TABi: Type-Aware Bi-encoders for End-to-End Entity Retrieval

Anonymous ACL submission

Abstract

Entity retrieval—retrieving information about entities in a query—is a core step in open-domain tasks, such as question answering or fact checking. However, state-of-the-art entity retrievers struggle to retrieve rare entities in queries. There are two key challenges: (1) most retrievers are trained on unstructured text about entities and ignore structured data about entities that can be challenging to learn from text, such as entity types, and (2) methods that leverage structured types are not designed for end-to-end retrieval, which is necessary for open-domain tasks. In this work, we introduce TABi, a method to jointly train bi-encoders on unstructured text and structured types for end-to-end retrieval. TABi uses a type-enforced contrastive loss to encode type information in the embedding space and trains over datasets from multiple open-domain tasks to learn to retrieve entities. We demonstrate that this simple method can improve retrieval of rare entities on the AmbER sets, while maintaining strong overall performance on retrieval for open-domain tasks when compared to state-of-the-art retrievers. We also find that TABi produces embeddings that better capture types on a nearest neighbor type classification and an entity similarity task.

1 Introduction

Entity retrieval (ER) is the process of finding the most relevant entities in a knowledge base for a natural language query.[1] Entity retrieval is crucial for open-domain tasks, where systems are provided with a query without the context needed to answer the query (Karpukhin et al., 2020). For example, 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—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 struggle to resolve ambiguous mentions of rare "tail" entities (Chen et al., 2021).

A key challenge is that most entity retrievers are trained on unstructured text about entities, such as mention contexts and entity descriptions (Wu et al., 2020; Cao et al., 2021). These methods overlook structured data about entities that can be challenging to learn from unstructured text alone—such as entity types, which group similar entities together under a category (e.g. athlete). As a result, retrievers make mistakes even when the type is clear from the query, e.g. retrieving George Washington the president when the query is asking about an athlete.

While several works (e.g. Gupta et al., 2017; Onoe and Durrett, 2020; Orr et al., 2021) have successfully leveraged types to improve tail performance, they require mention boundaries indicating the location of the mention in the query. However, mention boundaries are usually unknown in open-domain tasks. Thus, using these methods on open-domain tasks requires a mention detection stage, which can introduce additional errors.[2]

In this work, we introduce TABi, a simple method for training entity retrievers on structured types and unstructured text for end-to-end retrieval without mention boundaries. TABi uses a bi-encoder model, building on dense retrieval methods (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. TABi adds a type-enforced loss term that pulls

[1] Following Chen et al. (2021), we focus on the page-level document retrieval setting—where the the mention boundaries are unknown and documents correspond to entities (e.g. Wikipedia pages).

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

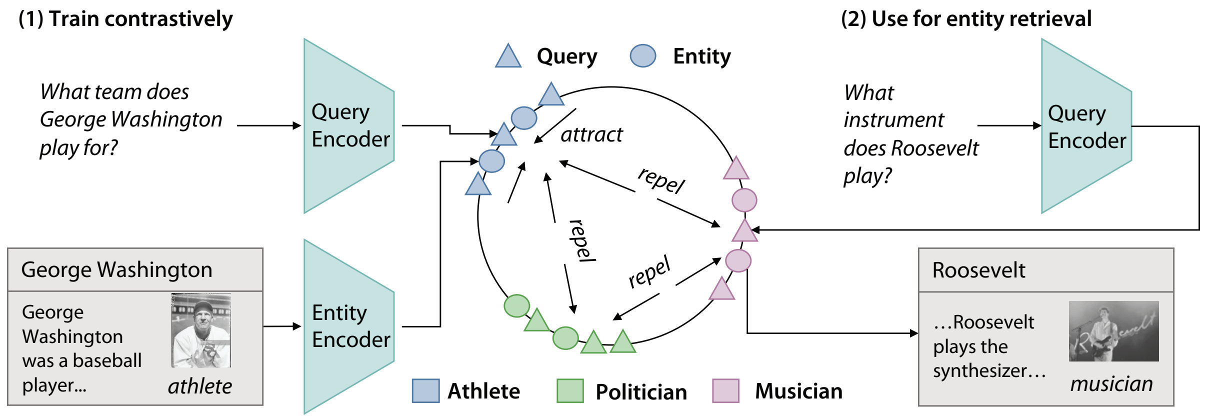

Figure 1: TABi uses a query and entity encoder to embed queries and entities in the same space. To encourage embeddings of the same type (e.g. athlete) to be close, TABi introduces a type-enforced contrastive loss that pulls query embeddings of the same type together and pushes query embeddings of different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

