Linking Entities to Unseen Knowledge Bases with Arbitrary Schemas

Yogarshi Vyas **Miguel Ballesteros**
Amazon AI
{yogarshi, ballemig}@amazon.com

Abstract

In entity linking, mentions of named entities in raw text are disambiguated against a knowledge base (KB). This work focuses on linking to unseen KBs that do not have training data and whose schema is unknown during training. Our approach relies on methods to flexibly convert entities with several attribute-value pairs from arbitrary KBs into flat strings, which we use in conjunction with state-of-the-art models for zero-shot linking. We further improve the generalization of our model using two regularization schemes based on shuffling of entity attributes and handling of unseen attributes. Experiments on English datasets where models are trained on the CoNLL dataset, and tested on the TAC-KBP 2010 dataset show that our models are 12% (absolute) more accurate than baseline models that simply flatten entities from the target KB. Unlike prior work, our approach also allows for seamlessly combining multiple training datasets. We test this ability by adding both a completely different dataset (Wikia), as well as increasing amount of training data from the TAC-KBP 2010 training set. Our models are more accurate across the board compared to baselines.

1 Introduction

Entity linking consists of linking mentions of entities found in text against canonical entities found in a target *knowledge base* (KB). Early work in this area was motivated by the availability of large KBs with millions of entities (Bunescu and Paşca, 2006). Most subsequent work has followed this tradition of linking to a handful of large, publicly available KBs such as Wikipedia, DBPedia (Auer et al., 2007) or the KBs used in the now decade-old TAC-KBP challenges (McNamee and Dang, 2009; Ji et al., 2010). As a result, previous work always assumes complete knowledge of the *schema* of the target KB that entity linking models are trained for, *i.e.* how many and which *attributes* are used to represent entities in the KB. This allows training supervised machine learning models that exploit the schema along with labeled data that link mentions to this *a priori* known KB. However, this strong assumption breaks down in scenarios which require linking to KBs that are not known at training time. For example, a company might want to automatically link mentions of its products to an internal KB of products that has a rich schema with several attributes such as product category, description, dimensions, etc. It is very unlikely that the company will have training data of this nature, *i.e.* mentions of products linked to its database.

Our focus is on linking entities to unseen KBs with arbitrary schemas. One solution is to annotate data that can be used to train specialized models for each target KB of interest, but this is not scalable. A more generic solution is to build entity linking models that work with arbitrary KBs. We follow this latter approach and build entity linking models that link to target KBs that have not been observed during training.[1] Our solution builds on recent models for zero-shot entity linking (Wu et al., 2020; Logeswaran et al., 2019). However, these models assume the same, simple KB schema during training and inference. We generalize these models to handle different KBs during training and inference, containing entities represented with an arbitrary set of *attribute-value* pairs.

This generalization relies on two key ideas. First, we convert KB entities into strings that are consumed by the models for zero-shot linking. Central to the string representation are special tokens called **attribute separators**, which represent frequently occurring attributes in the training KB(s), and carry over their knowledge to unseen KBs during inference (Section 4.1). Second, we generate more flexible string representations by shuffling entity attributes before converting them to strings,

[1] "Unseen KBs" refers to scenarios where we neither know the entities in the KB, nor its schema.

	Generic EL	Zero-shot EL (Logeswaran et al., 2019)	Linking to any DB (Sil et al., 2012)	This work
Test entities not seen during training		✓	✓	✓
Test KB schema unknown				✓
Out-of-domain test data		✓		✓
Unrestricted Candidate Set	✓	✓		✓

Table 1: This table compares the entity linking framework in the present work with those in previous work.

and by stochastically removing attribute separators to generalize to unseen attributes (Section 4.2).

Our primary experiments are cross-KB and focus on English datasets. We train models to link to one KB during training (*viz.* Wikidata), and evaluate them for their ability to link to an unseen KB (*viz.* the TAC-KBP Knowledge Base). These experiments reveal that our model with **attribute-separators** and the two generalization schemes are 12–14% more accurate than the baseline zero-shot models. Ablation studies reveal that all components individually contribute to this improvement, but combining all of them yields the most accurate models.

Unlike previous work, our models also allow seamless mixing of multiple training datasets which link to different KBs with different schemas. We investigate the impact of training on multiple datasets in two sets of experiments involving additional training data that links to **(a)** a third KB that is different from our original training and testing KBs, and **(b)** the same KB as the test data. These experiments reveal that our models perform favorably under all conditions compared to baselines.

2 Background

Conventional entity linking models are trained and evaluated on the same KB, which is typically Wikipedia, or derived from Wikipedia (Bunescu and Paşca, 2006; Ling et al., 2015). This limited scope allows models to use other sources of information to improve linking, including alias tables, frequency statistics, and rich metadata.

