Knowledge Graphs for Enhancing Large Language Models in Entity Disambiguation

Gerard Pons(✉), Besim Bilalli, and Anna Queralt

Universitat Politècnica de Catalunya, UPC-BarcelonaTech, Barcelona, Spain
{gerard.pons.recasens,besim.bialli,anna.queralt}@upc.edu

Abstract. Recent advances in Large Language Models (LLMs) have positioned them as a prominent solution for Natural Language Processing tasks. Notably, they can approach these problems in a zero or few-shot manner, thereby eliminating the need for training or fine-tuning task-specific models. However, LLMs face some challenges, including hallucination and the presence of outdated knowledge or missing information from specific domains in the training data. These problems cannot be easily solved by retraining the models with new data as it is a time-consuming and expensive process. To mitigate these issues, Knowledge Graphs (KGs) have been proposed as a structured external source of information to enrich LLMs. With this idea, in this work we use KGs to enhance LLMs for zero-shot Entity Disambiguation (ED). For that purpose, we leverage the hierarchical representation of the entities' classes in a KG to gradually prune the candidate space as well as the entities' descriptions to enrich the input prompt with additional factual knowledge. Our evaluation on popular ED datasets shows that the proposed method outperforms non-enhanced and description-only enhanced LLMs, and has a higher degree of adaptability than task-specific models. Furthermore, we conduct an error analysis and discuss the impact of the leveraged KG's semantic expressivity on the ED performance.

Keywords: Knowledge Graphs · Entity Disambiguation · Large Language Models

1 Introduction

The association of textual mentions in a document to the entities they refer to in a Knowledge Graph (KG) is crucial for many Natural Language Processing (NLP) applications, such as question answering or information retrieval. This task is known as Entity Linking (EL), and it is a fundamental step in the transformation of unstructured text into structured knowledge. EL is usually performed as a pipeline with three different steps. The first one is Mention Detection, which detects the text spans that could possibly be linked to entities. It is followed by the Candidate Generation stage, which selects for each mention the top k entities from the KG that could refer to it, usually based

G. Demartini et al. (Eds.): ISWC 2024, LNCS 15231, pp. 162–179, 2025.
https://doi.org/10.1007/978-3-031-77844-5_9

on precomputed probability distributions from entity-mention hyperlink pairs. Finally in the Entity Disambiguation (ED) step, a final entity is selected from the previously generated set.

Usually, the ED problem is tackled by designing and training task-specific models with large amounts of data (e.g., Wikipedia dumps) [4,9]. In recent years, language models have been used for this task by making use of the mention's context in the document to disambiguate between the possible solutions [6,9, 19]. Additionally, some approaches incorporate the candidates' descriptions [23], classes [30] (e.g., the categories they are tagged with in Wikipedia) or both [4] in the model's input, by generating encodings for these text items. The addition of this knowledge allows zero-shot ED, enabling the models to classify entities that may have not been seen during training time.

Lately, new advances in Large Language Models (LLMs) such as GPT-3 [7], GPT-4 [2] or LLaMA-2 [39], have demonstrated remarkable performance in numerous NLP problems [43]. Given their large-scale and diverse training corpus, they are good candidates to perform tasks, even zero-shot ones, where general knowledge is needed for language processing, such as ED [44]. However, these LLMs still face some challenges, such as hallucination (i.e., the generation of statements that are factually incorrect) [18], and the lack of knowledge about concepts outside their training corpus. To mitigate these issues, the use of KGs to enhance LLMs has been proposed to address different problems [31]. There exist a large variety of KGs, storing information which can be encyclopedic (e.g., DBpedia [3] or YAGO [36], which extract information from Wikipedia), commonsense knowledge (e.g., ConceptNet [35], with information such as $\langle house, has, door \rangle$ or $\langle bed, usedFor, sleep \rangle$) or domain specific [1]. The explicit and structured knowledge they contain can be used to enhance the performance of LLMs, by leveraging it either during pre-training by enriching the training data [17], or during the inference stage [5,37,41]. Following the nomenclature proposed in [31], in this work we focus on KG-enhanced LLM inference, and apply it to the ED task. Our approach takes advantage of KGs to avoid retraining the LLM, and improves the effectiveness of zero-shot LLM approaches for ED.

Solving the ED problem using only LLMs would require to instruct them to choose one of the entities from the candidate set given the document containing the mention. Instead, we propose to extract the candidates' class taxonomy from a KG and use it to guide the disambiguation. For example, taking Query 1 in Fig. 1, given 'MTV awards' appearing in the context, the entity 'Justin' is more likely to be a *Musician* than a *Politician*. Thus, we can use this context to guide the LLM by eliminating invalid solutions such as 'Justin Trudeau', rather than letting the LLM directly predict the entity. Moreover, when all the remaining candidates fall directly under the same class, we retrieve the candidates' descriptions from a Knowledge Base (KB), such as Wikipedia, and append them to the disambiguation prompt (see Fig. 1, query 2). With this Retrieval Augmented Generation (RAG) [21] stage, we provide the LLM with reliable information,