query embeddings of the same type together and pushes query embeddings of different types apart. Additionally, motivated by "universal" dense retrievers (Maillard et al., 2021), TABi trains over multiple open-domain datasets to support end-to-end retrieval. Finally, while the type-enforced loss improves performance over rare entities by encouraging the retriever to pay attention to type context, it may compromise performance over popular entities. We find that a simple re-ranker with two non-learned components—a sparse retriever and popularity statistics—helps maintain performance over popular entities.

We demonstrate that TABi can improve rare entity retrieval for three open-domain tasks (question answering, fact checking, and slot filling), while still performing strongly overall. TABi improves the top-1 retrieval accuracy by 7.5 points on average on the tail AmbER sets (Chen et al., 2021) when mention boundaries are known and by 32.1 when they are unknown, while performing comparably to the state-of-the-art GENRE (Cao et al., 2021) retriever on the open-domain tasks in the KILT benchmark (Petroni et al., 2021). Our method achieves this lift *without* hard negative sampling, which is commonly thought to be critical for bi-encoders (Gillick et al., 2019; Karpukhin et al., 2020) but increases training time[3] and can degrade rare entity performance (Botha et al., 2020).

We also validate that TABi better encodes types in the dense embedding space than baseline dense entity retrievers through embedding visualization, nearest neighbor type classification, and a novel entity similarity task. Surprisingly, on the entity similarity task, which requires learning finer-grained type hierarchies, TABi is even competitive with knowledge graph embedding methods. Code will be released upon publication.

To summarize, our contributions are as follows:

- We introduce TABi, a simple method that jointly uses structured types and unstructured text to train bi-encoders for end-to-end retrieval through a new type-enforced contrastive loss.
- We demonstrate that TABi can improve retrieval of rare entities for open-domain NLP tasks, while maintaining strong overall performance.
- We validate that our approach can better capture types in query and entity embeddings than baseline dense entity retrievers.

2 Preliminaries

We review the problem setup (§2.1), entity retrieval task (§2.2), and bi-encoder model (§2.3).

2.1 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 $D = \{(q_i, e_i, y_i, t_i)\}_{i=1}^{n}$. Similar to augmentations in contrastive learning (Chen et al., 2020), for a query-entity pair (q, e), we consider the query q as a "view" of the entity description e.

[3] We find that adding just one hard negative for each example increases the time per epoch by 1.7×.

[4] To simplify notation, we define a single type label. In experiments, we define the type label as the set of types assigned to the entity and type equivalence as all types matching.

2.2 Entity retrieval task

Given a query q as input, the task of entity retrieval is to return the top-K entity candidates relevant to the query from $\mathcal{Y}$. As $|\mathcal{Y}|$ is often on the order of millions, it is important for entity retrieval systems to be scalable. 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.

2.3 Bi-encoders for entity retrieval

The bi-encoder model consists of a query encoder $f : \mathcal{Q} \to \mathbb{R}^d$ and an entity encoder $g : \mathcal{E} \to \mathbb{R}^d$. 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:

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

where $\psi(v, w) = \exp(f(v)^\top g(w)/\tau)$ is the similarity score between the embeddings of v and w, and τ is a temperature hyperparameter. Intuitively, 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. Note that batches are often constructed with hard negative samples to improve overall quality (e.g. Gillick et al., 2019). In this work, we introduce a new loss for training bi-encoders and compare against the InfoNCE loss in §4 and §5.

3 Approach

TABi jointly leverages structured types and unstructured text to train bi-encoders for end-to-end entity retrieval. TABi is a bi-encoder that takes as input queries and entity descriptions (§3.1) and uses a type-enforced contrastive loss (§3.2). At inference, TABi uses nearest neighbor search to retrieve entities followed by an inexpensive re-ranker (§3.3).

3.1 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 and simply the query tokens if they are unknown (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 WordPiece tokens of the entity's title, types, and a description, with each component separated by an `[E_s]` token (following Wu et al. (2020) and additionally including types in the input). 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., 2020), we then apply L2 normalization to the embeddings.

