Conversational Entity Retrieval from a Knowledge Graph using Aggregation of Fine-grained Relevance Signals with Graph Convolutions and Self-Attention

Mona Zamiri
mona.zamiri@wayne.edu
Wayne State University
Detroit, USA

Alexander Kotov
kotov@wayne.edu
Wayne State University
Detroit, USA

Abstract

The recently introduced task of Conversational Entity Retrieval from a Knowledge Graph (CER-KG) presents unique research challenges due to the complexity of the queries along with the necessity to consider KG structure and the context of an information-seeking dialog. This paper proposes a novel approach to CER-KG that first constructs a sub-graph around each candidate response entity, which includes its neighboring KG components, such as other entities, literals, categories and predicates, and then scores and ranks each candidate answer entity with **D**iverse **R**elevance signal **A**ggregation via **G**raph c**ON**volution (DRAGON), a novel learning-to-rank neural architecture for CER-KG. Unlike previous approaches to CER-KG, DRAGON directly takes a large number of fine-grained relevance signals as input and learns to effectively aggregate and transform those signals into the ranking scores of candidate response entities. In particular, a set of sparse and structured vectors of relevance features used as input to DRAGON measure lexical and semantic similarity between a query in the current turn or responses from the past turns of an information-seeking dialog and each node in the candidate response entity's sub-graph. DRAGON then propagates the relevance signals in feature vectors around the sub-graph using graph convolution layers and aggregates those signals into the candidate response entity ranking score with multi-head attention and fully-connected layers. This design enables DRAGON to attenuate noisy relevance signals from the local KG neighborhood during propagation and attend to the signals from the most important nodes in the candidate entity sub-graph. Our results demonstrate that DRAGON yields significant gains in retrieval accuracy over the previously proposed approach for CER-KG and performs comparably to a much larger fine-tuned cross-encoder architecture.

CCS Concepts

• **Information systems → Retrieval models and ranking**.

Keywords

Entity Retrieval, Conversational Search, Graph Convolutional Networks, Knowledge Graphs, Neural Ranking

WSDM '26, Boise, ID, USA

ACM ISBN 979-8-4007-2292-9/2026/02
https://doi.org/10.1145/3773966.3778001

ACM Reference Format:
Mona Zamiri and Alexander Kotov. 2026. Conversational Entity Retrieval from a Knowledge Graph using Aggregation of Fine-grained Relevance Signals with Graph Convolutions and Self-Attention. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining (WSDM '26), February 22–26, 2026, Boise, ID, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3773966.3778001

1 Introduction

With the growing popularity of knowledge graphs (KGs), the need for efficient and effective access to the vast information stored in them has become increasingly important. Entity Retrieval from a Knowledge Graph (ERKG) [9, 13, 14, 19], a well-studied problem where a ranked list of entities from a given KG, such as DBpedia [17] or Wikidata [33], needs to be retrieved and ranked in response to a textual query, is an important step towards addressing this need. The recently introduced task of Conversational Entity Retrieval from a Knowledge Graph (CER-KG) [39], an extension of ERKG to a conversational setting, and QBLink-KG [39], a benchmark for CER-KG used in this work, pose unique challenges to IR methods due to the complexity of queries and KG structure as well as the necessity to consider the multi-turn context of an information-seeking dialog. For example, consider the query *"In 1967, this Soviet spaceflight crashed, killing Vladimir Komarov, who became the first in-flight fatality in the history of space exploration"* from QBLink-KG. As shown in Figure 1, the only text associated with *"Soyuz 1"*, *"Voskhod 1"* and *"Vostok 3"*, the candidate response entities for this query, are their names, which have no lexical overlap with the query. However, there are many other components of DBpedia, the target KG of QBLink-KG, such as entities (e.g. *"Vladimir Komarov"*), predicates (e.g. *"crashed in"*), literals (e.g. *"Spacecraft launched in 1967"*) and categories (e.g. *"space accidents and incidents"*) in the same KG triplets with the candidate response entities, which can be viewed as clues that can separate the correct response (*"Soyuz 1"*) from very similar (all three candidates are Soviet spacecrafts from the 1960s), but incorrect ones (*"Voskhod 1"* and *"Vostok 3"*).

This example also illustrates other key challenges of CER-KG: (1) queries are verbose and contain both essential clues (*"1967"*, *"Soviet spaceflight"*, *"crashed"*, *"killing Vladimir Komarov"*) to locating the correct response and additional details (*"the first in-flight fatality in the history of space exploration"*) that aren't valuable from retrieval perspective, and may potentially mislead CER-KG methods to returning incorrect responses; (2) candidate entity sub-graph for the correct response entity (*"Soyuz 1"*) besides important relevance signals also contains a significant number of noisy signals in the form of the nodes, such as *"1967 in the Soviet Union"*, *"mission"* and

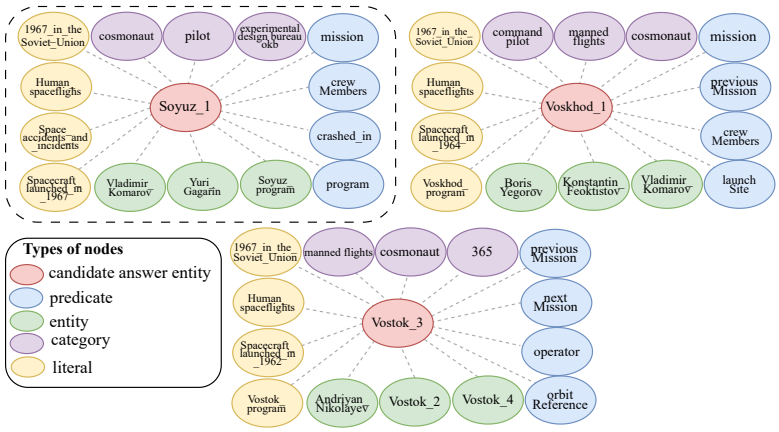

Figure 1: DBpedia sub-graphs with various KG components from the same triplets with the candidate response entities for the query *"In 1967, this Soviet spaceflight crashed, killing Vladimir Komarov, who became the first in-flight fatality in the history of space exploration"*. The sub-graph of the correct response entity *"Soyuz 1"* is highlighted with a dashed rectangle.

"Human spaceflights", which, despite lexical and semantic overlap with the query, are also part of the sub-graphs for other candidate entities, thereby not helping to pinpoint the correct response; (3) since queries in CER-KG are part of multi-turn interactions, conversational context in the form of the preceding dialog turns should be taken into account when ranking responses for the current turn.

An existing approach to CER-KG [39] first builds 10-dimensional feature vectors for candidate answer entities. Each dimension in these vectors corresponds to a *coarse-grained* feature aggregating lexical or semantic similarity scores between the current-turn query or a preceding answer and all KG structural components of the same type (entities, literals, predicates, categories) in the *entire* candidate answer entity sub-graph. This approach then ranks candidate answer entities based on their compact feature vectors by a multi-layer feed-forward neural network.