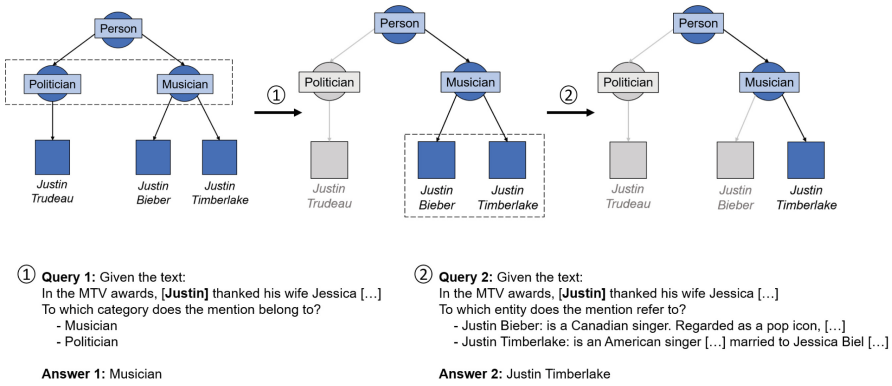

Fig. 1. Overview of the two steps of our approach.

reducing hallucination and enabling the LLM to perform predictions over new or unusual entities which may not have been present in the training corpus.

Therefore, our contributions are as follows:

- We present a method to enhance LLMs in the ED task by leveraging the candidate entity class taxonomies available in KGs. Moreover, we also augment the prompt with the entity descriptions, in order to allow the disambiguation of unseen or difficult entities.
- We evaluate the method against non-enhanced LLMs, description-only enhanced LLMs and a task-specific model by using ten ED datasets. The results show that our approach improves the disambiguation capabilities of LLMs and has a higher degree of adaptability to different domains than the task-specific model.
- We discuss how using KGs with different levels of semantic expressivity (e.g., YAGO and DBpedia) affects the proposed pruning algorithm, by studying both the ED results and the algorithm's performance.
- We study and classify the cases in which our method fails to correctly disambiguate the mention and provide insights on the possible improvements.

The remainder of the paper is structured as follows. In Sect. 2 we discuss the Related Work. In Sect. 3 we formalize the problem and present the proposed approach. In Sect. 4 we describe the different experiments, discuss the results, and conduct an error analysis. Finally, in Sect. 5 we provide our conclusions and ideas for future work.

2 Related Work

This section begins with an overview of different ED methods which leverage external information to improve their predictions. Then, the recently emerged concept of KG-enhanced LLM prompting is introduced, enumerating some of the proposed methods for tackling NLP tasks.

2.1 Knowledge-Augmented ED

Various ED approaches use model architectures that leverage the mention, its surrounding context and candidate entities to generate a solution [6,9,19]. However, some recent works incorporate additional knowledge to the model's input in order to improve the disambiguation of entities which are not present in the training dataset. This extra information is usually gathered from online sources (e.g., Wikipedia) and provided in the form of entity descriptions [23], entity types [30] or both [4]. Additionally, some works leverage the structured information contained in KGs to enhance the model's performance. In [33], information about the entity types from DBpedia and knowledge graph embeddings extracted from Wikipedia's graph structure are incorporated into the model's input. In [38], KG triples are used to train a component of the model's architecture which predicts the existence of facts between mentions in a given document. The result of this prediction is used as input for the final model, which also leverages entity types and descriptions. Finally in [27], the triples from the KG are verbalized and appended to the input sentence before being fed to the model.

These knowledge-augmented ED approaches incorporate the additional information to their model's input, which are mainly built by leveraging LLMs such as BERT [10], RoBERTa [22] or BART [20], and need to be trained or fine-tuned with large amounts of data (e.g., Wikipedia dumps with millions of entities). In contrast, in our approach we rely on the new generation of generative LLMs (e.g., GPT-3 [7], GPT-4 [2] or LLaMA-2 [39]), and solve the ED task by prompting the LLMs in a zero-shot manner without needing to train a task-specific model. This prompting approach has also been explored in [44], where the document's context and the inherent knowledge from the LLM are enriched with the entity textual descriptions, following a RAG approach [21]. RAG has been shown to be useful for incorporating new or relevant information to LLMs, and it has also been leveraged in a specific step of our proposal. However, our main focus is on the usage of KGs to obtain the class hierarchy for the candidate entities, which allows our method to solve the ED task by guiding the LLM to the correct answer (see Sect. 3.2).

2.2 KG-Enhanced LLM Prompting

LLMs can be used to solve a wide range of tasks, not just ED. However, as introduced in Sect. 1, LLMs suffer from problems such as hallucination, which can be accentuated if the information requested is outdated or not present in the training data. Retraining LLMs to incorporate this missing knowledge is expensive and time-consuming, and fine-tuning them could lead to problems such as catastrophically forgetting (i.e., the LLMs' tendency to lose previously obtained knowledge when being fine-tuned with new data) [24]. To solve these issues, KGs can be used as a source of additional structured information in different NLP tasks. In particular, information from a KG can be added to the prompt fed to the LLMs, a technique coined as KG Prompting [31], which has already been explored for Question Answering. In [41], the approach starts by

identifying the entities in the question, and then the KG is queried to build subgraphs including them. After that, the LLM is prompted to comprehend and aggregate the subgraphs, and based on the consolidated result it is asked to reason over it and provide the answer. Similarly in [37], the LLM generates these subgraphs by iteratively exploring a KG to create a reasoning path over it. In each iteration, if the LLM believes that has enough information, an answer is provided. Otherwise, it is prompted to continue to traverse the graph, adding the most promising relation to the existing reasoning path each time. Finally in [5], the entities are also first extracted from the question, which are then used to retrieve the triples they participate in within the KG. Then, the triples are verbalized and appended to the prompt as context, which is fed to the LLM to obtain the answer.

