

MASTER'S THESIS COMPUTING SCIENCE

The Relevance of (Relevant) Entity Presence

Reproducibility of Entity-Centric Retrieval for Document Reranking

NORBERT BOUDENS (s1059405)

July 2026

Supervisor:

prof. dr. ir. Arjen P. de Vries

Second Reader:

dr. Mart Lubbers

Radboud University



Abstract

Query-Specific Document and Entity Representations (QDER) has shown state-of-the-art effectiveness on multiple information retrieval benchmarks. This thesis attempts to reproduce this promising approach. In an initial attempt, we were unable to find results that resemble the convincing effects measured in the original work. A deep dive into the published artifacts and a comparative study reveal: (1) inconsistencies between the paper’s descriptions and the published code; (2) selective reporting of metrics ignoring unjudged documents; and, critical in combination with the previous observation, (3) a leakage of relevance assessments due to oversampling entities known to appear in relevant documents. Due to the complexities of the ranking pipeline, the final models gain knowledge about the relevance of entities, while the framework is simultaneously presented as an open-world retrieval solution. We conclude that the evaluated method cannot realistically determine entity relevance from documents and queries alone without ground-truth assistance. To investigate the requirements for improving entity-based retrieval, we implement ERM3, an extension of the standard RM3 framework that incorporates entity-aware representations. Our evaluation of ERM3 establishes a more accurate foundation for understanding the genuine impact of entity presence on retrieval effectiveness when subjected to unbiased, open-world conditions.

Acknowledgements

I want to thank my supervisor, **Arjen P. de Vries**, and colleague **Chris Kamphuis** for their time, guidance, and readiness to discuss new ideas, talk through issues, and help me process new findings and information as it came up during the study. I am also incredibly grateful for our collaboration on the resulting *SIGIR 2026* paper, which serves as a substantial learning point and personal milestone for me.

My thanks also go to my colleagues at **Spinqe** for the opportunity to discuss these ideas. At first, information retrieval was quite new to me, and I am grateful for the space to develop the skills and knowledge that were invaluable for this work.

Finally, I want to thank **Kelly Hoang** for her patience and support throughout the entire process.

CONTENTS

Contents	3
1 Introduction	5
1.1 Research Questions	5
2 Background	7
2.1 The Information Retrieval Task	7
2.2 Classical Retrieval Models	7
2.3 Neural Retrieval Models	11
2.4 Entity-Centric Retrieval	14
2.5 Multi-stage Retrieval and Neural Reranking	16
2.6 Evaluation Metrics	16
2.7 Pseudo-Relevance Feedback	19
2.8 Evaluation Environments and Reproducibility	20
3 Original Study	22
3.1 Model Architecture	23
4 Reproduction	26
4.1 Data and Evaluation	26
4.2 Differences between Paper and Code	27
4.3 Empirical Pipeline Configurations	30
4.4 Reproduction Results	31
4.5 Using the BERT-Based Entity Ranker	32
4.6 Configurations with Neural Entity Ranking	34
4.7 Entity Ranker Reproduction Results	37
5 Results Analysis	38
5.1 Document Ranking Composition	38
5.2 Using Document-Based Entity Ranking	40
5.3 Using Neural Entity Ranking	41
5.4 Case Study: “Transportation Tunnel disasters”	43
6 Comparative Analysis of DREQ and QDER	45
6.1 DREQ Run Analysis	46
6.2 Entity and Document Filtering	47
6.3 Reconstructed Ranking Results	48
6.4 Summary of DREQ Analysis	49

7	Entity-Centric Relevance Models	51
7.1	Implementation and Setup	52
7.2	Results of ERM3	56
7.3	Oracle Significance Analysis	57
7.4	Case Study: “National Museum of the American Indian”	58
8	Discussion	61
8.1	Architectural Reproducibility	61
8.2	Impact of Candidate Filtering	62
8.3	Implications for Open-World Entity Retrieval	63
8.4	Author Correspondence and Clarifications	64
8.5	Data Transparency and Reproducibility	65
9	Conclusion	66
9.1	Future Work	66
	References	67

1 Introduction

A recent paper published at the *Special Interest Group on Information Retrieval (SIGIR) 2025* conference showcased an impressive breakthrough in retrieval effectiveness achieved by using the **Query-Specific Document and Entity Representations (QDER)** [3] framework. Motivated by these results, we sought to integrate this method into real-world systems that rank documents using existing knowledge graphs. However, our initial experiments failed to yield the reported improvements. Suspecting that this performance gap stemmed from implementation differences, the focus of this thesis shifted toward a comprehensive reproducibility study.

Upon reproducing the original work by faithfully following the proposed model architecture, our analysis revealed severe inconsistencies between the paper and its official implementation. Most notably, we identified a critical **leakage of relevance assessments** caused by oversampling entities known to appear in relevant documents. This systematic flaw was traced back to the methodology of the paper’s precursor published in the *European Conference on Information Retrieval (ECIR) 2024* [4]. While the original framework was compared against open-world baselines, our findings demonstrate it actually operates as a closed-world system with access to ground-truth information. The results of this reproducibility study have since been accepted for publication at *SIGIR 2026*.

In a realistic setting, a retrieval model cannot rely on such artificial advantages; instead, it must distinguish useful entities from a massive pool of noisy, ambiguous candidates. To address this, we conducted a follow-up study to investigate whether entities can genuinely improve retrieval without shortcuts. For this purpose, we introduce the **Entity-Centric Relevance Model (ERM3)**. This model extends the pseudo-relevance feedback framework of **Relevance Model 1 (RM1)** [13] and **Relevance Model 3 (RM3)** [7] into a **shared term-entity retrieval space**.

1.1 Research Questions

The first part of our study focuses on the base effectiveness of the QDER framework. Driven by the gap in retrieval quality identified in our initial experiments, we examine whether the original results can be independently verified and how they change when aligning the codebase with the published architecture. This motivates our first research question:

- (1) **RQ1:** To what extent can the effectiveness gains of QDER be reproduced, and how do they change when aligning the implementation with its proposed model architecture?

Beyond the model architecture, the complexity of the full operational setup, especially the data preparation steps, adds another layer of difficulty to the system. To see how these specific choices shape the retrieval task, we address our second question:

- (2) **RQ2:** How do dataset preparation and evaluation design choices impact the task definition and the reported effectiveness of entity-centric retrieval?

Finally, after identifying limitations of the core framework in our reproducibility study, we extend our analysis to see if these retrieval concepts can be applied in realistic, open-world settings. To explore how entity-centric retrieval performs when exposed to a noisy and ambiguous environment, we introduce our proposed ERM3 framework:

- (3) **RQ3:** How can entity-centric retrieval models move beyond raw presence-based signals to effectively identify useful entities within unfiltered open-world environments?

2 Background

To establish the technical foundations necessary for analyzing modern entity-centric information retrieval, this chapter details the transition from *traditional lexical retrieval* to *entity-centric reranking models*, the mechanisms used for unsupervised relevance estimation, and the metrics and contexts needed to evaluate and compare these models.

2.1 The Information Retrieval Task

Information Retrieval (IR) is the process of finding and retrieving relevant information from a large collection of data resources in response to a specific user query or need. It bridges the gap between users and massive amounts of structured or unstructured data across various formats (e.g., text, images, videos, and audio), with the primary goal of ensuring users receive information most relevant to their query.

The IR process begins when a user enters a query, which serves as a statement of their information need, for example: a search string in a web search engine. Unlike traditional relational database queries, which rely on strict Boolean constraints where a record either satisfies the query or is excluded entirely, IR queries are typically unstructured and ambiguous. In practice, this means there is rarely a single correct answer: multiple objects may partially satisfy the query, each to a different degree of **relevance**, leading IR systems to adopt a graded notion of relevance rather than a binary one.

An object represents a single retrievable unit in the collection; in this thesis, objects are either *documents* or *entities*, depending on the retrieval task. Initially, all objects in the collection form the candidate set. Given a query, a ranking model assigns each object from the candidate set a numeric relevance score and orders the candidates in descending order of estimated relevance. In practical applications, only the top-ranked objects are presented to the user, and the process may be repeated if the query is refined.

2.2 Classical Retrieval Models

Ranking models, used to estimate the relevance of an object given a query, differ primarily in how they represent objects and queries and in how they compute the relevance score used for ranking. Unless stated otherwise, we focus on document-based ranking models in this section. Within classical document retrieval, ranking approaches

are commonly grouped into three broad categories: Boolean models, Vector space models, and probabilistic models.

2.2.1 Document Preprocessing. Before applying lexical (text-based) ranking models, documents are transformed from raw text into representations suitable for retrieval. In this document-centric setting, the collection of documents is commonly referred to as a corpus. The process begins with **tokenization**, in which text is split into individual tokens, typically corresponding to words. Tokens are commonly normalized by lowercasing and **stemming**, which reduces different word forms to a common root term (e.g., both “retrieving” and “retrieval” may be generalized to the term “retriev”).

This stemming is often accompanied by a process called **stop word** removal. Stop words are common, non-descriptive words such as “the”, “and”, or “is” that occur frequently across documents and queries but carry little discriminative value for retrieval.

After preprocessing, documents are commonly represented using the **bag-of-words assumption**, in which each document is modeled as an unordered collection of its terms. Word order is ignored, and depending on the retrieval model, terms are represented either as binary indicators (present or absent) or as term frequencies that capture how often each term appears in a document.

These representations form the basis for classical ranking models, and are key to understanding our reasoning when creating the shared term-entity retrieval space described in Sec. 7.1.

2.2.2 Boolean Models. Boolean retrieval models are among the earliest approaches to information retrieval and are based on a strict binary notion of relevance. Documents are represented as sets of index terms, and queries are expressed as logical formulas over these terms using operators such as AND, OR, and NOT. A document is retrieved if and only if it satisfies the query constraints *exactly*.

Because relevance is binary, Boolean retrieval produces an unranked result set, where all retrieved documents are considered equally relevant. For example, the query “automobile repair manual” can be formulated as:

“automobile” AND “repair” AND “manual”.

Using this formulation, only documents containing all three terms are retrieved, regardless of whether excluded documents might still be contextually relevant. As a result, Boolean retrieval does not support partial matching or relevance-based ranking, and

instead resembles traditional data retrieval systems such as SQL database queries, where retrieval is defined by strict constraints rather than a graded notion of relevance.

2.2.3 Vector Space Models. Originally proposed by Salton et al. [23], the vector space model extends Boolean retrieval by allowing partial matching between queries and documents.

In this model, each document and query is represented as a vector in a high-dimensional space. If a document collection contains t unique terms, then each document d can be modeled as a t -dimensional vector

$$v_d = (w_{1,d}, w_{2,d}, \dots, w_{t,d}) , \quad (1)$$

where each dimension corresponds to a distinct term in the vocabulary, assuming a vocabulary of t terms shared across all documents and queries, and $w_{i,d}$ denotes the weight of term i in document d . Similarly, a query q is then represented as

$$v_q = (w_{1,q}, w_{2,q}, \dots, w_{t,q}) . \quad (2)$$

Unlike Boolean models, which retrieve documents only when query constraints are met exactly, the vector space model assigns **graded weights** to terms, enabling documents to match a query to varying degrees. Relevance is estimated by computing the similarity between a query vector v_q and a document vector v_d , most commonly using **cosine similarity**:

$$\cos(v_q, v_d) = \frac{v_q \cdot v_d}{\|v_q\| \|v_d\|} . \quad (3)$$

I.e., the dot-product of vectors v_q and v_d divided by the product of their L_2 , or Euclidean, norms.

The vector space model itself does not define *how* term weights are computed. Weights are either binary, indicating the presence or absence of a term, or real-valued, capturing varying degrees of importance for terms within documents and queries.

A key assumption of the vector space model is that terms differ in their **discriminative power**. Continuing with the query “automobile repair manual,” the term “automobile” may occur frequently across the collection and therefore contribute little to distinguishing relevant documents from non-relevant ones, while “repair” and “manual” are more specific and provide stronger signals of relevance.

This characteristic, the **specificity** of a term, is commonly captured using its Inverse Document Frequency (IDF). Motivated by the statistical interpretation of term specificity

introduced by Sparck Jones [24], IDF measures how informative a term is by quantifying its rarity across the document collection. Formally, the IDF of a term t is defined as

$$\text{idf}(t) = \log\left(\frac{N}{\text{df}_t}\right), \quad (4)$$

where N denotes the total number of documents in the collection and df_t is the number of documents containing term t .

When IR systems shifted from handling abstracts to processing full-text documents, *Term Frequency (TF)* became another indicator of term relevance, capturing how often a term appears within a particular document to create these real-valued weights. Combined with IDF, this results in the widely used TF-IDF weighting scheme, which balances local term frequency against global rarity to ensure longer documents do not unfairly dominate retrieval.

2.2.4 Probabilistic Relevance Models and BM25. Beyond binary matching and similarity-based scoring, relevance can also be modeled probabilistically. In this setting, Robertson and Sparck Jones [21] introduced a ranking framework in which each document is assigned a probability of being relevant to a given query, and documents are ranked by ordering these probability estimates in descending order.

Models derived from this framework estimate the probability that a document d is relevant to a query q . This probability depends on the interaction between the query and document representations and assumes that, for each query, there exists a set of documents that a user would judge as relevant. The goal of the probabilistic model is therefore to produce a ranking that favors documents that are more likely to belong to this set.

The most widely used probabilistic ranking model is *Best Matching 25 (BM25)*. Originally introduced by Robertson et al. [22], BM25 estimates the relevance of a document to a query by combining TF-IDF weighting and document length normalization.

Since all lexical retrieval experiments in our study include some use of Apache Lucene, we adopt the *Lucene default* formulation of BM25 [9]. Given a query q and document d , the BM25 relevance score is computed as

$$\text{rsv}_{\text{BM25}}(q, d) = \sum_{t \in q} \log\left(1 + \frac{N - \text{df}_t + 0.5}{\text{df}_t + 0.5}\right) \cdot \frac{\text{tf}_{t,d}}{k_1 \cdot (1 - b + b \cdot (\frac{L_{\text{lossy}}}{L_{\text{avg}}})) + \text{tf}_{t,d}}, \quad (5)$$

where N is the total number of documents in the collection, df_t is the number of documents containing term t , and $tf_{t,d}$ denotes the number of times t appears in document d .

Compared to the original, and typical formulations of BM25, Lucene introduces two practical changes:

- (1) The IDF term is smoothed by adding a constant of 1 inside the logarithm. This prevents negative weights when a term appears in more than half of the corpus ($df_t > N/2$), ensuring score stability for highly frequent terms.
- (2) Document length is compressed in a lossy manner to a single-byte value (L_{dlossy}) rather than the exact term or token count of document d . Here, L_{avg} denotes the average document length in the collection, allowing Lucene to account for document length efficiently while keeping scoring fast.

The terms k_1 and b are free parameters (hyperparameters) that, unless specified otherwise, are set to the Anserini default configuration ($k_1 = 0.9$ and $b = 0.4$) throughout our study.

2.3 Neural Retrieval Models

While classical, lexical models like BM25 are highly efficient in computing a relevance score, they are fundamentally limited by their reliance on exact term or keyword-based matching.

2.3.1 The Lexical Gap. This reliance creates a **lexical gap**, which occurs when a query and document use different vocabulary to express the same meaning. For example, a formal query like “automobile repair manual” would fail to match a document that uses terms like “car” or “guide” simply because the surface words do not overlap.

