# An Entity Linking Agent for Question Answering

**Yajie Luo[1*], Yihong Wu[1*], Muzhi Li[2], Fengran Mo[1], Jia Ao Sun[1], Xinyu Wang[3], Liheng Ma[3, 4], Yingxue Zhang[5], Jian-Yun Nie[1]**

[1]Université de Montréal
[2]The Chinese University of Hong Kong
[3]McGill University
[4]Mila - Quebec AI Institute
[5]Huawei Noah's Ark Lab
{yajie.luo, yihong.wu, fengran.mo, jia.ao.sun, jian-yun.nie}@umontreal.ca, mzli@cse.cuhk.edu.hk,
{xinyu.wang5, liheng.ma }@mail.mcgill.ca, yingxue.zhang@huawei.com

## Abstract

Some Question Answering (QA) systems rely on knowledge bases (KBs) to provide accurate answers. Entity Linking (EL) plays a critical role in linking natural language mentions to KB entries. However, most existing EL methods are designed for long contexts and do not perform well on short, ambiguous user questions in QA tasks. We propose an entity linking agent for QA, based on a Large Language Model that simulates human cognitive workflows. The agent actively identifies entity mentions, retrieves candidate entities, and makes decision. To verify the effectiveness of our agent, we conduct two experiments: tool-based entity linking and QA task evaluation. The results confirm the robustness and effectiveness of our agent.

## Introduction

Most Question Answering (QA) systems follow the retrieve-then-answer paradigm, where they retrieve information from a Knowledge Base (KB) to answer a given question (Mavi et al. 2024). Common KBs include textual corpora and knowledge graphs (Hogan et al. 2021). In addition to well-established, public, general KBs such as Wikipedia (Vrandečić and Krötzsch 2014), Wikidata (Vrandečić and Krötzsch 2014) and Freebase (Bollacker et al. 2008a), a recent trend involves constructing KBs from local, private, and domain-specific knowledge (Edge et al. 2024). Consequently, interfacing natural language with KBs is essential.

Entity Linking (EL) is the task of connecting natural language to KBs by mapping textual mentions to their corresponding entities within a KB (Mihalcea and Csomai 2007). Traditionally, EL systems consist of two main components: Mention Detection (MD) and Entity Disambiguation (ED) (Zhang, Hua, and Stratos 2021). MD, also known as Named Entity Recognition (NER) (Al-Moslmi et al. 2020), identifies potential entities from text. Subsequently, ED links these mentions to their unique entries in the KB (Pershina, He, and Grishman 2015).

Early approaches to EL generally follow a "detect-then-disambiguate" pipeline, first employing off-the-shelf MD systems before focusing on the ED task (Hoffart et al. 2011; Van Hulst et al. 2020). They impractically assumed all detected mentions need to be addressed, not considering the error propagation. Moreover, this separation ignores the latent dependencies between the two sub-tasks, leading to suboptimal performance. To address this limitation, researchers have shifted towards joint methods. One line of research involves end-to-end systems (Kolitsas, Ganea, and Hofmann 2018; Fang et al. 2019; De Cao et al. 2020). Another approach reformulates EL as a QA problem, leveraging the retriever-reader paradigm (Chen et al. 2017). In this framework, the retriever performs a role analogous to MD by proposing potential entity candidates, while the reader executes ED by selecting the correct entity from those candidates (Wu et al. 2019; Zhang, Hua, and Stratos 2021; Orlando et al. 2024; Xiao et al. 2023).

However, the previously mentioned methods are designed for EL in long-form documents. In contrast, this work focuses on EL for QA (Li et al. 2020; Shavarani and Sarkar 2025), which requires linking entities within the concise text of a user's query. The challenges in this QA setting are distinct; queries are typically short, ambiguous, and lack the rich context. This necessitates reasoning over implicit type constraints (Li et al. 2025) and background knowledge not explicitly provided in the question. Therefore, these unique constraints demand a novel EL method tailored for QA.

To address these challenges, we propose a LLM-based (Large Language Model) entity linking agent, inspired by recent advances in LLM agents (Liu et al. 2025; Wu et al. 2025). Our approach models the human cognitive workflow for the EL task, where a person identifies potential entities, uses search tools (e.g., Wikidata, Google Knowledge Graph[1] provide entity search engines) to find candidates, and then disambiguates them based on the available context. Our EL agent is designed to replicate this process. To achieve the target, the system must learn to **plan** by actively selecting entity mentions, **use tools** by querying a search engine, and **make decisions** by selecting the correct entity from the search results. These capabilities align with the definition of autonomous agents (Liu et al. 2025).

Our approach offers three primary advantages over existing methodologies. First, our agent-based workflow pro-

[*]Equal contribution.

[1]https://developers.google.com/knowledge-graph

vides superior retrieval flexibility. Prevailing methods (Wu et al. 2019; Zhang, Hua, and Stratos 2021) almost exclusively use dense retrieval (Karpukhin et al. 2020). While effective, dense retrieval struggles with certain types of entity references (Sciavolino et al. 2021). In contrast, our framework is retriever-agnostic. Specifically, we prefer lexical retrievers, e.g., BM25 (Robertson and Zaragoza 2009), as their keyword-matching mechanism is particularly effective given the entity-centric nature of our task. Second, by employing an LLM as its core, our agent leverages reasoning and common sense to overcome the challenge of limited context. This capability is critical for disambiguation when explicit clues are minimal. We further leverage Chain-of-Thought (CoT) prompting (Wei et al. 2022) for an robust and interpretable agent decision-making process. Furthermore, our method is highly adaptable. It can be implemented on large-scale LLMs (e.g., GPT-4, Claude, Gemini) through few-shot learning (Brown et al. 2020) or fine-tuned on smaller models via self-play training (Zelikman et al. 2022; Wu et al. 2025). Our method's effectiveness is confirmed through extensive experimentation.