In our approach, however, we solve a different task, ED, and we rely on the KG's ontology rather than on the annotated instances, guiding the disambiguation of the entities using the class hierarchy.

3 ED with KG-Enhanced LLMs

In this section we lay out the formulation of the problem to be solved and describe the two different steps of the proposed method.

3.1 Problem Formulation

Let $C = \{e_1, e_2, ...e_{|C|}\}$ be a set of k candidate entities belonging to a KG, containing a class hierarchy in which the entities are annotated, and m be a mention in a document d. The objective of ED is to assign to m the entity e it refers to, such that $e \in C$.

3.2 Method

Our proposed method for the disambiguation of the mention can be divided in two steps. First, a subgraph is generated containing the candidate entities together with their taxonomy of classes. Then, a pruning algorithm is applied to iteratively discard the candidate entities until there is only one left in the subgraph, which will be the solution (see Fig. 1). The implementation can be found in the Supplemental Material.

Subgraph Generation. Given the candidate set C for a mention m, a directed-acyclic graph (DAG) G is created from the KG, having the general class *Thing* as its 'root' (i.e., the only node without predecessors) and the candidate entities as 'leaves' (i.e., the nodes without successors). Note that G cannot be considered a tree as a node can have multiple predecessors (e.g., 'Justin Timberlake' is a linked to the class *Musician* and also to the class *Actor*).

First of all, the candidate entities are linked to the classes they belong to (see Fig. 2, step 1). Then, the classes that are not predecessors of any of the candidates

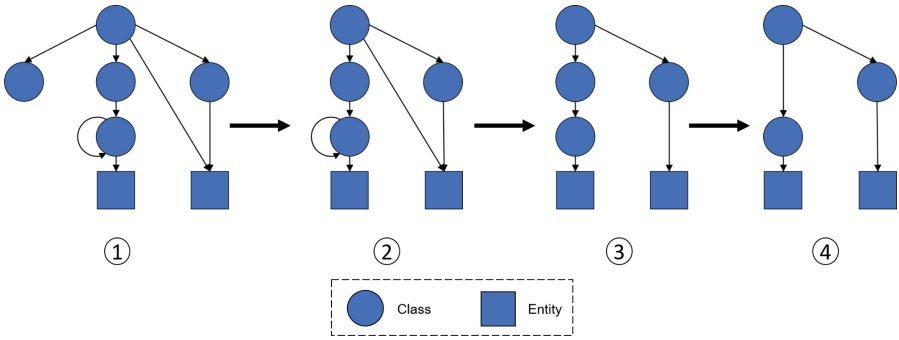

Fig. 2. Overview of the steps for the creation of the DAG.

are removed from G (see Fig. 2, step 2). Next, the relations that can be inferred by traversing G through more granular path of relations are also removed, as well as self-pointing relations (see Fig. 2, step 3). Finally, intermediate nodes which only have one direct successor and that successor is not an entity are also iteratively removed from G, linking the direct successor to the node's direct predecessors (see Fig. 2, step 4). With this last step, we aim to increase the granularity and ease the disambiguation, as the classes in the higher levels of the hierarchy tend to be more abstract (e.g., for the class *Musician*, the path from the root in DBpedia is *Thing* → *Species* → *Eukaryote* → *Person* → *Artist* → *Musician*). In some of the more complex KGs (e.g., YAGO), an entity could also be considered a class. Therefore, if there exist other entities in the candidate set that are linked to this entity, an extra preprocessing step is needed to transform the entity into a leaf, by removing the links to its direct successors while linking them to its direct predecessors.

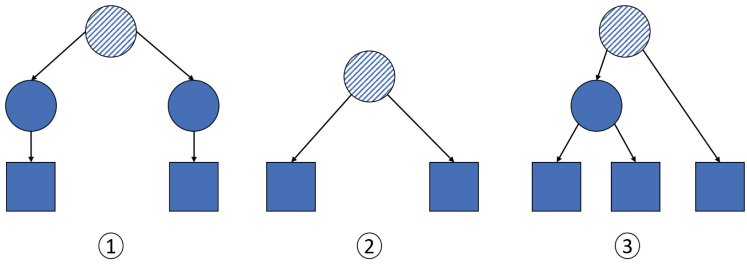

Fig. 3. Example of the three different configurations of the LCA's direct successors.

Algorithm 1: Pruning candidates

Input: Subgraph G, mention m, document d and entity *descriptions*
Output: Entity

```
1  candidates ← leaves(G);
2  while len(candidates) ≠ 1 do
3      LCA ← LCA(G, candidates);
4      directSuccessors ← directSuccessors(G, LCA);
5      if allDirSuccessorsAreClasses then
6          response ← multiChoice(directSuccessors ∪ {None}, m, d);
7          if response ≠ None then
8              G ← prune(G, directSuccessors \ {response});
9          else
10             response ← multiChoice(candidates, m, d, descriptions);
11             G ← prune(G, candidates \ {response});
12     else if allDirSuccessorsAreEntities then
13         response ← multiChoice(directSuccessors, m, d, descriptions);
14         G ← prune(G, directSuccessors \ {response});
15     else
16         D_c, D_e ← getClassesAndEntities(directSuccessors);
17         response ← multiChoice(D_c ∪ {Other}, m, d);
18         if response = Other then
19             G ← prune(G, D_c);
20         else
21             G ← prune(G, directSuccessors \ {response});
22     candidates ← leaves(G);
```