Beyond Conventional Entity Linking There have been several attempts to go beyond such conventional settings, *e.g.* by linking to KBs from diverse domains such as the biomedical sciences (Zheng et al., 2014; D'Souza and Ng, 2015) and music (Oramas et al., 2016) or even being completely domain and language independent (Wang et al., 2015; Onoe and Durrett, 2020). Lin et al. (2017) discuss approaches to link entities to a KB that simply contains a list of names without any other information. Sil et al. (2012) use database-agnostic features to link against arbitrary databases. However, their approach still requires training data from the target KB. In contrast, this work aims to train entity linking models that do not rely on training data from the target KB, and can be trained on arbitrary KBs, and applied to a different set of KBs. Pan et al. (2015) also do *unsupervised* entity linking by generating rich context representations for mentions using Abstract Meaning Representations (Banarescu et al., 2013), followed by unsupervised graph inference to compare contexts. They assume a rich target KB that can be converted to a connected graph. This works for Wikipedia and adjacent resources but not for arbitrary KBs. Logeswaran et al. (2019) introduce a novel zero-shot framework to "develop entity linking systems that can generalize to unseen specialized entities". Table 1 summarizes differences between our framework and those from prior work.

Contextualized Representations for Entity Linking Models in this work are based on BERT (Devlin et al., 2019). While many studies have tried to explain the effectiveness of BERT for NLP tasks (Rogers et al., 2020), the work by Tenney et al. (2019) is most relevant as they use probing tasks to show that BERT encodes knowledge of entities. This has also been shown empirically by many works that use BERT and other contextualized models for entity linking and disambiguation (Broscheit, 2019; Shahbazi et al., 2019; Yamada et al., 2020; Févry et al., 2020; Poerner et al., 2020).

3 Preliminaries

3.1 Entity Linking Setup

Entity linking consists of disambiguating entity mentions $\mathcal{M}$ from one or more documents to a target knowledge base, $\mathcal{KB}$, containing unique entities. We assume that each entity $e \in \mathcal{KB}$ is represented using a set of attribute-value pairs

$\{(k_i, v_i)\}_{i=1}^{n}$. The attributes k_i collectively form the *schema* of $\mathcal{KB}$. The disambiguation of each $m \in \mathcal{M}$ is aided by the *context* c in which m appears.

Models for entity linking typically consist of two stages that balance recall and precision.

1. **Candidate generation**: The objective of this stage is to select K candidate entities $\mathcal{E} \subset \mathcal{KB}$ for each mention $m \in \mathcal{M}$, where K is a hyperparameter and $K << |\mathcal{KB}|$. Typically, models for candidate generation are less complex (and hence, less precise) than those used in the following (re-ranking) stage since they handle all entities in $\mathcal{KB}$. Instead, the goal of these models is to produce a small but high-recall candidate list $\mathcal{E}$. Ergo, the success of this stage is measured using a metric such as recall@K *i.e.* whether the candidate list contains the correct entity.

2. **Candidate Reranking**: This stage ranks the candidates in $\mathcal{E}$ by how likely they are to be the correct entity. Unlike candidate generation, models for re-ranking are typically more complex and oriented towards generating a high-precision ranked list since the objective of this stage is to identify the most likely entity for each mention. This stage is evaluated using precision@1 (or accuracy) *i.e.* whether the highest ranked entity is the correct entity.

In traditional entity linking, the training mentions $\mathcal{M}_{train}$ and test mentions $\mathcal{M}_{test}$ both link to the same KB. Even in the zero-shot settings of Logeswaran et al. (2019), while the training and target domains and KBs are mutually exclusive, the schema of the KB is constant and known. On the contrary, our goal is to link test mentions $\mathcal{M}_{test}$ to a knowledge base $\mathcal{KB}_{test}$ which is not known during training. The objective is to train models on mentions $\mathcal{M}_{train}$ that link to $\mathcal{KB}_{train}$ and directly use these models to link $\mathcal{M}_{test}$ to $\mathcal{KB}_{test}$.

