDuckGo para reunir contextos relevantes para a maioria dos cenários, selecionando os 3 principais documentos. Cada resultado inclui um título, um snippet de visualização e o

URL da página da web. Para simplificar nosso processo, utilizamos apenas o título e o texto de pré-visualização, sem extrair mais dados da URL. Para tarefas de QA multi-hop, utilizamos o BM25 para extrair contextos de documentos candidatos fornecidos por conjuntos de dados específicos, descartando instâncias em que os contextos não se alinham com os documentos de suporte.

Durante a inferência para tarefas de QA de salto único, nossa abordagem padrão envolve a recuperação de três contextos do DuckDuckGo. Para cenários de QA de saltos múltiplos, utilizamos o modelo **text-embedding-3-large** (OpenAI, 2023) para identificar os três contextos mais relevantes dos documentos candidatos em cada etapa.

C Resultados adicionais

Nesta seção, mostramos o impacto da taxa de retenção em outras duas tarefas de QA multi-hop. Conforme discutido na Seção 4.4, fenômenos semelhantes são observados nas Figuras 8 e 9, indicando que a regeneração de respostas com base nos contextos fornecidos é crucial, o que, por sua vez, afetará o desempenho das tarefas subsequentes.

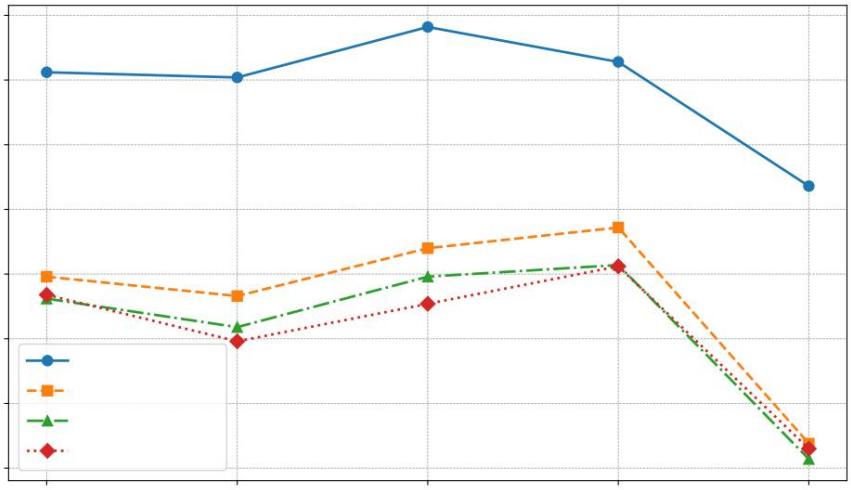

Figura 8: O impacto da taxa de retenção no 2WikiMultihopQA.

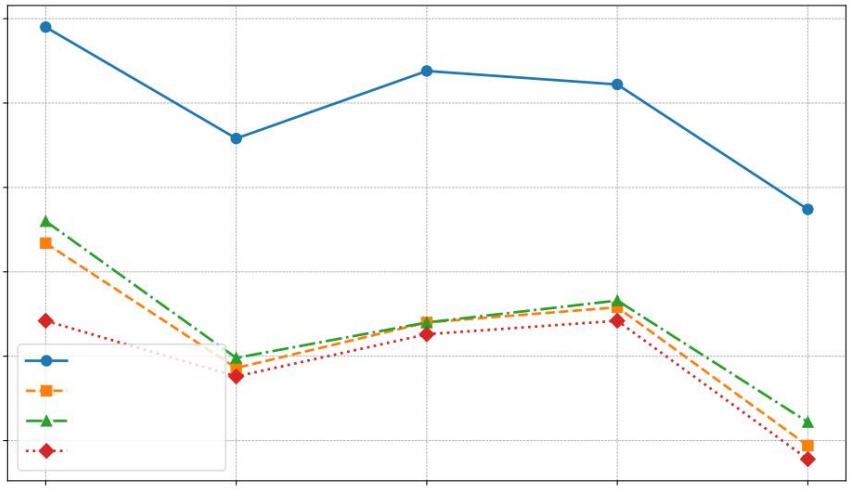

Figura 9: O impacto da taxa de retenção no MuSiQue.