3.2 Type-Enforced Contrastive Loss

We propose a contrastive loss that incorporates structured 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 usually 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:

$$L_{TABi}(Q, E) = \alpha L_{type}(Q) + (1 - \alpha) L_{ent}(Q, E), \quad (2)$$

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 as a query q and $N_{type}(q)$ be the set of all queries in a batch with a different type than q. Then

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

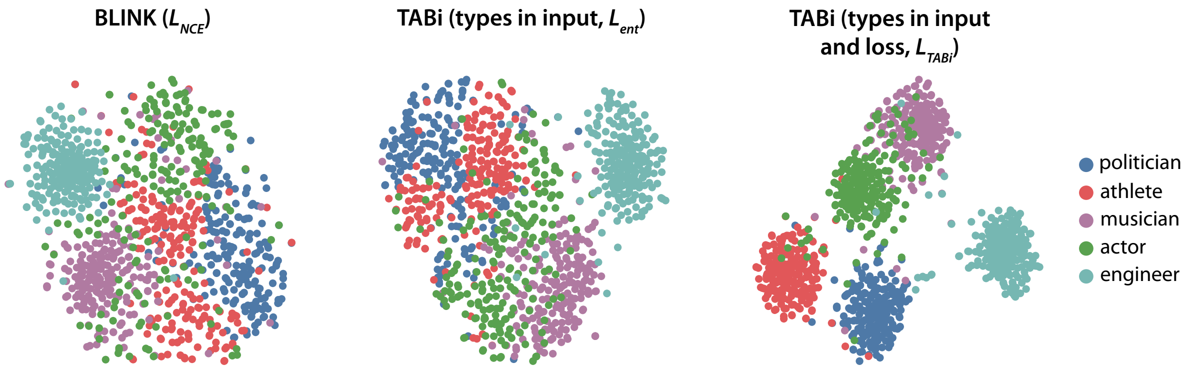

Figure 2: t-SNE visualizations of BLINK and TABi entity embeddings.

$L_{type}(Q)$ is:

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

Next, $L_{ent}(Q, E)$ 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 as x (excluding x itself). Let $N_{ent}(x)$ be the set of all queries and entity descriptions in a batch with a different gold entity from x. Then $L_{ent}(Q, E)$ is:

$$L_{ent}(Q, E) = \frac{-1}{|Q \cup E|} \sum_{x \in Q \cup E} \frac{1}{|P_{ent}(x)|} \sum_{x^+ \in P_{ent}(x)} \log \frac{\psi(x, x^+)}{\psi(x, x^+) + \sum_{x^- \in N_{ent}(x)} \psi(x, x^-)}. \quad (4)$$

Note that 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/entities.[7] We also normalize embeddings before computing ψ. TABi only forms negative pairs over examples in a random batch and does not use hard negative sampling.

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) or L_{ent} on its own (even when the entity input includes types).

3.3 Inference

We precompute entity embeddings and use nearest neighbor search to retrieve the top-K most similar entity embeddings to a query embedding. 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). Similarly, TABi 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. It then linearly combines these scores with their corresponding global entity popularity (e.g. Wikipedia page views) using a tunable weight κ. While this introduces two hyperparameters (λ and κ), they are inexpensive to tune since the bi-encoder does not depend on them.

4 Retrieval Experiments

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

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 eight baselines. Two baselines are non-learned: Alias Table (prior), an alias table which sorts candidates by their prior probability with the mention computed over the

[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 §4.4.

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

BLINK training dataset, and TF-IDF, which uses sparse embeddings of normalized word frequencies. We compare against BLINK (Wu et al., 2020), the state-of-the-art dense entity retriever, and GENRE (Cao et al., 2021), an autoregressive retriever that generates the full entity name from the mention. We also compare against ELQ (Li et al., 2020), which finetunes the BLINK bi-encoder jointly with mention detection and entity disambiguation tasks, and DPR (Karpukhin et al., 2020), which mirrors our query input when there are no mention boundaries. Finally, we include two re-rankers: BLINK with a cross-encoder to re-rank the top 10 candidates from the bi-encoder, and Bootleg (Orr et al., 2021), a Transformer-based model that re-ranks candidates from an alias table using types and knowledge graph relations. BLINK uses Flair (Akbik et al., 2019) for mention detection and Bootleg uses a heuristic n-gram method for mention detection. We use pretrained models for baselines and include more details in Appendix A.1.