3.2 Zero-shot Entity Linking

The starting point (and baselines) for our work are the state-of-the-art models for zero-shot entity linking, which we briefly describe here (Wu et al., 2020; Logeswaran et al., 2019).[2]

Candidate Generation Our baseline candidate generation approach relies on similarities between mentions and candidates in a vector space to identify the candidates for each mention (Wu et al., 2020) using two BERT models. The first BERT model encodes a mention m along with its context c into a vector representation $\mathbf{v}_m$. $\mathbf{v}_m$ is obtained from the pooled representation captured by the [CLS] token used in BERT models to indicate the start of a sequence. In this encoder, a binary (0/1) indicator vector is used to identify the mention span. The embeddings for this indicator vector (indicator embeddings) are added to the token embeddings of the mention as in Logeswaran et al. (2019). The second unmodified BERT model (*i.e.* not containing the indicator embeddings as in the mention encoder) independently encodes each $e \in \mathcal{KB}$ into vectors. The candidates $\mathcal{E}$ for a mention are the K entities whose representations are most similar to $\mathbf{v}_m$. Both BERT models are fine-tuned jointly using a cross-entropy loss to maximize the similarity between a mention and its corresponding correct entity, when compared to other random entities.

Candidate Re-ranking The candidate re-ranking approach uses a BERT-based cross-attention encoder to jointly encode a mention and its context along with each candidate from $\mathcal{E}$ (Logeswaran et al., 2019). Specifically, the mention m is concatenated with its context on the left (c_l), its context on the right (c_r), and a single candidate entity $e \in \mathcal{E}$. An [SEP] token, which is used in BERT to separate inputs from different segments, is used here to separate the mention in context, from the candidate. This concatenated string is encoded using BERT[3] to obtain, $\mathbf{h}_{m,e}$ a representation for this mention/candidate pair (from the [CLS] token). Given a candidate list $\mathcal{E}$ of size K generated in the previous stage, K scores are generated for each mention, which are subsequently scored using a dot-product with a learned weight vector ($\mathbf{w}$). Thus,

$$\mathbf{h}_{m,e} = \text{BERT}(\text{[CLS]}\ c_l\ m\ c_r\ \text{[SEP]}\ e\ \text{[SEP]}),$$
$$\mathit{score}_{m,e} = \mathbf{w}^T \mathbf{h}_{m,e}.$$

The candidate with the highest score is chosen as the correct entity, *i.e.*

$$e^* = \arg\max_{i=1}^{K} \ \mathit{score}_{m,e_i}.$$

[2] We re-implemented these models and verified them by comparing results with those in the original papers.

[3] This BERT model also contains the indicator embeddings as described in candidate generation.

4 Linking to Unseen Knowledge Bases

The models in Section 3 were designed to operate in settings where the entities in the target KB were only represented using a textual description. For example, the entity *Douglas Adams* would be represented in such a database using a description as follows: "*Douglas Adams was an English author, screenwriter, essayist, humorist, satirist and dramatist. He was the author of The Hitchhiker's Guide to the Galaxy.*"

However, linking to unseen KBs requires handling entities with an arbitrary number and type of attributes. The same entity (*Douglas Adams*) can be represented in a different KB using attributes such as "name", "place of birth", etc. (top of Figure 1). This raises the question of whether such models, that harness the power of pre-trained language models, generalize to linking mentions to unseen KBs, including those without such textual descriptions. This section presents multiple ideas to this end.

4.1 Representing Arbitrary Entities using Attribute Separators

One way of using these models for linking against arbitrary KBs is by defining an *attribute-to-text* function f, that maps arbitrary entities with any set of attributes $\{k_i, v_i\}_{i=1}^n$ to a string representation e that can be consumed by BERT, *i.e.*

$$e = f(\{k_i, v_i\}_{i=1}^n).$$

If all entities in the KB are represented using such string representations, then the models described in Section 3 can directly be used for arbitrary schemas. This leads to the question: *how can we generate string representations for entities from arbitrary KBs such that they can be used for BERT-based models*? Alternatively, what form can f take?

A simple answer to this question is **concatenation** of the values v_i, given by

$$f(\{k_i, v_i\}_{i=1}^n) = v_1\ v_2\ ...\ v_n.$$

We can improve on this by adding some structure to this representation by teaching our model that the v_i belong to different segments. As in the baseline candidate re-ranking model, we do this by separating them with [SEP] tokens. We call this **[SEP]-separation**. This approach is also used by Logeswaran et al. (2019) and Mulang' et al. (2020)

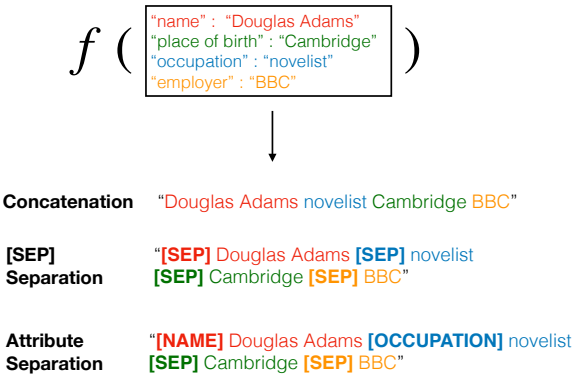

Figure 1: Shown here are three ways of representing an entity with arbitrary attribute-values (Section 4.1). **Concatenation** simply concatenates all values, **[SEP]-separation** separates values using [SEP] tokens, and **attribute separation** introduces special tokens based on frequently occurring attributes (which in this example are "name" and "occupation").

