Incorporating Knowledge about Entities in Conversational Search

Hai Dang Tran

Dissertation zur Erlangung des Grades
des DOKTORS DER INGENIEURWISSENSCHAFTEN (DR.-ING.)
der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

Saarbrücken, 2025

Day of Colloquium	18/07/2025
Dean of the Faculty	Prof. Dr. Roland Speicher
Chair of the Committee	Prof. Dr. Isabel Valera
Reporters	
Reviewer	Prof. Dr. Gerhard Weikum
Reviewer	Prof. Dr. Andrew Yates
Reviewer	Prof. Dr. Klaus Berberich
Academic Assistant	Dr. Easa AliAbbasi

Abstract

Although trained on a vast amount of data, language models (LMs) struggle to fully capture information about entities. This issue is especially noticeable for tail entities, which are sparsely covered or entirely absent from knowledge bases. Indeed, language models' knowledge of tail entities is limited because these entities are unlikely to appear frequently in the pre-training dataset. Modern information retrieval (IR) methods rely on language models. Therefore, they struggle to interpret tail entities and questions about them. To bridge this gap, we propose incorporating knowledge about entities into LM-based models in information retrieval.

This thesis proposes three methods to improve the state of the art.

- Our first IR method, EVA, addresses the challenge of leveraging knowledge about entities to understand questions. This approach integrates knowledge from pre-trained embeddings of Wikipedia entities into BERT's dense retrieval representations. We introduce a novel approach to precompute multiple representation views for each document to maximize the likelihood of capturing the users' intents. Each view encodes a particular aspect, based on a group of related entities within the document.

- Our second method, CONSENT, tackles the challenges of contextualization and handling informal questions in conversational IR setting. This approach particularly focuses on the challenge of understanding questions about tail entities. We design CONSENT to contextualize the current question using a selected subset of previous conversation turns and entities. In CONSENT, we leverage Sentence-BERT in a zero-shot setting to infer similarities among entities, conversation turns, the current question, and candidate answers. Based on the similarities, the entity selection and answer ranking are modeled and solved jointly using integer linear programming.

- We propose our third method, EECATS, to fight three challenges simultaneously: contextualizing questions, handling long-tail entities in conversational IR, and maintaining efficiency for interactive responses. In EECATS, we fine-tune a Sentence-BERT model to infer relatedness between entities in the conversation

history and the current question. The current question is then expanded with the most related entities. This expanded question is fed through a retrieval and reranking pipeline. EECATS achieves efficiency through fast score aggregation and effectiveness through fine-tuning the Sentence-BERT model for entity selection.

Kurzfassung

Obwohl Sprachmodelle (Language Models: LMs) mit großen Datenmengen trainiert wurden, haben sie Schwierigkeiten, Informationen über Entitäten zu erfassen. Dieses Problem ist besonders bei Tail-Entitäten spürbar, die in Wissensdatenbanken spärlich oder gar nicht vorhanden sind. Tatsächlich ist das Wissen von Sprachmodellen über Tail-Entitäten begrenzt, da diese im vortrainierten Datensatz nicht häufig vorkommen. Moderne Methoden des Information Retrieval (IR) basieren auf Sprachmodellen. Daher haben sie Schwierigkeiten, Tail-Entitäten und Fragen dazu zu interpretieren. Um diese Lücke zu schließen, schlagen wir vor, Entitätswissen in LM-basierte Modelle für das Information Retrieval zu integrieren.

Diese Dissertation präsentiert drei neuartige Methoden zur Verbesserung des aktuellen Stands der Technik:

- Unsere erste Methode, EVA, befasst sich mit der Herausforderung, Wissen über Entitäten zum Verständnis von Fragen zu nutzen. Dieser Ansatz kombiniert und integriert Wissen aus vortrainierten Wikipedia-Entitäteinbettungen in die dichte Repräsentation des Sprachmodells BERT. Wir verwenden einen neuartigen Ansatz, um mehrere Darstellungsansichten für jedes Dokument vorzuberechnen und so die Wahrscheinlichkeit zu maximieren, Benutzerabsichten zu erfassen. Jede Ansicht kodiert einen bestimmten Aspekt basierend auf einer Reihe verwandter Entitäten innerhalb eines Dokuments.

- Unsere zweite Methode, CONSENT, befasst sich mit den Herausforderungen der Kontextualisierung und Verarbeitung informeller Fragen in Konversationen. Dieser Ansatz konzentriert sich auf das Verständnis von Fragen zu Long-Tail-Entitäten. Wir entwickeln CONSENT, um die aktuelle Frage mit einer ausgewählten Teilmenge vorheriger Dialogschritte und Entitäten zu kontextualisieren. In dem CONSENT verwenden wir Sentence-BERT in einem Zero-Shot-Setting, um Ähnlichkeiten zwischen Entitäten, Dialogschritten, der aktuellen Frage und Antwortoptionen zu ermitteln. Basierend auf diesen Ähnlichkeiten werden Entitätsauswahl und Antwortrangfolge gemeinsam modelliert und mithilfe der ganzzahligen linearen Programmierung gelöst.

- Mit unserer dritten Methode, EECATS, bewältigen wir drei Herausforderungen: die Kontextualisierung von Fragen, den Umgang mit den Long-Tail-Entitäten in Konversationen und die Aufrechterhaltung der Effizienz interaktiver Antworten. Wir verwenden Fine-Tuning für ein Sentence-BERT-Modell, um die Verwandtschaft zwischen Entitäten im Gespräch und der aktuellen Frage zu bestimmen in EECATS. Die aktuelle Frage wird dann um die am engsten verwandten Entitäten erweitert. Diese erweiterte Frage wird durch eine Retrieval- und Ranking-Pipeline gespeist. EECATS erreicht Effizienz durch schnelle Score-Aggregation und Effektivität durch die Feinabstimmung des Sentence-BERT-Modells für die Entitätsauswahl.

Acknowledgments

I want to express my sincere gratitude to my supervisors: Prof. Gerhard Weikum and Prof. Andrew Yates. Gerhard always opens doors and offers me many chances to grow and improve. Andrew, with his enthusiasm and support, organizes weekly meetings to supervise me, even during his busy workload. Their supervision is invaluable, supporting me to develop both professionally and personally.

I am thankful to Prof. Klaus Berberich, Prof. Isabel Valera, and Dr. Easa AliAbbasi for respectively serving as my external reviewer, committee chair, and academic assistant. Thank you for your time and contributions.

I want to send my warm thanks to Mrs. Phi Nga Le Thi and her family, my Vietnamese friends Vinh Thinh Ho and Minh Tung Phung. I am also grateful to Petra Schaaf and colleagues in the research group. Your support is very meaningful to me. Many thanks to my table tennis friends, I enjoy the fun time we play together.

Finally and above all, I am especially thankful to my big family: my grandparents, parents, uncles, aunts, brothers, and sisters. You are always beside me, and your love makes me feel confident about the career path I chose. To my father Huu Hai Tran, my mother Thi Kim Vuot Ta, and my brother Hai Ha Tran: your unconditional love and support shape a better version of me every single day.

Great love to my wife Thi Ha Nguyen, and my baby Ha Chi Tran ♡

Contents

Chapter 1

Introduction

1.1 Motivation

There is a long research history in Information Retrieval (IR), which tackles the problem of finding relevant information from a big collection of documents. Probabilistic approach is remarkable in IR research, with BM25 [Robertson and Zaragoza, 2009] leveraging exact match signals as one of typical methods. The input of these IR methods is a user question and the output is the ranking list of relevant documents. The user still needs to look into top provided documents, and find the satisfying answer. Modern IR methods leverage pre-trained language models (PLM) like BERT [Devlin et al., 2019] to greatly improve the search effectiveness over the probabilistic methods, and avoid the vocabulary mismatch issue. With the improved effectiveness, modern IR methods are designed with the aim to give the answer (sentence or passage) directly to the user, making the searching more convenient since reading through top documents to find relevant information is not needed. Consider the following question and answer example:

Q: Who won the EHF women's final four?
A: Vipers Kristiansand won 33:31 in a tight match.

The effectiveness of modern IR methods comes from the language understanding capability of PLMs. However, for long-tail entities (e.g. EHF) not appearing frequently in the pre-training data, the PLMs are not equipped with sufficient knowledge about them, resulting in low effectiveness for handling questions with this kind of entities. The answer example above could be hard to be retrieved due to this long-tail challenge. This issue motivates our research to leverage entity knowledge from an external source such as Wikipedia, to complement PLM for IR tasks. We pay attention to long-tail entities, where knowledge about them plays an important role and thus significantly improve search effectiveness of PLM-based IR methods.

IR supports one-shot settings where users pose single question independently of prior inputs, even when the questions are related to each other. Consider the following question, which is related to the above example question, but the respective answer still needs to be answered separately:

Q: Who was the MVP of the EHF women's final four?
A: Marketa Jerabkova, right-back player of Vipers Kristiansand, is the MVP.

To fulfill such in-depth information needs, the user needs to repeat terms in different questions, such as "EHF" in both questions above, since there is no context understanding across sessions. To make the search process more natural and interactive, Conversational Information Retrieval (CIR) was introduced as a dialog system with the task of searching. CIR is IR where users ask questions in multiple turns, with the current question be-ing answered within the context of the prior history. In this setting, CIR methods are required to understand questions in the respective contexts. As the conversation goes into follow-up turns, there is a high chance that long-tail entities are mentioned and become the topic of interest. In other words, the setting of long-tail entities and that of CIR fit with each other by nature. Consider the following conversation example, with questions becoming more human-like than in the one-shot setting, and the conversation needs to deal with long-tail entities.

Q_1: Who won the EHF women's final four?
A_1: Vipers Kristiansand won 33:31 in a tight match.
Q_2: Who was the MVP?
A_2: Marketa Jerabkova, right-back player of vipers.
Q_3: The team's goalkeeper: how many saves?
A_3: Norwegian legend Lunde got 20 saves in the final.
Q_4: Where did the shooter play earlier?
A_4: Before joining vipers, Marketa played for Thuringen.
Q_5: How many goals did she score for that team?
A_5: Jerabkova scored 193 goals for HC Thuringen.

From the example, we can see several long-tail entities such as HC Thuringen and Marketa Jerabkova in the follow-up turns. It is not likely that PLMs have much knowledge about them. Using knowledge about such entities from external sources could be leveraged, to enhance the search effectiveness. Existing methods for context and entity understanding typically require heavy-computing techniques, resulting in high latency. We aim to overcome this bottleneck with i) low response latency methods (efficiency), while ii) being competitive in providing relevant answers (effectiveness).

1.2 Challenges and Scope

1.2.1 Challenges

There are several challenges that we need to address for developing IR and CIR methods, under the direction of external knowledge source integration.

Challenge C1: Question understanding and contextualization. This is the fundamental challenge of IR where user questions need to be interpreted correctly to find the relevant answers. In the CIR setting, this challenge is even in the different level that follow-up questions can be informal, incomplete and require prior history as context. For example, it is difficult to understand question Q_5: "How many goals did she score for that team?" in Section 1.1 without looking into the previous turns. CIR methods are required to pick Marketa Jerabkova and HC Thuringen as relevant information pieces from prior context, to interpret the question Q_5 correctly. Similarly, when processing Q_3, CIR methods need to understand the team as Vipers Kristiansand in the context of EHF final match.

Challenge C2: Long-tail entities. To understand the user intent properly, knowledge about entities plays an important role. For example, knowing that HC Thuringen is a sports team should be helpful to understand question Q_5. Similarly, knowing Marketa Jerabkova as a shooter is useful for processing Q_3. Recall that both state-of-the-art IR and CIR methods rely on PLM, which has limited knowledge about long-tail entities though. Such limitation introduces another challenge for these methods, to handle questions about this kind of entities. One straightforward solution is to iteratively re-train PLMs with examples about long-tail entities. However, since re-training takes long time to be completed, several large language models (LLM) such as GPT [Brown et al., 2020, OpenAI, 2023] cannot directly produce answers about new emerging entities. It is impossible for GPT models to produce answers about entities that just emerged yesterday in particular news, and have never appeared in the pre-training data.

Challenge C3: Efficiency. IR and CIR methods are inefficient when they heavily rely on PLM inference, due to expensive computations. Typically, highly effective techniques require long query latency and high costs in resource consumption. For example, the iterative re-training approach does not only face the issue of emerging entities, but also suffers from huge computational costs. This point motivates us to follow the efficient approach for designing IR and CIR methods.

1.2.2 Scope

Various prior works [Hofstätter et al., 2021, Nogueira and Cho, 2019, Hai et al., 2023] in the literature have addressed the challenge C1, since understanding questions is crucial for effective IR methods. The challenge C3 on efficiency also attracted a lot of research [Formal et al., 2021b, Formal et al., 2021a, Formal et al., 2022, Yu et al., 2021], since low latency response is another necessary requirement in IR. Fewer IR works considered challenge C2 of long-tail entities, especially in the CIR subfield. This motivates us to design IR methods to tackle C2, and jointly address C1 and C3 at the same time.

There are three main works in this dissertation. The first work EVA [Tran and Yates, 2022] is a method focusing on the three above challenges in the one-shot IR setting. To the best of our knowledge, our second work CONSENT [Tran et al., 2024] and our third work EECATS [Tran et al., 2025] are the first two methods tackling CIR jointly with long-tail entities. In CONSENT, we focus on challenges C1 and C2. We additionally address challenge C3 with EECATS, while ensuring that the method is competitive in effectiveness (C1) and can cope with long-tail entities (C2).

1.3 Contributions

This dissertation offers the following contributions; each of them provides novel findings to the IR research community:

EVA. We design EVA [Tran and Yates, 2022] as an IR method that provides relevant answers from a text collection to a given user query. In this work, we investigate methods for enriching dense query and document representations with entity information from an external source. Our proposed method identifies groups of entities in a text and encodes them into a dense vector representation, which is then used to enrich vector representations of the text from BERT [Devlin et al., 2019]. To handle documents that contain many loosely-related entities, we devise a strategy for creating multiple entity representations that reflect different views of a document. For example, a document about a scientist may cover aspects of her personal life and recent work, which correspond to different views of the entity. In an evaluation on MS MARCO [Craswell et al., 2021a] benchmarks, we find that enriching query and document representations in this way yields substantial increases in search effectiveness.

CONSENT. We propose CONSENT [Tran et al., 2024] as a CIR method, that deals with incomplete and informal follow-up questions in multiple conversation turns. This

work addresses the direction where user questions are about tail entities, not featured in knowledge base (KB) and sparsely covered by PLMs. We devise CONSENT for selectively contextualizing a user utterance with turns, KB-linkable entities, and mentions of tail and out-of-KB entities. CONSENT derives relatedness weights from Sentence-BERT [Reimers and Gurevych, 2019] similarities and employs an integer linear program (ILP) for judiciously selecting the best context cues for a given set of candidate answers. This method couples the contextualization and answer-ranking stages, and jointly infers the best choices for both. Experimental evaluations on two benchmarks indicate the effective performance of CONSENT on both long-tail and popular entities.

EECATS. We create EECATS [Tran et al., 2025], another CIR method centering on long-tail entities, with the aim to improve efficiency over CONSENT and still compete in effectiveness. To this end, we define relatedness scores between current questions and previous entity mentions in the conversation, and use these to expand the current question. Answers for expanded questions are computed by a pipeline of filtering and ranking stages. The key contributions of this design are: 1) for effectiveness, the relatedness model is judiciously fine-tuned for the conversational search setting, and 2) for efficiency, the answer retrieval stages exploit an efficient aggregation of inferred and cached token embeddings. Experiments show that our method is efficient and outperforms all GPT baselines on effectiveness over a benchmark about tail entities.

1.4 Publications

EVA and CONSENT and EECATS, all of the contributions above were accepted as publications in top-tier IR conferences. The author of this thesis is also the first author of all three works.

- **Dense Retrieval with Entity Views.** Hai Dang Tran, Andrew Yates. In Proceedings of *The 31st ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.
- **Conversational Search with Tail Entities.** Hai Dang Tran, Andrew Yates, Gerhard Weikum. In Proceedings of *The 46th European Conference on Information Retrieval (ECIR)*, 2024.
- **Efficient and Effective Conversational Search with Tail Entity Selection.** Hai Dang Tran, Andrew Yates, Gerhard Weikum. In Proceedings of *The 47th European Conference on Information Retrieval (ECIR)*, 2025.

1.5 Organization

The three works, EVA and CONSENT and EECATS, are presented as follows in the rest of this dissertation. The next chapter discusses the background that is commonly used in the three works. This includes preliminaries on CIR, and prominent IR techniques like sparse and dense retrieval. After that, we describe EVA in Chapter 3, with the focus on integrating entity knowledge into dense retrieval models. This is followed by CONSENT in Chapter 4, focusing on the joint relevant entity selection and answer ranking in CIR, using external knowledge. We present EECATS in Chapter 5, still under the approach of entity selection, but with significantly improved efficiency over previous work. Finally, Chapter 6 is the conclusion of this thesis, with discussion about potential future directions.

Chapter 2

Background

2.1 Classical Sparse Information Retrieval

Classical sparse IR methods are designed to cast a question and a document into separate sparse vector representations, where each dimension corresponds to a term in a preset dictionary. Only vector elements of terms appearing in the question or document have non-zero values, which are calculated by statistical measures. The number of non-zero values is much smaller than the dictionary size, resulting in sparse representations. One of the notable methods in sparse IR is BM25 [Crestani et al., 1998].

Definition 1 (BM25) *BM25 is a probabilistic classical IR method. Given a question q and a document d, the relevance score $s(q, d)$ is calculated as follows:*

$$s(q,d) = \sum_{i=1}^{n} IDF(q_i) \frac{TF(q_i,d)(k+1)}{TF(q_i,d) + k(1 - b + b\frac{|d|}{avg\_len})}$$

$$IDF(q_i) = ln(\frac{|C| - DF(q_i) + 0.5}{DF(q_i) + 0.5} + 1) \ (\forall \ 1 \leq i \leq n)$$

where n is the number of terms in the question q, $TF(q_i, d)$ is the number of times the term q_i appears in the document d. Additionally, k and b are preset constants, $|d|$ is the length of d and avg_len is the average length of documents in the collection C. $DF(q_i)$ is the number of documents in C containing term q_i.

It can be seen that the two main statistics are IDF (inverse document frequency) and TF (term frequency). The IDF measures the term importance, and intuitively BM25 score increases when many important terms appear in the document. Before diving into more details, we need to clarify two main criteria for IR methods. Both criteria are pursued in this dissertation.

Definition 2 (Effectiveness and Efficiency) *Effectiveness of an IR method is its ability to retrieve relevant documents for a given question. Efficiency refers to the method's average runtime latency.*

The bag-of-words approach of BM25 enables this method to be compatible with inverted indexes, and thus makes it very efficient. Though it was invented decades ago, this method is still used frequently as a first-stage retriever in modern IR systems. BM25 relies on exact-match signals to calculate the relevance score, meaning that a term must appear in both the question and the document to contribute to the ranking score. Though it is a notable method in classical sparse IR, in general, BM25 does not perform as well as modern IR methods in terms of effectiveness. The exact-match constraint is the issue that modern neural IR methods aim to overcome.

Example 1 *For the question "Who was the MVP of the EHF women's final four?", a particular answer containing "biggest star of the game" may not have a high rank, simply because it lacks the keyword "MVP".*

2.2 Neural Information Retrieval

Neural IR is a research area that applies deep learning techniques to improve search effectiveness. Neural IR methods can be divided into two groups: bi-encoders and cross-encoders.

2.2.1 Bi-encoder

Recall that questions and documents are cast into different representations in the classical sparse retrieval. Bi-encoders [Karpukhin et al., 2020, Zhan et al., 2020, Xiong et al., 2021] follow a similar approach. Formally, they are defined as follows.

Definition 3 (Bi-encoder) *Bi-encoder models encode a question and a candidate answer into two separate vector representations. Subsequently, the relevance score can be computed as the Cosine similarity of these vectors. The bi-encoder can be trained to minimize the mean squared loss function.*

Variants of bi-encoders can differ in their choice of similarity function (e.g. inner product) and loss function (e.g. cross-entropy). The main advantage of bi-encoders over classical sparse retrieval is that bi-encoders leverage the language understanding capability of PLMs. In contrast, classical sparse IR methods depend on exact-match

statistical weights such as TF and IDF. Prominent approaches with bi-encoders can be dense retrieval and learned sparse retrieval.

Dense Retrieval. Dense retrieval models can be defined as follows.

Definition 4 (Dense Retrieval) *Dense retrieval models are bi-encoders where representations of questions and candidate answers are dense vectors. A dense vector is often derived from the [CLS] token embedding after processing the question or candidate answer (prefixed with [CLS]) through a BERT-based bi-encoder. Like bi-encoders, inner product or cosine similarity can be used to compute the relevance score. Dense retrieval models can also be trained with square loss or cross-entropy loss functions. This goal is to ensure the representations of the question and its relevant answer are close to each other, resulting in a high similarity score.*

Example 2 *The question "Who was the MVP of the EHF women's final four?" and the answer "Marketa Jerabkova, right-back player of Vipers Kristiansand, is the best player of the tournament" are encoded into dense vector representations with bi-encoder models. Based on related terms such as "final", "MVP", "best player", "tournament", the two dense vectors are expected to be close to each other and the relevance score is likely high.*

Unlike classical IR, dense retrieval does not rely on an explicit vocabulary-based representation. PLM-based bi-encoder models are designed to capture the semantic meaning of text.

Dense retrieval is **efficient** because the candidate answer can be encoded at indexing time. Additionally, this approach is compatible with approximate nearest neighbor (ANN) search tools such as FAISS [Johnson et al., 2019], which enhances scalability. One limitation of dense retrieval models is that they require large amounts of training data and perform poorly when data is scarce. To mitigate this issue, TAS-B [Hofstätter et al., 2021] was developed to augment training data and enhance the search effectiveness. TAS-B explores negative examples based on clusters of training questions, and includes similar questions from the same cluster into a training batch. Random answers to a similar question are considered to be hard negative. Besides, in TAS-B the negative answers of the same batch are mined such that they have a balanced relevance score, not too far from that of a positive answer, according to a mother model. The balancing strategy also helps increase the chance of having negative examples in the same batch.

The above dense retrieval methods leverage only one representation for each question or answer. Another approach is ColBERT [Khattab and Zaharia, 2020], which computes

the maximal similarity between each question token embedding and the corresponding candidate answer token embeddings, and then aggregates these values to determine the relevance score.

Example 3 *For the example above about the MVP of the final four, "MVP" and "final" from the question are maximally similar and match with "best player" and "tournament" from the answer, respectively. These similarities contribute to the final relevance score.*

ColBERT is an efficient method that benefits from ANN-based indexing. However, ColBERT requires ANN search for each question token embedding, making it less efficient than other bi-encoder methods that use a single representation per question or candidate answer. In terms of effectiveness, the maximal similarity operator can capture more complex semantic relationship between question and answer than the inner product operator, which makes ColBERT more effective than other dense retrieval models.

Learned Sparse Retrieval. With their superior effectiveness, dense retrieval models such as TAS-B [Hofstätter et al., 2021] are an alternative to BM25 [Crestani et al., 1998]. However, due to their lack of interpretability, these models are less transparent than classical sparse IR methods. To harness the effectiveness of deep learning techniques and, at the same time, improve interpretability, the approach of learned sparse retrieval (LSR) has attracted many researchers.

Definition 5 (Learned Sparse Retrieval) *This approach is a variant of the bi-encoder where separate sparse representations are encoded for a question and a candidate answer, and each dimension is for a term in the dictionary. The value of each dimension represents the importance of the term. A PLM is leveraged to compute the vector, instead of relying on statistics such as TF and IDF. The dot product is typically used to measure the similarity between the question and the candidate answer. The training objective can be a combination of cross-entropy loss, and regularization loss to enforce sparsity.*

The LSR approach highlights important terms in the question and the candidate answer, and thus is more explainable and debuggable than dense retrievers. In addition, the approach does not face the exact-match limitation, since estimating term importance using PLM is a form of question expansion.

Example 4 *"Best" and "star" terms can be found as salient to the question "Who was the MVP of the EHF women's final four?", overcoming the term mismatch issue. A PLM such as BERT can be used to calculate the importance of question terms, instead of relying on IDF.*

Furthermore, the LSR approach is compatible with inverted indexes, and thus is **efficient**. One of the prominent methods is SPLADE [Formal et al., 2021b], where a log-saturation function is used to estimate the term importance. This function ensures sparsity while mitigating term dominance, which can occur when certain question terms are excessively repeated in a candidate answer. The method is competitive with other dense retrieval models in terms of effectiveness, and significantly better in zero-shot settings. In other words, SPLADE is less data-hungry than dense retrieval models.

2.2.2 Cross-encoder

The key difference between bi-encoders and cross-encoders [Lin et al., 2020, Dai and Callan, 2019, MacAvaney et al., 2019, Akkalyoncu Yilmaz et al., 2019, Li et al., 2020] is that cross-encoders jointly encode question and candidate answer, rather than encoding them separately.