This gap is driven by the inherent ambiguity of natural language, specifically:

- (1) **Synonymy:** When two terms have distinctly different surface forms, but share the same meaning (e.g., a query for “physician” failing to match a document using “doctor”).
- (2) **Polysemy:** When a single term has multiple meanings depending on the context (e.g., “Java” can refer to the programming language, the island, or a type of coffee).

Modern systems address these issues by moving from keyword-based matching toward semantic and structured representations.

2.3.2 Vector Embeddings. To resolve synonymy, modern neural models project text into **embeddings**, vectors in a dense, continuous space. Unlike classical sparse models where dimensions represent explicit vocabulary terms (sec. 2.2.3), the dimensions of a dense embedding represent latent semantic features learned from data. Because semantic similarity determines vector positioning, terms with similar meanings are mapped close together in the shared space.

2.3.3 Contextualized Embeddings & BERT. Static embedding models assign a single, fixed vector to each term, failing to resolve polysemy when meaning depends on the surrounding context. Contextualized models, such as the *Bidirectional Encoder Representations from Transformers (BERT)* [6] language model, address this limitation by operating on sequences of **tokens**.

Through a **self-attention** mechanism, BERT processes token sequences jointly rather than as independent units, in contrast to bag-of-words approaches. The representation of each token is computed by dynamically weighting its relationship to every other token in the text. This bidirectional awareness allows the model to shift the vector representation of a token based on its specific context, effectively mitigating polysemy.

For instance, while a static model maps the token “Java” to a single vector that conflates its different meanings, a contextualized model shifts the vector dynamically based on neighboring context. If surrounded by tokens like “code” or “compiler”, the self-attention weights guide the representation toward a computer science-related region of the embedding space rather than a geographic one.

However, these contextualized token representations do not inherently specify a scoring function for ranking. Instead, additional architectural mechanisms are required to translate these dynamic representations into rankable relevance scores.

2.3.4 Cross-Encoders. The **cross-encoder** architecture resolves this by processing the concatenated query and candidate document content simultaneously [18]. This unified input allows the self-attention mechanism to operate globally across the query-document boundary, enabling every query token to be compared directly to every document token.

Put simply, the relevance score is computed through a three-step process: concatenation, encoding, and scoring. First, the query tokens q and document d tokens are concatenated into a single input sequence $c = [q; d]$. This sequence is then passed

through an encoder model to generate a joint representation $h = \text{Encoder}(c)$, which a scoring function f maps to a scalar relevance score:

$$s(q, d) = f(\text{Encoder}([q; d])) . \quad (6)$$

While self-attention captures deep contextual relevance signals, it introduces a computational bottleneck. Because the query and document must be processed together, document representations cannot be pre-computed. Therefore, each unique query-document pair must be evaluated at query time, making cross-encoders computationally infeasible for large-scale retrieval.

2.3.5 Bi-Encoders. The **bi-encoder**, or dual-encoder, architecture bypasses the computational cost of cross-encoders by processing the query and the candidate document independently [11]. Because the two text sequences are processed through separate channels, query tokens cannot interact with or attend to document tokens during the encoding stage.

In contrast to cross-encoders, which project a single joint representation h into a relevance score, the bi-encoder maps both inputs into a shared vector space independently. The query q and document d are processed by separate encoder streams, to generate individual vector embeddings, which, combined with a lightweight scoring function f , can then evaluate their similarity:

$$s(q, d) = f(\text{Encoder}_q(q), \text{Encoder}_d(d)) . \quad (7)$$

This avoids the runtime cost of cross-attention, allowing the document vectors to be indexed offline (before querying), but it forces the model to compress all semantic, contextual information into those two isolated representations.

While this architecture enables efficient retrieval over large datasets, representing an entire text sequence as a single global vector reduces contextual granularity. As a result, distinctions between different words or passages are averaged together, limiting the ability of the model to capture relevance to specific topics within multifaceted (multi-topic) documents.

For instance, compressing a multi-topic document into a single fixed-size vector forces the model to average distinct semantic directions. This coarse-grained representation overgeneralizes across the entire text, producing a central, non-specific vector that is not strongly aligned with any individual concept. Consequently, when a query targets

a specific subtopic buried within the document, the fine-grained context may be diluted by the surrounding text, causing the similarity measure to fail during retrieval.

2.4 Entity-Centric Retrieval

While neural language models capture implicit semantic relationships through dense vector spaces and embeddings, they typically operate over surface text or token sequences. *Entity-Centric Retrieval* refers to a class of approaches that shift emphasis from these text-based units to explicit real-world concepts, referred to as **entities** (e.g., objects, people, organizations, or locations).

Entities do not occupy a single, fixed role in retrieval systems; in certain approaches, they are treated as the primary units of retrieval, while in others they are derived from documents or queries and used to estimate relevance. As a result, entity-centric methods are generally not intended to replace document-centric retrieval, but rather to extend it by incorporating structured entity-level information. In this study, we focus on such entity-centric approaches, specifically the QDER architecture, which forms the core of our reproducibility study (secs. 3 to 5), and its predecessor DREQ (sec. 6).

2.4.1 Knowledge Graphs. Entity-centric retrieval approaches commonly rely on a *Knowledge Graph (KG)* to represent entities and their relations in a structured form. A knowledge graph, often described as a form of semantic network, models real-world entities as nodes and the relations between them as directed edges, resulting in an explicit graph-based representation of entities.

A knowledge graph can be distinguished from a *Knowledge Base (KB)*, which is a broader information repository that may include unstructured data, such as documents or images, in addition to structured representations such as graphs. Thus, a KG is a structured, graph-based component of a KB that explicitly models entities and their relations. One of the most well-known examples is Wikidata¹, the open and collaboratively maintained knowledge graph underlying Wikipedia, which contains over 120 million items (entities).

In practice, knowledge graphs are commonly expressed using directed *subject-predicate-object* triples, where each triple represents a relation between two entities. For example, using data from Wikidata, the statement (*Java*, *part of*, *Java Platform*) expresses a “part of” relation from the entity *Java* to *Java Platform*. By traversing these relational edges,

¹<https://www.wikidata.org/wiki>

retrieval models can follow paths across multiple connected entities and incorporate relational structure into relevance estimation.

2.4.2 Entity Linking. In entity-centric retrieval, the system moves beyond terms by mapping textual mentions of entities to their corresponding unique entries in a knowledge graph. Throughout this study, we refer to these entries by their unique identifiers, or **entity IDs**, while the resulting pairings between text mentions and entity IDs are referred to as **entity links** (also called annotations).

While dense embeddings bridge the lexical gap through semantic similarity and contextualized representations, text remains inherently noisy due to ambiguous words and variations in vocabulary. Generating these entity links addresses this issue by providing the retrieval model with an explicit, canonical representation of a concept. This explicit mapping allows a model to more directly capture the intended underlying meaning independent of the specific vocabulary used in the text.

However, this approach introduces a hard dependency on the accuracy of the **entity linker**. If the linker produces an incorrect entity link, it introduces incorrect concepts into the pipeline, which can directly degrade retrieval quality, as discussed in Sec. 7.4.1.

The process executed by an entity linker is commonly divided into three sub-tasks:

- (1) **Mention detection:** Identifies and extracts entity mentions from raw text, such as recognizing “Paris” and “France” as named entities.
- (2) **Candidate selection:** Retrieves a list of potential matching entries from a knowledge base (e.g., Wikidata or DBPedia) for each detected mention.
- (3) **Entity disambiguation:** Selects the most likely entity from the generated candidate set based on the surrounding context, aiming to identify the correct entity.

Because this final disambiguation step involves uncertainty, modern linkers typically include some measure of confidence with each mapped entity. Frameworks like the *Radboud Entity Linker (REL)* [25] and the *WAT entity linker* [20] follow this paradigm, allowing retrieval models to weigh or filter links based on these confidence signals.

2.4.3 Joint Word and Entity Representations. Integrating the entity links resolved during the linking phase into a neural retrieval pipeline requires a compatible representation. Certain embedding models like Wikipedia2Vec [29] resolve this by mapping vocabulary words and entities into the same continuous vector space.

By encoding both textual context and graph structure into this shared space, words and entities yield comparably low-dimensional vectors. This uniformity allows ranking models to compute similarity scores directly across these representations using standard metrics like cosine similarity. However, these models typically inherit the high computational cost of dense neural architectures, making full-corpus evaluation impractical.

2.5 Multi-stage Retrieval and Neural Reranking

To resolve the computational limitations posed by neural approaches, modern retrieval systems typically employ a **multi-stage retrieval pipeline** to balance efficiency with ranking accuracy. Instead of evaluating complex neural or entity-centric models across millions of candidates, the pipeline uses a computationally more efficient model as a first-stage retrieval. A standard lexical model like BM25 is frequently used here to quickly narrow the full corpus down to a smaller candidate set, typically the top 1000 results for a given query. Once this manageable set of candidates is isolated, secondary reranking methods are applied to refine the document order and improve the overall ranking.

2.6 Evaluation Metrics

The retrieval architectures and pipelines outlined in the preceding sections emphasize a recurring balance between execution speed and retrieval quality. In information retrieval, this trade-off is formalized as a fundamental distinction between **efficiency** (computational overhead and execution speed) and **effectiveness** (ranking accuracy). While efficiency determines whether an architecture is viable for real-world deployments, effectiveness serves as the primary focus of retrieval model evaluation in IR research. Effectiveness is assessed by measuring the quality of the ranked list produced for a given query using standardized evaluation metrics.

In practice, the scale of modern document collections makes it infeasible for human judges to evaluate every document for every query. Instead, test collections are constructed by annotating only a select subset of documents, which leaves a portion of the documents unjudged. By default, standard evaluation tools like trec_eval² treat

²https://github.com/usnistgov/trec_eval

these unjudged documents as non-relevant. Keeping this configuration transparent and consistent across all runs is essential for valid system comparisons (tab. 6).

2.6.1 Relevance Judgments. Evaluating ranking quality requires a ground-truth baseline to determine what constitutes a correct result. In information retrieval, this baseline is established through **relevance judgments**, which are typically human-made annotations indicating whether an object or entity satisfies a given query. Standardized test collections, such as those provided by the *Text REtrieval Conference (TREC)*, bundle these judgments with a fixed set of queries and a static corpus.

These annotations are typically structured as query-document pairs, where each pair is assigned a discrete score. While binary judgments are common (relevant and non-relevant), many collections use a three-point graded relevance scale: non-relevant (0), marginally relevant (1), or highly relevant (2). These discrete scores provide the ground truth required by evaluation metrics to quantify the ability of a system to rank relevant documents over non-relevant ones.

2.6.2 Evaluation Metrics. The following metrics measure the quality of a ranked result list for a given query [17]. Let Q denote the set of queries, and let $q \in Q$ be an individual query. For query q , $D_q = (d_1, d_2, \dots)$ denotes the ranked list of retrieved documents, and R_q denotes the set of all ground-truth relevant documents for q . Assuming binary relevance, let $\text{rel}_q(d_i) \in \{0, 1\}$ indicate whether document d_i is relevant to query q . The top- K results correspond to the first K documents of D_q (i.e., $(d_1, \dots, d_K)$).

- **Precision at K ($P@K$):** measures the fraction of retrieved documents in the top- K positions that are relevant to the query q :

$$P@K = \frac{1}{K} \sum_{i=1}^K \text{rel}_q(d_i) . \quad (8)$$

$P@K$ evaluates the quality of the top- K retrieved documents while treating all positions within the cutoff equally. It does not consider the ordering of documents within the top- K , nor the total number of existing relevant documents for the query. For example, if a query has 10 relevant documents in the entire collection, and five are retrieved within the top-20 results, $P@20 = 5/20 = 0.25$. This score remains 0.25 regardless of whether those documents occupy ranks one to five or ranks 16 to 20 within the cutoff. Furthermore, because the metric does not scale by the total number

of relevant documents, a score of 0.25 fails to reflect that the system successfully retrieved half of the relevant documents within the top-20 positions.

- **Mean Average Precision (MAP):** evaluates the ranked list across all ground-truth relevant documents, accounting for both the precision and specific positions. *Average Precision (AP)* for a single query q is defined as the average of precision values calculated at each rank where a relevant document is retrieved:

$$AP(q) = \frac{1}{|R_q|} \sum_{i=1}^N P@i \cdot \text{rel}_q(d_i), \quad (9)$$

where N is the number of retrieved documents and $P@i$ is the precision at cutoff i . Unlike $P@K$, AP accounts for the total pool of relevant documents via a normalization factor ($1/|R_q|$), penalizing systems based on the number of unretrieved relevant documents.

MAP is the macro-average of AP scores across the entire query set Q :

$$MAP = \frac{1}{|Q|} \sum_{q \in Q} AP(q). \quad (10)$$

Because it accounts for both position and the total relevant pool to provide a stable average across queries of various difficulties, we primarily use MAP as the metric for system comparison in our study.

- **Mean Reciprocal Rank (MRR):** evaluates a system based entirely on the position of the first relevant document. This aligns with scenarios like question answering [26], where a user typically stops looking after they find a single correct answer. The *Reciprocal Rank (RR)* for a single query q is the inverse of the rank of the first relevant document in the ranking (rank_q):

$$RR(q) = \frac{1}{\text{rank}_q}. \quad (11)$$

Crucially, if no relevant document is found for a query, its RR is zero.

MRR is the average of these scores across the query set Q :

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} RR(q). \quad (12)$$

Unlike MAP , MRR focuses strictly on the highest-ranked relevant document and ignores any subsequent results.

- **(Normalized) Discounted Cumulative Gain ($nDCG@K$):** leverages different degrees of graded relevance assessments (sec. 2.6.1), where the value added by a relevant document diminishes the lower it appears in the ranking [8]. This decreasing reward reflects user behavior, since users in real-world scenarios are substantially less likely to look further down a list before giving up or reformulating their query. To implement this penalty, DCG applies a logarithmic discount to relevant documents at lower ranks. The standard formulation of *Discounted Cumulative Gain* (DCG) at a cutoff K scales the relevance grades linearly:

$$DCG@K = \sum_{i=1}^K \frac{\text{rel}_q(d_i)}{\log_2(i+1)}, \quad (13)$$

where $\text{rel}_q(d_i)$ here represents the discrete graded relevance of the document at rank i (rather than a binary score). However, to align with the standard implementation in `trec_eval`, which places greater emphasis on highly relevant documents, relevance grades are scaled exponentially:

$$DCG@K = \sum_{i=1}^K \frac{2^{\text{rel}_q(d_i)} - 1}{\log_2(i+1)}. \quad (14)$$

Because raw DCG scores are highly query-dependent and scale based on how many relevant documents happen to exist for a query, the value is divided by the *Ideal DCG* ($IDCG@K$). The $IDCG$ represents the maximum possible DCG for that query, calculated by sorting the ground-truth documents in perfect descending order of their graded relevance. Normalized, the $nDCG@K$ is then calculated as:

$$nDCG@K = \frac{DCG@K}{IDCG@K}. \quad (15)$$

This normalization bounds the final score in $[0,1]$, ensuring a consistent scale that can be averaged across queries with varying numbers of relevant documents.

2.7 Pseudo-Relevance Feedback

When a system lacks explicit feedback or labels indicating which documents are relevant, as is typically the case in real-world systems, *Pseudo-Relevance Feedback* (PRF) is commonly used to estimate a relevance distribution from initial retrieval results. Standard PRF approaches, such as $RM3$ [7] (eqs. (28) and (29)), operate under the assumption that the top-ranked documents from a first-stage retrieval are likely to be

relevant. Traditionally, these techniques analyze the distribution of terms within this top- k candidate subset to expand or refine the original query.