to separate the entity attributes in their respective KBs.

$$f(\{k_i, v_i\}_{i=1}^n) = \text{[SEP]}\ v_1\ \text{[SEP]}\ v_2\ ...\ \text{[SEP]}\ v_n$$

The above two definitions of f use the values v_i, but not the attributes k_i, which also contain meaningful information. For example, if an entity seen during inference has a *capital* attribute with the value "*New Delhi*", seeing the *capital* attribute allows us to infer that the target entity is likely to be a place, rather than a person, especially if we have seen the *capital* attribute during training. We capture this information using **attribute separators**, which are reserved tokens (in the vein of [SEP] tokens) corresponding to attributes. In this case,

$$f(\{k_i, v_i\}_{i=1}^n) = [K_1]\ v_1\ [K_2]\ v_2\ ...\ [K_n]\ v_n.$$

These $[K_i]$ tokens are not part of the default BERT vocabulary. Hence, we augment the default vocabulary with these new tokens and introduce them during training the entity linking model(s) based on the most frequent attribute values seen in the target KB of the training data, and randomly initialize their token embeddings. During inference, when faced with an unseen KB, we use attribute separators for only those attributes that have been observed during training, and use the [SEP] token for the remaining attributes.

Figure 1 illustrates the three instantiations of f. In all cases, attribute-value pairs are ordered in descending order of the frequency with which they appear in the training KB. Finally, since both

the candidate generation and candidate re-ranking models we build on use BERT, the techniques discussed here can be applied to both stages, but we only focus on re-ranking.

4.2 Regularization Schemes for Improving Generalization

Building models for entity linking against unseen KBs requires that such models do not overfit to the training data by memorizing characteristics of the training KB. This is done by using two regularization schemes that we apply on top of the candidate string generation techniques discussed in the previous section.

The first scheme, which we call **attribute-OOV**, prevents models from overtly relying on individual $[K_i]$ tokens and generalize to attributes that are not seen during training. Analogous to how out-of-vocabulary tokens are commonly handled (Dyer et al., 2015, *inter alia*), every $[K_i]$ token is stochastically replaced with the [SEP] token during training with probability p_{drop}. This encourages the model to encode semantics of the attributes in not only the $[K_i]$ tokens, but also in the [SEP] token, which is used when unseen attributes are encountered during inference.

The second regularization scheme discourages the model from memorizing the order in which particular attributes occur. Under **attribute-shuffle**, every time an entity is encountered during training, its attribute/values are randomly shuffled before it is converted to a string representation using the techniques from Section 4.1.

5 Experiments and Discussion

5.1 Data

Our held-out test bed is the TAC-KBP 2010 data (LDC2018T16) which consists of documents from English newswire, discussion forum and web data (Ji et al., 2010).[4] The target KB ($\mathcal{KB}_{test}$) is the TAC-KBP Reference KB and is built from English Wikipedia articles and their associated infoboxes (LDC2014T16).[5] Our primary training and validation data is the CoNLL-YAGO dataset (Hoffart et al., 2011), which consists of documents from the CoNLL 2003 Named Entity Recognition task (Tjong Kim Sang and De Meulder, 2003) linked to multiple KBs.[6] To ensure that our training and target KBs are different, we use *Wikidata* as our training KB.[7] Specifically, we use the subset of entities from Wikidata with a Wikipedia page. We ignore all mentions without a corresponding entity in the KB, both during training and inference, leaving the task of handling such NIL entities to future work. Finally, we use the Wikia dataset (Logeswaran et al., 2019) for experiments that investigate the impact of multiple datasets (Section 5.5).[8] Table 2 describes the sizes of these various datasets along with the number of entities in their respective KBs.

	Number of mentions	Size of target KB
CoNLL-YAGO (train)	18.5K	5.7M
CoNLL-YAGO (val.)	4.8K	
Wikia (train)	49.3K	0.5M
Wikia (val.)	10.0K	
TAC KBP 2010 (test)	1.7K	0.8M

Table 2: Number of mentions in our training, validation, and test sets, along with the number of entities in their respective KBs.

While covering similar domains, Wikidata and the TAC-KBP Reference KB have different schemas. Wikidata is more structured and entities are associated with statements represented using attribute-value pairs, which are short snippets rather than full sentences. The TAC-KBP Reference KB contains both short snippets like these, along with the text of the Wikipedia article of the entity. The two KBs also differ in size, with Wikidata containing almost seven times the number of entities in TAC KBP.