Definition 6 (Cross-encoder) *Cross-encoder models are PLMs that take input as a concatenation of a question and a candidate answer, then infer a relevance score. The models classify candidate answers into relevant and irrelevant classes. The relevance score is the probability that candidate answers are relevant. Cross-encoders can be fine-tuned to minimize the cross-entropy loss function.*

Example 5 *Regarding the question and answer about the MVP player of the final four, the concatenation can be formed as "[CLS] Who was the MVP of the EHF women's final four? [SEP] Marketa Jerabkova, right-back player of Vipers Kristiansand, is the best player of the tournament". This concatenation is fed through the cross-encoder, to obtain the likelihood that the candidate answer is relevant to the question.*

There are interactions between question and candidate answer tokens during encoding, hence candidate answer representations cannot be precomputed at indexing time. As a consequence, this design is not compatible with ANN search; thus it is not as efficient as bi-encoders. However, regarding **effectiveness**, in general cross-encoders outperform bi-encoders. Cross-encoders are typically leveraged as a re-ranker in a search pipeline, on a limited search space of candidate answers. A popular model is the cross-encoder [1] for MS MARCO, which is fine-tuned on the MS MARCO [Nguyen et al., 2016] dataset for ad-hoc passage retrieval.

[1] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

2.3 Conversational Information Retrieval

2.3.1 Conversational Question Rewriter

To bridge the gap between ad-hoc IR to conversational IR, the conversational question rewriter approach is discussed in many works.

Definition 7 (Conversational Question Rewriter) *In the approach of the conversational question rewriter, conversational questions are translated into self-contained context-free questions, so that ad-hoc IR can be applied directly.*

Example 6 *For example, the question Q_5 "How many goals did she score for that team?" in the conversation in Section 1.1 should be converted to "How many goals did Marketa Jerabkova score for HC Thuringen?".*

It can be seen that there are two separate steps under this approach: question rewriting, followed by ranking candidate answers. These two steps can be performed on different PLMs, and errors in the rewriting step can negatively impact the search effectiveness of the second step. There are typically two ways to rewrite:

- Full re-writing: self-contained questions are completely rewritten as the generated output from PLM [Yu et al., 2020, Vakulenko et al., 2021b, Voskarides et al., 2020, Vakulenko et al., 2021a, Elgohary et al., 2019, Kim et al., 2021, Chen et al., 2022, Farzana et al., 2023, Kostric and Balog, 2024, Jang et al., 2024, Ke et al., 2022, Mao et al., 2023].

- Question expansion: the current question is expanded with information from the prior conversation context [Krasakis et al., 2022, Lin et al., 2021, Yu et al., 2021, Christmann et al., 2023].

CQR [Yu et al., 2020], one of the popular conversational question rewriters, is built on top of GPT-2 [Radford et al., 2019]. In CQR, ad-hoc IR questions in consecutive sessions are converted to conversational questions, forming training data for rewriting models. For generating conversational questions, a rule-based approach or another PLM is leveraged to simplify ad-hoc questions. The PLM for simplifying questions is fine-tuned on a manually crafted dataset. Conversational question rewriters, including CQR, were a prominent trend for CIR in TREC CAsT 2020 [Dalton et al., 2020].

2.3.2 Conversational Dense Retrieval

This approach integrates dense retrieval into the CIR, and is formally defined as follows.

Definition 8 (Conversational Dense Retrieval) *A conversational dense IR model encodes a combination of the current question with all or part of prior history (contextualization) into a dense representation. It also encodes a candidate answer into a different dense representation. The two representations are compared by inner product or Cosine similarity for calculating the relevance score.*

The approach is different from conversational question rewriter, since there is no explicit rewriting step. Besides, conversational dense retrieval has the same limitation of data-hungriness as the ad-hoc dense retrieval. ConvDR, a popular conversational dense retriever, mitigates this issue by distilling knowledge from a teacher model, which is dense ad-hoc retriever ANCE [Xiong et al., 2021]. Specifically, ConvDR is trained to make sure its generated embeddings of conversational questions are close to the ANCE embeddings of the respective context-free rewritten questions. ConvDR is a small LM, thus it may struggle with handling long contexts. To mitigate this issue, the method contextualizes the current question with only past questions in the history. However, this strategy may lead to information loss and fail to understand the user's needs when the current question follows up on a specific previous answer.

Example 7 *With the conversation example about handball final four in Section 1.1, ConvDR encodes the concatenation of question Q_5 "How many goals did she score for that team?" with all previous questions to a representation. Like other dense retrieval models, this representation should be close to that of answer A_5 "Jerabkova scored 193 goals for HC Thuringen" (also encoded by ConvDR).*

2.3.3 Conversational Learned Sparse Retrieval

Learned sparse retrieval can be leveraged in the CIR research area.

Definition 9 (Conversational Learned Sparse Retrieval) *Learned sparse retrieval models aim to encode the current question with all or part of the history context (contextualization) into a sparse representation. The model also encodes a candidate answer into another sparse representation, and then established LSR applies.*

CoSPLADE [Hai et al., 2023], an extension of SPLADE [Formal et al., 2021b, Formal et al., 2021a, Formal et al., 2022, Lassance and Clinchant, 2022], is one typical method of

this kind. CoSPLADE has several strengths: compatible with inverted indexes, highly efficient, and more **transparent** than dense retrieval methods. For contextualization, CoSPLADE pays attention to all questions and top-k last answers in the prior history (k is a preset number) to understand the current question. CoSPLADE bases on this context to identify important relevant terms.

Example 8 *For example, with the question Q_5 in the conversation about Marketa Jerabkova in Section 1.1, it is expected that CoSPLADE extracts several relevant terms from all prior questions and answers A_2 to A_4 (when $k = 3$) such as "marketa", "jerabkova", "thuringen", "hc", with their scores indicating how salient they are according to the question Q_5. These terms are combined with Q_5, to help clarifying information need and improve the search effectiveness.*

CoSPLADE distills knowledge from its mother model SPLADE over self-contained rewritten questions. Specifically, CoSPLADE is trained on CANARD [Elgohary et al., 2019] where every conversational question is translated to a context-independent question by crowdsourcing. CoSPLADE is trained such that its expansion of the contextualized current question is similar to SPLADE's expansion of the respective context-free question.

Example 9 *The manually rewritten question of Q_5 in the conversation example of Section 1.1 is "How many goals that Marketa Jerabkova scores for her team Thuringen HC" could be fed through SPLADE, and the output becomes the supervision signals for CoSPLADE over the Q_5 and the prior history.*

CoSPLADE, when combined with a re-ranker monoT5 [Nogueira et al., 2020], could achieve competitive effectiveness performance on TREC CAsT 19 [Dalton et al., 2019] and CAsT 20 [Dalton et al., 2020].

2.3.4 Re-ranking with Pre-trained Language Models

Re-ranking pipelines are pretty common in IR. A typical pipeline starts with high recall, efficiency-oriented methods such as BM25, ConvDR or CoSPLADE, followed by re-ranker methods over a smaller search space.

Definition 10 (Re-ranking with Pre-trained Language Models) *PLMs could be used in the final stage for re-ranking. This design harnesses the effectiveness of PLMs in natural language understanding, which are typically pre-trained on large-scale datasets. By asking whether the answer is relevant to the question, the probability of generating "yes" token by the PLMs could be used as the relevance score.*

The pipeline is still efficient by its fast first-stage retrieval, and the effective re-rankers are only invoked on a limited number of candidate answers. The cross-encoder is a favorite choice for re-rankers. The combination of CoSPLADE [Hai et al., 2023] with the language model MonoT5 [Nogueira et al., 2020] is an example for a popular re-ranking pipeline.

In the CoSPLADE-MonoT5 pipeline, the re-ranker takes as input the concatenation of the current question, all previous questions and the most important terms found by CoSPLADE. Later MonoT5 outputs the probability that a "yes" token is generated when asked if a candidate answer is relevant to the concatenated input. The probability is then used as the relevance score for the final ranking.

Example 10 *Coming back to the example of the handball final four conversation in Section 1.1, the following steps can be performed to expand the current question Q_5:*

- *CoSPLADE extracts terms such as "marketa", "jerabkova", "thuringen", "hc" from Q_5 with part of prior history (all past questions and answers from A_2 to A_4).*
- *These terms and previous questions are used to expand the current question: "How many goals did she score for that team? Context: Who won the EHF women's final four? ... Where did the shooter play earlier? Keywords: marketa, jerabkova, thuringen, hc".*

The extracted terms play an important role in the effectiveness of the pipeline CoSPLADE + T5.

In general, the **more** training data is exploited, the bigger the language models are required and the higher effectiveness is gained. Motivated by this, the final re-ranking stage in several pipelines can be an LLM such as GPT-3.5 [Brown et al., 2020] or GPT-4 [OpenAI, 2023]. After the first-stage retrieval, for each candidate answer, the LLM is asked if the answer is relevant to the current question within the conversational context of prior turns (question and answer). Similar to MonoT5, the probability that LLM generates a "yes" token is measured and used as the relevance score. Since this CIR task is pretty standard to LLMs, and these models are likely to have a vast amount of conversation data in the pre-training, it is sufficient to use LLMs in a zero-shot setting.

2.4 Entity Linking and Representations

Recall that in this thesis, we aim to harness entity knowledge for CIR. Knowledge bases (KBs) are a rich source of entity knowledge, and support this approach. To access this

source, we need to infer which KB entities are mentioned in the question and a candidate answer. This is the task of entity linking, which is defined as follows.

Definition 11 (Entity Linking) *Entity linking consists of two sub-tasks: named entity recognition (NER) followed by named entity disambiguation (NED). The NER task is to identify entity mentions in the text. The entity mention could be unclear since many KB entities could have the same mention, thus the task of NED is to clarify these by mapping mentions to the correct entities in the KB.*

Example 11 *From the input text "Marketa Jerabkova, right-back player of Vipers Kristiansand, is the most valuable player", "Marketa Jerabkova" and "Vipers Kristiansand" should be detected as entity mentions by NER. By NED, "Marketa Jerabkova" and "Vipers Kristiansand" should be respectively clarified as Marketa_Jerabkova and Vipers_Kristiansand in the KB such as Wikipedia, based on the context of the input.*

Named entity recognition. Early NER methods are rule-based, where hand-crafted or pattern-based rules are leveraged to capture entity names. These methods could obtain high accuracy in specific domains [Sekine and Nobata, 2004, Mikheev et al., 1999], but typically suffer the recall and are hard to generalize in other domains.

The next advance for NER is the statistical probabilistic model, and [Finkel et al., 2005] with Gibbs sampling is a typical method. This method pays attention to improving the consistency of NER by looking into long distance constraint, i.e. same entity mentions are likely to have the same entity types, even if they are not in close positions. Another approach is to harness machine learning techniques such as decision tree [Szarvas et al., 2006], support vector machine [Arora et al., 2019] for recognizing entities. This approach improves the performance over the rule-based one, but requires more data annotations.

The recent rise of deep learning architecture such as transformer [Vaswani et al., 2017] could boost up the accuracy of NER methods [Li et al., 2023, Keraghel et al., 2024]. With the power from transformer, GLiNER [Zaratiana et al., 2024] has a competitive performance in NER, and also offers the flexibility of entity types. Indeed, GLiNER takes custom entity types and a text as an input, then builds representations for spans and the entity types, followed by assigning types to the spans. Large language models (LLM) can also be a solution to NER, especially when the data is not much available. GPT-NER [Wang et al., 2023] is a typical method that turns NER into generation task. The method is fed with a text input, and outputs that text with annotated entities. Leveraging LLM can contribute to competitive accuracy in NER, but it requires significantly more resource computing than other approaches.

Regarding NER tools, Spacy [2] is one of the popular choices, which provides useful NER for researchers and practitioners. Dexter [Ceccarelli et al., 2013] also features NER, leveraging a pre-built dictionary to look up entity mentions as N-grams of the question and answer. This approach obtains good coverage with informal questions, where a lot of named entities are not capitalized.

Named entity disambiguation. There are few approaches to NED methods such as leveraging statistics, graphs and deep learning. Dexter [Ceccarelli et al., 2013] is one of the typical statistics-based methods, where the probability of a Wikipedia entity candidate given a mention is determined by the structure of Wikipedia hyperlinks and anchor texts. In Dexter, disambiguating different mentions are independent from each other. Recall that Dexter also provides NER features, covering a good recall of entities for informal questions. Furthermore, the efficiency of Dexter is a crucial factor for processing questions in IR. These are the two benefits for us to use Dexter as the entity linker in this thesis.

The graph-based methods [Hoffart et al., 2011, Han et al., 2011] take the dependence of disambiguated entities into account. A common assumption for the graph-based methods is that, disambiguated entities and the input text should be related to each other. These methods model mentions, entities and input texts as vertices in a graph, and measure the similarities as the weights for the edges among them. The final goal is to select a sub-graph as a combination of annotated entities, such that the total accuracy of the disambiguation is maximal.

Recently, the neural deep learning approach [Sevgili et al., 2020] could be applied to NED, where transformer models could be used to encode a pair of mention and surrounding context into an embedding [Yamada et al., 2022, Peters et al., 2019b, Wu et al., 2020]. The mention embedding could be compared to the representation of KB entities for calculating the respective similarity, and then pick the optimal one for disambiguation. In [Peters et al., 2019b], the representation of mention is an aggregation from the embeddings of the tokens in that mention. Meanwhile, in [Yamada et al., 2022], context is suffixed by all the mentions in it and is subsequently fed into a transformer model, to infer the embedding for mentions. The transformer model is trained on a task where few suffixed entities are masked, and is forced to predict the entities. Another way is to insert a special token before the concatenation of mention-context, and obtain the embedding of that token for the representation of mention [Wu et al., 2020]. Along with representations of mentions, those of KB entities also play an important role in NED, and we will discuss more in the rest of this section.

[2] https://spacy.io/

In-KB entity representation. The in-KB entity representation is a vector embedding, which encodes knowledge about the entity. This representation is useful for capturing similarities among entities and texts in entity linking, and is also an important component to inject knowledge in IR tasks. A knowledge graph (KG) is a graph where vertices are entities and the relations (or predicates) among them are the edges. From the observation that two related entities should have direct or few edges between each other on a knowledge graph, a common approach is to ensure connected entities should have similar embeddings. In other words, link structure is a crucial factor for such way.

Example 12 *For example, entities Marketa_Jerabkova and Czech_national_team should have similar representations since there is a semantic tie between the former entity and the latter entity.*

A prominent method for in-KB entity representation is Wikipedia2vec [Yamada et al., 2016], which combines the skip-gram model of co-located proximity words with the graph link structure. This is to ensure embeddings of those are close to each other: i) connected entities, ii) words within the same context window and iii) anchor context words with the respective entities. DeepWalk [Perozzi et al., 2014] is another KG-based approach for building entity representations, where nodes and generated random paths are treated as vocabularies and sentences in a training corpus. These paths are used to tune a skip-gram model, with the aim to maximize the chance of going to neighbor nodes from a given node. TransE [Bordes et al., 2013] is an alternative to Wikipedia2vec and DeepWalk, where both entities and relations have vector representations. Let s, p, o be the TransE representations of the subject node, predicate and object node on the KG, TransE is trained with the constraint $s + p \approx o$. This constraint helps indicate if a relation exists between the two entities. Alternatively, LLMs can be used to generate representations for in-KB entities. The first Wikipedia paragraph (the entity introduction text) can be fed to the LLM such as OpenAI Embedding [3] to obtain the entity representation.

2.5 Evaluating Retrieval

In this section, we discuss metrics for measuring search performance. Given a top-ranked list R of answers, which is the search result of a particular IR method, we need to calculate how effective it is. Let r_i $(1 \leq i \leq |R|)$ be the relevance judgement of i-th answer according to the question in the ad-hoc IR or to the question with prior history context in the CIR. The simplest judgement form is binary: 0 for irrelevant and 1 for

[3] https://platform.openai.com/docs/guides/embeddings

relevant. These judgements can be graded, for example, with ratings from 0 to 3 to represent irrelevant, slightly relevant, largely relevant and perfectly relevant. The length N of R is typically preset to 1 or 3 or 10, which are the most popular choices for reflecting user attention on mobile phones and laptop screens.

Precision. The precision (P) of a ranked list R is defined as:

$$P(R) = \frac{\sum_{i=1}^{N} r_i}{N}$$

r_i $(1 \leq i \leq N)$ denotes a binary judgement. The precision metric is pretty straightforward and easily understandable, however, it does not take the order of relevant results into consideration.

Reciprocal rank. The reciprocal rank (RR) of a result list R is computed as:

$$RR(R) = \frac{1}{j}$$

where j $(1 \leq j \leq N)$ is the smallest rank where the respective j-th answer is relevant. If there is no such relevant answer, then the reciprocal rank is 0. This metric differs from precision, as the highest-ranked relevant answer counts. However, this is too simple for questions that can have multiple relevant answers.

Average precision. The average precision (AP) of a ranked list R can be formulated as:

$$AP(R) = \frac{\sum_{i=1}^{N} P@i(R) \times r_i}{\sum_{d \in C} r(d)}$$

where C is the collection of answers, and $r(d)$ is the judgement of answer d with respect to the given question. For this metric, the precision cutoff is counted only at the positions of relevant answers in the ranked list. AP is high when the most relevant answers are at the top. Unlike P or RR, AP can measure the ranking quality when there are multiple relevant answers. Though difficult to understand, AP can reflect precision and recall of the ranking simultaneously.

Normalized discounted cumulative gain. Like AP, this metric considers the positions of relevant answers in the ranked list. The following formula, namely discounted cumulative gain (DCG), is introduced to measure how valuable a ranking list R is:

$$DCG(R) = \sum_{i=1}^{N} \frac{2^{r_i} - 1}{log_2(i+1)}$$

The intuition is that, 2^{r_i} is used to weight the relevance of i-th answer, and $log_2(i+1)$ is

used for reducing the impact of relevant answers when they are in low ranks. Based on this metric, the ideal discounted cumulative gain (iDCG) can be computed. The iDCG is the DCG of the ideal ranked list, where all relevant answers are put on top. The iDCG is the highest possible DCG value that a ranked list can have. The normalized discounted cumulative gain (nDCG) is then computed as:

$$nDCG(R) = \frac{DCG(R)}{iDCG}$$

When the ranked list R is ideal, its nDCG is 1, and it is 0 if there are no relevant answers in R. The nDCG is the most popular metric used in IR.

Chapter 3

EVA: Dense Retrieval with Entity Views

3.1 Introduction

3.1.1 Motivation

Search with dense representations has quickly become an effective alternative to statistical methods that use an inverted index (e.g., BM25) [Lin et al., 2020]. Ranking with dense representations is efficient with the support of approximate nearest neighbor search [Johnson et al., 2019], which uses a specialized index to quickly identify the document representations (approximately) closest to a query representation. These representations are produced by encoding queries and documents (during an indexing step) as dense vectors using a pre-trained language model (PLM) like BERT [Devlin et al., 2019]. To give a concrete example of their success, dense retrieval approaches have performed well on the TREC Deep Learning collections [Craswell et al., 2022, Craswell et al., 2021b, Craswell et al., 2020] in terms of both efficiency and effectiveness. For example, a BERT-based model trained with topic-aware sampling (TAS) [Hofstätter et al., 2021] improves over a tuned BM25 approach by at least 35% nDCG@10 on the 2019 and 2020 TREC Deep Learning track while having low query latency.

3.1.2 Limitations

However, work has found that the PLM models powering such approaches do not capture full information about real world entities [Heinzerling and Inui, 2021]. Especially for uncommon entities, this limitation becomes more clear since PLMs both have difficulty disambiguating entities with similar surface forms [Heinzerling and Inui, 2021] and are less likely to have seen information about these entities during pretraining. Even the most powerful PLMs cannot capture information about an entity that was not present in the data or that emerged after the PLM was produced.

3.1.3 Approach

These limitations motivate approaches for encoding information about entities outside of the PLM itself. We investigate techniques for enriching the representations of query and document text using such entity representations. In our approach, a textual (query or document) representation from a PLM is combined with an entity representation derived from embeddings of the entities present in the input (query or document) text. We leverage external entity embeddings [Yamada et al., 2016] as a lightweight way to incorporate information about entities. Rather than being tied to a PLM, these embeddings are created and stored independently.

Method. A document may contain many entities; we find that the straightforward approach of creating a single vector representation of all a document's entities does not perform well. Instead, we propose a multiple representation approach in which we create several entity representations with respect to different clusters of entities in a text. This can be understood as creating different views of a document that focus on different groups of entities. These entity views are then combined with a textual representation of the document and indexed for approximate nearest neighbor (ANN) search. We find that this is an effective and efficient strategy: enriching representations from a strong BERT-based model (TAS) yields significant improvements on the TREC Deep Learning [Craswell et al., 2020] and DL-HARD [Mackie et al., 2021] and MS MARCO [Craswell et al., 2021a].

Example. Figure 3.1 illustrates the entity views present in a short passage about the scientist Lilli Hornig. The first view is a generic one that captures Lilli Hornig without focusing on any specific aspect. The second and third views capture entities related to Lilli's scientific work on the Manhattan project, while the remaining two views capture entities related to Lilli's studies at Bryn Mawr College. Our approach captures these different views in order to match different aspects of an entity. For example, the query *Lilli Hornig's work at Los Alamos* better matches the second and third views related to the Manhattan project, whereas a query about Lilli's personal life or education better matches the last two views.

3.1.4 Contributions

Our salient contributions are:

- a novel approach for improving search by enriching query and document representations with entity views that represent different groups of the entities in a document,

- exploration and analyses of alternative approaches for enriching query and document representations with entities,
- extensive evaluations of the proposed approach and baselines on four datasets, with effectiveness improvements up to a 12% increase in nDCG with a TAS BERT base,
- and a release of our code [1] for reproductions of our approach.

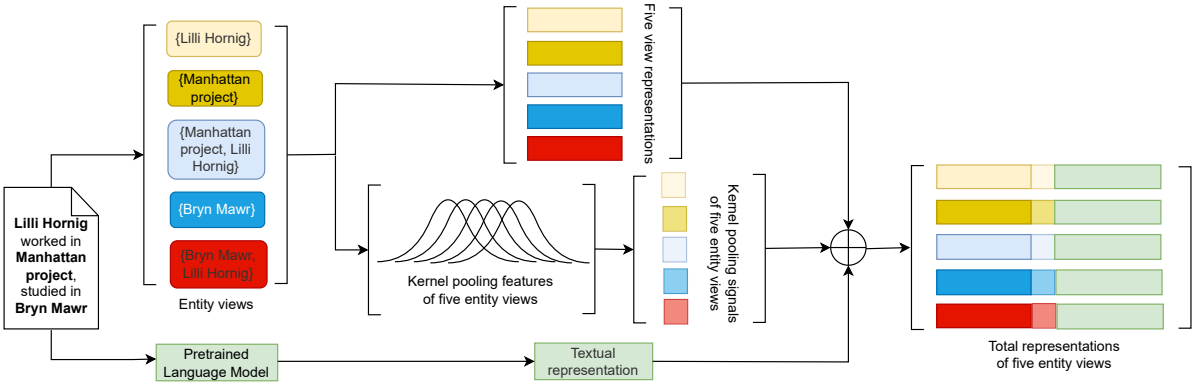

Figure 3.1: Overview of EVA with multiple representations. Entity clusters such as {Lilli Hornig}, {Manhattan project}, {Manhattan project, Lilli Hornig}, {Bryn Mawr} and {Bryn Mawr, Lilli Hornig} can be understood as different entity views of the passage. EVA generates one total representation for each view, which enriches a textual representation with the entities present.

3.2 Background

Textual Representation. Pre-trained language models like BERT can encode the semantic meaning of text as a dense vector representation. This forms the basis for the bi-encoder approach of ranking with dense representations, which is the basis for the textual representations we enrich in our approach. Specifically, we build on a distilled BERT-based [Sanh et al., 2020] model trained with topic aware sampling (TAS) [Hofstätter et al., 2021], which ensures that queries in the same batch are similar to each other (and thus that hard negative examples are chosen). We choose this TAS BERT model to support text understanding for two reasons. First, this approach was demonstrated to have state-of-the-art effectiveness while being highly efficient, because it is compatible with ANN search. Second, this approach is faster to train and to process passages at

[1] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/neural-ir

indexing time than larger BERT variant, while still being effective. To create textual representations of queries and passages, we follow the TAS BERT process [Hofstätter et al., 2021].

Definition 12 (Textual Representation) *Given a query or passage as text t, the textual representation $R_{text}(t)$ of t is formed by passing t to a PLM: $R_{text}(t) = PLM_{CLS}(t)$, where PLM can be TAS or any other BERT-like model.*

Entities and Similarity. We leverage pretrained Wikipedia2vec [Yamada et al., 2016] embeddings to represent entities and words. The similarity between two embeddings is calculated with Cosine similarity. We say two entities are related to each other if their similarity is above some predefined threshold. Later, we will build clusters of related entities, which are subsets of one or more related entities in a given passage.

