TABi: Type-Aware Bi-Encoders for Open-Domain Entity Retrieval

Megan Leszczynski **Daniel Y. Fu** **Mayee F. Chen** **Christopher Ré**
Department of Computer Science, Stanford University
{mleszczy, danfu, mfchen, chrismre}@cs.stanford.edu

Abstract

Entity retrieval—retrieving information about entity mentions in a query—is a key step in open-domain tasks, such as question answering or fact checking. However, state-of-the-art entity retrievers struggle to retrieve rare entities for ambiguous mentions due to biases towards popular entities. Incorporating knowledge graph types during training could help overcome popularity biases, but there are several challenges: (1) existing type-based retrieval methods require mention boundaries as input, but open-domain tasks run on unstructured text, (2) type-based methods should not compromise overall performance, and (3) type-based methods should be robust to noisy and missing types. In this work, we introduce TABi, a method to jointly train bi-encoders on knowledge graph types and unstructured text for entity retrieval for open-domain tasks. TABi leverages a type-enforced contrastive loss to encourage entities and queries of similar types to be close in the embedding space. TABi improves retrieval of rare entities on the Ambiguous Entity Retrieval (AmbER) sets, while maintaining strong overall retrieval performance on open-domain tasks in the KILT benchmark compared to state-of-the-art retrievers. TABi is also robust to incomplete type systems, improving rare entity retrieval over baselines with only 5% type coverage of the training dataset. We make our code publicly available.[1]

1 Introduction

Entity retrieval (ER) is the process of finding the most relevant entities in a knowledge base for a natural language query.[2] ER is crucial for open-domain NLP tasks, where systems are provided with a query without the information needed to answer the query (Karpukhin et al., 2020). For instance, to answer the query, *"What team does George Washington play for?"* an open-domain system can use an entity retriever to find information about George Washington in a knowledge base. Retrieving the correct George Washington in the query above—George Washington the baseball player, rather than George Washington the president—requires the retriever to recognize that keywords "team" and "play" imply George Washington is an athlete. However, recent work has shown that state-of-the-art retrievers exhibit popularity biases and struggle to resolve ambiguous mentions of rare "tail" entities (Chen et al., 2021).

The goal of our work is to improve rare entity retrieval for open-domain NLP tasks. Rare entities are challenging to retrieve when they share a name with more popular entities. For instance, in a sample of Wikipedia, mentions of George Washington refer to the president 93% of the time, so a retriever can do very well by learning a popularity bias and returning the president whenever it sees "George Washington." This strategy performs poorly on rare entities like George Washington the baseball player. To retrieve a rare entity instead of a popular entity for an ambiguous mention, the retriever needs to learn to leverage context cues to overcome the popularity bias. However, existing state-of-the-art retrievers for open-domain tasks (e.g., GENRE (Cao et al., 2021), DPR (Karpukhin et al., 2020)) are only trained on unstructured text, making it challenging for them to learn to associate context cues (e.g. "team" and "play") with groups of entities (e.g., athletes).

A promising approach to overcome popularity biases is to incorporate types (e.g., athlete or politician) from a knowledge graph into the retriever. A key advantage of types is that contextual cues learned over popular entities can generalize to rare entities of the same types. However, there are several challenges with using types for open-domain retrieval. First, existing methods that use types assume mention boundaries are provided in the input (Gupta et al., 2017; Onoe and Durrett, 2020; Orr et al.,

[1] https://github.com/HazyResearch/tabi

[2] We use entity retrieval to refer to the page-level document retrieval setting, where entities correspond to Wikipedia pages.

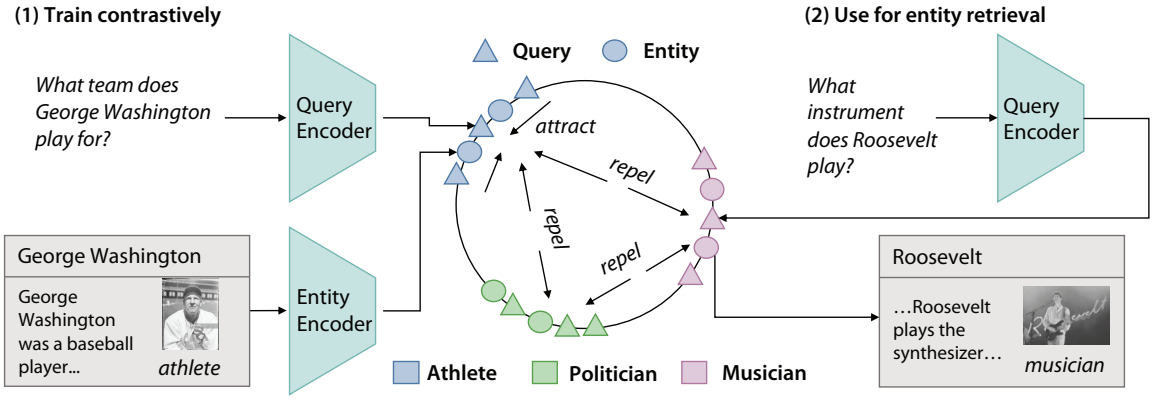

Figure 1: TABi uses a query and entity encoder to embed queries and entities in the same space. To encourage embeddings of the same type (e.g. athlete) to be close, TABi introduces a type-enforced contrastive loss that pulls query embeddings of the same type together and pushes query embeddings of different types apart.

2021), but open-domain tasks run over unstructured text. These methods can suffer significant quality degradation without mention boundaries.[3] Second, while it is important to do well on tail entities, the ideal retriever also needs to maintain strong performance over popular entities, balancing learning popularity biases with learning contextual cues. Finally, a retriever that incorporates types needs to be robust to incorrect and missing types, as type labels can be noisy and knowledge graphs can be incomplete.

In this work, we introduce TABi, a method for training entity retrievers on knowledge graph types and unstructured text. TABi builds on the bi-encoder model for dense retrieval (e.g., Wu et al., 2020; Karpukhin et al., 2020) (Figure 1). Bi-encoders learn embeddings of queries and entities contrastively: query embeddings are pulled close to their ground truth entity embedding and pushed away from other entity embeddings.

Our key insight is that type information should also be learned contrastively, as opposed to more straightforward approaches like adding the type as textual input. TABi adds a type-enforced contrastive loss term that pulls query embeddings of the same type together and pushes query embeddings of different types apart. As a result, TABi clusters embeddings by type more strongly than simply adding the type as input or not using types at all (Figure 2), and thus performs better on nearest neighbor type classification and entity similarity tasks. Finally, motivated by "universal" dense retrievers (Maillard et al., 2021), TABi trains over multiple open-domain tasks in addition to entity disambiguation to support retrieval without mention boundaries.

[3]We find retrieval performance can drop 40% (relative) by using mention detection v. gold mention boundaries.

Our experiments show that TABi addresses the challenges of using types for open-domain retrieval. First, we find that training a bi-encoder over multiple open-domain tasks significantly improves average top-1 tail retrieval by 29.1 points compared to existing state-of-the-art baselines. Our type-enforced loss further improves average top-1 tail retrieval by nearly 6 points. Second, TABi maintains strong overall retrieval performance on popular entities, nearly matching or outperforming the state-of-the-art multi-task model, GENRE, on the eight open-domain KILT tasks (Petroni et al., 2021). Third, TABi is robust to missing and incorrect types, obtaining 79% of the lift from the type-enforced loss even when only 5% of the training examples have type annotations. Finally, we also explore a hybrid model that combines TABi with a sparse retriever and popularity statistics. We find the hybrid model can lead to strong performance even when TABi is trained without hard negative sampling, a standard but computationally expensive training procedure.

To summarize, our contributions are as follows:

- We introduce TABi, a method to train bi-encoders on knowledge graph types and unstructured text through a new type-enforced contrastive loss for open-domain entity retrieval.
- We demonstrate that TABi improves rare entity retrieval performance, maintains strong overall retrieval performance, and is robust to noisy and missing types on AmbER and KILT.
- We validate that our approach can better capture types in query and entity embeddings than baseline dense entity retrievers through embedding visualization, nearest neighbor type classification, and an entity similarity task.

2 Preliminaries

We review the problem setup, task, and bi-encoders.

Problem setup Let $q \in \mathcal{Q}$ be a query, $e \in \mathcal{E}$ be an entity description, $y \in \mathcal{Y}$ be the entity label from the knowledge base, and $t \in \mathcal{T}$ be the type label.[4] We assume as input a labeled dataset $D = \{(q_i, e_i, y_i, t_i)\}_{i=1}^{n}$, where n is the number of examples.