This paper proposes and explores an alternative approach to feature computation, aggregation and transformation into the candidate response ranking score. Specifically, the proposed approach first constructs a sub-graph around each candidate response entity, where the nodes are the candidate entity itself along with various KG components (entities, predicates, categories and literals) in the same KG triplets, and then computes a *structured* vector of *fine-grained* relevance features for each node in this sub-graph. The diverse relevance signals in these *sparse* feature vectors capture lexical (based on term overlap) and semantic (based on distributed representations) similarity between each node in the candidate entity sub-graph and the current-turn query or the dialog context.

The feature vectors associated with the nodes of the candidate entity sub-graph are propagated, aggregated and transformed into a ranking score of the candidate entity with **D**iverse **R**elevance signal **A**ggregation via **G**raph c**ON**volution (DRAGON), a novel learning-to-rank architecture for CER-KG that consists of graph convolution, multi-head attention and fully-connected layers. Unlike the prior approach to CER-KG [39] that, by adopting straightforward aggregation of a large number of relevance signals into compact feature vectors, disregards the candidate entity's local sub-graph topology and loses the ability to control the influence of each node on the relevance score of the candidate entity, DRAGON takes as input a large number of vectors with fine-grained relevance signals and learns to effectively aggregate and transform those vectors into the ranking scores of candidate responses. Graph convolution layers enable DRAGON to amplify relevance signals from important nodes in the candidate sub-graph and attenuate signals from the noisy ones. To improve the transformation of aggregated relevance signals into a ranking score, DRAGON applies a multi-head self-attention to the relevance vectors for the nodes in the candidate entity sub-graph obtained through graph convolutions. This allows DRAGON to dynamically weigh the influence of similar nodes in the KG neighborhood of the candidate answer on its relevance to the query, capturing relevance patterns missed by graph convolutions.

In summary, this paper addresses two key research questions in the context of CER-KG: (1) how to accurately capture fine-grained semantic and lexical relevance signals for candidate answer entity ranking? and (2) how to efficiently aggregate diverse relevance signals into the ranking score of candidate KG answer entities in a KG structure- and dialog-aware fashion? Its main technical contributions are: (1) novel approach and learning-to-rank neural architecture for CER-KG that utilize sparse feature vectors directly capturing fine-grained and dialog-aware semantic and lexical relevance signals in the candidate answer entities and their immediate KG neighborhood; (2) innovative application of graph convolutional networks (GCNs) and self-attention to propagate, filter and aggregate diverse relevance features into the ranking scores of candidate responses in learning-to-rank neural architectures.

2 Related Work

2.1 Graph Convolutional Networks

By enabling controlled representation learning on graphs, GCNs constitute one of the methodological backbones of our approach. The original spectral GCN [16] introduced a localized first-order

approximation of graph convolutions, producing node representations that fuse feature information with local graph topology. Subsequent GCN variants, such as the inductive graph sampling and aggregation (GraphSAGE) model [12] learn local neighborhood sampling and aggregation functions that generalize to unseen nodes, while the relational GCN architecture [26] extends the basic convolution to multi-relational edges, enabling to address two important KG-related tasks: entity classification and link prediction. Other graph neural network (GNN) architectures incorporate attention and address scalability issues. In particular, in Graph Attention Networks [31] each node can learn to attend to neighbors with different weights, and Message Passing Neural Networks [10] provide a unifying view of GNNs through iterative message and update functions. These advances demonstrated that representations of graph nodes can be enriched by propagating information from other nodes, making GCNs a powerful tool for reasoning on KGs.

2.2 GCN-based Approaches to QA and KGQA

Some of the previously proposed methods for Knowledge Graph Question Answering (KGQA) ground a natural language question and candidate answers in a subgraph of a KG and then apply a GNN to reason over that subgraph. These methods can be divided into three methodological categories: 1) **Methods for multi-hop GNN-based reasoning over a static subgraph**: Knowledge-Aware Graph Networks [18] constructs a schema graph from ConceptNet connecting question and answer entities and uses a GCN combined with Long Short-Term Memory networks and hierarchical path-based attention to score each candidate answer. Likewise, the Multi-hop Graph Relation Network [8] attaches a multi-hop relational reasoning module to a pre-trained language model, performing multi-hop, multi-relational inference over a subgraph of the KG. QA-GNN [37] further connects the question and answer context with the KG by merging question and answer nodes into a joint graph; node representations for both text and KG elements are then mutually updated via a GNN, enabling the model to learn jointly assess relevance and reason. 2) **Fusion of KG and text for open-domain QA**: GRAFT-Net [28] fuses text and KG by constructing a question-specific heterogeneous subgraph of text passages and KG entities, and applies a GNN over this subgraph to extract answers. PullNet [27] iteratively builds a question-focused subgraph with a GCN pinpointing which current nodes merit expansion from a text corpus or a KG to grow the subgraph, and a similar GCN is used to extract the answer from the subgraph. 3) **GNNs for conversational QA**: EXPLAIGNN [5] builds a heterogeneous graph from a mixture of sources and then iteratively reduces this graph via a GNN to locate the answer.

This prior work demonstrates that question subgraph-based reasoning can improve candidate answer selection, yet none of the previously proposed methods apply GCNs or GNNs to aggregate relevance feature vectors of the nodes in those subgraphs into candidate answer scores. Furthermore, they do not weave conversational context into the sub-graph itself – in the proposed approach, each node in the sub-graph is associated with a vector of features capturing similarity with a current query and the dialog context. *In DRAGON, the GCN layers propagate these dialog-aware relevance signals, acting as feature aggregators and filters rather than being a machinery for learning graph node representations that GCNs were originally invented for.*

2.3 Entity Retrieval from KGs and Its Conversational Extension

The task of ERKG, which is conceptually (results are always entities) and methodologically (resulting entities, unlike KGQA, can be obtained with traditional IR pipelines consisting of candidate selection and ranking steps), has been studied in parallel with KGQA. By constructing a plain [22] or multi-field [4, 20, 29, 41] entity document from all information about an entity provided by a KG (such as textual labels, predicates, attribute values, and categories), early methods framed ERKG as ad hoc plain or structured document retrieval with candidate entity document selection based solely on lexical overlap with queries and scoring done by traditional (TF-IDF, BM25, BM25F) retrieval functions [22, 29]. Building on that foundation, specialized lexical retrieval models [20, 41] and non-neural learning-to-rank approaches [4] adopted the same structured entity documents, but focused on more effectively combining lexical matching statistics, yielding consistent improvements in retrieval accuracy.

The neural approaches to ERKG range from attentive feed-forward networks to determine important phrases in queries and learn per-field relevance weights while matching query and entity text in a shared latent space [1] to transformer-based encoders that map both queries and entities to dense representations [9, 35] or decoders that autoregressively generate answer entities directly [6]. Moving beyond point-wise embeddings, Query2Box [24] represents both conjunctive queries and KG entities as hyper-rectangular regions in the same vector space, thereby capturing intersection semantics and hierarchical constraints that plain embeddings cannot. Recent approaches moved towards graph-based reasoning that exploits the topology of the query-induced subgraph. Specifically, such methods first construct a subgraph anchored in the query and then apply a ranking model to score the nodes of this subgraph [14].

