ParaGraph: Mapping Wikidata Tail Entities to Wikipedia Paragraphs

Natalia Ostapuk
University of Fribourg
Fribourg, Switzerland
natalia.ostapuk@unifr.ch

Djellel Difallah
NYU Abu Dhabi
Abu Dhabi, UAE
djellel@nyu.edu

Philippe Cudré-Mauroux
University of Fribourg
Fribourg, Switzerland
pcm@unifr.ch

***Abstract*—Bridging unstructured data with knowledge bases is an essential task in many problems related to natural language understanding. Traditionally, this task is considered in one direction only: linking entity mentions in a text to their counterpart in a knowledge base (also known as entity linking). In this paper, we propose to tackle this problem from a different angle: linking entities from a knowledge base to paragraphs describing those entities. We argue that such a new perspective can be beneficial to several applications, including information retrieval, knowledge base population, and joint entity and word embedding. We present a transformer-based model, ParaGraph, which, given a Wikidata entity as input, retrieves its corresponding Wikipedia section. To perform this task, ParaGraph first generates an entity summary and compares it to sections to select an initial set of candidates. The candidates are then ranked using additional information from the entity's textual description and contextual information. Our experimental results show that ParaGraph achieves 87% Hits@10 when ranking Wikipedia sections given a Wikidata entity as input. The obtained results show that ParaGraph can reduce the information gap between Wikipedia-based entities and tail entities and demonstrate the effectiveness of our proposed approach towards linking knowledge graph entities to their text counterparts.**

***Index Terms*—Linked Data, Knowledge Graphs, Entity Linking**

I. Introduction

Knowledge Graphs (KGs) provide a source of rich, structured, and machine-readable data for a variety of real-world applications, from question answering and Information Retrieval (IR) to virtual assistants and recommender systems. An essential task in any Natural Language Processing (NLP) approach involving knowledge graphs is *entity linking*. The goal of entity linking is to provide "semantic grounding" for some text leveraging entities in a KG, by determining which textual contents refer to which specific entities [1]. However, this process is typically handled in one direction only.

In this paper, we introduce a "reverse" entity linking task, which we refer to as *entity mapping*. Entity mapping can be formally defined as follows:

Definition 1. *Entity Mapping: Given a knowledge graph entity and a collection of paragraphs, find the paragraph that best describes the input entity.*

In this context, a "paragraph" refers to a self-contained unit of text dedicated to a single concept or named entity.

Tackling the entity mapping task has a number of important implications:

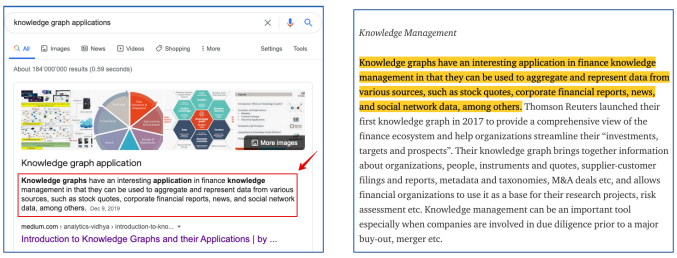
- Detecting paragraphs best describing a KG entity can significantly improve the quality and the efficiency of knowledge base augmentation approaches, which extract new facts from the Web;
- For IR, detecting a specific paragraph that is most relevant to a given query can help users navigate through search results. This feature is incidentally partially implemented in Google Search (Figure 1);
- Document structuring, which is a subtask of Natural Language Generation, can also benefit from links between entities and paragraphs: a text segment describing an entity is a good candidate to become a separate section. This application is particularly useful to Wikipedia page editors: developing a consistent section structure for new pages is not a straightforward task, and section recommendation systems can alleviate this problem.

Although *entity mapping* can be viewed as a typical IR task, we argue that it is different in regard to the content of the target document: while in IR any **relevant** document may match the query, in *entity mapping* we are aiming at retrieving documents (paragraphs) representing the **summary** of a given concept with its labels and relations. For example, in *entity mapping* an entity *fondue* should not be mapped onto a paragraph describing the preparation process of the dish, since such a paragraph only covers one aspect of a given entity. On the other hand, the same paragraph can be highly relevant to a query *fondue* in the context of IR.

In this paper, we focus in particular on one entity mapping task — mapping Wikidata entities onto Wikipedia paragraphs. Wikidata[1] is a collaborative, multilingual knowledge graph hosted by the Wikimedia Foundation. Wikidata has the advantage of being curated by humans and of being tightly integrated with multiple Wikimedia projects (e.g., Wikipedia, Wikimedia Commons, or Wiktionary). Every Wikipedia article across all languages has a corresponding and unique language-independent Wikidata entity. This mapping between Wikipedia

978-1-6654-8045-1/22/$31.00 ©2022 IEEE

[1]https://www.wikidata.org/



(a) Featured snippet (b) Source webpage

Fig. 1. (a) The featured paragraph is a special box appearing on top of a Google search results page, and contains an answer to the user's query. (b) After clicking on the link from the featured paragraph, the user is directed to the relevant fragment of text on the source web page. This fragment is additionally highlighted.

and Wikidata is beneficial for both projects. On the one hand, it facilitates information extraction and standardization of Wikipedia articles across languages, which can benefit from the standard structure and values of their Wikidata counterpart, e.g., for populating infoboxes. On the other hand, Wikipedia articles are routinely updated, which in turn keeps Wikidata fresh and useful for online applications.

However, the Wikipedia editorial guidelines require that an entity be notable or worthy of notice to be added to the encyclopedia, which does not hold for all Wikidata entities. Hence, only a fraction of Wikidata entities have a corresponding Wikipedia article in some language[2]. We refer to the remaining entities, which do not have an article in Wikipedia, as *orphans*. In the absence of a textual counterpart, orphans often suffer from incompleteness and lack of maintenance.

Our present effort stems from the observation that a substantial number of orphan entities are indeed represented in Wikipedia, but not at the page level; orphan entities are often described within existing Wikipedia articles in the form of sections, subsections, and paragraphs of a more generic concept or fact. For example, the English Wikipedia does not have a dedicated page about "Tennis racket", it is instead embedded in the "Racket" page as a section[3], whereas it can be found as a standalone (orphan) entity on Wikidata ("Q153362").[4] Interestingly, even a short section describing an orphan Wikidata entity can carry useful information that could enrich the entity with additional facts and relationships. Such pieces of information are unfortunately buried inside long articles without direct relevance to the main subject. Instead, we propose to establish a fine-grained mapping between Wikidata orphan entities and Wikipedia (sub-) sections.

