Efficient One-Pass End-to-End Entity Linking for Questions

Belinda Z. Li[♠*] **Sewon Min**[♢]
Srinivasan Iyer[†] **Yashar Mehdad**[†] **Wen-tau Yih**[†]
[♠]MIT CSAIL [♢]University of Washington [†]Facebook AI
bzl@mit.edu sewon@cs.washington.edu
{sviyer,mehdad,scottyih}@fb.com

Abstract

We present ELQ, a fast end-to-end entity linking model for questions, which uses a biencoder to jointly perform mention detection and linking in one pass. Evaluated on WebQSP and GraphQuestions with extended annotations that cover multiple entities per question, ELQ outperforms the previous state of the art by a large margin of +12.7% and +19.6% F1, respectively. With a very fast inference time (1.57 examples/s on a single CPU), ELQ can be useful for downstream question answering systems. In a proof-of-concept experiment, we demonstrate that using ELQ significantly improves the downstream QA performance of GraphRetriever (Min et al., 2019).[1]

1 Introduction

Entity linking (EL), the task of identifying entities and mapping them to the correct entries in a database, is crucial for analyzing factoid questions and for building robust question answering (QA) systems. For instance, the question "*when did shaq come to the nba?*" can be answered by examining *Shaquille O'Neal*'s Wikipedia article (Min et al., 2019), or its properties in a knowledge graph (Yih et al., 2015; Yu et al., 2017). However, real-world user questions are invariably noisy and ill-formed, lacking cues provided by casing and punctuation, which prove challenging to current end-to-end entity linking systems (Yang and Chang, 2015; Sorokin and Gurevych, 2018). While recent pre-trained models have proven highly effective for entity linking (Logeswaran et al., 2019; Wu et al., 2020), they are only designed for entity disambiguation and require mention boundaries to be given in the input. Additionally, such systems have only been evaluated on long, well-formed documents like news articles (Ji et al., 2010), but not on short, noisy text. Also, most prior works have focused mainly on improving model prediction accuracy, largely overlooking efficiency.

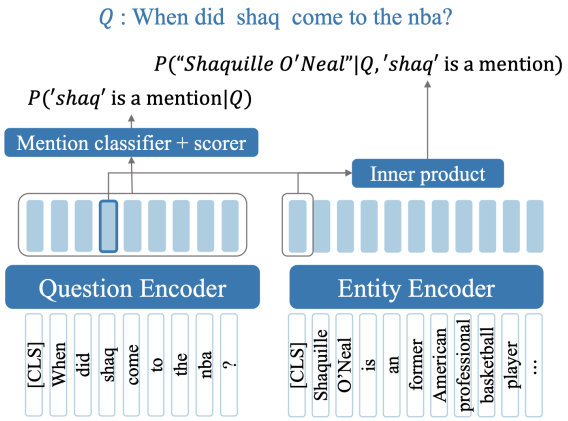

Figure 1: Overview of our end-to-end entity linking system. We separately encode the question and entity. We use the question representations to jointly detect mentions and score candidate entities through inner-product with the entity vector.

In this work, we propose ELQ, a fast and accurate entity linking system that specifically targets questions. Following the Wikification setup (Ratinov et al., 2011), ELQ aims to identify the mention boundaries of entities in a given question and their corresponding Wikipedia entity. We employ a biencoder based on BERT (Devlin et al., 2019) as shown in Figure 1. The entity encoder computes entity embeddings for all entities in Wikipedia, using their short descriptions. Then, the question encoder derives token-level embeddings for the input question. We detect mention boundaries using these embeddings, and disambiguate each entity mention based on an inner product between the mention embeddings (averaged embedding over mention tokens) and the entity embeddings. Our model ex-

[*]Work done while at Facebook AI.
[1]Code and data available at https://github.com/facebookresearch/BLINK/tree/master/elq

tends the work of Wu et al. (2020) but with one major difference: our system does not require pre-specified mention boundaries in the input, and is able to jointly perform mention detection and entity disambiguation in just one pass of BERT. Thus, at inference time, we are able to identify multiple entities in the input question efficiently.

We extend entity disambiguation annotations from Sorokin and Gurevych (2018) to create an end-to-end question entity linking benchmark. Evaluated on this benchmark, we are able to outperform previous methods in both accuracy and run-time. ELQ has much faster end-to-end inference time than any other neural baseline (by $2\times$), while being more accurate than all previous models we evaluate against, suggesting that it is practically useful for downstream QA systems. We verify the applicability of ELQ to practical QA models in a proof-of-concept experiment, by augmenting GraphRetriever (Min et al., 2019) to use our model, improving its downstream QA performance on three open-domain QA datasets (by up to 6%).