## Related Works

**Entity Linking** The task of entity linking is typically divided into two sequential sub-tasks: mention detection and entity disambiguation (Kolitsas, Ganea, and Hofmann 2018). Early works in entity linking adopted graphed-based disambiguation for consistency (Kulkarni et al. 2009; Yosef et al. 2011; Ganea et al. 2016), typically by constructing a graph over candidate entities from all mentions and performing joint inference to ensure coherent linking. With the advent of deep learning, research has shifted toward retrieval-based EL frameworks. These models encode both mention contexts and entity candidates into a shared embedding space, enabling efficient similarity-based retrieval (Sil et al. 2018; Gillick et al. 2019; Le and Titov 2019; Ganea and Hofmann 2017; Wu et al. 2019). Specifically, ELQ (Li et al. 2020) adopted a bi-encoder architecture to jointly perform mention detection and linking in one pass. RefinED (Ayoola et al. 2022) is an end-to-end entity linking model that jointly performs mention detection, fine-grained entity typing, and disambiguation in a single forward pass. Entqa (Zhang, Hua, and Stratos 2021) introduced a retrieval-and-reading framework inspired by QA, where a dense retriever first selects candidate entities, followed by a reader that verifies each entity in context. In contrast to prior methods that rely on dense retrieval, our approach is retriever-agnostic and leverages a LLM as an agent to identify mentions, utilize external tools, and make decisions.

**Question Answering** Open-domain question answering systems have traditionally employed a retrieve-and-read pipeline, first retrieving context from a large corpus before generating an answer (Chen et al. 2017). Early research in QA was largely divided into two distinct tasks: information retrieval, which focuses on finding documents that support an answer, and Machine Reading Comprehension (MRC), which involves extracting an answer from a given passage (Zeng et al. 2020). The emergence of LLMs (Chowdhery et al. 2023; Achiam et al. 2023; Touvron et al. 2023), has reduced the need for explicit MRC modules in QA tasks, due to their strong reasoning and language understanding abilities. Nonetheless, LLMs often struggle with obscure or long-tail questions, necessitating the use of external retrieval systems for augmentation (Lewis et al. 2020). However, conventional retrieval methods have persistent weaknesses. Sparse retrievers cannot fully grasp semantic meaning or support fuzzy matching (Karpukhin et al. 2020), while dense retrievers can be ineffective for entity-centric questions (Sciavolino et al. 2021). To address this gap, recent research has explored entity retrieval as an alternative approach (Shavarani and Sarkar 2024; Salnikov et al. 2023; Mohammed, Shi, and Lin 2018; Lukovnikov, Fischer, and Lehmann 2019). Following this line of work, our research investigates the application of entity linking for QA.

**LLM Agent** With the rapid progress of LLMs, recent research has introduced autonomous LLM agents, which can independently perceive goals, plan actions, interact with external tools, and make decisions without step-by-step instruction (Yao et al. 2023; Hong et al. 2023). More examples include self-supervised tool-usage learning via API calls (Schick et al. 2023), a debate-driven multi-agent framework (Du et al. 2023), self-organizing agents for open-ended tasks (Chen et al. 2024), role-based collaborative planning agents (Hao et al. 2025), all of them demonstrate the effectiveness of autonomous LLM agents in addressing complex, real-world tasks. Inspired by these advances, we design an autonomous LLM-based agent tailored for entity linking in question answering, capable of identifying entity mentions, retrieving relevant candidates, and making context-aware linking decisions.

## Methodology

### Task Definition

A knowledge base (Lan et al. 2021) is a repository of information, such as knowledge graphs (Hogan et al. 2021) or a text corpus. An entity is a unique object within a KB. Let $\mathcal{T}$ denote the text space. Given a document $d \in \mathcal{T}$ and a KB $\mathcal{K}$, the task of **entity linking** (Shen, Wang, and Han 2015) is to identify entities $e_i \in \mathcal{K}$ mentioned in $d$. This task is typically decomposed into two sub-tasks: mention detection and entity disambiguation. MD identifies text spans, or potential mentions $m_i \in \mathcal{T}$, within document $d$ that may refer to an entity. ED maps each detected mention $m_i$ to its corresponding entity $e_i$ in the knowledge base $\mathcal{K}$.

For this work, we assume $\mathcal{K}$ is sufficiently comprehensive to contain all relevant entities. For mentions that do not have a corresponding entity in the KB (i.e., false mentions), we allow the target entity an empty entity (Zhang, Hua, and Stratos 2021).

### Entity Linking Agents

This work focuses on EL within the context of question answering, who aims to develop systems providing direct answers to user queries (Choi et al. 2018). In contrast to traditional EL, which targets long-form, context-rich documents,

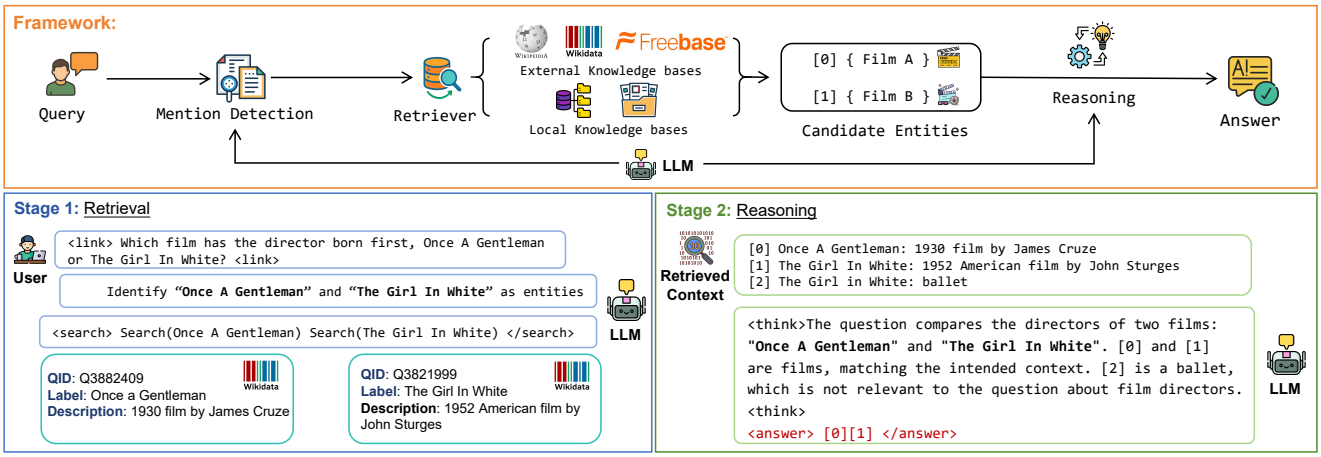


Figure 1: Entity Linking Agent (ELA) Framework.

