Entity-aware Transformers for Entity Search

Emma J. Gerritse
Radboud University
emma.gerritse@ru.nl

Faegheh Hasibi
Radboud University
f.hasibi@cs.ru.nl

Arjen P. de Vries
Radboud University
a.devries@cs.ru.nl

ABSTRACT

Pre-trained language models such as BERT have been a key ingredient to achieve state-of-the-art results on a variety of tasks in natural language processing and, more recently, also in information retrieval. Recent research even claims that BERT is able to capture factual knowledge about entity relations and properties, the information that is commonly obtained from knowledge graphs. This paper investigates the following question: Do BERT-based entity retrieval models benefit from additional entity information stored in knowledge graphs? To address this research question, we map entity embeddings into the same input space as a pre-trained BERT model and inject these entity embeddings into the BERT model. This entity-enriched language model is then employed on the entity retrieval task. We show that the entity-enriched BERT model improves effectiveness on entity-oriented queries over a regular BERT model, establishing a new state-of-the-art result for the entity retrieval task, with substantial improvements for complex natural language queries and queries requesting a list of entities with a certain property. Additionally, we show that the entity information provided by our entity-enriched model particularly helps queries related to less popular entities. Last, we observe empirically that the entity-enriched BERT models enable fine-tuning on limited training data, which otherwise would not be feasible due to the known instabilities of BERT in few-sample fine-tuning, thereby contributing to data-efficient training of BERT for entity search.

CCS CONCEPTS

• **Information systems** → **Language models**.

KEYWORDS

Entity retrieval, transformers, BERT, entity embeddings

1 INTRODUCTION

Pre-trained language models (LMs) such as BERT [11] and its successors [27, 38, 52] learn rich contextual information about words from large-scale unstructured corpora and have achieved intriguing results on a variety of downstream tasks in natural language

processing (NLP) [11, 25, 26] and information retrieval (IR) [32, 46]. It is even shown that these language models have the capability of capturing a tremendous amount of world knowledge, including information about real-world entities otherwise found in knowledge graphs (KGs) [35, 42]. For example, language models can predict masked objects in cloze sentences such as "The native language of Mammootty is _____" and "The Sharon Cuneta Show was created in _____", where each of them demonstrates a ⟨`subject`, `relation`, `object`⟩ triple in a knowledge graph.

Language models, however, fail to perform complex reasoning about entities, as they cannot capture sparse world facts [23, 35, 39]. In information retrieval systems, users ask complex queries about entities, such as "What is the second highest mountain in the world?" or "Most famous civic-military airports." Answering such queries requires leveraging rich human curated information about entities in the knowledge graphs. To bridge the gap between LMs and KGs, recent works enhance language models with rich structured entity information from knowledge graphs, showing superior performance for knowledge-driven NLP tasks such as relation classification, entity typing, and cloze-style question answering [23, 34, 37, 43, 45, 50, 54]. Despite this success, no previous study has examined the effect of entity-enriched language models for answering entity-oriented queries in IR.

This work explores the benefit of enriching BERT-based retrieval models with auxiliary entity information from KGs for the *entity retrieval* task, where users' queries are better answered with entities rather than passages or documents [2]. It is shown that entity information improves performance of document and entity retrieval tasks [9, 15, 18, 47, 48]. Yet, these studies were performed using traditional retrieval models, without utilizing LMs such as BERT.

The development of BERT-based models for entity retrieval faces a major challenge: there is limited training data for the entity retrieval task, and fine-tuning of BERT in data-constraint regimes leads to instabilities in model performance [13, 36, 53]. More specifically, the official entity retrieval dataset, DBpedia-Entity v2 [21], contains 467 queries; considered a small dataset for fine-tuning of a large neural language model. When fine-tuning BERT (especially its large variant) on a target task, the training data should be large enough that the model parameters get close to the ideal setting of the target task, or otherwise it causes forgetting of what the model has already learned.

Against this background, our first and primary research question is ***RQ1: Can an entity-enriched BERT-based retrieval model improve the performance of entity retrieval?*** To address this question, we propose an entity-enriched BERT-based retrieval model, EM-BERT, where factual entity information is injected into the monoBERT model [32] in the form of Wikipedia2Vec [49] entity embeddings that are transformed into the BERT's word piece vector space [37]; see Figure 1. The EM-BERT model is first trained with the MS

MARCO passage dataset and further fine-tuned on the DBpedia-Entity v2 collection (in a cross validation setup). Our experiments indicate that our EM-BERT model improves state-of-the-art entity retrieval results by 11% with respect to NDCG@10. We also make the intriguing finding that while fine-tuning of the plain BERT-based retrieval model (monoBERT) on DBpedia-Entity v2 is brittle and prone to degenerate performance, the same fine-tuning process using the entity-enriched BERT model (EM-BERT) results in a stable model and brings consistent and meaningful improvements. We posit that this is attributed to direct injection of entity information into the BERT model, which brings the input distribution of the model close to the output label space, thereby contributing to data-efficient training of BERT-based models for entity-oriented tasks.

After observing the effectiveness of our EM-BERT model, we focus on understanding the whens and whys of entity-enriched BERT models for information retrieval tasks. Our next research question is: **RQ2:** *When and which queries are helped by the EM-BERT model?* We observe that EM-BERT mainly helps queries that are annotated with at least one entity; plain and entity-enriched BERT models perform on par with each other for queries without linked entities. We further examine queries with linked entities and find that EM-BERT is most helpful when entity mentions are broken into multiple word pieces by the BERT tokenizer (e.g. mention "Tuvalu" in Figure 1). This indicates that directly injecting entity information into the BERT-based models is particularly important for less popular entities, which BERT does not recognize as a single word (or phrase) because they are less observed during pre-training.

In our third question, we investigate **RQ3:** *Why does the EM-BERT model work and what does it learn during the fine-tuning stage?* We approach this by comparing the embeddings produced at the final layer of the EM-BERT network for both entities and their mention tokens. We find that the learned entity representations, unlike their corresponding word piece representations, form clear clusters in the embedding space, capturing the relation between entities. We also study some examples, and show that attention weights of entity tokens are larger than other tokens for queries requesting lists of entities or queries related to less known entities.

Finally, we question **RQ4:** *How does our entity-enriched BERT-based model perform on other ranking tasks?* To address this research question, we apply the EM-BERT model to the passage ranking task, where entity information is known to be less important and a large amount of training data can be used. We observe that the entity information that is embedded in the plain BERT-based retrieval model is enough for addressing this task and auxiliary entity information from knowledge graphs does not bring significant improvements.

In summary, this work makes the following contributions:

- We study the effect of enriching BERT with entity information for information retrieval tasks and propose an entity-enhanced BERT-based re-ranker EM-BERT.
- We establish new state-of-the-art results for the entity retrieval task on DBpedia-Entity v2 collection.
- We show that our entity-enriched model, unlike its equivalent model based on plain BERT, is robust against instabilities of BERT in data-constraint regimes and introduce EM-BERT as a data-efficient BERT-based model for entity retrieval.