Both during training and inference, we only retain the 100 most frequent attributes in the respective KBs. The attribute-separators (Section 4.1) are created corresponding to the 100 most frequent attributes in the training KB. Candidates and mentions (with context) are represented using strings of 128 sub-word tokens each, across all models.

[4] https://catalog.ldc.upenn.edu/LDC2018T16

[5] https://catalog.ldc.upenn.edu/LDC2014T16

[6] http://resources.mpi-inf.mpg.de/yago-naga/aida/download/aida-yago2-dataset.zip

[7] Retrieved from https://dumps.wikimedia.org/wikidatawiki/entities/ in March, 2020.

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

5.2 Hyperparameters

All BERT models are uncased BERT-base models with 12 layers, 768 hidden units, and 12 heads with default parameters, and trained on English Wikipedia and the BookCorpus. The probability p_{drop} for **attribute-OOV** is set to 0.3. Both candidate generation and re-ranking models are trained using the BERT Adam optimizer (Kingma and Ba, 2015), with a linear warmup for 10% of the first epoch to a peak learning rate of 2×10^{-5} and a linear decay from there till the learning rate approaches zero.[9] Candidate generation models are trained for 200 epochs with a batch size of 256. Re-ranking models are trained for 4 epochs with a batch size of 2, and operate on the top 32 candidates returned by the generation model. Hyperparameters are chosen such that models can be run on a single NVIDIA V100 Tensor Core GPU with 32 GB RAM, and are not extensively tuned. All models have the same number of parameters except the ones with attribute-separators which have 100 extra token embeddings (of size 768 each).

Candidate generation Since the focus of our experiments is on re-ranking, we use a fixed candidate generation model for all experiments that combines the architecture of Wu et al. (2020) (Section 3) with **[SEP]-separation** to generate candidate strings. This model also has no knowledge of the test KB and is trained only once on the CoNLL-Wikidata dataset. It achieves a recall@32 of 91.25 when evaluated on the unseen TAC-KBP 2010 data.

5.3 Research Questions

We evaluate the re-ranking model (Section 3) in several settings to answer the following questions:

1. Do the attribute-to-text functions (Section 4.1) generate useful string representations for arbitrary entities? Specifically, can these representations be used with the re-ranking model (Section 3) to link to the unseen $\mathcal{KB}_{test}$?
2. Do all three key components—**attribute-separators** (Section 4.1), **attribute-shuffling**, and **attribute-OOV** (Section 4.2)—contribute equally to the final model?
3. Does training on more than one KB with different schemas help models in more accurately linking to $\mathcal{KB}_{test}$?

[9] https://gluon-nlp.mxnet.io/api/modules/optimizer.html#gluonnlp.optimizer.BERTAdam

Model	Accuracy
concatenation	47.2 ± 7.9
[SEP]-separation	49.1 ± 2.6
attribute-separation (no reg.)	54.7 ± 3.8
++**attribute-OOV**	56.2 ± 2.5
++**attribute-shuffle**	58.2 ± 3.6
++**attribute-OOV** + shuffle	61.6 ± 3.6

Table 3: Training on CoNLL-Wikidata and testing on the TAC-KBP 2010 test set reveals that using **attribute-separators** instead of [SEP] tokens yields string representations for candidates that result in more accurate models. Regularization schemes (Section 4.2) further improve accuracy to 61.6% without using any training data from the test KB.

4. Do improvements for generalizing to unseen $\mathcal{KB}_{test}$ also translate to scenarios where there is training data that also links to $\mathcal{KB}_{test}$?

For all experiments, we report the mean and standard deviation of the accuracy across five runs with different random seeds.

5.4 Main results

Our primary experiments focus on the first two research questions and study the accuracy of the model that uses the re-ranking architecture from Section 3 with the three core components introduced in Section 4 *viz.* **attribute-separators** to generate string representations of candidates, along with **attribute-OOV** and **attribute-shuffle** for regularization. We compare this against two baselines without these components that use the same architecture and use **concatenation** and **[SEP]-separation** instead of **attribute-separators**. As a reminder, all models are trained as well as validated on CoNLL-Wikidata and evaluated on the completely unseen TAC-KBP 2010 test set.

Results confirm that adding structure to the candidate string representations via [SEP] tokens leads to more accurate models compared to generating strings by concatenation (Table 3). Using **attribute-separators** instead of [SEP] tokens leads to an absolute gain of over 5% and handling unseen attributes via **attribute-OOV** further increases the accuracy to 56.2%, a 7.1% increase over the [SEP] baseline. These results show that the **attribute-separators** capture meaningful information about attributes, even when only a small number of attributes from the training data (15) are observed during inference.