In entity-centric retrieval, this feedback mechanism can be adapted to operate over entities rather than vocabulary terms. The same core assumption applies: documents ranked highly by the initial retrieval stage are treated as weak indicators of relevance. Instead of exploiting their textual content directly for query expansion, these documents are used to estimate a distribution over entities, which can then guide subsequent retrieval or reranking stages by emphasizing entities that are more likely to be relevant to the query. This maps document-level signals to entity-level relevance estimates, providing the baseline logic used in frameworks like QDER (see sec. 3).

2.8 Evaluation Environments and Reproducibility

The reliability and generalizability of IR systems depend heavily on the environment in which they are evaluated and the transparency of the experimental setup.

2.8.1 Open-World versus Closed-World. Evaluation frameworks typically distinguish between contexts based on the noise present in the candidate set, creating an explicit distinction between an **open-world** and a **closed-world** setting. While this distinction is not always explicitly stated in related work, it is crucial for our analysis and will be discussed throughout this study.

The majority of systems operate in an open-world setting, representing a realistic retrieval scenario where a model predicts relevance from unjudged data. For example, a system might encounter a large, unfiltered set of entities associated with a query result. In this case, the model must identify *truly relevant* entities while filtering out the non-relevant ones, without making use of external relevance signals.

In contrast, a closed-world or oracle-based setting uses the ground-truth labels (sec. 2.6.1) to simplify the retrieval task. For example, in entity-centric retrieval, this approach simplifies the retrieval task by evaluating the model on a pre-filtered set where relevant and non-relevant entities are pre-filtered or isolated. While this may demonstrate the theoretical upper bound of a model, it bypasses the primary challenge of noisy data and *does not reflect real-world performance*.

2.8.2 Reproducibility Study. The term **reproducibility** is often used ambiguously across various research fields. To avoid confusion, we follow the official ACM definitions³, which divide the concept into three tiers based on who runs the experiment and what experimental artifacts are used:

- **Repeatability:** The original research team reliably repeats their own computation using the same measurement procedure and experimental setup over multiple trials.
- **Reproducibility:** An independent research team obtains the same results using the artifacts and experimental setup of the original author.
- **Replicability:** An independent research team obtains the same results using completely independent implementations and artifacts developed from scratch.

³<https://www.acm.org/publications/policies/artifact-review-badging>

3 Original Study

Query-specific Document and Entity Representations (QDER) [3] presents a neural reranking model centered on the dynamic refinement of document representations, published in *SIGIR 2025*. The authors of QDER address the fundamental challenge of modeling complex semantic relationships between queries and documents. In their work, they list three critical limitations in current state-of-the-art models:

- (1) **Vocabulary Mismatch:** Traditional methods based on lexical matching suffer from vocabulary mismatch, failing to capture deeper semantic connections where query and document terminology do not overlap.
- (2) **Static Representations:** Neural rankers rely on static, query-agnostic document representations. Such static embeddings cannot adapt to nuanced, multifaceted queries, leading to substantial under-performance.
- (3) **Similarity Metric Limitations:** Traditional similarity measures, e.g. the dot product or cosine similarity, may fail to capture the complex, semantic relationships required for accurate estimation of relevance.

The proposed QDER framework is introduced to address these limitations, using dynamic, query-specific interaction modeling with a focus on the role that entities may play to improve effectiveness. To this end, the authors introduce the following mechanisms:

Query-Centric Representations: QDER uses an attention-guided mechanism to dynamically refine document tokens and entities based on their perceived relevance to the query. Using scores from an initial retrieval phase, they refine the document representations to align with query intent, highlighting query-related elements while mitigating noise introduced by imperfect entity linking.

Late Aggregation: Building on ColBERT’s idea of late interaction [12], the authors introduce ‘late aggregation’. This approach is based on preserving and processing textual token and entity representations separately throughout the ranking process using parallel channels. These representations are only combined at the final scoring step, with the aim to prevent loss of fine-grained semantic signals.

Bilinear Interaction Modeling: To address limitations of traditional similarity metrics like cosine similarity, QDER utilizes learnable bilinear projections to combine interaction

signals more effectively, and to better capture the complex relationships between query and document representations.

Multi-Granular Interaction Modeling: The framework decomposes the interactions between query and document representations into two types of semantic relationships: *Semantic Alignment* for precise matches and *Semantic Complementarity* to capture broader contextual or complementary relevance signals.

3.1 Model Architecture

The QDER architecture defines a dual-channel, multi-vector processing pipeline, to associate linguistic signals with high-level semantic entity knowledge.

3.1.1 Dual-Channel Processing.

The text channel captures fine-grained linguistic patterns. Query q and document d are passed through a BERT-based encoder [6] to produce contextualized token-level embeddings (Q^t and D^t):

$$Q^t = \text{Encoder}(q) \in \mathbb{R}^{l_q \times d_t}, \quad D^t = \text{Encoder}(d) \in \mathbb{R}^{l_d \times d_t}, \quad (16)$$

where l_q and l_d denote the sequence lengths of the query and document texts, and d_t is the text embedding dimension.

The entity channel operates in parallel to the text channel. It utilizes a separate entity-based space. In this space, query and document entities (Q^e and D^e) are represented by pre-computed Wikipedia2Vec [29] embeddings:

$$Q^e \in \mathbb{R}^{n_q \times d_e}, \quad D^e \in \mathbb{R}^{n_d \times d_e}, \quad (17)$$

where n_q and n_d equal the number of entities linked for both the query and document text, and d_e corresponds to the dimensionality of the entity embeddings.

Document entities D^e are produced by running an entity linker over the corpus (sec. 4.1). Query entities Q^e are the top- k ($k = 20$) entities selected from $\mathcal{E}$, the pool of entities extracted from candidate documents, using a BERT-based entity ranker.

3.1.2 Dynamic Attention-Guided Interaction. Rather than computing interactions between query-agnostic, static representations, QDER implements an attention-based approach to identify semantic alignment between document tokens and entities and create query-centric representations well aligned with the query context. The *Text Channel*

Attention highlights the most important document tokens for each query token by calculating an attention matrix A^t , to produce a query-centric document representation $\tilde{D}^t$:

$$A^t = \text{softmax}(Q^t D^{t\top}) \in \mathbb{R}^{l_q \times l_d}, \quad \tilde{D}^t = A^t D^t \in \mathbb{R}^{l_q \times d}. \quad (18)$$

Similarly, the *Entity Channel Attention* emphasizes document entities with a semantic relationship to the query elements:

$$A^e = \text{softmax}(Q^e D^{e\top}) \in \mathbb{R}^{n_q \times n_d}, \quad \tilde{D}^e = A^e D^e \in \mathbb{R}^{n_q \times d}. \quad (19)$$

3.1.3 Interaction Modeling. After creating the attention-weighted, query-centric representations, QDER defines and utilizes two types of semantic relationships through element-wise operations:

Semantic Alignment: Element-wise multiplication ($\odot$) emphasizes strong semantic similarities by amplifying shared feature dimensions between the query and the query-centric document representation:

$$M^t = Q^t \odot \tilde{D}^t, \quad M^e = Q^e \odot \tilde{D}^e. \quad (20)$$

Semantic Complementarity: Element-wise addition ($+$) captures how their co-occurrence in the document provides broader contextual information that supports the query:

$$C^t = Q^t + \tilde{D}^t, \quad C^e = Q^e + \tilde{D}^e. \quad (21)$$

In both channels, these so-called “multi-granular” interactions are then aggregated via mean pooling over the query length dimension by dividing by l_q to produce fixed-dimensional representations. These pooled features are then transformed using an external retrieval score (e.g., BM25) as a scaling factor to ground semantic signals in lexical evidence. Finally, instead of summing the different interaction vectors, they are concatenated column-wise into a single unified representation h .

3.1.4 Final Scoring and Ranking. This representation (h) is passed through a bilinear interaction layer to compute a neural ranking score value (rsv_{QDER}). The authors of QDER justify the use of a bilinear projection over standard similarity metrics for its ability to capture sophisticated cross-pattern dependencies, thereby allowing the model to learn how different text or entity signals reinforce one another. In typical hybrid fashion, the neural score is then combined with the lexical ranking score value rsv_{BM25}

(see eq. (5)) via linear interpolation into the final document score:

$$\text{rsv}_{\text{hybrid}}(q,d) = \lambda \cdot \text{rsv}_{\text{BM25}}(q,d) + (1 - \lambda) \cdot \text{rsv}_{\text{QDER}}(q,d) , \quad (22)$$

where λ is learned using Coordinate Ascent optimized for Mean Average Precision (MAP).

4 Reproduction

4.1 Data and Evaluation

4.1.1 Dataset. Our reproduction of QDER [3] focuses on results for the *TREC Robust 2004* dataset [27]. This dataset contains 528,155 documents and 250 topics taken from TREC disks 4 and 5, where relevance judgments are provided for 249 of the topics. We exclusively use the title queries for our evaluation. Using *Robust04* allows for a direct comparison with the experimental framework and ablation study presented in the QDER paper.

4.1.2 Data Preparation. At the time of our reproduction, no data artifacts were available for both training or evaluating the QDER model. The authors have, however, shared guidelines and code for both data preparation and model training. We collect the necessary external data following their published instructions as follows:

- (1) **Entity-Linked Corpus:** The WAT entity-linker [20] annotates each document in the corpus with its corresponding entity mentions, resulting in 557,334 unique entity links, with an average of 29 entities per document.
- (2) **Entity Embeddings:** We source entity embeddings from the MMEAD [10] framework following the QDER methodology. This resource provides entity links and pre-trained Wikipedia2Vec [29] embeddings for the *MS MARCO* collection [1]. MMEAD yields 514,584 pre-trained entity embeddings (for 92% of the linked entities).
- (3) **Initial Ranking:** For our initial ranking, we retrieve the top- k ($k = 1,000$) documents per topic using Anserini [30] (via Pyserini [15]) and its BM25+RM3 implementation (hereafter referred to as BM25*) on a Lucene index pre-built for the *TREC Robust 2004* collection. This initial ranking is fixed for all experiments.

Resources: We share both the code⁴ and data artifacts⁵ for this reproduction study.

4.1.3 Candidate Generation. The candidate documents found in the initial retrieval are transformed into pointwise query-document data for training and evaluation of the QDER model, using the published data preparation scripts. We refer to these query-document pairs as our training and evaluation candidates. Based on ground-truth relevance, each example pair is labeled as either **positive** if the document has a query

⁴<https://github.com/informagi/trorep>

⁵<https://doi.org/10.5281/zenodo.18611335>

relevance of one or higher for that query, or as **negative** if the document has a judgment of zero or is unjudged.

4.1.4 Balancing. The authors describe a method for **balancing** the training data, to ensure an equal number of positive and negative examples in the training candidate pool. This is achieved by taking the labeled query-document pairs from the initial ranking and down-sampling the majority relevance class to create equal-sized subsets. While this process effectively follows a closed-world approach (sec. 2.8.1), where candidates are excluded based on ground-truth relevance, it is appropriate *if* it is applied strictly to the training data. Given the ground-truth relevance judgments of the *Robust04* dataset, balancing reduces the number of negative examples for each query.

4.1.5 Evaluation Paradigm. We use five-fold cross-validation as outlined in the QDER paper. That is, for each of five runs, we train a model on four folds and evaluate on the topics and candidates of the remaining fold. We report effectiveness using Mean Average Precision (*MAP*), Precision at 20 (*P@20*), Mean Reciprocal Rank (*MRR*), and Normalized Discounted Cumulative Gain at 20 (*nDCG@20*) as measures of effectiveness. Statistical significance is measured using the Student’s paired *t*-test with a threshold of $p < 0.05$ unless mentioned otherwise.

4.1.6 Hyperparameters. Following the original work, the QDER model is trained end-to-end using its PyTorch [19] implementation. This involves fine-tuning a *bert-base-uncased* encoder for 10 epochs using the Adam optimizer with a 2×10^{-5} learning rate.

4.2 Differences between Paper and Code

Our reproduction draws from both the model architecture (sec. 3.1) and the published source code. Having identified several discrepancies between these sources, we evaluate multiple model variants to account for them. We begin with a configuration based on the official implementation and transition toward the architecture specified in the paper.

4.2.1 Entity Ranking. First, the authors write that they “*train a BERT-based entity ranker [...] to map entities in $\mathcal{E}$ to queries*” [3, sec. 3.2] to transform the pre-computed entity embeddings into query-centric representations (sec. 3.1.3). However, the instructions in the official code repository deviate from this description. Instead of training a separate

entity ranker model, the implementation extracts a ranked list of entities from the initial ranking by aggregating their document scores; *i.e.*, for a query q , let D_q be the set of retrieved documents and $S(d)$ the score for document $d \in D_q$ in the initial ranking. Then, for each unique entity e found in D_q , the entity score $S_q(e)$ is computed as

$$S_q(e) = \sum_{d \in D_q: e \in d} S(d) . \quad (23)$$

These scores are subsequently min-max normalized to produce the final entity ranking for the query q .

The transfer of document scores to entities follows the assumption made by the authors that “*elements within relevant documents are themselves often relevant*” [3, sec. 1]. Conceptually, this aligns with weak supervision or relevance feedback mechanisms using Relevance Models [13]. Just as pseudo-relevance feedback methods like RM3 estimate a relevance distribution over *terms* (sec. 2.7), this approach estimates a distribution over *entities* based on their presence in the initially retrieved documents.

We hereafter refer to this approach as the *document-based entity ranker*, to distinguish it from the BERT-based *neural entity ranker* described in Sec. 4.5.

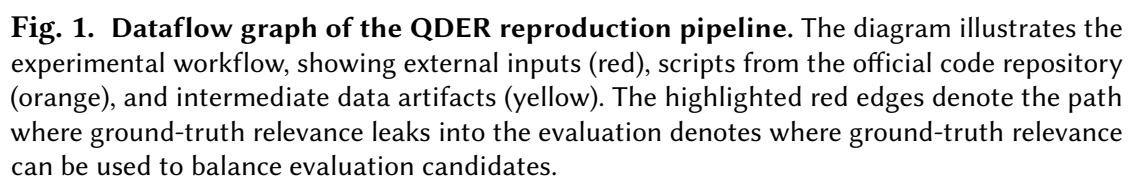
4.2.2 Embedding Interactions. Another notable difference between the description in the paper and the implementation concerns the embedding interactions in the text- and entity-channel. The model architecture defines two primary types of semantic relationships (eqs. (20) and (21)), which aim to provide a comprehensive view of query-document relevance (sec. 3.1.3). The authors contend that these interactions emphasize and reveal strong semantic similarities and overlapping themes between the query and document representations.

Additionally, the implementation includes a third interaction, not described in the model architecture (sec. 3.1.3), calculated via element-wise subtraction:

Semantic Difference:

$$S^t = Q^t - \tilde{D}^t , \quad S^e = Q^e - \tilde{D}^e . \quad (24)$$

The authors briefly mention this extra interaction in their introduction and ablation study, noting, however, that addition and multiplication have superior robustness compared to subtraction. Nevertheless, the model implementation uses *all three* types of interactions by default.



4.3 Empirical Pipeline Configurations

As a primary reference, we illustrate our reproduction pipeline using a data-flow graph (fig. 1). This visualization details the transformation from source data to the final document ranking, incorporating the five-fold cross-validation process.

Using the official data preparation and training scripts as a foundation, we first define four QDER configurations based on Equation (23).