Pruning Algorithm. The pruning algorithm is outlined in Algorithm 1. Given the generated graph G and the initial candidate entities (i.e., its leaves), the algorithm starts by finding the Lowest Common Ancestor (LCA) of the candidate entities. The LCA is defined as the deepest node (i.e., the furthest from the root) which is an ancestor of all the candidates (see dashed nodes in Fig. 3). Then, the direct successors of the LCA are retrieved, which leads to three different scenarios:

1. **All the direct successors are classes (Fig. 3, case 1):** The LLM is prompted to select to which classes the mention m belongs to. All the candidate classes that are not chosen by the LLM are removed from G, along with all the nodes that have become disconnected from the root. This case corresponds to lines 5–12 in Algorithm 1.
2. **All the direct successors are entities (Fig. 3, case 2):** The LLM is prompted to directly select the entity m refers to. Here, the description of each candidate entity is retrieved from a KB and appended to the prompt. The non-selected candidates are then removed from G. This case corresponds to lines 13–15 in Algorithm 1.

3. **Direct successors are classes and entities (Fig. 3, case 3):** The direct successors are organized into classes (D_c), and entities (D_e). The LLM is then prompted to select a class from $D'_c = D_c \cup Other$, where *Other* is an additional class which encompasses D_e. If the LLM selects a class belonging to D_c, the remaining classes and the entities D_e are removed from G. If *Other* is selected, the classes from D_c are removed. Finally, the nodes which have become disconnected from the root are also removed. This case corresponds to lines 16–22 in Algorithm 1.

During the initial tests it was found that the LLM may not return a valid response when it considered that none of the presented classes matched the mention. Therefore, in *case 1* we additionally add the class *None*, which triggers a *case 2* prompt with the remaining candidates if it is selected. Finally, in order to guarantee that the LLM always has information about the entity before making a decision, the response is assessed by the LLM when a single entity is left after a *case 1* or *case 3* step. If it is negatively evaluated, a complete *case 2* prompt is triggered.

The algorithm runs until there is only one leaf (i.e., entity) left in G, which will be the final response. Therefore, in the worst-case scenario there will be k iterations.

4 Experiments

In this section we discuss the experiments performed. First, we describe the experimental settings, then we evaluate our proposal against different methods and also analyze the effect of the KG used. Finally we study the different scenarios that lead to our method failing to correctly disambiguate the mention.

4.1 Settings and Datasets

Datasets. We evaluate the approach on ten popular ED datasets, the same as in [44], which are from news and online articles (MSN [8], AQU [25], ACE04 [32], CWEB [11], R128 [34] and R500 [34]), from Wikipedia (WIKI [13], OKE15 [28] and OKE16 [29]) or from hand-crafted, brief and ambiguous sentences (KORE [15]). These datasets contain documents for which one or various mentions have been annotated with the ground truth entity they refer to. The dataset statistics are summarized in Table 1.

Candidate Sets. To allow for better comparability, we borrow the candidate sets from [44], which combine two methods to obtain sets of size 10. First, as done in previous works [4,9,19], Wikipedia hyperlink count statistics from mention-entity pairs are used to generate the candidates. If not enough candidates are found, the set is augmented by generating candidates with the BLINK model [42], which is based on dense retrieval from context and descriptions.

Table 1. Overview of ten considered datasets' statistics.

	# Docs	# Mentions	Avg. # Characters
KORE	50	144	76.4
ACE04	35	257	2285.0
OKE16	173	288	186.2
R500	357	524	164.8
OKE15	101	536	183.9
R128	113	650	818.8
MSN	20	656	3380.1
AQU	50	727	1415.9
WIKI	319	6793	1624.6
CWEB	320	11154	7575.9

Knowledge Graphs. To obtain the hierarchical representation of the classes we use YAGO [36]. It primarily leverages the information from Wikipedia's infoboxes for generating the relations between entities, and for the taxonomy it borrows the top-level representation from the schema.org ontology [12], which is further refined by carefully integrating it with the fine-grained Wikidata [40] taxonomy. Additionally, in Sect. 4.3 we study the effect of the granularity of the annotation of the classes. To this end, we use another KG with a more simple class hierarchy, DBpedia [3], which is also built on top of Wikipedia but uses a shallow and manually created ontology to define the representation of classes. Finally, to retrieve the entity descriptions we use Wikipedia as a KB, and they are truncated at 250 characters before being appended to the prompt.

Evaluation Metric. We report our results with inKB micro-F1 score (see Eq. 1). InKB means that we only consider a mention if the ground truth entity is present in the KG used. To allow comparability between KGs (i.e., YAGO and DBpedia do not have the same entities annotated), we also report the results by considering the percentage of the Gold F1 score achieved. The Gold F1 score is the maximum inKB micro-F1 score that could be obtained if the method always produced a correct answer when possible, as the candidate sets do not always contain the ground truth entity.

$$\text{micro-F1} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})} \tag{1}$$

Additionally, given the differences in dataset sizes we report the weighted average, weighting each score by considering the number of instances of each dataset.

Large Language Models. To perform our experiments we use GPT-3.5, concretely the *gpt-3.5-turbo-1106* model from OpenAI API, setting its temperature

to 0 to decrease the randomness and the creativity of the response, as we are interested in factual answers. The reason behind the selection of this LLM is in a trade-off between reasoning capabilities, operating cost and API availability.