Kernel Pooling. Given a set of entities X and a passage p, let Y be a set of entities and non stop-words in p. The importance degree of X in context p can be estimated by the number of high similarities between elements of X and Y. The kernel pooling approach [Xiong et al., 2017b], which has also been applied to entity/word similarities [Xiong et al., 2017c, Xiong et al., 2017a], can be leveraged to detect whether X is crucial to p. To do so, we need to define the relatedness matrix between X and Y:

Definition 13 (Relatedness Matrix) *Given the above sets X and Y, their relatedness matrix $T \in \mathbb{R}^{|X|\times|Y|}$ is defined as $T_{i,j} = sim(X_i, Y_j)$ where X_i and Y_j are i-th and j-th elements of X and Y, respectively.*

The ranking score of this KNRM approach [Xiong et al., 2017b] can be built based on the relatedness matrix. Following the original work:

Definition 14 (KNRM Score) *The KNRM score $S_{knrm}(X, p)$ of X and p is formulated as follows:*

$$S_{knrm}(X, p) = tanh(w_{knrm}^T \phi(X, p) + b_{knrm})$$

$$\phi(X, p) = \sum_{i=1}^{|X|} log\vec{Z}(R_i)$$

$$\vec{Z}(R_i) = \langle Z_1(R_i), ..., Z_K(R_i)\rangle$$

$$Z_l(R_i) = \sum_{j=1}^{|Y|} exp(-\frac{(T_{i,j} - \mu_l)^2}{2\sigma_l^2})$$

where K, μ and σ are hyper-parameters (see Section 3.4) while w_{knrm} and b_{knrm} are a learned vector and a bias.

KNRM can be viewed as creating differentiable histograms. This approach builds kernel pooling feature $\phi(X, p)$ by measuring how elements in R are distributed around and close to K different levels of similarities μ. We incorporate this kernel pooling feature in our approach to prioritize passages where query entities are important.

3.3 EVA Methodology

Given a collection of passages and a query, our goal is to identify the top relevant passages for the query. To do so, we propose the EVA method, which stands for entity views in dense retrieval. The overview of EVA is described in Figure 3.1. We use a novel multiple entity view technique to convert each passage into several total representations, which each emphasize a different view of the passage by focusing on different entity clusters. For example, the five views in Figure 3.1 highlight different aspects of Lilli Hornig, such as her work (*Manhattan project*) and personal life (*Bryn Mawr*). We calculate one total representation for each view, consisting of:

- a textual representation encoding the semantic meaning of a passage using a BERT-based encoder,
- an entity representation encoding information about the entities in the entity view,
- and, optionally, a kernel pooling signal.

While each passage may have multiple entity views (and thus multiple representations), for queries we use only one entity cluster containing all query entities. EVA follows the common bi-encoder approach where a relevance score is calculated as the dot product of two total representations from query and passage. To ensure the efficiency of EVA, we leverage an ANN search library [Johnson et al., 2019] optimized for the bi-encoder approach. In Section 3.3.1, we begin by building a single representation per passage. In Section 3.3.2 we expand from single to multiple representations and discuss in detail.

3.3.1 Single Representation

Existing bi-encoder approaches produce relevance scores by considering the semantic matching between query and document representations; we additionally consider separate representations of the entities in the texts and a KNRM signal. That is, our approach involves combining an entity representation and KNRM score with a textual representation obtained from a pretrained language model like TAS BERT. Together with semantic

understanding from this BERT model, entity representation provides knowledge about real world entities. Besides, KNRM score supports ranking model to detect whether query entities are important to the passage.

Single Entity Representation. The single representation approach produces one entity representation for a given query and one entity representation for each passage. In both cases, we identify entities present in the text (using an entity linker) and subsequently obtain the pretrained embedding for each entity. Creating an entity representation for the query is straightforward since queries are relatively short and focused.

Definition 15 (Query Entity Representation) *Let $E_{all}(q)$ be the set of all entities mentioned in query q. The query entity representation $R_{all}(q)$ is then the average embedding of entities in $E_{all}(q)$.*

The most straightforward approach for creating an entity representation of a passage is to include all entities present in the passage, which is analogous to how $R_{all}(q)$ is created:

Definition 16 (Query-independent Passage Entity Representation) *After identifying the set $E_{all}(p)$ of all entities mentioned in p, the query-independent passage entity representation $R_{all}(p)$ is the average embedding of entities in $E_{all}(p)$.*

In this query-independent approach, the average pooling may not be focused on the query's information need if p includes entities in many topics. There also may be wrong entity detection in the passage p, which potentially results in noise in the entity representation. This passage representation is used in EVA Single.

Query-aware Total Representation. Alternatively, we can make the assumption that q is known in order to quantify the effect of excluding off-topic entities. With this approach, we select only the entities in p that are close to those in q before performing average pooling. This approach enables a passage's entity representation to focus on the query and ignore unrelated entities.

Definition 17 (Query-aware Passage Entity Representation) *Let $E_{focus}(p)$ be the set of passage entities which have maximum similarity with query entities. The query-aware passage entity representation $R_{focus}(p)$ is average embedding of entities in $E_{focus}(p)$. This procedure is described in Algorithm 1. The parameter α is a predefined threshold to decide whether a query entity has any close entity in the passage.*

To reuse the query-aware passage entity representation, we still need to assume query q is known in the rest of this section. We will propose a solution to remove this assumption in Section 3.3.2.

Input: Query q and passage p
Output: Query entity representation for q and query-aware passage entity representation for p
$E_{all}(q)$ = set of entities in q
$R_{all}(q)$ = average embedding of entities in $E_{all}(q)$
$E_{focus}(p) = \{\}$
for e_q *in* $E_{all}(q)$ **do**
  e_p = entity in p having the maximum Cosine similarity max with e_q
  if $max > \alpha$ **then**
    $E_{focus}(p) = E_{focus}(p) \cup e_p$
  end
end
$R_{focus}(p)$ = average embedding of entities in $E_{focus}(p)$
return $R_{all}(q), R_{focus}(p)$

Algorithm 1: Query-aware passage entity representation

Before being integrated with a textual representation, the entity representation is projected using a learned matrix W_{entity} to create a transformed entity representation:

Definition 18 (Transformed Entity Representation) *We define the transformed entity representation $R_{trans}(t)$ of text t as:*

$$R_{trans}(t)^T = \begin{cases} R_{all}(t)^T W_{entity} & \text{if } t \text{ is query} \\ R_{focus}(t)^T W_{entity} & \text{if } t \text{ is passage} \end{cases}$$

To create a query-aware total representation of a query or passage, the text's transformed entity representation can be combined with its textual representation as follows.

Definition 19 (Query-aware Total Representation) *Formally, query-aware total representation $R_{total}(t)$ of query or passage t is:*

$$R_{total}(t) = R_{text}(t) \oplus R_{trans}(t)$$

where $\oplus$ is concatenation operator that we use to combine textual and transformed entity representations.

The method which uses this total representation for ranking is EVA Single-QA. In later experiments (Table 3.5) we additionally consider other integration approaches for combining the textual and transformed entity representations (e.g., summation rather than concatenation). With the goal of using kernel pooling to detect the importance of query entities in the passages, we aim to combine KNRM with $R_{total}(t)$.

Definition 20 (Kernel Pooling Signal) *Given a set of entities X, we define kernel pooling signal $S_{kps}(X, t)$ of X with the text t as:*

$$S_{kps}(X,t) = \begin{cases} 1 & \text{if } t \text{ is query} \\ S_{knrm}(X,t) & \text{if } t \text{ is passage} \end{cases}$$

Definition 21 (Query-aware Total Representation with Kernel Pooling) *The query-aware total representation with kernel pooling $R_{total\_knrm}(t)$ of text t is:*

$$R_{total\_knrm}(t) = R_{total}(t) \oplus S_{kps}(E_{all}(q), t)$$

This version of total representation is leveraged in EVA Single-QA-K. To perform ranking, the dot product $\otimes$ of the total representations for query and passage are used as relevance scores $S_{rel\_knrm}(q, p)$:

$$S_{rel\_knrm}(q,p) = R_{total\_knrm}(q) \otimes R_{total\_knrm}(p) =$$

$$(R_{text}(q) \otimes R_{text}(p)) + (R_{trans}(q) \otimes R_{trans}(p)) + S_{knrm}(E_{all}(q), p)$$

This formula indicates why one is used as the kernel pooling signal for the query: we want to add a query entities-passage KNRM score to the final ranking score of Single-QA-K.

The formulas for total representation and relevance score describe our variant EVA Single-QA ranking models. Entity embeddings obtained from Wikipedia2vec are frozen in our model. Meanwhile the weights of components such as TAS BERT, W_{entity}, w_{knrm}, b_{knrm} are co-learned to produce the total representation in method EVA Single-QA-K. In EVA Single-QA, only TAS BERT and W_{entity} are co-learned to build ranking model. Training details and hyperparameters are presented later in Section 3.4.3.

3.3.2 Multiple Representations

Thus far we have assumed that only a single passage entity representation is created. This means that we either face the off-topic entity issue as in EVA Single or the representation must be created at query time as in EVA Single-QA. In this section we describe an approach to create multiple entity representations for each passage, which mitigates the issue of including off-topic entities without requiring that representations are built at query time.

Multiple Entity Views. Observe that entities in $E_{all}(q)$ should be related to each other since q should describe a coherent information need. Besides, from Algorithm 1 it

can be seen that every entity in a relevant passage's $E_{focus}(p)$ is similar to at least one entity in $E_{all}(q)$, which results in relatedness among entities in $E_{focus}(p)$ itself. Thus it is not necessary to wait until q is known to identify a set of entities $E_{focus}(p)$. Instead, we can find all possible clusters in the passage p at indexing time and the query-aware $E_{focus}(p)$ should be close to one of them.

Let M be the upper bound for $|E_{all}(q)|$. In Algorithm 1 each entity in $E_{all}(q)$ corresponds to at most one entity in $E_{focus}(p)$, so $|E_{all}(q)| \geq |E_{focus}(p)| \Rightarrow M \geq |E_{focus}(p)|$. When analyzing training dataset in Section 3.4.2, it can be seen that in more than 99% queries, the number of entities is upper bounded by 2, thus $M = 2$ is a reasonable default. Since M is a small value, we can quickly enumerate every subset C of entities in p with size $l \leq M$ and check whether C is a cluster. The entity cluster C can be leveraged to replace $E_{focus}(p)$ for building total representations; this key idea allows us to remove the known query assumption.

Example 13 *In the short passage "Lilli Hornig worked in Manhattan project, studied in Bryn Mawr", there are three entities: Lilli Hornig, Bryn Mawr and Manhattan project. In this example, with parameter $M = 2$, five entity clusters of maximum size 2 can be formed such as {Lilli Hornig}, {Bryn Mawr}, {Manhattan project}, {Lilli Hornig, Manhattan project} and {Lilli Hornig, Bryn Mawr}. Note that one entity may belong to many different clusters, e.g. {Lilli Hornig} and {Lilli Hornig, Bryn Mawr}. However, {Bryn Mawr, Manhattan project} is not a cluster because Bryn Mawr and Manhattan project are not related to each other.*

While we can re-use the total representations of queries from Section 3.3.1, passage representations need to be computed in a way that is independent of the query.

Definition 22 (Transformed Cluster Representation) *Given passage p and an entity cluster C in p, let $R_{cluster}(C)$ be an average embedding of entities in C. The transformed cluster representation $R_{trans\_cluster}(C)$ of C is then:*

$$R_{trans\_cluster}(C)^T = R_{cluster}(C)^T W_{entity}$$

where W_{entity} is the learnt matrix after the training for method EVA Single-QA in Section 3.3.1.

Definition 23 (Cluster Total Representation) *Given passage p and an entity cluster C in p, we define the cluster total representation $R_{total\_cluster}(C, p)$ of passage p with cluster C as:*

$$R_{total\_cluster}(C, p) = R_{text}(p) \oplus R_{trans\_cluster}(C)$$

The corresponding query total representation is the same as in EVA Single-QA. Thus, as described, we have a single total representation per query and multiple total representations per each passage (one per cluster). For simplicity, the weights in TAS BERT and W_{entity} are all reused after the training for EVA Single-QA, without further fine-tuning. Detailed steps for generating multiple total representations are described in Algorithm 2, which are conducted at indexing time. After the cluster total representations are available, they are added to ANN index.

Input: Passage p
Output: Multiple cluster total representations of p
$E_{all}(p)$ = set of all entities in p
$clusters = \{\}$
for *every non-empty subset* $C \subset E_{all}(p)$ *with size* $l \leq M$ **do**
  if $l == 1$ *or (every pair of entities in* C *has Cosine similarity* $> \beta$*)* **then**
    $clusters = clusters \cup C$
  end
end
$total\_reps = \{\}$
for C *in* $clusters$ **do**
  $R_{total\_cluster}(C, p)$ = cluster total representation of p with cluster C
  $total\_reps = total\_reps \cup R_{total\_cluster}(C, p)$
end
return $total\_reps$

Algorithm 2: Multiple cluster total representations of passage

Intuitively Algorithm 2 scans through the passage and finds entity clusters, which can be understood as different document views that emphasize different entities. The step of building total representation per entity cluster, at its core, is a plan to maximize the chance that query entities are matched against relevant passages.

Compared to using a single query-aware representation, ranking time is significantly improved because time-consuming representation generation (including heavy BERT inference) is moved from query time to index time. Besides, our approach expands from one to multiple representations, which tackles a fundamental capacity issue in representation learning [Luan et al., 2020].

Cluster total representation $R_{total\_cluster}$ is used in **EVA Multi**, which is one of our main methods. After ANN search in EVA Multi, there can be many top ranked total representations of the same passage. In this case, we deduplicate by selecting the one with the highest relevance score for the final ranking.

Kernel Pooling Integration. With the aim to prioritize ranking of passages dedicated

to query entities, we integrate KNRM into EVA Multi. To do that, we use the following total representation:

Definition 24 (Cluster Total Representation with KNRM) *Given passage p and an entity cluster C in p, we define the cluster total representation with KNRM of passage p and cluster C ($R_{total\_cluster\_knrm}(C, p)$) as follows:*

$$R_{total\_cluster\_knrm}(C, p) = R_{total\_cluster}(C, p) \oplus S_{kps}(C, p)$$

where weights of TAS BERT, W_{entity}, w_{knrm}, b_{knrm} are all re-used after EVA Single-QA-K training. The corresponding query total representation formula is the same as in EVA Single-QA-K.

Note that this type of total representation calculates the kernel pooling signal using a cluster C rather than all query entities. The total representation is independent from a query and can be built at index time. However, this formula prioritises the ranking of passages in which cluster C is important. To ensure that query entities are dedicated in top ranked passages, we need to choose the cluster in which the entities are close to query entities. Such idea motivates the definition of cluster-query entities attention:

Definition 25 (Cluster-Query Entities Attention) *Given a query and an entity cluster C, C attends to query entities if each entity in C has high similarity (above a predefined threshold α) with at least one query entity and vice versa.*

With this attention technique, in the final search result we retain only top ranked total representations $R_{total\_cluster\_knrm}(C, p)$ of cluster C and passage p from ANN search such that C attends to query entities. If many total representations of the same passage fulfill this condition, then we obviously pick the one with the highest relevance score. The combination of EVA Multi and KNRM and cluster-query entities attention forms **EVA Multi-K**, which is our second main method in this work.

3.4 Experimental Setup

We implement our approach in TensorFlow [Abadi et al., 2016] and initialize our models from the TAS BERT checkpoint [Hofstätter et al., 2021]. Entities are represented by 100-dimension embeddings obtained from Wikipedia2vec [Yamada et al., 2016]. We use FAISS [Johnson et al., 2019] to index and perform approximate nearest neighbor search on the passage representations produced by our models. In the rest of this section,

we describe in detail the data, training procedure, and hyperparameters used in our experiments. We release our code and experimental outputs along with our publication.

3.4.1 Datasets

We evaluate our approach using four datasets built on top of the MS MARCO passage collection[2]: the MS MARCO development queries, the TREC Deep Learning (DL) Track 2019 [Craswell et al., 2020], DL 2020 [Craswell et al., 2021b], and DL HARD [Mackie et al., 2021]. These datasets consist of queries issued to a Web search engine and related passages extracted from Web pages. We build upon the Dexter entity linker [Ceccarelli et al., 2013] in order to annotate texts with entities, i.e. detecting and linking entities in texts to Wikipedia. The original Dexter implementation is static and extracts entities based on their commonness. We additionally enable context awareness capability to improve entity disambiguation in Dexter. We use this improved version to automatically annotate all the queries and passages used in our experiments. On the TREC DL 19, DL 20, and DL HARD datasets, graded relevance judgments are used with a four-point scale where 0 is non-relevant and 3 is perfectly relevant. Following the TREC Deep Learning track, we calculate nDCG@10 using these graded judgments. Regarding MRR@10 and MAP@1000, we calculate these metrics using binarized judgments where only labels of 2 and 3 are treated as relevant.

If the document frequency of an entity on Wikipedia is high (i.e. bigger than a threshold $\delta = 3000$) then such entity is common, otherwise it is uncommon. Annotating common entities is not needed in this work since it is likely that BERT was already trained on the corresponding entity mentions. Thus we only keep annotations of **uncommon** entities in training and testing data, so that the entity representations and computations are not wasted on common entities that BERT is likely to recognize.

3.4.2 Entity Summary Statistics

We analyze entity detection for queries and passages in the above datasets by calculating the number of unique entities per query or passage text. Note that we only retain uncommon entities when analyzing and doing experiments. Table 3.1 shows summary statistics for the MS MARCO training queries and the test queries from TREC DL 19, TREC DL 20, DL HARD and MS MARCO development set. It can be seen from the training data that most queries have a single entity detected, with a small number of

[2]We do not consider the Deep Learning 2021 dataset, which uses a different collection.

# Ent	Train Queries		Test Queries	
	Count	Fraction	Count	Fraction
0	130353	0.435	3442	0.483
1	149073	0.497	3232	0.454
2	19207	0.064	416	0.058
3+	1367	0.004	37	0.005
Total	300000		7127	
Avg	0.640		0.587	

Table 3.1: Summary statistics of the queries, grouped by the number of entities.

# Ent	Train Passages		Collection Passages	
	Count	Fraction	Count	Fraction
0-2	201932	0.337	3309263	0.375
3-5	261200	0.435	3731425	0.422
6-7	82416	0.137	1103501	0.125
8+	54452	0.091	697634	0.078
Total	600000		8841823	
Avg	3.87		3.63	

Table 3.2: Summary statistics of the passage collection, grouped by the number of entities.

queries having two or more entities. Based on these statistics we decide to choose $M = 2$ to generate multiple cluster total representations.

Table 3.2 shows summary statistics on the MS MARCO passages. Most passages have from 1 to 7 entities, and on average approximately 4 entities. This average number affects the average number of cluster total representations per passage. In Algorithm 2, we have parameters $\beta = 0.9$ and $M = 2$, which results in 3.7 representations per passage on average after indexing.

3.4.3 Training and ANN Settings

The above TAS BERT is 6 layer DistilBERT [Sanh et al., 2020], which has 768 dimensions, and was pretrained on the MS MARCO dataset. The entity vector has 100 dimensions and size of the learned matrix W_{entity} is 100×100 in our setup.

We use AdamW [Loshchilov and Hutter, 2017] optimizer to train our EVA models. The initial learning rate is 3×10^{-5} and the warm up takes 3% of the learning process. These models are trained using 4 Quadro RTX 8000 GPUs in parallel with mixed mode precision where each GPU handles batch size of 128. Our training is in two epochs with 300 thousand training triples from MS MARCO and pairwise hinge loss [Cao et al., 2006]. For example, with EVA Single-QA-K our loss function is $\sum max\{0, 1 - S_{rel\_knrm}(q, pos) + S_{rel\_knrm}(q, neg)\}$ where q, pos, neg are respectively query, positive passage, negative passage of a particular training triple.

For kernel pooling hyper-parameters, we set $K = 6$ with similarity levels $\mu_1 = 0.1$, $\mu_2 = 0.3$, $\mu_3 = 0.5$, $\mu_4 = 0.7$, $\mu_5 = 0.9$ and $\mu_6 = 1.0$. Besides, $\sigma_1 = ... = \sigma_4 = \sigma_5 = 0.1$ for soft matches and $\sigma_6 = 0.001$ for capturing hard exact matches.

Our approach and dense retrieval baselines are indexed with FAISS. These models share the same FAISS settings: the number of clusters *nlist* when building index is $4 \times \sqrt{N}$ where N is the total number of vectors in such index (this is the recommended

Methods	Lat	TREC DL 19			TREC DL 20			DL HARD			MS MARCO Dev		
	(ms)	nDCG	MRR	MAP	nDCG	MRR	MAP	nDCG	MRR	MAP	nDCG	MRR	MAP
Low latency													
BM25	13	0.506	0.702	0.301	0.480	0.653	0.286	0.285	0.465	0.159	0.228	0.184	0.193
ANCE	25	0.621	0.763	0.361	0.605	0.786	0.373	0.335	0.446	0.193	0.368	0.311	0.317
ERNIE Tuned	29	0.574	0.728	0.326	0.573	0.760	0.348	0.287	0.388	0.163	0.320	0.267	0.274
ERNIE Multi	70	0.669†	0.822	0.422†	0.631†	0.891†	0.394†	0.329	0.452	0.198	0.344†	0.291†	0.296†
TAS BERT	28	0.693	0.835	0.442	0.673	0.812	0.451	0.360	0.472	0.224	0.395	0.334	0.340
EVA Single	40	0.672	0.853	0.429	0.642	0.813	0.428	0.363	0.481	0.224	0.374	0.316	0.322
EVA Multi	76	0.733	0.853	**0.483**	**0.694**	**0.855**†	**0.456**	0.397†	0.521†	0.240	**0.407**†	0.346†	0.350†
EVA Multi-K	74	**0.743**†	**0.879**	0.482	0.680	0.827	0.440	**0.402**†	**0.532**†	**0.253**	0.406†	**0.347**†	**0.351**†
Higher latency													
EVA Single-QA	2039	0.737	0.862	0.443	**0.701**	**0.856**	0.444	0.389	0.515	0.221	0.402	0.342	0.346
EVA Single-QA-K	3839	**0.747**	**0.874**	**0.447**	0.685	0.838	0.439	0.397	0.534	0.232	0.405	0.347	0.351
BM25 + T5	5052	0.718	0.865	0.443	0.683	0.837	**0.462**	**0.408**	**0.585**	**0.238**	**0.443**	**0.380**	**0.383**
Best Reported	-	0.765	0.928	0.503	0.803	0.915	0.545	0.408	0.585	0.238	-	0.463	-

Table 3.3: Ranking effectiveness and latency (lat) of EVA and baselines on four datasets. The † symbol indicates a significant improvement of bi-encoder models over the corresponding base TAS BERT model or ERNIE Tuned model (paired t-test, $p<0.05$)

default). Employing these approximations reduces metrics compared to using an exact index as in some of the original works (e.g., from 0.340 to 0.334 MRR on MS MARCO Dev with TAS BERT).

3.5 Evaluation

We evaluate a range of baselines and EVA variants to quantify the impact of enriching representations with multiple entity views. The primary EVA variants are built on top of the state-of-the-art TAS BERT base model, providing a comparison point to show how much incorporating entities can improve ranking. We also include ERNIE Multi, which combines the ERNIE [Sun et al., 2020] base model with entity views, in order to demonstrate that our approach brings improvements even with a PLM designed to capture entity information.

- **BM25** uses exact match features to rank passages [Robertson and Zaragoza, 2009].
- **TAS BERT** performs ranking using only the underlying BERT-based model that produces textual representations for EVA. This is a state-of-the-art bi-encoder model on the TREC Deep Learning passage collections [Hofstätter et al., 2021].
- **ANCE** is a state-of-the-art bi-encoder model with a training procedure that iteratively

mines hard negative examples from the entire document collection [Xiong et al., 2021].

- **EVA Single** uses *query-independent passage entity representation* to create a single total representation $R_{single\_total}(t)$ of text t (t could be a query or passage):

$$R_{single\_total}(t) = R_{text}(t) \oplus (W^T_{entity} R_{all}(t))$$

  With this method, entity attention technique is disabled, so an average of all entity embeddings is used. The matrix W_{entity} is learned from MS MARCO training data and we index single total representations of passages with FAISS to boost up efficiency.
- **EVA Single-QA** uses *query-aware total representation* to encode passages, which is described in Section 3.3.1. Given that this approach must compute entity representations at query time, it is implemented as a reranking step after BM25 retrieval.
- **EVA Single-QA-K** is a variant of EVA Single-QA where a *query-aware total representation with kernel pooling* is leveraged. It is also used in reranking stage after BM25 retrieval.
- **EVA Multi** is our main method in Section 3.3.2 which creates multiple total representations for each passage. We set the parameters $M = 2$ and $\beta = 0.9$. On average the number of total representations for each passage is 3.7.
- **EVA Multi-K** leverages a *kernel pooling signal* in total representation. This method shares the same other settings with EVA Multi and is the second main method in Section 3.3.2.
- **BM25 + T5 (Zero-Shot)** uses T5 as a cross-encoder [Nogueira et al., 2020] to rerank the top 1000 results from a BM25 first-stage ranking.
- **ERNIE Tuned** is initialized as Baidu ERNIE version 2 [Sun et al., 2020], which already had knowledge about entities thanks to an entity masking technique. Later we fine-tune this model for search using the same training data as in EVA methods.
- **ERNIE Multi** has the same setting as EVA Multi except that it uses the Baidu ERNIE [Sun et al., 2020] instead of TAS BERT to encode textual representation. We fine-tune this model using the same training data as the EVA methods.
- **Best Reported** shows the best result from the MS MARCO leaderboard or from the corresponding paper [Craswell et al., 2020, Craswell et al., 2021b, Mackie et al., 2021].