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 rare entities in the challenging setting where mentions are ambiguous, and KILT evaluates overall retrieval performance. See Appendix A for dataset statistics.

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), tune our re-ranker on each dev set, and report on the test set.

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 standard 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 (e.g. provide evidence for FC or contain the answer for QA).

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 numbers for KILT baselines and refer the reader to Petroni et al. (2021) for details on the baselines.

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.[8]

Training data We train two versions of TABi. 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 Wikipedia sentences.[9] For end-to-end retrieval experiments, we follow Cao et al. (2021) and train on all KILT training data (which includes data for open-domain tasks) and contains 11.7M sentences (Petroni et al., 2021).

For type labels, we use the FIGER (Ling and Weld, 2012) type system with 113 types. Similar to Ling and Weld (2012), we add the types of the gold entity for each example as the type labels. While types can be incomplete and may not occur

[8]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.

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

	Fact Checking				Slot Filling				Question Answering					
	H		N		H		N		H		N		Average	
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**	<u>23.7</u>	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	<u>52.0</u>	24.8	10.5	**76.8**	<u>55.7</u>	30.7	13.5	<u>78.3</u>	<u>55.7</u>	67.3	33.8	55.7	36.9
BLINK	55.8	45.8	7.4	3.9	<u>74.7</u>	30.3	32.1	16.1	**83.8**	43.8	<u>71.3</u>	<u>44.5</u>	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
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	<u>59.9</u>	30.7	32.6	19.9	67.1	52.6	<u>72.9</u>	<u>59.5</u>	62.9	28.4	61.1	32.4	<u>59.4</u>	<u>37.2</u>
TABi	**75.0**	**76.5**	<u>38.5</u>	**41.9**	72.8	**82.9**	**80.0**	**77.3**	74.5	**80.5**	**79.5**	**56.7**	**70.0**	**69.3**

Table 1: Accuracy@1 on AmbER. H refers to the human subset and N refers to the non-human subset. †Models with an alias table. Best score **bolded** and second best <u>underlined</u>.

	Fact Checking				Slot Filling				Question Answering					
	H		N		H		N		H		N		Average	
Model	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail	Head	Tail
Alias Table (Prior)†	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	23.4	20.8
BLINK (Bi-encoder)	77.5	66.5	77.0	46.0	76.9	55.9	63.8	29.9	78.4	<u>55.8</u>	71.0	34.8	74.5	48.2
BLINK	<u>81.8</u>	61.0	<u>81.6</u>	<u>58.5</u>	75.4	30.5	64.8	35.7	<u>83.8</u>	43.9	<u>74.9</u>	45.7	<u>76.8</u>	45.9
Bootleg†	**83.0**	<u>70.7</u>	**82.1**	56.6	**84.9**	<u>58.8</u>	**76.1**	**54.7**	**86.3**	51.2	**79.2**	**56.5**	**82.4**	<u>58.1</u>
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	70.2	39.6
TABi	81.7	**84.4**	79.9	**64.2**	<u>77.4</u>	**79.0**	<u>66.0</u>	<u>36.5</u>	75.6	**79.0**	69.8	<u>50.4</u>	75.4	**65.6**
Ablations														
TABi ($\alpha = 0$)	82.1	81.3	72.2	53.2	72.6	77.8	61.8	26.6	75.2	77.4	58.7	35.1	70.8	58.6
TABi ($\mathcal{L}_{NCE}$)	81.6	84.4	79.1	63.6	76.8	77.6	65.0	34.4	75.5	79.2	67.7	47.5	74.7	64.5
TABi (no re-ranker)	77.7	85.6	77.0	59.5	72.9	80.6	60.7	40.7	77.0	80.8	68.7	50.5	72.4	66.3

Table 2: Accuracy@1 on AmbER (GOLD) (includes mention boundaries). All models are trained on Wikipedia data. H refers to the human subset and N refers to the non-human subset. †Models with an alias table. Best score **bolded** and second best <u>underlined</u> (excluding ablations).

in the query, we find the type labels are sufficient for improving the type embedding quality in §5.