Model	Accuracy
[SEP]-separation	62.6 ± 0.8
attribute-separation ++attribute-OOV + shuffle	66.8 ± 2.8

Table 4: Adding the Wikia dataset to training improves accuracy of both our model and the baseline, but our model still outpeforms the baseline by over 4%.

Shuffling attribute-value pairs before converting them to a string representation using **attribute-separators** also independently provides an absolute gain of 3.5% over the model which uses **attribute-separators** without shuffling. Overall, models that combine **attribute-shuffling** and **attribute-OOV** are the most accurate with an accuracy of 61.6%, which represents a 12% absolute gain over the best baseline model.

Prior work (Raiman and Raiman, 2018; Cao et al., 2018; Wu et al., 2020; Févry et al., 2020) reports higher accuracies on the TAC data but they are fundamentally incomparable with our numbers due to the simple fact that we are solving a different task with three key differences: **(1)** Models in prior work are trained and evaluated using mentions that link to the same KB. On the contrary, we show how far we can go without such in-KB training mentions. **(2)** The test KB used by these works is different from our test KB. Each entry in the KB used by prior work simply consists of the name of the entity with a textual description, while each entity in our KB is represented via multiple attribute-value pairs. **(3)** These models exploit the homogeneous nature of the KBs and usually pre-train models on millions of mentions from Wikipedia. This is beneficial when the training and test KBs are Wikipedia or similar, but is beyond the scope of this work, as we build models applicable to arbitrary databases.

5.5 Training on multiple unrelated datasets

An additional benefit of being able to link to multiple KBs is the ability to train on more than one dataset, each of which links to a different KB with different schemas. While prior work has been unable to do so due to its reliance on knowledge of $\mathcal{KB}_{test}$, this ability is more crucial in the settings we investigate, as it allows us to stack independent datasets for training. This allows us to answer our third research question. Specifically, we compare the **[SEP]-separation** baseline with our full model that uses **attribute-separators**, **attribute-shuffle**, and **attribute-OOV**. We ask whether the differences observed in Table 3 also hold when these models are trained on a combination of two datasets *viz.* the CoNLL-Wikidata and the Wikia datasets, before being tested on the TAC-KBP 2010 test set.

% of TAC training data	**[SEP]-sep.**	**Attribute-sep.** w/ reg.	**Attribute-sep.** w/o reg.
0%	49.1 ± 2.6	61.6 ± 3.6	
1%	62.4 ± 3.1	69.0 ± 0.5	70.0 ± 2.8
5%	70.1 ± 2.5	72.8 ± 1.5	76.0 ± 1.6
10%	74.5 ± 2.0	76.0 ± 0.8	77.8 ± 1.6
25%	80.1 ± 1.2	78.8 ± 0.4	80.8 ± 1.0
50%	81.8 ± 1.0	80.5 ± 0.4	82.8 ± 1.1
75%	83.1 ± 1.0	81.1 ± 0.2	84.0 ± 0.5
100%	84.1 ± 0.6	81.8 ± 0.9	84.9 ± 0.7
TAC-only		83.6 ± 0.7	83.8 ± 0.9

Table 5: Experiments with increasing amounts of training data that links to the inference KB reveal that models with **attribute separators** but without any regularization are the most accurate across the spectrum.

Adding the Wikia dataset to training increases the accuracy of the full model by 6%, from 61.6% to 66.8% (Table 4). In contrast, the baseline model observes a bigger increase in accuracy from 49.1% to 62.6%. While the difference between the two models reduces, the full model remains more accurate. These results also show that the seamless stacking of multiple datasets allowed by our models is effective empirically.

5.6 Impact of schema-aware training data

Finally, we investigate to what extent do components introduced by us help in linking when there is training data available that links to the inference KB, $\mathcal{KB}_{test}$. We hypothesize that while **attribute-separators** will still be useful, **attribute-OOV** and **attribute-shuffle** will be less useful as there is a smaller gap between training and test scenarios, reducing the need for regularization.

For these experiments, models from Section 5.4 are further trained with increasing amounts of data from the TAC-KBP 2010 training set. A sample of 200 documents is held out from the training data as a validation set. The models are trained with the exact same configuration as the base models, except with a smaller constant learning rate of 2×10^{-6} to not overfit on the small amounts of data.