3.5.1 Effectiveness

The results of evaluating the above methods are shown in Table 5.2. The methods are divided into low latency and higher latency groups where the former requires the average query latency on all datasets below 100ms.

RQ1: What is the benefit of enriching BERT's representations with entities? A PLM is unlikely to encode all facts about entities, especially uncommon or emerging entities (e.g., a new movie or product). Thus, knowledge about entities and how important they are in a passage context should be complementary to BERT. This is supported by results in Table 5.2 where the nDCG@10 of both EVA Multi variants are substantially higher than those of TAS BERT. This indicates that the introduction of entities is a significant improvement. Comparing EVA-Multi with the Multi-K variant, we see that the kernel pooling signal usually results in a slight improvement. This indicates that the multi-view entity representation is the major factor providing improvements over TAS.

We see a similar result when comparing ERNIE Tuned with ERNIE Multi. Incorporating entities with ERNIE Multi consistently improves over the base ERNIE Tuned model across datasets, with significant improvements on all but DL HARD. Note that before being fine-tuned for search task, ERNIE Tuned already contained some knowledge about entities due to its specialized pre-training [Sun et al., 2020]. This shows that the way we incorporate entity views is complementary to ERNIE's pre-training techniques.

RQ2: Do we need multiple representations per passage? If multiple passage representations are not required (e.g., because all entities in a passage are likely to be related anyway), then we would expect the EVA Single method to perform well. From Table 5.2, it can be seen that metrics of EVA Single are lower than those of both Multi and of Multi-K, which supports the argument that off-topic or noisy entities are a problem for the Single approach. Even when the annotations are perfect, a passage may contain entities that are unrelated to the query, which would dilute the influence of the related entities. EVA Multi mitigates this problem by forming a representation for each cluster of related entities. Indeed, thanks to creating a representation per entity cluster, the chance of matching query entities is higher than EVA Single, leading to higher effectiveness.

Comparing EVA Multi with Single-QA, which creates a single entity representation focused on the query, we see that Multi is an efficient alternative. In the worst case Multi performs only slightly worse than the query-aware method while being substantially more efficient, because it is compatible with ANN search (since entity representations can be formed before the query is received). We discuss these efficiency issues in next section.

RQ3: What is the effectiveness of EVA Multi variants compared to other methods? From Table 5.2 it can be seen that EVA Multi variants consistently have the highest nDCG@10,

Queries	nDCG@10		Change
	TAS	EVA Multi	
Improves			
lps laws definition	0.539	0.879	0.340
tracheids are part of _____.	0.438	0.754	0.316
what is *mamey*	0.566	0.805	0.239
Does not improve			
what is a alm	0.000	0.000	0.000
meaning of *shebang*	0.908	0.845	-0.063

Table 3.4: Example queries and their metrics

MRR@10 and MAP@1000 among low latency methods. Indeed, the Multi variants outperform the underlying TAS BERT model, ANCE, ERNIE Tuned and BM25 across all three datasets. The EVA Multi-K method is competitive with the Best Reported method on DL HARD, though it substantially lags behind on other datasets.

RQ4: What is the impact of entities on different kinds of queries? To see how entities affect various query types, we compare the nDCG@10 from TAS BERT and EVA Multi for specific queries in Table 3.4. EVA Multi can substantially outperform TAS BERT on queries where detected entities are uncommon (e.g., *mamey* and *tracheids* appear in fewer than 50 Wikipedia articles). The incorporation of these entities can be complementary to BERT's representation. Non-popular entities appear to cover the majority of instances where EVA Multi outperforms TAS BERT; they are also our motivation for combining entity and text representations. Obviously, when no entities are returned by the entity linker step, there is no difference between EVA Multi and TAS BERT. For instance, we cannot disambiguate any entity in the query "what is a alm". In some cases where only the wrong entity is detected, such as when *shebang* is recognized as a character sequence in the Unix Shell script rather than an English vocabulary term (last row), using Multi can decrease effectiveness.

3.5.2 Efficiency Analysis

Along with effectiveness, efficiency is an important factor since search systems need to run at scale. As described above, we avoid the known query assumption in building total representations to enhance scalability of EVA.

RQ5: What is the effect of removing the known query assumption? The known query assumption when building total representations requires BERT-based retrieval methods to invoke BERT inference at ranking time, which is slow. A quick mitigation is to leverage an efficient method such as BM25 for the first phase, and then re-rank the top 1000 results in the following stage (as in Single-QA). Meanwhile in EVA Multi and EVA

Multi-K and Single, the total representations are computed independently from queries at the indexing stage.

From Table 5.2, we can see the average search time per query (in milliseconds) of EVA methods and baselines on the four datasets. Experiments of all methods are conducted on the same server for fair comparisons. The EVA Single-QA takes more than 2 seconds per query. Meanwhile EVA Multi and EVA Multi-K take about 75 milliseconds across datasets thanks to moving heavy computations such as BERT inference to the indexing stage. EVA Multi variant methods are more than *26 times faster* than EVA Single-QA and EVA Single-QA-K. EVA Multi is slightly slower than ANCE, TAS, ERNIE and EVA Single due to indexing more passage representations, but this yields a significant effectiveness improvement as shown in Table 5.2. Besides, EVA Multi methods are much faster than methods in the higher latency group.

3.5.3 Model Settings Selection

Recall from Section 3.3.1 that we wish to analyze different ways of integrating knowledge with TAS BERT. Along with the proposed concatenation in EVA Single-QA, there are alternatives such as max and sum pooling for the transformed entity representation and textual representation. These alternatives require column dimension of the W_{entity} in Definition 18 to be the same as with TAS BERT (i.e., the W_{entity} should project vectors to a 768-dimensional space to be compatible with BERT's representations). All the other training settings among the corresponding models remain the same. Note that we do not use kernel pooling signal in these models.

RQ6: What integration operator should be used? To compare the three operators, we use the corresponding models to infer ranking scores of passages in MS MARCO collection for test queries in MS MARCO development set, and subsequently measure effectiveness such as nDCG@10, MRR@10 and MAP@1000. Regarding search pipeline, BM25 method is used in retrieval stage, subsequently top 1000 passages are reranked by one of the candidate models. Reranking allows us to experiment with different integration operators without re-running expensive indexing procedures. The effectiveness metrics are presented in Table 3.5 and we can see that concatenation is the clear winner, which outperforms the other two in all metrics. This experiment supports our decision of using concatenation to integrate knowledge to TAS.

The hyperparameters M and β have an impact on effectiveness and the index size of EVA Multi-K, which we consider in the following question:

RQ7: What is the impact of the hyperparameters M and β? Recall from Section 3.4.2 that more than 99% training queries have maximum 2 entities, therefore we use a default

Operators	MS MARCO Dev		
	nDCG	MRR	MAP
Sum	0.393	0.335	0.339
Max	0.388	0.330	0.334
Concat	**0.396**	**0.341**	**0.343**

Table 3.5: Varying aggregation operators.

Params		Index	Dev		Dev 2E	
M	β		nDCG	MRR	nDCG	MRR
1	-	×3.6	0.406	0.347	0.236	0.203
2	0.9	×3.7	0.406	0.347	0.236	0.203
2	0.7	×5.0	0.405	0.347	0.234	0.204
2	0.5	×7.8	0.407†	0.349†	0.257†	0.226†
3	0.5	×13.5	0.407†	0.349†	0.256†	0.226†

Table 3.6: Varying parameters M and β. The † symbol indicates a significant increase over $M = 1$ (paired t-test, $p < 0.05$).

$M = 2$ in EVA methods. This section explores the performance of EVA Multi-K with different values of M and β. It is not necessary to have parameter β for $M = 1$ since entity cluster has only one entity. Regarding $M = 2$, we use three thresholds 0.9, 0.7 and 0.5 for β. With $M = 3$, only $\beta = 0.5$ is chosen.

Recall that our EVA Multi-K method is designed to maximize the chance that query entities are matched with entity views of relevant passages. With entity views of size more than one, e.g. $M = 2$, passages and queries could potentially be matched based on more than one entity. To focus on these cases, we create a new Dev 2E dataset containing all of the MS MARCO Dev queries with at least two detected entities. Subsequently, we measure the size of index and effectiveness of EVA Multi-K (nDCG@10 and MRR@10) for both MS MARCO Dev and Dev 2E in Table 3.6.

Considering the results for Dev 2E, there is a significant effectiveness increase from $M = 1$ to $M = 2, \beta = 0.5$ in Table 3.6. This demonstrates the importance of capturing entity views when considering queries that contain multiple entities. In the latter setting more entity clusters are indexed than in the former one, which leads to better chances to capture relevant search results of queries with at least two entities. Only a small fraction (5.8%) of MS MARCO Dev queries contain more than one entity, however, so on this dataset the effectiveness improvement from increasing M is negligible.

It can be seen that effective measures of EVA Multi-K are on par among five settings while index size grows from 3.6 to 13.5 times bigger in MS MARCO Dev. With the same $\beta = 0.5$, $M = 3$ does not make any difference over $M = 2$, which supports our decision about default value of M after analyzing query entities in training data. All things considered, our default values of $M = 2$ and $\beta = 0.9$ are reasonable and balance a good trade-off between effectiveness and index size on MS MARCO Dev and Dev 2E.

3.6 Related Work

We describe two lines of related work: efforts to leverage deep learning to create improved ranking methods and efforts to use entities to improve ranking.

Neural IR. State-of-the-art neural approaches to ranking can be divided into two categories: bi-encoder methods focus on building separate vector representations of queries and documents that are compared to infer relevance scores (e.g., [Karpukhin et al., 2020, Xiong et al., 2021, Khattab and Zaharia, 2020, Hofstätter et al., 2021]), whereas cross-encoder methods aim to measure the similarity between query and document directly by taking them both as inputs (e.g., [Dai and Callan, 2019, MacAvaney et al., 2019, Akkalyoncu Yilmaz et al., 2019, Li et al., 2020]). Though highly effective, the cross-encoder approach is inefficient because all candidate documents must be processed at inference time. The bi-encoder approach ("dense retrieval") can be highly efficient when coupled with ANN search [Johnson et al., 2019], because an ANN index can quickly identify the most relevant documents for a given query vector (i.e., those documents with the highest inner products or cosine similarities). Dense document representations are computed and stored in an ANN index at indexing time, which is independent of the query. In this work, we follow the bi-encoder approach to harness its superior efficiency.

Sharing the same idea of building representations for documents at indexing time and being independent from query with bi-encoder, ColBERT [Khattab and Zaharia, 2020] extends from text-level to term-level vector representations, where each term in the query or document has one representation. While effective, this approach introduces a new inefficiency: storing a vector representation of each document term requires substantially more disk space than storing a single representation for the entire document. To fix this issue, a representation compression technique is implemented to cut down space requirement in ColBERTv2 [Santhanam et al., 2022].

Leveraging entities in IR. With the development of knowledge bases (KBs) such as Yago [Suchanek et al., 2007], Wikidata [Vrandečić and Krötzsch, 2014], DBpedia [Auer et al., 2007] and Freebase [Bollacker et al., 2008], several lines of research have investigated how these resources can be leveraged to make information systems smarter. Prior to neural methods, researchers have investigated a variety of ways to make use of facts and entities from KBs. For example, queries can be extended with entity descriptions [Xu et al., 2009], a range of features can be derived from a KB and used as features with a learning to rank method [Dalton et al., 2014], and entities can be used to create vector representations in which each dimension corresponds to an entity [Liu and Fang, 2015]. [Balog, 2018] provides an extensive survey of pre-neural approaches using entities.

In the context of neural IR methods, researchers have primarily investigated how

entities can be used by pre-BERT ranking methods. Researchers have investigated how pre-trained entity embeddings can be used in an interaction-based ranking method [Xiong et al., 2017c, Xiong et al., 2017a] and how they can be jointly learned leveraging word embeddings [Liu et al., 2018]. We employ a kernel pooling component following these approaches [Xiong et al., 2017c, Xiong et al., 2017a]. Crucially, these approaches are interaction-based and cannot be applied directly to dense retrieval. Contemporaneous work [Gerritse et al., 2022, Chatterjee and Dietz, 2022] has demonstrated that interaction-based (cross-encoder) PLMs can benefit from entity representations.

ERNIE [Sun et al., 2020] is a prominent PLM that incorporates entity knowledge through one of its pretraining tasks (i.e., predicting the surface form of a masked entity). ERNIE has been applied to ranking with strong results as part of the larger RocketQA [Qu et al., 2021] method. While there are many effective performance enhancement techniques implemented in RocketQA, in the current work we focus on studying how entity knowledge specifically can improve dense retrieval. To do this, we fine-tune the model used in RocketQA (i.e., ERNIE) for search and evaluate its performance, finding that our entity enrichment approach is complementary to ERNIE's entity knowledge.

Other approaches have been proposed for incorporating entity knowledge into a pre-trained language model in a general setting (not for ranking) [Peters et al., 2019a, Zhang et al., 2019, Liu et al., 2020]. Injecting entities into BERT has yielded great improvements in NLP tasks such as relation extraction and entity typing [Peters et al., 2019a, Pörner et al., 2019] or the GLUE [Wang et al., 2019] benchmark [Zhang et al., 2019]. While these approaches can be used to add information about entities to a pre-trained language model, they suffer from the same issues with entities that were not seen during training (discussed in the introduction). Motivated by the success of prior work, we focus on approaches that overcome these drawbacks by creating entity-representations independently of a pre-trained language model.

3.7 Summary

In this work we presented EVA, an approach for enriching dense query and document representation with multiple views of the entities in them. Results on several test collections indicated EVA's effectiveness, which significantly outperforms state-of-the-art single-vector baselines. Results on a subset of MS MARCO queries containing two or more entities indicate EVA brings further benefits for more complex queries. While we use TAS BERT and ERNIE in our experiments, our approach for enriching dense representations is general and could be combined with any single-vector model.

Chapter 4

CONSENT: Conversational Search with Tail Entities

4.1 Introduction

4.1.1 Motivation and Problem

Conversational search is an IR mode where users pose questions and receive system-provided answers in a multi-turn dialog. In this chapter, we consider a setting where user inputs are short, colloquially formulated questions, and system answers are sentences retrieved and inferred from a large corpus of text documents (e.g., Wikipedia articles or news articles). An example is this conversation about the 2022 champion's league in women's handball:

Q_1: Who won the EHF women's final four?
A_1: Vipers Kristiansand won 33:31 in a tight match.
Q_2: Who was the MVP?
A_2: Marketa Jerabkova, right-back player of vipers.
Q_3: The team's goalkeeper: how many saves?
A_3: Norwegian legend Lunde got 20 saves in the final.
Q_4: Where did the shooter play earlier?
A_4: Before joining vipers, Marketa played for Thuringen.
Q_5: How many goals did she score for that team?
A_5: Jerabkova scored 193 goals for HC Thuringen.

This example exhibits three challenges of conversational search:

C1. **Informal utterances:** The user's formulations can be colloquial and grammatically flawed, using shorthand notations and omitting certain parts. Question Q3 above is

an example.

C2. **Contextualization:** The questions are often incomplete, as humans expect the system to understand their intent from the previous turns of the conversation. Question Q2 is an example: "MVP" (short for Most Valuable Player) is not self-contained, it needs the "EHF women's final four" as context.

C3. **Emerging and tail entities:** Mentions in previous turns' questions or answers are essential for proper context. Some entity mentions may refer to real-world entities that are not (yet) captured in a knowledge base (KB). This increases the ambiguity of phrases like "the shooter" and "that team". It is crucial that the system infers that Q4 needs Q2/A2 as context and that "team" in Q5 refers to A4, not to A1 or Q3.

4.1.2 Limitations

Prior works address only C1 and C2. Methods are based on either rewriting user utterances into complete, self-contained questions (e.g., [Yu et al., 2020, Vakulenko et al., 2021b, Tredici et al., 2021]) or on using neural encoders to contextualize the user's inputs (e.g., [Gao et al., 2023]). The latter typically incorporate either only the immediately preceding and/or the very first turn of a conversation, or they incorporate the complete history of the conversation as context. None of these methods is robust in the presence of tail entities and the absence of large amounts of labeled training data.

Challenge C3 has been overlooked in the literature. With neural encoding of all prior turns, out-of-KB (OKB) entities are captured by surface names. However, there is no background information in the KB for proper interpretation. Also, surface names may collide with identical names by other entities in the KB, creating ambiguity and confusion. For example, "Lunde" is a frequent name that matches many entities. If linked incorrectly, it would add confusing signals.

4.1.3 Approach

This chapter addresses the C3 challenge of dealing with emerging and tail entities. As these are entangled with the contextualization problem, we consider C2 and C3 jointly. On C1, we employ standard techniques. We name our method ConSEnT, for conversational search with entities in the tail.

CONSENT works in two stages. In the first stage, candidate sentences are retrieved from the underlying corpus (a large archive of news articles [Hoffart et al., 2014] and Wikipedia articles). This can use a variety of IR techniques, including BM25, dense retrieval [Yu et al., 2021], SentenceBERT (SBERT) [Reimers and Gurevych, 2019] sematic

search or neural retrieval with sparse vectors [Hai et al., 2023]. We aim at a pool of at least 100 top-scoring matches.

The second stage performs ranking the candidate snippets. CONSENT detects all entity mentions in the conversation history. The mentions that can be linked to a KB with high confidence, are mapped and augmented with salient KB facts. The non-linkable, potentially OKB mentions are kept in their surface forms. In principle, we could now contextualize the current question with this entire set, but this would be indiscriminate and aggravate the task. In the example, "that team" would be encoded with both Vipers Kristiansand and HC Thuringen as context. Therefore, our method judiciously selects only KB entities and OKB mentions that are likely to be relevant for the current question. We achieve this by devising an integer linear program (ILP) that reasons over all cues jointly, selects the relevant entities and mentions, and yields the final ranking of candidate answers. The ILP coefficients are learned from a small set of training conversations, using SGD for pairwise ranking loss.

4.1.4 Contributions

This chapter offers three novel contributions:

- CONSENT is the first work that addresses the challenge of handling emerging and tail entities in conversational search. Our method copes well with mentions of OKB entities and tail in-KB entities.
- The new benchmark, CONSENT Data, automatically generates difficult conversations from Wikipedia, enforcing a large fraction of OKB entities.
- Experiments show that the CONSENT method outperforms a number of baselines by a substantial margin. On questions with tail entities, we outperform even LLM-based competitors that use GPT-3.5 or GPT-4.

Code and data of our work CONSENT are released on GitHub [1].

4.2 CONSENT Methodology

4.2.1 Overview

Our method, called CONSENT, takes as input a conversational history (or history) $H = \{(Q_1, A_1), (Q_2, A_2), \ldots (Q_{i-1}, A_{i-1})\}$ and a current question Q_i. CONSENT searches answers of the current question Q_i from a text corpus, which contains a large number of

[1] https://github.com/haidangtran1989/CONSENT

news and Wikipedia articles. These sources are rich in tail and out-of-KB (OKB) entities. In this work, questions are informal user utterances, answers are complete sentences. To support searching, we leverage in-KB entities $E = \{e_1, e_2, \dots\}$ as well as mentions $M = \{m_1, m_2, \dots\}$ of OKB entities in H. E contains only entities that an entity linker can map to Wikipedia with high confidence (i.e. higher than a preset threshold). All other mentions (treated as OKB) are included in M, possibly including different names for the same entity. In this work, entity is used to refer to a mention, in-KB entity or OKB entity.

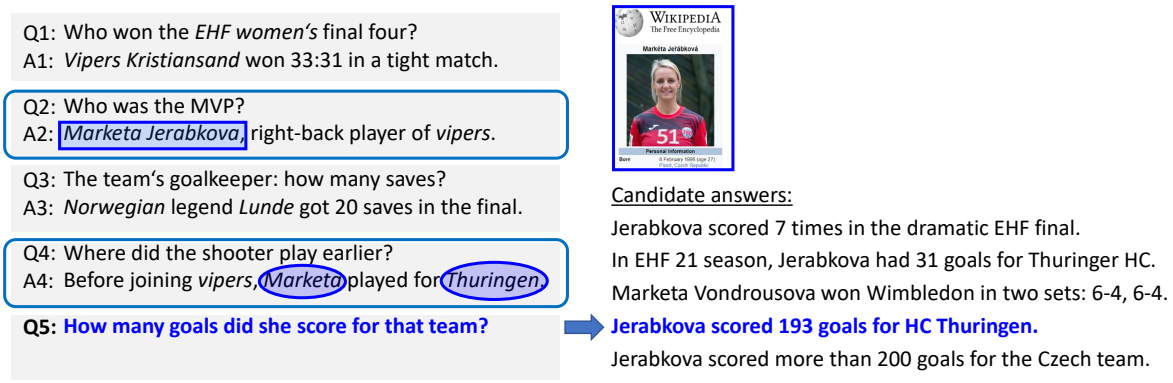

Figure 4.1: Contextualization and Ranking by CONSENT.

CONSENT addresses two sub-problems in a coupled manner: 1) **contextualizing** the current question to understand the information need, and 2) **ranking** the answer sentences. For the first task, CONSENT judiciously selects the following kinds of context items and potentially augments them:

- turns $H^* \subseteq H$ (a turn is a pair of a question and its answer in H).
- in-KB entities $E^* \subseteq E$ that appear in one or more of H^*.
- mentions $M^* \subseteq M$ that appear in one or more of H^*.

In-KB entities are augmented with the first sentence of their respective Wikipedia articles. For mentions M^*, which are not linkable to the KB, this enrichment is not feasible. Consider the history and current question of the introduction's example, shown in Figure 4.1. To contextualize Q_5, we should incorporate turns 2 and 4, as they are key to interpret the phrases "she score" and "that team" in Q_5. In addition, we can sharpen the focus by including the in-KB entity "Marketa Jerabkova" and the mentions "Marketa" and "Thuringen", giving higher weight to these cues.

For answer ranking, CONSENT operates over a pool of initial candidates, obtained by a baseline retriever, such as BM25, dense retrieval or sparse retrieval. In experiments, we use a search pipeline of BM25 and SBERT [Reimers and Gurevych, 2019] to retrieve

100 candidates. The CONSENT ranking for these top-100 considers textual similarity, and also exploits knowledge about semantic types of entities. For example, the type *handball player* of Marketa Jerabkova is important. There are other notable entities for the mention "Marketa", including a prominent tennis player, which would confuse and dilute the answering.

The answer ranking depends on how the question is contextualized. Conversely, the best choice of turns, entities and mentions for contextualization may depend on the answer candidates and the ranking approach. Therefore, CONSENT addresses both tasks jointly, by means of an integer linear program (ILP). The ILP has 0-1 variables for selecting or not selecting context items, and has an objective function that scores each candidate answer. The model has a small number of hyper-parameters to weigh the influence of different aspects, learned via SGD optimization from conversations with ground-truth answers.

4.2.2 Joint Inference on Contextualization and Answer Ranking

For contextualizing question Q given its history H and a candidate answer A, we consider three kinds of cues: turns, in-KB and OKB entities within these turns. Which cues are selected and which ones are disregarded is modeled by introducing the following 0-1 decision variables:

- $X_i = 1$ if the i^{th} turn H_i in the history H is selected, 0 otherwise $(1 \leq i \leq |H|)$
- $Y_j = 1$ if the j^{th} entity EM_j of $E \cup M$ is selected, 0 otherwise $(1 \leq j \leq |E \cup M|)$

To ensure that we select only entities from selected turns, the following constraint must be satisfied:

- $\forall i \forall j (1 \leq i \leq |H|, 1 \leq j \leq |E \cup M|) : X_i \geq Y_j$ if j^{th} entity of $E \cup M$ appears in i^{th} turn of the history H

In addition, we can control the maximum number of selected turns and entities, with model configuration parameters K and L, by the constraints:

- $\sum_{i=1}^{|H|} X_i \leq K$ and $\sum_{j=1}^{|E \cup M|} Y_j \leq L$ where K and L are preset constants.