3 Proposed Approach

Given an information-seeking dialog $\mathcal{D} = \{(q_1, r_1), \ldots, (q_n, r_n)\}$ with n turns, the task of CER-KG defined in [39] is to produce a response r_i for the query q_i in the ith dialog turn, which is a KG entity that can be obtained by retrieving and ranking a set of candidate answer entities C_{q_i} from a KG while taking into account the dialog context in the form of the responses from the preceding turns $r_{i-1}, \ldots, r_1$ of $\mathcal{D}$. The proposed approach to CER-KG[1] illustrated in Figure 2 consists of three steps: 1) selection of candidate response entities and construction of the sub-graphs around them; 2) feature extraction; and 3) ranking of candidate response entities with DRAGON; which are discussed in detail below.

3.1 Candidate Answer Entities Selection and Sub-graphs Construction

Given the massive scale of modern KGs, information-seeking approaches involving them require strategies to narrow down the

[1] source code is publicly available at https://github.com/teanalab/DRAGON

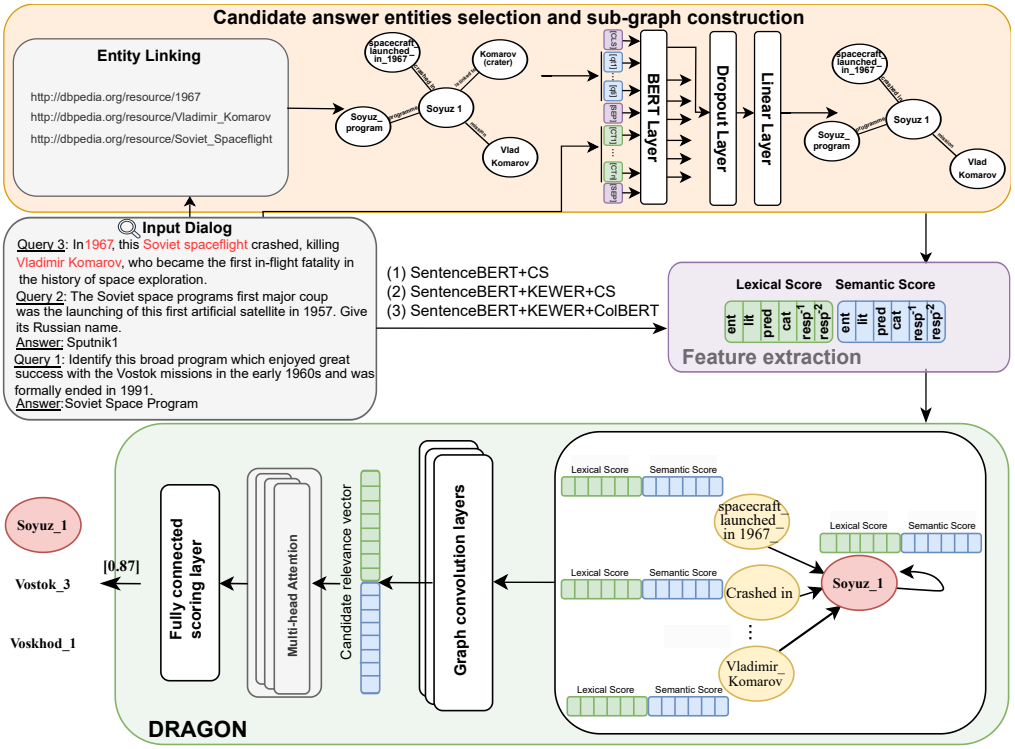

Figure 2: An overview of the proposed approach for conversational entity retrieval from a knowledge graph.

search space. To this end, the ERKG, CER-KG, and KGQA methods first typically select a sub-graph or a set of sub-graphs from a KG, which are then used to obtain the response(s). In a similar vein, given a query q_i in the ith turn of an information-seeking dialog $\mathcal{D}$, the proposed approach uses the method in [39] to select a set of candidate response entities $C_{q_i} = \{c_1, c_2, ..., c_k\}$ for this dialog turn, which includes all KG entities linked from q_i[2] and all other KG entities in the same KG triplets as the linked entities. Next, a sub-graph $\mathcal{G}_{c_j} = (\mathcal{V}_{c_j}, \mathcal{E}_{c_j})$ specific to each candidate response entity $c_j \in C_{q_i}$ is constructed. The set of nodes $\mathcal{V}_{c_j}$ in this sub-graph consists of the candidate response entity c_j and all its immediate neighbors $\mathcal{N}(c_j)$ in a KG, which includes all entities, predicates, categories or literals that are part of the same KG triplets as c_j. Undirected edges $\mathcal{E}_{c_j}$ in this sub-graph are between c_j and each KG neighbor $n \in \mathcal{N}(c_j)$ that belongs to the same KG triplet. The k sub-graphs $\{\mathcal{G}_{c_1}, \ldots, \mathcal{G}_{c_k}\}$ that are constructed for each candidate response entity to q_i contain both relational and textual information about it from a KG. Based on the findings from the prior work on CER-KG [39], all relevance signals necessary to identify the correct response entity are localized within the local KG neighborhood around it. The key challenges facing CER-KG methods are thus to accurately capture and aggregate those signals into the ranking scores of candidate response entities that place the correct response over incorrect ones.

Given significant variance in the degree of entities in a typical KG, the proposed method employs a two-step pruning procedure to ensure manageable size of all candidate entity sub-graphs. The first step in this procedure limits the maximum size of these sub-graphs to eliminate the outliers, while a more sophisticated approach is employed in the second step to reduce the size of these sub-graphs.

To determine the upper bound for the size of candidate entity sub-graphs, we examined the number of neighboring nodes for candidate answer entities for queries in the training set. This analysis revealed that 90% of the candidate answer entities have fewer than 1,000 nodes in their sub-graphs. To eliminate the negative effect of outlier candidate answer entities with more than 1,000 neighboring nodes on the efficiency of the semantics-based approach employed in the second step while also preserving representativeness of the sub-graphs, we randomly retained 1,000 of each outlier entity's neighboring nodes. Candidate entity sub-graphs with fewer than 1000 nodes were unaffected by this step.

