Increasing Entity Linking upper bound through a more effective Candidate Generation System

Anonymous ACL submission

Abstract

Entity Linking (EL) aligns entity mentions in text to entries in a knowledge base. It usually comprises of two phases: candidate generation and candidate ranking. While most methods focus on the latter phase, it is candidate generation that sets the upper bound for both time and accuracy of an EL system. We propose a simple approach for improving candidate generation by efficiently embedding mention-entity pairs in dense space through a BERT-based bi-encoder. Specifically, we introduce a new pooling function and incorporate entity type side-information. We achieve a new state-of-the-art 84.28% recall of the gold entity in the Zero-shot EL dataset with just 50 candidates, compared to the previous 82.06% with 64 candidates. We report the results from extensive experimentation using our proposed model on both seen and unseen entity datasets. Our results suggest that our approach could be a useful complement to existing EL methods.

1 Introduction

Entity Linking (EL) aims at matching entity mentions in a document with entries in a knowledge base (KB) or a dictionary of entities. Accurately linking entity mentions to entities plays a key role in various natural language processing (NLP) tasks, including information extraction (Lin et al., 2012; Hasibi et al., 2016), KB population (Dredze et al., 2010), content analysis (Huang et al., 2018) and question answering (Li et al., 2020). EL finds application in many fields, including technical writing, digital humanities, and biomedical data analysis.

While EL systems typically rely on external KBs and assume entities at inference time known, real-world applications are usually accompanied by minimal to zero labeled data, highlighting the importance of approaches that can generalize to unseen entities. Logeswaran et al. (2019) introduced zero-shot EL, where mentions must be linked to unseen entities, without in-domain labeled data, given only the entities' text description. By comparing two texts - a mention in context and a candidate entity description - zero-shot EL appears closer to the reading comprehension task.

Most EL systems consist of two subsystems: i) Candidate Generation (CG), where for each entity mention the system retrieves candidate entities related to the mention and document, and ii) Candidate Ranking (CR), where the system chooses the most probable entity among the retrieved candidates. The goal of this work is to advance the state of the art in the CG phase (Wu et al., 2020), in order to set a higher accuracy *ceiling* for CR and for EL overall.

The contributions of this paper can be summarised as: i) We introduce a CG approach, based on BERT-based bi-encoder that exploits a new pooling function managing to encode mention and entities in the same dense space very effectively. ii) We achieve state-of-the-art results of 84.28% recall at top-50 candidates, compared to 82.06% at top-64 of (Wu et al., 2020), measured on the Zero-shot EL test dataset. We thus increase the CR recall by 3%, while requiring 21.88% fewer candidates, allowing for more accurate and faster inference. iii) State-of-the-art results on both seen and unseen entity sets, with and without entity type information, reveal the robustness of our model in both the typical and the zero-shot EL settings.

2 Related Work

Former work has pointed out the importance of building entity linking systems that can generalize to unknown named entities, either via a reduced candidate set or through a more robust candidate ranker. Our work falls into the first category, namely the candidate generation phase. For CG, traditional methods have been based on string comparison (Phan et al., 2017) and alias tables, lacking rich representation and thus being restricted to a small entity set. Over the last years, several researchers (Sil et al., 2012; Murty et al., 2018; Logeswaran et al., 2019) focused on frequency-based methods, with most of them following TF-IDF and BM25 approaches. Gillick et al. (2019) introduced a simple neural bi-encoder and showed that encoding mentions and entities in a dense space works well. Inspired by this idea, Wu et al. (2020) proposed the current state-of-the-art CG model, using a more robust transformer-based bi-encoder (Humeau et al., 2019). Their model uses a BERT-based bi-encoder to encode mentions and entity descriptions in a dense space, where the top-K entities are then retrieved based on the two vectors' maximum dot product. Our work extends (Wu et al., 2020) by investigating other than the default BERT (Devlin et al., 2019) pooling functions, additional entity type side-information, and alternate retrieval methods.