To address this new task, we introduce ParaGraph – a novel approach for mapping KG entities onto paragraphs. The ParaGraph framework works in two stages. First, it selects candidate paragraphs by representing entities and paragraphs in the same vector space and measuring the cosine similarity between them for an initial candidate ranking. At this stage, we focus on generating an entity representation semantically similar to the natural language description of this entity. To achieve this, we propose a novel entity summarization model called **ENTSUMM**. ENTSUMM is built on top of BERT's encoder by stacking several inter-fact transformer layers to capture entity-level features for extracting facts. In a second phase, ParaGraph re-ranks the selected candidates and returns the top paragraphs as the most likely matches for a given entity. The two-stage architecture of ParaGraph ensures its scalability and efficiency even for large collections of text (for instance, we extracted more than 18 million paragraphs from the English Wikipedia alone).

To the best of our knowledge, we are the first to explore the problems of entity mapping and entity summarization using attention-based transformers. In summary, we make the following key contributions:

- We introduce the new task of *entity mapping*, which has a number of important applications and can advance the state of the art of many NLP tasks;
- We explore the potential of attention-based transformers for entity summarization in an unsupervised learning scenario and propose a novel approach based on language model encoders;
- We propose a new framework, ParaGraph, which efficiently maps Wikidata orphan entities onto Wikipedia sections;
- We release[5] an open dataset of links between Wikidata and Wikipedia English sections, which can be used to evaluate further entity mapping techniques.

The rest of the paper is organized as follows. In Section II we review approaches that are related to our task, particularly in the context of Entity Linking and entity-centric information retrieval. We introduce our problem definition and our two-stage architecture in Section III. Section IV describes the datasets we collected to evaluate our method and baselines. Finally, we conclude with a discussion on future work in Section V.

II. RELATED WORK

Our work lies at the crossroads of three research topics: *Entity Linking*, *Information Retrieval*, and *Entity Summarization*. In the following, we review the work done in these areas, and we cover subjects and systems closely related to our task.

a) Entity Linking: is the task of linking named entities mentioned in some text with their corresponding entities in a knowledge base. This task was first tackled using classical IR and NLP techniques [2] but recently benefited from advances in deep learning and language models. A number of recent works [3], [4] learn dedicated word embeddings for entities in that context, or reuse existing language models (like BERT) to guide the linking process [5]. A specific application of entity linking in the context of Wikipedia is often referred to as Wikification [6]–[8], i.e., cross-referencing documents with Wikipedia.

[2]For example, the English Wikipedia covers only 10% of Wikidata entities
[3]https://en.wikipedia.org/wiki/Racket_(sports_equipment)#Tennis
[4]https://www.wikidata.org/wiki/Q153362

[5]https://doi.org/10.5281/zenodo.7360787

One interesting direction in entity linking is *Entity Aspect Linking* which aims to link a named entity to a particular aspect of this entity [9]–[11]. Given an entity mention in some specific context, the goal is to link it to one from a set of predefined aspects that captures the addressed topic. Nanni et al. [9] suggest to derive a catalog of aspects from the top-level sections of the entity's Wikipedia article.

b) Entity Alignment: Our task is related to entity alignment, with the difference that we aim to map entities and paragraphs from two knowledge bases. In such tasks, the number of candidate matching pairs grows quadratically with the size of the KBs, making the alignment task intractable for datasets such as Wikidata. Entity alignment methods use blocking techniques [12], human-computation with probabilistic reasoning [13], [14], and entity embeddings [15]–[17].

c) Information Retrieval and Knowledge Graphs: Information Retrieval traditionally focuses on finding documents based on an information need formulated as a keyword query. In the past few years, the interplay between IR methods and KGs has received increasing attention [1], in particular by leveraging KGs for improving information retrieval techniques or by enriching KGs using IR techniques.

Entities taken from a KG can be leveraged within an IR system in order to help improve the understanding of a user's intent, queries, and documents beyond what can be achieved through word tokens on their own [1]. Dalton et al. [18] leverage entity-oriented information for query expansion to enrich the query with additional features from entities. Xiong and Callan [19] propose EsdRank – a document retrieval technique which treats entities from KGs as objects connecting query and documents. Their feature vectors are derived from the relationships between entities and documents, and another feature vector, which represents the relationship between the entity and the query. Xiong et al. [20] introduce Explicit Semantic Ranking (ESR), a ranking technique that leverages knowledge graph embeddings. ESR represents queries and documents by embeddings entities in the knowledge graph. The semantic relatedness between the representations of query and documents is then computed in the embedding space. Another approach incorporating KG embeddings is proposed in [21]: the authors model query and documents as word-based representations and entity-based representations simultaneously. They further consider four types of interaction derived from words and entities in the query and documents, and subsequently use them as features for ranking.

Typical KG-related tasks such as entity discovery, relation extraction, link prediction, or document filtering can also benefit from IR techniques. For instance, text classification and trend detection techniques from IR can be useful for discovering novel entities [22], [23]. Entity-centric document filtering determines whether a document contains an important piece of information about an entity. This task is of particular interest in the context of our paper. [1] classifies approaches to entity-centric document filtering into two types: *entity-dependent* and *entity-independent*. Entity-dependent approaches learn a single model for each entity and utilize text classification or language modeling techniques [24], [25]. In contrast, entity-independent methods do not learn the specifics of each entity directly and use generic features extracted from the entity and document pairs to perform the filtering [26].

d) Entity Summarization: is the task of computing an optimal compact summary for an entity by selecting a size-constrained subset of triples [27]. Existing methods are mainly unsupervised and leverage statistical and ontological features such as frequency, centrality, informativeness, or diversity to identify the most salient triples [28]–[31]. In recent years, a number of efforts have been made to use deep neural networks for entity summarization [32], [33].

The entity summarization problem is related to document summarization. These two tasks are fundamentally different: entities are represented by structured triples, whereas sentences in a document are unstructured text. However, some document summarization techniques could be adapted to entity summarization. In particular, the ENTSUMM model proposed in this paper was inspired by a text summarization approach described in [34].

e) Semantic Textual Similarity (STS): is an NLP task to quantitatively assess the semantic similarity between two text snippets. STS is one of the fundamental tasks in natural language processing. Recently, deep learning models based on transformer architectures [35], [36] have demonstrated state-of-the-art performance on the STS benchmark dataset [37].

Sentence-BERT (SBERT) [38] is a modification of the BERT network which is able to derive semantically meaningful sentence embeddings. SBERT architecture ensures its scalability and efficiency while achieving the state-of-the-art performance on the STS task. In this paper, we use SBERT in several scenarios, e.g., for finding most similar pairs of Wikipedia sections (Section III-A) and for measuring the semantic similarity between a Wikipedia section title and an entity label (Section III-B).