2 Related Work

Much prior work on entity linking has focused on long, grammatically coherent documents that contain many entities. This setting does not accurately reflect the difficulties of entity linking on questions. While there has been some previous work on entity linking for questions (Sorokin and Gurevych, 2018; Blanco et al., 2015; Chen et al., 2018; Tan et al., 2017), such works (mostly from the pre-BERT era) utilize complex models with many interworking modules. For example, Sorokin and Gurevych (2018) proposes a variable-context granularity (VCG) model to address the noise and lack of context in questions, which incorporates signals from various levels of granularity by using character-level, token-level, and knowledge-base-level modules. They also rely on external systems as a part of the modeling pipeline.

In this work, we take a much simpler approach that uses a biencoder. Biencoder models have been used in a wide range of tasks (Seo et al., 2019; Karpukhin et al., 2020; Wu et al., 2020). They enable fast inference time through maximum inner product search. Moreover, as we find, biencoders can be decomposed into reusable question and entity encoders, and we can greatly expedite training by training one component independently of the other.

3 Problem Definition & ELQ Model

We formally define our entity linking task as follows. Given a question q and a set of entities $\mathcal{E} = \{e_i\}$ from Wikipedia, each with titles $t(e_i)$ and text descriptions $d(e_i)$, our goal is to output a list of tuples, $(e, [m_\text{s}, m_\text{e}])$, whereby $e \in \mathcal{E}$ is the entity corresponding to the mention span from the m_s-th to m_e-th token in q. In practice, we take the title and first 128 tokens of the entity's Wikipedia article as its title $t(e_i)$ and description $d(e_i)$.

We propose an end-to-end entity linking system that performs both mention detection and entity disambiguation on questions in one pass of BERT.

Given an input question $q = q_1 \cdots q_n$ of length n, we first obtain question token representations based on BERT (Devlin et al., 2019):

$$[\mathbf{q_1} \cdots \mathbf{q}_n]^\intercal = \text{BERT}([\texttt{CLS}]\, q_1 \cdots q_n [\texttt{SEP}]) \in \mathbb{R}^{n \times h},$$

where each $\mathbf{q_i}$ is a h-dimensional vector. We then obtain entity representations $\mathbf{x}_e$ for every $e_i \in \mathcal{E}$.

$$\mathbf{x}_e = \text{BERT}_{[\texttt{CLS}]}([\texttt{CLS}] t(e_i) [\texttt{ENT}] d(e_i) [\texttt{SEP}]) \in \mathbb{R}^h,$$

where [CLS] indicates that we select the representation of the [CLS] token. We consider candidate mentions as all spans $[i, j]$ (i-th to j-th tokens of q) in the text up to length L.

Mention Detection To compute the likelihood score of a candidate span $[i, j]$ being an entity mention, we first obtain scores for each token being the start or the end of a mention:

$$s_\text{start}(i) = \mathbf{w}_\text{start}^\intercal \mathbf{q}_i, \quad s_\text{end}(j) = \mathbf{w}_\text{end}^\intercal \mathbf{q}_j,$$

where $\mathbf{w}_\text{start}, \mathbf{w}_\text{end} \in \mathbb{R}^h$ are learnable vectors. We additionally compute scores for each token t being part of a mention:

$$s_\text{mention}(t) = \mathbf{w}_\text{mention}^\intercal \mathbf{q}_t,$$

where $\mathbf{w}_\text{mention} \in \mathbb{R}^h$ is a learnable vector. We finally compute mention probabilities as:

$$p([i, j]) = \sigma(s_\text{start}(i) + s_\text{end}(j) + \sum_{t=i}^{j} s_\text{mention}(t)).$$

Entity Disambiguation We obtain a mention representation for each mention candidate $[i, j]$ by averaging $\mathbf{q_i} \cdots \mathbf{q_j}$, and compute a similarity score

s between the mention candidate and an entity candidate $e \in \mathcal{E}$:

$$\begin{aligned} \mathbf{y}_{i,j} &= \frac{1}{(j-i+1)} \sum_{t=i}^{j} \mathbf{q}_t \in \mathbb{R}^h, \\ s(e,[i,j]) &= \mathbf{x}_e^\intercal \mathbf{y}_{i,j}. \end{aligned}$$