4.3.1 Configuration 1: All-Interactions. The first configuration serves as our baseline. It implements QDER using the published instructions and scripts for data transformation, training, and evaluation. Specifically, it applies *all three* types of embedding interactions (sec. 4.2.2), including the subtraction-based operation present in the code but absent from the model architecture description (sec. 3.1.3).

4.3.2 Configuration 2: No-Subtract. The second configuration follows the paper’s model architecture strictly by excluding the subtraction-based embedding interaction (*Semantic Difference*) from both the text- and entity-channel.

4.3.3 Configuration 3 & 4: Balanced. The balanced configurations build upon their unbalanced counterparts (Configurations 1 and 2) by applying the balancing procedure from Sec. 4.1.4 to both training and evaluation candidates. This setting corresponds to following the highlighted edge in Figure 1, where the unified data preparation script processes both splits.

This process ensures an equal distribution of positive and negative candidates based on external ground-truth relevance signals (sec. 4.1.3), effectively transitioning the evaluation to a closed-world setting (sec. 2.8.1).

While balancing training data is standard practice to prevent majority-class bias, applying it to evaluation candidates alters the reranking task substantially. Notably, Configurations 3 and 4 use explicit relevance information on the test data and therefore cannot be deployed in practice. Because it is not immediately clear from the official repository instructions that this balancing step should be omitted for evaluation data, strictly following the documentation could inadvertently lead to this unintended scenario.

Table 1. Effectiveness comparison between the original study [3] and our reproduction configurations. Results on *TREC Robust 2004* (title queries) for QDER, with entity rankings generated via heuristic aggregation (eq. (23)).

Source	Version	Interactions	MAP	MRR	P@20	nDCG@20
Original Study	BM25*	n.a.	0.2915	0.6693	0.3839	0.4354
	QDER	[3, Tab. 1] <i>unknown</i>	0.6082	0.9751	0.7361	0.7694
	QDER	[3, Tab. 4] No Subtraction	0.5667	0.9714	0.7313	0.7610
	QDER	[3, Tab. 4] All Interactions	0.5091	0.9598	0.6807	0.7176
	QDER	[3, Tab. 4] All Interactions	0.5091	0.9598	0.6807	0.7176
Reproduction	BM25*	n.a.	0.2908	0.6892	0.3873	0.4442
	QDER	Config. 1 All Interactions	0.2401	0.5524	0.3086	0.3581
	QDER	Config. 2 No Subtraction	0.2837	0.6795	0.3797	0.4377
	QDER (Balanced)	Config. 3 All Interactions	0.5681	0.8089	0.6263	0.6852
	QDER (Balanced)	Config. 4 No Subtraction	0.6420	0.9401	0.7269	0.7979
	QDER (Balanced)	Config. 4 No Subtraction	0.6420	0.9401	0.7269	0.7979

4.4 Reproduction Results

Table 1 summarizes our empirical reproduction results alongside the primary [3, tab. 1] and ablation [3, tab. 4] results reported in the original study. We identify an inconsistency between these evaluations, specifically regarding the *No-Subtract* effectiveness scores. Because of this, we include both sets of results for this setup to ensure a comprehensive comparison.⁶

Comparing the reproduced results to those reported by the authors, our unbalanced configurations yield substantially lower effectiveness scores across all metrics. *E.g.*, the MAP of our *No-Subtract* configuration (0.2837) differs by a factor of at least *two* from the published results (0.5667 and 0.6082). In fact, the unbalanced configurations perform *worse* than the initial BM25* ranking, with a statistically significant degradation ($p < 0.01$) observed for both *All-Interactions* (Configuration 1, $p < 0.0001$) and *No-Subtract* (Configuration 2, $p = 0.0047$).

The impact of balancing the evaluation data is evident from the increase in scores across all metrics for both the *No-Subtract* and *All-Interactions* configurations. By filtering out the vast majority of negative candidates, using explicit relevance information, the results of the balanced configurations are inflated up to a MAP of 0.64 and nDCG@20 of 0.80, bringing our results close to those of the original publication.⁷

⁶This divergence likely stems from an undocumented LTR step used within the authors’ original pipeline (sec. 8.5).

⁷While the MAP score exceeds their reported value of 0.61 for *No-Subtract*, metrics like precision (0.73 and 0.74) align closely with the authors’ results.

Our results do, however, support the key finding reported in the original ablation study: excluding the subtraction interaction in favor of only addition and multiplication yields superior effectiveness, regardless of balancing the evaluation data. While balancing this data offers a plausible explanation for the high scores reported in the original work, it cannot be concluded with certainty that this experimental setup alone explains the performance gap between our reproduced findings and the published results.

4.5 Using the BERT-Based Entity Ranker

As mentioned in the model architecture (sec. 3.1), QDER introduces two distinct retrieval models: the *document reranker model* and a *BERT-based entity ranker*. While the former has been the primary focus so far (sec. 4.3), we continue our reproduction study by investigating the entity ranking component. As the published instructions omit specific details on training an entity ranker, we reconstruct the methodology using the published code and adhering to the architectural principles outlined by the authors:

- (1) The authors state that they “*transfer the relevance labels from documents to their entities, assuming relevant documents contain relevant entities.*” [3, sec. 3.2]. If a document is marked as relevant, the entities linked within it are also treated as relevant to the query.
- (2) The entity pool ($\mathcal{E}$) is defined as the set of all entities mentioned in the documents found by the initial retrieval (top-1000). Consequently, *all* of these entities are candidates for query representation.
- (3) The BERT-based entity ranker is trained using entity descriptions from DBpedia [14] to determine which entities are relevant for the query.

Cross-referencing this information with the data preparation code (sec. 4.1.3), we identify two scripts responsible for generating the training and evaluation data for the entity ranker⁸. Crucially, these scripts are absent from the published reproduction guidelines and their implementations deviate from the methodology outlined above, as follows:

4.5.1 Entity QREs. The first script (`make_entity_qrels.py`) creates entity-level relevance judgments, hereafter referred to as the *Entity QREs*.

⁸This was confirmed through correspondence with the authors in Sec. 8.4.

While label transfer (principle 1) would propagate the relevance judgments of a document to the entities mentioned within, the implementation enforces a more restrictive labeling logic. It assigns a positive label to an entity only if it appears *exclusively* in relevant documents; likewise, it assigns a negative label if it appears *exclusively* in non-relevant documents. A third subset, entities appearing in both relevant and non-relevant documents for a given query, is filtered out entirely. This effectively ensures that the model is trained exclusively on “distinguishing” entities, which are those that act as unambiguous indicators of relevance. Consequently, relevance information about the documents ‘leaks’ into the entity data: the presence of a positive entity in a candidate document guarantees that the document itself is relevant to the query.

4.5.2 Query-Entity Candidates. The script (`make_entity_data`) processes this query-dependent entity information with DBpedia [14] entity descriptions to produce the candidates used for both training and evaluation of the entity ranker (principle 3). It converts this entity data into a pointwise training format, where each example consists of a query, the entity identifier, a textual description sourced from DBpedia, and the corresponding relevance label.

By treating the entity ID as a document identifier and the description as document content, these entity candidates effectively share the same format and purpose as the training data prepared for the document reranker (sec. 4.1.3). Adopting these document-based naming conventions allows the scripts to transform the *entity ranking task* into a *document ranking task*, enabling direct application of the training methodology for the document reranker for the entity ranking model. Crucially, however, and contrary to principle 2, the implementation bypasses the initial ranking stage. It populates the entity pool ($\mathcal{E}$) *directly* from the Entity QREs, linking the candidates one-to-one to document-level ground truth.

4.5.3 Evaluation Biases. The logic for generating candidate entities introduces three primary biases into the evaluation setup. Critically, these biases are integrated during the creation of a global candidate pool, which is fully processed and filtered before any fold assignment has taken place.⁹ Because of this approach, the same artificial constraints applied to the training data are inherited by the evaluation data.

⁹The authors later refer to this as a “master-then-split” approach in their updated guidelines (sec. 8.4).

First, the script used to generate the pool of candidates enforces the relevance-based balancing method (sec. 4.1.4) to sample an equal number of positive and negative entities. While balancing is optional for the document reranker, the entity-level implementation applies it indiscriminately to the entire candidate set with no option for exclusion. As observed in the document-level results (tab. 1), balancing the evaluation candidates artificially inflates effectiveness scores across all metrics.

Second, because the candidate pool is restricted *solely* to entities present within the Entity QREs, the evaluation inherits the previously identified removal of shared entities, where the portion of the entity space ($\mathcal{E}$) containing the most ambiguous, non-discriminative examples is removed before the data is ever even split. Since high-frequency entities are more likely to appear in both relevant and non-relevant documents, their removal leaves an evaluation set that corresponds to an almost unambiguous signal of relevance.

Third, because the evaluation candidates are exclusively derived from the Entity QREs, the selection process incorporates entities that can only be known from ground-truth relevance judgments rather than the initial retrieval results. While the pool of candidate entities $\mathcal{E}$ should be derived from the initial BM25* ranking, the provided scripts bypass this constraint by pulling entities *directly* from document-level relevance assessments. Consequently, entities mentioned exclusively in relevant documents, which would otherwise not be found by the initial retrieval step, are still injected into the evaluation pool.

Together, these observations detail how a potentially realistic ranking scenario has been transformed into one that depends on a filtered and balanced pool of candidates, where entity selection is derived from ground-truth relevance rather than the results of the initial retrieval.

4.6 Configurations with Neural Entity Ranking

This reconstruction of the BERT-based entity ranker completes the reproduction of the dual-model QDER architecture outlined in the original paper. We introduce two additional configurations that build on the *No-Subtract* approach (Configuration 2, sec. 4.3.2) by incorporating the entity ranker into the reranking process.

To achieve this, we chain the output of the entity ranker as an input for the document reranker, replacing the heuristic aggregation model (eq. (23)) as the *Entity Ranking*

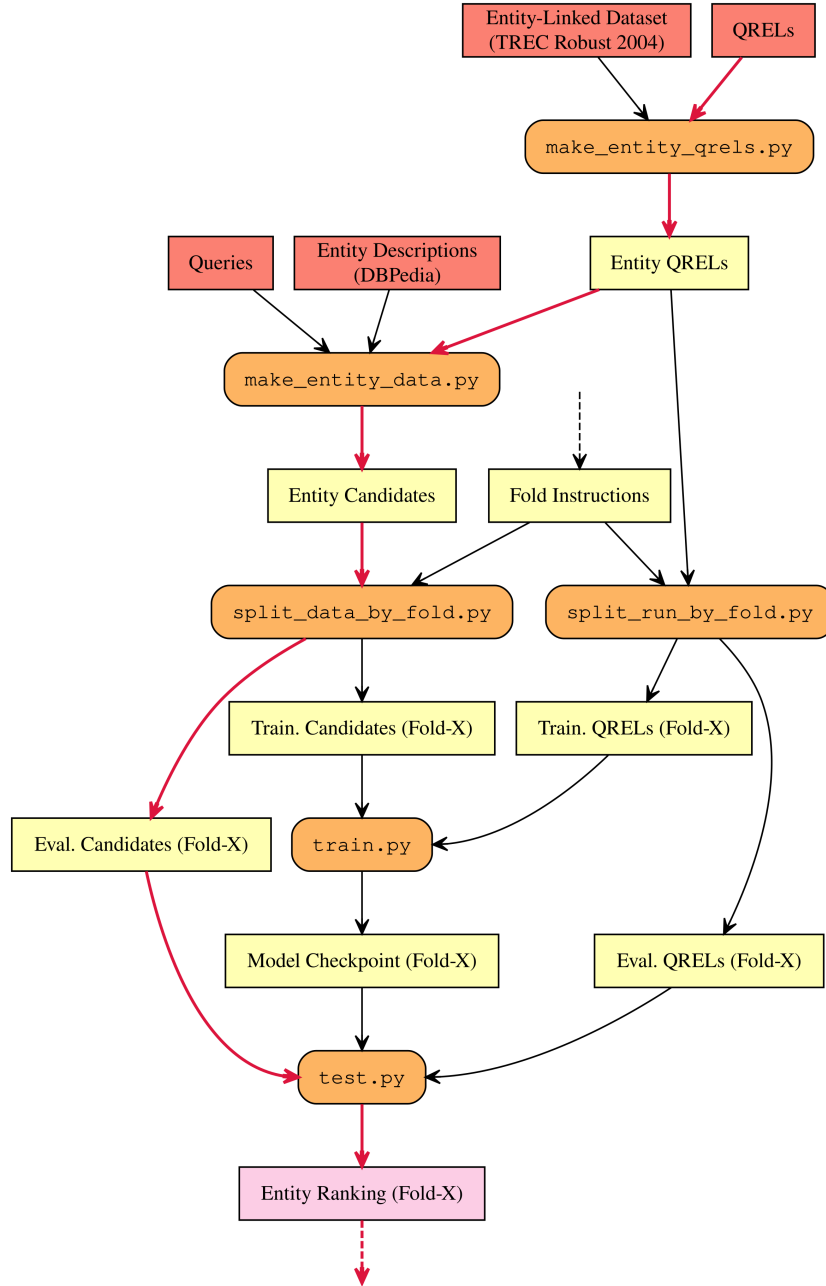


Fig. 2. Dataflow graph of the QDER entity ranker reproduction pipeline. The diagram outlines the experimental workflow, showing external inputs (red), scripts from the official code repository (orange), and intermediate data artifacts (yellow). The highlighted red edges map the path where document-level relevance information leaks into the final entity ranking used by the document reranker model.

component in the document reranker pipeline (fig. 1). Although the entity ranker follows its own distinct data preparation pipeline (fig. 2), we maintain consistency by using the same training hyperparameters and base model as the document reranker (sec. 4.1).

The raw entity scores generated by eq. (23) are unconstrained. To use these scores to refine the entity representations inside the model (sec. 3.1), we apply *min-max normalization* across all entities of a query to construct the entity weight vector w_q . Restricting these weights to a $[0,1]$ range ensures a consistent scale for the dynamic attention mechanisms (sec. 3.1.2) and prevents extreme scores from dominating the interactions. Structurally, this vector w_q serves as the scaling factor for the pooled multi-granular interaction features before final feature concatenation, which is required to prevent gradient explosion and numerical overflow in the final bilinear projection layer (sec. 3.1.4). While the original study does not specify a normalization method, our decision to use min-max normalization is informed by its use in the published implementation (sec. 4.2.1).

4.6.1 Configuration 5: Biased Entity Ranking. In this configuration, we evaluate the entity ranker model using the candidate entities as they would be generated using the published code. This *biased* configuration represents our closest attempt at a full reproduction of the dual-model approach described in the QDER paper, utilizing the provided implementation and the *No-Subtract* architecture. Thus, this configuration ignores the deviations identified within the data preparation implementation (sec. 4.5.3). Consequently, this configuration suffers from ground-truth leakage by relying on relevance information that is unavailable in real-world deployments. This likely results in an overestimation of the true effectiveness of the model (sec. 6.2).

4.6.2 Configuration 6: Unbiased Entity Ranking. In our final configuration, we modify the data generation process to better align with a more realistic retrieval scenario. We produce the entity ranking by considering the complete set of entities with mentions in the documents found in the initial retrieval, as described in sec. 3.1.1. This involves adjusting the candidate generation script to remove the ground-truth filtering process explained in Sec. 4.5.3, thereby transitioning the evaluation from an *artificial closed-world setting* back to a *realistic open-world approach*.