III. THE PARAGRAPH MODEL

This section introduces our proposed approach for entity mapping, which we refer to as ParaGraph. We start by formally introducing our problem and by giving an overview of our end-to-end pipeline, before introducing each of its components in more detail.

Problem Statement. We consider a bipartite graph $\mathcal{B}$ whose vertices consist of two disjoint subsets: $\mathcal{E}$, representing Wikidata entities together with their relationships, and $\mathcal{P}$, representing Wikipedia paragraphs[6]. Given an initial limited number of edges between $\mathcal{E}$ and $\mathcal{P}$, our goal is to correctly match vertices from $\mathcal{E}$ to $\mathcal{P}$. In other words, for a given Wikidata entity, we aim to find relevant paragraphs in Wikipedia. The Wikidata Knowledge Graph is given as a set $\mathcal{S}$ of triples (h, r, t) consisting of head and tail entities $h, t \in \mathcal{E}$ linked by a relationship $r \in \mathcal{R}$. Each entity or relationship can have a label and/or a description associated to it. A triple represents a relationship between two entities e.g., (Belgium, Capital_of,

[6]extracted from all English Wikipedia sections

Brussels). Here and throughout the paper, we use "triple" to refer to the data format, and "fact" to describe natural language form of the triple.

Paragraph Framework. To tackle this problem, we propose a two-stage mapping algorithm. Figure 2 depicts the overall architecture of our pipeline. The ParaGraph framework includes two modules: a *Semantic Search Module* and a *Ranking Module*. The semantic search module is a scalable component designed to retrieve relevant candidate paragraphs efficiently, while the ranking module is designed to identify entity-to-text relatedness using graph traversal. The framework takes an entity and a set of paragraphs as input. First, the semantic search model retrieves a candidate set for the given entity, effectively pruning the search space and producing an initial ranking. It works by embedding the paragraphs (regarded as documents $\mathcal{P}$) and the entities ($\mathcal{E}$) into the same vector space, measures the pairwise distance between the entity and the paragraphs, and returns the top-k paragraphs that are most similar to a given entity. Subsequently, the ranking model re-ranks the candidate paragraphs by leveraging additional information, i.e., entity labels, section titles, and relationships between a candidate paragraph and the article it belongs to, and finally outputs the paragraph that is deemed the most relevant to the entity.

A. Semantic Search Module

The main challenge of the semantic search stage is to retrieve paragraphs that are relevant to an input entity. In the following, we detail the main building blocks of our semantic search module. The semantic search module constructs an entity-summary from the entity data, and embeds both the entity-summary and all the sections in Wikipedia using a transformer based language model for semantic matching.

1) Textual Form of Entities (Lexicalization): Our semantic search task can be approached as a *semantic textual similarity* problem. For example, one can represent the entity as some text and use a unified language model for encoding both entity descriptions and paragraphs in the same vector space. We follow this approach by relying on the textual information attached to an entity in the form of labels, descriptions, properties, and related entities' labels. This approach is viable as we observe that 91% of Wikidata entities have at least an English description and/or a label associated with them. However, descriptions in Wikidata tend to be very succinct – for example, the average length of English descriptions is 4.7 words. Luckily, entities have additional information attached as facts representing relationships with other entities in the knowledge graph.

Concretely, to convert an entity s into a textual form str_s, we create a string composed of the entity label and description str_s = "$< s.label >$ *is a* $< s.description >$.", then concatenate it to all facts from triples (s, p, o) as str_s += "$< p.label > < o.label >$." (see Figure 2 for an example).

2) Textual Summarization of Entities: In the previous step, we built a textual form of an entity without imposing any order or importance to its facts. However, while facts provide valuable information about an entity, not all of them are equally useful with respect to our problem.

We aim to generate an entity vector representation that is as close as possible to a description of this entity in natural language in order to find the best paragraph matches. To select the most informative and representative facts, we adopt a text summarization approach inspired by recent advances in language modeling using attention-based transformers. In the following, we briefly introduce key notions from [34] that proposes a document-level encoder based on BERT called BERTSUMEXT, and describe it in the context of our entity summarization procedure.

Figure 3 (upper left panel) shows the components of the BERTSUMEXT architecture. An input *textual form of an entity* is first preprocessed by inserting two special tokens. [CLS] is prepended to the beginning of each fact; this token collects features for the fact it precedes. Token [SEP] is inserted after each fact as an indicator of fact boundaries. Each fact is transformed into a sequence of individual word tokens (for simplicity, we represent each fact with a single token in the figure.) Next, each token is assigned three kinds of embeddings: (i) **Token Embeddings** indicate the meaning of each token using pre-trained Word2Vec [39], (ii) **Segmentation Embeddings** are used to distinguish between two adjacent facts: embedding E_A or E_B is assigned to $fact_i$ depending on whether i is odd or even, and (iii) **Position Embeddings** indicate the position of each token within the text sequence.

These three embeddings are added and then fed to a pre-trained BERT model. The output vectors T_{CLS_i}, which correspond to the i-th [CLS] token can be used as the representation for $fact_i$. Several inter-fact transformer layers are then stacked on top of BERT outputs, to capture entity-level features for extracting summaries. The final output layer is a sigmoid classifier, which assigns a relevance weight to each fact.

3) Semantic Search: To perform the semantic search, we propose ENTSUMM, a model that learns an entity summary representation which is aligned with relevant paragraphs. The learning objective of our model is to generate an entity representation which is as close as possible in a vector space to the representation of the paragraph referring to the same entity.

Our semantic search module (fully illustrated in Figure 3) proceeds in the following steps:

1) The Entity Summarization module takes the entity textual form and uses BERTSUMEXT to compute the relevance score for each individual $fact_i$. It also produces a fixed-sized vector representation for each fact T_{CLS_i};
2) In parallel, the Paragraph Representation module feeds a paragraph to a separate pre-trained BERT model and the output vector, corresponding to the T_{CLS} token, is used as the representation for this paragraph;
3) The Entity Summary Representation module ENTSUMM computes the weighted sum of all facts, multiplying their vector representations by their corresponding scores