4.2 Results

To evaluate the proposed method we compare it to a non-enhanced LLM baseline and to ChatEL [44], the only approach that to the best of our knowledge also directly prompts LLMs to solve in a zero-shot manner the ED task, without training or fine-tuning any model. Additionally, we compare it to ReFinED [4], the task-specific model, which requires extensive training, that obtained the best ED performance in the results reported in [44]:

- **Baseline:** The baseline consists in asking the LLM to directly select one of the entities within the set of candidates. Therefore, it does not have the class representation nor the entities' description. This baseline corresponds to a non-enhanced LLM approach.
- **ChatEL** [44]**:** The ED task is solved in two steps. First, the LLM is asked to describe what the mention in the document is referring to. Then, another prompt is created asking the LLM to select the candidate entity that best matches the description generated in the previous response, by also enriching the candidates with their descriptions from Wikipedia. It must be noted that in our approach an answer is always returned. However, in [44] an empty result is produced (i.e., a prediction is not performed) when the LLMs' response does not contain any candidate entity. Thus, the computation of the precision (and consequently the inKB and Gold F1-score) differs, as the number of false negatives can potentially be reduced. These observed differences in the F1-score have been mitigated by computing the achieved gold percentage, making the proposals comparable.
- **ReFinED** [4]**:** Is a ED-specific method built over the RoBERTa architecture, leveraging entity types and descriptions. It is pretrained with a Wikipedia dataset, with more than 100M mention-entity pairs, and fine-tuned on AIDA-CoNLL [16], a news related ED dataset with approximately 25.000 annotated mentions.

The results are shown in Table 2. First of all, it can be observed that the proposed approach outperforms the baseline in all of the datasets. This demonstrates that even with the vast amounts of data with which the LLMs have been trained and their reasoning capabilities, the addition of external knowledge on the prompts and the guidance during the disambiguation can be helpful to improve the performance on the ED task. One of the most frequent mistakes made by the baseline approach is to give more importance to the context than to the mention. For instance, in the sentence *'A six-game begins this Friday in Phoenix and the team hopes to get O'Neal [...]'*, the baseline links the mention to the entity 'Phoenix Suns', presumably given the basketball context. However, the mention is referring to a place, which is correctly resolved by the KG-enhanced

Table 2. Results for the inKB micro F1-score for the ED experiments with ten datasets. The ChatEL and ReFinED scores are taken from the results reported by the authors in [44]. The weighted average weights each score by taking into consideration the sizes of the datasets. The best score for each dataset is highlighted in **bold**.

		KORE	ACE04	OKE16	R500	OKE15	R128	MSN	AQU	WIKI	CWEB	Avg.	Wt. avg.
Baseline	F1-Score	68.2	89.1	59.0	77.4	64.1	68.7	82.3	62.4	69.5	65.0		
	Gold F1	76.5	95.4	82.2	85.3	82.2	83.6	94.0	96.2	89.4	89.3		
	% Gold	89.3	93.4	71.7	90.8	78.0	82.1	87.5	64.8	77.7	72.8	80.8	75.4
ReFinED [4]	F1-Score	56.7	86.4	79.4	70.8	78.1	68.0	89.1	86.1	84.1	73.8		
	Gold F1	88.0	96.9	90.3	92.1	90.3	91.1	97.0	98.1	94.4	94.3		
	% Gold	64.4	89.1	**87.9**	76.8	86.4	74.6	**91.8**	**87.7**	**89.0**	**78.2**	82.8	**82.6**
ChatEL [44]	F1-Score	78.7	89.3	75.2	82.2	75.8	78.9	88.1	76.7	79.1	70.9		
	Gold F1	88.0	96.9	90.3	92.1	90.3	91.1	97.0	98.1	94.4	94.3		
	% Gold	89.4	92.1	83.2	89.2	83.9	86.6	90.8	78.1	83.7	75.1	85.2	79.3
$Our_{DBpedia}$	F1-Score	71.3	89.4	65.9	75.4	73.5	75.0	84.2	72.0	72.5	67.7		
	Gold F1	80.1	95.7	79.8	85.6	82.2	85.9	94.1	96.3	90.4	89.4		
	% Gold	88.9	93.3	82.5	88.0	**89.3**	87.3	89.4	74.8	80.2	75.7	85.0	78.8
Our_{YAGO}	F1-Score	71.8	88.7	65.8	78.3	70.3	75.8	81.2	72.0	74.4	69.6		
	Gold F1	79.6	94.3	83.7	85.2	82.3	84.8	92.2	94.4	88.9	89.3		
	% Gold	**90.1**	**94.0**	78.6	**91.9**	85.4	**89.4**	88.0	76.2	83.6	77.9	**85.5**	81.1

approach, as in the first iteration the LLM correctly disambiguates between the classes *Organization*, *Place*, *Product* or *FictionalEntity*.

Regarding the comparison with ChatEL, it can be observed that better results are obtained by our approach in 6 out of 10 datasets, with a weighted average score of 1.8% points higher. Additionally, in the complete ChatEL evaluation GPT-4 is used, which is bigger and more powerful LLM than GPT-3.5 [2] with a cost per token more than 20 times higher.[1] Therefore, even while using a less powerful LLM, the proposed approach leads to improvements in the ED task. Additionally, the added cost of the manipulation of the graph structure (i.e., finding the LCA and pruning) is limited by the small number of candidates used in ED, which typically ranges from 5 to 30, and its execution time is two orders of magnitude lower than the LLM calls.