Entity retrieval task Given a query q as input, the entity retrieval task is to return the top-K entity candidates relevant to the query from $\mathcal{Y}$. Since our primary motivation is open-domain NLP tasks, we focus on the page-level document retrieval setting, where we assume that each document corresponds to an entity (e.g., Wikipedia page) and that no mention boundaries are provided as input.

Bi-encoders for entity retrieval The bi-encoder model consists of a query encoder $f : \mathcal{Q} \rightarrow \mathbb{R}^d$ and an entity encoder $g : \mathcal{E} \rightarrow \mathbb{R}^d$. Most bi-encoders (e.g., Gillick et al., 2019; Wu et al., 2020) are trained with the InfoNCE loss (van den Oord et al., 2018), in which "positive" pairs of examples are pulled together and "negative" pairs of examples are pushed apart. For a particular query q, let its positive example e^+ be the entity description for the respective gold entity and its negative examples $N_e(q)$ be the set of all other entity descriptions in the batch. For a batch with queries Q and entity descriptions E, the loss is defined as:

$$L_{NCE}(Q,E) = \frac{-1}{|Q|} \sum_{q \in Q} \log \frac{\psi(q,e^+)}{\psi(q,e^+) + \sum_{e^- \in N_e(q)} \psi(q,e^-)},$$

where $\psi(v,w) = \exp(f(v)^\top g(w)/\tau)$ is the similarity score between the embeddings v and w, and τ is a temperature hyperparameter. L_{NCE} pulls each query embedding close to the entity embedding for its gold entity and pushes it away from all other entity embeddings in the batch. Batches are often constructed with hard negative samples to improve overall quality (e.g., Gillick et al., 2019).

3 Approach

TABi leverages knowledge graph types and unstructured text to train bi-encoders for open-domain entity retrieval. TABi takes as input queries and entity descriptions and uses a type-enforced contrastive loss. At inference time, TABi uses nearest neighbor search to retrieve entities.

Input The query q is represented as the WordPiece (Wu et al., 2016) tokens in the query, with special tokens `[M_s]` and `[M_e]` around the mention if the mention boundaries are known (matching the input of Wu et al. (2020) with mention boundaries and Karpukhin et al. (2020) without). The entity description e is represented as the first 128 WordPiece tokens of the entity's title and a description (i.e., Wikipedia page), with each component separated by an `[E_s]` token, following Wu et al. (2020). We fine-tune the standard BERT-base pretrained model (Devlin et al., 2019) for both the query and entity encoders and take the final hidden layer representation corresponding to the `[CLS]` token as the query and entity embeddings. Similar to work in contrastive learning (Chen et al., 2020b), we then apply L2 normalization to the embeddings.

Type-Enforced Contrastive Loss We propose a contrastive loss that incorporates knowledge graph types and builds on the supervised contrastive loss from Khosla et al. (2020). Our goal is to encode types in the embedding space, such that the embeddings of queries and entities of the same type are closer together than those of different types. Types are often not sufficient to distinguish an entity, so we also want to embed queries and entities with similar names close together.

To achieve these two goals, our loss is a weighted sum of two supervised contrastive loss terms, L_{type} and L_{ent}. For a randomly-sampled batch from dataset D with queries Q and entity descriptions E, TABi's loss L_{TABi} is given by:

$$L_{TABi}(Q,E) = \alpha L_{type}(Q) + (1-\alpha) L_{ent}(Q,E), \quad (1)$$

where $\alpha \in [0,1]$ (we use $\alpha = 0.1$ in our experiments).

$L_{type}(Q)$ uses type labels to form positive and negative pairs over queries.[5] Let $P_{type}(q)$ be the set of all queries in a batch that share the same type t as a query q and $N_{type}(q)$ be the other queries in the

[4] To simplify notation, we define a single type label. In experiments, we define the type label as a set of entity types and type equivalence as 50% of types matching (see Appendix B.4).

[5] We contrast queries in L_{type} because we find it is more difficult to learn the query type than the entity type.

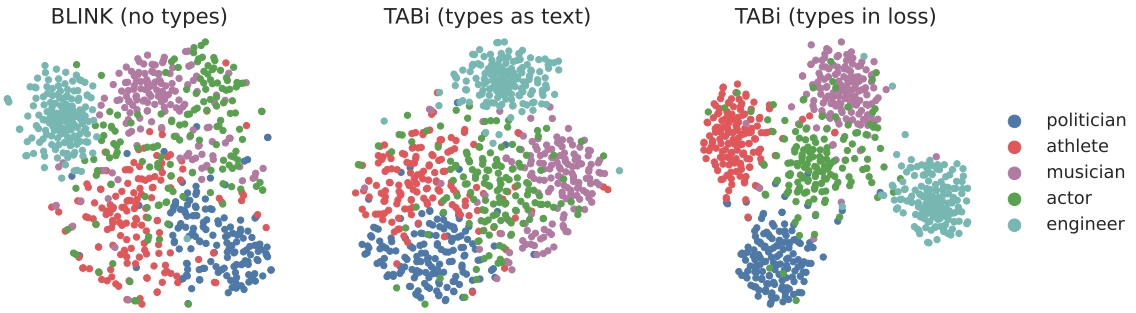

Figure 2: t-SNE visualizations of entity embeddings. (a) BLINK trained with L_{NCE} without types. (b) TABi trained with L_{ent} with types only as text in the input. (c) TABi trained with the type-enforced loss L_{TABi}.

batch. Then $L_{type}(Q)$ is:

$$L_{type}(Q) = \frac{-1}{|Q|} \sum_{q \in Q} \frac{1}{|P_{type}(q)|} \sum_{q^+ \in P_{type}(q)} \log \frac{\psi(q,q^+)}{\psi(q,q^+) + \sum_{q^- \in N_{type}(q)} \psi(q,q^-)}. \quad (2)$$

$L_{ent}(Q,E)$ uses entity labels to form positive and negative pairs over queries and entity descriptions.[6] Let x be a query or entity description, and $P_{ent}(x)$ be the set of all queries and entity descriptions in a batch that share the same gold entity y as x. Let $N_{ent}(x)$ be the set of all other queries and entity descriptions in the batch. Then $L_{ent}(Q,E)$ is:

$$L_{ent}(Q,E) = \frac{-1}{|Q \cup E|} \sum_{x \in Q \cup E} \frac{1}{|P_{ent}(x)|} \sum_{x^+ \in P_{ent}(x)} \log \frac{\psi(x,x^+)}{\psi(x,x^+) + \sum_{x^- \in N_{ent}(x)} \psi(x,x^-)}. \quad (3)$$

We tie the weights of the query and entity encoders such that $f(\cdot) \equiv g(\cdot)$ so that ψ is well-defined for all pairs of queries and entities.[7] We also normalize embeddings before computing ψ. Following recent work (Gillick et al., 2019; Karpukhin et al., 2020), we use hard negative sampling to add the top nearest incorrect entities for each query to the batch.[8] We follow Botha et al. (2020) to balance the hard negatives by fixing the ratio of positive to negative examples allowed for each entity, reducing the proportion of hard negatives that are rare entities (see Appendix A.4).

[6] In contrast, L_{NCE} only compares query-entity pairs. We find that additionally comparing query-query and entity-entity pairs for L_{ent} helps in §4.2.

[7] Both encoders take a list of tokens as input.

[8] We train with three hard negatives for each query.

The key difference between L_{type} and L_{ent} is the set of positive and negative pairs. L_{type} forms pairs by type, which clusters queries of the same type in the embedding space. L_{ent} forms pairs by gold entity, which clusters queries and entities with similar names in the embedding space. Figure 2 shows that L_{TABi} produces embeddings that cluster better by types than those produced by L_{NCE} (BLINK (Wu et al., 2020)) or L_{ent} with types simply added as text to the entity encoder input.

Inference We precompute entity embeddings and use nearest neighbor search to retrieve the top-K most similar entity embeddings to a query embedding. While our standard configuration does not use a re-ranker, in Section 4.2 we also study the impact of adding an inexpensive re-ranker which linearly combines TABi's scores with sparse retriever scores and popularity statistics (see Appendix A.5). Prior work has shown that a hybrid model that combines sparse retrievers (e.g. TF-IDF) and dense retrievers can improve performance (Karpukhin et al., 2020; Luan et al., 2021) and that entity popularity can help disambiguation (Ganea and Hofmann, 2017).

4 Retrieval Experiments

Our experiments find that TABi can improve rare entity retrieval for open-domain NLP tasks while maintaining strong overall retrieval performance.

4.1 Experimental setup

We describe the baselines, evaluation datasets, knowledge base, and training data. We include additional setup details in Appendix A.