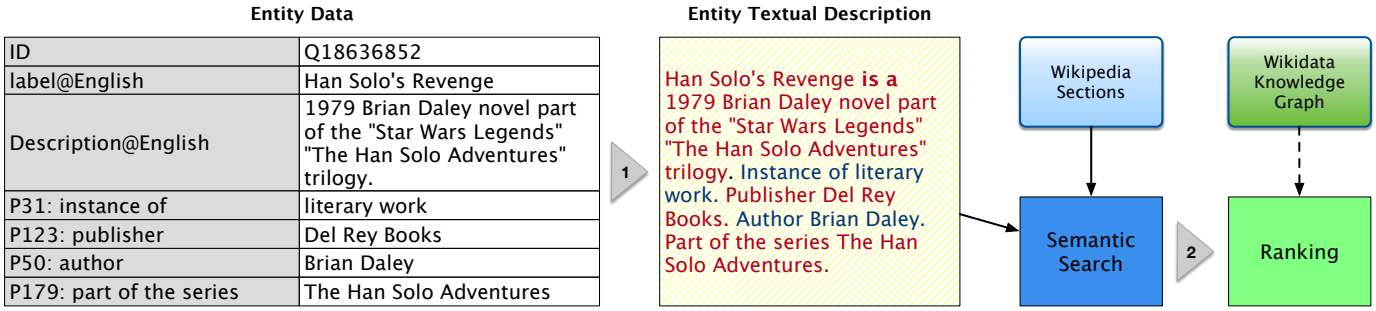


Fig. 2. The ParaGraph architecture. Our framework has two main stages. For a given entity with sufficient data, (1) *Semantic Search Module* generates its textual representation and identifies candidate paragraphs, while (2) the *Ranking Module* uses additional features extracted from Wikidata for candidate ranking.

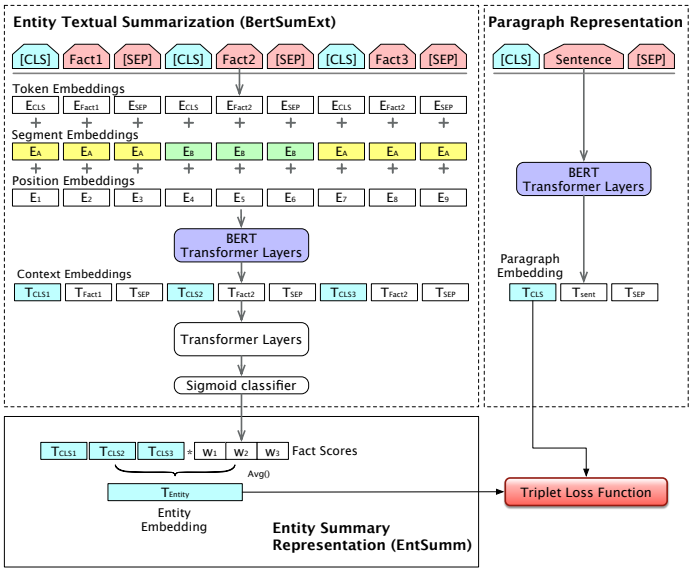


Fig. 3. The architecture of EntSumm. Our entity summarization component is based on BertSumExt [34]. The entity is first lexicalized by concatenating the label, the description, and the facts. The transfer is fine-tuned to the task of paragraph matching.

computed above. The output vector T_{entity} represents the final entity embedding;

4) Finally, the cosine similarity between the paragraph vector and its corresponding entity vector is calculated. The learning objective of the model is to minimize the distance between those two vectors.

Training. We train EntSumm on a dataset consisting of pairs of Wikipedia abstracts[7] and their corresponding Wikidata entities. The intuition behind this choice is as follows. The goal of the semantic search step is to encode an entity representation and a corresponding paragraph so that their vectors were as close as possible in the vector space. Train such an encoder requires a dataset of *(paragraph, entity)* pairs, where *paragraph* is a textual description of *entity*. A Wikipedia abstract is an introduction to the article and a summary of its most important contents[8] and hence abstracts can serve as a benchmark for our task. As loss function, we adopt triplet loss, which simultaneously pushes an input (entity representation) E closer to a positive example P and further from a negative example N (P, N are paragraph representations). The exact loss is given by the following formula:

$$\mathcal{L}(E,P,N) = max(||f(E)-f(P)||^2-||f(E)-f(N)||^2+\alpha,0) \quad (1)$$

where α is the margin between positive and negative pairs.

For efficiently training a model with triplet loss, negative examples N should be sufficiently similar to positive examples P, otherwise minimizing the loss function becomes trivial. More formally, a useful negative example should satisfy the following distance property: $d(E,N) \approx d(E,P)$. To obtain such negative examples for our task, we compute the pairwise semantic similarity of all paragraphs in the dataset and for each positive paragraph we randomly select a negative example from the top-10 most similar paragraphs. We used a pre-trained SBERT model [38] to compute those semantic similarities.

B. Ranking Module

To refine the results of the previous step, we subsequently re-rank the candidate paragraphs selected by our semantic search model. To achieve this, we devise the following set of features.

1) Similarity Score: Once trained, the semantic search module described above can efficiently compute a **similarity score** between an input entity, and all paragraphs in $\mathcal{P}$. While this model can efficiently achieve a high recall, the resulting candidate ranking at the top is sub-optimal. Upon further inspection, we observed that even if a paragraph is semantically similar to an entity, i.e. mentions all important facts, it does not guarantee that the paragraph is a summary of this entity. For example, the entity Q74950 (*Objectivism*) is in our case mapped to the paragraph *Objectivist movement* of article *Ayn Rand*. Although this paragraph contains all the necessary facts, it does not describe objectivism itself but rather Ayn Rand's role in it.

[7] A Wikipedia abstract is an introduction to the article and a summary of its most important contents.

[8] https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Lead_section

2) Path Relevance Score: To deal with the above issue, it is essential to understand the relationships between an entity and the article a candidate paragraph belongs to. In the above example, entity Q74950 (*Objectivism*) is connected to entity Q132524 (*Ayn Rand*) by the *founded by* relation. It might be a strong signal that none of *Ayn Rand* paragraphs contains a summary of *objectivism*. On the other hand, if an article and an entity are connected by a *has part* relation, there is a good chance that one of the article's paragraphs can be mapped to this entity. The intuition here is that the relationship chain between the input entity and the main article's entity carries an important signal about the candidate paragraph.

To predict which articles most probably contain a relevant paragraph, we train a simple Naive Bayes path classifier. Given an input entity e and an article's entity a, we first compute the path in the Wikidata graph between them. Each edge in the path is encoded as one hot vector of dimension $|\mathcal{R}|$ (the size of the relations vocabulary). The sum of path edges is then fed to the classifier, which outputs the probability that one of a's paragraphs contains the summary of entity e. To consider both incoming and outgoing edges, we duplicate each edge in the graph reversing its direction and assigning it label $\langle r \rangle\_reverse$.

3) Section Title to Label Score: Our path classifier predicts which articles contain relevant paragraph, but it does not help when the semantic search model returns several paragraphs belonging to the same article. In that case, we leverage the semantic similarity between an entity label and a section title as an additional feature. Effectively we compute the sentence similarity between the input entity label and the candidate paragraph's section title using a pre-trained SBERT model.