To score answer candidate A we need to define an objective function, which is conditioned on A. The ILP is invoked once for each of the initially retrieved top-100 candidates, to compute the final ranking. Intuitively, the objective is to reward the most informative cues, by setting their decision variables X_i and Y_j to 1. Before formally presenting the objective function, we need to define several building blocks:

Definition 26 (Text-text relatedness (Rel^{TT})**)** *Given two text snippets u and v, their text-text relatedness $Rel^{TT}(u, v)$ is the Cosine similarity between their text embeddings derived via SBERT [2] [Reimers and Gurevych, 2019].*

Definition 27 (Turn-question relatedness (Rel^{TQ})**)** *The relatedness $Rel^{TQ}(T, Q)$ between a given turn T and a given question Q is calculated as $Rel^{TQ}(T, Q) = Rel^{TT}(T, Q)$.*

Definition 28 (Turn-answer relatedness (Rel^{TA})**)** *Similarly, the relatedness score $Rel^{TA}(T, A)$ between a given turn T and a given answer candidate A is $Rel^{TT}(T, A)$.*

Definition 29 (Verbalized entity types ($types^{V}$)**)** *Given an entity e in the history, we predict its types from its context C^e. If e is in-KB, C^e is the first sentence in the respective Wikipedia article. Otherwise (e is OKB), we take the turn containing e as the C^e. We feed the name of e and C^e through a neural entity typing tool [Onoe et al., 2021] to obtain the types. The verbalized entity types $types^{V}(e)$ of e is the concatenation of the entity mention with all of its types.*

Example 14 *The verbalized entity types of "Marketa Jerabkova" and "Thuringen" (conversation example in Section 4.1) are respectively "Marketa Jerabkova is person, player, athlete, contestant" and "Thuringen is team, club, organization".*

Definition 30 (Entity-question relatedness (Rel^{EQ})**)** *The relatedness $Rel^{EQ}(e, Q)$ of entity e and current question Q is calculated as $Rel^{EQ}(e, Q) = Rel^{TT}(types^{V}(e), Q)$.*

Definition 31 (Entity-answer relatedness (Rel^{EA})**)** *The relatedness $Rel^{EA}(e, A)$ between a given entity e and a given answer A is: $Rel^{EA}(e, A) = Rel^{TT}(types^{V}(e), A)$.*

Definition 32 (Question-answer base relatedness (Rel^{QA}_{Base})**)** *Given a current question Q and a candidate answer A, the question-answer base relatedness $Rel^{QA}_{Base}(Q, A)$ is the relevant score that we obtain from a particular IR model such as cross-encoder [3].*

With these designed building blocks, the to be maximized ILP objective function could be formulated as follows:

$$\alpha_1(\sum_{i=1}^{|H|} X_i Rel^{TQ}(H_i, Q)) + \alpha_2(\sum_{i=1}^{|H|} X_i Rel^{TA}(H_i, A)) + \alpha_3(\sum_{j=1}^{|E \cup M|} Y_j Rel^{EQ}(EM_j, Q))$$

[2] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[3] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

$$+\alpha_4(\sum_{j=1}^{|E\cup M|} Y_j Rel^{EA}(EM_j, A)) + \alpha_5 Rel^{QA}_{Base}(Q, A) - \alpha_6 \sum_{i=1}^{|H|} X_i - \alpha_7 \sum_{j=1}^{|E\cup M|} Y_j$$

where $\alpha_1 \ldots \alpha_7$ are tunable hyper-parameters, learned from withheld conversations with ground-truth answers (Section 4.4). The last terms, weighted by hyper-parameter α_6, α_7, can be viewed as a regularizer that aims to reward parsimony: select as few of the turns and entities as possible, hence the negative sign. To solve the ILP, we use the Gurobi optimizer [4]. The optimal value of the above function is the CONSENT score of candidate answer A given Q and H.

4.3 Automatic Benchmark Construction

Existing benchmarks for conversational search mostly feature prominent entities. We constructed a benchmark dataset with specific focus on tail entities (either OKB or with Wikipedia articles that have few in-links below a specified threshold). To this end, we sample tail entities from the Wikipedia 2018 event page [5] and related articles, to construct facts around these entities, which serve as gold answers for conversations. We pursue this "time-travel" approach, as many of the former tail entities have become prominent as of now. The 2018 snapshot also allows us to relate to the available text corpus, consisting of 2017-2018 news articles from [Hoffart et al., 2014] and the 2017 Wikipedia articles. The questions themselves are generated using GPT-3.5 [Brown et al., 2020] from the constructed sequence of gold answers.

4.3.1 Gold Answer Sampling

Conversations are initialized with a news snippet A_0 from the 2018 Wikipedia event page. This gives us a seed entity for generating T turns with gold answers $A_1, A_2, ..., A_T$. To allow shifts in the conversations, the i^{th} turn selects a random tail entity e_i mentioned in one of $A_0, A_1, ..., A_{i-1}$. We extract the entity e_i by detecting Wikipedia anchor link. Thus e_i is in-KB entity in 2023, but in 2018 (the time we use in testing), e_i can be treated as OKB or tail entity. We call e_i the target entity of the i^{th} turn. For each e_i, we retrieve sentences from its 2023 Wikipedia article that contain the entity name, disregarding any sentence that mentions a date later than 2018 (sampled facts of e_i). The text-text relatedness of the two texts is measured by the Cosine between their two SBERT [6] [Reimers and Gurevych, 2019] embeddings. We select top-g sampled facts of e_i

[4] https://www.gurobi.com/
[5] https://en.wikipedia.org/wiki/2018
[6] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

by text-text relatedness to the previous A_{i-1}, for coherent conversations, and randomly pick one of these as A_i. These steps are formally presented in Algorithm 3. In the next section, we focus on how to generate questions from these gold answers.

Input: News snippet A_0, number of conversation turns T
Output: Golden conversation answers
e_0 = random tail entity in A_0
$S = \{e_0\}$
$A = \{\}$
for i *in* $\{1, 2, ..., T\}$ **do**
  e_i = random tail entity in S
  F = sampled facts of e
  G = top-g facts in F, having the highest text-text relatedness with A_{i-1}
  A_i = random fact in G
  H = all entities in A_i
  $A = A \cup A_i$
  $S = S \cup H$
end
return A

Algorithm 3: Sampling conversation gold answers $A_1, A_2, ..., A_T$ from news A_0

4.3.2 Question Generation

Context: Cheddar Man was relatively small compared to modern Europeans, with an estimated stature of around 1.66 metres (5 ft 5 in), and weighing around 66 kilograms (146 lb).
Spoken Short Question containing Cheddar Man: How tall Cheddar Man was?
More Spoken Short Question without subject: How tall?

...

Context: {{*answer*}}
Spoken Short Question containing {{*entity_name*}}:
More Spoken Short Question without subject:

Figure 4.3.1: Question-generation prompt to create questions from gold answer {{*answer*}}, {{*entity_name*}} is the target entity name.

For each A_i ($1 \leq i \leq T$), we ask GPT-3.5 [Brown et al., 2020] (*text-davinci-003*) to generate a question with the prompt consisting of A_i and the instruction to provide a

short, spoken-style question that has A_i as its answer (*question-generation prompt* in Figure 4.3.1). We also want the generated question to be related to the respective target entity e_i by a constraint that e_i should appear in the question (question type 1, e.g. "How tall Cheddar Man was?"). The question can be shortened to be more informal by the prompt (question type 2, e.g. "How tall?"). The i-th turn is a topic-jumping turn if $i > 1$ and $e_i \neq e_{i-1}$. We use question type 1 for the first turn ($i = 1$) or the topic jumping turn, and type 2 for others (turns that discuss the same target entities as the previous ones). Question type 2 is used in cases where the topic is not shifted, hence the question can be pretty short and informal. We randomly generate several candidate questions and pick the question q_i^* with the highest text-text relatedness with A_i, to ensure coherence. The news snippet A_0 is not included in the conversation history for training and evaluation, it is only used for question generation.

4.3.3 Judgements

While the above process yields questions and their canonical answers, we also need to assess whether answers returned by systems are relevant. To do so, we identify the top three relevant system answers for every question and method, followed by de-duplication and judging. In the judgement process, we determine the relevance of answers automatically using two prompts. First, with the *answer-check prompt* in Figure 4.3.2, we assess a system answer's relevance by providing a series of questions and answers in the conversation (gold answers in the history and the system answer in the current turn) to GPT-4, followed by asking whether the system candidate answer is relevant to the current question.

Q: {{q_1^*}} A: {{A_1}}
...
Q: {{q_{t-1}^*}} A: {{A_{t-1}}}
Q: {{q_t^*}} A: {{*candidate*}}

Does the last A answer the last Q?

Figure 4.3.2: Answer-check prompt to verify if the candidate answer {{*candidate*}} is relevant to the current question q_t^* regarding to prior history of $t-1$ questions and $t-1$ gold answers.

Second, with the *entity-check prompt* in Figure 4.3.3, we assess the relevance of a system answer against the target entity of the current turn. We provide the system

answer to GPT-4, followed by asking whether it provides information related to the target entity.

{{*candidate*}}

Does the above text provide information related to {{e_t}}?

Figure 4.3.3: Entity-check prompt to verify if the candidate answer {{*candidate*}} is relevant to the target entity of the current turn e_t.

A system answer is judged as correct if and only if the responses to both prompts start with "Yes". This way we construct the entire benchmark dataset in a completely automated manner. Further details are given on the dataset and code repository [7].

4.4 Experimental Setup

We conduct experiments on two datasets: our new CONSENT dataset centering on tail entities and ConvQuestions [Christmann et al., 2019], focusing on popular in-KB entities. We measure nDCG@3, Precision@1 and MRR@3, which are abbreviated as nDCG, P and MRR, respectively. We create the CONSENT dataset by following the above construction procedure, which yields 371 conversations with 1902 questions total. These are split into a training set of 146 conversations with 751 questions and a test set of 225 conversations with 1151 questions.

To assess the quality of the automatically created conversations and their ground-truth answers, we randomly sample 115 test questions (10%) and ask crowd workers for additional relevance judgements using Amazon Mechanical Turk [8] (AMT). For each question, we provide the judges with the conversation history and a set of 3 candidate answers randomly selected from the automatic assessment. We ask the judges to select all correct answers, which could vary between 0 (all false) and 3 (all correct). We use three crowd workers per question and treat an answer as judged correct when it is selected by at least two judges. Comparing these to the GPT-generated assessments, the GPT judgements have an average accuracy of 90.9%. Considering the positive class, the GPT judgements have a precision and recall of 85.1% and 79.1%, respectively. These results indicate that GPT and human crowd workers often agree, which suggests our automated

[7] https://github.com/haidangtran1989/CONSENT
[8] https://www.mturk.com/

evaluation is well-designed and gives meaningful results.

The popularity of in-KB entity is the number of incoming links in 2017 Wikipedia. We denote an in-KB entity as *rare*, *uncommon* and *popular* if its popularity is respectively in the ranges $[1, P^R], [P^R + 1, P^U], [P^U + 1, \infty)$, where $P^R < P^U$ are specified thresholds. We categorize the CONSENT test questions into OKB, rare, and uncommon subsets based on whether the corresponding target entity is in-KB and its popularity. These OKB, rare and uncommon subsets contain 284, 535, and 332 questions, respectively. Systems retrieve answers from a corpus of nearly 1.5 million STICS [Hoffart et al., 2014] 2017-2018 news articles and approximately 5.5 million 2017 Wikipedia articles. Answers in the conversation history are gold answers sampled from Wikipedia 2023, which ensures that the history of the current question is the same for every method.

Due to high cost of running GPT on ConvQuestions, we reduce the benchmark's size by randomly sampling 10% of conversations in the test set, which yields 224 conversations and 1120 questions. On this dataset, there are no target entities and the gold answers are in entity name format. We treat an answer as relevant if it contains a term from the corresponding gold entity name and the output of the *answer-check prompt* (Section 4.3) starts with "yes".

4.4.1 Training

We fine-tune existing baselines following the training setup used in their original works. We train the CONSENT method with the following positive and negative learning examples. Let A^* be the golden answer of a question Q (in CONSENT trainset) and Y^* be the decision variables for entity selection where only entities appearing in both history H and A^* are selected. Let X^* be the decision variables for turn selection where only the turns containing the above-selected entities are selected. Each positive example is formed as a combination (H, Q, A^*, X^*, Y^*). Negative learning examples are formed by the following two strategies. For the first strategy, we randomly select entities in H and the turns containing them to build X^-, Y^-. Random selection is noisy, resulting in a negative example (H, Q, A^*, X^-, Y^-). The intuition is that when we have a good answer, but turn and entity selection are bad then the learning combination is negative. For the second strategy, we feed H, Q to a particular search method (e.g. SBERT ranker [Reimers and Gurevych, 2019]) and randomly select a low-ranked answer A^- (i.e. out of top three) to form a negative example (H, Q, A^-, X^-, Y^-). That is, with the negative answer and the selection of turns and entities having noises, the learning combination is also negative.

This yields 78,092 training pairs in total, each pair consists of a positive and a negative

example. We target that the objective function value (Section 4.2.2) of the positive example is higher than that of every respective negative example. For each learning example, decision variables and relatedness scores are constants, because neural models generating these scores are frozen during training. Based on the examples, we use a pairwise ranking loss [Cao et al., 2006] to train our parameters $\alpha_1, \alpha_2, ...\alpha_7$. The initial learning rate is 5×10^{-2} with a warm up stage of 3% of the whole training, the number of epochs is 5. We default the hyper-parameters $K = L = 3$, this will be discussed more in the next section. With these K and L, after training, we gain parameters $\alpha_1 = 0.94$, $\alpha_2 = 2.58$, $\alpha_3 = 1.05$, $\alpha_4 = 5.09$, $\alpha_5 = 0.39$, $\alpha_6 = 0.89$ and $\alpha_7 = 1.46$.

4.4.2 Methods

We report the performance of our approach and of the following baselines, which cover a wide range of approaches.

- **BM25**: We prefix a question with entities from the history to form the query input of BM25 [Robertson and Zaragoza, 2009]. Sentences from the corpus are indexed for this baseline.

- **ZeCo2**: The current question prefixed with history and each answer from the top results of BM25 are fed through ZeCo2 [Krasakis et al., 2022] to get the relevance score.

- **ConvDR**: We take a ConvDR [Yu et al., 2021] model trained on CAsT 20 [Dalton et al., 2020] and fine tune it with CONSENT and ConvQuestions trainsets before testing on the respective tests. We distill knowledge from ANCE [Xiong et al., 2021] during the training.

- **CQR**: We fine tune CQR [Yu et al., 2020] with CONSENT and ConvQuestions trainsets to rewrite utterances in the corresponding testsets into self-contained questions, which are then fed to a pipeline of BM25 and SBERT [9] [Reimers and Gurevych, 2019].

- **CoSPLADE**: We fine tune original CoSPLADE model [Hai et al., 2023] with training data of CONSENT and ConvQuestions (MonoT5 [10] as a teacher model), followed by evaluation on the respective testsets.

[9] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2
[10] https://huggingface.co/castorini/monot5-base-msmarco-10k

- **SBERT Zero**: For the current question Q, we prefix entity names in the history to get an extended query Q', which is issued to BM25 to get the top-D articles (not sentences as in the above version of BM25). We split these articles into sentences and rerank the sentences against the question Q using zero-shot SBERT (the same model as in CQR).
- **GPT-3.5**: We obtain the top-S sentences from SBERT Zero, and then use the *answer-check prompt* (Section 4.3) to rank relevant answers with GPT-3.5 by the probability of generating "yes". We do not use the *entity-check prompt* since it requires knowing the target entity, which is unknown to the ranker.
- **GPT-3.5 Fict**: This is the same as GPT-3.5, except that we replace each person's name in the conversations and answers with a made-up fictional name. We use gender-neutral replacements, so coreferences like "he" or "she" could refer to them. With the fictional names, the model is less able to rely on memorized information to produce answers, emphasizing its ability to find the answer using evidence. We do not report results on the OKB, Rare, and Uncommon subsets due to the changes of the target entities.
- **GPT-4 Fict**: This baseline modifies GPT-3.5 Fict to use GPT-4, which does not provide the probability of generated tokens. To break ties, we randomize the order of answers with the same predicted relevance. We repeat this tie-breaking 15 times and report the best ordering in terms of nDCG@3.
- **CONSENT**: CONSENT is our main method, using the methodology in Section 4.2 to rerank the top-S answers from SBERT Zero. SBERT Zero also provides the base relatedness score $Rel^{QA}_{Base}(Q, A)$.

4.5 Experimental Results

We consider research questions (RQs) evaluating CONSENT's performance on two conversational search datasets, investigating its performance across turns in a conversation, and investigating the impact of different components on CONSENT's performance.

RQ1: How does CONSENT perform compared to state-of-the-art baselines?
Results on the CONSENT test are shown in Table 4.1. The CONSENT method substantially outperforms the baselines across all CONSENT test subsets, with significant improvements over SBERT Zero and GPT in most cases. For example, CONSENT significantly outperforms GPT-4 Fict (14% better) in terms of nDCG@3 on All. This

Methods	All			OKB			Rare			Uncommon		
	nDCG	P	MRR	nDCG	P	MRR	nDCG	P	MRR	nDCG	P	MRR
BM25	0.070	0.073	0.100	0.037	0.028	0.047	0.065	0.067	0.090	0.106	0.120	0.162
ConvDR	0.120	0.120	0.155	0.069	0.067	0.087	0.120	0.114	0.153	0.162	0.175	0.217
ZeCo2	0.146	0.151	0.192	0.093	0.095	0.123	0.110	0.110	0.148	0.248	0.265	0.322
CQR	0.196	0.208	0.236	0.123	0.134	0.152	0.168	0.176	0.201	0.304	0.322	0.363
CoSPLADE	0.231	0.238	0.283	0.150	0.134	0.180	0.179	0.181	0.221	0.384	0.419	0.473
SBERT Zero	0.256	0.265	0.313	0.135	0.144	0.173	0.243	0.247	0.299	0.380	0.398	0.456
GPT-3.5 Fict	0.233	0.224	0.288	-	-	-	-	-	-	-	-	-
GPT-3.5	0.262	0.258	0.320	0.154	0.162	0.212	0.259	0.260	0.314	0.360	0.337	0.422
GPT-4 Fict	0.279†	0.275	0.338†	-	-	-	-	-	-	-	-	-
CONSENT	**0.319**†‡	**0.328**†‡	**0.385**†‡	**0.199**†‡	**0.204**†	**0.246**†	**0.300**†‡	**0.295**†	**0.357**†‡	**0.454**†‡	**0.488**†‡	**0.548**†‡

Table 4.1: Results on CONSENT test. The † and ‡ symbols denote significance (paired t-test, $p<0.05$) over SBERT Zero and all GPT baselines, respectively.

	BM25	ConvDR	ZeCo2	CQR	CoSPLADE	SBERT Zero	GPT-3.5	CONSENT
nDCG	0.045	0.047	0.108	0.272†	0.296†	0.232	**0.306**†	0.288†
P	0.042	0.047	0.112	0.280†	0.302†	0.244	**0.305**†	0.292†
MRR	0.064	0.049	0.156	0.339†	0.374†	0.295	**0.377**†	0.351†

Table 4.2: Results on sampled ConvQuestions. The † symbol denotes significance (paired t-test, $p<0.05$) over SBERT Zero.

trend could also be seen when we compare CONSENT against GPT-3.5. Indeed, regarding nDCG@3, CONSENT significantly outperforms GPT-3.5 on All, OKB, Rare and Uncommon (22%, 29%, 16% and 26% better, respectively). It is interesting to see that the nDCG@3 of GPT-3.5 drops 12% after changing real names to fictional names (GPT-3.5 Fict), suggesting that some of GPT's performance is due to memorizing related facts. The ConvDR, ZeCo2, CQR, and CoSPLADE neural retrieval methods for conversational search typically underperform the SBERT Zero and GPT methods, illustrating the difficulty of this setting with tail entities. BM25 is substantially worse than these methods, highlighting the big gap between ad-hoc and conversational search.

Results on the sampled ConvQuestions dataset, which focuses on popular in-KB entities rather than tail entities, are shown in Table 4.2. We use the joint contextualization and answer ranking technique on top of the base model SBERT Zero, thus we compare CONSENT against the base model to see the impact of such technique. The nDCG@3 of our method significantly outperforms that of SBERT Zero over four CONSENT test subsets and sampled ConvQuestions. This shows that our methodology really improves the search effectiveness. CONSENT performs competitively in Table 4.2. Here, GPT-3.5 and CoSPLADE perform best across metrics, with GPT-3.5 performing about 6% nDCG@3 better than CONSENT and CoSPLADE performing about 3% nDCG@3 better. The other neural baselines, ConvDR, ZeCo2, CQR, and SBERT Zero, consistently perform worse than CONSENT. BM25 generally performs the worst, as it did on CONSENT test.

Overall, these results indicate CONSENT substantially improves over existing approaches when conversations focus on tail entities while still performing well with conversations about popular entities. In the rest of this RQ, we dive into a test example to get more insights of our method CONSENT and the baseline GPT-3.5.

Q1′: What did Katy Gallagher accuse Mitch Fifield of?
A1′: In 2016, Katy Gallagher accused fellow senator Mitch Fifield of "mansplaining" during a debate in a Senate committee hearing regarding social services legislation, which subsequently went viral.

Q2′: Where Mitch got his education?
A2′: Mitch was born in Sydney, the son of two bank employees, and was educated at Barker College and the University of Sydney, where he graduated with a Bachelor of Arts.

Q3′: What role assumed in September 2015?
A3′: Mitch replaced the 29th Australian Prime Minister, Malcolm Turnbull, as Minister for Communications, in September 2015.

Q4′: What constituted the Council of BC org?
A4′: The Council of Barker College was originally constituted by the Barker College Ordinance of 1919.

Q5′: Where is located?
A5′: Barker College is situated on a 44-hectare (110-acre) campus in suburban Hornsby, 25 kilometres (16 mi) to the north of Sydney, with additional facilities located in the Blue Mountains, and The Grange, located at Mount Victoria.

Figure 4.5.1: A test conversation in our CONSENT dataset.

We can see a conversation from the CONSENT test set in Figure 4.5.1, the conversation has five turns and each turn consists of a question and a gold answer. CONSENT selects the first turn and the entity Mitch Fifield to contextualize the question Q2′, and ranks the candidate answers in Figure 4.5.2. The answer ranking by GPT-3.5 for the same question is shown in Figure 4.5.3. It can be seen that CONSENT finds a relevant answer (ranked 3rd) with support from Mitch Fifield who is selected as the relevant entity. Meanwhile, GPT-3.5 does not produce any relevant answers. Mitch Fifield likely does not appear

frequently in GPT-3.5's pre-training dataset, resulting in irrelevant answer candidates.

1. Educated at McKinnon Secondary College and the University of Melbourne, Paterson worked as a special adviser for Senator Mitch Fifield ...
2. During the campaign, Tallian was critical of Governor Mitch Daniels and his 2009 cut of US $450 million from primary, secondary and higher education in Indiana and she vowed to use her position to defend education funding.
3. Fifield was born in Sydney, New South Wales, the son of two bank employees, and was educated at Barker College and the University of Sydney ...

Figure 4.5.2: Answer ranking for Q2′ by CONSENT.

1. He was educated at Dookie Agricultural College, Parade College and La Trobe University, and has qualifications in economics and agricultural science.
2. He graduated from the University of Georgia School of Law in 1982 and joined the law firm of conservative Georgia congressman Pat Swindall.
3. He became politically active while studying at Monash University where he graduated with Bachelor of Jurisprudence and Bachelor of Laws degrees.

Figure 4.5.3: Answer ranking for Q2′ by GPT-3.5.

The rankings by CONSENT and GPT-3.5 for Q3′ are respectively shown in Figure 4.5.4 and 4.5.5. To interpret Q3′, CONSENT selects Mitch Fifield and Senate as relevant entities. With this contexualization, CONSENT is again more effective than GPT-3.5, with two relevant answers in the top-3, while only one relevant answer for GPT-3.5.

1. Fifield served as the Assistant Minister ... then became the Minister for Communications from 21 September 2015.
2. Mitch Fifield, since 21 September 2015.
3. Fifield replaced the current Australian Prime Minister, Malcolm Turnbull, as Minister for Communications, in September 2015.

Figure 4.5.4: Answer ranking for Q3′ by CONSENT.

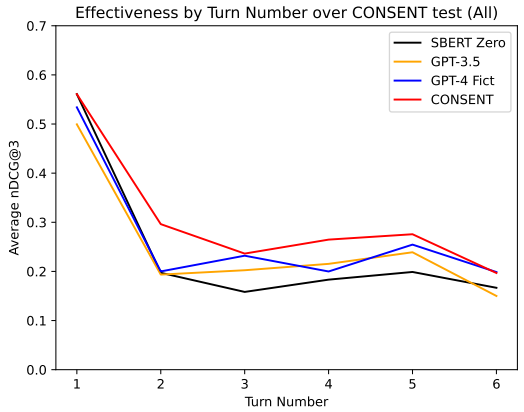

Figure 4.2: nDCG@3 for best performing methods on All by turn.

Settings			CONSENT			
K	L	Mode	All	OKB	Rare	Uncommon
1	1	DE	0.285	0.150	0.272	0.423
3	3	DE	**0.319**	**0.199**	**0.300**	0.454
5	5	DE	0.317	0.194	0.297	**0.456**
∞	∞	DE	0.311	0.187	0.290	0.452
3	3	NA	0.132	0.092	0.137	0.159
3	3	NC	0.308	0.175	0.294	0.445
3	3	NE	0.307	0.191	0.286	0.440
3	3	NT	0.288	0.167	0.272	0.417
3	3	NI	0.318	0.196	0.299	0.453

Table 4.3: Ablation study: nDCG@3 for CONSENT variants.

1. Fifield served as the Assistant Minister ... then became the Minister for Communications from 21 September 2015.
2. His appointment as Ambassador of Australia was announced on December 2015.
3. He was the Treasurer of Australia in the Abbott Government from 18 September 2013 until September 2015.

Figure 4.5.5: Answer ranking for Q3′ by GPT-3.5.

RQ2: How do methods perform across conversation turns?
In Figure 4.2 we report the nDCG@3 of top-performing methods on different turns on the full CONSENT test (All), with a maximum of 6 turns. These four methods perform similarly on the first turn, with CONSENT and SBERT Zero marginally higher than the GPT models. Performance sharply drops between the first and second turns for all methods, illustrating the importance of contextualizing questions. On turn two, CONSENT clearly outperforms SBERT Zero and the GPT models. On later turns, CONSENT continues to outperform GPT-3.5 and consistently matches or exceeds GPT-4 Fict's performance. The GPT models outperform SBERT Zero in almost all cases, which is likely due to the strong language understanding of these models. The fact that CONSENT consistently outperforms base model SBERT Zero after the first turn again confirms the importance of contextualization.