- We perform thorough analysis of our EM-BERT model and add to our understanding of the model. We unfold when and for which queries this model work, what it learns, and how it works.

The resources developed within the course of this paper are available at https://github.com/informagi/EMBERT.

2 RELATED WORK

Graph Embeddings. After the introduction of word embeddings methods like Word2Vec [29], neural embedding methods became increasingly popular. Word2Vec uses a single-layer neural network to either predict a word depending on its context (Continuous Bag of Word) or the context depending on a word (Skip-Gram). The resulting hidden layer of this network captures a representation of words and can be used for other downstream tasks. After the success of these, embedding methods were expanded to capture other types of knowledge as well, for example knowledge graphs. These so-called graph embeddings aim to embed the nodes of a knowledge graph in a way that entities and their relations are mapped accordingly into the vector space. One of the earliest introduced methods was TransE [4], which optimizes embeddings for head-relation-tail triples.

The graph embedding method utilized in this paper is Wikipedia2Vec [49]. This method is similar to Word2Vec, but is extended to be used on text-heavy knowledge bases like Wikipedia. Instead of solely relying on words or edges between entities, it uses a combination of words, entities, and anchor texts. Using either Skip-Gram or Continuous Bag of Words method like in Word2Vec, it employs a single layer neural network to predict neighboring words to capture word similarities, entities relations (based on neighboring entities), and word-entity relations (based on surrounding anchor-text). Through this combination, words and entities get embedded into the same vector space.

Entity Retrieval. Graph embeddings have been used to assist in *entity-oriented information retrieval*. Gerritse et al. [16] uses Wikipedia2Vec embeddings for entity retrieval on the DBpedia-Entity v2 collection [21]. This is done by first tagging entities in the queries by using the TAGME entity linker [14]. Retrieved entities are then re-ranked by computing the cosine similarity score between the retrieved entities and the tagged entities, improving upon the state of the art for entity retrieval. KEWER [31] is another method for entity retrieval based on graph embeddings. It learns joint embeddings for both words and entities, combining this with entities annotated queries using SMAPH [7]. KEWER also improved previous state of the art for entity retrieval. The most recent graph-based entity ranking method, ENT Rank [12], utilizes entity-neighbor-text relations in a learning-to-rank model. ENT Rank incorporates a variety of entity features based on text, co-occurring entities, and entity neighbor relations, and performs competitively on entity ranking tasks. The paper, however, uses a different version of the DBpedia-Entity v2 collection and calculates the evaluation metrics only over judged entities, making it incomparable to our and the aforementioned entity retrieval methods.

The BEIR [40] benchmark provides a number of zero-shot retrieval models for several IR databases, including the DBpedia-Entity v2 entity retrieval collection. The best performing entity
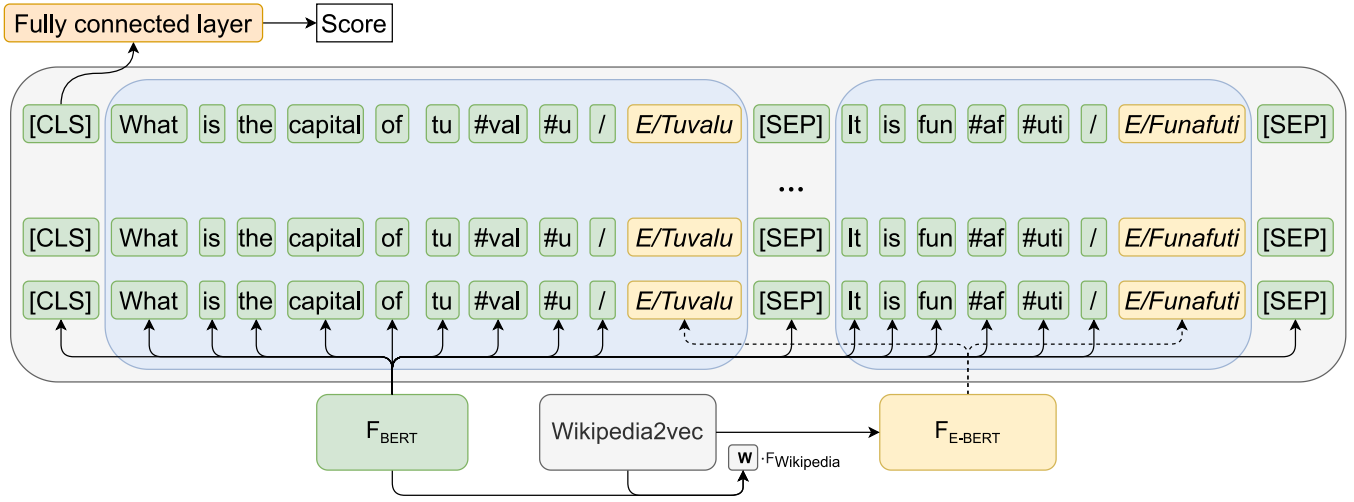
Figure 1: Illustration of the EM-BERT model. Entity annotated query and documents are tokenized and mapped to their corresponding vector representations using F_{BERT} and F_{E-BERT} functions.

retrieval model in BIER, BM25 + CE, re-ranks the top 100 entities retrieved by BM25, and then uses the MiniLM cross-encoder model [44], trained with the knowledge distillation setup provided by Hofstätter et al. [22].

Entity Linking. Utilizing entity information for retrieval models requires identifying entity information from documents and queries [3, 17, 19]. This is performed by annotating text with entity linking toolkits [6, 7, 14, 20, 41]. In this paper, we use the REL entity linker [41], which is the state-of-the-art open source entity linking toolkit. REL detects mentions using Flair [1] embeddings. REL performs candidate selection based on Wikipedia2Vec embeddings, and entity disambiguation based on latent relations between entity mentions in the text.

Transformer-based Rankers. Bidirectional Encoder Representations from Transformers (BERT) [11] is a language representation model, pre-trained to capture bi-directional relations between words. The pre-trained BERT model can be fine-tuned for various NLP tasks. BERT can be used for binary relevance classification by feeding it a sentence A and sentence B, and using the BERT classifier token (further denoted as [CLS]) to predict relevance. This setup has been shown to be effective for question answering.

BERT-based models are effective for *information retrieval* as well, and due to their relatively low number of input tokens became especially popular for *passage retrieval*. Several BERT-based methods have made enormous improvements on the previous state of the art [32, 46]. MonoBERT [32] is a point-wise re-ranker, in which the BERT model is used as a binary relevance classifier. MonoBERT, at the time of release, obtained state-of-the-art results on the MS MARCO and TREC CAR datasets. After pre-training, the ranking is done by first making an initial ranking with a baseline method like BM25, and then re-ranking a query and passage by feeding a pair of sentences A (query) and B (passage) to BERT.