Learning to Rank. To produce the final ranking score, we use a popular Learning To Rank model, LambdaRank [40]. Given an input entity and a set of candidate paragraphs, LambdaRank ranks paragraphs w.r.t. an entity based on the features introduces earlier in this section, namely:

- **SimScore**: The semantic search score, which indicates how well a candidate paragraph summarizes the entity;
- **PathScore**: The path classifier score, which indicates the probability of a given article containing a relevant paragraph;
- **TitleScore**: The semantic similarity score between a section title and an input entity label.

IV. Experiments and Results

In this section, we report on a set of experiments we have conducted to evaluate the performance of ParaGraph. In the following, we first introduce our dataset, and then describe our experimental protocol for evaluating both components of our framework, i.e. our semantic search model and our ranking model.

A. Data Collection

Our efforts are specifically aimed at establishing a fine-grained mapping between Wikidata entities and Wikipedia sections. Currently, Wikidata is integrated with Wikipedia at the article level, i.e. every Wikipedia article has a corresponding entity in Wikidata. At this point, Wikidata does not support using anchors links as sitelinks, i.e. linking to a specific section of a page. For that reason, generating a dataset containing *(Wikidata entity, Wikipedia section)* pairs is a challenge of its own. We collected training examples from 3 different sources:

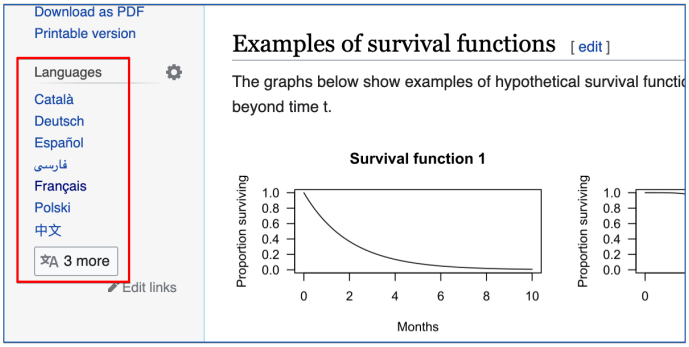


Fig. 4. A list of Wikidata sitelinks on a English Wikipedia page about Survival Analysis Functions. The cross-lingual French sitelink points to a subsection in a more general article[10] highlighting the difference in treatment of entities across languages.

1) Mappings between Wikidata entities and Wikipedia articles. In this case, we consider the abstract of an article (a block of text between the title and the table of content) as a paragraph describing an entity;
2) Interlingual Wikipedia. For example, the Wikidata entity Q2915096 contains a sitelink to the English Wikipedia page *Survival_function*, and all the other Wikidata sitelinks are listed on this page in the left column (Figure 4). A link to a section can be added to this list and thus can be mapped to the source Wikidata entity, as is the case for the French language;
3) Links between sections and full articles. When a section is a summary of another article, a link to that article appears immediately under the section heading (Figure 5). The article, in turn, is mapped to a Wikidata entity, thus bridging this entity with the source section.

We used mappings between Wikidata entities and Wikipedia **abstracts** (1) to train and evaluate the Semantic Search Model (Section III-A). To evaluate the performance of the Semantic Search Model on sections as well as the overall performance of our framework, we collected mappings between Wikidata entities and Wikipedia **sections** as described in (2) and (3).

Although the data collection methods described above mostly bring true positive examples, the test set still contains a considerable amount of false positive pairs. For example, an article *Ayn Rand* contains a section *Objectivist movement*, which links to the corresponding full article. However, this section is not a summary of *Objectivist movement* but rather describes *Ayn Rand*'s role in it and hence should not be matched with *Objectivism* entity. These errors in the dataset affects evaluation results, and we revisit this issue in Section IV-D.

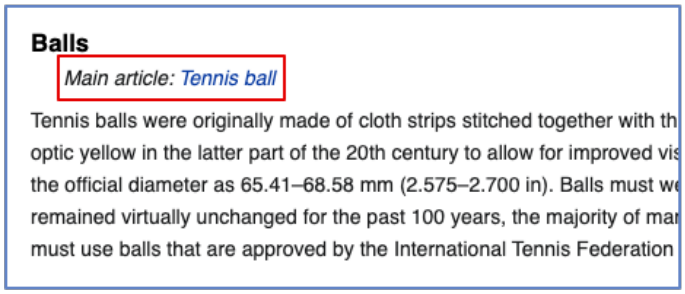


Fig. 5. Section "Balls" of a Wikipedia article "Tennis" with a link to an article "Tennis ball"

B. Semantic Search Evaluation

In this subsection, we conduct a series of experiments to evaluate the Semantic Search component of ParaGraph. For this component of our architecture, we aim to answer the following questions:

- **Q1:** How effective is our entity summarization when compared to other entity representation methods?
- **Q2:** How effective is our method in mapping textual entity representations onto Wikipedia **abstracts**?
- **Q3:** How effective is our method in mapping textual entity representations onto Wikipedia **sections**?
- **Q4:** How effective is EntSumm for the entity summarization task?

Evaluation Protocol. Our evaluation is performed on three test subsets:

1) Entity2Abstract: contains 100'000 *(entity, abstract)* pairs and a paragraph pool of 1 million abstracts;
2) Entity2Section: contains 41'000 *(entity, section)* pairs and a paragraph pool of 306'000 sections from 26'000 articles;
3) RichEntity2Section: contains 9'700 *(entity, section)* pairs, where each entity is represented by at least 10 facts. The paragraph pool is the same as in the Entity2Section dataset;

At evaluation time, for each entity from the test set we search the most similar paragraph from the paragraph pool. The objective of the semantic search step is to select candidates for further ranking, therefore our main metric at this stage is Hits@K, i.e., the proportion of correct documents in the top-K ranked documents. Higher Hits@K values indicate that more relevant paragraphs get into the candidate subset. In our experiments, we report values of Hits@1, Hits@10 and Hits@50 metrics.

Experimental Design and Baselines. As a baseline for our method, we approach the semantic search task as a a Semantic Textual Similarity (STS) problem. We conduct a series of experiments where an entity is represented as a single chunk of text. In this case, our training process is equivalent to fine-tuning a BERT model for our specific task, which is mapping textual descriptions of entities onto relevant paragraphs. We note that the structure of two pieces of text we compare is considerably different: on one hand, there is a paragraph written in natural language, composed of relatively long coherent sentences; on the other hand, the second textual description of the entity is a bag of short, synthetic phrases (facts) only vaguely similar to natural language text. With this in mind, in our baseline approach we fine-tune two independent BERT models to encode separately a paragraph and a textual entity representation. Effectively, we replace EntSumm with a BERT model (**Q1**). We experiment with three entity lexicalization strategies:

1) BaselineLabel: In this dataset, we use the label of the entity only.
2) BaselineLabelDescr: In this dataset, we uses a combination of the label and description of the entity.
3) BaselineFacts: In this dataset, we use the full entity textual form (composed by our lexicalization mechanism introduced in section III-A1).