We then compute a likelihood distribution over all entities, conditioned on the mention $[i,j]$:

$$p(e|[i,j]) = \frac{\exp(s(e,[i,j]))}{\sum_{e' \in \mathcal{E}} \exp(s(e',[i,j]))}.$$

Training We jointly train the mention detection and entity disambiguation components by optimizing the sum of their losses. We use a binary cross-entropy loss across all mention candidates:

$$\begin{aligned} \mathcal{L}_{\text{MD}} = -\frac{1}{N} \sum_{\substack{1 \le i \le j \le \\ \min(i+L-1,n)}} \Big(& y_{[i,j]} \log p([i,j]) \\ & + (1 - y_{[i,j]}) \log\left(1 - p([i,j])\right) \Big), \end{aligned}$$

whereby $y_{[i,j]} = 1$ if $[i,j]$ is a gold mention span, and 0 otherwise. N is the total number of candidates we consider.[2]

The entity disambiguation loss is given by

$$\mathcal{L}_{\text{ED}} = -\log p(e_g|[i,j]),$$

where e_g is the gold entity corresponding to mention $[i,j]$.

To expedite training, we use a simple transfer learning technique: we take the entity encoder trained on Wikipedia by Wu et al. (2020) and freeze its weights, training only the question encoder on QA data. In addition, we mine hard negatives. As entity encodings are fixed, a fast search of hard negatives in real time is possible.

Inference Figure 1 shows our inference process. Given an input question q, we use our mention detection model to obtain our mention set $\mathcal{M} = \{[i,j] : 1 \le i \le j \le \min(i+L-1, n), p([i,j]) > \gamma\}$, where γ is our threshold (a hyperparameter). We then compute $p(e,[i,j]) = p(e|[i,j])p([i,j])$ for each mention $[i,j] \in \mathcal{M}$, and threshold according to γ. In contrast to a two-stage pipeline which first extracts mentions, then disambiguates entities (Févry et al., 2020), a joint approach grants us the flexibility to consider multiple possible candidate mentions for entity linking. This can be crucial in questions as it can be difficult to extract mentions from short, noisy text in a single step.

More implementation details can be found in Appendix D.

[2] If $n \ge L$, $N = L(L+1)/2 + (n-L)L$. Otherwise, $N = n(n+1)/2$.

Data	Train		Test	
	#Q	#E	#Q	#E
$\text{WebQSP}_{\text{EL}}$	2974	3242	1603	1806
$\text{GraphQ}_{\text{EL}}$	2089	2253	2075	2229

Table 1: Dataset statistics of $\text{WebQSP}_{\text{EL}}$ and $\text{GraphQ}_{\text{EL}}$. #Q and #E indicate the number of questions and entities, respectively.

4 Experiments

4.1 Data

We evaluate our approach on two QA datasets, WebQSP (Yih et al., 2016) and GraphQuestions (Su et al., 2016), with additional entity annotations provided by Sorokin and Gurevych (2018). The original datasets do not have all mention boundary labels annotated. Therefore, in order to evaluate both mention detection and entity disambiguation, we extend previous labels and create new end-to-end question entity-linking datasets, $\text{WebQSP}_{\text{EL}}$ and $\text{GraphQ}_{\text{EL}}$.[3] In line with our task definition, all entities presented in each question are labeled with $(e,[m_{\text{s}},m_{\text{e}}])$, whereby $e \in \mathcal{E}$ is the entity corresponding to the mention span from the m_{s}-th to m_{e}-th token in q. We ask four in-house annotators to identify corresponding mention boundaries, given gold entities in the questions. We exclude examples that link to `null` or no entities, that are not in Wikipedia, or are incorrect or overly generic (e.g. linking a concept like *marry*). To check inter-annotation agreement amongst the 4 annotators, we set aside a shared set of documents (comprised of documents from both datasets) that all 4 annotators annotated. We found exact-match inter-annotator agreement to be 95% (39/41) on this shared set.

Table 1 reports the statistics of the resulting datasets, $\text{WebQSP}_{\text{EL}}$ and $\text{GraphQ}_{\text{EL}}$. Following Sorokin and Gurevych (2018), we use $\text{WebQSP}_{\text{EL}}$ for training and $\text{GraphQ}_{\text{EL}}$ for zero-shot evaluation.