4.2 Results on rare entities

We find TABi can improve retrieval of rare entities for ambiguous mentions. On AmbER, TABi improves average tail accuracy@1 by 32.1 points in the end-to-end setting compared to baselines (Table 1). Note that GENRE and TABi are trained on KILT data (which includes open-domain tasks), while BLINK, ELQ, and Bootleg are trained on Wikipedia entity linking data, and DPR is trained on question answering data. We then compare on AmbER (GOLD) where all models are trained on Wikipedia entity linking data and mention boundaries are available (Table 2). TABi outperforms baselines on average tail accuracy@1 by 7.5 points. BLINK and Bootleg perform much better on AmbER (GOLD) than on AmbER, suggesting that mention detection introduces significant error.

4.3 Overall performance results

We find that TABi has strong overall performance. On AmbER, TABi outperforms all retrievers for accuracy@1 over the head (Table 1). On AmbER (GOLD), TABi outperforms GENRE and the BLINK (bi-encoder) on the head, despite not using hard negative sampling (Table 2). Bootleg, which leverages an alias table, has the top performance on the head on AmbER (GOLD). On KILT, we find that TABi nearly matches GENRE across the tasks, outperforming GENRE on three tasks (Table 3). Appendix B.2 reports results for our baselines on the KILT dev set and shows similar trends.

4.4 Ablations

Table 2 reports three ablations. First, we evaluate the impact of the type-based loss term (L_{type}) by setting $\alpha = 0$. We find that accuracy@1 drops by 7.0 points on the tail and 4.6 points on the head, even though types are still in the input. Second, we evaluate the impact of comparing all pairs of enti-

	Fact Check.	Slot Filling		Question Answering				Dial.
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
TF-IDF*	50.9	44.7	60.8	28.1	34.1	46.4	13.7	49.0
DPR-BERT*	72.9	-	40.1	**60.7**	25.0	43.4	-	-
DPR*	55.3	13.3	28.9	54.3	25.0	44.5	10.7	25.5
Multi-task DPR*	74.5	69.5	80.9	59.4	42.9	61.5	15.5	41.1
RAG*	61.9	28.7	53.7	59.5	30.6	48.7	11.0	57.8
BLINK*	63.7	59.6	78.8	24.5	46.1	65.6	9.3	38.2
GENRE†	83.6	**79.4**	**95.8**	60.3	51.3	**69.2**	15.8	**62.9**
TABi	**85.4**	78.1	91.6	59.3	**52.7**	67.9	**18.8**	60.5

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

ties and mentions by using the standard InfoNCE loss instead of L_{ent}. We find that using L_{NCE} results in an accuracy@1 drop of 0.7 and 1.1 points over the head and tail, respectively. Finally, we evaluate the impact of the re-ranker (which is used to maintain head performance) by only using the scores of the bi-encoder. Using only the bi-encoder results in an accuracy@1 drop of 3.0 points over the head and slightly improves the tail.

5 Embedding Quality Analysis

We evaluate how well our method captures types through embedding visualization (§5.1) and nearest neighbor type classification (§5.2). We also evaluate how well TABi learns fine-grained type hierarchies with an entity similarity task (§5.3).

5.1 Embedding visualization

We use t-SNE to qualitatively evaluate how well bi-encoders cluster entity embeddings by type. We select five types and sample entities that belong to each type from the KILT-E knowledge base. In Figure 2, we see TABi forms tighter type clusters than BLINK. We observe that types are not captured as well when the types are not included in the loss—even when the type is present in the input. This suggests that our type-based loss term helps encode types in embedding space.

5.2 Type classification

To better understand how well embeddings are clustered by type, we evaluate query and entity embeddings on two nearest neighbor type classification tasks. Given an embedding, the model retrieves the 10 nearest embeddings and predicts the types as the majority types of the neighbors.[10] We use strict accuracy, loose micro F1, and loose macro F1 metrics for evaluation (Zhang et al., 2019).

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

Dataset	Model	Acc.	Micro F1	Macro F1
FIGER	BLINK	15.8	40.5	25.1
	TABi	**51.2**	**74.4**	**77.6**
OntoNotes	BLINK	21.5	34.2	42.3
	TABi	**36.8**	**54.8**	**60.4**

Table 4: Mention type classification using a nearest neighbor classifier over query embeddings.