Entities and Transformers. During pre-training, BERT does not explicitly get structured data about entities. Some researchers have claimed that adding this may not be necessary since BERT captures this information implicitly [35, 42]. Others set out to enrich BERT with entity information. One of the first works combining entities with transformers, ERNIE [54], is an enhanced language representation model, where the combination of entities and text is used to fine-tune the model. In KnowBert [34], contextual word embeddings are also enhanced by integrating information about entities. This is done by introducing a Knowledge Attention and Recontextualization component, which takes information in one layer of a BERT network, computes the enhanced entity information of that layer, and feeds it to the next layer. This method improves on BERT for entity-related tasks like entity linking.

E-BERT [37] is another method to enhance BERT with entity information. The main point which distinguishes E-BERT from the previously mentioned work is that it does not require any additional training of the network; it only requires the computation of one large transformation matrix. This allows the method to be applied to any fine-tuned transformer model. Since Wikipedia2Vec embeds both words and entities in the same vector space, the embeddings of words in Wikipedia2Vec can be used to align a mapping. E-BERT has been shown to improve performance on unsupervised QA tasks like LAMA, as well as downstream tasks like relation classification.

Other works combining language models and knowledge graphs include [5], which enhances BERT with entities by using BERT itself for entity linking and seeing how much information of knowledge graphs is already contained in BERT. Next, BERT classifies texts to predict whether certain tokens belong to an entity. The authors show that specifically fine-tuning BERT on entity information greatly improves their scores on entity linking, thus showing that pre-trained BERT does not embed all information about entities yet, claiming that additional entity information does not help if either entities are too scarce in the data or if the task does not require entity knowledge.

3 METHOD

In this section, we first provide a brief overview of knowledge graph embeddings and Wikipedia2Vec (Section 3.1), and then describe the EM-BERT model, which is an entity-enhanced version of a BERT-based retrieval model (Section 3.2), followed by a description of EM-BERT query-document input representation.

3.1 Background

Knowledge graph embeddings provide vector representations of entities in a knowledge graph, projecting entity properties and relations into a continuous vector space. These embeddings have proven effective not only for entity-related tasks such as entity linking [51], but also for general tasks such as question answering [31]. Knowledge graph embeddings can be constructed purely based on graph topology (i.e., entities and their relations) [4], or combining graph topology with additional textual information such as the entity descriptions [51]. In this paper, we use Wikipedia2Vec embeddings [49], which extends the skip-gram model [29] with entity relations and descriptions from Wikipedia and maps words and entities into a shared vector space. Following [37], we transform Wikipedia2Vec entity embeddings into the BERT word piece embedding space and use them as input to the BERT model, similar to a BERT word piece vector.

Formally, Wikipedia2Vec embeds a list of words $\mathbb{L}_{word}$ and a list of entities $\mathbb{L}_{ent}$ into a vector space $\mathbb{R}^{d_{Wikipedia}}$, where $d_{Wikipedia}$ is dimensions of the Wikipedia embeddings. The embeddings are trained by fitting a one layer neural network with the loss:

$$\mathcal{L} = \mathcal{L}_w + \mathcal{L}_e + \mathcal{L}_a,$$

which consist of three cross-entropy loss functions:

- $\mathcal{L}_w$: predicting context words of a given word.
- $\mathcal{L}_e$: predicting neighboring entities of a given entity.
- $\mathcal{L}_a$: predicting anchor text of an entity to combine words and entities in a single embedding space.

The weight layer of this neural network results in the function $F_{Wikipedia} : \mathbb{L}_{word} \cup \mathbb{L}_{ent} \rightarrow \mathbb{R}^{d_{Wikipedia}}$, which embeds both words and entities in the Wikipedia vector space. BERT, on the other hand, tokenizes text into the word piece dictionary $\mathbb{L}_{wordpiece}$. We use the lookup function $F_{BERT} : \mathbb{L}_{wordpiece} \rightarrow \mathbb{R}^{d_{BERT}}$ to transform word pieces into the d_{BERT}-dimensional native BERT word piece vector space, the resulting vector being the input for the BERT model.

3.2 EM-BERT

Our proposed model, referred to as EM-BERT, incorporates entity embeddings into a point-wise document ranking approach. In its essence, EM-BERT combines E-BERT [37] with monoBERT [32] and predicts a retrieval score for a query-document pair, each represented by a sequence of words and entities. The model takes as input the concatenation of the query tokens $t_{q_1}, ..., t_{q_n}$ and the document tokens $t_{d_1}, ..., t_{d_m}$, where each token t_i is either a BERT native word piece token or an entity token.

Entity-enriched BERT. The off-the-shelf BERT model only accepts its native word pieces. To incorporate entity embeddings into BERT, we need to align entity vectors with word piece vectors.

Following [37], this is done by a linear transformation of entity vectors to BERT-like vectors. Since $\mathbb{L}_{wordpiece}$ does not contain any entities, the Wikipedia2Vec word dictionary $\mathbb{L}_{Word}$ is used to obtain the linear transformation $\mathbf{W} \in \mathbb{R}^{d_{BERT} \times d_{Wikipedia}}$, learned from the Wikipedia2Vec word vector space $F_{Wikipedia}[\mathbb{L}_{Word}]$ and BERT word piece space $F_{BERT}[\mathbb{L}_{wordpiece}]$:

$$\mathbf{W} = argmin_W \sum_{x \in \mathbb{L}_{word} \cap \mathbb{L}_{wordpiece}} ||W \cdot F_{Wikipedia}(x) - F_{BERT}(x)||_2^2. \quad (1)$$

Here, the intersection between $\mathbb{L}_{word}$ and $\mathbb{L}_{wordpiece}$ is taken to ensure that the selected words have embeddings in both Wikipedia and BERT vector spaces. The equation computes matrix $\mathbf{W}$, in which the distance between $\mathbf{W} \cdot F_{Wikipedia}(x)$ and $F_{BERT}(x)$ is minimal for all x. Using $\mathbf{W}$, we can then construct the function F_{E-BERT}, which maps both entities and word tokens to the BERT input vector space:

$$F_{E-BERT}(a) = \begin{cases} \mathbf{W} \cdot F_{Wikipedia}(a), & a \in \mathbb{L}_{ent} \\ F_{wordpiece}(a), & \text{otherwise.} \end{cases} \quad (2)$$

The aligned entity vectors are then fed into BERT when an entity is mentioned in the input text. The entities are obtained by annotating the text with an entity; e.g., given the text "Who produced films starring Natalie Portman", the annotated sentence becomes "Who produced films starring Natalie Portman ENTITY/NataliePortman". This text is then tokenized as "who produced films starring natalie port ##man **/ENTITY/Natalie_Portman**," where the entity ID (in bold face) is a single token, embedded by F_{E-BERT} (Eq. 2). We note that the transformation matrix $\mathbf{W}$ can be fit into any pre-trained or fine-tuned BERT model, thus making E-BERT embeddings usable for every available BERT model.

Retrieval. The retrieval phase is based on monoBERT [33], which is a multi-stage ranking method with BERT and has been shown to achieve competitive results on MS MARCO passage retrieval [30] and TREC-CAsT [8]. The simplicity of this model, coupled with its high effectiveness enables us to discern the effect of entity information on BERT-based retrieval models and obtain an understanding of the whys and whens of entity-enriched BERT models for information retrieval tasks.