Baselines We compare against text-only baselines, which do not use types, to evaluate to what extent using types can improve performance over existing methods. We also compare against type-aware baselines, which use types and text, to better

understand the challenges with incorporating types.

- *Text-only baselines:* Alias Table sorts candidates by their prior probabilities with the mention in the BLINK training dataset. TF-IDF uses sparse embeddings of normalized word frequencies. DPR (Karpukhin et al., 2020) is a dense passage retriever that does not use mention boundaries. BLINK (Bi-encoder) (Wu et al., 2020) is a state-of-the-art dense entity retriever which uses mention boundaries; we also compare against BLINK with a cross-encoder to re-rank the top 10 candidates from the bi-encoder. ELQ (Li et al., 2020) finetunes the BLINK bi-encoder jointly with mention detection and entity disambiguation tasks. GENRE (Cao et al., 2021) is an autoregressive retriever that generates the full entity name from the mention. We use pretrained models for all text-only baselines, with the exception of Alias Table and TF-IDF, which are non-learned.
- *Type-aware baselines:* Bootleg (Orr et al., 2021) is a Transformer-based model that re-ranks candidates from an alias table using types and knowledge graph relations. We also introduce two baselines for encoding types in open-domain retrievers: GENRE-type and TABi-type-text. GENRE-type includes the types as part of the entity name, and thus must generate the entity name along with its types. TABi-type-text adds the types as textual input to the entity encoder instead of the loss function and uses L_{ent} for training. We use a pretrained model for Bootleg, fine-tune a pretrained model of GENRE to create GENRE-type, and fine-tune TABi-type-text from a BERT-base pretrained model (Devlin et al., 2019).

Evaluation datasets We use 14 datasets from two benchmarks: Ambiguous Entity Retrieval (AmbER) (Chen et al., 2021) and Knowledge Intensive Language Tasks (KILT) (Petroni et al., 2021). AmbER evaluates retrieval of ambiguous rare entities, and KILT evaluates overall retrieval performance.

AmbER. AmbER (Chen et al., 2021) spans three tasks in open-domain NLP—fact checking, slot filling, and question answering—and is divided into human and non-human subsets, for a total of 6 datasets. AmbER tests the ability to retrieve the correct entity when at least two entities share a name (i.e. are ambiguous). The queries are designed to be resolvable, such that each query should contain enough information to retrieve the correct entity. AmbER also comes with "head" (i.e. popular) and "tail" (i.e. rare) labels, using Wikipedia page views for popularity. We split AmbER into dev and test (5/95 split) and report on the test set.[9]

We create a variant of this dataset–AmbER (GOLD)–with gold mention boundaries. While we focus on open-domain tasks, where mention boundaries are often unknown, AmbER (GOLD) enables us to evaluate disambiguation in isolation.

Following Chen et al. (2021), we report accuracy@1 (i.e. top-1 retrieval accuracy), which is the percentage of queries where the top-ranked entity is the gold entity. As multiple entities share a name with the query mention (by the dataset definition), this metric captures how well a model can use context to disambiguate.

KILT. We consider 8 evaluation datasets across the four open-domain tasks in the KILT (Petroni et al., 2021) benchmark (fact checking (FC), question answering (QA), slot filling (SF), and dialogue). All examples have been annotated with the Wikipedia page(s) that help complete the task.

Following Petroni et al. (2021), we report R-precision (Beitzel et al., 2009). Given R gold entities, R-precision is equivalent to the proportion of relevant entities in the top-R ranked entities. With the exception of FEVER and HotPotQA, which may require multiple entities, R-precision is equivalent to accuracy@1. We compare against published and leaderboard numbers for KILT and refer the reader to Petroni et al. (2021) for baseline details.

Knowledge base We create a filtered version of the KILT knowledge base (Petroni et al., 2021) with 5.45M entities that correspond to English Wikipedia pages. We remove Wikimedia internal items (e.g., disambiguation pages, list articles) from the KILT knowledge base, since they do not refer to real-world entities. We refer to our knowledge base as KILT-E (KILT-Entity) and use it for all models at inference time for fair comparison.[10]

Training data We train two versions of TABi to understand the performance with and without mention boundaries in the input. For retrieval experiments with mention boundaries and embedding quality experiments, we train on the BLINK (Wu et al., 2020) training data, which consists of 8.9M

[9] We use AmbER dev to select re-ranker hyperparameters in Section 4.2.

[10] As an exception, we report existing numbers for baselines with the full KILT knowledge base (5.9M entities) on the KILT benchmark test sets due to a benchmark submission limit. See Appendix B.2 for dev results with KILT-E knowledge base.

	Fact Checking				Slot Filling				Question Answering				Average	
	H		N		H		N		H		N			
Model	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
TF-IDF	27.8	29.3	23.0	21.8	26.7	23.5	17.3	13.7	24.2	22.6	18.2	13.9	22.9	20.8
DPR	25.3	14.3	47.7	23.7	13.9	5.1	48.6	22.2	21.0	8.8	52.1	23.4	34.8	16.3
BLINK (Bi-encoder)	56.4	52.0	24.8	10.5	**76.8**	55.7	30.7	13.5	78.3	55.7	67.3	33.8	55.7	36.9
BLINK	55.8	45.8	7.4	3.9	74.7	30.3	32.1	16.1	**83.8**	43.8	71.3	44.5	54.2	30.7
ELQ	43.5	37.4	5.3	2.2	74.4	44.1	59.5	27.1	77.5	47.2	62.0	30.7	53.7	31.4
GENRE	59.9	30.7	32.6	19.9	67.1	52.6	72.9	59.5	62.9	28.4	61.1	32.4	59.4	37.2
Bootleg†	48.7	37.0	3.7	2.5	65.1	48.0	47.5	26.7	74.8	48.0	60.5	44.2	50.0	34.4
GENRE-type	32.2	50.6	**55.7**	34.9	34.9	68.0	75.4	69.6	41.6	55.8	72.1	47.6	52.0	54.4
TABi-type-text	76.7	60.4	39.0	36.8	71.6	86.3	82.5	85.2	69.6	66.1	82.3	57.0	70.3	65.3
TABi	**83.5**	**73.3**	40.7	**41.7**	75.1	**89.4**	**85.6**	**88.0**	78.0	**74.3**	**83.0**	**66.1**	**74.3**	**72.1**
TABi ($\alpha=0$)	77.6	61.9	41.4	39.1	70.9	87.1	83.2	85.9	72.5	66.3	82.2	57.7	71.3	66.3
TABi (L_{type}+L_{NCE})	80.5	64.7	42.0	42.2	69.1	87.7	83.9	87.3	72.2	67.7	81.3	61.8	71.5	68.6

Table 1: Retrieval accuracy@1 on AmbER (H for human, N for non-human subsets). (Top) text-only methods, (middle) type-aware methods, and (bottom) ablations. †Models with an alias table. See Section 4.2 for training data details. Best score **bolded**, second best underlined (excluding ablations).

	Fact Checking				Slot Filling				Question Answering				Average	
	H		N		H		N		H		N			
Model	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
Alias Table†	45.9	6.6	45.8	7.9	45.9	6.5	45.7	7.8	45.7	6.5	45.3	7.9	45.7	7.2
TF-IDF	27.8	29.3	23.0	21.8	26.7	23.5	17.3	13.7	24.2	22.6	18.2	13.9	22.9	20.8
BLINK (Bi-encoder)	77.5	66.5	77.0	46.0	76.9	55.9	63.8	29.9	78.4	55.8	71.0	34.8	74.1	48.2
BLINK	81.8	61.0	81.6	58.5	75.4	30.5	64.8	35.7	83.8	43.9	74.9	45.7	77.1	45.9
GENRE	70.9	44.5	72.9	40.6	70.6	39.0	64.8	33.1	71.1	40.6	70.3	40.0	70.1	39.6
Bootleg†	83.0	70.7	**82.1**	56.6	**84.9**	58.8	**76.1**	**54.7**	**86.3**	51.2	**79.2**	**56.5**	**82.0**	58.1
GENRE-type	69.7	60.8	75.9	48.5	70.9	54.3	66.7	37.2	70.7	54.6	72.5	46.7	71.1	50.3
TABi-type-text	81.5	75.0	78.9	58.1	78.5	62.1	63.1	38.6	80.0	61.5	68.2	42.0	75.0	56.2
TABi	**84.4**	**82.3**	80.4	**63.5**	78.5	**68.6**	64.5	39.1	81.5	**69.8**	71.8	51.6	76.9	**62.5**

Table 2: Retrieval accuracy@1 on AmbER (GOLD) (with mention boundaries). (Top) text-only, (bottom) type-aware methods. All models are trained on Wikipedia. †Models with an alias table. Best score **bolded**, second best underlined.