3 Our Approach

This section presents, in brief, the bi-encoder architecture, the mentions and entities input format, five additional, other than the default [CLS], pooling functions proposed for better sequences' representation, and at last, the similarity measures used to retrieve candidates.

3.1 Bi-encoder

Architecture We use a BERT-based bi-encoder, following (Wu et al., 2020) to model the mention-entity pairs. The mention context and the entity description are encoded into vectors that pass through a transformer (BERT) encoder. Upon the result, a function that reduces the sequence of vectors into a single one is used, typically being the [CLS] token of the last hidden layer of each transformer. We investigate five additional pooling functions leading to a better representation.

Sequences Representation The default (w.o. entity type) mention's representation is shaped by the mention itself and its surrounded context, following the representation from (Wu et al., 2020):

$$[CLS]\ ctxt_l\ [M_s]\ mention\ [M_e]\ ctxt_r\ [SEP]$$

In the case of entity type incorporation, we append in the beginning the mention's type as a special token along with the mention itself separated from the default input ($[CLS]$ [*type*] $mention$ [*M-SEP*] $ctxt_l$ ... $[SEP]$).

In accordance to this architecture, the default entity representation is also composed by the sub-words of the entity's title and description, separated by the special token $[ENT]$.

$$[CLS]\ title\ [ENT]\ description\ [SEP]$$

In case of entity type incorporation, the entity's type is appended in the beginning as a special token ($[CLS]$ [*type*] *title* $[ENT]$ *description* $[SEP]$).

Sub-words in both mention and entity representations are restricted to a predefined max length (see section 4.2). Namely, in the case of the mention input, we first locate the mention in the document and keep as much surrounding context. In contrast, we keep the title and as many possible sub-words of the description starting from the beginning of the entity input.

Regarding entity-type side information we incorporate 18 generic entity types in total, captured from recognizing mention's and entity title's entity types using spaCy (Honnibal et al., 2020) model[1]. The set of entity types includes the *UNK* type for unclassified mentions/entity titles. Also, all entity types are encoded as special tokens; consequently, sequence representations that make use of special tokens also consider the entity type embeddings.

Pooling Functions Sequence representation plays a significant role in a model's performance and researchers have suggested handling the variable-length input in many ways for NLP (Hagiwara, 2021). This section presents five additional pooling functions, other than the default [CLS], leading to better candidates: i) *Average of all tokens*: Averages the vectors of all tokens in the last hidden layer, ii) *Sum of all tokens*: Sums the vectors of all tokens in the last hidden layer, iii) *Average of special tokens*: Averages only the vectors of special tokens in the last hidden layer, iv) *Sum of special tokens*: Sums only the vectors of special tokens in the last hidden layer, and v) *Concatenation of special tokens*: Concatenates only the vectors of special tokens in the last hidden layer.

Optimization The score, $s(m, e_i)$, of the entity candidate e_i given a mention m is computed by the dot-product $\mathbf{y_m} \cdot \mathbf{y_{e_i}}$. The network is trained to maximize the score of the correct entity with respect to the entities of the same batch. For each training pair (m_i, e_i) in a batch of B pairs, the loss

[1] https://github.com/explosion/spacy-models/releases/tag/en_core_web_lg-2.3.0

is computed as:

$$L(m_i, e_i) = -s(m_i, e_i) + log(\sum_{j=1}^{B} \exp(s(m_i, e_j)))$$

3.2 Retrieval Methods

We retrieve candidates by computing the similarity between each mention's context with each entity and construct the candidate set with the h more similar ones. To find the similarity between two vectors $\mathbf{A} = [a_1, a_2, ..., a_n]$ and $\mathbf{B} = [b_1, b_2, ..., b_n]$, the cosine, Euclidean and dot product similarity measures were used.