In the multi-stage ranking method, for a query q, documents are ranked with an initial ranking method; e.g., BM25. The top-k documents, denoted as $D_K = \{d_1, d_2, \ldots, d_k\}$, are then passed to a second ranker for re-ranking; here a BERT-based retrieval model. Every query-document pair $\langle q, d_i \in D_K \rangle$ is passed to BERT as two sentences A and B, with the separation token [SEP] in between. The BERT [CLS] classification vector is used as input to a single layer neural network to obtain the probability s_i of the document d_i being relevant to the query q. To train this model, the cross-entropy loss is used:

$$L = - \sum_{j \in J_{pos}} log(s_j) - \sum_{j \in J_{neg}} log(1 - s_j), \quad (3)$$

where J_{pos} and J_{neg} are indexes of relevant and non-relevant documents for all queries.

Putting all the pieces together, the training process of the EM-BERT model is summarized as follows. First, the transformation matrix $\mathbf{W}$ is trained using Wikipedia2Vec embeddings and BERT

Table 1: Statistics about document length and number of linked entities in queries and documents of the used datasets, with mean and standard deviation. Document and query length information is computed without adding entity information.

Dataset	size	avg length	avg entities	total linked
MS MARCO dev queries	6980	6.95 ± 2.54	0.34 ± 0.57	2392
MS MARCO passages	8841823	58.27 ± 22.94	2.55 ± 3.22	22532200
DBpedia-Entity v2 queries	467	6.55 ± 2.88	0.82 ± 0.64	381
DBpedia-Entity v2 short abstracts	389622	55.93 ± 25.95	6.88 ± 3.93	2678843

word pieces. All queries and documents are annotated with an entity linker, and the tagged queries and documents are tokenized and mapped to the corresponding vector representations using function F_{E-BERT} . This input is fed into the EM-BERT model and the model is first fine-tuned on MS MARCO passages and then on the entity retrieval collection.

Query-Document Representation. In the EM-BERT model (similar to monoBERT), queries are truncated to have a maximum token length of 64, and the combination of the query, passage, and separation tokens are truncated to have a maximum length of 512 tokens. Following [37], we concatenate mentions and entities, separated by the token '/'; i.e., "mention + '/' + EntityID." The final representation of query-document pairs is "[CLS] + Query + [SEP] + Document + [SEP];" see Figure 1.

4 EXPERIMENTAL SETUP

To address our research question, we compare the EM-BERT model with several baselines on the entity retrieval task and analyze the results. Due to limited training data for entity retrieval, we perform two-stage fine-tuning: we first fine-tune our models on the annotated MS MARCO passage dataset, and then continue fine-tuning on DBpedia-Entity v2 collection. In the following, we describe our experimental setup.

4.1 Entity Linking and Embeddings

We use the REL entity linking toolkit [41] to annotate texts with entities. REL is known to have high precision compared to other entity linking toolkits such as GENRE [6] and TAGME [14], ensuring that limited noise (incorrectly annotated entities) is added to the model. REL incorporates Wikipedia2Vec embeddings trained on the Wikipedia link graph. To avoid a mismatch between annotated entities and Wikipedia2Vec entity embeddings [12, 16], we follow the REL Wikipedia2Vec training procedure, using the Wikipedia dump of 2019-07-01, setting *min-entity-count* parameter to zero, with a dimensionality of 500.

4.2 First Stage Fine-tuning

Collection. For the first stage fine-tuning of EM-BERT, we use the MS MARCO passage ranking collection, consisting of 8,841,823 passages, extracted from web documents retrieved by Bing. MS MARCO provides a training set with approximately 800k queries, a development set with 6980 queries with public relevance judgments, and an evaluation set with 6837 queries and private relevance judgments.

Although MS MARCO passage ranking is not an entity-related task, REL annotates 29% of queries in the development sets with at

least one entity; e.g., "what were the *brothers grimm* names." The statistics of this collection (with REL annotations) are reported in Table 1. It shows that on average every query and document is annotated with 0.34 and 2.55 entities respectively, indicating that a fair amount of entities is present in this dataset.

Training. We fine-tune our EM-BERT model on the officially provided MS MARCO training set. This set provides triples of a query, a positive example, and a negative example, making the ratio of positive-negative documents equal during training. We refer to these triples as "sample." Starting from the monoBERT model provided by HuggingFace,[1] we fine-tune EM-BERT with randomly selected 300k samples (cf. §3.2); fine-tuning with more samples, up until 900k, did not give any further improvements on the development set. We use an accumulated batch size of 64, a learning rate of 10^{-6}, 40k warm-up steps, and AdamW as optimizer. Fine-tuning for 300k samples took around 34 hours on a single GeForce RTX 3090 with 24 GB memory.

4.3 Second Stage Fine-tuning

Collection. For the second stage fine-tuning, we use the standard entity retrieval collection, DBpedia-Entity v2 [21]. The collection is based on DBpedia 2015-10 and consists of four different categories of queries: (i) *SemSearch*: named entity queries looking for particular entities; e.g., "mario bros", (ii) *INEX-LD*: keyword-style entity-oriented queries; e.g., "bicycle benefits health", (iii) *ListSearch*: queries asking for a list of entities; e.g., "Hybrid cars sold in Europe", and (iv) *QALD-2*: natural language questions about entities; e.g., "Who produces Orangina?" Relevance judgements in this dataset are graded: Highly relevant (2), Relevant (1), and irrelevant (0).

Documents in the DBpedia-Entity v2 collection are entities, each represented by an abstract and other metadata (stored in RDF format). These entity abstracts provide concise descriptions about entities and are comparable in length to MS MARCO passages. Both queries and entity abstracts are highly entity-oriented, reflected by the average number of entity annotations for queries and abstracts in Table 1.

Training. We consider entity abstracts as entity descriptions and use them as documents in monoBERT and EM-BERT. Since the DBpedia-Entity v2 collection provides a limited amount of training data (only 467 queries), we take the fine-tuned model on the MS MARCO passage dataset (Section 4.2) and continue fine-tuning for both monoBERT and EM-BERT models using the provided folds for 5-fold cross validation, each fold containing 48k query-entity pairs

---

[1] https://huggingface.co/castorini/monobert-large-msmarco

Table 2: Results on DBpedia-Entity v2 collection. Superscripts 1/2/3/4 denote statistically significant differences (better or worse) compared to base method/monoBERT (1st)/monoBERT/EM-BERT (1st), respectively. The highest value per column is marked with underline and in bold. The highest value per column and per block is marked with underline.