Evaluation Metrics Using the rule defined by Carmel et al. (2014), a prediction is correct only

[3] Data available at `http://dl.fbaipublicfiles.com/elq/EL4QA_data.tar.gz`

Training Data	Model	WebQSP$_{EL}$				GraphQ$_{EL}$ (zero-shot)			
		Prec	Recall	F1	#Q/s	Prec	Recall	F1	#Q/s
WebQSP$_{EL}$	VCG†	82.4	68.3	74.7	0.45	54.1	30.6	39.0	0.26
	ELQ	90.0	85.0	87.4	1.56	60.1	57.2	58.6	1.57
Wikipedia	TAGME	53.1	27.3	36.1	**2.39**	49.6	36.5	42.0	**3.16**
	BLINK	82.2	79.4	80.8	0.80	65.3	61.2	63.2	0.78
	ELQ	86.1	81.8	83.9	1.56	69.8	**69.8**	69.8	1.57
Wikipedia + WebQSP$_{EL}$	ELQ	**91.0**	**87.0**	**89.0**	1.56	**74.7**	66.4	**70.3**	1.57

Table 2: Results on WebQSP$_{EL}$ and GraphQ$_{EL}$ test data, under 3 training settings. '#Q/s' (number of questions per second) indicates inference speed on 1 CPU. Models trained in comparable settings are clustered together. Overall highest scores are **bolded**, while highest scores per setting are underlined.
†VCG results are different from numbers in the original paper as the evaluation sets are slightly different.

if the groundtruth entity is identified *and* the predicted mention boundaries overlap with the groundtruth boundaries. (This is sometimes known as "weak matching".) Specifically, let $\mathcal{T}$ be a set of gold entity-mention tuples and $\widehat{\mathcal{T}}$ be a set of predicted entity-mention tuples, we define precision (p), recall (r) and F1-score (F_1) as follows:

$$\mathcal{C} = \{e \in \mathcal{E} | [m_s, m_e] \cap [\widehat{m}_s, \widehat{m}_e] \neq \varnothing, (e, [m_s, m_e]) \in \mathcal{T}, (e, [\widehat{m}_s, \widehat{m}_e]) \in \widehat{\mathcal{T}}\},$$

$$p = \frac{|\mathcal{C}|}{|\widehat{\mathcal{T}}|}, \quad r = \frac{|\mathcal{C}|}{|\mathcal{T}|}, \quad F_1 = \frac{2pr}{p+r}.$$

Baselines We use the following baselines: (1) **TAGME** (Ferragina and Scaiella, 2012), a lightweight, on-the-fly entity linking system that is popular for many downstream QA tasks, being much faster than most neural models (Joshi et al., 2017; Sun et al., 2018; Min et al., 2019), (2) **VCG** (Sorokin and Gurevych, 2018), the current state-of-the-art entity linking system on WebQSP, and (3) biencoder from **BLINK** (Wu et al., 2020). As BLINK requires pre-specified mention boundaries as input, we train a separate, BERT-based span extraction model on WebQSP in order to predict mention boundaries (details in Appendix B).

4.2 Results

Table 2 show our main results. We find that BERT-based biencoder models far outperform the state-of-the-art (VCG) on both datasets, in performance and in runtime. Moreover, ELQ outperforms all other models trained in a comparable setting, and is much more efficient than every other neural baseline (VCG and BLINK). ELQ is also up to 2.3× better than TAGME — in the case of WebQSP$_{EL}$.

Performance ELQ outperforms BLINK, suggesting that it is possible to train representations from a single model to resolve both entity references as well as mention boundaries of all entities in text, without restricting the model to focusing on a single marked entity as in BLINK.

	WQ	NQ	TQA
TF-IDF†	20.8	28.7	54.0
TAGME + GRetriever†	31.8	33.5	55.0
ELQ$_{Wiki}$ + GRetriever	37.4	**37.4**	**55.4**
ELQ$_{QA}$ + GRetriever	**37.7**	37.0	54.7

Table 3: QA result (Exact Match) on the test set of WebQuestions (WQ), Natural Questions (NQ) and TriviaQA (TQA). ELQ$_{Wiki}$ represents our model trained on Wikipedia data, while ELQ$_{QA}$ represents our model trained on Wikipedia+WebQSP$_{EL}$ data.
†Result taken from (Min et al., 2019).