For the task-specific model, we can observe that it obtains a better performance in 6 of the datasets, and an average weighted score of 1.5% points higher. However, it is worth noting that it has been trained over a huge Wikipedia dataset and fine-tuned on an ED dataset about news, and for the only dataset out of these domains, KORE, our model outperforms it by more than 25% points. Therefore, the LLM methods show a greater degree of adaptability, and could compensate the decrease in performance on some datasets by not requiring the training of specific models.

[1] https://openai.com/pricing.

4.3 KG Expressivity Impact

In this section we evaluate how the differences in the semantic expressivity of the taxonomy of classes in the KG affects our approach. Concretely, we explore if reducing the granularity of the taxonomy affects its disambiguation capabilities. To this end, we use YAGO and DBpedia KGs, whose statistics are summarized in Table 3. It can be observed that YAGO has more than a thousand times as many classes as DBpedia, and nearly doubles the average depth of the path from an entity to the root. Therefore, YAGO has a more granular class representation and also annotates more semantic interpretations of the entities. For instance, as it is exemplified in Fig. 4, in the annotation of Barcelona in YAGO a distinction is made between its representation as a *Place* and as a *Organization*, whereas in DBpedia Barcelona is only considered as a *Place*. Additionally, we can also observe the difference in the number of classes and its granularity. For example, DBpedia stops at the city level, while YAGO classifies the municipalities also within their country and region.

To evaluate the two KGs under study, we repeat the same experimental settings as in Sect. 4.1 for DBpedia, keeping in mind that the inKB entities do not completely overlap on both KGs, thus affecting the Gold F1-score. The results can be seen in Table 2, where in 7 out of 10 datasets the more granular class representation, YAGO, has a better performance, and the weighted average score is 2.4% points higher. This reinforces the hypothesis that having a more semantically rich class taxonomy can help in the disambiguation task. We can observe that YAGO does not outperform DBpedia primarily in the OKE datasets, which

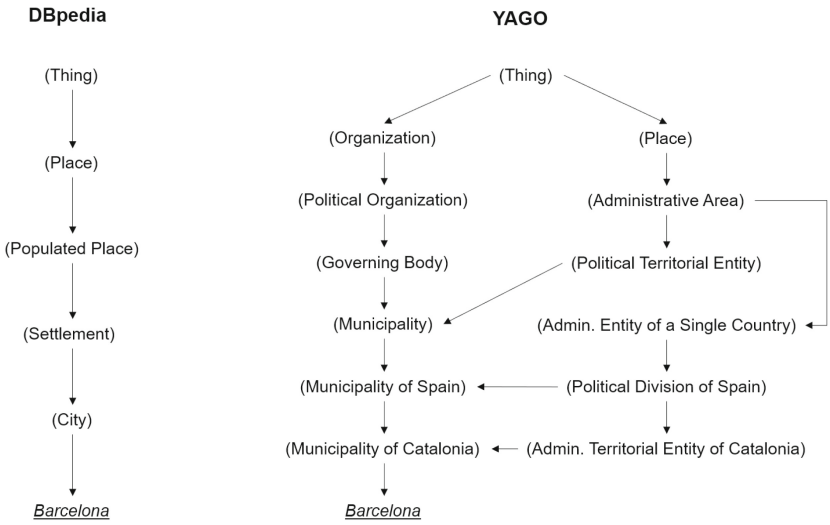

Fig. 4. Class representation of the entity Barcelona in DBpedia (left) and YAGO (right) KGs.

contain a large number of mentions referring to generic occupations (e.g., Governor, Judge, Engineer, etc.). For instance, in the second iteration of the method for the sentence *'As governor, Reagan raised taxes [...]'*, the disambiguation is between the entity 'Governor' which is the ground truth answer, and the class *Head of Government*, which has other entities as successors (e.g., 'Governor of California'). This causes a *case 3* (see Sect. 3.2) disambiguation between *Head of Government* and *Other*, which leads to the LLM selecting the former as it properly fits the context.

Table 3. Metric comparisons from YAGO and DBpedia KGs [14].

	DBpedia	YAGO
# Instances	5,044,223	6,349,359
# Classes	760	819,292
Avg. tree depth	3.51	6.61
Avg. branching factor	4.53	8.48

Regarding the number of iterations both KGs exhibit a similar behavior, having a mean value close to 2.2 (see Table 4). Therefore, even though YAGO has a deeper taxonomy, it is compensated by its superiority in semantic expressivity and mitigated by the elimination of intermediary nodes in the preprocessing step (see Sect. 3.2). Hence, given that the execution time of the graph manipulation is two orders of magnitude lower than the LLM calls, using deeper graphs does not significantly affect the performance.

Table 4. Percentage of disambiguated entities in which the pruning algorithm reached a final single entity within the specified number of iterations.

	Iterations						Avg. Iterations
	1	2	3	4	5	6	
YAGO	26.24%	37.36%	26.60%	8.30%	1.32%	0.15%	2.21
DBpedia	23.57%	43.00%	26.68%	6.12%	0.42%	0.01%	2.18

4.4 Error Analysis

We thoroughly examined and categorized the scenarios that led to our method producing an incorrect disambiguation, as understanding them is crucial for assessing the capabilities and limitations of LLMs in this task.

Ground Truth Errors. These errors consider the inaccuracies in the annotation of the datasets. For instance, in the sentence *'[...] it is required excellent English communication skills [...]'*, the mention English is annotated as 'England' instead of 'English Language'.