NDCG	SemSearch		INEX-LD		ListSearch		QALD-2		Total	
	@10	@100	@10	@100	@10	@100	@10	@100	@10	@100
ESIM$_{cg}$ [16]	0.417	0.478	0.217	0.286	0.211	0.302	0.212	0.282	0.262	0.335
KEWER [31]	-	-	-	-	-	-	-	-	0.270	0.310
BLP-TransE [10]	0.631	0.723	0.446	0.546	0.442	0.540	0.401	0.482	0.472	0.562
BM25F+KEWER [31]	0.661	0.733	0.468	0.530	0.440	0.521	0.386	0.474	0.483	0.560
BM25	0.425	0.523	0.298	0.330	0.274	0.322	0.192	0.243	0.291	0.349
+monoBERT (1st)	0.588^1	0.655^1	0.406^1	0.459^1	0.422^1	0.458^1	0.350^1	0.406^1	0.437^1	0.490^1
+monoBERT	0.577^1	0.640^{12}	0.409^1	0.452^1	0.423^1	0.449^{12}	0.347^1	0.392^{12}	0.435^1	0.479^{12}
+EM-BERT (1st)	0.593^1	0.667^{13}	0.395^1	0.448^1	0.436^1	0.465^1	0.351^1	0.403^1	0.440^1	0.492^{13}
+EM-BERT	0.612^1	0.672^1	0.392^1	0.434^{12}	0.478^{1234}	0.469^{13}	0.375^{1234}	0.418^{134}	0.461^{1234}	0.495^{13}
BM25F-CA [21]	0.628	0.720	0.439	0.530	0.425	0.511	0.369	0.461	0.461	0.551
+monoBERT (1st)	0.605	0.682^1	0.427	0.501	0.440	0.510	0.405	0.490	0.467	0.544
+monoBERT	0.591^2	0.663^{12}	0.434	0.489^{12}	0.438	0.496^2	0.401	0.467^2	0.463	0.526^{12}
+EM-BERT (1st)	0.609	0.695^3	0.418	0.490^1	0.452	0.518^3	0.398	0.483	0.466	0.544^3
+EM-BERT	0.621	0.695	0.399	0.469^{12}	0.492^{1234}	0.535^{23}	0.435^{1234}	0.512^{1234}	0.486^{1234}	0.553^3
GEEER [16]	0.660	0.736	0.466	0.552	0.452	0.535	0.390	0.483	0.487	0.572
+monoBERT (1st)	0.643	0.733	0.486	0.564	0.493^1	0.554^1	0.449^1	0.525^1	0.515^1	0.591^1
+monoBERT	0.633^2	0.725^2	0.481	0.563	0.492^1	0.553^1	0.439^{12}	0.519^{12}	0.508^{12}	0.586^{12}
+EM-BERT (1st)	0.650	0.743	0.483	0.561	0.504^1	0.562^1	0.444^1	0.522^1	0.517^{13}	0.594^{13}
+EM-BERT	0.664	0.744	0.479	0.561	0.544^{1234}	0.579^{1234}	0.483^{1234}	0.543^{1234}	0.541^{1234}	0.604^{1234}

with approximately 34% relevant and 66% non-relevant entities. In this process, both models are fine-tuned for one epoch using all training data of each fold, resulting in 5 different models each. The fine-tuned models are then used for ranking the test queries in the corresponding folds. Note that in this fine-tuning process, no hyper-parameter optimization is performed. In this process, both models are fine-tuned for one epoch using the training data of each fold, resulting in 5 different models each. The fine-tuned models are used for ranking the test queries in the corresponding folds. Note that in this fine-tuning process, no hyper-parameter optimization is performed. For both models an aggregated batch size of 64, learning-rate of 10^{-6}, and AdamW optimizer is used. Warmup step is changed to 4000, following the 10% warmup step rule.

4.4 Baselines

We compare EM-BERT with the state-of-the-art entity retrieval methods using term-matching, neural, and BERT-based approaches. The baselines are:

BM25: The BM25 run, based on the short abstracts of entities, with parameters $k = 0.9$ and $b = 0.4$. Similar to monoBERT and EM-BERT, this run uses only abstract information of entities (unlike other methods that use reach representation of entities).

BM25F-CA [21]: The fielded variant of BM25, obtaining scores from 5-field representation of entities. Field weights are computed using Coordinate Ascent on each train fold, and then scored on the corresponding test fold. This is the best non-neural run and also the best baseline reported in [21].

ESIM$_{cg}$ [16]: ESIM$_{cg}$ ranks entities based on the aggregated cosine similarity scores between the Wikipedia2Vec embedding of each linked entity in the query and the target entity. Entities in this method are linked using TAGME [14]. The method re-ranks the BM25F-CA run.

GEEER [16]: GEEER linearly combines the BM25F-CA score and entity-query similarities based on ESIM$_{cg}$. The weights of this linear combination are computed on each train fold using Coordinate Ascent, and scored accordingly on each test fold. Similar to ESIM$_{cg}$, GEEER re-ranks the BM25F-CA run.

KEWER [31]: KEWER uses an embedding method with joint word and entity embeddings. A query-entity score is computed on all query terms by the weighted sum of the similarity between the target entity embedding and the term, re-ranking the BM25F-CA run.

BM25F+KEWER [31]: The model linearly combines the KEWER score as listed above with BM25F-CA and re-ranks the BM25F-CA run.

BLP-TransE [10]: BLP-TransE is a BERT-based method, combining a BERT cross-encoder architecture with TransE graph embeddings. It encodes queries and entities, uses the similarity between these encodings as query score, and then re-ranks and linearly combines the BM25F-CA scores.

monoBERT: Our Pygaggle implementation of the monoBERT model. This implementation achieves slightly better performance compared to results reported in [33]: MRR@10 of 0.379 vs 0.372 on the MS MARCO development set. Following the multi-stage re-ranking method [33], we re-rank three entity retrieval runs using monoBERT: BM25, BM25F-CA, and GEEER (the best baseline run). The re-ranking is performed on the top-1000 entities of the BM25 and BM25F-CA, and top-100 entities of the GEEER run (as GEEER re-ranks the BM25F-CA run).
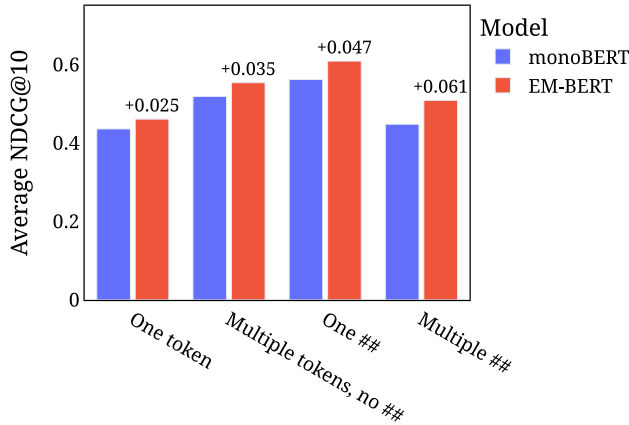
Figure 2: Comparison of re-ranked BM25F-CA run with monoBERT and EM-BERT for different mention tokenization categories.

Figure 3: Mention tokenization categories for the for types of queries in DBPedia-Entity v2 collection.