We first evaluate the query type by sampling 10k training examples from the FIGER (Ling and Weld, 2012) and OntoNotes (Gillick et al., 2014) training sets and evaluating on the test sets with query embeddings. We find that TABi outperforms BLINK by 33.9 micro F1 points on FIGER and 20.6 micro F1 points on OntoNotes, confirming that our loss encourages nearby query embeddings to share the same type (Table 4). Next, we sample entities from our knowledge base to evaluate the entity type. We find that TABi outperforms BLINK on both coarse and fine types by 7.0 and 6.9 micro F1 points, respectively (see Appendix C.2). This further confirms that our loss helps the query and entity embeddings encode types.

5.3 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 create a dataset of entity pairs where two entities have a high similarity score if they share a fine-grained type and a lower similarity score if they only share a coarse type. We sample 500 entity pairs that share Wikidata types of varying coarseness and use the KGTK Semantic Similarity toolkit (Ilievski et al., 2021)[11] to automatically assign ground truth similarity scores between the pairs using a weighted Jaccard similarity metric (see Appendix C.3).

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

	TransE	ComplEx	BLINK	TABi
Spearman ρ	62.4	63.4	59.4	**69.7**

Table 5: Spearman rank correlation on an entity similarity task over pairs of Wikidata entities.

In Table 5 we compare 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). We find that TABi can outperform BLINK and even the knowledge graph embeddings at this task. 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

Our approach has a couple limitations. First, we assume a high-coverage, relatively coarse type system is available. If many entities in the training set do not have types, the gains of using a type-enforced contrastive loss would be reduced. Furthermore, 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 batch have the same type. Second, our method is designed for open-domain tasks (e.g. QA) which tend to have short queries as input and where types are often a strong signal for disambiguation. We observe that knowledge graph relations and co-reference, which our method does not optimize for learning, are important for longer input, such as with entity linking tasks. 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; Raiman and Raiman, 2018; Gillick et al., 2019; Onoe and Durrett, 2020; Orr et al., 2021). Most closely related to our work are Gillick et al. (2019) and Gupta et al. (2017). Gillick et al. (2019) train dense entity 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 to limit the candidates. Generally, prior works that use types assume mention boundaries are given as input and were not designed for learned end-to-end retrieval. Finally, similar to our work, Gupta et al. (2017), Onoe and Durrett (2020), and Orr et al. (2021) demonstrate that using types can improve disambiguation of rare entities.

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 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 existing 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

In this work, we introduce a method to train bi-encoders on both unstructured text and structured types through a type-enforced contrastive loss. As our method simply changes the bi-encoder loss, it generalizes to both dense entity and document retrieval approaches and can be trained for end-to-end retrieval for open-domain NLP tasks. Our experiments find that our loss can improve retrieval of rare entities for ambiguous mentions and can better capture types in the embeddings. Moreover, we find that by adding an inexpensive re-ranker, which leverages two non-learned components (a sparse retriever and popularity statistics), our method can achieve overall retrieval quality comparable to much more expensive models. We hope our work inspires future work on integrating structured data into pretrained models.

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*.

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 *Proceedings of the International Conference on Learning Representations (ICLR)*.

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.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, 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 *Proceedings of the International Conference on Learning Representations (ICLR)*.

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.

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 *Proceedings of the International Conference on Learning Representations (ICLR)*.

Filip Ilievski, Pedro Szekely, Gleb Satyukov, and Amandeep Singh. 2021. User-friendly comparison of similarity algorithms on wikidata. *Computing Research Repository*, arXiv: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. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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*, volume 33, pages 18661–18673. Curran Associates, Inc.

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 Weld. 2012. Fine-grained entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and M. 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 and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

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. *Computing Research Repository*, arXiv: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.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

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 International Conference on Machine Learning*.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *Computing Research Repository*, arXiv: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. *Computing Research Repository*, arXiv: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 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 6.

Model	# Parameters
Alias Table (Prior)	0
TF-IDF	0
DPR	220M
BLINK (Bi-encoder)	680M
BLINK	1.0B
ELQ	680M
Bootleg	1.3B
GENRE	406M
TABi	110M

Table 6: Number of model parameters.

For Alias Table (Prior), 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.[12] 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).

For ELQ, we use the code provided in the ELQ repository.[13] 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.[14] We use the model version from July 2021.

For GENRE, we use the code provided in the GENRE repository.[15] 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).

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

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

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

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

A.2 Evaluation datasets

We include statistics on the evaluation datasets described in Section 4.1 in Table 7 (AmbER) and Table 8 (KILT). 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 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. 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 linking datasets. With the exception of the entity linking examples, the additional examples have a gold entity label, but no gold mention boundaries.