Wikipedia sentences.[11] For retrieval experiments without mention boundaries, we follow Cao et al. (2021) and train on all KILT training data (which includes open-domain tasks) and contains 11.7M sentences (Petroni et al., 2021). For type labels, we use the 113 types from the FIGER (Ling and Weld, 2012) type set. To assign entity types, we use a direct mapping of Wikidata entities to Freebase entities to find the FIGER types associated with each entity in Freebase. To assign query types, we follow Ling and Weld (2012) and add the types of the gold entity for each query as the query type labels. While types can be incomplete and not present in the query, we find that the type labels are sufficient for improving the embedding quality (§5).

4.2 Results

Rare entities TABi improves retrieval of rare entities for ambiguous mentions. On AmbER, TABi improves average tail accuracy@1 by 34.9 points compared to existing text-only baselines and 6.8 points compared to type-aware baselines (Table 1). Note that GENRE, GENRE-type, TABi-type-text, and TABi are trained on KILT data (which includes open-domain tasks), while BLINK, ELQ, and Bootleg are trained on Wikipedia entity disambiguation data, and DPR is trained on question answering data. See the ablations for a discussion of the training data impact. On AmbER (GOLD) where all models are trained on Wikipedia entity disambiguation data and mention boundaries are available (Table 2), TABi outperforms baselines on average tail accuracy@1 by 4.4 points. BLINK and Bootleg perform much better on AmbER (GOLD) than on AmbER, suggesting that mention detection introduces significant error.

Overall performance TABi maintains strong performance overall. On AmbER, TABi outperforms all retrievers for average accuracy@1 over the head

[11] We remove examples with gold entities not in KILT-E.

	Fact Check.	Slot Filling		Question Answering				Dial.	
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW	**Avg.**
TF-IDF*	50.9	44.7	60.8	28.1	34.1	46.4	13.7	49.0	41.0
DPR*	55.3	13.3	28.9	54.3	25.0	44.5	10.7	25.5	32.2
Multi-task DPR*	74.5	69.5	80.9	59.4	42.9	61.5	15.5	41.1	55.7
BLINK*	63.7	59.6	78.8	24.5	46.1	65.6	9.3	38.2	48.2
GENRE†	83.6	79.4	95.8	60.3	51.3	69.2	15.8	**62.9**	64.8
KGI**	75.6	74.4	**98.5**	63.7	-	60.5	-	55.4	-
Re2G**	**88.9**	80.7	-	**70.8**	-	**72.7**	-	60.1	-
TABi	84.4	**81.9**	96.2	62.6	**53.1**	70.4	**18.3**	59.1	**65.8**

Table 3: R-precision on KILT open-domain tasks (test data). *Numbers from Petroni et al. (2021). †Numbers from Cao et al. (2021). **Numbers from KILT leaderboard. Best score **bolded** and second best underlined.

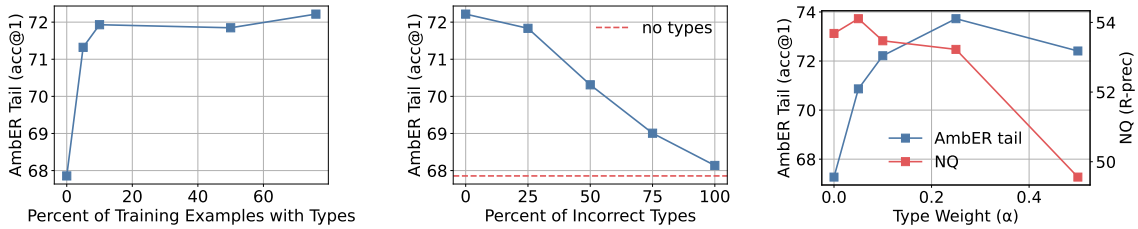

Figure 3: Robustness of TABi to missing types (left) and incorrect types (middle) in the training dataset. Sensitivity of TABi to the type weight α (right).

(Table 1). On AmbER (GOLD), TABi follows Bootleg, which leverages an alias table limiting the number of candidates, and BLINK, which uses an expensive cross-encoder for re-ranking. On KILT, we find that TABi outperforms GENRE, the best performing multi-task retriever[12] overall by 1 point and sets the state-of-the-art on three KILT tasks (Table 3).

Ablations Table 1 reports ablations. First, to measure the impact of types, we remove the type-enforced loss (L_{type}) by setting $\alpha = 0$. This is equivalent to training TABi with just L_{ent}. Compared to full TABi, the average accuracy@1 drops by 5.8 and 3.0 points on the tail and head, demonstrating the importance of the type-enforced loss, particularly over the tail. Moreover, we observe that TABi ($\alpha = 0$) still outperforms the BLINK bi-encoder by 29.4 points over the tail. As the BLINK bi-encoder is trained only on entity disambiguation, this suggests that additionally training over open-domain tasks leads to substantial improvements (see Appendix B.1). Second, we evaluate the impact of using L_{ent} instead of the standard L_{NCE} to compare pairs of queries and entity descriptions based on their gold entity (Section 3). Compared to full TABi (which uses $L_{type} + L_{ent}$), TABi ($L_{type} + L_{NCE}$) incurs an average accuracy@1 drop of 3.5 and 2.8 points over the tail and head, respectively.

[12]TABi and GENRE use a single model across all tasks, whereas KGI (Glass et al., 2021) and Re2G (anonymous), train a separate model for each task.

Robustness to noise We run two experiments to simulate incomplete and noisy type annotations. First, we randomly remove types from a proportion of the training set. Figure 3 (left) shows TABi achieves 79% of the lift on AmbER tail with just 5% type coverage. Second, we randomly flip the types of a proportion of the training set to a type that has no type overlap with the gold type. Figure 3 (middle) shows TABi can still achieve >2 points of lift over no types even when 50% of the types are incorrect. Surprisingly, even 100% incorrect types does not hurt performance over using no types.

Type weight sensitivity Figure 3 (right) shows TABi's sensitivity to the type weight α on the AmbER tail and Natural Questions (NQ) (Kwiatkowski et al., 2019), a task in KILT. We find there can be a tradeoff on some datasets: too small of an α is not sufficient to learn the type from the query context, whereas too large of an α can start to reduce overall performance. To balance this tradeoff, we set $\alpha = 0.1$ in all experiments.

Re-ranking We evaluate (1) whether an inexpensive re-ranker that combines TABi with sparse retrieval and popularity scores can further improve performance, and (2) whether hard negative sampling is necessary when we use a re-ranker. Table 4 shows that re-ranking can improve accuracy@1 over by 1.5 and 0.6 points over the head and tail in AmbER, respectively. Without hard negative sampling, the

Model	Avg. Head	Avg. Tail
TABi	74.3	72.1
TABi ($\alpha = 0$)	71.3	66.3
TABi + RR	75.8	72.7
TABi + RR (no hard negatives)	70.4	70.7

Table 4: Average head/tail accuracy@1 on AmbER when TABi is combined with an inexpensive re-ranker (RR).

Dataset	Model	Acc.	Micro F1	Macro F1
FIGER	BLINK	15.8	40.5	25.1
	TABi	**49.0**	**72.8**	**76.6**
OntoNotes	BLINK	21.5	34.2	42.3
	TABi	**38.6**	**57.3**	**63.3**

Table 5: Mention type classification using a nearest neighbor classifier over query embeddings.

performance of TABi decreases, especially over the head. However, TABi with the re-ranker and no hard negative sampling can still nearly match TABi ($\alpha = 0$)—the strong bi-encoder baseline without types—over the head and outperforms it over the tail, despite TABi ($\alpha = 0$) using hard negative sampling. This suggests that there may be alternatives to hard negative sampling, such as incorporating structured data, for achieving strong performance on some tasks.

5 Embedding Quality Analysis

We evaluate how well TABi captures types through embedding visualization, nearest neighbor type classification, and an entity similarity task.

Embedding visualization We use t-SNE to qualitatively evaluate how well bi-encoders cluster entity embeddings by type. Figure 2 shows that TABi forms tighter type clusters than BLINK for five FIGER types.[13] Types are not captured as well when the type is only present in the input and not the loss. This suggests that our type-based loss term helps encode types in the embedding space.

Type classification To better understand how well embeddings are clustered by type, we evaluate query and entity embeddings using KNN classification with $K = 10$.[14] We use strict accuracy, loose micro F1, and loose macro F1 metrics for evaluation (Zhang et al., 2019). TABi outperforms BLINK on KNN classification over query embeddings on FIGER and OntoNotes, confirming that our loss encourages nearby query embeddings to share the same type (Table 5). Appendix C.2 reports KNN experiments on entity embeddings, where we find TABi outperforms BLINK on KNN classification of both coarse and fine types, confirming our loss also helps the entity embeddings encode types.