4 Experiments

4.1 Datasets & Evaluation Metric

We evaluate our model on three distinct datasets, including both the zero-shot and the standard EL setting to show its efficiency across all EL settings. The first dataset is the Zero-shot EL (Zeshel) dataset[2], the prevailing benchmark for zero-shot EL. The dataset includes documents from sixteen distinct domains, among which there is no overlap in the different splits. Precisely, the training set includes 49,275 labeled mentions, while the validation and test sets consist of 10K unseen mentions each. As a second dataset, we use the extra 5K mentions of seen entities from the Zeshel training set to explore the models' generalization on the typical EL setting. The third dataset is the AIDA CoNLL-YAGO (Hoffart et al., 2011) dataset, a dataset that contains assignments of entities to the mentions of named entities annotated for the original [CoNLL] 2003 NER task. The entities are identified by YAGO2 entity identifier, by Wikipedia URL, or by Freebase mid.

To evaluate the performance of our model, we report the recall at top-K, i.e. we assess the performance on the subset of test instances for which the gold entity is among the top-k retrieved candidates.

4.2 Model Settings

We use the *bert-base-uncased* model with hidden layer dimension $D_h = 768$, which we fine-tune with maximum sequence length $D_t = 128$. For fine-tuning, we assigned $batch\_size = 8$, $epochs = 5$, all the BERT's layers are updated during back-propagation, learning rate $lr = 3e^{-5}$ and $weight\_decay = 0.01$. Moreover, we fine-tune our model using the Adam optimization scheme (Loshchilov and Hutter, 2017) with $\beta_1 = 0.9, \beta_2 = 0.999$ and a linear learning rate decay schedule. We minimize loss using cross entropy criterion.

Experiments were conducted on a PC with 15GB RAM, an AMD FX-8350 @4.00 GHz and NVIDIA TITAN X with 12GB. Training took about 150min, while CG on the zero-shot test set took 90min, 60min and 50min for the cosine, Euclidean and dot product methods, respectively.

4.3 Model Comparison

We compare our CG model and its variants across three entity linking benchmark datasets, covering both the standard and the zero-shot supervision settings. However, our focus is primarily on the zero-shot setting due to the inherent difficulties. Therefore, for the zero-shot EL evaluation, we compare our best CG model against its variants and previous state-of-the-art methods on the test set of the *Zeshel* dataset, while for the standard EL evaluation, we compare our best CG model against its variants on the *Heldout Train Seen* and *AIDA CoNLL-YAGO* datasets, which include entities that have been already seen in the training data.

Table 1 shows our model to significantly outperform the previous state-of-the-art works in the zero-shot set with fewer candidates regardless of chosen pooling function. Precisely, the best version of our model achieves 2.22 and 15.15 higher recall than (Wu et al., 2020) and (Logeswaran et al., 2019) respectively, using 21.88% fewer candidates. Additionally, we observe that both datasets following the typical EL set, where mentions are linked to previously seen entities in the training data, exhibit significantly higher recall than the zero-shot (Zeshel) dataset. However, it seems that the performance of each pooling function differs for the two datasets. Still though, concatenating the special tokens of the final input representation yields the best results in two out of three datasets.

In order to provide a better overview of the proposed model, we show in Figure 1 our best model's recall across various top-K retrieved candidates for all three datasets. Following observations from Table 1, the *Heldout Train Seen* and *AIDA CoNLL-YAGO* datasets present a far better recall than the *Zeshel* dataset. At the same time, it is worth mentioning the 80% and 86% recall achieved from the first candidate for the two datasets, respectively. Regarding the *Zeshel* dataset, although choosing more candidates could further increase recall, we

[2] https://github.com/lajanugen/zeshel

Method	Test*	Heldout Train Seen**	AIDA CoNLL-YAGO**	Top-K
(Logeswaran et al., 2019)	69.13	-	-	64
(Wu et al., 2020)	82.06	-	-	64
Ours (sum)	79.64	97.60	99.67	50
Ours (sum special)	82.90	98.26	99.69	50
Ours (CLS)	83.52	98.42	**99.78**	50
Ours (avg special)	83.83	98.76	99.64	50
Ours (avg)	84.06	98.56	99.75	50
Ours (conc special)	**84.28**	**98.86**	99.75	50