To decrease the size of candidate entity sub-graphs by eliminating the parts that are unlikely to contain any significant relevance signals, we employed a pruning method from [14]. Specifically, we fine-tuned the pre-trained BERT [7] model to determine the degree of semantic relatedness of all entities in the candidate entity sub-graph to the query. As illustrated in the sub-graph pruning section of Figure 2, the query q_i and each node n in the sub-graph $\mathcal{N}(c_j)$ for the candidate response entity c_j are scored with the function σ below that is based on the neural architecture with BERT encoder, dropout and linear projection layers from [14] to identify the least relevant nodes to the query q_i according to:

[2]to ensure fair comparison, we used the same entity linker as in [39]

$$\sigma(q_i, n) = \lambda * cos(BERT_{cls}(q_i), BERT_{cls}(n)) + (1 - \lambda) * \hat{\sigma}(q_i, n) \quad (1)$$

where $BERT_{cls}(\cdot)$ denotes the scalar output obtained from the last layer of this architecture - linear projection of the embedding of the [CLS] token, $cos(\cdot, \cdot)$ denotes the cosine similarity function, $\lambda \in [0, 1]$ is a hyperparameter to balance the two components of the scoring function and $\hat{\sigma}(\cdot, \cdot)$ is the ground truth semantic relatedness score computed as follows:

$$\hat{\sigma}(q_i, n) = \begin{cases} 1, & \text{if } n \in \mathcal{N}(c_j) \text{ and } c_j \text{ is relevant,} \\ -1, & \text{otherwise.} \end{cases}$$

The above model was trained on the training set of QBLink-KG using the mean squared error loss. We then ranked all nodes in each candidate entity sub-graph with the trained model and retained 100 top-ranked nodes in each sub-graph.

3.2 Feature Extraction

To accurately assess the relevance of each candidate entity $c_j \in C(q_i)$ to q_i, a feature vector $\bar{\boldsymbol{h}}$ containing diverse relevance signals is first constructed for c_j and each $n \in \mathcal{V}_{c_j}$. This vector captures lexical and semantic similarity scores between c_j or n and q_i as well as the information-seeking dialog context that consists of r_{i-1} and r_{i-2}, the responses in two preceding information-seeking dialog turns[3], and was shown to be important for the accurate assessment of the relevance of c_j in prior work on CER-KG [39]. $\bar{\boldsymbol{c}}_j \in \mathbb{R}^{12}$ or $\bar{\boldsymbol{n}} \in \mathbb{R}^{12}$, 12-element *sparse feature vectors* associated with c_j or $n \in \mathcal{V}_{c_j}$ are constructed using lexical $f_w(\cdot, \cdot)$ and semantic $f_s(\cdot, \cdot)$ similarity scoring functions based on lexical and distributed representations of c_j or n and q_i, r_{i-1}, r_{i-2}:

$$\bar{\boldsymbol{h}} = [\mathtt{ent}_w, \mathtt{lit}_w, \mathtt{pred}_w, \mathtt{cat}_w, \mathtt{resp}_w^{-1}, \mathtt{resp}_w^{-2}, \mathtt{ent}_s, \mathtt{lit}_s, \mathtt{pred}_s, \mathtt{cat}_s, \mathtt{resp}_s^{-1}, \mathtt{resp}_s^{-2}], \quad (2)$$

where $\bar{\boldsymbol{h}}$ corresponds to $\bar{\boldsymbol{c}}_j$ in the case of the feature vector for a candidate answer entity c_j and to $\bar{\boldsymbol{n}}$ in the case of the feature vector for a node $n \in \mathcal{V}_{c_j}$. The functions used to compute the features, along with the feature descriptions, are provided in Table 1.

There are two important differences between the feature vectors used as input to DRAGON and the feature vectors employed by NACER, a previously proposed neural architecture for entity ranking in CER-KG [39]. First, unlike the method in [39], which constructs vectors of *coarse-grained* features *only for candidate answer entities*, the proposed method captures *fine-grained* relevance signals directly in the feature vectors corresponding to *each node in the candidate answer entity sub-graph*. DRAGON then propagates and aggregates those feature vectors using graph convolutions in subsequent layers. Second, as a consequence of their fine-grained nature, the feature vectors constructed by the proposed method are *sparse*, since the vector elements that correspond to the node types other than the one that the node associated with the vector belongs to are set to zero (e.g. the elements $\mathtt{ent}_w$, $\mathtt{pred}_w$, $\mathtt{cat}_w$, $\mathtt{ent}_s$, $\mathtt{pred}_s$, $\mathtt{cat}_s$ of the feature vector for a literal are set to 0). This ensures that each feature vector, regardless of the type of the node it corresponds to, has a consistent size and structure that preserves type-specific information during feature vector aggregation.

Lexical features are computed as a weighted Jaccard similarity coefficient between the bag-of-word token sets $\mathcal{A} = \{a_1, ..., a_n\}$ and $\mathcal{B} = \{b_1, ..., b_m\}$:

$$f_w(\mathcal{A}, \mathcal{B}) = \frac{\sum_{w \in \mathcal{A} \cap \mathcal{B}} SIF(w)}{\sum_{w \in \mathcal{A} \cup \mathcal{B}} SIF(w)}, \quad (3)$$

When computing lexical similarity, each token w is assigned a smoothed inverse frequency weight [25]:

$$SIF(w) = \frac{\lambda}{\lambda + n(w)}, \quad (4)$$

where λ is a hyperparameter and $n(w)$ is the frequency of w in the KG. This way, $f_w(\mathcal{A}, \mathcal{B})$ quantifies lexical similarity as the term overlap between $\mathcal{A}$ and $\mathcal{B}$ normalized by the union of distinct terms in both sets while prioritizing rare (i.e. low-frequency) terms.

Semantic features are computed using a non-parametric function $f_s(\mathbf{a}, \mathbf{b})$ based on $\mathbf{a}$ and $\mathbf{b}$, distributed (i.e. dense vector) representations of a and b. We experiment with the following approaches for computing semantic similarity between the distributed representations of q_i, r_{i-1}, r_{i-2} and the nodes $n \in \mathcal{N}(c_j)$. All of these approaches utilize publicly available[4] same-space embeddings of words and entities, predicates, categories, and literals in DBpedia created by the KEWER method [19]:

- **SBERT+CS**: uses pre-trained SentenceBERT (SBERT) [23], BERT fine-tuned to project short textual fragments into a vector space, where semantically similar fragments are close to each other. SBERT was shown to be superior to BERT and other neural architectures for the task of assessing semantic textual similarity [3]. In this case, $f_s(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T\mathbf{b}$, where $\mathbf{a}$ and $\mathbf{b}$ are distributed representations of a and b obtained using mean-pooling over all output embeddings of SBERT;
- **SBERT+KEWER+CS**: uses pre-trained SBERT [23] combined with the K-Adapter framework [34] to inject KG-specific information in the KEWER embeddings into the distributed representations produced by the pre-trained SBERT. In this case, $f_s(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T\mathbf{b}$;
- **SBERT+KEWER+ColBERT**: uses SBERT with the K-Adapter for the KEWER embeddings and $f_s(\mathbf{a}, \mathbf{b}) = ColBERT(\mathbf{a}, \mathbf{b})$, which computes semantic similarity as a sum of the dot products between all pairs of the most similar token embeddings created by SBERT for a and b [15].