[13] We choose popular types with low overlap in entities.

[14] As a query or entity can have multiple types, we cast type classification as a multi-label classification problem.

	TransE	ComplEx	BLINK	**TABi**
Spearman ρ	62.4	63.4	59.4	**68.6**

Table 6: Spearman rank correlation on our proposed entity similarity task over pairs of Wikidata entities.

Entity similarity ranking To understand how well our method learns finer-grained type hierarchies, we create a novel entity similarity task inspired by word similarity tasks (Schnabel et al., 2015). The goal is to rate the similarity of entity pairs, where the pair has a high score if the two entities share a fine type and a lower score otherwise. We assign ground truth similarity scores to 500 entity pairs that share Wikidata types[15] of varying coarseness using a weighted Jaccard similarity metric from the KGTK Semantic Similarity toolkit (Ilievski et al., 2021)[16](see Appendix C.3).

Table 6 compares the Spearman rank correlation of the inner products of BLINK and TABi entity embeddings with the ground truth similarity scores, as well as two popular knowledge graph embeddings, TransE (Bordes et al., 2013) and ComplEx (Trouillon et al., 2016) (for which we use cosine similarities between entity pairs provided by KGTK). TABi outperforms BLINK and the knowledge graph embeddings. This is surprising, since the knowledge graph embeddings are trained on triples which include Wikidata types, whereas TABi is only trained with coarser-grained FIGER types.

6 Discussion

We discuss limitations of TABi. First, we assume a relatively coarse type system is available. To pull together query embeddings of the same type, the type system needs to be sufficiently coarse-grained and the batch size large enough such that multiple examples in a randomly sampled batch have the same type. Second, our method is designed for open-domain tasks, which tend to have short queries and strong type disambiguation signals. However, there are disambiguation signals that may be present in queries,

[15] We use the "instance of" (P31), "subclass of" (P279), and "occupation" (P106) relations to extract types from Wikidata.

[16] https://github.com/usc-isi-i2/kgtk-similarity

such as the existence of a knowledge graph relation between two entities, that TABi does not optimize for learning. To address this, we are interested in incorporating other forms of structured data, including different modalities, into our model as future work.

7 Related Work

Entity disambiguation with types Our work is inspired by prior work that has used types for entity disambiguation (Ling et al., 2015; Gupta et al., 2017; Gillick et al., 2019; Onoe and Durrett, 2020; Chen et al., 2020a; Orr et al., 2021). Most closely related are Gillick et al. (2019) and Gupta et al. (2017). Gillick et al. (2019) train dense retrievers with Wikipedia categories as input, but do not include types in the loss function. On the other hand, Gupta et al. (2017) incorporate types through multi-task learning with type prediction, but rely on alias tables. Generally, prior works that use types assume mention boundaries are given as input. Similar to our work, Gupta et al. (2017), Onoe and Durrett (2020), and Orr et al. (2021) show that using types can improve disambiguation of rare entities. Finally, types have also been shown to improve performance on coreference resolution (Khosla and Rose, 2020) and natural language generation (Dong et al., 2021).

Entity typing A task closely related to our work is entity typing, or predicting the set of types for a mention (e.g., Ling and Weld, 2012; Gillick et al., 2014; Onoe et al., 2021). A key difference is that entity typing methods often learn explicit type embeddings to perform type classification, whereas TABi only learns query and entity embeddings. Entity typing methods could be used to add type labels to the training data as an alternative to TABi's approach that uses a direct knowledge graph type mapping.

Retrieval for open-domain NLP There has been extensive work on dense retrieval for open-domain NLP tasks (e.g. Lee et al., 2019; Karpukhin et al., 2020; Oğuz et al., 2020). However, most prior work has assumed unstructured text as the only input. As an exception, Oğuz et al. (2020) incorporate structured data, such as knowledge graph relations and tables, into dense retrieval by flattening the structured data into text and adding it to the retrieval index. This approach is complementary to TABi, which incorporates the structured data into the loss to learn better representations of the index.

Alternatives to bi-encoders Several works have focused on improving the bi-encoder model by leveraging multiple embeddings for each query or candidate (Humeau et al., 2020; Khattab and Zaharia, 2020; Luan et al., 2021). These approaches are complementary to TABi—which maintains a single embedding for each query and candidate—and may lead to further quality improvements at some computational expense.

8 Conclusion

We introduce a method to train bi-encoders on unstructured text and knowledge graph types through a type-enforced contrastive loss. Our loss can improve retrieval of rare entities for ambiguous mentions, while maintaining strong overall performance on open-domain NLP tasks. We hope our work inspires future work on integrating structured data into pretrained models.

Acknowledgements

We thank Simran Arora, Ines Chami, Neel Guha, Laurel Orr, Maya Varma, Sen Wu, and the anonymous reviewers for their helpful feedback. We gratefully acknowledge the support of NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ARL under No. W911NF-21-2-0251 (Interactive Human-AI Teaming); ONR under No. N000141712266 (Unifying Weak Supervision); ONR N00014-20-1-2480: Understanding and Applying Non-Euclidean Geometry in Machine Learning; N000142012275 (NEPTUNE); NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, Google Cloud, Salesforce, Total, the HAI-GCP Cloud Credits for Research program, the Stanford Data Science Initiative (SDSI), the NSF Graduate Research Fellowship under No. DGE-1656518, the Department of Defense under the National Defense Science and Engineering Graduate Fellowship Program, and members of the Stanford DAWN project: Facebook, Google, and VMWare. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of NIH, ONR, or the U.S. Government.

Broader Impact

We believe that our work has the potential to positively impact underrepresented populations. A key benefit of our method is improved retrieval of rare entities, which infrequently or never occur in the training dataset. Rare entities may not only consist of individuals from underrepresented populations, but may also be entities that are of interest to underrepresented populations (e.g., songs, locations). While we hope our work will have a positive impact, we also caution that our method is susceptible to biases present in standard pretrained language models and large Internet-based training datasets. We finetune our model from a BERT-base pretrained model using BLINK and KILT training datasets, which include content from Wikipedia, Reddit, trivia websites, and crowd-sourced questions and dialogue.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Steven M. Beitzel, Eric C. Jensen, and Ophir Frieder. 2009. *Average R-Precision*, pages 195–195. Springer US, Boston, MA.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7833–7845, Online. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Anthony Chen, Pallavi Gudipati, Shayne Longpre, Xiao Ling, and Sameer Singh. 2021. Evaluating entity disambiguation and the role of popularity in retrieval-based NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4472–4485, Online. Association for Computational Linguistics.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020a. Improving entity linking by modeling latent entity type information. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7529–7537. AAAI Press.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020b. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Xiangyu Dong, Wenhao Yu, Chenguang Zhu, and Meng Jiang. 2021. Injecting entity types into entity-guided text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 734–741, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural

attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Daniel Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *Computing Research Repository*, arXiv:1412.1820.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, and Alfio Gliozzo. 2021. Robust retrieval augmented generation for zero-shot slot filling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1939–1949, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Filip Ilievski, Pedro Szekely, Gleb Satyukov, and Amandeep Singh. 2021. User-friendly comparison of similarity algorithms on wikidata. *ArXiv preprint*, abs/2108.05410.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 39–48. ACM.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Sopan Khosla and Carolyn Rose. 2020. Using type information to improve entity coreference resolution. In *Proceedings of the First Workshop on Computational Approaches to Discourse*, pages 20–31, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6433–6441, Online. Association for Computational Linguistics.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 2021. Multi-task retrieval for knowledge-intensive tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1098–1111, Online. Association for Computational Linguistics.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2051–2064, Online. Association for Computational Linguistics.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8576–8583. AAAI Press.

Laurel Orr, Megan Leszczynski, Neel Guha, Sen Wu, Simran Arora, Xiao Ling, and Christopher Ré. 2021. Bootleg: Chasing the tail with self-supervised named entity disambiguation. In *Conference on Innovative Data Systems Research (CIDR)*.

Barlas Oğuz, Xilun Chen, Vladimir Karpukhin, Stanislav Peshterliev, Dmytro Okhonko, M. Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2020. Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering. *ArXiv preprint*, abs/2012.14610.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, Lisbon, Portugal. Association for Computational Linguistics.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1271–1280, Valencia, Spain. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv preprint*, abs/1807.03748.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Z. Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason R. Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv preprint*, abs/1609.08144.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*,

pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Appendix

A Experimental Setup Details

A.1 Baselines

We use pretrained models for all learned text-only baselines. For fair comparison, we use the KILT-E knowledge base at inference time for all models (see Section 4.1 for details on the knowledge base). We include model parameter counts in Table 7. Note that DPR, BLINK (Bi-encoder), BLINK, ELQ, TABi-type-text, and TABi require an index of embeddings to be stored in addition to the model parameters for fast inference.