Table 3: Comparison of results for queries without and with at least one linked entity. Superscripts 1/2/3 show statistically significant differences compared to BM25F-CA/monoBERT (1st)/monoBERT, respectively.

DBpedia-Entity v2	NDCG@10 (>1 en)	NDCG@10 (no-en)
BM25F-CA	0.481	0.414
+monoBERT (1st)	0.487	0.422
+monoBERT	0.482	0.420
+EM-BERT	0.516^{123}	0.420

4.5 Evaluation Metrics

For evaluation of entity retrieval, we use the commonly reported metric in [21]; the Normalized Discounted Cumulative Gain (NDCG) at ranks 10 and 100. Statistical significant differences of NDCG@10 and NDCG@100 values are determined using the two-tailed paired t-test with p-value < 0.05.

5 RESULTS AND ANALYSIS

In the following, we evaluate our entity-enriched BERT-based retrieval model and answer our four research questions (Sections 5.1-5.4) listed in Section 1.

5.1 Entity Retrieval Results

In this section, we answer our first research question: *RQ1: Can an entity-enriched BERT-based retrieval model improve the performance of entity retrieval?*
We compare retrieval performance of EM-BERT with a variety of models on the DBpedia-Entity v2 collection in Table 2. The baseline (cf. Section 4.4) results are presented in the top part of the table. The next three compartments in the Table summarise results of re-ranking BM25, BM25F-CA, and GEEER runs with the following models:

- **monoBERT (1st)**: The monoBERT model after first stage fine-tuning; cf. Section 4.4.
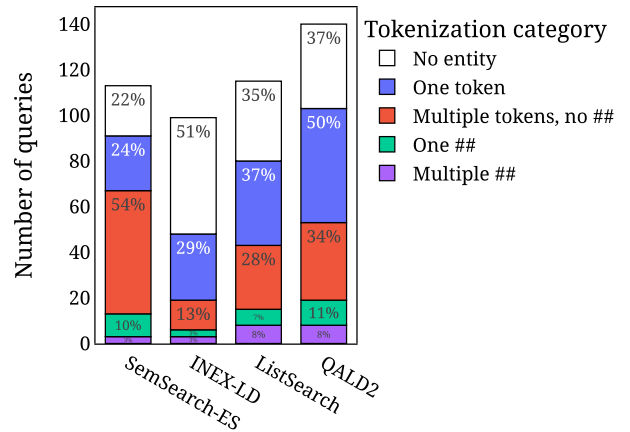
- **monoBERT**: The monoBERT model after second stage fine-tuning on DBpedia-Entity v2.
- **EM-BERT (1st)**: The EM-BERT model fine-tuned on the MS MARCO passage collection. Re-ranking setup is similar to the monoBERT runs; cf. Section 4.4.
- **EM-BERT**: The EM-BERT model after second stage fine-tuning. Setup is similar to the previous run.

As shown in Table 2, re-ranking of the GEEER run with EM-BERT markedly outperforms all baselines and establishes a new state-of-the-art result on the DBpedia-Entity v2 collection, with 11% improvements over the best run. When re-ranking the BM25 and BM25F-CA runs, we observe the same trend that EM-BERT outperforms all the corresponding BERT-based baselines. Comparing different query categories, EM-BERT improves over all runs by a large margin for ListSearch and most importantly for QALD-2 queries; improvements over GEEER are 20% and 24% with respect to NDCG@10 for ListSearch and QALD-2 queries, respectively. QALD queries are complex natural language queries that are hard to answer for most neural and non-neural models, reflected by the lowest NDCG scores on all baseline models.

Table 2 also shows that monoBERT performance is decreased when fine-tuned with a limited amount of training data. The drop in the performance of monoBERT is expected due to the known instabilities of BERT with few-sample training [53], which can cause forgetting of what the model has already learned. The striking observation, however, is that EM-BERT results are improved when trained on the same data and using the same procedure. This indicates that entity-enriched BERT models enable fine-tuning on limited training data, thereby contributing to data-efficient training of BERT for entity-oriented tasks.

Based on these results, we can positively answer our first research question: *our entity-enriched BERT-based model significantly improves state-of-the-art results on entity retrieval, and more interestingly, it is robust against instabilities of BERT when fine-tuned with limited training data.*

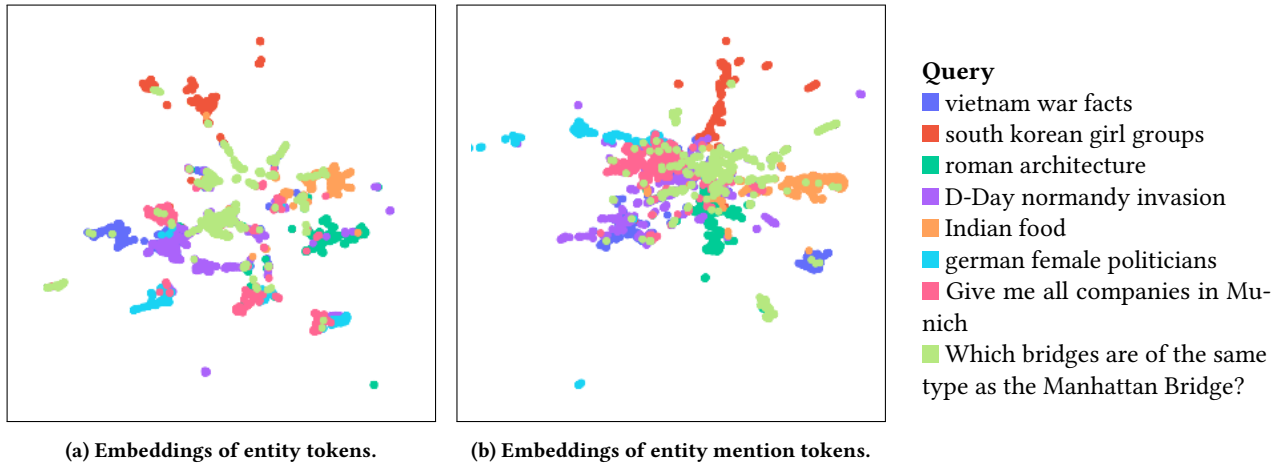(a) Embeddings of entity tokens. (b) Embeddings of entity mention tokens.

Figure 4: UMAP plots of final layer EM-BERT embeddings of entity tokens and their corresponding mention tokens. Queries were randomly selected from queries with between 100 and 200 relevant entities.

5.2 Query Analysis

In the following we answer our research question *RQ2: When and which queries are helped by the EM-BERT model?*

Entity annotations of queries. Knowing that not all queries in DBpedia-Entity v2 have a linked entity, we expect EM-BERT to perform better on queries with linked entities than on queries without them. To study this hypothesis, we compare NDCG@10 for two subsets of DBpedia-Entity v2 queries: with and without at least one linked entity. Table 3 shows the results. We see substantial improvements of EM-BERT over monoBERT and monoBERT (1st) for queries with linked entities, confirming our hypothesis.