RQ3: What is the impact of CONSENT's components on its performance?
In this section, we investigate the impact of CONSENT's hyper-parameters K and L and the impact of excluding different types of relatedness scores. To do so, we define the following modes, which each removes some part of CONSENT.

- DE (default): This is the full method as described in Section 4.2.

- NA (no answer): We separate contextualization and answer ranking by removing answer relatedness scores, using only Rel^{TQ}, Rel^{EQ} and turn/entity parsimony for contextualization. We create an expanded question by prefixing the current question with the selected turns and all inferred types of the selected entities. The expanded question is fed through ConvDR to get the final result.

- NC (no turn context): We drop the relatedness scores Rel^{TQ} and Rel^{TA} and the turn parsimony component.

- NE (no entity): We drop the entity relatedness scores Rel^{EQ} and Rel^{EA} and the entity parsimony component.

- NT (no training): We skip training and instead set $\alpha_1, \alpha_2, ..., \alpha_7$ to one.

- NI (no integer constraint): We relax the integer constraint in the ILP solving.

Results using different hyper-parameter values and modes are shown in Table 4.3. Considering the impact of hyper-parameters when using the default mode (DE), CONSENT's performance decreases about 11% on the full test set when the number of selected turns and entities is reduced ($K = L = 1$). Increasing these hyper-parameters to 5 does not have much effect, with performance slightly better on the uncommon subset and slightly worse on the other subsets. Setting these hyper-parameters to ∞ also does not have much effect, with a small drop in performance across subsets. The default setting of $K = L = 3$ is thus a reasonable choice, so we fix this setting when reporting results with other CONSENT modes.

Compared to the full method (DE), removing answers results in the largest drop on the full test set (58%), which indicates that considering cues from candidate answers is essential. Similarly large drops in performance also occur on the OKB, rare, and uncommon subsets. Setting alphas to fixed values rather than determining them during training (NT) results in the second largest drop in performance of about 10% on the full data. The remaining modes result in substantially smaller drops of 4% or less when removing turn context (NC), removing entity relatedness scores (NE), or removing the integer constraint (NI). All of these modes result in performance drops on all subsets. These results illustrate that CONSENT is robust to changes in its components and hyper-parameters as long as answer cues are considered.

4.6 Related Work

Conversational Search. Prior works have focused on the challenges C1 and C2 (see Section 4.1). One approach is to employ co-reference resolution and related techniques to rewrite user utterances into complete, self-contained questions (e.g., [Yu et al., 2020, Vakulenko et al., 2021b, Tredici et al., 2021, Voskarides et al., 2020, Vakulenko et al., 2021a, Elgohary et al., 2019, Kim et al., 2021]). These methods are brittle, though. The second line of methods is to use neural encoders and leverage language models for contextualizing the user's inputs (e.g., [Gao et al., 2023, Krasakis et al., 2022, Lin et al., 2021]). Some techniques focus on incorporating only the immediately preceding and/or the very first turn of a conversation. This is a good heuristic for capturing the relevant context, but easily fails for sophisticated conversations. Recent methods use the entire conversational history to encode context, relying on training data for computing appropriate (attention) weights (e.g., [Qu et al., 2019, Christmann et al., 2022]). This is viable when labeled conversations are abundant, but with limited amounts of training data the approach is all but robust.

Popular instantiations of this line of contextualization methods include the following. ConvDR [Yu et al., 2021] is a dense retrieval method [Lin et al., 2020], where the current question is prefixed with the entire history to encode a vector representation. These vectors are compared to encodings of candidate answers by computing inner products. Zeco2 [Krasakis et al., 2022] uses ColBERT [Khattab and Zaharia, 2020] to build representations in a zero-shot manner. CoSPLADE [Hai et al., 2023] follows the paradigm of learned sparse retrieval, to identify salient keywords in the history and leverage these for question understanding. Obviously, generative models like GPT-3.5 [Brown et al., 2020] and GPT-4 [OpenAI, 2023] can also be utilized to encode questions with their conversational history. None of these methods give consideration to out-of-KB and tail entities.

Benchmarks. Prior benchmarks for conversational search focus on prominent entities; there are hardly any questions about long-tail or out-of-KB entities [Elgohary et al., 2019, Aliannejadi et al., 2021, Dalton et al., 2019, Dalton et al., 2020, Dalton et al., 2021, Christmann et al., 2019, Christmann et al., 2022, Adlakha et al., 2021]. Some of these are specifically designed to evaluate subtasks like question rewriting (CANARD [Elgohary et al., 2019]), conversation clarification (ClariQ [Aliannejadi et al., 2021]), or detecting topical shifts in conversations (TopiOCQA [Adlakha et al., 2021]). None of the existing benchmarks for conversational search contains a substantial number of conversations focused on long-tail entities.

4.7 Summary

In this work, we proposed the CONSENT method for conversational search with tail entities. By jointly performing answer ranking and contextualization, CONSENT outperforms a range of baselines by a substantial margin. On a new benchmark of conversations about tail entities, CONSENT substantially improves over state-of-the-art methods while remaining competitive on the ConvQuestions benchmark about popular entities. Ablations of the CONSENT method confirm the importance of selective contextualization.

Chapter 5

EECATS: Efficient and Effective Conversational Search with Tail Entity Selection

5.1 Introduction

5.1.1 Motivation and Problem

Conversational information retrieval (CIR) systems provide user with answers to a sequence of questions in a multi-turn session [Zamani et al., 2023]. As we see in Chapter 4, user inputs are too informal to be processed as self-contained queries in the CIR setting, so it is crucial to combine questions with context cues from previous turns. Additionally, the questions themselves are often about emerging entities and events, which may not exist in Wikipedia. In this chapter, a system answer is a text passage extracted from news articles or Wikipedia pages, rather than answer sentences as in Chapter 4. Typically, in the passage setting, the challenge of search efficiency is greater than in the sentence setting, and efficiency is one of the key factors we explore in this chapter.

To illustrate our CIR setting, consider the example conversation on the left side of Figure 5.1 referring to the 2024 Olympic competition in women's table tennis. The example shows two key problems:

- All follow-up questions Q2 through Q4 are far from being self-contained and need context for proper interpretation. Q3 is also highly informal and would be meaningless outside a conversation. It needs to be *contextualized* with the previous turns.
- The conversation refers to entities in the *long tail* of popularity. In Q4, "her team" refers to Annett Kaufmann and the sports club SV Boeblingen. However, Annett Kaufmann is not featured in the English Wikipedia at all , and the mention "Boeblin-

gen" in A3 would likely be associated with a town rather than the sports club. Thus, entity linking to Wikipedia and obtaining background knowledge is not an option in this case.

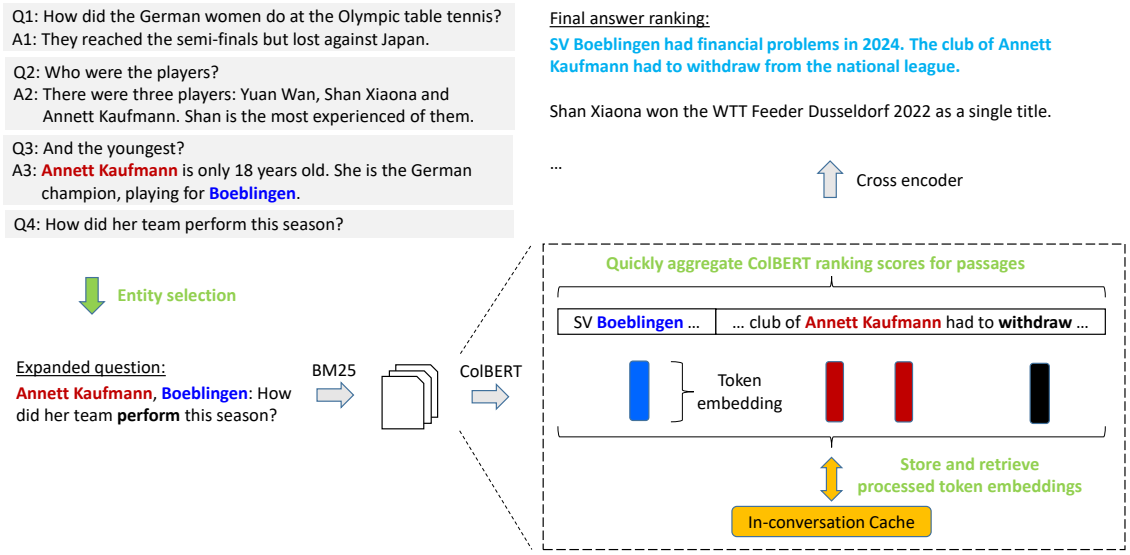

Figure 5.1: Overview design of EECATS with entity selection to build the expanded question. Such expanded question is fed through a pipeline of BM25, ColBERT and cross-encoder for passage ranking. We leverage quick ColBERT ranking score aggregation and in-conversation cache for efficiency. Entity selection plays a crucial role in search effectiveness.

5.1.2 Limitations

Prior works have addressed these issues to some degree [Hai et al., 2023, Christmann et al., 2023, Tran et al., 2024, Krasakis et al., 2022]. Contextualization with information from previous turns can take different forms: i) rewriting the user input into a full-fledged and self-contained question, ii) expanding the input with selected turns of the prior conversation and/or relevant entities mentioned there, or iii) identifying relevant entities and obtaining background information (e.g. from Wikipedia) for completing the question.

For coping with tail entities, our prior work CONSENT [Tran et al., 2024] proposed a framework of question-mention relatedness based on SBERT [Reimers and Gurevych, 2019] similarities. However, SBERT embeddings are designed for linguistic similarity and are agnostic to the specific task of relating entity mentions to informal questions.

In addition, a second problematic dimension is that prior methods are computationally expensive, by extensively relying on cross-encoders, graph neural networks or integer

programming. The goal of this chapter is to improve the *efficiency* of *CIR with long-tail entities*, while being competitive in *effectiveness*.

5.1.3 Approach

Our method, EECATS for Efficient and Effective Conversational passage retrieval with tAil enTity Selection, reconciles efficiency and effectiveness. We adopt the idea of expanding questions with entity mentions including long-tail entities from our prior work CONSENT [Tran et al., 2024], but devise very different algorithms with much lower computational cost. EECATS has two stages:

1. **Question Contextualization:** For each user input, EECATS selects a small number (e.g. top-2) of the most relevant entity mentions from prior turns. Entity linking is exercised only for clearly unambiguous names, while all other mentions are kept at the level of surface names. We judiciously fine-tune an SBERT model for entity-question relatedness, using the respective turn (question and answer) text for unlinkable mentions, and Wikipedia paragraph with the turn for linked entities. The selected entity mentions are added to the expanded question.
2. **Retrieval and Answering:** EECATS feeds the expanded question to a pipeline of filtering and ranking steps. First, relevant documents are found by BM25; then the top-100 passages are computed by a bi-encoder, namely, ColBERT [Santhanam et al., 2022]. The final step is the re-ranking of these candidates by a cross-encoder [Nogueira and Cho, 2019]. The rationale of this three-step pipeline is to minimize expensive computations. A key point is to avoid repeated computation of embeddings by using a fast embedding aggregation technique and an in-conversation cache.

5.1.4 Contributions

The salient contributions of this work are:

- We propose a new way of contextualizing user inputs in CIR. By *fine-tuning* SBERT to the specific task of selecting relevant entity mentions from the conversational history, we present the retrieval stage with informative cues.
- We devise techniques for reducing the computational costs, including *efficient aggregation* of judiciously computed and cached embeddings.
- Experiments show that EECATS outperforms the best prior baselines. For questions about tail entities, EECATS is significantly better in nDCG@3 than the best GPT competitor. For efficiency, EECATS improves latency by a factor of 37x. Our code

and data are published on GitHub [1].

5.2 Overview

Given a user question Q_i with prior conversation history $H = \{(Q_1, A_1), (Q_2, A_2), ..., (Q_{i-1}, A_{i-1})\}$, we aim to find the most relevant passage-level answers to Q_i from a collection of news articles and Wikipedia pages. Figure 5.1 gives a pictorial overview of the EECATS architecture and methodology.

As we pay particular attention to tail entities, we identify entity mentions in prior turns $(Q_j, A_j)(1 \leq j < i)$ in two categories:

- **In-KB entities** are those mentions, in previous questions or answers, that can be linked to a knowledge base (KB) with high confidence. In the example above, these would be Olympic table tennis tournament, Japan, Germany (the countries) and Shan Xiaona, all of which have their own Wikipedia articles.
- **Out-of-KB (OKB) entities** are those mentions that either are not featured in Wikipedia at all or cannot be disambiguated with confidence. In the example, these are Yuan Wan, Annett Kaufmann and Boeblingen. The first two do not have Wikipedia articles, whereas the latter is too ambiguous, with one meaning denoting the city which would be misleading here.

Both in-KB and out-of-KB entities are candidates for question contextualization. EECATS selects the top most-relevant mentions and adds them to an expanded question. This selection bases on the respective turn (including both question and answer) for out-of-KB entities, and both short paragraph from Wikipedia and the respective turn for in-KB entities.

For retrieving passages and ranking answers, EECATS runs the expanded question through a judiciously designed 3-stage pipeline of filtering and ranking. To avoid the computational cost of processing a huge number of passages, we first employ a BM25 retriever at the document level, and then run a light-weight ColBERT bi-encoder on the passages of the best first-round documents only. These two stages serve to narrow down the candidate space in an efficient way, but the ColBERT ranking is not competitive. Therefore, we have an effectiveness-oriented cross-encoder as stage-3 ranker of the second-round results. Note that this final cross-encoder needs to process only the top-k (typically top-100) passages that come out of the stage-2 ranker.

In the second stage, we need to compute a large amount of embedding vectors for

[1] https://github.com/haidangtran1989/EECATS

tokens in the passages using ColBERT. To mitigate this cost, we devised techniques for lowering the GPU consumption and accelerating the query response times. Most notably, we leverage a light-weight aggregation of sentence-level embeddings, in-conversation caching and reuse of embeddings.

5.3 Relevant Entity Selection

We need to quantify the relatedness between current question and each entity mention in the preceding turns. To this end, we introduce various concepts.

Definition 33 (Text Representation) *We use SBERT [Reimers and Gurevych, 2019] for encoding a given text snippet t. The text representation $Rep^T(t)$ of t is the embedding produced by feeding t through SBERT.*

This text representation is a building block for the other encodings in this work. SBERT is our choice for its efficiency. Note that we apply it only to relatively short snippets, not necessarily well-formed sentences but not much longer.

Definition 34 (Entity Context) *Given an entity mention e in the j-th turn of the conversation history, the entity context $Con(e)$ of mention e is defined as follows:*

- *If e is out-of-KB, $Con(e)$ is the j-th turn (i.e., the concatenation of the j-th question and the respective answer).*
- *If e is in-KB and the mention can be linked to the KB with confidence above a threshold, then $Con(e)$ is the concatenation of the first Wikipedia paragraph of e and turn j.*

We consider the in-KB entity descriptions from Wikipedia paragraphs as potentially valuable background for the relevant entity selection. For example, *"Shan Xiaona is a Chinese-born naturalised German table tennis player..."* is useful information about the entity. The turn text where the entity is mentioned can also provide strong cues, most importantly, semantic types. For example, in turn 3 of the example conversation in Figure 5.1, we could infer that *"Boeblingen"* refers to a sports team and not to a city, which is useful for interpreting Q_4.

We combine the mention's surface string (as it appears in the turn) and its background context into a knowledge representation.

Definition 35 (Knowledge Representation) *Given an entity mention e, the knowledge representation $Rep^K(e)$ of e is $Rep^K(e) = Rep^T(e\ [SEP]\ Con(e))$.*

By this definition, two identical out-of-KB mentions in different turns can have different knowledge representations. This is our intention, as the topical context may change in the course of conversation. To capture the flow of the user's information needs across turns, we need to compute the following representation.

Definition 36 (Flow Representation) *Given a turn j and the current turn number i $(1 \leq j < i)$, the flow representation $Rep^F(j,i)$ of the $j-th$ turn with regard to the current turn i is computed as $Rep^F(j,i) = Rep^T(Q_j\ Q_{j+1}\ ...\ Q_{i-1})$.*

By concatenating questions from the j-th up to but excluding the current turn, we preserve the order of the evolving context. We do not include answers in this representation, as this could excessively long texts that cannot be easily processed with a light-weight encoder (such as SBERT, intentionally excluding heavy-weight alternatives like GPT-4). Observing that both entity knowledge and the conversation flow contribute to relevant entity selection, we combine these into the final entity representation.

Definition 37 (Entity Representation) *Given a mention e in the j-th turn, the entity representation $Rep^E(e,j,i)$ of e with regard to the current turn i is $Rep^E(e,j,i) = Rep^K(e) + Rep^F(j,i)$, where the sum operator is the element-wise vector sum.*

Finally, we define a question representation that can be compared to entity representations, and then calculate their relatedness.

Definition 38 (Question Representation) *For question Q_i (i is the current turn number), the question representation $Rep^Q(Q_i)$ of Q_i is defined as $Rep^Q(Q_i) = Rep^T(Q_i)$.*

Definition 39 (Entity Relatedness) *Given an entity mention e in the j-th turn, the relatedness between e and current question Q_i $(j < i)$ is the inner product of their representations: $Rel(e,j,Q_i) = \langle Rep^E(e,j,i), Rep^Q(Q_i)\rangle$.*

Our design captures both static background about entities and dynamic cues from the conversation for entity selection. We compute the relatedness of the current question to all mentions in the conversation (questions and answers) and select the top-L most related entities. The selected ones are added, in their surface forms, to the current question as a prefix, forming the expanded question that is fed into the retrieval stages. For example, Q4 of the conversation in Figure 5.1 will be expanded into "Annett Kaufmann, Boeblingen: How did her team perform this season?".

Fine-Tuning SBERT for Entity Selection. The SBERT bi-encoder is trained for comparing two sentences under purely linguistic aspects. Our setting differs in that we

compare an informal question against a candidate entity and its respective turn. To overcome this "impedance mismatch", we derive training data for the specific task of scoring entity relevance for a given question. This is then used for fine-tuning SBERT.

Definition 40 (Positive and Negative Training Entities) *We use an LLM to automatically complete a training set of questions. Later we use these complete questions as ground-truth annotations. We treat all history mentions that appear in the completed question as being relevant for the interpretation of the original question (positive training entities), and others as irrelevant ones (negative training entities).*

Example 15 *Assume that the conversation in Figure 5.1 is in the training set. The question Q4: How did her team perform this season? would be completed by GPT-4 into Q4*: How did the Boeblingen team of Annett Kaufmann perform this season?. History mentions Annett Kaufmann and Boeblingen are positive training entities to Q4 since they appear in Q4*, while Yuan Wan and Shan Xiaona are negative training entities.*

Definition 41 (Training Data) *Let E^+ and E^- denote the positive training entities and the negative training entities to a given question Q_i. We can fine-tune SBERT by giving it pairs of entities and their respective turns with the training objective to score $Rel(e^+, j^+, Q_i)$ higher than $Rel(e^-, j^-, Q_i)$ for every pair (e^+, e^-) where $e^+ \in E^+$ and $e^- \in E^-$, j^+ and j^- are respective turn numbers of e^+ and e^-. Each pair (e^+, e^-) forms a single sample in the training data for additional fine-tuning SBERT.*

Furthermore, we use the pairwise ranking loss [Cao et al., 2006] to train for entity selection, starting from initial SBERT [2] parameter values loaded from Hugging Face. For this fine-tuning of SBERT, we use a batch size of 64 and an initial learning rate $3e-5$ and two epochs. In total, we used $10,940$ training pairs (e^+, e^-) derived from 840 completed questions of the training set for EECATS (see Section 5.5). Note that this is a training-time method only. At inference time, no information about relevant entities is available upfront.

5.4 Passage Ranking

We feed the expanded question into a pipeline of retrieval and re-ranking stages.

[2] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

5.4.1 Document Retrieval with BM25

To avoid having to process a huge number of passages with expensive neural models, we first pre-filter the corpus at the document level, using the highly efficient and reasonably effective BM25 method [Robertson and Zaragoza, 2009]. Documents are entire news articles or entire Wikipedia pages. We retain a pool of highest scoring results, like the top-100 documents. For the following stages, all passages in these documents are candidates to be processed.

5.4.2 Passage Ranking with Bi-Encoder

The bi-encoder computes representation vectors for the tokens in the expanded question and each candidate passage separately, aggregates these in a judicious manner, and computes scores for ranking passages. In our implementation, we choose to employ ColBERT [Santhanam et al., 2022] as a powerful and relatively light-weight bi-encoder. In the following we present how the representation vectors are constructed and compared.

Definition 42 (Expanded Question Token Representation) *The token representation $Rep^{EQW}(w)$ of a token w in the expanded question q' is the token embedding of w when we feed q' through the bi-encoder.*

For the representation of candidate passages, we start with encoding tokens within their sentence-level contexts. For encoding entire passages, we build on aggregation techniques for combining multiple embedding vectors, as this is much more efficient (and still effective) compared to encodings for longer full-passage texts.

Definition 43 (Sentence Token Representation) *Given a token w in a sentence S, the sentence token representation $Rep^{SW}(w)$ of w is the token embedding of w when we feed S through the bi-encoder.*

Since questions in a conversation are related to each other, it is likely that various sentences of the top-100 documents from BM25 are repeated in different turns. We store sentence token representations in an in-conversation cache, and reuse them when a sentence re-appears in candidate passages of a future turn. This caching opportunity is another factor in our design choice for encoding sentences rather than entire passage, as the smaller granularity increases the chance of cache hits.

With the independently computed question token representations and sentence token representations, we can now define how these are aggregated into vectors at the passage level for scoring and ranking.

Definition 44 (Maximal Similarity Vector of Sentence) *For expanded question q' and sentence S with respective token sequences $w_1, w_2, ..., w_H$ and $v_1, v_2, ..., v_G$; the maximal similarity vector of S against q' is calculated as $Rep^S(S, q') = \{r_1, r_2, ..., r_H\}$ where $r_k = \max_{j=1}^{G} Cos(Rep^{EQW}(w_k), Rep^{SW}(v_j))(1 \leq k \leq H)$ and Cos is the Cosine similarity.*

Definition 45 (Maximal Similarity Vector of Passage) *For expanded question q' and passage A with sentences $S_1, S_2, ..., S_Z$, the maximal similarity vector $Rep^P(A, q')$ of A against q' is $Rep^P(A, q') = \max_{k=1}^{Z} Rep^S(S_k, q')$ where max is element-wise maximum operator.*

The element-wise maximum operator is a light-weight technique, which allows us to efficiently compute maximal similarity vectors for passages of various lengths by aggregating from the smaller granularity of their sentences. This is a design choice towards an efficient solution. The computation cost of the entire stage-2 filtering consists of running all single sentences (not in cache) through the bi-encoder. All further aggregation steps are very light-weight. Subsequently, the bi-encoder infers ranking scores as follows.

Definition 46 (Bi-Encoder Ranking Score) *Consider a passage A with maximal similarity vector $Rep^P(A, q') = \{r'_1, r'_2, ..., r'_H\}$, where H is the number of tokens in the expanded question q' and each r'_i is the maximal similarity between token embeddings of A and the i-th token embedding of $q'(1 \leq i \leq H)$. We obtain the bi-encoder ranking score for A by summing up the maximal similarities over all question tokens: $\sum_{k=1}^{H} r'_k$.*

5.4.3 Shortlist Ranking with Cross-Encoder

The bi-encoder-based ranking of all passages gives us a shortlist of top-C candidates, say with $C = 100$. For these and only this small set of best candidates, we employ a computationally heavier cross-encoder.

Definition 47 (Cross-encoder Ranking Score) *Given the expanded question q' and a candidate passage, we concatenate them and run this combined token sequence through the cross-encoder [3] [Nogueira and Cho, 2019] model. The cross-encoder's output is the final score of the passage with regard to the question.*

[3] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

5.5 Experimental Setup

We conduct experiments on two types of benchmarks: (i) a new EECATS dataset that focuses on questions about tail entities that can be answered with short passages, and (ii) the TREC Conversational Assistance (CAsT) 2019 and 2020 datasets, which contain common entities and have been widely used in prior work (e.g. [Yu et al., 2020, Vakulenko et al., 2021b, Lin et al., 2021, Kostric and Balog, 2024, Hai et al., 2023, Yu et al., 2021]). To improve CIR efficiency without sacrificing effectiveness, we use these benchmarks to evaluate both effectiveness and efficiency.

Metrics. For effectiveness, we measure nDCG@3, Precision@1, and MRR@3 (abbreviated as nDCG, P and MRR) on the benchmarks' test questions and report micro-averaged numbers (i.e., average over all questions, not over conversations). For efficiency we measure the average query latency (lat) and the average number of Tera FLOPS (TF) per question. The latter is estimated by the calflops tool [4] based on tensor sizes. With the exception of the GPT methods, all efficiency metrics are computed on the same NVIDIA Tesla A40 GPUs. For the GPT methods, we measure the response times of the API calls; we cannot report Tera FLOPS numbers as we do not have access to run-time details.