EL for QA concentrate on short, ambiguous queries. In this scenario, the objective is not to identify every possible entity, but rather to pinpoint the entities of interest, who are crucial for resolving the query. The challenge is maintaining high accuracy despite the limited contextual information. This necessitates new EL methods with advanced language understanding capabilities.

To address this challenge, we propose **ELA** (Entity Linking Agent), an LLM-based agent for entity linking. This agent leverages the inherent language understanding and reasoning capabilities of LLMs to identify entity mentions and select contextually relevant candidates. As illustrated in Figure 1, our agent's workflow mimics the human approach to this task. Given a query $q \in \mathcal{T}$, the agent first identifies potential entity mentions $\{m_i\}$. It then utilizes a search tool `Search()` to retrieve a Top-K list of candidate entities for each mention. Subsequently, the agent select the most appropriate entity from each list based on the query $q$ and the other retrieved candidates. This workflow aligns with the retriever-reader paradigm, which is widely adopted in prior works (Wu et al. 2019; Zhang, Hua, and Stratos 2021). The following sections detail each component of this agent.

**Retrieval** The retrieval stage comprises two tasks: mention recognition and tool use. Given a query $q$, the agent needs to identify a set of potential mentions, $\{m_i\}$, within the text. Each mention is defined as a span of text within $q$. For example, in the query, *"Which film has the director born first, Once A Gentleman or The Girl In White?"*, the agent should identify *"Once A Gentleman"* and *"The Girl In White"* as mentions, as these named entities are critical for answering the question. For the tool-use task, the agent is instructed to use the detected mentions as inputs for a search engine. To achieve this, the LLM is prompted to generate its output in a structured format, `Search(X)`, where `X` is the mention to be searched. This format allows for straightforward parsing using regular expressions. For efficiency, multiple mentions are processed in a single forward pass of the LLM. The specific instructions and a complete example are provided in Appendix.

A key feature of our workflow is that it is retriever-agnostic. While prior works often relied on dense retrievers (Wu et al. 2019; Zhang, Hua, and Stratos 2021), such models can exhibit limitations in precise entity recognition—a critical function in EL (Sciavolino et al. 2021)—despite their strength in capturing semantic meaning. Our design, therefore, is not bound to a specific retriever, which offers significant flexibility. The framework can accommodate any search backend, from public APIs like Wikidata or the Google Knowledge Graph to specialized retrievers. The optimal choice depends on the nature of the mentions. For explicit mentions that act as strong lexical signals, sparse retrievers like BM25 (Robertson and Zaragoza 2009) may be preferred. Conversely, when entities are mentioned implicitly, dense retrievers are likely to be more effective due to their semantic capabilities.

**Reader** In the reader stage, the agent selects relevant entities $e_i$ from a Top-K candidate set $\{c_j\}_{j=1}^{K}$ for each mention, based on the query $q$ and retrieved context. Each candidate $c_j$ comprises a title and a short description (e.g., *"The girl in white: 1952 American film by John Sturges"*). The agent evaluates these candidates and identifies semantically relevant matches, necessitating an understanding of both the query intent and entity descriptions.

Rather than directly prompting the LLM for answers, we employ CoT reasoning (Wei et al. 2022). Prior work (Guo et al. 2025; Jaech et al. 2024) demonstrates that CoT enhances model reasoning and problem-solving capabilities. In our framework, the CoT rationale resembles: *"The query concerns the director of two films... While both [x] and [y] reference 'the girl in white,' [x] corresponds to a Van Gogh painting..."* Additionally, CoT provides partial explainability, aiding error diagnosis and agent decision analysis. The agent's CoT outputs are encapsulated within the `<think>` tag.

**Implementation** Despite LLMs' versatility, we still need to instruct them to follow the predefined workflow. In-

context learning (Brown et al. 2020) is a way to let LLM follow the instruction without training. We prepare a few examples and inject them via text prompts so that LLMs know how to response in different cases. This method is ad-hoc, easy to implement; however, it might suffer in some cases like that a small size models cannot completely understand the instructions or that those few-shot prompts seriously slow the inference speed.

To mitigate these issues, we introduce a lightweight RL-style self-play fine-tuning method (Zelikman et al. 2022). First, we generate trajectories (query, CoT, answer) via in-context learning, then filter for those with exact matches to gold entities. These high-quality trajectories train the LLM using cross-entropy loss (Radford et al. 2018), enabling smaller models to achieve competitive performance. Post fine-tuning, few-shot examples can be omitted during inference, eliminating their computational overhead without sacrificing accuracy.

LLM-based agents are often criticized for computational inefficiency, but recent advances in AI infrastructure alleviate this concern. Leveraging optimizations like vLLM (Kwon et al. 2023), our system processes 1,000 queries in under 5 minutes on a single `NVIDIA H100` GPU for a 7B model. Further improvement is achievable with multi-GPU deployment or next-generation hardware.

### Entity Linking Meets Question Answering

Traditional information retrieval is dominated by two paradigms: sparse and dense retrieval, both of which have inherent limitations. Sparse methods, such as BM25 (Robertson and Zaragoza 2009), operate on lexical matching and thus struggle with semantic nuances, failing to retrieve relevant documents that do not share exact keywords with the query. Conversely, dense retrievers (Karpukhin et al. 2020), which rely on semantic similarity, can be ineffective for entity-centric questions where precise identification is critical (Sciavolino et al. 2021).