As we use distant supervision to assign type labels, they may not actually occur in the context, introducing noise. Additionally, we do not have types for all entities. We are able to assign types to 73% of examples in the BLINK training data and 87% of examples in the KILT training data.

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) with a linear learning rate schedule and 10% warmup. Unlike BLINK, we train TABi without hard negative sampling.

We conduct a grid search for the type weight α, temperature τ, and initial learning rate by training for one epoch on the BLINK training set and selecting the best values on the BLINK dev set (9,938 Wikipedia examples).[16] We sweep α in $\{0.1, 0.25\}$, τ in $\{0.01, 0.05\}$, and the initial learning rate in {1e-4, 5e-4} for a total of 8 trials. From our grid search, the best hyperparameters were as follows: $\alpha = 0.1$, temperature $\tau = 0.05$, and initial learning rate=5e-4.

We use the same hyperparameter configuration for training on both the BLINK training data and the KILT training data. Like BLINK, we also train for 4 epochs (for both datasets). We use 16 A100 GPUs for training (25 min/epoch for BLINK training data, 40 min/epoch for KILT training data).

A.5 Re-ranking details

We use two tunable weights for the re-ranker: λ is a weight on the sparse retriever scores and κ is a weight on the global entity popularity scores. We use the TF-IDF retriever that we use as a baseline as 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. We normalize scores before linearly combining and re-rank the top-10 scores. Note that tuning these weights does not require re-training or re-running the bi-encoder evaluation.

We tune λ and κ on each of the 14 dev sets (6 dev sets for AmbER and 8 dev sets for KILT) 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 $\{0.0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$. We include the best configuration for each dev set in Table 9.

B Extended Retrieval Results

B.1 AmbER results

We extend the results on AmbER included in Section 4. First, we report results for the consistency metric introduced in Chen et al. (2021) for top-1 retrieval in Table 10. 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

[16] We remove examples that do not have a gold entity in the KILT-E knowledge base.

Dataset	Dev			Test			Type of Queries
	Total	# Head	# Tail	Total	# Head	# Tail	
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

Table 7: AmbER dataset statistics.

	# Dev	# Test	Type of Queries
FEVER	10,444	10,100	Mutated Wikipedia claims
T-REx	5,000	5,000	Subject-relation facts
Zero Shot RE	3,724	4,966	Subject-relation facts
Natural Questions	2,837	1,444	Search engine questions
HotpotQA	5,600	5,569	Crowd-sourced questions
TriviaQA	5,359	6,586	Trivia questions from trivia sites
ELI5	1,507	600	Reddit questions
Wizard of Wikipedia	3,054	2,944	Crowd-sourced dialogue

Table 8: KILT dataset statistics.

		λ	κ
AmbER (GOLD)	Human FC	0.00	0.25
	Non-human FC	0.75	0.00
	Human SF	0.00	0.25
	Non-human SF	0.50	0.25
	Human QA	0.50	0.00
	Non-human QA	0.25	0.00
AmbER	Human FC	0.50	0.00
	Non-human FC	0.25	0.00
	Human SF	0.00	0.25
	Non-human SF	0.25	0.00
	Human QA	0.25	0.00
	Non-human QA	0.25	0.25
KILT	FEVER	0.75	0.75
	T-REx	0.50	0.00
	Zero Shot RE	0.75	0.50
	Natural Questions	0.50	1.00
	HotpotQA	1.00	0.75
	TriviaQA	0.50	1.25
	ELI5	0.50	1.75
	Wizard of Wikipedia	0.75	1.25

Table 9: Best configuration for re-ranker weights λ (sparse retriever weight) and κ (popularity weight) tuned on the corresponding dev sets.

entity for the mention, which would result in a low consistency value. We find that TABi outperforms all models on this metric. Second, we include results for top-10 retrieval accuracy (accuracy@10) on AmbER to understand the retrieval performance at larger K (Table 11). We find that TABi continues to outperform baselines on average.

Model	**FC**		**SF**		**QA**		**Avg.**
	H	N	H	N	H	N	
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
Bootleg	3.0	0.0	26.7	15.5	31.6	27.8	17.4
GENRE	4.3	1.0	28.3	39.2	10.9	13.9	16.3
TABi	**44.1**	**4.0**	**60.0**	**61.8**	**56.7**	**41.9**	**44.7**

Table 10: Consistency results on AmbER for top-1. The consistency is the fraction of mentions where all queries for a mention are correct.