We further evaluate our baseline models and EntSumm on two test sets: Entity2Abstract and Entity2Section (**Q2** and **Q3**).

Finally, to investigate how effective our entity summarization method is compared to representing an entity with all available facts (**Q4**), we evaluate the EntSumm and BaselineFacts models on the RichEntity2Section test set. We evaluate this aspect on a separate test set because Entity2Section test set used in the previous series of experiments contains insufficient data to demonstrate the difference between the two models. More specifically, entities in Entity2Section are on average associated with 10 facts only, while in RichEntity2Section – with 20 facts. The performance of an entity summarization model is more evident for bigger entities, represented by a substantial number of facts.

Results. Table I gives the results of our semantic search approach evaluated on the task of mapping entities onto Wikipedia abstracts. These results demonstrate that our proposed method to represent an entity with its textual (Wikidata label and Wikidata description) and/or contextual (facts) information is effective for aligning entities and their textual counterparts in the vector space. As expected, Wikidata labels alone perform worse than other entity representation strategies. However, a combination of label and description reaches over 98% precision at Hits@10, and adding further contextual information does not improve this result significantly.

To evaluate our approach on the target task, i.e., mapping entities onto Wikipedia sections, we run the same set of experiments on the Entity2Section test set. Table II summarizes the results. A considerable decrease in performance compared to the previous results can be explained by several factors:

- The Entity2Abstract dataset contains only Wikipedia abstracts, where each abstract is a summary of exactly one entity. Sections, on the other hands, can cover different aspects of different entities, and

[10]https://fr.wikipedia.org/wiki/Analys_de_survie#Fonction_de_survie

TABLE I
PERFORMANCE OF THE SEMANTIC SEARCH COMPONENT (ENTSUMM) ON ENTITY2ABSTRACT DATASET

	Hits@1	Hits@10	Hits@50
BASELINELABEL	85.17%	96.20%	97.95%
BASELINELABELDESCR	93.97%	98.38%	99.15%
BASELINEFACTS	94.54%	98.90%	99.43%
ENTSUMM	**95.61%**	**99.14%**	**99.58%**

TABLE II
PERFORMANCE OF THE SEMANTIC SEARCH COMPONENT (ENTSUMM) ON ENTITY2SECTION DATASET

	Hits@1	Hits@10	Hits@50
BASELINELABEL	39.73%	70.76%	82.16%
BASELINELABELDESCR	46.81%	76.42%	86.61%
BASELINEFACTS	47.40%	**80.02%**	90.28%
ENTSUMM	**48.79%**	**80.02%**	**90.75%**

thus the ENTITY2SECTION dataset is inherently more ambiguous.

- As discussed in Section IV-A, the ENTITY2SECTION test set is generated automatically and contains a substantial amount of errors. We estimate the impact of this factor below.

Nevertheless, BASELINEFACTS and ENTSUMM models achieve 90% precision at Hits@50, which indicates that for 90% entities, correct paragraphs are ranked in the top-50 paragraphs. Since top paragraphs are further re-ranked at the next stage of our pipeline, we consider these results as satisfactory.

Consistent with previous results, the BASELINELABEL model performs worse than other models. It falls in line with our earlier hypothesis that labels alone do not convey enough information to represent an entity for our task. Moreover, Table II demonstrates that representing entity with its facts (BASELINEFACTS and ENTSUMM) results in better performance compared to label plus description representation (BASELINELABELDESCR). The difference is more significant at Hits@10 and Hits@50: it indicates that BASELINELABELDESCR can handle the cases when a target paragraph is clearly distinguishable from other paragraphs and can be ranked with confidence at top-1, but fails when there are several similar candidate paragraphs and the target paragraph cannot be detected unambiguously.

From these results, we observe that the difference between BASELINEFACTS and ENTSUMM is insignificant. Our assumption is that for entities with few facts ENTSUMM cannot select the most representative ones but rather outputs all available facts, which is equivalent to BASELINEFACTS. To investigate this, we created a test set which contained only entities with at least 10 facts and re-evaluated both BASELINEFACTS and ENTSUMM on it. Results in Table III confirm our assumption: on rich entities, the advantage of ENTSUMM over the baseline is more evident.

TABLE III
PERFORMANCE OF THE SEMANTIC SEARCH COMPONENT (ENTSUMM) ON RICHENTITY2SECTION DATASET

	Hits@1	Hits@10	Hits@50
BASELINEFACTS	51.57%	81.43%	90.82%
ENTSUMM	**57.17%**	**83.75%**	**92.10%**

TABLE IV
PERFORMANCE OF PARAGRAPH WITH DIFFERENT RANKING STRATEGIES ON ENTITY2SECTION

	Hits@1	Hits@10
SimScore	48.79%	80.02%
TitleScore	57.95%	80.06%
PathScore	31.08%	59.87%
SimScore + PathScore	50.72%	83.47%
SimScore + TitleScore + PathScore	**70.27%**	**87.34%**

C. Ranking Evaluation

In this subsection, we evaluate the Ranking component of ParaGraph. For this component of our architecture, we focus on the following questions:

- **Q1:** Is the path in the KG between a given entity and the entity corresponding to the main article of a section a useful feature for ranking the candidates?
- **Q2:** How do the different components of our ranking model affect the final performance of our pipeline?

Evaluation Protocol. Evaluation is performed on the ENTITY2SECTION test set described in Section IV-B. As ranking is the final stage of our pipeline, we are primarily interested in Hits@1 and Hits@10 metrics to evaluate the overall performance of our approach.

Experimental Design. To answer questions **Q1** and **Q2**, in this series of experiments we conduct an ablation study of different components of the ranking model. We proceed in the following steps:

1) We apply our semantic search model ENTSUMM to select a pool of candidate paragraphs. Based on the evaluation results of the semantic search model, we set the size of the candidate pool $k = 50$.
2) For each candidate from the pool we compute three scores as discussed in Section III-B, i.e., semantic similarity score (SimScore), path relatedness score (PathScore), and section title similarity score (TitleScore).
3) We re-rank the candidates according to different combinations of these scores. For evaluation, we assume that if a target paragraph does not get into a candidate pool, the match for an entity is not found.