Unsurprisingly, the accuracy of all models increases as the amount of TAC training data in-

Category	Mention	Gold Entity	Prediction	% Errors
Specific	... Richard Blumenthal of Connecticut and Roy Cooper of North Carolina ...	Connecticut	Hartford, Connecticut	33%
Generic	According to the NY Times, "Foreigners and Lhasa residents who witnessed the violence ...	The New York Times	The New York Times Company	6%
Context	... the Tombigbee River, some 25 miles (40 kilometers) north of Mobile. It's on a river ...	Mobile, Ohio	Mobile River	33%
Related	The striker arrived from his native city of Santos in a helicopter ...	Santos (city)	Santos, F.C.	21%
String	the apparent public support for Camara in Guinea had put the organisation in a quandary	Guinea	Senegal	7%

Table 6: Categorization of model errors, with an example of each, along with the number of errors observed for each category (out of 100). The underlined text under **Mention** marks the span containing the actual mention.

creases (Table 5).[10] As hypothesized, the smaller generalization gap between training and test scenarios makes the model with only **attribute separators** more accurate than the model with both **attribute separators** and regularization.

Crucially, the model with only **attribute separators** is the most accurate model across the spectrum. Moreover, the difference between this model and the baseline model sharply increases as the amount of schema-aware data decreases (*e.g.* when using 13 annotated documents, *i.e.* 1% of the training data, we get a 9% boost in accuracy over the model that does not see any schema-aware data). These trends show that our models are not only useful in settings without any data from the target KB, but also in settings where limited data is available.

5.7 Qualitative Analysis

Beyond the quantitative evaluations above, we further qualitatively analyze the predictions of the best model from Table 3 to provide insights into our modeling decisions and suggest avenues for improvements.

5.7.1 Improvements over baseline

First, we categorize all *newly correct mentions*, *i.e.* mentions that are correctly linked by the top model but incorrectly linked by the **[SEP]-separation** baseline by the *entity type* of the gold entity. This type is one of person (PER), organization (ORG), geo-political entity (GPE), and a catchall unknown category (UKN).[11] This categorization reveals that the newly correct mentions represent about 15% of the total mentions of the ORG, GPE, and UKN categories and as much as 25% of the total mentions of the PER category. This distributed improvement highlights that the relatively higher accuracy of our model is due to a holistic improvement in modeling unseen KBs across all entity types.

Why does PER benefit more than other entity types? To answer this, we count the fraction of mentions of each entity type that have at least one column represented using **attribute separators**. This counting reveals that approximately 56–58% of mentions of type ORG, GPE, and UKN have at least one such column. On the other hand, this number is 71% for PER mentions. This suggests that the difference is directly attributable to more PER entities having a column that has been modeled using **attribute separators**, further highlighting the benefits of this modeling decision.

5.7.2 Error Analysis

To identify the shortcomings of our best model, we categorize 100 random mentions that are incorrectly linked by this model into six categories (demonstrated with examples in Table 6), inspired by the taxonomy of Ling et al. (2015).

Under this taxonomy, a common error (33%) is predicting a more *specific* entity than that indicated by the mention (the city of Hartford, Connecticut, rather than the state). The reverse is also observed

[10] The 0% results are the same as those in Table 3.

[11] This entity typing is present in the KB.

(*i.e.* the model predicts a more *general* entity), but far less frequently (6%). Another major error category (33%) is when the model fails to pick up the correct signals from the *context* and assigns a similarly named entity of a similar type (*e.g.* the river Mobile, instead of the city Mobile, both of which are locations). 21% of the errors are cases where the model predicts an entity that is related to the gold entity, but is neither more specific, nor more generic, but rather of a different type (Santos Football Club instead of the city of Santos).

Errors in the last category occur when the model predicts an entity whose name has no string overlap with that of the gold entity or the mention. This likely happens when the signals from the context override the signals from the mention itself.

6 Conclusion

The primary contribution of this work is a novel framework for entity linking against unseen target KBs with unknown schemas. To this end, we introduce methods to generalize existing models for zero-shot entity linking to link to unseen KBs. These methods rely on converting arbitrary entities represented using a set of attribute-value pairs into a string representation that can be then consumed by models from prior work.

There is still a significant gap between models used in this work and schema-aware models that are trained on the same KB as the inference KB. One way to close this gap is by using automatic table-to-text generation techniques to convert arbitrary entities into fluent and adequate text (Kukich, 1983; McKeown, 1985; Reiter and Dale, 1997; Wiseman et al., 2017; Chisholm et al., 2017). Another promising direction is to move beyond BERT to other pre-trained representations that are better known to encode entity information (Zhang et al., 2019; Guu et al., 2020; Poerner et al., 2020).