3.3 Candidate Answer Entity Ranking with DRAGON

To address the challenge of effectively dealing with a large number of sparse and potentially noisy relevance signals in the candidate response entity itself and its KG neighborhood, we propose DRAGON, a learning-to-rank architecture based on graph convolutions and self-attention. As shown in Figure 2, DRAGON takes as input the sparse feature vectors containing lexical and semantic relevance signals for the candidate response entity itself and all nodes in its

[3] since information-seeking dialogs in QBLink-KG consist of up to 3 turns. DRAGON feature vectors can be easily expanded to accommodate more preceding responses

[4] https://rb.gy/dldyo9

Feature	Feature function	Feature description
$\texttt{ent}_w$	$f_w(q_i, e)$	computed only for the nodes $e \in \mathcal{V}_{c_j}$ that correspond to entities (set to 0 for the nodes of other types) and measures similarity between lexical representations of q_i and an entity e (candidate entity response c_j or a KG entity that is either subject or object in the same KG triplet as c_j)
$\texttt{lit}_w$	$f_w(q_i, l)$	computed only for the nodes $l \in \mathcal{V}_{c_j}$ that correspond to literals (set to 0 for the nodes of other types) and measures similarity between lexical representations of q_i and a literal l in the same KG triplet as the candidate entity response c_j
$\texttt{pred}_w$	$f_w(q_i, p)$	computed only for the nodes $p \in \mathcal{V}_{c_j}$ that correspond to predicates (set to 0 for the nodes of other types) and measures similarity between lexical representations of q_i and a predicate p in the same KG triplet as the candidate entity response c_j
$\texttt{cat}_w$	$f_w(q_i, c)$	computed only for the nodes $c \in \mathcal{V}_{c_j}$ that correspond to categories (set to 0 for the nodes of other types) and measures similarity between lexical representations of q_i and a category c in the same KG triplet as the candidate entity response c_j
$\texttt{resp}_w^{-1}$	$f_w(r_{i-1}, n)$	computed for all nodes $n \in \mathcal{V}_{c_j}$ and measures similarity between lexical representations of r_{i-1}, the response in the preceding dialog turn, and n
$\texttt{resp}_w^{-2}$	$f_w(r_{i-2}, n)$	computed for all nodes $n \in \mathcal{V}_{c_j}$ and measures similarity between lexical representations of r_{i-2}, the response two dialog turns ago, and n
$\texttt{ent}_s$	$f_s(\mathbf{q}_i, \mathbf{e})$	computed only for the nodes $e \in \mathcal{V}_{c_j}$ that correspond to entities (set to 0 for the nodes of other types) and measures semantic similarity between distributed representations of q_i and an entity e (candidate entity response c_j or a KG entity that is either subject or object in the same KG triplet as c_j)
$\texttt{lit}_s$	$f_s(\mathbf{q}_i, \mathbf{l})$	computed only for the nodes $l \in \mathcal{V}_{c_j}$ that correspond to literals (set to 0 for the nodes of other types) and measures semantic similarity between distributed representations of q_i and a literal l in the same KG triplet as candidate entity response c_j
$\texttt{pred}_s$	$f_s(\mathbf{q}_i, \mathbf{p})$	computed only for the nodes $p \in \mathcal{V}_{c_j}$ that correspond to predicates (set to 0 for the nodes of other types) and measures semantic similarity between distributed representations of q_i and a predicate p in the same KG triplet as the candidate entity response c_j
$\texttt{cat}_s$	$f_s(\mathbf{q}_i, \mathbf{c})$	computed only for the nodes $c \in \mathcal{V}_{c_j}$ that correspond to categories (set to 0 for the nodes of other types) and measures semantic similarity between distributed representations of q_i and a category c in the same KG triplet as the candidate entity response c_j
$\texttt{resp}_s^{-1}$	$f_s(\mathbf{r_{i-1}}, \mathbf{n})$	computed for all nodes $n \in \mathcal{V}_{c_j}$ and measures semantic similarity between distributed representations of r_{i-1}, the response in the preceding dialog turn, and n
$\texttt{resp}_s^{-2}$	$f_s(\mathbf{r_{i-2}}, \mathbf{n})$	computed for all nodes $n \in \mathcal{V}_{c_j}$ and measures semantic similarity between distributed representations of r_{i-2}, the response two dialog turns ago, and n

Table 1: Semantic and lexical similarity features associated with candidate answer entities and the nodes in their sub-graphs that capture relevance signals used by DRAGON for ranking candidate answer entities.

sub-graph and aggregates these vectors into a candidate entity ranking score with 3 main components: 1) feature aggregation layers; 2) multi-head self-attention layer; and 3) candidate entity scoring layer; as discussed below.

3.3.1 Feature Aggregation Layers. The use of graph convolutions for aggregation of relevance signals in KGs and, more specifically, for aggregation of the candidate answer entity's relevance feature vector with the feature vectors in its local KG neighborhood is one of the key innovations of the proposed method. To this end, DRAGON includes graph convolution layers, which propagate relevance signals from the feature vectors in the neighboring nodes and combine them with the relevance features of each candidate response entity to enrich them contextually. Unlike the typical applications of GCNs [36] to learn from scratch the distributed representations of graph nodes based on both graph topology and text associated with each node, the graph convolution layers in DRAGON operate on the pre-computed relevance feature vectors, utilizing the message-passing and noise-filtering mechanisms of GCNs to smooth and enrich the relevance features of the candidate answer entity with the most important relevance signals from its KG neighborhood. The key intuitions behind this innovative use of GCNs are threefold. First, after aggregation with the feature vectors of the neighboring nodes, the message-passing capabilities of GCNs ensure that the vectors associated with relevant response entities incorporate more relevance signals than the vectors associated with irrelevant entities. Second, since relevance signals capture the similarity of the dialog context (i.e., current query and preceding answers) not only with the candidate response entity, but also with the KG components in its immediate neighborhood, the message-passing mechanism of GCNs would propagate the relevance signals from the KG neighborhood of relevant candidate response entities when the relevance signals in their own feature vectors are not strong enough to rank them at the top (e.g. due to indirect and descriptive nature of a current-turn query or dependencies on previous responses in the dialog). Third, the noise-filtering capabilities of GCNs enable DRAGON to handle a large number of relevance signals, some of which may be spurious, thereby allowing it to be effective for complex and verbose queries.

Formally, given $\mathcal{G}_{c_j}$ and the matrix $\mathbf{H}$ of feature vectors for all nodes in $\mathcal{G}_{c_j}$, each graph convolution layer in DRAGON iteratively updates $\mathbf{H}$ by propagating feature vectors around $\mathcal{G}_{c_j}$ and integrating the feature vectors of each node with those of its neighbors:

$$\mathbf{H}^{(l+1)} = \delta\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right) \tag{5}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of $\mathcal{G}_{c_j}$ with added self-loops, $\tilde{\mathbf{D}}$ is the diagonal node degree matrix of $\tilde{\mathbf{A}}$, $\mathbf{H}^{(l)}$ is the matrix

of node relevance vectors prior to the lth graph convolution layer (with $\mathbf{H}^{(0)}$ corresponding to the matrix with the original feature vectors), $\mathbf{W}^{(l)}$ is a trainable convolution layer-specific weight matrix that can be viewed as a relevance signal filter, and $\delta(\cdot)$ denotes a non-linear activation function.