5.5.1 EECATS Tail Entity Benchmark

We construct a new benchmark to support our focus on tail entities with passage answers. To do so, we adapt the automatic benchmark construction approach taken by our prior work CONSENT [Tran et al., 2024] to produce minimal passage answers that do not contain redundant information (i.e., passages should only contain information needed to answer the question).

# Statistics	Dataset	Training set	Test set
# Conversations	335	221	114
# Questions	1276	840	436
Average # Turns	3.82	3.8	3.82

Table 5.1: Statistics for the EECATS Dataset.

This approach starts with sampling gold answers containing statements about a target entity from a current (2024) version of Wikipedia. Then GPT-3.5 [Brown et al., 2020] is used to generate a question that fits each gold answer, with prompts following CONSENT.

[4] https://huggingface.co/spaces/MrYXJ/calculate-model-flops

To construct entire conversations with occasional shifts of pivot entities, we also follow the procedure of the prior work CONSENT. We split the set of conversations into a training set and a test set.

To assert the long-tail nature of the entities, we follow CONSENT and run all questions, for all methods, on a document collection of news articles from 2018 [Hoffart et al., 2014] and Wikipedia articles from 2017. At this time, many of the entities were indeed not (yet) featured in Wikipedia, but covered in the contents of some articles. Each question is assigned to one of three popularity groups: out-of-KB, rare-in-KB, and uncommon-in-KB, based on the popularity of the respective pivot entity in the answer collection from 2017–2018. Popularity is determined by the number of in-links, using thresholds for the three groups as in the prior work CONSENT.

To judge system answers, we adapt the automatic judgment approach proposed by CONSENT, which is built on top of GPT-4 [OpenAI, 2023]. The authors report this approach closely agreed with human judgments (e.g., achieving an accuracy of 90.9% when an answer sample was labeled by crowd workers). Given our requirement that the passage answers returned be minimal (i.e., contain no unneeded information), we change a relevant passage's label to non-relevant whenever the passage has redundant information. An answer has redundant information if, after removing its first or last sentence, the resulting sub-passage is still judged relevant by CONSENT's automatic judgment approach. We call this tail-entity benchmark the EECATS dataset. Table 5.1 gives statistics about the dataset.

5.5.2 TREC Conversational Assistance Benchmarks

The TREC CAsT 2019 [Dalton et al., 2019] and 2020 [Dalton et al., 2020] benchmarks consist of 173 questions over 20 conversations and 216 questions over 25 conversations, respectively. The benchmark questions are paired with answer passages from the MS MARCO [Craswell et al., 2021a] and TREC Complex Answer Retrieval [Dietz et al., 2018] collections. When evaluating on CAsT, we train all methods on the CANARD [Elgohary et al., 2019], because CAsT does not have a separate training set.

5.5.3 Implementation and Hyper-Parameters

To extract entity mentions from text, we use Spacy [5]. We use Dexter [Ceccarelli et al., 2013] to link mentions to in-KB entities, with Dexter's confidence threshold set high

[5] https://spacy.io/

(98%). On the CAsT 2019 and 2020 benchmarks, which are less entity focused than the EECATS benchmark, we use Spacy to identify noun phrases and named entities that we treat as mentions.

When providing a method with the conversation history along with a new question, we always use *canonicalized answers* for the past answers, to ensure the history fits with the current question and the history is identical for all methods. So we exclude the risk of accumulating effects from erroneous answers, for fair comparison. The canonicalized answers are obtained as follows. On the EECATS and CAsT 2020 benchmarks, we use the provided gold answers. On the CAsT 2019 benchmark, which only provides judgments for pooled responses from various systems, we select an answer with the highest relevance label as canonical answer.

To set the hyper-parameter L, the maximum number of entities selected for question contextualization, we take an adaptive approach based on the relatedness scores. When the difference of scores between the top-1 entity and the second-best entity is higher than 1.0, then we solely pick the top-1 entity; otherwise we use $L = 2$. This adaptive setting is the default choice for L. The threshold of 1.0 follows the margin used in the pair-wise ranking loss that we use to fine-tune SBERT for entity relatedness. In pilot experiments, increasing L beyond 2 did not improve performance.

Methods	**Lat**	**TF**	**All**			**Out-of-KB**			**Rare**			**Uncommon**		
	(s)		nDCG	P	MRR	nDCG	P	MRR	nDCG	P	MRR	nDCG	P	MRR
ConvDR	**0.15**	**0.5**	0.060	0.062	0.084	0.045	0.049	0.063	0.068	0.067	0.088	0.066	0.070	0.102
CQR GPT	66.96	-	0.102	0.078	0.134	0.089	0.056	0.109	0.090	0.089	0.125	0.138	0.088	0.180
CoS + T5	51.33	573	0.124	0.147	0.177	0.136	0.148	0.183	0.066	0.089	0.106	0.200	0.237	0.279
GPT-3.5	50.18	-	0.136	0.138	0.187	0.124	0.120	0.165	0.139	0.150	0.189	0.147	0.140	0.212
GPT-4o Ano	101.52	-	0.173	0.188	0.233	-	-	-	-	-	-	-	-	-
CONSENT	96.95	248	0.193	0.227	0.265	0.146	0.141	0.188	0.144	0.178	0.206	0.327	**0.412**	0.452
EECATS	1.34	21.7	**0.277**†‡	**0.278**†‡	**0.339**†‡	**0.249**†‡	**0.232**†‡	**0.299**†‡	**0.249**†‡	**0.228**‡	**0.289**†‡	**0.358**‡	**0.412**‡	**0.468**‡

Table 5.2: Ranking effectiveness and efficiency (lat and TF stand for average quesion latency and Tera FLOPS) on the four groups of the EECATS test set. The † and ‡ symbols indicate significant improvements over the CONSENT and all GPT baselines (CQR GPT and GPT 3.5 and GPT 4o Ano), respectively (paired t-test, $p<0.05$).

5.6 Evaluation

In this section, we address research questions about the effectiveness and efficiency of EECATS and the importance of its components. We include the following strong

baselines in the evaluation:

- **ConvDR**: This method [Yu et al., 2021] is a conversational dense retriever, obtaining the relevance score as the inner product of question and answer representations.

- **CoS + T5**: CoSPLADE [Hai et al., 2023] is a cutting-edge CIR method, where salient terms in the history are used to support question understanding. CoSPLADE uses these salient terms for sparse retrieval, and subsequently monoT5 [Nogueira et al., 2020] is used to re-rank passages in the top articles returned by CoSPLADE.

- **GPT-3.5**: GPT-3.5 [Brown et al., 2020] (Turbo) is a large language model. All mentions in the history are prefixed to the current question Q_i to build an expanded question Q'_i. To identify candidate passages, this method first uses Q'_i as an input to BM25 to search for the top-D articles. Next, the passages from the top-D articles are reranked against Q_i using the same cross-encoder in EECATS. Finally, the top-G passages from the cross-encoder's ranking are reranked again by asking GPT-3.5 whether the passage is relevant to the question given the conversation history. The probability of generating "yes" token is the final ranking score of the respective passage.

- **CQR GPT**: This method uses GPT 3.5 (Turbo) to rewrite conversation questions (with prior history) into self-contained ones. Like the above *GPT-3.5* baseline, the self-contained questions are fed through BM25 and cross-encoder. Subsequently, GPT-3.5 (Turbo) is asked if top candidate passages are relevant to the self-contained questions and probability of "yes" token is also used as the relevance score.

- **GPT-4o Ano**: This is the same as GPT-3.5 baseline, except that we use GPT-4o and consistently replace names in the test conversations with anonymous names. We replace names to keep a realistic setting with out-of-KB entities. As GPT-4o has parametric memory covering recent Wikipedia and huge news collections, it would treat most of our out-of-KB-as-of-2018 entities as richly covered topics. This would boil down to comparing our old corpus against a much larger recent corpus. Hence the anonymization of entity names is introduced. The popularity groups are no longer meaningful after the name replacement, so we only show the performance on the All group. Note that we give an advantage to GPT-3.5 baseline by not replacing real names to anonymous ones.

- **CONSENT**: CONSENT [Tran et al., 2024] is the state-of-the-art conversational search method for tail entities, which jointly performs contextualization and answer ranking. However, this approach is slow because the joint processing relies on integer programming. CONSENT obtains candidate passages in the same way as the GPT-3.5 baseline, and then CONSENT re-ranks these passages.

In the evaluation with EECATS benchmark, we train ConvDR and CoSPLADE + T5 and CONSENT using the same training data used with the EECATS method.

5.6.1 Effectiveness and Efficiency

In this section, we consider research questions (RQs) about effectiveness and efficiency compared to baselines (RQ1), effectiveness on common entities (RQ2), redundant information in answers (RQ3), and how effectiveness changes across turns (RQ4).

RQ1: How does EECATS perform on tail entities compared to strong baselines? Table 5.2 reports the results for the EECATS benchmark. On almost all groups, EECATS significantly outperforms the state-of-the-art method CONSENT with nDCG@3 being 8 percentage points higher for All conversations, and even higher for the out-of-KB and Rare groups. The performance gains of EECATS over the large language models GPT-3.5 and GPT-4o Ano is even higher. Obviously, the LLMs struggle with long-tail entities that are not well covered in their training data. The highest contrast in nDCG@3 between EECATS and others is shown on the out-of-KB and Rare groups, which consist of entities with the lowest popularities.

The effectiveness of EECATS does not come at the cost of efficiency. EECATS is more than 37 times faster in question latency and more than 11 times faster in Giga FLOPS than all baselines but ConvDR, which struggles to perform well with the limited training examples available in EECATS.

Figure 5.6.1 shows a conversation from the EECATS testset. EECATS identifies Alvy Ray Smith as the relevant entity for Q3′, then forms the respective expanded question "Alvy Ray Smith: What significant contribution was made in computer graphics?". Thanks to this expansion, the top-1 ranking of EECATS is relevant, as we can see in Figure 5.6.2. By contrast, GPT-3.5 fails to understand Q3′ correctly, resulting in an irrelevant ranking, as shown in Figure 5.6.3. The top-1 answer sounds relevant, but it is actually about another entity (Edwin Catmull). Moreover, second and third-ranked answers are also not relevant. Alvy Ray Smith is not likely to appear frequently in the pre-training dataset of GPT-3.5, leading to a misunderstanding of Q3′.

Q1′: What did John Lasseter do after being fired from Disney?
A1′: While putting together a crew for the planned feature, John Lasseter had made some contacts in the computer industry, among them Alvy Ray Smith and Ed Catmull at Lucasfilm Computer Graphics Group. After being fired, and feeling glum knowing his employment with Disney was to end shortly, John Lasseter visited a computer graphics conference in November 1983 at the Queen Mary in Long Beach, where he met and talked to Catmull again. Catmull inquired about The Brave Little Toaster, which John Lasseter explained had been shelved...

Q2′: Where does Alvy reside now?
A2′: Alvy retired from Microsoft in 1999 to spend his time giving talks, making digital photographs, doing scholarly genealogy, and researching technical history. He lives in Seattle, Washington. In 2010 Alvy married Alison Gopnik, author and Professor of Psychology at the University of California, Berkeley.

Q3′: What significant contribution was made in computer graphics?
A3′: While at Xerox PARC in 1974, Alvy worked with Richard Shoup on SuperPaint, one of the first computer raster graphics editor, or 'paint', programs. Alvy's major contribution to this software was the creation of the HSV color space, also known as HSB. He created his first computer animations on the SuperPaint system.

Figure 5.6.1: A test conversation in our EECATS dataset.

1. Smith's major contribution to this software was the creation of the HSV color space, also known as HSB. He created his first computer animations on the SuperPaint system...
2. Alvy Ray Smith III is an American noted pioneer in computer graphics.
3. He is a founding member of three organizations which pioneered computer graphics for digital special effects and film with Edwin Catmull and Alvy Ray Smith, including; New York Institute of Technology Computer Graphics Lab, Lucasfilm Computer Division, and Pixar, financed by Steve Jobs.

Figure 5.6.2: Answer ranking for Q3′ by EECATS.

1. In 2006, he was awarded with the IEEE John von Neumann Medal for pioneering contributions to the field of computer graphics in modeling, animation and rendering.
2. Graphics is responsible for displaying art and image data effectively and meaningfully to the user. It is also used for processing image data received from the physical world. Computer graphic development has had a significant impact on many types of media and has revolutionized animation, movies, advertising, video games, and graphic design generally.
3. Catmull saw Sutherland's computer drawing program Sketchpad and the new field of computer graphics in general as a major fundament in the future of animation, combining his love for both technology and animation, and decided to be a part of the revolution from the beginning...

Figure 5.6.3: Answer ranking for Q3′ by GPT-3.5.

RQ2: When dealing with common entities, does EECATS remain effective? We consider the performance of EECATS and baselines on CAsT 2019 and 2020, where the focus is on common entities. These datasets provide passage collections, so all methods use BM25 to find passages. To provide a fair comparison, all methods use the Mono T5 model [6] [Nogueira et al., 2020] for re-ranking. The query input to Mono T5 is in the format *Query:* Q_i. *Context:* $Q_1, Q_2, ...Q_{i-1}$. *Keywords:* K (i is the current turn). The keywords K are extracted salient terms for CoS + T5 and selected entity mentions for CONSENT and EECATS.

Methods	CAsT 19			CAsT 20		
	nDCG	P	MRR	nDCG	P	MRR
ConvDR	0.466	0.555	0.621	0.327	0.477	0.546
CONSENT	0.457	0.549	0.621	0.334	0.366	0.433
CoS + T5	0.478	**0.630**	**0.707**	0.368	**0.519**	**0.578**
EECATS	**0.482**	0.566	0.647	**0.379**	0.468	0.520

Table 5.3: Results on the TREC CAsT 2019 and 2020 test sets.

The results in Table 5.3 show that EECATS continues to perform well on these datasets. On both CAsT 2019 and 2020, EECATS achieves the best nDCG@3. However, CoS + T5 performs better on P@1 and MRR@3. This is because questions in CAsT focus on common entities, which are frequently mentioned in the pre-training data of the LMs inside CoS + T5.

[6] https://huggingface.co/castorini/monot5-base-msmarco-10k

RQ3: How do the methods perform when relevant answers are allowed to contain redundant information? Recall that we require relevant passages to contain no relevant sub-passages. To understand the impact of this requirement on our results, we evaluate all methods with this constraint removed, so that redundant passages are still counted in the effectiveness metrics. Table 5.4 shows results for this relaxed setting. It can be seen that EECATS continues to perform best on the All, out-of-KB, and Rare groups. However, the GPT methods perform much better under this relaxed setting. For nDCG@3 on the All group, GPT-4o Ano is still substantially inferior to EECATS, but the gap is smaller than before. Inspecting the outputs of the GPT methods indicates that they are good at capturing relevant information, but they are biased to long answers. Not penalizing redundancy thus helps them.

Methods	**EECATS Test Groups**			
	All	Out-of-KB	Rare	Uncommon
ConvDR	0.064	0.049	0.068	0.075
CoS + T5	0.201	0.224	0.118	0.303
CONSENT	0.276	0.264	0.203	0.407
CQR GPT	0.279	0.310	0.204	0.358
GPT-3.5	0.303	0.311	0.218	**0.426**
GPT-4o Ano	0.311	-	-	-
EECATS	**0.363**†‡	**0.358**†	**0.335**†‡	0.414

Table 5.4: nDCG@3 of methods on the groups of the EECATS test set when we relax the no-redundancy constraint, with the † and ‡ showing respective significant differences over CONSENT and GPT (3.5 and 4o Ano) (paired t-test, $p<0.05$).

RQ4: How do EECATS and the baselines perform across conversation turns? We compare the average nDCG@3 across turns from the EECATS test set in Figure 5.2. The average nDCG@3 of the EECATS method is consistently higher than that of CONSENT and GPT-4o Ano across all turns. However, the effectiveness of all methods substantially decreases from the first to the last turn, which shows the challenge of CIR as the length of a conversation increases. Even with a large language model like GPT-4o, which has a good ability to handle long context, performance gradually goes down as the number of turns increases. Relatively, however, EECATS by far performs best, with stable performance until turn 5.

5.6.2 Ablation Study

In this section, we ablate key components in EECATS in order to assess how crucial the different stages in our retrieval-and-ranking pipeline are. We also study the influence of incorporating the knowledge representation and flow representation in the entity selection.

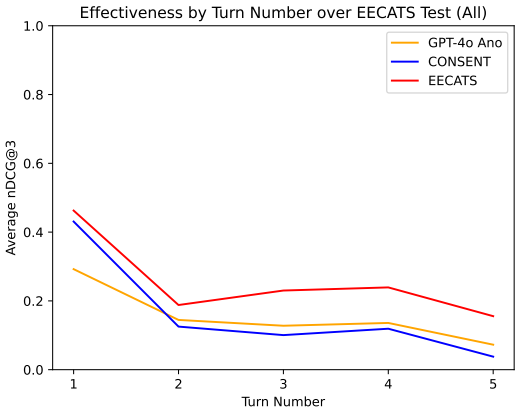

Figure 5.2: Average nDCG@3 of our EECATS and other baselines by turn number over all questions of EECATS test set.

RQ5: How important are the components in the EECATS pipeline? Recall that the EECATS pipeline consists of three steps: BM25 to find candidate documents, ColBERT (CB) to efficiently rank passages from these documents, and a cross-encoder (CE) to accurately re-rank the top-100 passages returned by ColBERT. We call this default pipeline CB & CE. We additionally consider the impact of using two simplified pipelines: using only the cross-encoder to rank all passages from the documents returned by BM25 (CE) and using only ColBERT to rank all passages from the documents returned by BM25 (CB).

Table 5.5 reports results. Comparing the default CB & CE pipeline with the CE pipeline indicates that the nDCG@3 of both pipelines is similar, but the default pipeline is 6.6 times faster and requires much less Tera FLOPS. Thus, the full pipeline substantially wins on the efficiency dimension.

Comparing the default pipeline with the CB pipeline, we observe that the nDCG@3 of the default pipeline is significantly higher and both pipelines have similar latency. These results illustrate that the 3-stage pipeline is essential for achieving both strong effectiveness and strong efficiency.

We discuss the efficiency of our method with the default setting (CB & CE pipeline) in detail on the EECATS test set. The measurements provided are averages per turn. From the top articles returned by BM25, we obtain 8,788 passages per turn, which may overlap. After ranking the passages in these documents with ColBERT, we narrow down the search space to the top $C = 100$ passages for re-ranking by the cross-encoder. In the default EECATS, ColBERT ranks 8,788 passages in 0.766 seconds, whereas it takes 8.39 seconds to rank all these passages with a cross-encoder. This speed difference is

due to the fact that ColBERT encodes each sentence in the top articles of the BM25 output once, and the maximal similarity vectors of passages are calculated efficiently with passages of consecutive sentences. The hit rate of our in-conversation cache is around 17%. Overall, the efficiency of the EECATS 3-stage pipeline comes from three factors: minimizing the number of times a cross-encoder is invoked, efficiently aggregating scores for ColBERT, and using in-conversation caching.

Pipelines	**Lat** (seconds)	**TF**	**EECATS Test Groups**			
			All	Out-of-KB	Rare	Uncommon
CB & CE	1.34	21.7	**0.277**	**0.249**	**0.249**	**0.358**
CE	8.88	46.0	0.265	0.232	0.236	0.351
CB	**1.23**	**21.1**	0.188	0.163	0.150	0.281

Table 5.5: Effectiveness (nDCG@3) and efficiency (latency and Tera FLOPS) of EECATS when using different pipelines.

RQ6: How essential are the knowledge and flow representations for entity selection? We investigate the importance of the knowledge and flow representations for detecting relevant entities. To do so, we introduce three modes in addition to the *Default* mode, which is our methodology presented in Section 5.3 that leverages both of these above components. The *No Question Flow* mode uses only the knowledge component (i.e., the final entity representation is identical to the entity knowledge representation). The *No Knowledge* mode leverages only the question flow to build entity representations. In the *No Entity Selection* mode, we drop the entire entity selection component and simply pick all mentions in the conversation history for the expanded question. In the *Zero* mode, the entity selection SBERT model is not fine tuned with our trainset.

Mode	**EECATS Test Groups**				**CAsT**	
	All	Out-of-KB	Rare	Uncommon	20	19
Default	**0.277**	**0.249**	**0.249**	**0.358**	**0.393**	**0.490**
No Question Flow	0.270	0.244	**0.249**	0.335	0.364	0.453
No Knowledge	0.232	0.181	0.200	0.346	0.316	0.425
No Entity Selection	0.152	0.140	0.079	0.284	0.168	0.194
Zero	0.204	0.176	0.134	0.349	0.271	0.369

Table 5.6: nDCG@3 of EECATS variants with different settings for knowledge and flow representation.

Table 5.6 reports nDCG@3 results using the various modes. The performance of the *No Question Flow* mode is competitive with *Default* one, which indicates that question flow is beneficial but does not play a vital role. In contrast to question flow, entity knowledge is important as demonstrated by the *No Knowledge* mode: when we remove

the knowledge component, the nDCG@3 of *Default* EECATS drops substantially. The *No Entity Selection* mode results in a very large drop compared to the *Default* mode, showing that entity selection is crucial. We can see that there is a big nDCG@3 gap between *Default* and *Zero*, proving the importance of fine tuning SBERT for entity selection. Overall, these results indicate that the knowledge representation and entity selection fine tuning are essential, and the flow representation also contributes to effectiveness.

5.7 Related Work

Contextualization. In CIR [Zamani et al., 2023, Gao et al., 2023, Thomas et al., 2021], conversational questions are processed by *rewriting* them into complete, self-contained questions [Yu et al., 2020, Vakulenko et al., 2021b, Voskarides et al., 2020, Vakulenko et al., 2021a, Elgohary et al., 2019, Kim et al., 2021, Chen et al., 2022, Farzana et al., 2023, Kostric and Balog, 2024, Jang et al., 2024, Ke et al., 2022, Mao et al., 2023], or by *expanding* questions with relevant entities or relevant question-answer pairs in the history [Krasakis et al., 2022, Lin et al., 2021, Yu et al., 2021, Christmann et al., 2023].

A typical and top-performing method for contextualization is CoS + T5 [Hai et al., 2023, Nogueira et al., 2020], which follows the question expansion paradigm, utilizing cues like the most relevant terms from the conversation history. It adapts and extends sparse retrieval methods, specifically SPLADE [Formal et al., 2021b, Formal et al., 2021a, Formal et al., 2022, Lassance and Clinchant, 2022], for conversations. CoSPLADE encodes contextualized questions and candidate answers into separate embedding vectors and takes their inner products as relevance scores. CoSPLADE automatically learns the incorporation of relevant terms from prior turns. It computes a decent first-stage ranking, but extensively relies on an LM-powered stage-two re-ranker Mono T5 [Nogueira et al., 2020], a T5 [Raffel et al., 2020] model fine-tuned on the MS MARCO.

CIR with Tail Entities. Very few prior methods are geared for coping with long-tail entities on less popular topics. They rely on sufficient training data for typical conversations, and they need to leverage background information from knowledge sources like Wikipedia or Wikidata. Neither of these prerequisites are easily satisfied when it comes to tail entities.

Methods for long-tail entities have been intensively studied for product recommenders (e.g. [Zhao et al., 2023]), which have a very different setting from CIR. Traditional document search with tail entities has been investigated in some works (e.g. [Reinanda et al., 2016, Garigliotti et al., 2019]). These could be alternatives for contextualization alone, but would require suitable background corpora, and would incur computational

costs. Recent work on product search [Peng et al., 2024] discusses specific fine-tuning of LLMs for long-tail items. This is part of a commercial search engine (which cannot be reproduced for comparisons), and pays attention to one-time queries rather than the CIR setting.

The state-of-the-art method for CIR with tail entities is CONSENT [Tran et al., 2024]. It is based on identifying relevant entity mentions in the conversation history using relatedness scores between the current question, prior turns, and prior entity names. Relatedness is computed by an available SBERT model. The actual selection of entities for question expansion is made by solving an integer linear program (ILP). This is done jointly with answer ranking, and the ILP is run for each answer candidate (i.e. the top-100 in [Tran et al., 2024]). The CONSENT method achieves good NDCG results for a difficult benchmark with tail entities, but it is very slow.

Efficiency. The most effective CIR methods have high latency because they are based on computationally heavy techniques, like large language models (GPT or CoS + T5 [Hai et al., 2023]), graph neural networks (ExplaiGNN [Christmann et al., 2023]) or integer linear programming (CONSENT [Tran et al., 2024]). In contrast, a key point of our work is to design EECATS for low resource consumption and fast response times.

5.8 Summary

This chapter presents a new approach to reconciling effectiveness and efficiency for conversational search. Prior methods emphasized answer quality at the cost of using computationally expensive techniques like large language models or integer linear programming. Our EECATS method, on the other hand, keeps all but the last stage of the retrieval pipeline as light-weight as possible, making use of acceleration techniques like the efficient bi-encoder score aggregation and the in-conversation cache. Only the cross-encoder-based ranking in the last stage requires intensive use of GPUs, but this is applied only to the top-100 candidates and thus easily affordable.

As we pay special attention to tail entities in conversations about less popular topics, EECATS introduces novel techniques for effectiveness as well. Most notably, the selection of context entities makes use of specifically fine-tuned SBERT model.