Table 2. Effectiveness of the neural entity ranking configurations. Comparison between the original QDER published results and our reproduction configurations based on entity rankings generated by a BERT-based entity ranker model, using title queries on TREC Robust 2004. Top half describes the results of the original paper, bottom half describes the reproduced results.

Version		MAP	MRR	P@20	nDCG@20
BM25*		0.2915	0.6693	0.3839	0.4354
QDER	[3, Tab. 1]	0.6082	0.9751	0.7361	0.7694
QDER	[3, Tab. 4]	0.5667	0.9714	0.7313	0.7610
BM25*		0.2908	0.6892	0.3873	0.4442
QDER	Config. 5	0.3726	0.8310	0.4972	0.5794
QDER	Config. 6	0.2919	0.6935	0.3829	0.4427

4.7 Entity Ranker Reproduction Results

Table 2 presents the effectiveness results for the entity ranker Configurations 5 and 6. While the implementation-based configurations align with the published code, these new results represent our reconstruction of the complete dual-model architecture.

The reproduced results fall substantially short of those published in the original paper. Whereas the authors report a MAP of 0.61, our *Biased* configuration (sec. 4.6.1), which is most closely aligned with the published implementation, reaches a MAP of 0.37. Even with the dedicated entity ranker yielding a statistically significant improvement over the baseline ($p < 0.0001$), the reproduction fails to reach the expected levels of effectiveness.

We observe a further decrease in effectiveness for the *Unbiased* configuration (sec. 4.6.2), with a MAP of 0.29, which is a statistically insignificant improvement ($p = 0.3398$) over the initial retrieval results (BM25*). The gap between the effectiveness of the *Biased* and the *Unbiased* configurations highlights the impact of pre-filtering the candidate entity pool using information leaked from the ground-truth relevance judgments.

Filtering the entity pool produces an effect comparable to the document-level balancing configurations (Configurations 3 and 4, sec. 4.3.3). Both filtering and balancing artificially inflate MRR in particular, by omitting a substantial number of negative candidates that would otherwise populate the top of the ranking. In our evaluation, this yields an MRR of 0.83 for the biased entity pool and 0.94 for the balanced document candidates, mirroring the near-perfect MRR scores reported in the original study (0.98).

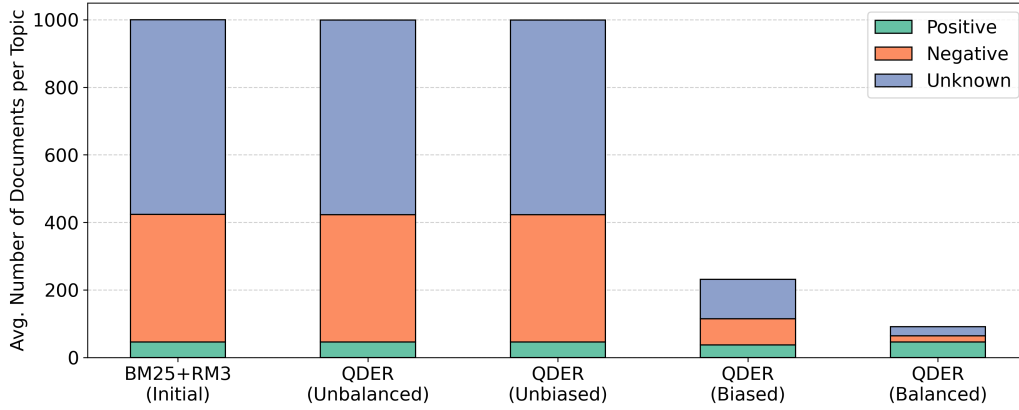


Fig. 3. Document candidate distribution across configurations. Average number of documents per topic for the BM25* baseline and each reproduction configuration of the QDER reranker on the TREC Robust 2004 dataset, categorized by relevance class (positive, negative, or unknown).

5 Results Analysis

We move to analyzing how the distinct filtering and aggregation approaches taken across the reproduction configurations dictate the final document *and* entity rankings produced by the QDER framework.

5.1 Document Ranking Composition

As described in Sec. 2.5, a standard reranking task strictly reorders a static candidate set provided by a first-stage retrieval; typically, as is the case with QDER, starting from an initial ranking produced by BM25*. However, our reproduction efforts demonstrate that the QDER pipeline can fundamentally alter this set of candidate documents through both filtering and balancing. In these cases, specific hyperparameter settings or subtle variations in the data preparation pipeline for either the document- or the entity-level ranker model can cause documents to be dropped before the final ranking.

To visualize this, we showcase the influence of each of the different configurations on the relevance-class composition of the final document ranking (fig. 3). By decomposing the per-topic average document counts into groups of positive, negative, and unknown relevance (sec. 4.1.3), we can observe how the relevance distribution shifts across the different configurations.

Table 3. Comparison between the BM25* initial ranking and QDER’s regular and balanced rerankings. Focused on the four documents considered in the balanced rankings for TREC topic #666, we report the ranks of these four documents in the results. With document ground-truth relevance shown in **bold**.

Document ID		Rank BM25*	Rank QDER (unbalanced)	Rank QDER (balanced)
LA112390-0008	(1)	20	19	2
LA120290-0084	(0)	26	15	1
FBIS3-39349	(1)	246	210	3
FT921-3507	(0)	611	596	4

5.1.1 Balancing of Candidates. The unfiltered configurations, namely *Unbalanced* (Configuration 2, sec. 4.3.2) and *Unbiased* (Configuration 6, sec. 4.6.2), retain nearly all 1,000 candidate documents per topic on average (see Figure 3). In contrast, when document-level balancing is applied prior to evaluation (*Balanced*), the final ranking shrinks to an average of only 91 documents per topic. As shown by the composition of the balanced configuration (fig. 3), this results in an equal distribution of positive and non-positive (negative and unknown) documents, confirming the effects of the balancing method (sec. 4.1.4).

Critically, filtering the evaluation candidates based on ground-truth relevance introduces data leakage. Removing the vast majority of negative candidates artificially lowers the difficulty of the reranking task, dramatically increasing the probability of relevant documents appearing at higher ranks by chance. This artificially inflates performance metrics and invalidates direct comparisons against the unfiltered baselines and configurations (tab. 1).

5.1.2 Document Filtering: “Thatcher resignation impact”. We investigate the score inflation caused by this balancing method by comparing the *No-Subtract* configuration in its unbalanced and balanced states (Configuration 2 and 4). For TREC topic #666 (“Thatcher resignation impact”), the initial BM25* run identifies *only* two relevant documents. While the unbalanced configurations reorder the full 1,000-candidate pool of documents, the balanced method restricts the evaluation to just four: the two relevant ones and two non-relevant samples.

Table 4. Entity prevalence in the document-based entity ranking. Prevalence and average rank of high-prevalence entities within the document-based entity rankings across the 250 topics in TREC Robust 2004.

Entity Title	Prevalence in Top-20	Avg. Rank
United States	1.00	2.39
United Kingdom	0.81	14.63
Europe	0.70	22.61
Russia	0.64	36.95
China	0.61	43.12

This artificial filtering of negative candidates trivializes the reranking task. The probability of ranking non-relevant documents over relevant ones is reduced considerably, as all but two negative documents are excluded. This becomes apparent when comparing the ranks of the four documents between the *initial*, *unbalanced*, and *balanced* rankings (tab. 3). Despite QDER incorrectly promoting a non-relevant document (LA120290-0084) over a relevant document (LA112390-0008), both relevant documents achieve substantially higher ranks due to the shrunken document pool. For this topic, the *No-Subtract* configurations exhibit a dramatic increase in reciprocal rank across the initial (0.050), *Unbalanced* (0.053), and *Balanced* (0.500) rankings. Specifically, the transition to a balanced candidate set artificially inflates average precision from approximately 0.02 to 0.39.

5.2 Using Document-Based Entity Ranking

Having established the effects of the balancing method used by QDER, we now evaluate the entity rankings consumed by the document-level reranker (fig. 1). As discussed previously, the published implementation derives the per-query entity rankings directly from the initial ranking, using a heuristic aggregation of initial document scores to create entity-level scores (sec. 4.2.1). This heuristic approach makes the resulting entity ranking highly susceptible to popularity bias rather than reflecting the genuine topical relevance of an entity to the query.

5.2.1 Highly Frequent Entities. This score transfer method is inherently biased toward entities with high document frequency, completely lacking a penalization mechanism for highly frequent entities, such as an inverse document frequency (IDF) weight.

Because the heuristic aggregates scores across the entire retrieved document set, the final score of an entity directly depends on its global prevalence. This characteristic is not an inherent flaw; for a given topic q , if an entity is consistently present in highly relevant documents for q , it warrants a high score for that query q .

However, since the entity-channel limits the entities that are most representative for a query to those in the top-20 highest-ranking candidates (sec. 3.1.1), this aggregation results in the selection of entities that appear consistently throughout the corpus. As shown in tab. 4, the entity “United States” appears in the twenty highest-ranking entities for all 250 topics, with an average rank of 2.39, while other entities like “United Kingdom” and “Europe” show similarly high prevalence. This suggests that the current selection of entities representative of a query is dominated by general, highly-frequent entities, instead of more desirable query-specific ones.

5.2.2 Impact on Interaction Layers. This introduces a popularity bias that injects static noise into interaction layers (sec. 3.1.2), as the same high-frequency entities are highlighted, irrespective of the topic or query. Since the top ranks are consistently occupied by these same global entities, the entity-channel effectively functions as a constant background signal rather than a dynamic feature: this prevents the attention layers from distinguishing between topics and renders the interaction mechanism redundant.

We conclude that the document-based entity ranking fundamentally fails to satisfy the theoretical desiderata of the QDER framework. By relying on unpenalized score aggregation, the implementation substitutes dynamic, query-centric entity representations with static, corpus-level popularity signals. This effectively renders the interaction layers redundant, as they are forced to highlight corpus-level noise regardless of the topic or query.

5.3 Using Neural Entity Ranking

Configurations 5 and 6 (see Sec. 4.6) comply more closely with the dual-model approach promoted in the QDER paper, where a BERT-based entity ranker provides the entities needed for the reranking model (figs. 1 and 2). Although both of these configurations employ the same neural entity-ranker model, the resulting rankings deviate due to differences in the candidate pool: a biased, filtered entity pool for Configuration 5 versus an unbiased, unfiltered one for Configuration 6.

5.3.1 Entity Pool Divergence. The QDER paper specifies the entity pool ($\mathcal{E}$) as the set of all unique entities linked within the candidate documents D_q for a topic q . In the initial retrieval (BM25*), this set would average $\approx 10,645$ unique entities per topic. However in reality, as we observed in the published implementation, this pool is derived directly from the constructed Entity QREs, which consists of 945 entities (refer to Sec. 4.5.3). Taking their intersection results in an average of *only* 396 unique candidate entities per topic.

I.e., while the QDER framework selects the top-20 entities from a total of 945 entities scored by the neural entity ranker, the majority of these are not present in the initial ranking but derived from ground-truth labels. More than 96% of the would-be entity pool, based on the initial BM25* ranking, is omitted due to the filtering and balancing processes (sec. 4.5.3), resulting in a biased candidate pool that relies predominantly on external relevance information.

To address this, our *Unbiased* configuration (Configuration 6, sec. 4.6.2) uses the entity pool $\mathcal{E}_q$ that corresponds to the description in the QDER paper, where all these entities mentioned in the top-ranked documents D_q are treated as valid candidates for ranking:

$$\mathcal{E}_q = \bigcup_{d \in D_q} \text{entities}(d), \quad (25)$$

where $\text{entities}(d)$ denotes the set of all entities linked within document d . In this unbiased ranking, the per-topic average is 10,446 unique entities.¹⁰

Table 5. Impact of entity pool filtering on ranker effectiveness. Effectiveness comparison of a QDER-based entity ranker model when evaluated on both the biased (filtered) and unbiased (complete) entity pools.

Entity Pool	MAP	MRR	P@20	nDCG@20
Filtered	0.7304	0.8059	0.7369	0.7386
Complete	0.0218	0.0985	0.0341	0.0330

5.3.2 Entity Ranker Evaluation. Using the generated entity-level relevance judgments, the *Entity QREs*, we can directly judge both the biased and unbiased entity rankings. The results, as summarized in tab. 5, show a dramatic performance decrease across all metrics, with a statistically significant decrease in MAP ($p < 0.0001$).

¹⁰A small percentage (1.9%) of the 10,645 candidate entities in the initial ranking are still excluded from further consideration, because they have no DBpedia description and the model cannot rank them.

The unbiased pool essentially re-introduces 96% of the theoretical pool that was omitted during the training phase of the model. Because the entity ranker model was supervised exclusively on “distinguishing” entities during training, but evaluated against the full, unfiltered entity set, the MAP drops from 0.73 to 0.02. This decrease in entity-level effectiveness mirrors the results observed in the document reranker between the *Biased* and *Unbiased* configurations (tab. 2). In both cases, the increase in effectiveness appears to be an artifact of reducing the pool of candidate entities.

The impact of this pool expansion is further evidenced by the shifts in entity ranks. In the *Biased* configuration, the average rank of a relevant entity is 362 per topic, with an average of 7 relevant entities positioned within the top-10. In the *Unbiased* configuration, the average rank drops to 4,831, with a negligible number (0.29) remaining in the top-10. This indicates that relevant entities are pushed out of the head of the ranking by shared entities: highly frequent, non-distinguishing entities that appear in both relevant and non-relevant documents. When these shared entities are re-introduced into the candidate pool, the model fails to effectively disambiguate them from query-relevant entities. These results indicate that the effectiveness of the proposed entity ranker is primarily derived from the pre-selection of the candidate pool, rather than a generalized ability to identify query-relevant entities.

5.4 Case Study: “Transportation Tunnel disasters”

The performance gap between the biased and unbiased entity configurations is best explained by the specific entities prioritized by the model. To illustrate the implications of *exclusively* using the distinguishing entities, we analyze TREC topic #363, “Transportation Tunnel disasters” (*Robust04*). This topic was originally used by the authors to demonstrate the interpretability of the dynamic attention mechanism (sec. 3.1.2), citing entities such as “blind curve”, “Mission Hills”, and “oxygen system” as key safety-critical or transportation-related signals [3, sec. 5.4]. While these entities may indeed provide intuitive semantic signals, the evaluation framework frequently discards them in favor of less relevant markers.

5.4.1 Entity-Level Judgments. The filtering logic used for generating the Entity QREs (sec. 4.5.3) creates a fundamental contradiction. By removing any entity that appears in both relevant and non-relevant documents, the pipeline systematically strips the dataset of core topical concepts like “Tunnel”. This process restricts the training signals

to a sparse set of highly distinguishing entities. For topic #363, the resulting labels are dominated by rare, non-topical individuals such as “Steve Propes,” “Richard Myers,” and “Tom Curry”.

These names function as anchors of relevance rather than topical signals. Instead of representing the broader topic of transportation disasters, they provide a high-precision shortcut that represents the information in the ground truth in terms of uniquely identifying entity occurrences, in essence labeling the documents by their known relevance. Consequently, the associations are grounded in the ground-truth relevance data used to filter the entity pool, allowing the model to identify specific documents via these markers rather than through a semantic link to tunnel accidents.

5.4.2 Model Ranking Analysis. When evaluated on an unbiased pool, the model maintains this preference for rare markers over actual topical concepts. Among the ten highest ranked entities, nine are specific person names with no discernible relation to the query. This indicates the system is not learning safety-critical features; it instead identifies markers of rarity, a behavior directly incentivized by the biased filtering of the training data.