To address these gaps, we consider EL as a complementary retrieval mechanism. The key advantage of EL is its ability to ground entity mentions in a query to their canonical entries in a knowledge base. Unlike methods that match text, EL can precisely pinpoint a specific entity, a property that is highly advantageous for downstream tasks like knowledge graph QA (Lan et al. 2021) and factoid QA (Stelmakh et al. 2022). Accordingly, we integrate our ELA agent into a complete QA pipeline to verify its effectiveness as a high-precision retrieval component.

This work also raises a fundamental question: should EL be treated as an independent, task-agnostic process, or should its objectives be defined by the downstream application? For QA, we contend that the goal of EL is to identify all "entities of interest" within a query. However, the definition of an "entity of interest" is itself ambiguous.

Consider the question: *When did Michael Jordan return to the NBA?* A traditional NER approach would identify both *Michael Jordan* and *NBA* as valid entities to be linked. From a QA perspective, however, only *Michael Jordan* is the core topic entity. Retrieving general information about the NBA is unlikely to yield the specific answer, making it a noisy entity in this context.

This distinction is critical as it directly impacts the evaluation of EL systems for QA. Most established EL benchmarks (Hoffart et al. 2011, 2012) are designed for document-level linking, where the objective is to identify every entity in a long text. A key pitfall of this approach is its failure to consider the utility for a downstream task. We argue that a successful EL system for QA must not only link entities but also distinguish between those essential for answering the question and those that are peripheral noise.

The development of such systems, however, is hindered by a scarcity of carefully constructed, QA-centric benchmarks. We therefore call for the creation of more sophisticated evaluation frameworks that measure the practical contribution of EL to downstream QA performance.

## Experiments

To evaluate the effectiveness of our entity linking agent, we conduct two experiments. The first experiment examines the agent's ability to leverage existing search tools (Tool Use) for identifying and linking entities mentioned in the input question. The second experiment applies the entity linking technique to a real-world QA task.

### Tool Use

Unlike previous works (Zhang, Hua, and Stratos 2021; Wu et al. 2019) that are tightly coupled with a specific dense retriever, our agentic method is retriever-agnostic. This design provides the flexibility to operate with any existing search tool. To demonstrate this capability, we perform entity linking using the search engines of both Wikidata (Vrandečić and Krötzsch 2014) and Wikipedia.

**Datasets** We evaluate our method on three widely-used question answering (QA) benchmarks: 2WikiMultiHopQA (2Wiki) (Ho et al. 2020), WebQSP (Yih et al. 2016), and ComplexWebQuestions (CWQ) (Talmor and Berant 2018). 2Wiki is a multi-hop QA benchmark built using Wikidata and Wikipedia. WebQSP is a single-hop dataset, while CWQ is its multi-hop extension featuring complex questions that require multi-step reasoning. In these datasets, each instance consists of a natural language question and its corresponding topic entities, which the entity linking system must identify. A challenge with WebQSP and CWQ is their reliance on Freebase (Bollacker et al. 2008b), a knowledge graph that has been deprecated. Therefore, following prior work (Li et al. 2020), we converted their original Freebase entity IDs to Wikidata QIDs using a mapping derived from the Freebase dump[2]. In addition, to satisfy our requirement, we further preprocessed the datasets. Details can be found in Appendix.

**Metrics** We use precision (Prec), recall, and accuracy (Acc) to assess the performance of our agent. Given the set of gold entities $g$ and the set of predicted entities $p$, these

[2]https://developers.google.com/freebase

| Source | Methods | 2Wiki | | | WebQSP | | | CWQ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Precision | Recall | Acc | Precision | Recall | Acc | Precision | Recall | Acc |
| Wikipedia | Azure | 92.49 | 91.00 | 88.20 | 11.10 | 10.75 | 10.30 | 78.10 | 79.07 | 60.60 |
| | ELQ (Li et al. 2020) | 75.05 | 74.35 | 72.1 | 91.05 | 92.05 | 86.60 | 81.77 | 82.35 | 68.40 |
| | ReFinED (Ayoola et al. 2022) | 76.63 | 79.40 | 69.10 | 89.93 | 90.80 | 84.90 | 81.46 | 83.48 | 68.90 |
| Wikidata | **ELA (Ours)** | | | | | | | | | |
| | w/ Llama-3.1-8B (few-shot) | 77.71 | 68.30 | 55.20 | 46.40 | 45.75 | 42.60 | 50.04 | 44.12 | 35.50 |
| | w/ Llama-3.1-8B (fine-tuned) | 84.45 | 81.40 | 76.60 | 83.49 | 81.78 | 78.25 | 78.00 | 68.02 | 55.80 |
| | w/ Llama-3.3-70B | 86.65 | 88.05 | 78.70 | 79.12 | 83.83 | 74.22 | 73.39 | 69.64 | 59.81 |
| | w/ DeepSeek-V3 | 88.03 | 88.05 | 79.70 | 86.40 | 88.20 | 83.40 | 81.44 | 82.72 | 68.40 |
| | w/ GPT-4.1 | 83.52 | 82.25 | 75.80 | 80.14 | 81.15 | 78.20 | 65.83 | 63.35 | 56.70 |
| Wikipedia | **ELA (Ours)** | | | | | | | | | |
| | w/ Llama-3.1-8B (few-shot) | 67.17 | 57.05 | 43.40 | 50.20 | 49.35 | 47.00 | 46.49 | 43.18 | 38.84 |
| | w/ Llama-3.1-8B (fine-tuned) | 89.65 | 89.35 | 86.50 | 84.52 | 82.65 | 79.20 | 78.65 | 67.75 | 54.50 |
| | w/ Llama-3.3-70B | 85.70 | 85.50 | 80.70 | 74.77 | 77.77 | 69.66 | 51.46 | 50.34 | 44.72 |
| | w/ DeepSeek-V3 | 91.92 | 91.40 | 88.00 | **91.18** | **92.55** | **88.20** | **82.43** | **84.04** | **73.05** |
| | w/ GPT-4.1 | **97.06** | **96.45** | **93.72** | 84.45 | 84.70 | 82.40 | 69.30 | 65.85 | 59.30 |