Tokenization of entity mentions. To further our understanding of helped queries, we investigate the difference in EM-BERT performance for various types of linked queries by looking into their mentions tokenization. The BERT tokenizer splits words into word pieces, which can be categorized into *beginning* word pieces, indicating either the start of a word or a full word, and *middle/end* word pieces starting with a '##'. BERT employs around 30k tokens, consisting of around 24k beginning tokens and 6k middle/end tokens. Given that Wikipedia has around 12 million distinct entities, we expect only the most common entities to be present as a single word piece. We hypothesize that there is a performance difference between these very common entities, which are often seen in their natural form during the BERT pre-training process, and the rarer entities, where their mentions are broken into several word pieces. We, therefore, divide entity mentions into four categories:

- **One token**: Mentions that are considered as a single token by the BERT tokenizer; e.g., "France" is tokenized to 'France'.
- **Multiple tokens, no ##**: Mentions that are tokenized to more than one token, where each token represents a single unbroken word; e.g., "Yoko Ono" is tokenized to 'Yoko', 'Ono'.
- **One ##**: Mentions that are split into multiple word pieces, of which exactly one is a middle/end word piece, e.g., "Weser" is be tokenized to 'wes', '##er'

- **Multiple ##**: Mentions which gets split into multiple word pieces, of which more than one is a middle/end word piece, for example "Frisian" is tokenized to 'fr', '##isi', '##an'.

We categorize each query with a linked entity in one of the previously mentioned categories, prioritizing the most complicated entity; i.e., if a query has two mentions belonging to "One token" and "Multiple ##" categories, the query is categorized as "Multiple ##". Figure 2 compares the performance of monoBERT and EM-BERT models for these four mention tokenization categories. It shows that gains of EM-BERT are more pronounced for queries in "Multiple ##" and "One ##" categories than other categories. This supports our claim that queries with uncommon entity mentions are most helped by including their entity embeddings in the EM-BERT model. Additionally, we illustrate the relation between different tokenization categories and the query types mention (cf. Section 4.3) in Figure 3. The plot shows that 51% of INEX-LD queries have no linked entities, explaining marginal improvements of EM-BERT for these queries. This plot also describes why EM-BERT performs best on ListSearch and QALD-2 queries: most of the less known entities fall in "Multiple ##" and "One ##" categories.

These results provide an answer to RQ2: *EM-BERT helps queries that are linked to at least one entity. Additionally, entity information provided by the EM-BERT model is most useful for less popular entities, which BERT cannot treat their mentions as a single token.*

5.3 Model Analysis

To investigate **RQ3:** *Why does the EM-BERT model work and what does it learn during the fine-tuning stage?*, we examine the proximity of the embedding tokens compared to the mention tokens in the final layer of the BERT network, and also discuss several examples of helped queries.

Entities vs. mention embeddings. To get an understanding of what BERT learns from entity information, we compare entity and mention embeddings in the final layer of EM-BERT. We randomly choose eight queries with 100-200 relevant entities. For each of those documents, we take the first four linked entities and extract

Table 4: Comparison between EM-BERT and monoBERT for two example queries. Each query is listed twice, first with the normal BERT tokenization, then with the EM-BERT tokenization, both with the attentions of the first layer and first attention head. The Rel. column indicates the relevance judgement for that entity. The last two columns show the corresponding ranks obtained by the monoBERT/EM-BERT models for the entity.

Query	Entity abstract	Rel.	Baseline	Comparison
give me all movies directed by francis ford cop ##pol ##a	christopher cop ##pol ##a (born january 25 , 1962) is a film director and producer	0	monoBERT: 2	EM-BERT: 551
give me all movies directed by francis ford cop ##pol ##a / ENTITY/Francis_Ford_Coppola	rumble fish / ENTITY/Rumble_Fish is an american / ENTITY/United_States 1983 drama film directed by francis ford cop ##pol ##a / ENTITY/Francis_Ford_Coppola .	2	monoBERT: 119	EM-BERT: 1
give me the capitals of all countries in africa	list of african depend ##encies — including the respective capitals .	0	monoBERT: 1	EM-BERT: 105
give me the capitals of all countries in africa / ENTITY/Africa	da ##kar / ENTITY/Dakar is the capital and largest city of senegal . it is located on the cap - ve ##rt peninsula / ENTITY/Cap-Vert on the atlantic coast / ENTITY/East_Coast_of_the_United_States and is the western ##most city in the old world / ENTITY/Europe and on the african / ENTITY/Africa mainland .	1	monoBERT: 82	EM-BERT: 1

their entity and mention embeddings from the final layer of EM-BERT. The mention embeddings are taken from the first starting token before that entity. We use UMAP [28] with the default settings to plot these embeddings in 2-dimensional space; see Figure 4. As is evident from the plots, the entity embeddings form more pronounced clusters than the entity mention embeddings. For example, in the entity embedding plot, American bridges and rivers build a cluster together with other entities based in the US or Canada, while being close to the corresponding queries. This also indicates that the entity information injected through entity embeddings prevails into the final layers of the BERT model, keeping a better representation of entity information than provided with the default BERT word piece tokens.

Insightful Examples. We discuss in detail a few (hand-picked) example queries that are especially helped or hurt by re-ranking with the EM-BERT model. Table 4 visualizes the attention weights of the [CLS] token for the first attention head on the first layer, for sample query-entity pairs. In the query "Give me all movies directed by Francis Ford Coppola", monoBERT incorrectly ranks other filmmaker members of the Coppola family, matching the last name only. EM-BERT, in contrast, finds the correct entity, as movies directed by Francis Ford Coppola are situated close to him in the embedding space (as opposed to the movies by his daughter Sofia Coppola). Table 4 shows that EM-BERT gives high attention to the entity tokens relevant to the entity and query. It also shows high attention to the words "movies", which is essential here.

We can also see that through fine-tuning on DBpedia-entity, EM-BERT learns to retrieve specific entities for queries requesting a list. For example for the query "Give me the capitals of all countries in Africa", while monoBERT's highest ranked entities are general overview pages like 'List of African dependencies', all of EM-BERT's highest-ranked entities are capitals of African countries like 'Dakar' and 'Porto-Novo'.

5.4 Performance on Passage Retrieval

Finally, we turn to our last research question, ***RQ4:** How does our entity-enriched BERT-based model perform on other ranking tasks?* To further our understanding on the effect of entity-enriched BERT for IR tasks, we compare the monoBERT and EM-BERT models after the first fine tuning stage on the MS MARCO collection. We observe no significant differences between the two models for both queries with and without linked entities. A similar observation is made for the TREC Conversation Assistant Track (TREC-CAsT) [24]. We posit that the added value of enhancing BERT-based retrieval models with explicit entity representations is mainly pronounced for entity-oriented tasks.