Runtime We record the inference speed on CPUs in number of questions processed per second for all models (Table 2). For BLINK, we report the combined speed of our span extraction model and the BLINK entity linker, in order to compare the *end-to-end* speeds. ELQ, which performs both detection and disambiguation in one pass of BERT, is approximately 2× faster than BLINK, which performs multiple passes, while also outperforming BLINK in F1 score. Moreover, against TAGME (Ferragina and Scaiella, 2012), ELQ is only 1.5× slower on WebQSP$_{EL}$ and 2.0× slower on GraphQ$_{EL}$, despite TAGME being a completely non-neural model (with much lower accuracy).

5 QA Experiments

To demonstrate the impact of improved entity linking on the end QA accuracy, we experiment with the task of textual open-domain question answering, using GraphRetriever (GRetriever) (Min et al., 2019). GRetriever uses entity linking to construct a graph of passages in the retrieval step and deploys

Experiment	ELQ (Wiki)	BLINK
MD + EL	87.3	82.2
MD only	94.6	92.9
EL only	90.2	86.6

Table 4: Analyzing the performance of the mention detector and entity linker respectively on $\text{WebQSP}_{\text{EL}}$ (dev). We compare our Wikipedia-trained model to BLINK. MD + EL refers to the end-to-end F1 score (the normal setup).

Error Type	$\text{WebQSP}_{\text{EL}}$	$\text{GraphQ}_{\text{EL}}$
Technically Correct	49.2	23.3
Not Enough Entities	13.1	51.8
Wrong Entities	26.2	20
Insufficient Context	11.5	5

Table 5: Breakdown of frequency of each error type on each dev set (in terms of % of all errors on that dataset). We use ELQ trained on Wikipedia + $\text{WebQSP}_{\text{EL}}$ here.

a reader model to answer the question. The original model uses TAGME for entity linking; we replace TAGME with ELQ and keep the other components the same, in order to isolate the impact of entity linking.[4] As an additional baseline, we also add the result of TF-IDF, implemented by Chen et al. (2017), a widely used retrieval system.

Results are shown in Table 3. Following literature in open-domain QA, we evaluate our approach on three datasets, WebQuestions (Berant et al., 2013), Natural Questions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). In particular, WebQuestions (WQ) and Natural Questions (NQ) consist of short, noisy questions from Web queries, in line with the motivation of our work. We observe that simply replacing TAGME with ELQ significantly improves performance, including 5.9% and 3.9% absolute improvements on WQ and NQ, respectively. While ELQ trained on Wikipedia achieves good results overall, further fine-tuning on $\text{WebQSP}_{\text{EL}}$ gives extra gains on WQ. This indicates that, if entity linking annotations in the same domain are available, using them to fine-tune ELQ can bring further gains.

6 Analysis

Mention Detector vs. Entity Linker We set up experiments to disentangle the capability of ELQ's entity linker and mention detector. First, to test just the mention detector (MD only), we measure just the mention boundary overlap between predicted and groundtruth mentions, ignoring the entity label. Next, to test just the entity linker (EL only), we give the entity linking component *gold* mention boundaries, and compute the resulting F1 score. We do this for both ELQ and BLINK. For comparability, we use the version of ELQ trained on Wikipedia. Results are reported in Table 4. Surprisingly, we find that *both* components of ELQ outperform BLINK, suggesting that the two tasks might *mutually benefit* from being trained jointly.

Runtime To confirm that our biencoder's main bottleneck is the BERT forward pass — and thus, investing in decreasing the number of BERT forward passes is valuable — we separately time each component of ELQ during inference. We run examples from $\text{WebQSP}_{\text{EL}}$ test set one at a time through ELQ, on 1 CPU, and average runtimes across all examples. Indeed, we find that the BERT forward pass to be the slowest component of the model, taking 0.683s, over $6\times$ slower than the next slowest component of the model, inner-product search (taking 0.107s). Everything else takes a combined total of 5.08×10^{-3}s.

Qualitative We manually examine all our model's errors on the $\text{WebQSP}_{\text{EL}}$ and $\text{GraphQ}_{\text{EL}}$ dev sets. We identify four broad error categories: (1) technically correct — where our model was technically correct but limitations in evaluation falsely penalized our model (i.e., we found a more or less precise version of the same entity), (2) not enough entities — where the model did not fully identify all entities in the question, (3) wrong entities — where our model linked to the wrong entity, (4) insufficient context — where the model made reasonable mistakes due to the lack of context (that even reasonable humans would make). Error type breakdowns can be found in Table 5.