Table 1: Entity Linking Agent performance. *ELA* stands for *Entity Linking Agent*. *Bolded values* indicate the best performance for each evaluation metric. Underlined values highlight the second-best performance. Acc denotes accuracy. We omit the percent sign (%) from all results. *w/* denotes the base model used with the agent (e.g., *w/ GPT-4.1* refers to using GPT-4.1 as the backbone language model).

metrics are defined as the follows:

$$\text{Prec} = \frac{|p \cap g|}{|p|} \ , \ \text{Recall} = \frac{|p \cap g|}{|g|} \ , \ \text{Acc} = \mathbb{1}(p == g) \ ,$$

where $\mathbb{1}$ is an indicator function (1 for exact match between $g$ and $p$ 0 for otherwise). Precision measures how many predictions are correct. Recall measures how many true entities are covered by the prediction. Accuracy, a.k.a. Micro F1, is the exact match between gold and predicted entities.

**Baselines** To evaluate the effectiveness of our approach, we compare it against three representative entity linking (EL) baselines: the commercial Azure Entity Linking system[3], ELQ (Li et al. 2020), and ReFinED (Ayoola et al. 2022).

- The Azure Entity Linking system is a closed-source commercial service from Microsoft that links entity mentions to Wikipedia.
- ELQ is an end-to-end model optimized for questions, employing a bi-encoder architecture to jointly perform mention detection and linking.
- ReFinED is a high-performance linker that combines dense retrieval with a transformer-based model for efficient and accurate entity disambiguation.

We chose ELQ and ReFinED because they are specifically designed for or evaluated on QA datasets, making them more suitable than the numerous EL methods developed for document-level tasks. The Azure system was included as it has been widely adopted in prior QA research (Ma et al. 2025). We used the official code and re-implement both ELQ and ReFinED in our experiments.

[3]https://docs.azure.cn/en-us/ai-services/language-service/entity-linking/overview

**Implementation Details** We experiment with four LLMs: Llama-3.1-8B-Instruct (Llama-8B), Llama-3.3-70B-Instruct (Llama-70B), DeepSeek-V3, and GPT-4.1. Among them, Llama-8B is used for both few-shot prompting and fine-tuning. While the others are evaluated under few-shot setting with chain-of-thought prompting. To accelerate inference, we use vLLM (Kwon et al. 2023). We set the temperature to 0.7, top-p to 0.8, and repetition penalty 1.05 if applicable. For retrieval, we set $k$ to 50 for the Top-$k$ returned list. To enhance the performance of smaller model, we perform full-parameter self-play fine-tuning on the Llama-8B model. We randomly selected 3000 samples from 2Wiki training set. For each instance, we first generate (query, chain of thought, answer) trajectories via context learning and retain only 1500 examples whose predicted entities exactly matched the gold entities for fine-tuning. More training details are in Appendix.

**Results** Table 1 presents the precision, recall, and Accuracy metrics for four methods across three datasets. Our results consistently show that our proposed ELA method outperforms all three baselines-Azure, ELQ, and RefinED on the three benchmarks. Specially, on the 2wiki dataset, for example, ELA with GPT-4.1 model achieves 93.72% accuracy, significantly higher than baselines. Even on the more challenging CWQ dataset, our methods remain competitive, with DeepSeek-V3 achieving 73.05% accuracy, was about 4.15% higher than the next best ReFinED method. These

| Methods | TriviaQA | | | PopQA | | |
|---|---|---|---|---|---|---|
| | Hit@1 | EM | F1 | Hit@1 | EM | F1 |
| Naive Gen, CoT | – | 77.60 | 84.20 | – | 34.00 | 43.76 |
| BM25 | 55.88 | 75.09 | 80.61 | 62.00 | 42.10 | 50.06 |
| Dense Retrieval | | | | | | |
| w/ intfloat/e5-large-v2 | 42.03 | 72.37 | 77.53 | 72.20 | 49.10 | 58.99 |
| w/ BAAI/bge-base-en-v1.5 | 67.39 | 78.37 | 83.85 | 75.60 | 51.70 | 61.09 |
| w/ BAAI/bge-large-en-v1.5 | 69.42 | 81.23 | 86.55 | 77.20 | 52.10 | 62.06 |
| **Entity Linking (Ours)** | 64.60 | 79.90 | 85.93 | 76.50 | 53.40 | 64.05 |

Table 2: Performance of entity linking in QA. *Naive Gen* denotes naive generation with Chain-of-Thought prompting. We omit the percent sign (%) from all results. *w/* indicates the use of different dense retrievers.

results validate the effectiveness of our ELA framework for the entity linking task. Moreover, large language models—such as LLaMA-3.3-70B, DeepSeek-V3, and GPT-4.1—consistently outperform smaller models like LLaMA-3.1-8B under the same few-shot prompting setting, suggesting that larger models are better at following complex entity linking instructions. We also observed that the best and second-best performance are consistently achieved when Wikipedia is used as knowledge source. This may be attributed to the fact that Wikidata may provide many candidate entities for a given mention, introducing more noise into the disambiguation process. Wikipedia, by contrast, aligns more naturally with QA datasets. However, this difference from the inherent characteristics of the knowledge bases themselves, rather than from our proposed method.

On the WebQSP dataset, the Azure Entity linking systems perform extremely poor with the accuracy of 10.3%, despite strong performance on the other datasets. This is likely because WebQSP is entirely lowercased, while Azure system is known to be case-sensitive. In constrast, ELA maintains high performance on WebQSP, highlighting the robustness of our approach.

**In-Context Learning** We initially explored using In-Context Learning (ICL), a method where LLMs learn new tasks from a few examples without parameter updates, to guide our entity linking agent. However, our experiments with an 8B model revealed several significant challenges with this approach. First, the model frequently failed to follow our instruction, causing errors in the downstream processing pipeline. Second, it struggled with fine-grained semantic distinctions, confusing closely related but distinct entities. Third, the model exhibited a tendency to over-infer or "hallucinate" queries. For instance, after identifying a named film, it would proceed to query for related but unmentioned entities like the film's director. To overcome these limitations, we considered two alternatives. Using a larger model, such as GPT-4, could improve instruction following but would incur prohibitive inference costs. Another one is to manually refine prompts and examples that better align with the specific requirements of our task through prompt engineering. But it requires significant time and manual effort. So we propose to finetune a local model specifically for EL tasks.