Results. The results of the ablation study are reported in Table IV. The first line of the table corresponds to the results of the semantic search stage, i.e., candidates are originally ranked according to the semantic search score alone.

Surprisingly, re-ranking candidates by $TitleScore$ score alone and not considering $SimScore$ score results in better

TABLE V
PERFORMANCE OF PARAGRAPH ON GOLDENENTITY2SECTION DATASET.

	Hits@1	Hits@10	Hits@50
Semantic Search	74.43%	94.83%	**98.13%**
Semantic Search + Ranking	**81.66%**	**97.50%**	**98.13%**

TABLE VI
COMPARISON OF PARAGRAPH AND IR BASELINES. (I): QUERY = LABEL. (II): QUERY = LABEL + DESCRIPTION

	Hits@1	Hits@10	Hits@50
BM25 (i)	46.94%	85.60%	**92.36%**
BM25 (ii)	43.10%	83.30%	91.75%
ColBERT (i)	41.95%	77.27%	90.46%
ColBERT (ii)	46.87%	80.12%	92.18%
ANCE (i)	39.25%	68.92%	82.11%
ANCE (ii)	41.61%	72.84%	85.78%
ParaGraph	**70.27%**	**87.34%**	90.75%

Hits@1, while Hits@10 remains the same. The paragraph's section title is a clear and unambiguous feature when it matches with the entity label, unfortunately, this information is not always available.

On the other hands, re-ranking candidates considering $PathScore$ alone considerably degrades the performance compared to the semantic search results. The reason is that sections from the same article are mostly ranked next to each other w.r.t. the query entity and the path score between the entity and the article is unable to distinguish them. This observation also explains why adding $PathScore$ to $SimScore$ only insignificantly affects the ranking performance.

The combination of all three scores results in the best performance, particularly at Hits@1. This demonstrates the effectiveness of using a learning-to-rank technique and proves the usefulness of considering the relationships between a candidate section and the main article.

D. Evaluation on Manually Labeled Data

As mentioned earlier, the automatically generated ENTITY2SECTION test set contains a substantial amount of errors. To factor out the potential impact of dataset errors on the evaluation results, we created GOLDENENTITY2SECTION, a dataset that contains 750 *(entity, section)* pairs which we labeled manually. We re-evaluate ENTSUMM and the overall performance of ParaGraph on this dataset (Table V). As expected, the performance of Paragraph on the manually labeled data is consistent with our previous results – the ranking complements the semantic search. More importantly, the results indicate that some of the slightly lower results achieved above can be partially attributed to the noise in the dataset, but does not jeopardize our approach and training process overall.

E. IR Baseline

As discussed in Section I, *entity mapping* can be viewed as a specific IR task, defined by taking an entity as input query and Wikipedia paragraphs as the document collection. However, we argue that this specific task should be treated as a new task in regard to the content of the target documents, which substantially differs from standard IR tasks. To demonstrate that the problem addressed in this paper does not have a straightforward IR solution, we compare the performance of ParaGraph to the performance of BM25 as well as two state-of-the-art neural IR models: ColBERT [41] and ANCE [42]. We took advantage of the Anserini toolkit for reproducible information retrieval research[11] to obtain these models.

Our evaluation was performed on the ENTITY2SECTION test set described in Section IV-B. We experimented with two entity textual forms as query: the label of the entity only, and a combination of the label and description of the entity. We note that we tried augmenting the query with a bag of facts, however, the results degraded significantly. The results of these experiments are reported in Table VI.

Our proposed model, ParaGraph, consistently outperforms BM25 and neural rankers on the task of *entity mapping*, which highlights the high effectiveness of our solution for the task at hand. The performance of the IR baselines is comparable to the performance of ENTSUMM (the semantic search module of ParaGraph). This emphasizes the importance of the re-ranking step and shows that *entity mapping* is significantly different from a standard information retrieval task: not all relevant documents retrieved by an IR model are suitable to represent the **summary** of an entity given as a query.

V. CONCLUSION AND FUTURE WORK

In this work, we introduced the task of *entity mapping*, which consists in establishing fine-grained mappings between entities and paragraphs describing them. We focus on a use-case involving Wikipedia and the Wikidata knowledge base as they are both prominent and tightly interrelated already. Our proposed system, ParaGraph, operates in a two-stage manner. First, we generate an entity summary and align it with paragraphs to reduce the search space. The entity summarization model trained in an unsupervised manner generates an entity representation in textual form. Second, ParaGraph re-ranks the selected candidates leveraging additional context information, i.e., we extract semantic similarities and relationships between the input entity and the parent article.

In our experiments, we show that ParaGraph is able to map knowledge graph entities onto text paragraphs with high accuracy, effectively achieving 86% Hits@10. In particular, we show that our framework is well suited for mapping tail entities with few properties which are not notable enough to have a dedicated article but might have a full subsection in a Wikipedia article. We also demonstrate that although *entity mapping* can be viewed as an IR problem, a traditional IR solution performs considerably worse on this task.

[11] http://anserini.io

In future work, we plan on extending our experiments in several directions. First, we want to explore the idea of unsupervised entity summarization and generate a textual summary in addition to the vector representation. The next step for the *entity mapping* task could be to extend it beyond the Wikidata-Wikipedia scenario and to develop a generalized model for mapping an entity to any collection of paragraphs. It would also be interesting to investigate how well the proposed approach adapt to other KGs. Finally, our task's definition could be extended to *fact mapping*, whereby individual triples could mapped to paragraphs or sentences.

REFERENCES

[1] R. Reinanda, E. Meij, and M. de Rijke, "Knowledge graphs: An information retrieval perspective," *Found. Trends Inf. Retr.*, vol. 14, no. 4, pp. 289–444, 2020.

[2] W. Shen, J. Wang, and J. Han, "Entity linking with a knowledge base: Issues, techniques, and solutions," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 2, pp. 443–460, 2015.

[3] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Joint learning of the embedding of words and entities for named entity disambiguation," in *Proc. SIGNLL Conf. Comput. Natural Language Learn.*, 2016, pp. 250–259.

[4] O. Ganea and T. Hofmann, "Deep joint entity disambiguation with local neural attention," in *Proc. of EMNLP Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2619–2629.

[5] S. Chen, J. Wang, F. Jiang, and C. Lin, "Improving entity linking by modeling latent entity type information," in *Proc. of AAAI Conf. on Artif. Intell.*, 2020, pp. 7529–7537.

[6] R. Mihalcea and A. Csomai, in *Proc. of 16th CIKM ACM Conf. on Inf. and Knowl. Management*, 2007, pp. 233–242.