7 Conclusion

We proposed an end-to-end model for entity linking on questions that jointly performs mention detection and disambiguation with one pass through BERT. We showed that it is highly efficient, and that it outperforms previous state-of-the-art models on two benchmarks. Furthermore, when applied to a QA model, ELQ improves that model's end QA accuracy. Despite being originally designed with questions in mind, we believe ELQ could also generalize to longer, well-formed documents.

[4] Min et al. (2019) used two reader models, ParReader++ and GraphReader; for simplicity, we only use ParReader++

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *EMNLP*.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *ACM International Conference on Web Search and Data Mining*.

David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June (Paul) Hsu, and Kuansan Wang. 2014. ERD'14: Entity recognition and disambiguation challenge. *SIGIR Forum*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *ACL*.

Lihan Chen, Jiaqing Liang, Chenhao Xie, and Yanghua Xiao. 2018. Short text entity linking with fine-grained topics. In *ACM International Conference on Information and Knowledge Management*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

P. Ferragina and U. Scaiella. 2012. Fast and accurate annotation of short texts with Wikipedia pages. *IEEE Software*.

Thibault Févry, Nicholas FitzGerald, Livio Baldini Soares, and Tom Kwiatkowski. 2020. Empirical evaluation of pretraining strategies for supervised entity linking. In *Automated Knowledge Base Construction*.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *Text analysis conference*.

Jeff Johnson, Matthijs Douze, and Herve Jegou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP*.

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 Change, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *TACL*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *ACL*.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *ACL*.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur P Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *ACL*.

Daniil Sorokin and Iryna Gurevych. 2018. Mixing context granularities for improved entity linking on question answering data across entity categories. In *Joint Conference on Lexical and Computational Semantics*.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for QA evaluation. In *EMNLP*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *EMNLP*.

Chuanqi Tan, Furu Wei, Pengjie Ren, Weifeng Lv, and Ming Zhou. 2017. Entity linking for queries by searching Wikipedia sentences. In *EMNLP*.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Zero-shot entity linking with dense entity retrieval. In *EMNLP*.

Yi Yang and Ming-Wei Chang. 2015. S-MART: Novel tree-based structured learning algorithms applied to tweet entity linking. In *ACL*.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL*.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *ACL*.

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved neural relation detection for knowledge base question answering. In *ACL*.

A Annotation Statistics

We show annotators a question and a gold entity in the question, and instruct them to annotate (i.e., put brackets around) the appropriate mention span.

For quality control, we created a shared annotation set by pulling a subset of examples from each dataset, and having all four annotators label that set. Inter-annotator agreement statistics on our shared set are shown in Table 6. Note that only 2 out of 41 shared examples did not have unanimous mention boundary agreement. The two conflicting examples, with the respective annotations, are shown below:

Question: 'who are the two state senators of georgia?' Entity: 'United States Senate'
A1: 'who are the two state [senators] of georgia?' A2: 'who are the two [state senators] of georgia?' A3: Entity not in question
Question: 'who was michael jackson in the wiz?' Entity: 'The Wiz (film)'
A1: 'who was michael jackson in [the wiz]?' A2: 'who was michael jackson in the [wiz]?'

B Span-Extraction Model for Mention Boundary Detection

The BLINK Entity Linker requires mention boundaries to be marked in the input. In order to evaluate against BLINK for end-to-end Entity Linking, we train a span-extraction model to first obtain candidate mention boundaries, and subsequently use BLINK on these candidate mentions. Our span extraction model first represents every token q_i in question $q = q_1, \cdots, q_n$ of length n using a dense representation using $\text{BERT}_{\text{base}}$:

$$\mathbf{q_1}, \cdots, \mathbf{q_n} = \text{BERT}(q_1, \cdots, q_n) \quad (1)$$

The model then computes a start span probability $p_s(q_i|q)$ and an end span probability $p_e(q_i|q)$ for every token q_i using learnable vectors $\mathbf{w_s}$ and $\mathbf{w_t}$ respectively:

$$p_s(q_i|q) = \frac{\exp(\mathbf{w_s}^\intercal \mathbf{q_i})}{\sum_j \exp(\mathbf{w_s}^\intercal \mathbf{q_j})} \quad (2)$$

$$p_e(q_i|q) = \frac{\exp(\mathbf{w_e}^\intercal \mathbf{q_i})}{\sum_j \exp(\mathbf{w_e}^\intercal \mathbf{q_j})} \quad (3)$$

The model is trained to maximize the likelihood of $p_s(q_s|q) \times p_e(q_e|q)$ for each correct mention $[q_s, q_e]$ in the training set of WebQSP, and similarly during inference, outputs the top-K scoring spans. These spans are used as mention boundary candidates, to evaluate BLINK in an end-to-end setting.