**Fine-tuning** We fine-tuned a LLaMA-3 8B model using a small, high-quality dataset of only 1,500 trajectories sampled from the 2Wiki training set. The resulting model exhibits three key advantages: high performance, strong cross-dataset generalization, and remarkable data efficiency. First, despite its relatively small size, our model achieves state-of-the-art performance. Notably, on WebQSP dataset, it performs better than the much larger LLaMA-3.3-70B model (74.22%) and even slightly outperforms GPT-4.1 (78.20%) with an accuracy of 78.25% in wikidata source. In addition to accuracy, the 8B model achieves much faster inference than larger models. Second, the model shows strong generalization capabilities. Although fine-tuned exclusively on 2Wiki, it achieves competitive results on both the WebQSP and CWQ datasets. Finally, this performance is achieved with exceptional data efficiency. The success with only 1,500 training examples underscores that our method does not require large-scale, costly fine-tuning. While not the primary focus of this work, we believe that scaling the training data and further optimizing the training process could yield even greater performance, which we leave for future work.

## Entity Linking in Question Answering

In this experiment, we apply EL as a retrieval method in QA task to evaluate whether it can effectively assist the task.

**Datasets** To verify the effectiveness of EL in real QA tasks, we further conduct experiments on two widely-used datasets: TriviaQA (Joshi et al. 2017) and PopQA (Mallen et al. 2022). TriviaQA is a large scale open-domain question answering dataset collected from trivia websites. PopQA is a popularity-aware open-domain-QA benchmark with each answer grounded in Wikipedia. For TriviaQA, we directly use the provided evidence documents as the retrieval corpus. In contrast, PopQA does not include associated context documents, making it unsuitable for direct use as a retrieval corpus. To address this, we construct a corpus for PopQA by crawling the Wikipedia pages corresponding to the entities linked to each question.

**Metrics** We evaluate the performance of applying entity linking to QA task using Hit@1, Exact Match (EM), and F1 score. Hit@1 is a retrieval metric that measures whether the top-1 predicted document matches the gold document. Since entity linking returns only a single document, we report Hit@1 for fair comparison with other retrieval-based methods. EM and F1 score are standard metrics in QA (Rajpurkar et al. 2016). EM measures the exact string match between the predicted and gold answers. The F1 score balances precision and recall, measuring the overlap between prediction and gold answers.

**Baselines** To assess the effectiveness of our proposed approach, we compare it against three types of QA system: a naive generation method with chain-of-thought prompting (Wei et al. 2022) by asking LLMs the question directly, sparse retrievers with LLMs, and dense retrievers with LLMs. We use the DeepSeek-V3 as the base model to generate final answers based on the retrieved documents in all settings. In the naive generation baseline, LLM answers questions only based on its own knowledge. Next, we adopt the BM25 algorithm (Robertson, Zaragoza et al. 2009) as a sparse baseline to retrieve relevant documents from Wikipedia. For dense retrieval, we evaluate three embedding models: intfloat/e5-large-v2 (Wang et al. 2022), BAAI/bge-base-en-v1.5[4] (Chen et al. 2023), and BAAI/bge-large-en-v1.5[5] (Chen et al. 2023). All models are used to encode full Wikipedia page texts into dense vector representations. To improve retrieval efficiency, we adopt Scalable Vector Search (SVS) library (Aguerrebere et al. 2023) for approximate nearest neighbor search.

**Implementation Details** For this experiment, we use DeepSeek-V3 as the base model for our proposed ELA method. Unlike the previous setup, which used existing search tools, we now employ a custom-built entity search system. Our search system is built on a BM25 retriever that leverages the title-content structure of the Wikipedia corpus. We indexed the unique title of every Wikipedia article, enabling the retriever to perform a lexical match between a query (i.e., an entity mention) and the article titles. For each query, the system returns a ranked list of the top k=35 candidate entities, where each candidate consists of the article's title and its first paragraph as a concise description. Once our ELA agent identifies the correct entity from this list, the full text of the corresponding article is retrieved and provided as context to the LLM to generate the final answer.

**Results** Table 2 presents the performance comparison of our method against several baselines on the TriviaQA and PopQA datasets. Our approach outperforms the naive chain-of-thought generation baseline and BM25 on the PopQA dataset, and also exceeds all dense retrieval methods except for BAAI/bge-large-en-v1.5 model. On TriviaQA dataset, our method slightly behind the dense retrievers in Hit@1, it still achieves competitive results, with 64.60% Hit@1 and 79.90% EM. It is important to note that our primary research objective is not to achieve state-of-the-art QA performance, but rather to investigate whether EL can serve as an effective retrieval mechanism to access essential information required for answering questions. To keep the setup simple, we adopt a basic strategy using a lightweight BM25 retriever to fetch the top-k Wikipedia titles relevant to the given entity name. Despite this simplicity, our method achieves strong Hit@1 performance, especially on PopQA. These results demonstrate the feasibility of incorporating entity linking into QA as a retrieval mechanism. Future work can explore more advanced retrieval strategies to further improve overall performance.

[4]https://huggingface.co/BAAI/bge-base-en-v1.5

[5]https://huggingface.co/BAAI/bge-large-en-v1.5

## Conclusion

In this work, we propose a LLM-based entity linking agent that adopts a retrieval-then-reader strategy enabling it to effectively identify mentions, retrieve candidate entities, and select the most relevant entities. Our experiments show that the agent not only achieves strong performance in entity linking across multiple datasets, but also performs competitively in question answering when used as a retrieval mechanism.