Table 1: Candidate Generation recall on the test set of Zeshel, its provided Heldout Train Seen, and the AIDA CoNLL-YAGO datasets. Baseline models are compared against our model's various sequence representation alternatives, with selected pooling function noted in parentheses. $*$ indicates zero-shot datasets and $**$ datasets following the typical EL set where all mentions have been already seen in the training data. All models use the dot product to retrieve candidates, while no side-information is used.

decided on fifty candidates due to its small enough size and performance improvement over the rest baselines.

Moreover, we experimented with all mentioned retrieval methods (Euclidean, Cosine, Dot) for all datasets to find out that the dot product always held the best results. Lastly, you can view examples of retrieved entity candidates in the Appendix A.

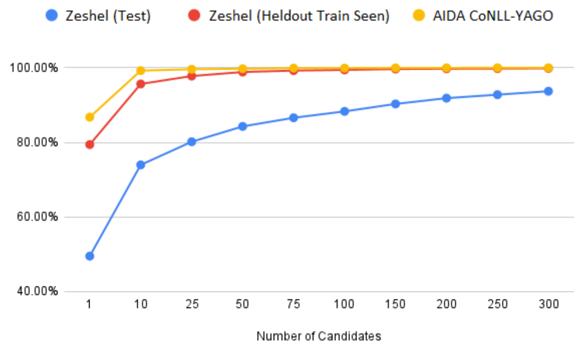

Figure 1: Top-K entity retrieval recall of *Ours (conc special)* model for the Zeshel Test, Heldout Train Seen and the AIDA CoNLL-YAGO datasets.

4.4 Effect of Entity Type Side Information

Analysis up to this point reveals the difficulty of linking mentions to unseen entities (zero-shot EL). For that reason, we wanted to test whether incorporating additional information, such as the entity type of the mentions, would boost our model's performance on the zero-shot set.

Results on Table 2 show that the addition of entity type in the majority of the representation techniques improves performance only slightly (up to 0.63%), while our proposed model continues to have the best results without incorporating additional information, a fact which strengthens our proposal. We attribute the low effect of the added entity types to the great number of unknown entities (60%) with manual data inspection showing that these are primarily associated with pronouns or coreferences.

	Recall@50	
Pooling Functions	w/o Ent.Type	Ent.Type
Ours (sum)	79.64	**80.27**
Ours (sum special)	82.90	**83.23**
Ours (CLS)	83.52	**83.83**
Ours (avg special)	83.83	**83.88**
Ours (avg)	84.06	**84.19**
Ours (conc special)	**84.28**	83.30

Table 2: Comparison of results with and without entity type side-information for each pooling function on Zeshel (Test) dataset. Reported results use the dot product to retrieve candidates.

5 Conclusion

We proposed a simple, yet effective CG model that sets a higher performance bar for CR and EL overall. Our model achieves a 2.22 higher threshold for CR and EL using 22% fewer candidates, compared to the previous state-of-the-art method in the zero-shot set. Our model accomplishes great in-domain results too, with the same parameters as in zero-shot EL, showing that the suggested method can be a valuable complement to any existing EL approach. Lastly, experiments using entity type side information highlight our model's robustness regardless of side information.

References

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.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 277–285.

Dan 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.

M. Hagiwara. 2021. *Real-World Natural Language Processing: Practical applications with deep learning*. Manning.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting entity linking in queries for entity retrieval. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 209–218.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Zhipeng Huang, Bogdan Cautis, Reynold Cheng, Yudian Zheng, Nikos Mamoulis, and Jing Yan. 2018. Entity-based query recommendation for long-tail queries. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(6):1–24.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*.

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.

Thomas Lin, Oren Etzioni, et al. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 84–88.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109.