Dataset	#Examples	#Agreement
WebQSP train	9	8
WebQSP dev	6	5
WebQSP test	10	10
GraphQs train	8	8
GraphQs test	8	8

Table 6: Statistics on the shared dataset annotated by all 4 annotators. We count exact-match, unanimous agreements, i.e., both mention boundaries must exactly match, and all 4 annotators must agree.

Training method	F1
Adversarial + Pre-trained candidate encoder	87.7
Pre-trained candidate encoder	46.1
None (entirely from scratch)	17.2

Table 7: Ablations on our training scheme after 20 epochs. Both transfer learning from the pre-trained candidate encoder *and* adversarially training on hard negatives expedite convergence.

C Analysis

Training Ablations Table 7 presents the contributions of each component of our training scheme to our final result. We record model performance on $\text{WebQSP}_{\text{EL}}$ (valid) after 20 (of 100) epochs of training on Wikipedia, having seen just 20% of the data. We note that both our transfer learning technique and our adversarial hard-negatives training expedites convergence.

Qualitative Error Analysis We record specific examples for each of our four error categories (technically correct, not enough entities, wrong entities, and insufficient context), detailed in Section 6. Note there were 61 total mistakes for $\text{WebQSP}_{\text{EL}}$ dev and 158 total mistakes for $\text{GraphQ}_{\text{EL}}$ dev. Specific examples can be found in Table 9.

D Implementation Details and Hyperparameters

Following Wu et al. (2020), we use $\text{BERT}_{\text{Large}}$ ($\sim$340M parameters) for the question and entity encoder. The span extraction model detailed in Appendix B, used for our BLINK baselines, is a $\text{BERT}_{\text{Base}}$ model ($\sim$110M parameters).

We lowercase all inputs to ELQ, during both training and inference time, to make it case-insensitive. For both training and inference, the mention scorer considers all spans up to length $L = 10$.

Training During training, we use FAISS (Johnson et al., 2019) for fast inner product search when mining hard negatives. We do this in real time, *inside* the training loop. As we do not update the entity encoder, we were able to train the model with a single FAISS index, greatly increasing the training speed. For further speedup, we use a hierarchical index (`IndexHNSWFlat`), with $efConstruction = 200$ and $efSearch = 256$.

For $\mathcal{L}_{\text{ED}}$ at each iteration, computing $s(e, [i, j])$ for every $e \in \mathcal{E}$ is intractable. We thus approximate $\mathcal{L}_{\text{ED}}$ by replacing $\mathcal{E}$ with $\mathcal{E}'$ for the softmax, where $\mathcal{E}'$ is a set of hard negative entities, specifically, negative entities that have the highest 10 similarity scores with the mention representation.

For our WebQSP$_{\text{EL}}$-trained model, we train for up to 100 epochs on WebQSP$_{\text{EL}}$ data, using batch size 128 and context window size of 20 tokens. For our Wikipedia-trained model, we split the data evenly into 100 chunks and train on each (thus, making one pass through Wikipedia overall). For Wikipedia, we use batch size 32 and a context window size of 128 tokens. For Wikipedia + WebQSP$_{\text{EL}}$ model, we take our Wikipedia-trained model and further fine-tune it on WebQSP$_{\text{EL}}$ for up to 100 epochs (using the WebQSP$_{\text{EL}}$ training settings). For all three training settings, we use the AdamW optimizer with learning rate 1e-5, coupled with a linear schedule with 10% warmup. We clip gradients to max norm 1.0.

Inference During inference, we consider all mention candidates $[i, j]$ with mention score $\log p([i, j]) \geq \gamma$. If no mention candidate has mention score $\geq \gamma$, we simply take the top-50-scoring mentions. γ is a threshold we tune on each dataset's dev data.