## References


Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aguerrebere, C.; Bhati, I.; Hildebrand, M.; Tepper, M.; and Willke, T. 2023. Similarity search in the blink of an eye with compressed indices. *Proceedings of the VLDB Endowment*, 16(11): 3433–3446.

Al-Moslmi, T.; Ocaña, M. G.; Opdahl, A. L.; and Veres, C. 2020. Named entity extraction for knowledge graphs: A literature overview. *IEEE Access*, 8: 32862–32881.

Ayoola, T.; Tyagi, S.; Fisher, J.; Christodoulopoulos, C.; and Pierleoni, A. 2022. Refined: An efficient zero-shot-capable approach to end-to-end entity linking. *arXiv preprint arXiv:2207.04108*.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008a. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, 1247–1250. New York, NY, USA: Association for Computing Machinery. ISBN 9781605581026.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008b. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 1247–1250.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Chen, J.; Jiang, Y.; Lu, J.; and Zhang, L. 2024. S-agents: Self-organizing agents in open-ended environments. *arXiv preprint arXiv:2402.04578*.

Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2023. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2309.07597.

Choi, E.; He, H.; Iyyer, M.; Yatskar, M.; Yih, W.-t.; Choi, Y.; Liang, P.; and Zettlemoyer, L. 2018. QuAC: Question Answering in Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2174–2184.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113.

De Cao, N.; Izacard, G.; Riedel, S.; and Petroni, F. 2020. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904*.

Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Metropolitansky, D.; Ness, R. O.; and Larson, J. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Fang, Z.; Cao, Y.; Li, Q.; Zhang, D.; Zhang, Z.; and Liu, Y. 2019. Joint entity linking with deep reinforcement learning. In *The world wide web conference*, 438–447.

Ganea, O.-E.; Ganea, M.; Lucchi, A.; Eickhoff, C.; and Hofmann, T. 2016. Probabilistic bag-of-hyperlinks model for entity linking. In *Proceedings of the 25th international conference on world wide web*, 927–938.

Ganea, O.-E.; and Hofmann, T. 2017. Deep joint entity disambiguation with local neural attention. *arXiv preprint arXiv:1704.04920*.

Gillick, D.; Kulkarni, S.; Lansing, L.; Presta, A.; Baldridge, J.; Ie, E.; and Garcia-Olano, D. 2019. Learning dense representations for entity retrieval. *arXiv preprint arXiv:1909.10506*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hao, R.; Hu, L.; Qi, W.; Wu, Q.; Zhang, Y.; and Nie, L. 2025. Chatllm network: More brains, more intelligence. *AI Open*, 6: 45–52.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Hoffart, J.; Seufert, S.; Nguyen, D. B.; Theobald, M.; and Weikum, G. 2012. KORE: keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, 545–554.

Hoffart, J.; Yosef, M. A.; Bordino, I.; Fürstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S.; and Weikum, G. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, 782–792.

Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; Melo, G. D.; Gutierrez, C.; Kirrane, S.; Gayo, J. E. L.; Navigli, R.; Neumaier, S.; et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4): 1–37.

Hong, S.; Zhuge, M.; Chen, J.; Zheng, X.; Cheng, Y.; Wang, J.; Zhang, C.; Wang, Z.; Yau, S. K. S.; Lin, Z.; et al. 2023. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P. S.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP (1)*, 6769–6781.

Kolitsas, N.; Ganea, O.-E.; and Hofmann, T. 2018. End-to-end neural entity linking. *arXiv preprint arXiv:1808.07699*.

Kulkarni, S.; Singh, A.; Ramakrishnan, G.; and Chakrabarti, S. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 457–466.

Kwon, W.; Li, Z.; Zhuang, S.; Sheng, Y.; Zheng, L.; Yu, C. H.; Gonzalez, J.; Zhang, H.; and Stoica, I. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 611–626.

Lan, Y.; He, G.; Jiang, J.; Jiang, J.; Zhao, W. X.; and Wen, J.-R. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. *arXiv preprint arXiv:2105.11644*.

Le, P.; and Titov, I. 2019. Boosting entity linking performance by leveraging unlabeled documents. *arXiv preprint arXiv:1906.01250*.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474.

Li, B. Z.; Min, S.; Iyer, S.; Mehdad, Y.; and Yih, W.-t. 2020. Efficient one-pass end-to-end entity linking for questions. *arXiv preprint arXiv:2010.02413*.

Li, M.; Yang, C.; Xu, C.; Song, Z.; Jiang, X.; Guo, J.; Leung, H.-f.; and King, I. 2025. Context-aware Inductive Knowledge Graph Completion with Latent Type Constraints and Subgraph Reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(11): 12102–12111.

Liu, B.; Li, X.; Zhang, J.; Wang, J.; He, T.; Hong, S.; Liu, H.; Zhang, S.; Song, K.; Zhu, K.; et al. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Lukovnikov, D.; Fischer, A.; and Lehmann, J. 2019. Pretrained transformers for simple question answering over knowledge graphs. In *International Semantic Web Conference*, 470–486. Springer.

Ma, S.; Xu, C.; Jiang, X.; Li, M.; Qu, H.; Yang, C.; Mao, J.; and Guo, J. 2025. Think-on-Graph 2.0: Deep and Faithful Large Language Model Reasoning with Knowledge-guided Retrieval Augmented Generation. In *The Thirteenth International Conference on Learning Representations*.

Mallen, A.; Asai, A.; Zhong, V.; Das, R.; Khashabi, D.; and Hajishirzi, H. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Mavi, V.; Jangra, A.; Jatowt, A.; et al. 2024. Multi-hop question answering. *Foundations and Trends® in Information Retrieval*, 17(5): 457–586.

Mihalcea, R.; and Csomai, A. 2007. Wikify! Linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, 233–242.

Mohammed, S.; Shi, P.; and Lin, J. 2018. Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks. In *Proceedings of NAACL-HLT*, 291–296.

Orlando, R.; Cabot, P.-L. H.; Barba, E.; and Navigli, R. 2024. ReLiK: Retrieve and LinK, Fast and Accurate Entity Linking and Relation Extraction on an Academic Budget. In *Findings of the Association for Computational Linguistics ACL 2024*, 14114–14132.