Our experiments showed substantial gains over state-of-the-art baselines on both effectiveness and efficiency, outperforming even large language models like GPT-4o.

Chapter 6

Conclusions

6.1 Take-Home Message

Language models (LMs) struggle to fully capture knowledge about entities, especially tail entities that infrequently appear in the pretraining dataset. This drawback affects the effectiveness of modern search methods relying on language models. In this thesis, we have pursued an important direction of leveraging knowledge about entities to address the drawback and enhance search quality. We encountered several challenges, including how to handle question understanding, long-tail entities, and efficiency. We design methods to fight the above three challenges, in both IR and CIR settings.

In Chapter 3, we introduce EVA [Tran and Yates, 2022] to tackle the challenge of incorporating entity knowledge into LM-based IR models for interpreting questions. In this work, we explore methods to enrich dense query and document representations with entity information from a knowledge source like Wikipedia. The proposed method identifies groups of entities in a text and encodes them into a dense vector representation, which is then used to enrich vector representations of the text from BERT [Devlin et al., 2019]. To handle documents containing many loosely-related entities, we devise a strategy for creating multiple entity representations that reflect different views of a document. In an evaluation on MS MARCO [Craswell et al., 2021a] benchmarks, we find that enriching query and document representations in this way obtains substantial improvements in search effectiveness. We publish the code of EVA on GitHub [1].

In Chapter 4, we propose CONSENT [Tran et al., 2024] as a CIR method, that deals with the challenge of incomplete and informal follow-up questions in multiple conversation turns. In particular, this work focuses on the challenge of user questions about tail entities, not available in knowledge base (KB) and sparsely covered by PLMs. We devise CONSENT for selectively contextualizing a user question with turns, KB-linkable entities,

[1] https://github.com/haidangtran1989/EVA

and mentions of tail and out-of-KB entities. CONSENT derives relatedness weights from Sentence-BERT [Reimers and Gurevych, 2019] similarities and employs an integer linear program (ILP) for judiciously selecting the best context cues for a given set of candidate answers. This method couples the contextualization and answer-ranking stages, and jointly infers the best choices for both. Experimental evaluations on two benchmarks show the effective performance of CONSENT on both long-tail and popular entities. The data and code of CONSENT are published on GitHub [2].

In Chapter 5, we build EECATS [Tran et al., 2025], another CIR method that addresses three key challenges: long-tail entities, improving efficiency over CONSENT and still competing in effectiveness. To achieve this, we define relatedness scores between the current question and previous entity mentions in the conversation, and use these to expand the current question. Answers for expanded questions are obtained through a retrieval and re-ranking pipeline. The key contributions of this design are: 1) for effectiveness, the relatedness model is judiciously fine-tuned for the conversational search setting, and 2) for efficiency, the answer retrieval stages exploit an efficient aggregation of inferred and cached token embeddings. Experiments demonstrate that our method is efficient and surpasses all GPT baselines on effectiveness over a benchmark about long-tail entities. The EECATS data and code are available on GitHub [3].

6.2 Future Directions

In this thesis, we propose approaches to the problem of incorporating entity knowledge into search systems. There are several relevant ways to further develop this area for future directions.

Time-aware information retrieval. Emerging entities are a key focus of this thesis, and time is often a notable constraint in questions about this kind of entity. For example, "What is the current performance of Marketa?" requires answers focusing on the current time, not the ones in the last year about the emerging handball star Marketa Jerabkova. Typically, this kind of question requires a proper understanding of the time constraint from the user information need (e.g. "from 2022", "in the last two years", etc). Later, the time constraint can be applied to filter out the irrelevant answer candidates.

Specific domain information retrieval. From this thesis, we can see the importance of knowledge about entities in search. In a specific expert domain such as law or finance,

[2] https://github.com/haidangtran1989/CONSENT
[3] https://github.com/haidangtran1989/EECATS

the necessity of knowledge about entities is even higher than in open domain. The expert domains require a great level of entity understanding, which cannot be guaranteed by current LLM training approaches. Therefore, these are the areas where entity knowledge plays an important role in improving search effectiveness.

Question clarification. Recall that long-tail entities are sparsely covered in the pre-training dataset of large language models and knowledge bases. This creates ambiguity for search systems to understand these entities. For example, Marketa can be easily misunderstood to a more popular entity, rather than Marketa Jerabkova. One solution to mitigate this is to allow search systems to clarify entities and search intents, such as "Do you mean Marketa Jerabkova or Marketa Irglova?". The elicited information can substantially improve search accuracy. This approach also makes these systems human-style, and more interactive. Entity knowledge can greatly contribute to the clarification process. For example, if the user responds to the above question like "I mean the handballer", then entity types support the CIR system to understand that Marketa Jerabkova is the intended entity, rather than the singer Marketa Irglova.

Incorporating entity knowledge in conversational learned sparse retrieval. To further improve efficiency in CIR, learned sparse retrieval offers a potential solution. This approach can even achieve lower latency than EECATS, since it is supported by the inverted index and does not rely on reranker such as cross-encoder. Simultaneously, the method remains effective because language models are leveraged to understand questions and infer term weights. However, LM-based sparse CIR models also face the challenge of long-tail entities. One way to overcome this is to use entity knowledge such as Wikipedia introduction text to build sparse representations. This enables learned sparse CIR methods to better understand entities and thus enhance their effectiveness. For example, from the introduction text "Marketa Jerabkova is a Czech handballer for Ikast Handbold and the Czech national team" and the current question "How many goals did the handballer score for her club", we need to identify key terms like "Marketa", "Jerabkova", "goals", "score", "Ikast", "Handbold". Although present in the introduction text, "Czech" and "national" are irrelevant and likely introduce noises in answer retrieval. Selecting the right entity (e.g. Marketa Jerabkova), and with this entity, selecting the right terms are the challenges in this problem.

Incorporating entity knowledge in CIR for lower-resource languages. In this thesis, we focus on IR methods for English. In lower-resource languages such as Vietnamese, Wikipedia does not contain as much content as its English counterpart. This results in higher chances that entities are long-tail while processing conversational

questions in lower-resource languages. Therefore, we need to rely more on inferring entity knowledge from conversation contexts, which requires training data in lower-resource languages. Training data is likely limited in such languages, making data augmentation necessary. One solution is to translate from English benchmarks, and this is a useful initial training phase. The next step can be human crowd-sourcing to provide additional training data and enhance search effectiveness.

List of Figures

List of Tables

Bibliography

[Abadi et al., 2016] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. Proceedings of the Conference on Operating Systems Design and Implementation (OSDI).

[Adlakha et al., 2021] Adlakha, V., Dhuliawala, S., Suleman, K., de Vries, H., and Reddy, S. (2021). Topiocqa: Open-domain conversational question answering with topic switching. Transactions of the Association for Computational Linguistics (TACL).

[Akkalyoncu Yilmaz et al., 2019] Akkalyoncu Yilmaz, Z., Yang, W., Zhang, H., and Lin, J. (2019). Cross-domain modeling of sentence-level evidence for document retrieval. Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).

[Aliannejadi et al., 2021] Aliannejadi, M., Kiseleva, J., Chuklin, A., Dalton, J., and Burtsev, M. (2021). Building and evaluating open-domain dialogue corpora with clarifying questions. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Arora et al., 2019] Arora, R., Tsai, C.-T., Tsereteli, K., Kambadur, P., and Yang, Y. (2019). A semi-Markov structured support vector machine model for high-precision named entity recognition. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. Proceedings of the International The Semantic Web and Asian Conference on Asian Semantic Web Conference (ISWC-ASWC).

[Balog, 2018] Balog, K. (2018). Entity-oriented search. Springer Nature.

[Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD).

[Bordes et al., 2013] Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. Proceedings of the Conference on Neural Information Processing Systems (NIPS).

[Brown et al., 2020] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. Proceedings of the Conference on Neural Information Processing Systems (NIPS).

[Cao et al., 2006] Cao, Y., Xu, J., Liu, T.-Y., Li, H., Huang, Y., and Hon, H.-W. (2006). Adapting ranking svm to document retrieval. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Ceccarelli et al., 2013] Ceccarelli, D., Lucchese, C., Orlando, S., Perego, R., and Trani, S. (2013). Dexter: An open source framework for entity linking. Proceedings of the International Workshop on Exploiting Semantic Annotations in Information Retrieval (ESAIR).

[Chatterjee and Dietz, 2022] Chatterjee, S. and Dietz, L. (2022). Bert-er: Query-specific bert entity representations for entity ranking. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Chen et al., 2022] Chen, Z., Zhao, J., Fang, A., Fetahu, B., Rokhlenko, O., and Malmasi, S. (2022). Reinforced question rewriting for conversational question answering. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Christmann et al., 2023] Christmann, P., Roy, R. S., and Weikum, G. (2023). Explainable conversational question answering over heterogeneous sources via iterative graph neural networks. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Christmann et al., 2019] Christmann, P., Saha Roy, R., Abujabal, A., Singh, J., and Weikum, G. (2019). Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion. Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM).

[Christmann et al., 2022] Christmann, P., Saha Roy, R., and Weikum, G. (2022). Conversational question answering on heterogeneous sources. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Craswell et al., 2021a] Craswell, N., Mitra, B., Campos, D., Yilmaz, E., and Lin, J. (2021a). MS MARCO: Benchmarking ranking models in the large-data regime. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Craswell et al., 2021b] Craswell, N., Mitra, B., Yilmaz, E., and Campos, D. (2021b). Overview of the trec 2020 deep learning track. Proceedings of the Text REtrieval Conference (TREC).

[Craswell et al., 2022] Craswell, N., Mitra, B., Yilmaz, E., Campos, D., and Lin, J. (2022). Overview of the trec 2021 deep learning track. Proceedings of the Text REtrieval Conference (TREC).

[Craswell et al., 2020] Craswell, N., Mitra, B., Yilmaz, E., Campos, D., and Voorhees, E. M. (2020). Overview of the trec 2019 deep learning track. Proceedings of the Text REtrieval Conference (TREC).

[Crestani et al., 1998] Crestani, F., Lalmas, M., Van Rijsbergen, C. J., and Campbell, I. (1998). "is this document relevant?... probably": A survey of probabilistic models in information retrieval. ACM Computing Survey.

[Dai and Callan, 2019] Dai, Z. and Callan, J. (2019). Deeper text understanding for IR with contextual neural language modeling. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Dalton et al., 2014] Dalton, J., Dietz, L., and Allan, J. (2014). Entity query feature expansion using knowledge base links. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Dalton et al., 2019] Dalton, J., Xiong, C., and Callan, J. (2019). CAsT 2019: The conversational assistance track overview. Proceedings of the Text REtrieval Conference (TREC).

[Dalton et al., 2020] Dalton, J., Xiong, C., and Callan, J. (2020). Cast 2020: The conversational assistance track overview. Proceedings of the Text REtrieval Conference (TREC).

[Dalton et al., 2021] Dalton, J., Xiong, C., and Callan, J. (2021). Cast 2021: The conversational assistance track overview. Proceedings of the Text REtrieval Conference (TREC).

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT).

[Dietz et al., 2018] Dietz, L., Verma, M., Radlinski, F., and Craswell, N. (2018). Trec complex answer retrieval overview. Proceedings of the Text REtrieval Conference (TREC).

[Elgohary et al., 2019] Elgohary, A., Peskov, D., and Boyd-Graber, J. (2019). Can you unpack that? learning to rewrite questions-in-context. Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).

[Farzana et al., 2023] Farzana, S., Zhou, Q., and Ristoski, P. (2023). Knowledge graph-enhanced neural query rewriting. Proceedings of the ACM Web Conference (WWW).

[Finkel et al., 2005] Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[Formal et al., 2021a] Formal, T., Lassance, C., Piwowarski, B., and Clinchant, S. (2021a). SPLADE v2: Sparse lexical and expansion model for information retrieval. arXiv:2109.10086.

[Formal et al., 2022] Formal, T., Lassance, C., Piwowarski, B., and Clinchant, S. (2022). From distillation to hard negative sampling: Making sparse neural ir models more

effective. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Formal et al., 2021b] Formal, T., Piwowarski, B., and Clinchant, S. (2021b). SPLADE: Sparse lexical and expansion model for first stage ranking. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Gao et al., 2023] Gao, J., Xiong, C., Bennett, P., and Craswell, N. (2023). Neural approaches to conversational information retrieval. Springer Cham.

[Garigliotti et al., 2019] Garigliotti, D., Albakour, D., Martinez, M., and Balog, K. (2019). Unsupervised context retrieval for long-tail entities. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Gerritse et al., 2022] Gerritse, E. J., Hasibi, F., and de Vries, A. P. (2022). Entity-aware transformers for entity search. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Hai et al., 2023] Hai, N. L., Gerald, T., Formal, T., Nie, J.-Y., Piwowarski, B., and Soulier, L. (2023). Cosplade: Contextualizing splade for conversational information retrieval. Proceedings of the European Conference on Information Retrieval (ECIR).

[Han et al., 2011] Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: a graph-based method. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Heinzerling and Inui, 2021] Heinzerling, B. and Inui, K. (2021). Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL).

[Hoffart et al., 2014] Hoffart, J., Milchevski, D., and Weikum, G. (2014). Stics: searching with strings, things, and cats. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Hoffart et al., 2011] 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. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Hofstätter et al., 2021] Hofstätter, S., Lin, S.-C., Yang, J.-H., Lin, J., and Hanbury, A. (2021). Efficiently teaching an effective dense retriever with balanced topic aware sampling. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Jang et al., 2024] Jang, Y., Lee, K.-i., Bae, H., Lee, H., and Jung, K. (2024). IterCQR: Iterative conversational query reformulation with retrieval guidance. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[Johnson et al., 2019] Johnson, J., Douze, M., and Jégou, H. (2019). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data.

[Karpukhin et al., 2020] Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and tau Yih, W. (2020). Dense passage retrieval for open-domain question answering. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Ke et al., 2022] Ke, X., Zhang, J., Lv, X., Xu, Y., Cao, S., Li, C., Chen, H., and Li, J. (2022). Knowledge-augmented self-training of a question rewriter for conversational knowledge base question answering. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Keraghel et al., 2024] Keraghel, I., Morbieu, S., and Nadif, M. (2024). Recent advances in named entity recognition: A comprehensive survey and comparative study. arxiv:2401.10825.

[Khattab and Zaharia, 2020] Khattab, O. and Zaharia, M. (2020). ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Kim et al., 2021] Kim, G., Kim, H., Park, J., and Kang, J. (2021). Learn to resolve conversational dependency: A consistency training framework for conversational question answering. ACL-IJCNLP.

[Kostric and Balog, 2024] Kostric, I. and Balog, K. (2024). A surprisingly simple yet effective multi-query rewriting method for conversational passage retrieval. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Krasakis et al., 2022] Krasakis, A. M., Yates, A., and Kanoulas, E. (2022). Zero-shot query contextualization for conversational search. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Lassance and Clinchant, 2022] Lassance, C. and Clinchant, S. (2022). An efficiency study for splade models. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Li et al., 2020] Li, C., Yates, A., MacAvaney, S., He, B., and Sun, Y. (2020). PARADE: Passage representation aggregation for document reranking. arXiv:2008.09093.

[Li et al., 2023] Li, J., Sun, A., Han, J., and Li, C. (2023). A survey on deep learning for named entity recognition. Proceedings of the IEEE International Conference on Data Engineering (ICDE).

[Lin et al., 2020] Lin, J., Nogueira, R., and Yates, A. (2020). Pretrained transformers for text ranking: BERT and beyond. arXiv:2010.06467.

[Lin et al., 2021] Lin, S., Yang, J., and Lin, J. (2021). Contextualized query embeddings for conversational search. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Liu et al., 2020] Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H., and Wang, P. (2020). K-BERT: enabling language representation with knowledge graph. Proceedings of the AAAI Conference on Artificial Intelligence (AAAI).

[Liu and Fang, 2015] Liu, X. and Fang, H. (2015). Latent entity space: a novel retrieval approach for entity-bearing queries. Information Retrieval Journal.

[Liu et al., 2018] Liu, Z., Xiong, C., Sun, M., and Liu, Z. (2018). Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[Loshchilov and Hutter, 2017] Loshchilov, I. and Hutter, F. (2017). Fixing weight decay regularization in adam. arXiv:1711.05101.

[Luan et al., 2020] Luan, Y., Eisenstein, J., Toutanova, K., and Collins, M. (2020). Sparse, dense, and attentional representations for text retrieval. Transactions of the Association for Computational Linguistics (TACL).

[MacAvaney et al., 2019] MacAvaney, S., Yates, A., Cohan, A., and Goharian, N. (2019). CEDR: Contextualized embeddings for document ranking. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Mackie et al., 2021] Mackie, I., Dalton, J., and Yates, A. (2021). How deep is your learning: the dl-hard annotated deep learning dataset. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Mao et al., 2023] Mao, K., Dou, Z., Liu, B., Qian, H., Mo, F., Wu, X., Cheng, X., and Cao, Z. (2023). Search-oriented conversational query editing. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[Mikheev et al., 1999] Mikheev, A., Moens, M., and Grover, C. (1999). Named entity recognition without gazetteers. Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL).

[Nguyen et al., 2016] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. arXiv:1611.09268v1.

[Nogueira and Cho, 2019] Nogueira, R. and Cho, K. (2019). Passage re-ranking with BERT. arXiv:1901.04085.

[Nogueira et al., 2020] Nogueira, R., Jiang, Z., Pradeep, R., and Lin, J. (2020). Document ranking with a pretrained sequence-to-sequence model. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Onoe et al., 2021] Onoe, Y., Boratko, M., and Durrett, G. (2021). Modeling fine-grained entity types with box embeddings. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[OpenAI, 2023] OpenAI (2023). Gpt-4 technical report. arXiv:2303.08774.

[Peng et al., 2024] Peng, W., Li, G., Jiang, Y., Wang, Z., Ou, D., Zeng, X., Xu, D., Xu, T., and Chen, E. (2024). Large language model based long-tail query rewriting in taobao search. Proceedings of the ACM Web Conference (WWW).

[Perozzi et al., 2014] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: online learning of social representations. Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD).

[Peters et al., 2019a] Peters, M. E., Neumann, M., IV, R. L. L., Schwartz, R., Joshi, V., Singh, S., and Smith, N. A. (2019a). Knowledge enhanced contextual word representations. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Peters et al., 2019b] Peters, M. E., Neumann, M., Logan, R., Schwartz, R., Joshi, V., Singh, S., and Smith, N. A. (2019b). Knowledge enhanced contextual word representations. Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).

[Pörner et al., 2019] Pörner, N., Waltinger, U., and Schütze, H. (2019). BERT is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised QA. arXiv:1911.03681v1.

[Qu et al., 2019] Qu, C., Yang, L., Qiu, M., Zhang, Y., Chen, C., Croft, W. B., and Iyyer, M. (2019). Attentive history selection for conversational question answering. Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM).

[Qu et al., 2021] Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, X., Dong, D., Wu, H., and Wang, H. (2021). Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[Radford et al., 2019] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

[Raffel et al., 2020] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research (JMLR).

[Reimers and Gurevych, 2019] Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Reinanda et al., 2016] Reinanda, R., Meij, E., and de Rijke, M. (2016). Document filtering for long-tail entities. Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM).

[Robertson and Zaragoza, 2009] Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends in Information Retrieval.

[Sanh et al., 2020] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2020). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. Proceedings of the Conference on Neural Information Processing Systems (NIPS).

[Santhanam et al., 2022] Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., and Zaharia, M. (2022). Colbertv2: Effective and efficient retrieval via lightweight late interaction. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[Sekine and Nobata, 2004] Sekine, S. and Nobata, C. (2004). Definition, dictionaries and tagger for extended named entity hierarchy. LREC.

[Sevgili et al., 2020] Sevgili, O., Shelmanov, A., Arkhipov, M. V., Panchenko, A., and Biemann, C. (2020). Neural entity linking: A survey of models based on deep learning. Semantic Web.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A core of semantic knowledge. Proceedings of the ACM Web Conference (WWW).

[Sun et al., 2020] Sun, Y., Wang, S., Li, Y., Feng, S., Tian, H., Wu, H., and Wang, H. (2020). ERNIE 2.0: A continual pre-training framework for language understanding. Proceedings of the AAAI Conference on Artificial Intelligence (AAAI).

[Szarvas et al., 2006] Szarvas, G., Farkas, R., and Kocsor, A. (2006). A multilingual named entity recognition system using boosting and c4.5 decision tree learning algorithms. Proceedings of the International Conference on Discovery Science (DS).

[Thomas et al., 2021] Thomas, P., Czerwinksi, M., Mcduff, D., and Craswell, N. (2021). Theories of conversation for conversational ir. ACM Transactions on Information Systems (TOIS).

[Tran and Yates, 2022] Tran, H. D. and Yates, A. (2022). Dense retrieval with entity views. Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM).

[Tran et al., 2024] Tran, H. D., Yates, A., and Weikum, G. (2024). Conversational search with tail entities. Proceedings of the European Conference on Information Retrieval (ECIR).

[Tran et al., 2025] Tran, H. D., Yates, A., and Weikum, G. (2025). Efficient and effective conversational search with tail entity selection. Proceedings of the European Conference on Information Retrieval (ECIR).

[Tredici et al., 2021] Tredici, M. D., Barlacchi, G., Shen, X., Cheng, W., and de Gispert, A. (2021). Question rewriting for open-domain conversational QA: best practices and limitations. Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM).

[Vakulenko et al., 2021a] Vakulenko, S., Longpre, S., Tu, Z., and Anantha, R. (2021a). Question rewriting for end to end conversational question answering. Proceedings of the ACM Web Search and Data Mining (WSDM).

[Vakulenko et al., 2021b] Vakulenko, S., Voskarides, N., Tu, Z., and Longpre, S. (2021b). A comparison of question rewriting methods for conversational passage retrieval. Proceedings of the European Conference on Information Retrieval (ECIR).

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. Proceedings of the Conference on Neural Information Processing Systems (NIPS).

[Voskarides et al., 2020] Voskarides, N., Li, D., Ren, P., Kanoulas, E., and de Rijke, M. (2020). Query resolution for conversational search with limited supervision. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Vrandečić and Krötzsch, 2014] Vrandečić, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. Communications of the ACM.

[Wang et al., 2019] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019). GLUE: A multi-task benchmark and analysis platform for natural language understanding. Proceedings of the International Conference on Learning Representations (ICLR).

[Wang et al., 2023] Wang, S., Sun, X., Li, X., Ouyang, R., Wu, F., Zhang, T., Li, J., and Wang, G. (2023). Gpt-ner: Named entity recognition via large language models. arXiv:2304.10428.

[Wu et al., 2020] Wu, L., Petroni, F., Josifoski, M., Riedel, S., and Zettlemoyer, L. (2020). Scalable zero-shot entity linking with dense entity retrieval. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).

[Xiong et al., 2017a] Xiong, C., Callan, J., and Liu, T.-Y. (2017a). Word-entity duet representations for document ranking. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Xiong et al., 2017b] Xiong, C., Dai, Z., Callan, J., Liu, Z., and Power, R. (2017b). End-to-end neural ad-hoc ranking with kernel pooling. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Xiong et al., 2017c] Xiong, C., Power, R., and Callan, J. (2017c). Explicit semantic ranking for academic search via knowledge graph embedding. Proceedings of the ACM Web Conference (WWW).

[Xiong et al., 2021] Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., and Overwijk, A. (2021). Approximate nearest neighbor negative contrastive learning for dense text retrieval. Proceedings of the International Conference on Learning Representations (ICLR).

[Xu et al., 2009] Xu, Y., Jones, G., and Wang, B. (2009). Query dependent pseudo-relevance feedback based on wikipedia. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Yamada et al., 2016] Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2016). Joint learning of the embedding of words and entities for named entity disambiguation. Proceedings of the SIGNLL Conference on Computational Natural Language Learning (CoNLL).

[Yamada et al., 2022] Yamada, I., Washio, K., Shindo, H., and Matsumoto, Y. (2022). Global entity disambiguation with BERT. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[Yu et al., 2020] Yu, S., Liu, J., Yang, J., Xiong, C., Bennett, P., Gao, J., and Liu, Z. (2020). Few-shot generative conversational query rewriting. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Yu et al., 2021] Yu, S., Liu, Z., Xiong, C., Feng, T., and Liu, Z. (2021). Few-shot conversational dense retrieval. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).

[Zamani et al., 2023] Zamani, H., Trippas, J. R., Dalton, J., and Radlinski, F. (2023). *Conversational Information Seeking*. Now Publishers.

[Zaratiana et al., 2024] Zaratiana, U., Tomeh, N., Holat, P., and Charnois, T. (2024). GLiNER: Generalist model for named entity recognition using bidirectional transformer. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

[Zhan et al., 2020] Zhan, J., Mao, J., Liu, Y., Zhang, M., and Ma, S. (2020). RepBERT: Contextualized text embeddings for first-stage retrieval. arXiv:2006.15498.

[Zhang et al., 2019] Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., and Liu, Q. (2019). ERNIE: enhanced language representation with informative entities. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).

[Zhao et al., 2023] Zhao, Z., Zhou, K., Wang, X., Zhao, W. X., Pan, F., Cao, Z., and Wen, J. (2023). Alleviating the long-tail problem in conversational recommender systems. Proceedings of the ACM Conference on Recommender Systems (RecSys).