Finally, while the focus of this work is only on English entity linking, challenges associated with this work naturally occur in multilingual settings as well. Just as we cannot expect labeled data for every target KB of interest, we also cannot expect labeled data for different KBs in different languages. In future work, we aim to investigate how we can port the solutions introduced here to multilingual settings as well develop novel solutions for scenarios where the documents and the KB are in languages other than English (Sil et al., 2018; Upadhyay et al., 2018; Botha et al., 2020).

Acknowledgements

The authors would like to thank colleagues from Amazon AI for many helpful discussions that shaped this work, and for reading and providing feedback on earlier drafts of the paper. They also thank all the anonymous reviewers for their helpful feedback.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web*, Lecture Notes in Computer Science, pages 722–735, Berlin, Heidelberg. Springer.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

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.

Samuel Broscheit. 2019. Investigating Entity Knowledge in BERT with Simple Neural End-To-End Entity Linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.

Razvan Bunescu and Marius Paşca. 2006. Using Encyclopedic Knowledge for Named entity Disambiguation. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 675–686, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from Wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 633–642, Valencia, Spain. Association for Computational Linguistics.

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.

Jennifer D'Souza and Vincent Ng. 2015. Sieve-Based Entity Linking for the Biomedical Domain. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 297–302, Beijing, China. Association for Computational Linguistics.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China. Association for Computational Linguistics.

Thibault Févry, Nicholas FitzGerald, and Tom Kwiatkowski. 2020. Empirical Evaluation of Pre-training Strategies for Supervised Entity Linking. In *Automated Knowledge Base Construction*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. *arXiv:2002.08909 [cs]*.

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, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *In Third Text Analysis Conference (TAC)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Karen Kukich. 1983. Design of a Knowledge-Based Report Generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Ying Lin, Chin-Yew Lin, and Heng Ji. 2017. List-only Entity Linking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 536–541, Vancouver, Canada. 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.

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, Florence, Italy. Association for Computational Linguistics.

Kathleen R. McKeown. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press, USA.

Paul McNamee and Hoa Trang Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*, volume 17, pages 111–113. National Institute of Standards and Technology (NIST) Gaithersburg, Maryland ....

Isaiah Onando Mulang', Kuldeep Singh, Chaitali Prabhu, Abhishek Nadgeri, Johannes Hoffart, and Jens Lehmann. 2020. Evaluating the Impact of Knowledge Graph Context on Entity Disambiguation Models. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2157–2160. Association for Computing Machinery, New York, NY, USA.

Yasumasa Onoe and Greg Durrett. 2020. Fine-Grained Entity Typing for Domain Independent Entity Linking. *arXiv:1909.05780 [cs]*.

Sergio Oramas, Luis Espinosa Anke, Mohamed Sordo, Horacio Saggion, and Xavier Serra. 2016. ELMD: An Automatically Generated Entity Linking Gold Standard Dataset in the Music Domain. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3312–3317, Portorož, Slovenia. European Language Resources Association (ELRA).

Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight. 2015. Unsupervised Entity Linking with Abstract Meaning Representation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1130–1139, Denver, Colorado. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. *arXiv:1911.03681 [cs]*.

Jonathan Raiman and Olivier Raiman. 2018. DeepType: Multilingual Entity Linking by Neural Type System Evolution. *arXiv:1802.01021 [cs]*.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What we know about how BERT works. *arXiv:2002.12327 [cs]*.

Hamed Shahbazi, Xiaoli Z. Fern, Reza Ghaeini, Rasha Obeidat, and Prasad Tadepalli. 2019. Entity-aware ELMo: Learning Contextual Entity Representation for Entity Disambiguation. *arXiv:1908.05762 [cs, stat]*.

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, Jeju Island, Korea. Association for Computational Linguistics.

Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. 2018. Neural cross-lingual entity linking. In *AAAI*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

Shyam Upadhyay, Nitish Gupta, and Dan Roth. 2018. Joint multilingual supervision for cross-lingual entity linking. In *EMNLP*.

Han Wang, Jin Guang Zheng, Xiaogang Ma, Peter Fox, and Heng Ji. 2015. Language and Domain Independent Entity Linking with Quantified Collective Validation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 695–704, Lisbon, Portugal. Association for Computational Linguistics.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in Data-to-Document Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. *arXiv:1911.03814 [cs]*.

Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2020. Global Entity Disambiguation with Pretrained Contextualized Embeddings of Words and Entities. *arXiv:1909.00426 [cs]*.

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.

Jin Guang Zheng, Daniel Howsmon, Boliang Zhang, Juergen Hahn, Deborah McGuinness, James Hendler, and Heng Ji. 2014. Entity Linking for Biomedical Literature. In *Proceedings of the ACM 8th International Workshop on Data and Text Mining in Bioinformatics*, DTMBIO '14, pages 3–4, Shanghai, China. Association for Computing Machinery.