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 and TABi outperform the other baselines across the tasks, and TABi continues to perform comparably to GENRE. Note that only GENRE 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.

We also report results on the KILT test and 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

	Fact Checking				Slot Filling				Question Answering					
	H		N		H		N		H		N		Average	
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
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	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
TABi	94.7	95.4	79.1	**80.2**	90.6	95.8	**96.4**	**97.8**	94.0	96.1	**95.9**	**95.7**	**91.8**	**93.5**

Table 11: Accuracy@10 on AmbER. H refers to the human subset and N refers to the non-human subset. †Models with an alias table. Best score **bolded** and second best underlined.

	Fact Check.	Slot Filling		Question Answering				Dial.
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
TF-IDF	48.4	57.4	72.8	20.1	43.4	27.8	4.6	38.8
DPR	57.0	14.9	44.3	54.5	25.5	46.2	16.1	26.9
BLINK (Bi-encoder)	64.4	59.4	84.3	35.1	43.1	61.6	11.3	26.0
BLINK	67.6	61.0	87.4	33.5	47.9	65.9	9.7	26.5
ELQ	65.1	71.2	95.0	42.4	45.9	67.7	9.2	26.8
Bootleg†	62.3	69.4	81.8	34.5	43.6	53.1	9.7	28.2
GENRE	85.0	**80.5**	**95.1**	**61.4**	**51.9**	**71.4**	13.6	**56.5**
TABi	**87.3**	79.0	94.8	59.4	50.4	68.9	**17.9**	56.4

Table 12: R-precision on KILT open-domain tasks (dev data). †Models with an alias table. Best score **bolded** and second best underlined.

query[17] 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 continues to have strong performance, performing comparably to GENRE, and outperforming other baselines (Table 13 (test) and Table 14 (dev)).

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 5.2). 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.

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

C.2 Nearest neighbor entity type classification

We include the setup and extended results for the entity type classification task from Section 5.2. 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.

	Fact Check.	Slot Filling		Question Answering				Dial.
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
TF-IDF	-	-	-	-	-	-	-	-
DPR+BERT	73.5	-	40.1	46.8	10.4	31.5	-	-
DPR	74.3	17.0	39.2	65.5	10.4	57.0	26.9	51.2
Multi-task DPR	87.5	83.9	93.1	**68.2**	28.4	68.3	27.5	67.1
RAG	75.6	33.0	59.5	67.1	12.6	57.1	22.9	74.6
BLINK+flair	-	-	-	-	-	-	-	-
GENRE	88.2	85.3	97.8	61.4	34.0	**75.1**	25.5	**77.7**
TABi	**91.4**	**87.4**	**98.9**	64.4	**37.0**	67.5	**29.4**	75.5

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. Best score **bolded** and second best underlined.

	Fact Check.	Slot Filling		Question Answering				Dial.
	FEV	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
TF-IDF	71.8	73.0	88.6	32.6	29.2	41.0	9.7	56.5
DPR	76.0	22.3	59.2	**63.9**	11.1	57.4	**31.0**	52.7
BLINK (Bi-encoder)	80.0	68.1	88.4	40.8	24.3	63.5	19.4	40.9
BLINK	82.9	69.6	89.6	43.7	27.4	66.9	22.3	44.6
ELQ	79.5	69.9	95.2	36.1	23.7	62.4	9.5	47.7
Bootleg†	81.0	74.3	85.6	37.2	26.3	69.4	14.0	49.3
GENRE	89.0	85.3	97.3	58.5	34.7	**75.7**	20.5	**75.0**
TABi	**91.2**	**87.7**	**99.2**	62.5	**35.3**	68.0	25.0	74.4

Table 14: Recall@5 on KILT open-domain tasks (dev data). †Models with an alias table. Best score **bolded** and second best underlined.

Dataset	Model	Acc.	Micro F1	Macro F1
Coarse-types	BLINK	81.1	89.0	84.1
	TABi	**92.9**	**96.0**	**96.2**
Fine-types	BLINK	71.6	82.0	77.5
	TABi	**80.8**	**88.9**	**87.5**

Table 15: Entity type classification using a nearest neighbor classifier over entity embeddings.

Table 15 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 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.[18] 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.

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