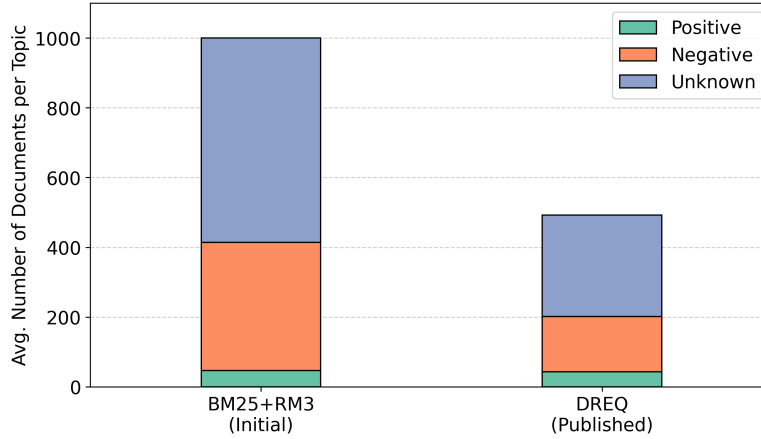


Fig. 4. DREQ document candidate distribution. Average number of documents per topic of the published rankings for the DREQ reranker on the TREC Robust 2004 dataset, split on relevance labels.

6 Comparative Analysis of DREQ and QDER

Since QDER does not include published training data or result rankings, we analyze its precursor, **Document Re-ranking Using Entity-Based Query Understanding (DREQ)** [4]. Published by the same authors as QDER in *ECIR 2024*, DREQ shares a near-identical architectural philosophy and has substantial code overlap. Specifically, its candidate generation scripts¹¹ function identically to the QDER implementation (sec. 4.5.2). In contrast to QDER, the authors provide data artifacts for DREQ, including the initial, baseline, and final rankings covering *TREC Robust 2004*. These resources allow us to directly validate the methodological flaws previously identified for the QDER framework (sec. 4.5).

Analysis of the DREQ source code confirms that Entity QREL creation exhibits the same artificial filtering of entities shared between relevant and non-relevant documents (sec. 4.5.3). Furthermore, the code also enforces mandatory balancing prior to partitioning the entity data into training and evaluation sets (cf. `make_entity_ranking_data.py`). This ensures the entity ranker is evaluated exclusively on a ground-truth-derived subset of entities, suggesting the leakage path and biases identified in QDER are not merely implementation artifacts (see Figure 2).

¹¹Namely `make_entity_qrels.py` and `make_entity_ranking_data.py`.

Table 6. Effectiveness comparison of the reported results and the published artifacts of DREQ. This analysis compares the results reported in the original work [4, tab. 1] against our independent evaluation of the published run-files using `trec_eval`.

Source	Evaluation	MAP	P@20	nDCG@20
<i>Original Paper</i>	unknown	0.57	0.73	0.75
<i>Reproduction</i>	<code>trec_eval -J</code>	0.56	0.71	0.74
	<code>trec_eval -c</code>	0.47	0.63	0.67

6.1 DREQ Run Analysis

Since DREQ provides its rankings in the form of run-files, the composition of candidate documents can be analyzed identically to Sec. 5.1. The final DREQ ranking exhibits a drastic 55.2% decrease in documents compared to the initial BM25+RM3 ranking. While the initial retrieval provides a candidate pool of 1,000 documents for each of the 250 *Robust04* topics, the final ranking produced by DREQ drops this to an average of 448 documents per topic, leaving 112,112 documents total.

Categorizing these documents by their ground-truth relevance judgments (fig. 4) reveals that the proportion of non-relevant documents decreases by 57.0%, that of unknown relevance by 50.3%, while relevant documents decrease by only 8.7%. Although presented as an entity-oriented neural reranking model, this disproportionate pruning indicates the pipeline functions primarily as a filter. By leveraging external relevance signals to remove noise, it artificially simplifies the reranking task.

6.1.1 Missing Topics. Analysis of the published ranking data reveals the exclusion of TREC topics #634 (“L-tryptophan deaths”) and #672 (“NRA membership profile”) for the data released. *Robust04* provides relevance assessments for 249 of its 250 topics in the ground truth; #672 is the only topic without ground-truth judgments. However, topic #634 is judged in the ground-truth; its omission from the final ranking can only be explained as a direct artifact of the data preparation pipeline (which we confirm in Sec. 6.3.1).

6.1.2 Evaluation Inconsistencies. Evaluating the distributed run-files with `trec_eval` using the default `-c` (complete) flag produces results that do not align with the original publication. This flag averages metrics over all 249 judged topics and assigns a score of

zero to any missing topics. However, using the `-J` flag, which computes metrics exclusively on judged documents and excludes those with unknown relevance altogether, yields scores that correspond to those reported in tab. 6.

This indicates the results reported by the authors were calculated by treating unjudged documents as non-existent rather than non-relevant, a setting the `trec_eval` manual advises against. Perhaps more concerning is that metrics reported for the baselines against which the model has been compared were *not* produced using the `-J` flag. Such an experimental setup results in an unfair comparison between the scores for DREQ and the baselines (see [4, tab. 1]), as the latter are penalized for unjudged documents, whereas the DREQ results ignore them.

6.2 Entity and Document Filtering

We explain how the data preparation of DREQ results in this filtered ranking by manually applying the discussed filtering logic to the initial ranking.

6.2.1 Entity QREs Construction. As detailed in the QDER framework (sec. 4.5.1), the Entity QREs are constructed by transferring document labels to entities. This step identifies entities found *exclusively* in positive documents and those *exclusively* in negative documents. The subset of entities present in both of these categories is filtered out and discarded. Applying this process to the entity-linked dataset published for DREQ reveals that an average of 3,482 (6.9%) unique entities per topic are filtered out due to this rule. This leaves an average of 895 (1.8%) positive entities and 45,930 (91.3%) negative entities per topic.

6.2.2 Entity Candidate Construction. The subsequent stage of the pipeline follows the same candidate generation logic described in Sec. 4.5.2 through the implementation in `make_entity_ranking_data.py`. As is the case with the QDER pipeline, 1 : 1 balancing is enforced for both training and evaluation data. Given that 1.8% of entities are labeled as positive, this mandatory balancing discards the vast majority of the entity pool ($\mathcal{E}$). Across topics, this process retains an average of 895 positive entities while sampling an equal number of the 45,930 available negatives. Therefore, over 98% of negative entities are filtered out before training and evaluation takes place.

Because the entity ranker is only ever evaluated on these filtered candidates, the final entity ranking does not provide scores for any of the discarded entities. For example, if a

topic in the initial ranking contains ten entities exclusively found in relevant documents, the entity ranker will produce a ranking of *at most* twenty entities, regardless of the number of entities present within the initial retrieval.

6.2.3 Document Candidates. When generating the document-level candidate data, the DREQ pipeline requires each document to contain at least one entity from the previously generated entity ranking, as implemented in `make_doc_ranking_data.py`. Any document from the initial retrieval that does not share an entity with this ranking is discarded.

Due to the restricted entity candidate pool from the previous step, which already excludes shared entities and 98% of negative entities, this requirement acts as a third filtering stage: documents that only contain discarded entities are removed from the pipeline before the document-level reranking takes place.

We reproduce this filtering logic on the published data. By iterating through the initial BM25+RM3 ranking, removing all documents that lack the entities present in the generated entity ranking, we create a filtered version of this ranking. This is done for both the Entity QREs we generated, and the ones published by the authors of DREQ.

Table 7. Overlap analysis between published DREQ and reconstructed, filtered rankings. Comparison of the final DREQ ranking against our filtered BM25+RM3 baseline to verify pruning logic. Top half of the table describes results presented in the original work, bottom half are results created for this experiment.

Ranking	Entity QREs	#Topics	#Docs	Overlap
BM25+RM3	-	250	250,000	100%
DREQ	Published	248	122,112	100%
Filtered	Published	248	122,112	100%
Filtered	Ours	249	135,905	63.6%

6.3 Reconstructed Ranking Results

To verify that document pruning is a direct result of the entity-filtering logic, we compare the reconstructed, filtered rankings against the DREQ ranking published by the authors (sec. 6.2).

We evaluate the reconstructed rankings by measuring the per-topic overlap as:

$$\text{Overlap}(D_q) = \frac{|D_{\text{filt},q} \cap D_{\text{pub},q}|}{|D_{\text{pub},q}|}, \quad (26)$$

where $D_{\text{filt},q}$ and $D_{\text{pub},q}$ represent the set of documents for topic q in the reconstructed and published rankings, respectively.

The results are summarized in Table 7. Using the published Entity QREs, our filtered ranking achieves a full overlap of 100%. This confirms that document pruning in the final DREQ ranking is a deterministic consequence of the entity-filtering logic, rather than a learned behavior of the model. However, generating the Entity QREs using the published code and dataset results in a lower overlap of 63.6% compared to the published artifacts, suggesting that the entity-sampling process itself is not fully deterministic.

6.3.1 Explaining the Missing Topics. While the Robust04 dataset contains 250 topics (249 with relevance judgments), the final DREQ ranking addresses only 248. As discussed, missing evaluation data for topic #672 is to be expected, as this topic lacks document judgments in the *Robust04* test collection, therefore no entity-level judgments will be created. The reconstructed results also explain the absence of topic #634 from the DREQ data, despite the presence of valid document judgments for this topic in the test collection.

Analysis of the published Entity QREs reveals that for topic #634 there are 35,693 negative entities, and *zero* labeled as positive. Due to the mandatory balancing procedure applied to the entity candidates, in both the QDER and DREQ pipeline, the absence of positive entities means no negative candidates are sampled. This results in no entity-level candidates, no document-level candidates, and subsequently no final ranking for topic #634, which in turn explains the absence of data regarding the topic in the published data artifacts.

6.4 Summary of DREQ Analysis

The analysis of the DREQ data preparation pipeline serves to highlight and validate the concerns identified during our reproduction of QDER. The 100% overlap in Table 7 confirms that document filtering is a deterministic result of the data preparation process, rather than a learned capability of the DREQ model.

By using external relevance signals to limit the entity candidate pool prior to ranking (sec. 6.2), the pipeline discards a disproportionate 55.2% of non-relevant documents compared to only 8.7% of relevant ones. This creates a class distribution (fig. 4) that mirrors the biased entity pool identified in our QDER experiments (sec. 5.1), suggesting a deficiency shared between the data preparation pipelines of both models.

Finally, the analysis showcases the high sensitivity of this process to the initial state of the data. While our reconstruction identified positive entities for topic #634, the published data did not, leading to the entire topic being discarded in the final ranking. In addition to this, the reported effectiveness of DREQ was inflated artificially by evaluating only on judged documents.

7 Entity-Centric Relevance Models

The entity-centric models analyzed in the previous chapters, QDER and DREQ, derive their reported effectiveness from a presence-based logic, where the relevance of an entity is tied directly to document-level ground-truth labels during the construction of the Entity QREs (sec. 4.5.1). While this demonstrates that entity signals can act as distinguishing indicators of relevance, the candidate selection in these pipelines has been found to rely on external signals that are strictly unavailable in practical retrieval scenarios.

Although both models show what is possible in a closed-world, oracle-based setting (sec. 2.8.1), they leave a critical and arguably more interesting question regarding the generalizability of entity signals unanswered:

Can entity signals be used to improve document ranking effectiveness in an open-world setting, where no prior knowledge of document relevance is available?

In a realistic, open-world setting, a model cannot rely on pre-filtered candidate pools or ground-truth labels; it must identify and distinguish useful, relevant entities from a massive pool of noisy candidates. Leveraging these signals is inherently difficult due to the natural noise and ambiguity of surface-level entities.

To investigate the behavior of entities within an unfiltered, open-world retrieval pipeline, we introduce the **Entity-Centric Relevance Model (ERM3)**. This model extends the Pseudo-Relevance Feedback (PRF) framework of RM1 [13], and its query-interpolation variant RM3 [7], to a **shared term-entity retrieval space**.

The primary goal of this chapter is to identify which aspects of an entity signal contribute to entity-based ranking effectiveness when using a standard initial retrieval. Our approach focuses on:

- (1) **Shared Retrieval Space:** Evaluating the impact of an index where both lexical terms and entities occupy a common retrieval space.
- (2) **Oracle Analysis:** Estimating the upper bound of effectiveness when using presence-based signals with known document relevance (i.e., an oracle-based approach mimicking the QDER and DREQ pipelines).
- (3) **Signal Types:** Evaluating the behavior of different entity characteristics, namely specificity (IDF), popularity, and semantic proximity, when used as expansion features.

7.1 Implementation and Setup

To create an experimental environment where both the lexical terms of a document and the entities linked within share the same retrieval space, we treat entities as terms.

7.1.1 Entities as Terms. To prevent the underlying retrieval engine from applying tokenization or stop-word filtering to our entity terms, fragmenting or filtering out the entity representations, we map each entity to a unique hash string via the MD5 message-digest algorithm. This transformation ensures that each entity is indexed and retrieved as a single, preserved token or term. The hash for a given entity is generated using its unique Wikipedia2Vec identifier [29] according to the following transformation:

$$\text{hash}_e = \text{md5}('e' + \text{id}_e), \quad (27)$$

where id_e is the numeric Wikipedia page ID associated with the entity in the Wikipedia2Vec mappings. The prepended character ‘e’ serves to ensure a consistent string format for the hash function and provides a distinct prefix for these synthetic entity terms, or entity hashes.

To mitigate semantic noise and avoid topic drift, we apply a confidence threshold to the entity-document mappings acquired via the WAT entity linker [20]. As described in the documentation of WAT, the entity linker assigns each annotation an attribute $\rho \in [0,1]$ that estimates its confidence in the annotation given the input text.

Neither the QDER nor the DREQ framework defines or implements a pruning threshold; however, the WAT service documentation states that ρ should be used to discard low-confidence annotations, suggesting a reasonable threshold typically falls in the range of $[0.1, 0.3]$. We select the midpoint of this interval, applying a threshold of $\rho \geq 0.2$, effectively filtering out annotations that are unlikely to be relevant to the content of the document.

7.1.2 Retrieval Pipeline. For the initial retrieval and index management, we use the Anserini toolkit [30], interfaced via Pyserini [15]. This allows us to perform an initial BM25 retrieval on the *Robust04* index to identify a candidate set of feedback documents.

However, the RM3 implementation within Anserini is incompatible with our entity hashes, as automatic term filters enforce a term length of at least two and at most twenty characters, excluding our entity hashes from the relevance model used for

query expansion. Consequently, Anserini’s native BM25+RM3 pipeline cannot use these hashes because they are filtered out before reaching the expansion stage.

To resolve this limitation while preserving the mechanics of the underlying framework, we reimplement the RM3 query expansion model of Anserini directly in Python.¹² To distinguish this Python-based implementation from the standard Anserini pipeline used in our previous experiments, we denote it as RM3[†]. This implementation allows us to use the set of initially retrieved documents and compute the expansion weights while treating our MD5 entity hashes as valid terms. Once the expanded query is constructed, we again use Anserini to execute a second retrieval. While we retain the standard filters Anserini implements for lexical terms, we explicitly exempt entity hashes from the length-based pruning.

To formalize this process, let C be the set of k documents ($k = 1000$) retrieved by the initial BM25 stage. We select the highest-ranked feedback documents $\mathcal{R} \subset C$ where $|\mathcal{R}|$ is limited by the feedback document parameter `fb_docs` (default `fb_docs = 10`).