The linker then retrieves the 10 closest entity candidates per mention boundary. We use the same hierarchical FAISS index as during training to expedite retrieval. Since the search is approximate, we expect some performance degradation. However, in practice, we found minimal performance degradation for significant speedup. On WebQSP$_{\text{EL}}$ dev set, F1 score decreased from $92.5 \rightarrow 91.9$, but run-time decreased from $127.0s \rightarrow 24.3s$ (for the entire dataset, with batch size 64).

As computing the softmax over all entities for $\log p(e|[i, j])$ is intractable, we simply softmax over our 10 retrieved candidates. At the end, we threshold the final joint score $\log p([i, j]) + \log p(e|[i, j])$ based on γ.

Model	WebQSP$_{\text{EL}}$	GraphQ$_{\text{EL}}$
ELQ, Wikipedia	-2.9	-3.5
ELQ, Wikipedia + WebQSP$_{\text{EL}}$	-1.5	-0.9

Table 8: Best threshold hyperparameter value γ, based on the respective development sets.

We use manual tuning and binary search to find the best-performing hyper-parameters for the threshold γ. We optimize for F1-score on the development sets of WebQSP$_{\text{EL}}$ and GraphQ$_{\text{EL}}$. Best settings are reported in Table 8.

As mention overlaps are not allowed in the questions data, we have an additional global step of removing overlapping mention boundaries — in the case of multiple entities, we greedily choose the highest-scoring entity each time, and remove all entities which overlap with it.

E Infrastructure Details

We ran all training distributed across 8 NVIDIA TESLA V100 GPUs, each with 32 GB of memory. For 80-CPU inference, we run on 2 chips of Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz with 20 cores (40 threads) each. For 1-CPU inference (reported in Table 2), we run only on a single core.

Dataset	Example Error
	Technically Correct (WebQSP$_{EL}$ 49.2%; GraphQ$_{EL}$ 23.3%)
WebQSP$_{EL}$	what type of **guitar** does **john mayer** play? GOLD: **john mayer** → "John Mayer" PRED: **guitar** → "Guitar"; **john mayer** → "John Mayer"
WebQSP$_{EL}$	what countries make up **continental europe** ? GOLD: **continental europe** → "Europe" PRED: **continental europe** → "Continental Europe"
	Not Enough Entities (WebQSP$_{EL}$ 13.1%; GraphQ$_{EL}$ 51.8%)
WebQSP$_{EL}$	what country is the **grand bahama island** in? GOLD: **grand bahama island** → "Grand Bahama" PRED:
WebQSP$_{EL}$	what **children's books** did **suzanne collins** wrote? GOLD: **children's books** → "Children's literature"; **suzanne collins** → "Suzanne Collins" PRED: **suzanne collins** → "Suzanne Collins"
GraphQ$_{EL}$	how many people found **o** together? GOLD: **o** → "Oxygen" PRED:
GraphQ$_{EL}$	the rockets **ares i** and **saturn 5** are made by who? GOLD: **ares i** → "Ares I"; **saturn 5** → "Saturn V" PRED: **ares i** → "Ares I"
GraphQ$_{EL}$	where does **spirit and opportunity** aim to land? GOLD: **spirit and opportunity** → "Mars Exploration Rover" PRED:
	Wrong Entities (WebQSP$_{EL}$ 26.2%; GraphQ$_{EL}$ 20%)
WebQSP$_{EL}$	which **kennedy** died first? GOLD: **kennedy** → "Kennedy family" PRED: **kennedy** → "John F. Kennedy"
WebQSP$_{EL}$	what team did **shaq** play for first? GOLD: **shaq** → "Shaquille O'Neal" PRED: **shaq** → "Tupac Shakur"
GraphQ$_{EL}$	**myuutsu** is what kind of pokemon? GOLD: **myuutsu** → "Mewtwo" PRED: **myuutsu** → "Kyjutsu"
GraphQ$_{EL}$	in the **bart** what kind of trains are used? GOLD: **bart** → "Bay Area Rapid Transit" PRED: **bart** → "Bart Simpson"
	Insufficient Context (WebQSP$_{EL}$ 11.5%; GraphQ$_{EL}$ 5%)
WebQSP$_{EL}$	what was **walt disney** 's first cartoon called? GOLD: **walt disney** → "The Walt Disney Company" PRED: **walt disney** → "Walt Disney"
GraphQ$_{EL}$	what botanical gardens to visit in **washington** ? GOLD: **washington** → "Washington, D.C." PRED: **washington** → "Washington (state)"

Table 9: Examples of each error type made by our model.