Minh C Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2017. Neupl: Attention-based semantic matching and pair-linking for entity disambiguation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1667–1676.

Avirup Sil, Ernest Cronin, Penghai Nie, Yinfei Yang, Ana-Maria Popescu, and Alexander Yates. 2012. Linking named entities to any database. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 116–127.

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.

A Appendix: Entity Links Examples

Tables 3, 4, and 5 present some examples of mention - entity links candidates, generated from our proposed model.

Mention	... When he took it off , **a boy** working for Insector Haga swiped it and inserted the card P̈arasite Paracide ïnto his Deck as he ran away with it . Jonouchi managed to catch the boy and met up with Anzu , Bakura and Sugoroku as he did so	
	Haga ’ s helper (manga)	Haga ’ s helper (manga) Haga ’ s helper was a boy employed by Insector Haga to help him cheat in a Duel against Katsuya Jonouchi . Biography . Haga promised the boy a rare card if he could sneak the card " Parasite Paracide " into Jonouchi ’ s Deck . ...
	Queue cutter	Queue cutter The " queue cutter " is an elementary or junior high school kid , who appeared once in the manga . Biography . The boy cut past Yugi in a queue to a Capsule Monster Chess coin machine , outside Old Man Dentures store . ...
	Daichi	Daichi Daichi is one of the children in Crow ’ s care in " Yu - Gi - Oh ! 5D ’ s " . He has black hair in a bowl cut and dark blue eyes . ...

Table 3: Example of mention and top-3 retrieved entity candidates on the Zeshel (Test) dataset. The correct entity link is the first from the candidate set.

Mention	... He stole the Type 7 shuttlecraft " " , intending to join a freighter on Beltane IX and asked Captain Picard to tell **his father** he ’ s sorry but had to do this	
	Jack Crusher	Jack Crusher Lieutenant Commander Jack R . Crusher was a Starfleet officer . Considered by Jean - Luc Picard to have been his best friend , he served under Picard ’ s command on the . He was husband to Beverly Crusher and father to Wesley Crusher . ...
	Kurland	Kurland Kurland was a Human male aboard the in 2364 . He had one son , Jake Kurland . When Jake tried to leave the " Enterprise " - D with a shuttlecraft that year , he asked Captain Picard to tell his father he was sorry but he could not stay aboard the ship . Picard told him he should bring back the shuttle and tell this his father himself . ( ) ...
	Gabriel Lorca	Gabriel Lorca Captain Gabriel Lorca was a male Human Starfleet officer who lived during the mid - 23rd century . He served as the commanding officer on board at least one Federation starship , ...

Table 4: Example of mention and top-3 retrieved entity candidates on the Zeshel (Test) dataset. The correct entity link is the second from the candidate set.

Mention	... Riker argues with him and is generally uncooperative . Remmick asks La Forge in engineering about **the incident** with Kosinski and the Traveler , and La Forge is forced to acknowledge that the captain lost control of the ship	
	USS Enterprise bridge holoprogram	USS Enterprise bridge holoprogram The USS " Enterprise " bridge holoprogram was a holodeck recreation of the bridge of the original . The program was accessed by Captain Montgomery Scott when he was on board the in 2369 after having been rescued from transporter stasis on the . ...
	Captain ’ s log , USS Enterprise (NCC - 1701)	Captain ’ s log , USS Enterprise (NCC - 1701) The captain ’ s log on the was the method used by the commanding officer to record the ship ’ s events . These logs included Captain James T . Kirk ’ s famous five - year mission . ( ) ...
	Traffic accident	Traffic accident A traffic accident was an incident involving vehicle s and their occupants in which the vehicle in question malfunctioned or crashed . Some accidents resulted in death . In 1930 , Edith Keeler died in a traffic accident after she crossed the street to find out why James T . Kirk had left her abruptly . ...

Table 5: Example of mention and top-3 retrieved entity candidates on the Zeshel (Test) dataset. The correct entity link is not part of the candidate set.