Following the RM3 logic as implemented in Anserini, we estimate the relevance model $P(w \mid Q_{\text{RM1}})$ as a weighted combination of document vectors given query Q :

$$P(w \mid Q_{\text{RM1}}) \propto \sum_{d \in \mathcal{R}} \text{rsv}_{\text{BM25}}(Q, d) \cdot P(w \mid d), \quad (28)$$

where $\text{rsv}_{\text{BM25}}(Q, d)$ represents the initial retrieval score for document d . In this implementation, term pruning occurs at two stages: first, individual document vectors are pruned to the top M feedback terms (default `fb_terms = 10`) before aggregation; second, the resulting relevance model is pruned again to the top M terms.

The final expanded query is constructed via linear interpolation between the original query maximum likelihood model $P(w \mid Q)$ and the relevance model:

$$P(w \mid Q_{\text{RM3}}) = \alpha \cdot P(w \mid Q) + (1 - \alpha) \cdot P(w \mid Q_{\text{RM1}}), \quad (29)$$

where α is the interpolation weight (default $\alpha = 0.5$) that determines the contribution of the original query to the final expanded representation.

When this pipeline is applied to our joint index, allowing both lexical terms and entity hashes to be selected for the expansion model, we denote this mixed configuration as RM3[†]_E.

¹²The Python RM3 implementation (RM3[†]) was verified against the Anserini baseline exclusively using lexical terms, showing no significant difference ($p = 0.6779$ for MAP).

7.1.3 Feature Redundancy. The $\text{RM3}_E^\dagger$ configuration introduces a risk of feature redundancy. This occurs when an entity representing a concept is already present as a lexical term in the model, resulting in a dual representation of the same concept. Because the retrieval pipeline treats the terms and entity hashes as independent tokens, this risks over-emphasizing specific concepts while limiting the discovery of additional expansion terms.

7.1.4 Entity-Based Relevance Model. To isolate the influence of specific entity characteristics and avoid the dual representation of terms and entities, we define an expansion model restricted to our entity hashes. In a standard RM3 scenario, lexical terms often dominate the expansion pool due to their higher corpus frequency. By limiting the relevance model to entity hashes, we prevent the entities from being drowned out, allowing the model to directly exploit characteristics such as specificity and semantic relevance.

We define the entity relevance model ERM1, derived from Anserini RM1 (eq. (28)), where the probability of an entity hash e is determined by:

$$P(e \mid Q_{\text{ERM1}}) \propto \sum_{d \in \mathcal{R}} \text{rsv}_{\text{BM25}}(Q, d) \cdot P(e \mid d) \cdot W(Q, e). \quad (30)$$

Integrating the ERM1 model into the standard RM3 framework yields our extended model, ERM3. We define this as a linear interpolation between the original query distribution and the entity relevance model:

$$P(\tau \mid Q_{\text{ERM3}}) = \alpha \cdot P(\tau \mid Q) + (1 - \alpha) \cdot P(\tau \mid Q_{\text{ERM1}}), \quad (31)$$

where τ denotes a generic token from the shared retrieval space $\mathcal{V} \cup \mathcal{E}$, comprising the lexical vocabulary $\mathcal{V}$ and the set of entity hashes $\mathcal{E}$.

We configure the weighting function $W(Q, e)$ to exploit three distinct entity properties:

ERM3 (IDF): To measure the impact of entity specificity, we reward rare entities and penalize common, often context-general, entities (e.g., “United States”). We use a smoothed version of the Inverse Document Frequency (see eq. (4)):

$$W_{\text{IDF}}(Q, e) = \log\left(\frac{N}{n_e + 1}\right) + 1, \quad (32)$$

where N is the total number of documents in the collection and n_e is the document frequency of entity e .

ERM3 (IPV): To evaluate popularity as a weighting signal, we use Inverse PageViews (IPV) as a second signal for specificity. While the IDF measures entity rarity within the collection, pageviews reflect the global popularity of an entity based on its Wikipedia traffic. We define the weighting function as:

$$W_{IPV}(Q, e) = \log\left(\frac{V}{v_e + 1}\right) + 1, \quad (33)$$

where V represents the total Wikipedia Pageviews during the 2024 calendar year, and v_e represents the total views for entity e during the same period. We use the Wikimedia pageview dump filtered for user traffic [28], excluding automated, non-human activity to more accurately represent genuine interest. Since the source data is partitioned by month, we aggregate the view counts for the full year to ensure that the weights reflect long-term importance rather than short-term trends. This approach assumes that entities with lower global interest are more specific and less prone to introducing topic drift.

ERM3 (SEM): To weight entities by their semantic proximity to the query, we calculate the cosine similarity between the query terms and entity hashes using the vector space provided by MMEAD [10]. Since this space contains both entity and word-level Wikipedia2Vec embeddings, we can directly compare query terms with candidate entities:

$$W_{SEM}(Q, e) = \prod_{q_i \in Q} \cos(q_i, e), \quad (34)$$

where $q_i \in Q$ represents the individual query terms. We use a product-based approach to ensure that an entity is only assigned a high weight if it exhibits strong semantic similarity across all terms in the query. By calculating the cosine similarity to each query term individually, we leverage the joint embedding space to identify entities that are contextually relevant to the entire query, thereby reducing the influence of entities that *only* relate to a single query term. For instance, in the query “Turkey Iraq water”, this method targets entities related to the shared regional context (e.g., “Euphrates”), rather than each country individually.

7.1.5 Oracle Configuration. To determine the theoretical upper bound of effectiveness for the entity-based feedback approach, we introduce an *Oracle* variant of ERM3. In contrast to the previously defined PRF framework, which relies on the set of top documents ($\mathcal{R}$), the oracle only considers documents with positive relevance within

that set for query expansion. We define the oracle feedback set $\mathcal{R}^*$ for a given query Q as the subset of candidates in $\mathcal{R}$ that are labeled as relevant in the ground-truth judgments:

$$\mathcal{R}^* = \{d \in \mathcal{R} \mid \text{rel}(Q, d) > 0\}, \quad (35)$$

where $\text{rel}(Q, d)$ denotes the relevance score for a given query-document pair in the ground-truth labels.

The expansion logic for this variant follows eq. (30), with the document summation restricted to this positive-relevance subset ($\mathcal{R}^*$). This setup serves as an evaluation of the presence-based logic used to construct the Entity QREs in QDER and DREQ (sec. 4.5.1), allowing us to evaluate whether document-level relevance alone is sufficient to identify useful, discriminative entities, or if weighting signals like W_{IDF} remain necessary even when document relevance is known.

Because this oracle operates exclusively on the output of a realistic first-stage retrieval, its theoretical ceiling is restricted. It functions as a simulation of a perfect user identifying relevant documents from an initial search space, rather than an absolute evaluation across the entire collection.

Resources: We share both the code¹³ and data artifacts¹⁴ for this specific chapter.

7.2 Results of ERM3

Table 8 presents the effectiveness of the entity-based pseudo-relevance feedback models across the 250 Robust 2004 topics. Integrating entities in the form of entity hashes directly into the RM3 pipeline (RM3_E^+) yields MAP scores nearly identical to the lexical baseline (0.2940 vs. 0.2941), though with a modest decrease in MRR (0.6931 vs. 0.6831). This indicates that, if we allow entities to be selected by the relevance model, they do not substantially enhance the overall ranking quality in a standard setting. On average, only 1.06 entity hashes are selected per query, with a maximum of 5, compared to an average of 9.8 lexical terms.

When restricting the model to entity-based expansion, all ERM3 configurations underperform the lexical baseline, with the best configuration (ERM3 + IDF) achieving a MAP of 0.2627. In contrast, the oracle configuration, which restricts feedback documents to ground-truth relevant documents, achieves a MAP of 0.3099. When combined with

¹³<https://github.com/noboud/entity-based-retrieval>

¹⁴<https://doi.org/10.5281/zenodo.21196846>

Table 8. Effectiveness comparison across our different entity-based PRF configurations (ERM3). Results use an initial BM25 retrieval stage and a minimum entity-linker confidence of $\rho \geq 0.2$ to evaluate the impact of different entity weighting strategies and oracle signals.

Category	Version	Weight	MAP	MRR	P@20	nDCG@20
<i>Baseline</i>	RM3 [†] _E		0.2941	0.6831	0.3876	0.4451
	RM3 [†]		0.2940	0.6931	0.3855	0.4463
	BM25		0.2547	0.6804	0.3620	0.4244
<i>Entity-Based</i>	ERM3	IDF	0.2627	0.6326	0.3586	0.4109
	ERM3	IPV	0.2600	0.6309	0.3510	0.4052
	ERM3	SEM	0.2593	0.6450	0.3566	0.4093
	ERM3		0.2587	0.6420	0.3542	0.4086
<i>Oracle</i>	ERM3 (Oracle)	IDF	0.3135	0.8982	0.4159	0.5190
	ERM3 (Oracle)		0.3099	0.8972	0.4139	0.5151

IDF weighting, this increases to 0.3135. This improvement suggests that even when document relevance is known, the discriminative power of IDF remains important for prioritizing the most useful entities for a query.

The effectiveness of W_{SEM} and W_{IPV} indicates that these specific signals lack the discriminative power required for query expansion. While W_{IPV} successfully models global entity popularity, it lacks the collection-specific alignment provided by IDF. Similarly, the product-based W_{SEM} creates a precise but highly restrictive weight, often penalizing relevant entities that only partially match the query terms.

The underperformance of ERM3 compared to the lexical baselines suggests that raw entity presence alone is insufficient. This highlights the underlying difficulty of an open-world approach. Similar to *Unbiased* in our QDER reproduction (Configuration 6, sec. 4.6.2), exposing the model to the full, unfiltered noise of the corpus severely caps maximum effectiveness. Without additional signals, common entities dominate the expansion model, diluting the focus and context of the query, similar to the heuristic-based entity ranking of QDER (sec. 5.2).

7.3 Oracle Significance Analysis

To determine if the performance gain from adding an IDF signal to the Oracle configuration is a genuine improvement, we conduct a statistical evaluation between

Table 9. Entity weight analysis for “National Museum of the American Indian” (TREC topic #664). Weights represent the percentage of the expanded query model, illustrating the impact of IDF and Oracle signals on entity selection across different ERM3 configurations.

Entity / Terms	ERM3 (%)	ERM3+IDF (%)	Oracle (%)	Oracle+IDF (%)
<i>Original Query Terms</i>	50.00	50.00	50.00	50.00
Smithsonian Institution	9.96	12.28	12.55	15.33
Native Americans in the United States	13.33	12.06	4.51	3.51
George Gustav Heye Center	3.74	6.37	5.11	8.63
New York City	5.25	3.72	7.17	5.04
National Museum of the American Indian	0.00	3.70	0.00	0.00
India	2.65	0.00	2.04	0.00
<i>Other Entities</i>	15.07	11.87	18.62	17.49

the two Oracle rankings across all topics. Specifically, we evaluate the shifts in MAP effectiveness between ERM3 (Oracle) and ERM3 (Oracle) with IDF weighting.

A Student’s paired t-test reveals a statistically significant improvement for the IDF-weighted Oracle ($p = 0.0131$), which is further supported by a Wilcoxon signed-rank test ($p = 0.0049$). Given that both tests result in $p < 0.05$, we conclude that the W_{IDF} signal provides a reliable boost to retrieval effectiveness, even with perfect knowledge of document-level relevance.

7.4 Case Study: “National Museum of the American Indian”

Topic #664 (“National Museum of the American Indian”) represents the largest decrease in effectiveness observed in our entity-based expansion experiments. Transitioning from the strong lexical $RM3^\dagger$ baseline (0.7981 MAP) to $RM3^\dagger_E$ reduces the effectiveness to 0.6385. Table 9 summarizes the entity weights assigned by the different ERM3 configurations discussed in our case study.

7.4.1 Entity Selection and Linker Noise. Entity linking is the first stage in our pipeline where noisy links can introduce semantic drift. Without confidence-based pruning ($\rho = 0.0$), the WAT linker frequently maps the surface term “Indian” to the country “India”. The term “Indian” is highly polysemous, as it can refer to either the indigenous peoples of the Americas or to people from India. Without correct disambiguation, this leads to substantial semantic drift. Co-occurrence analysis shows that 31.9% of documents containing the entity “Native Americans in the United States” also link to

“India”, revealing a systematic failure in disambiguating the term “Indian”. Implementing the $\rho \geq 0.2$ threshold reduces this overlap to 15.1%, effectively pruning noisy, unrelated annotations before they reach the expansion model.

7.4.2 Mixed Relevance Model. Moving from the lexical $\text{RM3}^\dagger$ baseline to the mixed relevance model ($\text{RM3}_E^\dagger$), which incorporates both lexical terms and entity hashes, yields a MAP decrease of 0.1595. For this topic, $\text{RM3}_E^\dagger$ selects only three entities:

- (1) **“Native Americans in the United States”**: Represents the primary demographic and cultural focus of the museum.
- (2) **“Smithsonian Institution”**: The parent organization that manages the facilities and collections of the museum.
- (3) **“New York City”**: The location of the George Gustav Heye Center, home to the original collection of the museum until 2004.

While these entities are contextually accurate and topically related to the query, their inclusion in the expansion model results in a substantial drop in retrieval effectiveness. The selected entities represent broad geographical and cultural features, possibly not specific enough to discriminate between documents relevant or irrelevant to the topic.

7.4.3 IDF Weighting. To analyze the influence of entity-specific signals without lexical interference, we transition to the entity-centric relevance model. Applying IDF weighting (W_{IDF}) to this configuration yields a 0.1404 MAP improvement over the unweighted entity baseline (ERM3).

An analysis of the relevance model used for query expansion reveals clear shifts in entity weights for topic #664 (tab. 9). The weighting function highlights rare, specific entities such as “Smithsonian Institution” and the “George Gustav Heye Center” (6.37%). Simultaneously, the model penalizes high-frequency entities such as “Native Americans in the United States” (12.06%) and removes the noisy link “India” entirely. These results indicate that specificity is a necessary signal to prevent context-general entities from dominating the expansion model, excessively broadening the context of the query.

7.4.4 Distinguishing Relevance from Usefulness. To evaluate entity-based expansion when document relevance is known, we restrict the feedback document set $\mathcal{R}$ to ground-truth relevant documents ($\mathcal{R}^*$). This oracle configuration yields a MAP of 0.8321 for the topic. Incorporating the IDF weighting function (Oracle + IDF) further increases effectiveness to 0.8491 MAP.

The weight shifts observed in tab. 9 confirm that even with perfect document-level relevance knowledge (Oracle), the model must still distinguish between relevance and usefulness (sec. 7.3). For example, the weight of “Native Americans in the United States” drops from 13.33% in the ERM3 baseline to 4.51% in the Oracle setup. With the addition of the IDF weighting (W_{IDF}), this entity is penalized further to a weight of 3.51%. While the entity is semantically relevant to the query, it is too broad to distinguish relevant and non-relevant documents.

In contrast, the model consistently prioritizes rare, specific entities across all configurations. The “Smithsonian Institution” sees an increase through every stage, reaching up to a weight of 15.33%, while the “George Gustav Heye Center” weight more than doubles from 3.74% to 8.63%. In this case, the W_{IDF} signal acts as a necessary addition to focus the expansion model on the specific context of the museum.

8 Discussion

Our reproduction study was motivated by the state-of-the-art retrieval effectiveness reported for the QDER framework [3]. However, our analyses reveal that these gains depend on fundamental closed-world constraints that were undocumented in the original work or its predecessor DREQ [4]. By evaluating the framework in a realistic setting, we identify the factors that limit its utility for open-world retrieval. The following sections address our research questions (sec. 1) and incorporate clarifications from our correspondence with the authors of QDER.