[7] R. West, D. Precup, and J. Pineau, "Completing wikipedia's hyperlink structure through dimensionality reduction," in *CIKM*. ACM, 2009, pp. 1097–1106.

[8] M. Gerlach, M. Miller, R. Ho, K. Harlan, and D. Difallah, "Multilingual entity linking system for wikipedia with a machine-in-the-loop approach," in *CIKM*. ACM, 2021, pp. 3818–3827.

[9] F. Nanni, S. P. Ponzetto, and L. Dietz, "Entity-aspect linking: Providing fine-grained semantics of entities in context," in *Proc. of 18th ACM/IEEE JCDL Joint Conf. on Digital Libraries*, 2018, pp. 49–58.

[10] S. Chatterjee and L. Dietz, "Entity retrieval using fine-grained entity aspects," in *SIGIR*. ACM, 2021, pp. 1662–1666.

[11] J. Ramsdell and L. Dietz, "A large test collection for entity aspect linking," in *Proc. of 29th ACM CIKM Conf. on Inf. and Knowl. Management*, 2020, pp. 3109–3116.

[12] G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, and W. Nejdl, "Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data," in *WSDM*. ACM, 2012, pp. 53–62.

[13] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," *PVLDB*, vol. 5, no. 11, pp. 1483–1494, 2012.

[14] G. Demartini, D. Difallah, and P. Cudré-Mauroux, "Large-scale linked data integration using probabilistic reasoning and crowdsourcing," *VLDB J.*, vol. 22, no. 5, pp. 665–687, 2013.

[15] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, "Multilingual knowledge graph embeddings for cross-lingual knowledge alignment," in *IJCAI*. ijcai.org, 2017, pp. 1511–1517.

[16] M. Luggen, J. Audiffren, D. Difallah, and P. Cudré-Mauroux, "Wiki2prop: A multimodal approach for predicting wikidata properties from wikipedia," in *WWW*. ACM / IW3C2, 2021, pp. 2357–2366.

[17] A. Assi, H. Mcheick, A. Karawash, and W. Dhifli, "Context-aware instance matching through graph embedding in lexical semantic space," *Knowl. Based Syst.*, vol. 186, 2019.

[18] J. Dalton, J. Allan, and D. A. Smith, "Passage retrieval for incorporating global evidence in sequence labeling," in *Proc. of 20th ACM CIKM Conf. on Inf. and Knowl. Management*, 2011, pp. 355–364.

[19] C. Xiong and J. Callan, "Esdrank: Connecting query and documents through external semi-structured data," in *Proc. of 24th ACM CIKM Conf. on Inf. and Knowl. Management*, 2015, pp. 951–960.

[20] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proc. of 26th WWW Int. Conf. on World Wide Web*, 2017, pp. 1271–1279.

[21] C. Xiong, J. Callan, and T. Liu, "Word-entity duet representations for document ranking," in *Proc. of 40th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2017, pp. 763–772.

[22] T. Lin, Mausam, and O. Etzioni, "No noun phrase left behind: Detecting and typing unlinkable entities," in *Proc. of EMNLP-CoNLL Joint Conf. on Empirical Methods Natural Lang. Process. and Comput. Natural Lang. Learn.*, 2012, pp. 893–903.

[23] J. Hoffart, Y. Altun, and G. Weikum, "Discovering emerging entities with ambiguous names," in *Proc. of 26th WWW Int. Conf. on World Wide Web*, 2014, pp. 385–396.

[24] X. Liu, J. Darko, and H. Fang, "A related entity based approach for knowledge base acceleration," in *Proc. of 22nd TREC Text REtrieval Conference*, 2013.

[25] J. Wang, D. Song, Q. Wang, Z. Zhang, L. Si, L. Liao, and C. Lin, "An entity class-dependent discriminative mixture model for cumulative citation recommendation," in *Proc. of 38th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2015, pp. 635–644.

[26] L. Bonnefoy, V. Bouvier, and P. Bellot, "A weakly-supervised detection of entity central documents in a stream," in *SIGIR*. ACM, 2013, pp. 769–772.

[27] Q. Liu, G. Cheng, K. Gunaratna, and Y. Qu, "Entity summarization: State of the art and future challenges," *J. Web Semant.*, vol. 69, p. 100647, 2021.

[28] S. A. Pouriyeh, M. Allahyari, K. J. Kochut, G. Cheng, and H. R. Arabnia, "ES-LDA: entity summarization using knowledge-based topic modeling," in *Proc. of 8th IJCNLP Int. Joint Conf. on Natural Lang. Process.*, 2017, pp. 316–325.

[29] K. Gunaratna, K. Thirunarayan, and A. P. Sheth, "FACES: diversity-aware entity summarization using incremental hierarchical conceptual clustering," in *AAAI*. AAAI Press, 2015, pp. 116–122.

[30] A. Thalhammer, N. Lasierra, and A. Rettinger, "Linksum: Using link analysis to summarize entity data," in *Proc. of 16th ICWE Int. Conf. on Web Eng.*, 2016, pp. 244–261.

[31] G. Cheng, T. Tran, and Y. Qu, "RELIN: relatedness and informativeness-based centrality for entity summarization," in *Proc. of 10th ISWC Int. Semantic Web Conf.*, 2011, pp. 114–129.

[32] D. Wei and Y. Liu, "ESA: entity summarization with attention," *CoRR*, vol. abs/1905.10625, 2019.

[33] Q. Liu, G. Cheng, and Y. Qu, "Deeplens: Deep learning for entity summarization," *CoRR*, vol. abs/2003.03736, 2020.

[34] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," in *Proc. of EMNLP-IJCNLP 2019 Conf. on Empirical Methods in Natural Lang. Process. and 9th Int. Joint Conf. on Natural Lang. Process.*, 2019, pp. 3728–3738.

[35] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[36] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019.

[37] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proc. of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP*, 2018, pp. 353–355.

[38] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *EMNLP/IJCNLP (1)*. Association for Computational Linguistics, 2019, pp. 3980–3990.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013, pp. 3111–3119.

[40] C. J. C. Burges, R. Ragno, and Q. V. Le, "Learning to rank with nonsmooth cost functions," in *Proc. of NIPS Advances in Neural Info. Process. Systems*, 2006, pp. 193–200.

[41] O. Khattab and M. Zaharia, "Colbert: Efficient and effective passage search via contextualized late interaction over BERT," in *SIGIR*. ACM, 2020, pp. 39–48.

[42] L. Xiong, C. Xiong, Y. Li, K. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk, "Approximate nearest neighbor negative contrastive learning for dense text retrieval," in *Proc. of 9th ICLR Int. Conf. on Learning Representations*, 2021.