3.3.2 Multi-Head Attention Layer. After aggregation of the relevance feature vectors with graph convolution layers based on the KG structure, DRAGON uses multi-head self-attention [30] to further enrich the relevance feature vectors associated with the nodes in the candidate entity sub-graph based on their similarity to the feature vectors associated with other nodes in the same sub-graph. Specifically, the node feature matrix $\mathbf{H}$ is mapped onto query ($\mathbf{Q}$), key ($\mathbf{K}$), and value ($\mathbf{V}$) using learnable linear projections:

$$\mathbf{Q} = \mathbf{W_Q H}, \quad \mathbf{K} = \mathbf{W_K H}, \quad \mathbf{V} = \mathbf{W_V H}$$

The attention weights are computed via a scaled dot-product attention:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right),$$

where d is a scaling factor. The updated representation of relevance vectors is obtained as:

$$\hat{\mathbf{H}} = \mathbf{A}\mathbf{V}.$$

3.3.3 Candidate Entity Scoring Layer. The final ranking score for the candidate response entity is computed by passing its updated relevance vector through a fully connected layer with sigmoid activation:

$$s(c_j) = \sigma(\mathbf{w}\hat{\mathbf{H}}_{(c_j,\cdot)} + b)$$

where $\hat{\mathbf{H}}_{(c_j,\cdot)}$ is the updated relevance vector corresponding to c_j, $\mathbf{w}$ and b are trainable weight vector and bias, respectively. The candidate response entity c_j is then ranked based on its score $s(c_j)$.

3.3.4 Loss Function. DRAGON was trained based on a list-wise ranking loss from [40] that minimizes the Kullback–Leibler divergence:

$$\mathcal{L} = KL(\text{softmax}(\mathbf{s})||\mathbf{y}) \cdot \alpha_r$$

between a distribution over the predicted candidate entity scores $\mathbf{s}$ and a ground-truth one-hot distribution over entity relevance labels $\mathbf{y}$ and a rank-based penalty α_r, which is proportional to the rank deviation of the correct answer and increases the loss when the gold candidate is ranked lower.

4 Experiments

4.1 Baselines

4.1.1 NACER. NACER [39] is a simple, but strong feature-based learning-to-rank neural architecture for CER-KG. We use the results of NACER's best-performing setup from [39], which utilizes BERT with the KEWER-based K-Adapter as the query encoder and the additive interaction function without parameter sharing for semantic similarity scoring.

4.1.2 ColBERT. Contextualized Late Interaction over BERT (ColBERT) [15] is a strong late-interaction method for dense retrieval that encodes a query and document independently with BERT and computes their relevance score as a sum of the maximum pair-wise token-level similarities. To adapt ColBERT to CER-KG, we treat candidate entity names as documents.

4.1.3 LLaMa. Two pre-trained foundation large language models (LLMs) from the LLaMA-v3 family [11] with 8 (LLaMa3-8b) and 70 billion (LLaMa3-70b) parameters with a zero-shot prompt describing the retrieval task.

4.1.4 SEE+MonoBERT. Similarity-based Early Exit (SEE) [2] is an approach to neural re-ranking that improves the run-time efficiency of MonoBERT [21] with a non-trainable early-exit mechanism. MonoBERT is a cross-encoder re-ranking architecture that feeds both query and document (candidate entity name, in the case of CER-KG) into the same BERT model thereby computing the relevance score based on all interactions between their tokens. SEE+MonoBERT-PT and SEE+MonoBERT-FT use the pre-trained and fine-tuned (on the training set of QBLink-KG) MonoBERT, respectively.

4.2 Configuration of DRAGON

The optimal configuration of DRAGON in terms of the number of graph convolution layers and self-attention heads was determined using the training and validation splits of QBLink-KG by varying these parameters while holding all others fixed.

Figure 3 illustrates the effect of varying the number of graph convolution layers on the retrieval accuracy of DRAGON. As follows from this figure, adding a second graph convolution layer yields pronounced improvement in Hits@1, although the improvement in Hits@10 is notably smaller. Adding a third layer results in an additional marginal increase in Hits@10, which comes at the expense of a moderate reduction in Hits@1. Addition of more graph convolution layers decreases both Hits@1 and Hits@10.

Figure 4 illustrates the impact of the number of self-attention heads on the retrieval accuracy of DRAGON. Moving from no self-attention layer to a self-attention layer with 1 through 8 heads steadily improves both retrieval metrics, with 8 heads providing the optimal balance between them, while expanding to 16 heads leads to a decrease in both metrics, indicating diminishing returns from over-attention.

4.3 Results

4.3.1 Retrieval Accuracy. Table 2 compares the retrieval accuracy in terms of Hits@1, Recall@1 (R@1), Hits@10, Recall@10 (R@10), and mean reciprocal rank (MRR) of various combinations of the optimal configuration of DRAGON with four different methods for measuring semantic similarity and two types of neural graph aggregator for the relevance feature vectors against the baselines. Several key conclusions can be drawn from the results in this table.

First, DRAGON consistently outperforms all baselines, including SEE+MonoBERT-PT and demonstrates retrieval accuracy comparable to SEE+MonoBERT-FT, a state-of-the-art cross-encoder architecture with substantially greater capacity (109,483,778 trainable parameters versus 7,537 parameters of the best-performing configuration of DRAGON). Furthermore, although the fine-tuned MonoBERT with the early exit mechanism achieves strong performance across all retrieval metrics, the same model without fine-tuning on the target benchmark performs very poorly, attaining significantly lower retrieval accuracy than all configurations of DRAGON and all of the baselines. This result indicates that effective use of cross-encoders for CER-KG heavily relies on time- and

Table 2: Retrieval accuracy of the DRAGON variants and the baselines on the test set of QBLink-KG. Statistical significance of the difference with NACER, ColBERT, and DRAGON using SBERT+KEWER+ColBERT for semantic similarity computation based on the two-tailed paired Student's t-test with $p = 0.05$ is indicated by †, ‡ and *, respectively.

Method	$f_s(a, b)$	Graph model	Hits@1	R@1	Hits@10	R@10	MRR
NACER	-	-	1121	0.6665	1602	0.9524	0.7575
ColBERT	-	-	1099	0.6534	1267	0.7533	0.7033
LLaMa3-70b	-	-	1036	0.6159	-	-	-
LLaMa3-8b	-	-	664	0.3948	-	-	-
DRAGON	SBERT+CS	GCN	779	0.4631	1506	0.8954	0.5399
DRAGON	BERT+KEWER+CS	GCN	870	0.5172	1508	0.8966	0.6257
DRAGON	SBERT+KEWER+ColBERT	GCN	1174‡	0.6980‡	1347‡	0.8008‡	0.7279‡
DRAGON	SBERT+KEWER+CS	GCN	1261†‡*	0.7497†‡*	1639†‡*	0.9744†‡*	0.8021†‡*
DRAGON	SBERT+KEWER+CS	GAT	908	0.5398	1590	0.9453	0.6177
DRAGON*	SBERT+KEWER+CS	GCN	1163	0.6914	1405	0.8353	0.7213
SEE+MonoBERT-PT	-	-	293	0.1742	1363	0.8103	0.3303
SEE+MonoBERT-FT	-	-	**1285**	**0.7640**	**1643**	**0.9768**	**0.8061**

Table 3: Retrieval accuracy of DRAGON using different types of relevance features.