8.1 Architectural Reproducibility

Our research directly addresses the first research question:

RQ1: *To what extent can the effectiveness gains of QDER be reproduced, and how do they change when aligning the implementation with its proposed model architecture?*

by demonstrating that we were unable to mirror the reported effectiveness gains of QDER when strictly aligning the implementation with the proposed model architecture. An audit of the provided codebase revealed substantial discrepancies between the documented neural pipeline and the actual implementation, leading us to evaluate six distinct configurations.

Yet, the reported effectiveness remains unattainable across all of our configurations. The first of these configurations, using the implementation in its provided state (Configuration 1, sec. 4.3.1), fails to reproduce the original gains. Even after aligning the implementation with the described model architecture (Configuration 6, sec. 4.6.2), the model fails to improve on its baseline.

We identify three systemic factors that drive this failure:

- (1) **Popularity Bias:** The document-based entity ranking logic is highly susceptible to popularity bias. Without the undocumented candidate filtering used in the original work, the model prioritizes entities based on their frequency throughout the corpus rather than their genuine topical relevance (sec. 5.2).
- (2) **Noise Sensitivity:** The framework lacks the robustness required for realistic candidate selection. Even with a BERT-based entity ranker, the system is highly sensitive to ambiguous entities. Transitioning to an open-world setting causes effectiveness to

drop from 0.37 to 0.29 MAP. In this environment, the framework fails to significantly outperform the standard lexical BM25+RM3 baseline (tabs. 2 and 5).

- (3) **Shortcut Learning:** The proposed embedding interactions (sec. 3.1.3) exploit training artifacts rather than facilitating semantic representations. The training pipeline removes core topical concepts, forcing the model to target markers of rarity (sec. 5.4). Consequently, the system prioritizes specific names like “*Steve Propes*” as high-precision shortcuts to documents, bypassing the semantic reasoning described in the original work [3, sec. 5.4].

Despite these failures, our study confirmed that omitting the subtraction interaction consistently improves effectiveness (tab. 1), as reported by the authors in their ablation study [3, sec. 5.2]. This suggests that while individual interaction components are functional, their potential benefits are currently overwhelmed by the noise inherent to open-world retrieval.

8.2 Impact of Candidate Filtering

The failure of the reranker in real-world settings is obscured by methodological choices that fundamentally redefine the retrieval task through candidate filtering. To address our second research question, we analyze the ground-truth-informed strategies:

RQ2: *How do dataset preparation and evaluation design choices impact the task definition and the reported effectiveness of entity-centric retrieval?*

Our analyses demonstrate that these strategies convert open-world retrieval into a closed-world evaluation problem. By artificially simplifying the evaluation environment, they create gains in effectiveness that are unattainable in practice.

At the document level, balancing removes a disproportionate number of non-relevant candidates, shrinking the pool by over 90% (sec. 5.1.1). This discriminative pruning substantially alters the distribution, drastically increasing the likelihood of ranking a relevant document over a non-relevant one purely by random selection. As a result, even when the model incorrectly reranks a non-relevant document over a positive one, it still achieves a higher overall rank than in the initial retrieval (tab. 3). While the original work described balancing as an optional preprocessing step used for training the second-stage document reranker (fig. 1), our reproduction reveals it is a mandatory constraint for the first-stage entity ranker (sec. 4.5.3 and fig. 2).

The *Biased* configuration (Configuration 5, sec. 4.6.1) demonstrates how these constraints transform open-world retrieval into a simplified classification problem. This step eliminates over 96% of the valid candidate entities identified during initial retrieval (sec. 5.3.1). The core of this bias lies in the isolation of distinguishing entities that act as unambiguous indicators of relevance. In a realistic setting, most entities are non-distinguishing and appear across both relevant and non-relevant documents, creating the noise required for a robust evaluation.

By applying these preprocessing steps, the framework preemptively clears this noise before ranking begins. This process allows the model to bypass the core challenges of disambiguation and representation. Instead, the mere presence of a known-relevant entity within a document acts as a direct indicator for query relevance. For the QDER framework, the resulting effectiveness scores reflect an artificial ceiling that is fundamentally unattainable in real-world retrieval systems where ground truth is not available at inference time.

We identified the predecessor model, DREQ, as following a similar pattern. Applying the biased filtering logic produces a document ranking fully overlapping with the results published by the authors, as shown in tab. 7. Furthermore, the document composition produced by our *Biased* approach (fig. 3) closely mimics the distribution found in the final ranking of DREQ (fig. 4), indicating that this filtering is a deliberate design choice rather than an accidental implementation artifact.

8.3 Implications for Open-World Entity Retrieval

The transition from an artificially constrained environment back to an unfiltered one (*Biased* vs. *Unbiased*) exposes the fundamental limitations of presence-based signals. Addressing our third research question:

RQ3: *How can entity-centric retrieval models move beyond raw presence-based signals to effectively identify useful entities within unfiltered open-world environments?*

we introduce the **Entity-Centric Relevance Model (ERM3)** to evaluate the presence-based logic used in QDER to create the Entity QREs described in Sec. 4.5.1. Our results indicate that defining entity relevance solely through document-level ground truth fails in open-world settings. These signals are overwhelmed by noise and fail to distinguish between semantic relevance and actual ranking utility (sec. 7.4.4).

Even in the *ERM3 (Oracle)* configuration (sec. 7.1.5), where the model uses perfect knowledge of document relevance, simply knowing an entity is present in a relevant document is insufficient to ensure a better ranking. While the Oracle provides an improvement over lexical methods, incorporating an IDF weighting signal to penalize common entities results in a statistically significant boost to retrieval effectiveness.

Topic #664 (“National Museum of the American Indian”, sec. 7.4) illustrates why this penalty is necessary. Broad entities like “Native Americans in the United States” are highly relevant to the query context but provide no discriminative power for ranking. Without an IDF signal to penalize these high-frequency terms, the expansion model is dominated by common entities that dilute the focus of the topic.

Consequently, the significant improvement observed when adding IDF weighting confirms that entity specificity is a prerequisite for effective open-world retrieval (sec. 7.3). In realistic settings, identifying relevant entities through presence logic is only one component of the task. Effectiveness depends on prioritizing rare, specific entities that provide the discriminative signals necessary to separate relevant documents from surrounding noise.

8.4 Author Correspondence and Clarifications

The inability to reproduce the reported effectiveness led us to share our findings directly with the authors of QDER. This dialogue resulted in the authors making substantial revisions to their published code and documentation, which now explicitly describe the dual-model approach used in our *Biased* and *Unbiased* configurations. Additionally, these updates formally document the filtering mechanisms we identified, verifying them as deliberate design features rather than implementation artifacts.

To reconcile the remaining ambiguities regarding the closed-world assumptions, we engaged in follow-up correspondence with the authors of QDER where they explicitly confirmed to us that the discriminative filtering and balancing of candidates are *core components* of the model design. During this technical exchange, they clarified that the primary question driving their framework is fundamentally whether entity signals *can* improve document retrieval effectiveness, rather than addressing open-world retrieval specifically. This direct clarification of the filtering mechanisms, also present in their revised documentation, verifies that the framework operates as a closed-world, oracle-based pipeline, where the ground truth is used to filter out noisy candidates.

Importantly, this closed-world assumption is not made explicit in the original presentation of results [3, 4, tab. 1], where QDER and DREQ are reported alongside open-world baselines such as BERT [6], EQFE [5], and CEDR [16]. In this context, readers may incorrectly infer that QDER is evaluated under the same open-world conditions. In practice, however, open-world models must rank over noisy, unfiltered document collections, whereas QDER is evaluated only after non-relevant candidates have been pruned using external signals of relevance.

Following our initial correspondence, the original authors published a subsequent study that, for the first time, reports results for QDER in an unfiltered, open-world setting [2]. When evaluated under these realistic conditions, the framework achieves a MAP of 0.294 [2, tab. 1], which closely aligns with the degraded MAP of 0.292 observed in our *Unbiased* configuration in Table 2.

8.5 Data Transparency and Reproducibility

Furthermore, the gap in MAP score between the reported overall results and the ablation study identified in Sec. 4.4 was explained by the inclusion of an additional **Learning-to-Rank (LTR)** stage. During our direct communications, the authors clarified that this third-stage LTR step involves rescoring the entire set of candidate documents found by the initial retrieval. While the documentation and code were updated following our inquiry, the accompanying data release for QDER remains partial; intermediate artifacts from the BERT-based entity ranker, the Entity QREs, and the final unfiltered rankings produced by this third stage are unavailable. The absence of these artifacts hinders verification of whether candidate filtering persists through the LTR stage, thereby further complicating comparisons to open-world systems.

The final rankings published for DREQ [4], the predecessor of QDER, do not utilize an LTR stage and remain filtered (sec. 6.3). Furthermore, an attempt to reconstruct the DREQ entity pools using the published scripts and datasets yielded only a 63.6% overlap with the published artifacts (tab. 7). This divergence suggests that the entity sampling and selection process is either non-deterministic or relies on undocumented parameters. Because the generated entity pool materially differs from the official version, the availability of intermediate artifacts is essential for ensuring full reproducibility.

9 Conclusion

Our reproduction study investigated the state-of-the-art effectiveness reported for the QDER framework. We were unable to reproduce the findings presented in the original work, identifying that the reported effectiveness gains fail to materialize under strict architectural alignment using the available instructions, code, and data. Although subsequent updates and correspondence with the authors clarified that the discriminative filtering of candidates was an intentional design choice, our analysis shows that this reliance on document-level ground truth fundamentally transforms their evaluation into a closed-world, oracle-based scenario.

To investigate how entity signals can actually be used in a realistic setting, we introduced the Entity-Centric Relevance Model (ERM3). Our evaluation demonstrates that raw presence-based signals are insufficient in unfiltered environments, showcasing that mere entity presence does not equate to ranking usefulness. True open-world effectiveness not only requires an entity to be relevant, but also relies on additional signals such as entity specificity and discriminative power.

9.1 Future Work

Our findings demonstrate a broader challenge for future research: the increasing complexity of modern retrieval systems can easily obscure or hinder their empirical analysis. A specific design choice, whether accidental or deliberate, can lead to interpretations of effectiveness that do not hold in realistic, open-world settings.

Moving forward, future research in entity-centric retrieval must prioritize evaluation in unfiltered environments to measure the true discriminative power of entity signals without ground-truth assistance. Additionally, future work should explore methods to approximate the effects of candidate filtering through entirely unsupervised means, such as employing paradigms like LLM-as-a-Judge [31] to replicate the discriminative filtering observed in QDER in open-world settings.

Finally, a strong attention to detail is crucial, and a release of not only the codebase but also the intermediate data artifacts is vital to provide a foundation for future study. This level of transparency is essential to ensure that evaluation is used to measure the inherent capabilities of a model rather than the artifacts of a supervised data pipeline.

References

- [1] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A Human Generated MACHine Reading COMprehension Dataset. arXiv:1611.09268 [cs.CL] <https://arxiv.org/abs/1611.09268>
- [2] Shubham Chatterjee. 2026. Entities as Retrieval Signals: A Systematic Study of Coverage, Supervision, and Evaluation in Entity-Oriented Ranking. arXiv:2604.05204 [cs.IR] <https://arxiv.org/abs/2604.05204>
- [3] Shubham Chatterjee and Jeff Dalton. 2025. QDER: Query-Specific Document and Entity Representations for Multi-Vector Document Re-Ranking. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Padua, Italy) (SIGIR '25). Association for Computing Machinery, New York, NY, USA, 2255–2265. doi:10.1145/3726302.3730065
- [4] Shubham Chatterjee, Iain Mackie, and Jeff Dalton. 2024. DREQ: Document Re-ranking Using Entity-Based Query Understanding. In *Advances in Information Retrieval: 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24–28, 2024, Proceedings, Part I* (Glasgow, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 210–229. doi:10.1007/978-3-031-56027-9_13
- [5] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval* (Gold Coast, Queensland, Australia) (SIGIR '14). Association for Computing Machinery, New York, NY, USA, 365–374. doi:10.1145/2600428.2609628
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423
- [7] Nasreen Jaleel, James Allan, W. Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and hard.
- [8] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (Oct. 2002), 422–446. doi:10.1145/582415.582418
- [9] Chris Kamphuis, Arjen P. de Vries, Leonid Boytsov, and Jimmy Lin. 2020. Which BM25 Do You Mean? A Large-Scale Reproducibility Study of Scoring Variants. In *Advances in Information Retrieval*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer International Publishing, Cham, 28–34.
- [10] Chris Kamphuis, Aileen Lin, Siwen Yang, Jimmy Lin, Arjen P. de Vries, and Faegheh Hasibi. 2023. MMEAD: MS MARCO Entity Annotations and Disambiguations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '23). ACM, New York, NY, USA, 2817–2825. doi:10.1145/3539618.3591887

- [11] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 6769–6781. doi:10.18653/v1/2020.emnlp-main.550
- [12] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) (SIGIR '20). Association for Computing Machinery, New York, NY, USA, 39–48. doi:10.1145/3397271.3401075
- [13] Victor Lavrenko and W. Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New Orleans, Louisiana, USA) (SIGIR '01). Association for Computing Machinery, New York, NY, USA, 120–127. doi:10.1145/383952.383972
- [14] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and Christian Bizer. 2014. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal* 6 (01 2014). doi:10.3233/SW-140134
- [15] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*. Association for Computing Machinery, New York, NY, USA, 2356–2362. doi:10.1145/3404835.3463238
- [16] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized Embeddings for Document Ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) (SIGIR'19). Association for Computing Machinery, New York, NY, USA, 1101–1104. doi:10.1145/3331184.3331317
- [17] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- [18] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *CoRR* abs/1901.04085 (2019). arXiv:1901.04085 <http://arxiv.org/abs/1901.04085>
- [19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA.
- [20] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *Proceedings of the First International Workshop on Entity Recognition & Disambiguation* (Gold Coast, Queensland, Australia) (ERD '14). Association for Computing Machinery, New York, NY, USA, 55–62. doi:10.1145/2633211.2634350

- [21] S. E. Robertson and K. Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27, 3 (1976), 129–146. doi:10.1002/asi.4630270302
- [22] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST, 109–126.
- [23] G. Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (Nov. 1975), 613–620. doi:10.1145/361219.361220
- [24] Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*. Taylor Graham Publishing, GBR, 132–142.
- [25] Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. 2020. REL: An Entity Linker Standing on the Shoulders of Giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (Virtual Event, China) (SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 2197–2200. doi:10.1145/3397271.3401416
- [26] Ellen Voorhees. 2000. The TREC-8 Question Answering Track Report. https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=151495
- [27] Ellen Voorhees. 2005. Overview of the TREC 2004 Robust Retrieval Track. doi:10.6028/NIST.SP.500-261
- [28] Wikitech. 2024. Data Platform/Data Lake/Traffic/Pageviews — Wikitech,. https://wikitech.wikimedia.org/w/index.php?title=Data_Platform/Data_Lake/Traffic/Pageviews&oldid=2202540 [Online; accessed 15-March-2026].
- [29] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 23–30.
- [30] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible Ranking Baselines Using Lucene. *J. Data and Information Quality* 10, 4, Article 16 (Oct. 2018), 20 pages. doi:10.1145/3239571
- [31] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (New Orleans, LA, USA) (NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 2020, 29 pages.