KG Errors. These errors encompass the problems derived from the annotation of the entities' classes in the KGs. For example, in the sentence *'Mars, Galaxy and Bounty are chocolate [...]'* the ground truth answer 'Bounty (chocolate bar)' is wrongly annotated in DBpedia as an *Architectural Structure*, causing the pruning algorithm to fail. Additionally, the annotations could also suffer from inconsistencies. For instance, the 'Supreme Court of Florida' falls under the *Organization* class, while the 'Supreme Court of California' is considered a *Building*.

Ambiguous Errors. Some datasets contain sentences with high degree of ambiguity. For instance, in the sentence *'Justin, Stefani and Kate are among the most popular people both on MTV and Twitter'*, the disambiguation between 'Justin Timberlake' and 'Justin Bieber' is not clear as both are popular celebrities in those platforms and have collaborated with the other mentioned artists. Moreover, there are some ground truth labels that could be argued to be incorrect. For example, in the sentence *'accepted the post of principal and only teacher at a primary school in rural Blaauwbosch, Newcastle.'*, principal is annotated in the ground truth as 'Principal (Academia)', yet for primary schools in the UK a more appropriate term would be 'head teacher', which is also found in the candidate set.

LLM Errors. Finally, some errors are produced by the LLM's response. These are usually originated by the LLM missing information from the context and incorrectly resolving the entity or by wrongly interpreting the mention and assigning it to an erroneous class.

In Table 5, all the errors from the two smaller datasets (i.e., KORE and ACE2004) have been classified according to the presented types of error. This study has not been extended to all the datasets as it is unfeasible due to their sizes. Regarding the ground truth error, it corresponds to the sentence *'Onassis married Kennedy on October 20, 1968'*, where the mention Onassis is annotated as 'Jacqueline Kennedy Onassis' instead of 'Aristotle Onassis'. For the KG errors, 2 are originated from a missing class annotation and 1 from a wrong labeling of an entity. Also, 3 errors for the ambiguous sentences are originated by the context not being sufficient to disambiguate the mention and in 2 of them the LLM's response could arguably be considered also correct (e.g., in the sentence *'The Isle of Wight festival in 1970 was the biggest at its time'*, the mention could be both referring to the musical festival and to the concrete festival's edition). Finally, 5 of the LLM errors are caused by missed context (e.g., in the short sentence *'Tiger lost the US Open'*, the mention Tiger, likely referring to Tiger

Woods, helps to disambiguate between 'US Open (tennis)' and 'US Open (golf)' but it is missed by the LLM) and 7 by a wrong interpretation of the class (e.g., in the sentence *'[...] ran adjacent to an advertisement for a golf tournament on Fox Sports sponsored by Sun Microsystems.'* the mention is interpreted as a TV program rather than a TV channel).

These last LLM errors could potentially be solved by using LLMs with more powerful reasoning capabilities. To explore this idea, a small experiment with GPT-4 and Mistral Large [26] has been run, where the models are able to correctly disambiguate 8 and 7 of these 12 errors, respectively.

Table 5. Error types for the ACE2004 and KORE datasets, using YAGO as the KG.

Error Type	# Errors
LLM error	12
Ambiguous error	5
KG error	3
Ground truth error	1

5 Conclusions

In this work we present a novel method to enhance LLMs with KGs to solve the ED task. For this purpose, we leverage the entities' class taxonomy annotated in a KG to gradually prune the candidates' search space. Additionally, when the disambiguation is at the entity level we add the entities' descriptions to the prompt. This proposal allows solving the ED task without training task-specific models or fine-tuning them on domain-specific or new data, which is a time-consuming and expensive process. In the experiments we show that the proposed method outperforms both non-enhanced and description-enhanced LLM approaches, and that it has a higher degree of adaptability to different domains than task-specific methods, which rely on the data they have been trained with. Additionally, we observe how using more semantically expressive KGs improves the ED results without degrading the pruning algorithm's performance. Finally, we analyze the different disambiguation errors and classify them according to their type, drawing conclusions about them. Specifically, and as a future line of work, the usage of more powerful LLMs could be studied, which could help in the disambiguation of difficult mentions.

Supplemental Material Statement: Datasets and scripts containing the prompts and the algorithms can be found in the attached repository.[2]

[2] https://github.com/gerardponsrecasens/KGLLMs4ED.

Acknowledgements. This work is supported by the Horizon Europe Programme under GA.101093164 (ExtremeXP) and the Spanish Ministerio de Ciencia e Innovación under project PID2020-117191RB-I00/ AEI/10.13039/501100011033 (DOGO4ML). Anna Queralt is a Serra Húnter Fellow.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abu-Salih, B.: Domain-specific knowledge graphs: a survey. J. Netw. Comput. Appl. **185**, 103076 (2021)
2. Achiam, J., et al.: GPT-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
3. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
4. Ayoola, T., Tyagi, S., Fisher, J., Christodoulopoulos, C., Pierleoni, A.: ReFinED: an efficient zero-shot-capable approach to end-to-end entity linking. In: NAACL (2022)
5. Baek, J., Aji, A.F., Saffari, A.: Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In: Proceedings of the First Workshop on Matching From Unstructured and Structured Data (MATCHING 2023) (2023). https://api.semanticscholar.org/CorpusID:260063238
6. Barba, E., Procopio, L., Navigli, R.: ExtEnD: extractive entity disambiguation. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2478–2488 (2022)
7. Brown, T., et al.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901 (2020)
8. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 708–716 (2007)
9. De Cao, N., Izacard, G., Riedel, S., Petroni, F.: Autoregressive entity retrieval. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, 3–7 May 2021. OpenReview.net (2021). https://openreview.net/forum?id=5k8F6UU39V
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
11. Gabrilovich, E., Ringgaard, M., Subramanya, A.: FACC1: freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0) (2013)
12. Guha, R.V., Brickley, D., Macbeth, S.: Schema.org: evolution of structured data on the web. Commun. ACM **59**(2), 44–51 (2016)
13. Guo, Z., Barbosa, D.: Robust named entity disambiguation with random walks. Semantic Web **9**(4), 459–479 (2018)
14. Heist, N., Hertling, S., Ringler, D., Paulheim, H.: Knowledge graphs on the web-an overview. Knowl. Graphs eXplainable Artif. Intell., 3–22 (2020)