Model	# Parameters
Text-only methods	
Alias Table	0
TF-IDF	0
DPR	220M
BLINK (Bi-encoder)	680M
BLINK	1.0B
ELQ	680M
GENRE	406M
Type-aware methods	
Bootleg	1.3B
GENRE-type	406M
TABi-type-text	110M
TABi	110M

Table 7: Number of model parameters.

For Alias Table, we compute the prior probability of a mention-entity pair over the BLINK training dataset.

For TF-IDF, DPR, and BLINK, we use the code provided in the KILT repository.[17] For the BLINK cross-encoder, we use $k = 10$ as the number of retrieved entities passed to the cross-encoder, following the recommended setting in Wu et al. (2020). BLINK uses Flair (Akbik et al., 2019) for mention detection when no mention boundaries are available.

For ELQ, we use the code provided in the ELQ repository.[18] We use the Wikipedia-trained ELQ model and the recommended settings for the Wikipedia model provided in the repository (threshold=-2.9). We find this outperforms the WebQSP-finetuned ELQ model on average on AmbER and KILT.

For Bootleg, we use the code provided in the Bootleg repository.[19] We use the model version from July 2021. Bootleg uses a heuristic n-gram method for mention detection when no mention boundaries are available.

For GENRE, we use the code provided in the GENRE repository.[20] We use the BLINK-trained model for experiments on AmbER (GOLD) and the KILT-trained model for experiments on AmbER and KILT. We use the default settings (beam size=10, context length=384 tokens).

For GENRE-type, we modify GENRE so that instead of just generating the entity name, the model must generate the entity name and type to predict an entity (e.g. "United States country"). First, we use the FIGER types from KILT-E to generate a new set of type-enhanced titles. We then train models for both the AmbER and AmbER (GOLD) settings. For AmbER experiments, we fine-tune from the GENRE KILT-pretrained model for 4 epochs on the KILT dataset. We set max tokens to 8,192 and train on 16 A100s. We sweep the learning rate in {1e-4, 3e-5, 1e-5, 1e-6} and select the best value on the KILT dev set using the macro-average R-precision across the eight open-domain tasks (best learning rate: 1e-6). For AmbER (GOLD) experiments, we fine-tune from the GENRE BLINK-pretrained model for 4 epochs on the BLINK dataset using the same learning rate (1e-6). For both models, we run inference using a trie created over the type-enhanced titles and a maximum output length of 20 tokens.

For TABi-type-text, we use the type as textual input to the entity encoder and no types are used in the loss function. Specifically, we insert the types after the entity title and before the description, separated by a special separator token. We use L_{ent} for training TABi-type-text and use the same training procedure as we use for TABi described in Appendix A.4. We fix the temperature to 0.05 and batch size to 4,096. We sweep the learning rate in {1e-4, 2e-4, 3e-4} for two epochs on the KILT training data and select the best value on the KILT dev set using the macro-average R-precision across the eight open-domain tasks (best learning rate: 2e-4). We use the same learning rate to train a model on the BLINK training data.

For all models, we report a single run.

[17] https://github.com/facebookresearch/KILT

[18] https://github.com/facebookresearch/BLINK/tree/main/elq

[19] https://github.com/HazyResearch/bootleg

[20] https://github.com/facebookresearch/GENRE

A.2 Evaluation datasets

We include statistics on the evaluation datasets described in Section 4.1 in Table 8. We report the head/tail subsets for AmbER as defined in Chen et al. (2021). Note we split AmbER randomly into dev (5%) and test (95%) splits and report results on test. We consider the open-domain tasks in KILT (fact checking, question answering, slot filling, and dialogue) and define the "head" as having a gold entity that is in the top 1% most popular entities by Wikipedia page views and the "tail" as being in the bottom 90% of entities by Wikipedia page views. We evaluate retrieval on eight datasets: FEVER (Thorne et al., 2018), T-REx (Elsahar et al., 2018), Zero Shot RE (Levy et al., 2017), Natural Questions (Kwiatkowski et al., 2019), HotPotQA (Yang et al., 2018), TriviaQA (Joshi et al., 2017), ELI5 (Fan et al., 2019), and Wizard of Wikipedia (Dinan et al., 2019).

A.3 Training data

We include additional details about the training data described in Section 4.1.

Unstructured text In the BLINK training data, each sentence has a single mention labeled with mention boundaries and a gold entity from a Wikipedia anchor link. The KILT training data is a superset of the BLINK training data, that additionally contains sentences from standard fact checking, slot filling, open domain QA, dialogue, and entity disambiguation datasets. With the exception of the entity disambiguation examples, the additional examples have a gold entity label, but no gold mention boundaries.

Knowledge graph types We describe (1) how we assign types to entities, and (2) how we assign types to queries. For both entities and queries, we use the FIGER type set Ling and Weld (2012) for types (i.e., each type label must be one of 113 types in the type set); however, our method is not specific to the FIGER type set and any type set with coarse types may lead to improvements.

Entity type assignments We assign types to entities via a direct mapping of entities to knowledge graph types. First, the majority of the entities in KILT-E have a unique QID in Wikidata. For these entities, we use a mapping from Wikidata to Freebase using the "P646" property in Wikidata. After finding the corresponding Freebase entity, we derive the FIGER types from its Freebase types, using the map from Ling and Weld (2012).

Query type assignments We follow Ling and Weld (2012) to assign types to queries through distant supervision. Specifically, we assign the types of the gold entity for the query as the types of the query. Thus we use a direct mapping of entity types from a knowledge graph, rather than use a probabilistic type classifier. Note that assigning query types through distant supervision (with the gold entity types) can be a noisy assumption. For instance, consider the query *"What was the outcome of the election for Arnold Schwarzenegger?"* with the gold entity Arnold Schwarzenegger. The query only implies that Schwarzenegger is a politician with the keyword "election". However, all types of the gold entity Arnold Schwarzenegger would be assigned to the query (e.g. "actor", "body builder", assuming the types were in the type set). As not all types associated with the gold entity may be implied by the query, this method can add noise to the query type labels.

Type statistics We are able to assign types to 73% of examples in the BLINK training data and 76% of examples in the KILT training data. In the BLINK training data, the average example with types has 2.1 types with a max of 9 types. In KILT training data, the average example with types has 2.0 types with a max of 9 types.

A.4 Training procedure

We describe the training procedure for TABi. We tie the query and entity encoders (i.e. use a single encoder) and initialize from a BERT-base pretrained model (Devlin et al., 2019). Following BLINK's protocol (Wu et al., 2020), we set the maximum context length to 32 tokens and the maximum entity description length to 128 tokens. We set the batch size to 4,096 and use the AdamW optimizer (Loshchilov and Hutter, 2019) and decay the learning rate by 50% every epoch.

We use balanced hard negative sampling, following Botha et al. (2020). Specifically, we only allow ten negative examples of an entity for each positive example in the training dataset. For all models of TABi, we train the first epoch using local in-batch negatives, and we mine for hard negatives at the end of every epoch. Starting at the second epoch, we train with both in-batch and hard negatives.

From results on preliminary experiments, we fix the temperature=0.05 and the type weight $\alpha = 0.1$. We then conduct a grid search for the initial learning rate by training for two epochs on the KILT training

Benchmark	Dataset	Dev			Test			Type of Queries
		Total	# Head	# Tail	Total	# Head	# Tail	
AmbER	Human FC	594	284	310	11,290	5,054	6,236	Templated claims
	Non-human FC	1,369	728	641	26,017	13,500	12,517	Templated claims
	Human SF	297	138	159	5,645	2,531	3,114	Subject-relation facts
	Non-human SF	684	355	329	13,009	6,759	6,250	Subject-relation facts
	Human QA	297	123	174	5,645	2,546	3,099	Templated questions
	Non-human QA	684	343	341	13,009	6,771	6,238	Templated questions
KILT	FEVER	10,444	6,406	614	10,100	-	-	Mutated Wikipedia claims
	T-REx	5,000	35	4,553	5,000	-	-	Subject-relation facts
	Zero Shot RE	3,724	111	2,974	4,966	-	-	Subject-relation facts
	Natural Questions	2,837	1,444	204	1,444	-	-	Search engine questions
	HotpotQA	5,600	2,115	797	5,569	-	-	Crowd-sourced questions
	TriviaQA	5,359	3,747	223	6,586	-	-	Trivia questions from trivia sites
	ELI5	1,507	644	168	600	-	-	Reddit questions
	Wizard of Wikipedia	3,054	1,963	142	2,944	-	-	Crowd-sourced dialogue

Table 8: Evaluation dataset statistics.

data and selecting the best value on the KILT dev set using the macro-average R-precision across the eight open-domain tasks. We sweep the initial learning rate in {1e-4, 2e-4, 3e-4} (best learning rate=3e-4).