Method (Graph model)	$f_s(a, b)$	Features	Hits@1	R@1	Hits@10	R@10	MRR
DRAGON (GCN)	-	Lexical	443	0.2634	1427	0.8484	0.3937
DRAGON (GCN)	SBERT+KEWER+CS	Semantic	660	0.3924	1401	0.8329	0.4795
DRAGON (GCN)	SBERT+KEWER+CS	Semantic+Lexical	**1261**	**0.7497**	**1639**	**0.9744**	**0.8021**

resource-intensive fine-tuning. Moreover, the negligible gain of cross-encoders over DRAGON in retrieval accuracy also comes with a significantly higher computational cost during inference. On dual Nvidia H100 NVL GPUs, DRAGON scores each candidate entity in QBLink-KG in approximately 0.007 seconds per query, while MonoBERT with SEE in about 0.03 seconds (4x slower). This highlights the trade-off between effectiveness and efficiency of CER-KG models, and demonstrates that DRAGON achieves competitive retrieval performance with substantially lower inference latency and model complexity. While NACER (an even smaller model than DRAGON) yields Hits@1 of 1121 and MRR of 0.7575, and ColBERT achieves 1099 and 0.7033, respectively, the best variant of DRAGON using a combination of graph convolution for feature vector aggregation and SBERT with KEWER adapter for semantic similarity computation attains Hits@1 of 1261 and MRR of 0.8021, corresponding to the relative improvements of 12.5% and 5.9% in Hits@1 and MRR, respectively, further highlighting the optimal combination of efficiency and effectiveness of the pipelines for CER-KG that aggregate semantic and lexical signals via graph convolution and refine them with self-attention. The lower retrieval accuracy (Hits@1 and Hits@10 drop to 1163 and 1405, respectively) of DRAGON*, the variant of DRAGON in which self-attention precede graph convolution layers, indicates that the order of graph convolution and attention layers plays an important role in such pipelines. Reversing it leads to decreased retrieval accuracy, which we attribute to premature aggregation of noisy relevance signals. Additionally, the zero-shot prompting of LLaMA3-70b achieves Hits@1 of 1036, significantly outperforming the smaller 8B variant, but remaining below the ColBERT's 1099. This result highlights that, while foundation LLMs capture some aspects of challenging IR tasks, such as CER-KG, in a zero-shot setting, they fall short not only of much smaller task-specific architectures, but also of both late-interaction and cross-encoder models.

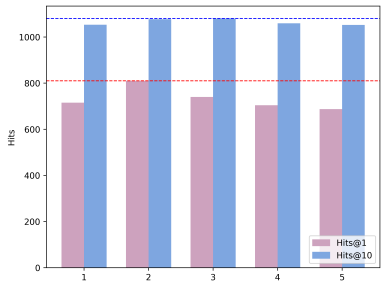

Figure 3: Impact of the number of graph convolution layers on retrieval accuracy of DRAGON.

Second, the choice of neural model for relevance signal aggregation significantly influences retrieval performance. In particular, DRAGON with SBERT and KEWER adapter for semantic similarity computation achieved Hits@1 of 908 and MRR of 0.6177 when using Graph Attention Networks (GAT) [32] for relevance signal aggregation, whereas substituting GAT with graph convolutions raises Hits@1 by nearly 39% and MRR by almost 30%. When investigating the reason behind this gap, we found that the immediate KG neighborhood of a candidate response entity typically contains

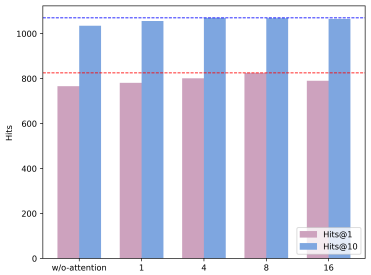

Figure 4: Impact of the number of self-attention heads on retrieval accuracy of DRAGON.

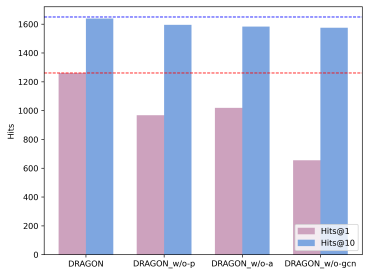

Figure 5: Impact of removing different components of DRAGON on its retrieval accuracy.

both important and noisy nodes. GAT's learned attention weights can over-emphasize irrelevant neighbors, thereby degrading the quality of the aggregated relevance vectors [38]. In contrast, more flexible aggregation leveraging noise filters built into graph convolutions effectively attenuates distracting and unimportant relevance signals from the immediate KG neighborhood of the candidate answer entities, even when the signal-to-noise ratio is small, which is typical of verbose queries and a conversational setting.

Third, combining SBERT with the K-Adapter framework to inject KG-specific information in KEWER embeddings into semantic relevance computation yields greater performance gains than using SBERT alone. Specifically, DRAGON using SBERT only attains Hits@1 of 779 and MRR of 0.5399, whereas the addition of KEWER increases Hits@1 to 1261 and MRR to 0.8021. At the same time, using ColBERT-style approach to computing semantic similarity via aggregation of maximum token-level similarity scores in conjunction with KEWER embeddings achieves Hits@1 of 1174 and MRR of 0.7279, trailing the combination of SBERT with KEWER. Unlike NACER, which also uses BERT+KEWER encoder, but trades fine-grained control over feature aggregation for learnable semantic feature functions and simpler neural architecture, DRAGON relies on a more sophisticated architecture leveraging simpler non-learnable semantic similarity functions. This design makes DRAGON more dependent on the quality of the input embedding space for computing semantic similarity, thus limiting BERT+KEWER+CS's effectiveness due to BERT's suboptimal alignment with semantic similarity-based distance metrics. In contrast, SBERT+KEWER+CS leverages an embedding space better aligned for computing semantic features with cosine similarity [23], leading to its superior performance even without learnable functions for relevance feature computation.

4.3.2 Ablation Studies. We conducted an ablation study on the best-performing configuration of DRAGON to measure the relative contribution of graph convolution and self-attention layers to its performance. Figure 5 illustrates Hits@1 and Hits@10 during this experiment. Specifically, we evaluated three variants of DRAGON. First, we examined whether the candidate answer entities selection and subgraph pruning method is effective at pruning unrelated nodes while preserving relevant candidate entities. To this end, we compare the performance of DRAGON with the configuration in which the node pruning step is omitted (denoted DRAGON_w/o-p). Comparing DRAGON_w/o-p with DRAGON shows that pruning contributes to better performance in terms of Hits@1 and Hits@10. Examination of the retrieval accuracy of DRAGON_w/o-a and DRAGON_w/o-gcn, the models ablating multi-head attention and replacing graph convolution with simple averaging of the relevance feature vectors, respectively, reveals decreases in Hits@10 and especially in Hits@1 in both cases.