6 CONCLUSION AND FUTURE WORK

In this research, we investigated the value of adding entity embeddings to BERT for entity search. We proposed an entity-enhanced BERT-based retrieval model, EM-BERT, and compared it with a regular BERT model. We found that EM-BERT improves substantially over the state-of-the-art results, showing that entity-oriented tasks benefit from entity embeddings in BERT, as is hypothesized in [5]. We found three categories where entities help the most: complex natural language queries, queries requesting a list of entities with a certain property, and queries with less known entities, in which their mentions get split in multiple tokens by the BERT tokenizer. We also showed that EM-BERT, unlike BERT-based retrieval models, can be fine-tuned with limited training data, thus being effective for lesser-resourced entity-related tasks.

For further research, the effect of the entity linking method can be investigated; here we used REL, which is an entity linker with high precision. It would be also interesting to evaluate the performance of other dense and sparse retrieval methods on entity-oriented queries and tasks. This, however, may require additional training data for fine-tuning on entity-oriented queries, which would create a supplementary MS Marco subset to train and evaluate ranking systems.

REFERENCES

[1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *Proc. of 27th International Conference on Computational Linguistics (COLING '18)*. 1638–1649.

[2] Krisztian Balog. 2018. *Entity-Oriented Search*. The Information Retrieval Series, Vol. 39. Springer.

[3] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and Space-Efficient Entity Linking in Queries. *Proc. of the Eighth ACM International Conference on Web Search and Data Mining* (2015), 179–188.

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proc. of the 26th International Conference on Neural Information Processing Systems (NeurIPS '13)*. 2787–2795.

[5] Samuel Broscheit. 2019. Investigating Entity Knowledge in BERT with Simple Neural End-To-End Entity Linking. In *Proc. of the 23rd Conference on Computational Natural Language Learning (CoNLL '19)*. 677–685.

[6] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *Proc. of the International Conference on Learning Representations (ICLR '21)*.

[7] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. 2016. A Piggyback System for Joint Entity Mention Detection and Linking in Web Queries. In *Proc. of the 25th International Conference on World Wide Web (WWW '16)*. 567–578.

[8] Jeffrey Dalton, Chenyan , and Jamie Callan. 2020. TREC CAsT 2019: The conversational assistance track overview. *arXiv preprint* (2020). arXiv:2003.13624

[9] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity Query Feature Expansion Using Knowledge Base Links. In *Proc. of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '14)*. 365–374.

[10] Daniel Daza, Michael Cochez, and Paul Groth. 2021. Inductive Entity Representations from Text via Link Prediction. In *Proc. of the Web Conference 2021 (WWW '21)*. 798–808.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of The North American Chapter of the Association for Computational Linguistics '19 (NAACL '19)*. 4171–4186.

[12] Laura Dietz. 2019. ENT Rank: Retrieving Entities for Topical Information Needs through Entity-Neighbor-Text Relations. In *Proc. of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. 215–224.

[13] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305* (2020).

[14] Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-Fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proc. of the International Conference on Information and Knowledge Management (CIKM '10)*. 1625–1628.

[15] Darío Garigliotti, Faegheh Hasibi, and Krisztian Balog. 2019. Identifying and Exploiting Target Entity Type Information for Ad hoc Entity Retrieval. *Information Retrieval Journal* 22, 3 (2019), 285–323.

[16] Emma Gerritse, Faegheh Hasibi, and Arjen De Vries. 2020. Graph-Embedding Empowered Entity Retrieval. In *Proc. of European Conference on Information Retrieval (ECIR '20)*.

[17] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2015. Entity Linking in Queries: Tasks and Evaluation. In *Proc. of the 2015 International Conference on The Theory of Information Retrieval (ICTIR '15)*. 171–180.

[18] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting Entity Linking in Queries for Entity Retrieval. In *Proc. of the 2016 ACM International Conference on the Theory of Information Retrieval (ICTIR '16)*. 209–218.

[19] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2017. Entity Linking in Queries: Efficiency vs. Effectiveness. In *Proc. of the 39th European Conference on Information Retrieval (ECIR '17)*. 40–53.

[20] Faegheh Hasibi, Krisztian Balog, Darío Garigliotti, and Shuo Zhang. 2017. Nordlys: A toolkit for entity-oriented and semantic search. In *Proc. of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. 1289–1292.

[21] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. DBpedia-Entity V2: A Test Collection for Entity Search. In *Proc. of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. 1265–1268.

[22] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. *arXiv preprint* (2020). arXiv:2010.02666

[23] Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020. X-FACTR: Multilingual Factual Knowledge Retrieval from Pretrained Language Models. In *Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP '20)*. 5943–5959.

[24] Hideaki Joko, Emma J Gerritse, Faegheh Hasibi, and Arjen P de Vries. 2022. Radboud University at TREC CAsT 2021. In *TREC*.

[25] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics* (2020), 64–77.

[26] Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for Coreference Resolution: Baselines and Analysis. In *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*. 5803–5808.

[27] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint* (2019). arXiv:1907.11692

[28] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. 2018. UMAP: Uniform Manifold Approximation and Projection. *The Journal of Open Source Software* 3, 29 (2018), 861.

[29] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS '13)*. 3111–3119.

[30] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proc. of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016*.

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

[32] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint* (2019). arXiv:1901.04085

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

[34] Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*. 43–54.

[35] Fabio Petroni, Tim Rocktäschel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language Models as Knowledge Bases?. In *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*. 2463–2473.

[36] Jason Phang, Thibault Fevry, and Samuel R. Bowman. 2019. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *arXiv preprint arXiv:1811.01088* (2019).

[37] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. In *Findings of the Association for Computational Linguistics (ELMNLP '20)*. 803–818.

[38] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

[39] Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics - On what Language Model Pre-training Captures. *Transactions of the Association for Computational Linguistics* 8 (2020), 743–758.

[40] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Proc. of 35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (NeurIPS '21)*.

[41] Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. 2020. REL: An Entity Linker Standing on the Shoulders of Giants. In *Proc. of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*.

[42] Chenguang Wang, Xiao Liu, and Dawn Song. 2020. Language Models are Open Knowledge Graphs. *arXiv preprint* (2020). arXiv:2010.11967

[43] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. *arXiv preprint* (2020). arXiv:2002.01808

[44] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS '21)*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.). 5776–5788.

[45] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2019. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *arXiv preprint* (2019). arXiv:1911.06136

[46] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering. In *Proc. of the 2019 Conference on Empirical Methods in

Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19). 5878–5882.

[47] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2017. Word-Entity Duet Representations for Document Ranking. In *Proc. of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. 763–772.

[48] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding. In *Proc. of the 26th International Conference on World Wide Web (WWW '17)*. 1271–1279.

[49] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP '20)*. 23–30.

[50] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proc. of the 2020 Conference on Empirical Methods in Natural*

[51] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proc. of The SIGNLL Conference on Computational Natural Language Learning (SIGNLL '16)*. 250–259.

[52] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems (NeurIPS '19)*. 5753–5763.

[53] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting Few-sample BERT Fine-tuning. In *Proc. of 2021 International Conference on Learning Representations (ICLR '20)*.

[54] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics (ACL '19)*. 1441–1451.

Language Processing (EMNLP '20). 6442–6454.