We use the same hyperparameter configuration for training on both the BLINK training data and the KILT training data. For models trained on BLINK data and KILT data, we train for 4 epochs using 16 A100 GPUs (approximately 2.2 hours/epoch for BLINK training data, 2.6 hours/epoch for KILT training data, including sampling for hard negatives).

A.5 Re-ranking details

While our standard configuration of TABi does not use a re-ranker, we explore using an inexpensive re-ranker on top of TABi. The re-ranker consists of two steps: first, it linearly combines the top-K entity scores from the bi-encoder with the top-K entity scores of a sparse retriever using a tunable weight λ. Second, it linearly combines these scores with their corresponding global entity popularity (e.g. Wikipedia page views) using a tunable weight κ. We normalize scores before linearly combining at each step.

More formally, let E be union of the set of retrieved entities from the bi-encoder and the sparse retriever. Then for an entity $e \in E$, where s_e indicates the score from the sparse retriever, d_e indicates the score from the dense retriever, and p_e indicates the popularity score, we compute the re-ranked score f_e as follows:

$$h_e = \lambda s_e + d_e$$
$$f_e = \kappa p_e + h_e$$

We use the baseline TF-IDF retriever for the sparse retriever (see Appendix A.1 for details). Like Chen et al. (2021), we use the monthly Wikipedia page views (from October 2019) as the measure of global entity popularity. Note that tuning these weights does not require re-training or re-running the bi-encoder evaluation.

For the experiments with the re-ranker, we tune λ and κ on each of the 6 dev sets for AmbER by first selecting λ that performs best on the linear combination of the bi-encoder and sparse retriever scores, and then fixing λ and tuning κ. For both λ and κ, we sweep in $\{0.0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$.

B Extended Retrieval Results

B.1 AmbER results

We extend the results on AmbER included in Section 4. First, we perform experiments to better understand the strong performance of the baseline TABi ($\alpha = 0$), which removes the type-based loss term. We primarily attribute the strong performance of TABi ($\alpha = 0$) relative to the BLINK (Bi-encoder) to the training data and perform baseline ablations in Table 9. We see that BLINK (Bi-encoder) and TABi ($\alpha = 0$) perform similarly when both are trained on BLINK data, which consists of Wikipedia entity disambiguation data. Training on the KILT data, which additionally includes multiple open-domain tasks, leads to significant

	Fact Checking				Slot Filling				Question Answering				Average	
	H		N		H		N		H		N			
Model	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
BLINK (Bi-encoder) + Flair	56.4	52.0	24.8	10.5	76.8	55.7	30.7	13.5	78.3	55.7	67.3	33.8	55.7	36.9
TABi ($\alpha=0$, BLINK data) + Flair	45.6	49.7	4.6	2.9	77.4	58.1	48.4	30.4	78.0	58.0	63.4	39.9	52.9	39.8
TABi ($\alpha=0$, KILT data) + Flair	44.7	44.9	7.5	7.4	80.0	84.0	74.5	73.6	83.4	70.2	77.6	56.5	61.3	56.1
TABi ($\alpha=0$, KILT data)	77.6	61.9	41.4	39.1	70.9	87.1	83.2	85.9	72.5	66.3	82.2	57.7	71.3	66.3

Table 9: Retrieval accuracy@1 on AmbER (H for human, N for non-human subsets). Impact of the training data on bi-encoder performance.

	Fact Checking				Slot Filling				Question Answering				Average	
	H		N		H		N		H		N			
Model	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
TF-IDF	76.4	76.1	60.9	60.6	80.4	82.9	52.6	50.0	78.1	82.3	58.9	54.2	67.9	67.7
DPR	47.9	27.9	72.6	43.2	34.0	14.0	74.3	43.6	46.0	22.2	77.5	45.4	58.7	32.7
BLINK (Bi-encoder)	89.5	90.1	81.5	71.6	94.5	95.9	48.9	41.2	94.9	95.8	90.9	86.3	83.4	80.1
BLINK	91.1	85.8	83.9	76.3	94.1	95.2	49.3	41.5	94.9	95.8	91.2	86.6	84.1	80.2
ELQ	78.4	61.1	66.8	37.2	74.5	44.1	59.7	27.1	77.5	47.2	62.1	30.7	69.8	41.2
GENRE	78.0	67.9	82.8	77.4	86.9	92.5	90.7	90.8	83.7	83.7	87.4	82.7	84.9	82.5
Bootleg†	98.3	97.6	69.9	65.7	96.5	93.6	66.8	56.2	97.1	96.7	74.8	76.3	83.9	81.0
GENRE-type	71.2	80.0	76.4	77.7	73.6	92.7	91.0	92.2	83.0	90.5	91.3	91.4	81.1	87.4
TABi-type-text	90.9	83.7	84.5	77.9	89.2	95.9	96.2	98.2	86.0	88.2	95.1	92.9	90.3	89.5
TABi	95.0	93.8	79.9	80.3	91.3	96.8	96.4	98.3	91.6	93.8	95.8	95.7	91.7	93.1

Table 10: Retrieval accuracy@10 on AmbER (H for human, N for non-human subsets). †Models with an alias table.

	FC		SF		QA		
Model	H	N	H	N	H	N	**Avg.**
TF-IDF	1.0	0.6	2.5	2.5	2.5	2.5	1.9
DPR	0.2	3.8	1.2	10.7	2.3	12.2	5.1
BLINK (Bi-enc)	9.4	0.7	36.1	6.4	35.9	20.5	18.2
BLINK	5.4	0.0	17.6	8.6	27.7	29.7	14.8
ELQ	3.9	0.0	24.7	12.4	29.6	16.2	14.5
GENRE	4.3	1.0	28.3	39.2	10.9	13.9	16.3
Bootleg	3.0	0.0	26.7	15.5	31.6	27.8	17.4
GENRE-type	3.3	7.4	17.2	50.9	15.8	28.6	20.5
TABi-type-text	17.3	2.1	60.0	69.1	40.9	44.8	39.0
TABi	40.0	4.2	65.6	74.3	53.6	52.6	48.4

Table 11: Consistency results on AmbER for top-1. The consistency is the fraction of mentions where all queries for a mention are correct.

lift. Removing the mention detector, Flair, leads to additional lift. Note that TABi ($\alpha=0$) can retrieve entities without mention detection since the KILT training data includes open-domain tasks which do not have mention boundaries.

Second, we include results for top-10 retrieval accuracy (accuracy@10) on AmbER to understand the retrieval performance at larger K (Table 10). We find that TABi continues to outperform baselines on average.

Finally, we report results for the consistency metric introduced in Chen et al. (2021) for top-1 retrieval in Table 11. This metric measures the proportion of mentions where all queries for the mention are correct. In particular, Chen et al. (2021) found that retrievers have a tendency to "collapse" all predictions for a mention to the most popular entity for the mention, which would result in a low consistency value. We find that TABi outperforms all models on this metric.

B.2 KILT results

We include R-precision results on the KILT dev sets for the tasks and baselines in the main paper in Table 12. As with the AmbER experiments, we use the KILT-E knowledge base for inference for all models. We see that GENRE, TABi-type-text, and TABi outperform the other baselines across the tasks, and perform comparably overall to each other. Recall that GENRE, GENRE-type, TABi-type-text, and TABi were trained on KILT training data. BLINK, ELQ, and Bootleg were trained on Wikipedia training data and DPR was trained on question answering data. GENRE-type performs substantially worse than GENRE overall, suggesting that incorporating types in the entity name degrades overall retrieval performance.