To quantify the relative importance of semantic versus lexical similarity signals in DRAGON, Table 3 reports Hits@1 and MRR when removing either type of features. Using only semantic similarity features causes Hits@1 to fall from 1261 to 660 (−47.7%) and MRR to fall from 0.8021 to 0.4795 (-40.2%), while using only lexical features yields Hits@1 of 443 (−64.9%) and MRR of 0.3937 (-50.1%). Although semantic features contribute more heavily to DRAGON's retrieval effectiveness, both feature types are critically important.

5 Conclusion

In this paper, we present a novel approach for conversational entity retrieval from a knowledge graph, which first constructs and prunes a sub-graph for each candidate response entity, then computes 12-dimensional sparse vectors of semantic and lexical relevance features for each node in this sub-graph and aggregates, enriches and transforms those vectors into the candidate response entity ranking score with a novel learning-to-rank neural architecture comprising graph convolution and self-attention layers. Experiments demonstrate that the proposed approach performs comparably to a much larger cross-encoder model and yields significant gains in retrieval accuracy over the previously proposed approach for CER-KG and a late-interaction approach to dense retrieval. We also experiment with different methods for computing semantic similarity based on distributed representations of KG components, query and dialog context and feature aggregation to determine the best configuration of the proposed approach. The proposed approach is not specific to DBpedia and can be easily adapted to other KGs, such as Wikidata.

Acknowledgments

This study was supported by the National Institutes of Health under the award number R21NR020388. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. We thank the anonymous reviewers for their detailed feedback and insightful suggestions.

Ethical Considerations

This work does not involve human subjects, personal data, or sensitive information. The proposed method is trained and evaluated solely on publicly available knowledge graphs and synthetic benchmarks. While improved entity retrieval models could be applied in broader systems with downstream implications, such as information access or conversational agents, our method operates at the representation and ranking level without user interaction or profiling. We believe this work poses no foreseeable ethical risks, but we encourage responsible application.

References

[1] Saeid Balaneshinkordan, Alexander Kotov, and Fedor Nikolaev. 2018. Attentive Neural Architecture for Ad-hoc Structured Document Retrieval. In *Proceedings of the 27th ACM International on Conference on Information and Knowledge Management (CIKM'18)*. 1173–1182.

[2] Francesco Busolin, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, Salvatore Trani, and Alberto Veneri. 2025. Efficient Re-ranking with Cross-Encoders via Early Exit. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'25)*.

[3] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity - Multilingual and Cross-lingual Focused Evaluation. In *Proceedigs of the 11th International Workshop on Semantic Evaluation (SemEval'17)*. 1–14.

[4] Jing Chen, Chenyan Xiong, and Jamie Callan. 2016. An empirical study of learning to rank for entity search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR'16)*. 737–740.

[5] Philipp Christmann, Rishiraj Saha Roy, and Gerhard Weikum. 2023. Explainable Conversational Question Answering over Heterogeneous Sources via Iterative Graph Neural Networks. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'23)*.

[6] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*.

[7] 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 (NAACL-HLT'19)*. 4171–4186.

[8] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable Multi-Hop Relational Reasoning for Knowledge-Aware Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*.

[9] Emma J Gerritse, Faegheh Hasibi, and Arjen P de Vries. 2022. Entity-aware Transformers for Entity Search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'22)*. 1455–1465.

[10] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*.

[11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, and *et al.* 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783* (2024).

[12] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*.

[13] Emma J. Gerritse, Faegheh Hasibi, and Arjen P. de Vries. 2020. Graph-embedding empowered entity retrieval. In *Proceedings of the 42nd European Conference on Information Retrival (ECIR'20)*.

[14] Parastoo Jafarzadeh, Zahra Amirmahani, and Faezeh Ensan. 2022. Learning to rank knowledge subgraph nodes for entity retrieval. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval (SIGIR'22)*. 2519–2523.

[15] 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 (SIGIR'20)*.

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedgins of the 5th International Conference on Learning Representations (ICLR'17)*.

[17] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[18] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*.

[19] Fedor Nikolaev and Alexander Kotov. 2020. Joint Word and Entity Embeddings for Entity Retrieval from a Knowledge Graph. In *Proceedings of the 42nd European Conference on Information Retrieval (ECIR'20)*. 141–155.

[20] Fedor Nikolaev, Alexander Kotov, and Nikita Zhiltsov. 2016. Parameterized fielded term dependence models for ad-hoc entity retrieval from knowledge graph. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR'16)*. 435–444.

[21] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424* (2019).

[22] Jeffrey Pound, Peter Mika, and Hugo Zaragoza. 2010. Ad-hoc Object Retrieval in the Web of Data.. In *In Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. 771–780.

[23] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP'19)*.

[24] Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over Knowledge Graphs in Vector Space using Box Embeddings. In *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*.

[25] Arora Sanjeev, Liang Yingyu, and Ma Tengyu. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the 5th International Conference on Learning Representations (ICLR'17)*.

[26] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *Proceedings of the 15th European Semantic Web Conference (ESWC'18)*.

[27] Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. 2019. PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*.

[28] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. 2018. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP'18)*.

[29] Alberto Tonon, Gianluca Demartini, and Philippe Cudré-Mauroux. 2012. Combining Inverted Indices and Structured Search for Ad-hoc Object Retrieval.. In *In Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'12)*. 125–134.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*.

[31] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph Attention Networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*.

[32] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. 2018. Graph attention networks. In *Proceedgins of the 6th International Conference on Learning Representations (ICLR'18)*.

[33] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[34] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2021. K-adapter: Infusing knowledge into pre-trained models with adapters. In *Findings of the Association for Computational Linguistics (ACL'21)*. 1405–1418.

[35] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP'20)*. 6397–6407.

[36] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2020), 4–24.

[37] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'21)*.

[38] Yang Ye and Ji Shihao. 2021. Sparse graph attention networks.. In *IEEE Transactions on Knowledge and Data Engineering*. 905–916.

[39] Mona Zamiri, Yao Qiang, Fedor Nikolaev, Dongxiao Zhu, and Alexander Kotov. 2024. Benchmark and Neural Architecture for Conversational Entity Retrieval from a Knowledge Graph. In *Proceedings of the ACM on Web Conference*

(WWW'24). 1519–1528.

[40] George Zerveas, Navid Rekabsaz, Daniel Cohen, and Eickhoff Carsten. 2022. CODER: An efficient framework for improving retrieval through COntextual Document Embedding Reranking. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP'22)*.

[41] Nikita Zhiltsov, Alexander Kotov, and Fedor Nikolaev. 2015. Fielded sequential dependence model for ad-hoc entity retrieval in the web of data. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. 253–262.