15. Hoffart, J., Seufert, S., Nguyen, D.B., Theobald, M., Weikum, G.: KORE: keyphrase overlap relatedness for entity disambiguation. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 545–554 (2012)
16. Hoffart, J., et al.: Robust disambiguation of named entities in text. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 782–792 (2011)
17. Hu, L., Liu, Z., Zhao, Z., Hou, L., Nie, L., Li, J.: A survey of knowledge enhanced pre-trained language models. IEEE Trans. Knowl. Data Eng. (2023)
18. Ji, Z., et al.: Survey of hallucination in natural language generation. ACM Comput. Surv. **55**(12), 1–38 (2023)
19. Le, P., Titov, I.: Improving entity linking by modeling latent relations between mentions. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 1595–1604. Association for Computational Linguistics, Melbourne, Australia, July 2018. https://doi.org/10.18653/v1/P18-1148
20. Lewis, M., et al.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880. Association for Computational Linguistics, Online, July 2020. https://doi.org/10.18653/v1/2020.acl-main.703
21. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474 (2020)
22. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
23. Logeswaran, L., Chang, M.W., Lee, K., Toutanova, K., Devlin, J., Lee, H.: Zero-shot entity linking by reading entity descriptions. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
24. Luo, Y., Yang, Z., Meng, F., Li, Y., Zhou, J., Zhang, Y.: An empirical study of catastrophic forgetting in large language models during continual fine-tuning. CoRR abs/2308.08747 (2023). https://doi.org/10.48550/ARXIV.2308.08747
25. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518 (2008)
26. Mistral AI: Mistral Large (2024). https://mistral.ai/news/mistral-large/
27. Mulang', I.O., Singh, K., Prabhu, C., Nadgeri, A., Hoffart, J., Lehmann, J.: Evaluating the impact of knowledge graph context on entity disambiguation models. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 2157–2160 (2020)
28. Nuzzolese, A.G., Gentile, A.L., Presutti, V., Gangemi, A., Garigliotti, D., Navigli, R.: Open knowledge extraction challenge. In: Gandon, F., Cabrio, E., Stankovic, M., Zimmermann, A. (eds.) SemWebEval 2015. CCIS, vol. 548, pp. 3–15. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25518-7_1
29. Nuzzolese, A.G., Gentile, A.L., Presutti, V., Gangemi, A., Meusel, R., Paulheim, H.: The second open knowledge extraction challenge. In: Sack, H., Dietze, S., Tordai, A., Lange, C. (eds.) SemWebEval 2016. CCIS, vol. 641, pp. 3–16. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46565-4_1
30. Onoe, Y., Durrett, G.: Fine-grained entity typing for domain independent entity linking. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 8576–8583 (2020)

31. Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: a roadmap. IEEE Trans. Knowl. Data Eng. (2024)
32. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to Wikipedia. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 1375–1384 (2011)
33. Ristoski, P., Lin, Z., Zhou, Q.: KG-ZESHEL: knowledge graph-enhanced zero-shot entity linking. In: Proceedings of the 11th Knowledge Capture Conference, pp. 49–56 (2021)
34. Röder, M., Usbeck, R., Hellmann, S., Gerber, D., Both, A.: N^3-a collection of datasets for named entity recognition and disambiguation in the NLP interchange format. In: LREC, pp. 3529–3533 (2014)
35. Speer, R., Chin, J., Havasi, C.: ConceptNet 5.5: an open multilingual graph of general knowledge. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
36. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706 (2007)
37. Sun, J., et al.: Think-on-graph: deep and responsible reasoning of large language model with knowledge graph (2023)
38. Ayoola, T., Fisher, J., Pierleoni, A.: Improving entity disambiguation by reasoning over a knowledge base. In: NAACL (2022)
39. Touvron, H., et al.: Llama 2: open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)
40. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
41. Wen, Y., Wang, Z., Sun, J.: MindMap: knowledge graph prompting sparks graph of thoughts in large language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (2024)
42. Wu, L., Petroni, F., Josifoski, M., Riedel, S., Zettlemoyer, L.: Scalable zero-shot entity linking with dense entity retrieval. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6397–6407. Association for Computational Linguistics, Online, November 2020. https://doi.org/10.18653/v1/2020.emnlp-main.519
43. Yang, J., et al.: Harnessing the power of LLMs in practice: a survey on ChatGPT and beyond. ACM Trans. Knowl. Discovery Data (2023)
44. Ding, Y., Zeng, Q., Weninger, T.: ChatEL: entity linking with chatbots. In: COLING-LREC (2024)