We also report results on the KILT test and

	Fact Check.	Slot Filling		Question Answering				Dial.	
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW	**Avg**
TF-IDF	48.4	57.4	72.8	20.1	43.4	27.8	4.6	38.8	39.2
DPR	57.0	14.9	44.3	54.5	25.5	46.2	16.1	26.9	35.7
BLINK (Bi-encoder)	64.4	59.4	84.3	35.1	43.1	61.6	11.3	26.0	48.2
BLINK	67.6	61.0	87.4	33.5	47.9	65.9	9.7	26.5	49.9
ELQ	65.1	71.2	95.0	42.4	45.9	67.7	9.2	26.8	52.9
GENRE	85.0	80.5	95.1	61.4	51.9	71.4	13.6	56.5	64.4
Bootleg†	62.3	69.4	81.8	34.5	43.6	53.1	9.7	28.2	47.8
GENRE-type	55.3	71.9	80.6	54.5	37.2	53.6	11.5	44.5	51.1
TABi-type-text	87.3	82.2	95.1	62.5	51.2	70.8	16.9	51.0	64.6
TABi	85.8	82.0	95.2	62.4	52.7	71.5	16.7	51.8	64.8

Table 12: R-precision on KILT open-domain tasks (dev data). (Top) text-only methods and (bottom) type-aware methods. †Models with an alias table.

	Fact Check.	Slot Filling		Question Answering				Dial.	
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW	**Avg.**
TF-IDF	-	-	-	-	-	-	-	-	-
DPR	74.3	17.0	39.2	65.5	10.4	57.0	26.9	51.2	42.7
Multi-task DPR	87.5	83.9	93.1	68.2	28.4	68.3	27.5	67.1	65.5
BLINK	-	-	-	-	-	-	-	-	-
GENRE	88.2	85.3	97.8	61.4	34.0	75.1	25.5	77.7	68.1
KGI	85.0	83.1	99.2	70.2	-	63.5	-	78.5	-
Re2G	92.5	89.0	-	76.6	-	74.2	-	80.0	-
TABi	88.6	89.4	98.7	64.9	35.5	69.2	28.2	69.1	67.9

Table 13: Recall@5 on KILT open-domain tasks (test data). We report numbers from Petroni et al. (2021) and the KILT leaderboard where available.

	Fact Check.	Slot Filling		Question Answering				Dial.	
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW	**Avg.**
TF-IDF	71.8	73.0	88.6	32.6	29.2	41.0	9.7	56.5	50.3
DPR	76.0	22.3	59.2	63.9	11.1	57.4	31.0	52.7	46.7
BLINK (Bi-encoder)	80.0	68.1	88.4	40.8	24.3	63.5	19.4	40.9	53.2
BLINK	82.9	69.6	89.6	43.7	27.4	66.9	22.3	44.6	55.9
ELQ	79.5	69.9	95.2	36.1	23.7	62.4	9.5	47.7	53.0
GENRE	89.0	85.3	97.3	58.5	34.7	75.7	20.5	75.0	67.0
Bootleg†	81.0	74.3	85.6	37.2	26.3	69.4	14.0	49.3	54.6
GENRE-type	66.9	80.1	89.7	54.4	23.4	58.4	18.5	62.4	56.7
TABi-type-text	90.6	89.1	98.0	63.4	34.1	71.3	25.9	64.6	67.1
TABi	89.3	88.8	98.3	63.1	34.2	70.0	25.6	64.8	66.8

Table 14: Recall@5 on KILT open-domain tasks (dev data). (Top) text-only methods and (bottom) type-aware methods. †Models with an alias table.

dev sets for recall@5. In addition to R-precision, recall@5 is reported on the KILT leaderboard and measures the proportion of gold entities for a query[21] that occur in the top-5 ranked entities. If there is a single gold entity, this is equivalent to accuracy@5. We find similar trends as seen with R-precision: TABi, TABi-type-text, and GENRE continue to have strong performance and outperform other baselines (Table 13 (test) and Table 14 (dev)).

B.3 Impact of batch size

We study the impact of the batch size on TABi by training on a 1M random sample of KILT training data for two epochs for batch sizes in {256, 512, 1024, 2048, 4096}. We hold all other hyperparameters constant. As we add n hard negative samples to the batch in the second epoch, the batch size in terms of the number of queries is reduced by a factor of $n+1$. Concretely, if the base batch size is 4,096 exam-

[21] The KILT benchmark supports multiple gold entities for a query.

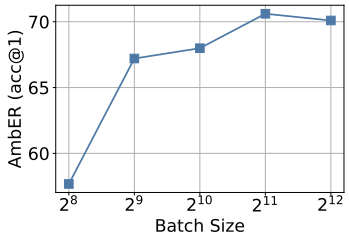

Figure 4: Accuracy@1 on AmbER for varying batch sizes.

ples and we use three hard negatives per query, each batch in the first epoch has 4,096 queries, while each batch in the second epoch has 1,024 queries. We define the batch size in terms of the number of queries in the first epoch. In Figure 4, we see that generally increasing the batch size improves the average accuracy@1 on AmbER (averaged over head and tail examples). However, we caution that this study is performed with all other hyperparameters held constant and a more optimal hyperparameter configuration may exist at different batch sizes. We use a batch size of 4,096 for all experiments in the main paper.

B.4 Type equivalence

We experiment with three type equivalence measures: (1) *Any-types*: two entities have equivalent types if any types overlap, (2) *All-types*: two entities have equivalent types if all types overlap, and (3) *gt50-types*: two entities have equivalent types if at least 50% of the types overlap. If the entities have an unequal number of types, then we take 50% of the greater number of types. An example of (3) is a query A with the types [“musician”, “person”] would be considered as having equal types to query B with the types [“musician”, “person”, “author”], since more than 50% of types of query B overlap with query A.

Our main experiments currently use approach (3), which is intuitively a softer equivalence than (2). However, interestingly we find (2) and (3) can have very similar performance, and both greatly outperform (1). We report the average top-1 accuracy results of training the three methods for 2 epochs on a 1M random sample of KILT in Table 15.

	Avg. Head	Avg. Tail
Any-types	67.9	63.0
All-types	71.0	69.8
gt50-types	71.0	69.0

Table 15: Top-1 accuracy on AmbER for different type equivalence measures.

C Extended Embedding Quality Analysis

C.1 Nearest neighbor mention type classification

We include additional details on the datasets used for mention type classification (experiments in Section 5). The FIGER test set has 563 examples and uses the 113 FIGER type taxonomy (Ling and Weld, 2012). We use the subset of the OntoNotes test set from Shimaoka et al. (2017) that removes pronominal mentions. We further remove examples that map to the "other" type, resulting in a final OntoNotes test set with 3,066 examples. The classifier uses 50 types from the OntoNotes type taxonomy (Gillick et al., 2014) across the sampled training set and the final test set. While the training sets use distant supervision to label mentions with types over Wikipedia and news reports, respectively, both test sets consist of manually annotated mentions in news reports.

C.2 Nearest neighbor entity type classification

We include the setup and results for the entity type classification task from Section 5. We create two datasets for entity type classification using the KILT-E knowledge base: Coarse-types and Fine-types. We use the seven coarse types in the FIGER type system as the coarse types and take the other types as fine types. We create the Coarse-types dataset by sampling without replacement 3,000 entities that correspond to the seven coarse FIGER types: "location", "person", "organization", "product", "art", "event", and "building". We divide the sampled entities into training and test sets for a total of 16,781 training examples and 4,195 test examples. Similarly, we create the Fine-types dataset by sampling without replacement 300 entities that correspond to the FIGER fine types. We discard fine types that do not have at least 300 entities, leaving 100 fine types. We then divide the sampled entities into training and test sets for a total of 23,884 training examples and 5,968 test examples.

Table 16 reports the results for entity type classification. We find that TABi outperforms BLINK, suggesting that our loss helps cluster entities by type in the embedding space.

C.3 Entity similarity task

We describe how we construct the dataset for the entity similarity task. We first find the closure of all Wikidata types assigned to each entity in the

Dataset	Model	Acc.	Micro F1	Macro F1
Coarse-types	BLINK	81.1	89.0	84.1
	TABi	**92.7**	**95.9**	**95.9**
Fine-types	BLINK	71.6	82.0	77.5
	TABi	**76.6**	**86.8**	**84.0**

Table 16: Entity type classification using a nearest neighbor classifier over entity embeddings.

KILT-E knowledge base. We then bucket Wikidata types by the frequency with which they occur in the KILT-E knowledge base (using five buckets). To include types of varying frequencies, we randomly sample 10 Wikidata types from each bucket (50 types total). Finally, we sample 10 pairs of entities for each type for a total of 500 entity pairs.

To assign "ground-truth" similarity values to each entity pair, we submit the entity pairs to the KGTK Semantic Similarity toolkit web API.[22] We use the Jaccard similarity metric returned by the toolkit as the ground-truth similarity. This metric assigns larger values if the types shared by two entities are more specific (i.e. fine-grained). As ground truth values are assigned automatically, there is some noise in the dataset. However, we observe that the trends on the entity similarity task generally follow the trends on the other embedding quality analysis tasks.

[22] https://github.com/usc-isi-i2/kgtk-similarity