Pershina, M.; He, Y.; and Grishman, R. 2015. Personalized page rank for named entity disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 238–243.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Robertson, S.; and Zaragoza, H. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4): 333–389.

Robertson, S.; Zaragoza, H.; et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4): 333–389.

Salnikov, M.; Le, H.; Rajput, P.; Nikishina, I.; Braslavski, P.; Malykh, V.; and Panchenko, A. 2023. Large language models meet knowledge graphs to answer factoid questions. *arXiv preprint arXiv:2310.02166*.

Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; and Scialom, T. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36: 68539–68551.

Sciavolino, C.; Zhong, Z.; Lee, J.; and Chen, D. 2021. Simple Entity-Centric Questions Challenge Dense Retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 6138–6148.

Shavarani, H.; and Sarkar, A. 2025. Entity Retrieval for Answering Entity-Centric Questions. In Shi, W.; Yu, W.; Asai, A.; Jiang, M.; Durrett, G.; Hajishirzi, H.; and Zettlemoyer, L., eds., *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*, 1–17. Albuquerque, New Mexico, USA: Association for Computational Linguistics. ISBN 979-8-89176-229-9.

Shavarani, H. S.; and Sarkar, A. 2024. Entity retrieval for answering entity-centric questions. *arXiv preprint arXiv:2408.02795*.

Shen, W.; Wang, J.; and Han, J. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2): 443–460.

Sil, A.; Kundu, G.; Florian, R.; and Hamza, W. 2018. Neural cross-lingual entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Stelmakh, I.; Luan, Y.; Dhingra, B.; and Chang, M.-W. 2022. ASQA: Factoid questions meet long-form answers. *arXiv preprint arXiv:2204.06092*.

Talmor, A.; and Berant, J. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 641–651.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Van Hulst, J. M.; Hasibi, F.; Dercksen, K.; Balog, K.; and De Vries, A. P. 2020. Rel: An entity linker standing on the shoulders of giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2197–2200.

Vrandečić, D.; and Krötzsch, M. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10): 78–85.

Wang, L.; Yang, N.; Huang, X.; Jiao, B.; Yang, L.; Jiang, D.; Majumder, R.; and Wei, F. 2022. Text embeddings by

weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Wu, L.; Petroni, F.; Josifoski, M.; Riedel, S.; and Zettlemoyer, L. 2019. Scalable zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*.

Wu, Y.; Ma, L.; Li, M.; Zhou, J.; Hao, J.; Leung, H.-f.; King, I.; Zhang, Y.; and Nie, J.-Y. 2025. Reinforcing Question Answering Agents with Minimalist Policy Gradient Optimization. *arXiv preprint arXiv:2505.17086*.

Xiao, Z.; Gong, M.; Wu, J.; Zhang, X.; Shou, L.; and Jiang, D. 2023. Instructed Language Models with Retrievers Are Powerful Entity Linkers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2267–2282.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Yih, W.-t.; Richardson, M.; Meek, C.; Chang, M.-W.; and Suh, J. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 201–206.

Yosef, M. A.; Hoffart, J.; Bordino, I.; Spaniol, M.; and Weikum, G. 2011. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, 4(12): 1450–1453.

Zelikman, E.; Wu, Y.; Mu, J.; and Goodman, N. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488.

Zeng, C.; Li, S.; Li, Q.; Hu, J.; and Hu, J. 2020. A survey on machine reading comprehension—tasks, evaluation metrics and benchmark datasets. *Applied Sciences*, 10(21): 7640.

Zhang, W.; Hua, W.; and Stratos, K. 2021. EntQA: Entity linking as question answering. *arXiv preprint arXiv:2110.02369*.

# Appendix

**Prompts for Entity Linking** The following is the complete prompt and few-shot examples used to guide the language model during entity identification and linking. It enforces strict constraints on what can be searched and linked, focusing only on explicitly mentioned named entities, as shown in Figure 2.

**Dataset preprocess** Both the WebQSP and CWQ datasets were originally constructed on top of the Freebase knowledge base. Each dataset consists of natural language questions paired with SPARQL queries. However, for the task of entity linking, explicit topic entities are required for each question, which are not provided in the original data. To obtain them, we first extract all Freebase entity IDs (MIDs) from the SPARQL queries and then apply a rule-based filtering strategy to identify the topic entity. For both datasets, the extracted Freebase MIDs are converted to Wikidata QIDs using a mapping derived from the Freebase data dump. We will publicly release the processed datasets to facilitate future research.

**Fine-tuning** We fine-tune the LLaMA-3.1-8B model using full parameter training to adapt it to the entity linking task. Training is conducted using the LLaMA Factory framework with the AdamW optimizer. We use a global batch size of 128, a maximum sequence length of 512, and train for 3 epochs. The learning rate is set to 2e-5 and decayed linearly with a warm-up ratio of 0.1.

**Entity Linking Prompt Template**

```
Your task is to identify and link named entities only. DO NOT try to solve or answer the question. Analyze the given
sentence and link key entities to their Wikidata
entries using the following process:
1. **Entity Identification**:
   - Only identify all concrete, specific entities in the sentence (proper nouns, titles, named concepts)
   - Focus on entities that need disambiguation (those with potential duplicates in Wikidata)
2. **Search Execution**:
   - Call Search() for each identified entity name EXACTLY as it appears
   - Enclose all searches in <search> tags
3. **Entity Selection**:
   - Compare entity descriptions with sentence context
   - Explicitly reject entities with mismatched types in <think> reasoning
4. **Output Formatting**:
   - ALWAYS show your reasoning in <think> tags first
   - List ALL relevant entity indexes in <answer> using [X][Y] format

Few-shot:

user = """<link>[Your question here]</link>"""
assistant = """<search> Search([Entity]) </search>"""
user = """ [0] xxxxx"""
assistant = """<think> [Brief reasoning here] </think>

<answer>[Your answer]</answer>"""
```

Figure 2: Entity Linking Prompts.