Fine-Grained Entity Typing for Domain Independent Entity Linking

Yasumasa Onoe and **Greg Durrett**
Department of Computer Science
The University of Texas at Austin
{yasumasa, gdurrett}@cs.utexas.edu

Abstract

Neural entity linking models are very powerful, but run the risk of overfitting to the domain they are trained in. For this problem, a domain can be narrowly construed as a particular distribution of entities, as models can even overfit by memorizing properties of specific frequent entities in a dataset. We tackle the problem of building robust entity linking models that generalize effectively and do not rely on labeled entity linking data with a specific entity distribution. Rather than predicting entities directly, our approach models fine-grained entity properties, which can help disambiguate between even closely related entities. We derive a large inventory of types (tens of thousands) from Wikipedia categories, and use hyperlinked mentions in Wikipedia to distantly label data and train an entity typing model. At test time, we classify a mention with this typing model and use soft type predictions to link the mention to the most similar candidate entity. We evaluate our entity linking system on the CoNLL-YAGO (Hoffart et al., 2011) dataset and show that our approach outperforms prior domain-independent entity linking systems. We also test our approach in a harder setting derived from the WikilinksNED dataset (Eshel et al., 2017) where all the mention-entity pairs are unseen during test time. Results indicate that our approach generalizes better than a state-of-the-art neural model on the dataset.

1 Introduction

Historically, systems for entity linking to Wikipedia relied on heuristics such as anchor text distributions (Cucerzan, 2007; Milne and Witten, 2008), tf-idf (Ratinov et al., 2011), and Wikipedia relatedness of nearby entities (Hoffart et al., 2011). These systems have few parameters, making them relatively flexible across domains. More recent systems have typically been parameter-rich neural network models (Sun et al., 2015; Yamada et al., 2016; Francis-Landau et al., 2016; Eshel et al., 2017). Many of these models are trained and evaluated on data from the same domain such as the CoNLL-YAGO dataset (Hoffart et al., 2011) or WikilinksNED (Eshel et al., 2017; Mueller and Durrett, 2018), for which the training and test sets share similar distributions of entities. These strong models can potentially memorize a specific entity distribution during training rather than learning how to link entities more generally. As a result, apparently strong systems in one domain may not generalize to other domains without fine-tuning.

In this work, we aim to use feature-rich neural models for entity linking in a way that can effectively generalize across domains. We do this by framing the entity linking problem as a problem of prediction of very fine-grained entity types. Ambiguous entity references (e.g., different locations with the same name, the same movie released in different years) often differ in critical properties that can be inferred from context, but which neural bag-of-words and similar methods may be unable to tease out. We use an inventory of tens of thousands of types to learn such highly specific properties. This represents a much larger-scale tagset than past work using entity typing for entity linking, which has usually used hundreds of types (Gupta et al., 2017; Murty et al., 2018; Raiman and Raiman, 2018). Critically, type prediction is the only learned component of our model: our final entity prediction uses a very simple heuristic based on summing posterior type probabilities.

To train our typing model, we collect data from Wikipedia targeting a range of types in a domain of interest. This type set is lightly specialized to the target domain, but importantly, the set is determined on the basis of purely unlabeled data in the domain (lists of candidates for the identified

Figure 1: Examples selected from the WikilinksNED development set (Eshel et al., 2017). The mention (in bold) is resolved to the topmost entity in each case. These correct entities can be distinguished by their fine-grained Wikipedia categories.

mentions). Moreover, because we use such a large type inventory, our model captures a wide range of types and can handle entity linking in both narrow settings such as CoNLL-YAGO and broader domain settings such as WikilinksNED. Our typing model itself is adapted from past work on ultra-fine grained entity typing in a different setting (Choi et al., 2018; Onoe and Durrett, 2019). As a high-capacity neural network model, this model can train on millions of examples and achieve strong performance for even rare types.

Our contributions are as follows: (1) Formulating entity linking as purely an entity typing problem. (2) Constructing a distantly-supervised typing dataset based on Wikipedia categories and training an ultra-fine entity typing model on it. (3) Showing through evaluation on two domains that our model is more effective than a range of other approaches trained from out-of-domain data, even Wikipedia data specialized to that particular domain.

2 Motivation and Setup

Figure 1 shows two examples from the WikilinksNED development set (Eshel et al., 2017) which motivate our problem setup. In the example (a), the most frequent Wikipedia entity given the mention "Ant" is the insect ant. In the Wikipedia dump, source anchors "Ant" point to the Wikipedia article about the insect *Ant*[1] 96% of the time and points to *Apache_Ant* 0.8% of the time. Since *Apache_Ant* is very rare in the training data, models trained on broad domain data will often prefer *Ant* in many contexts.

Predicting categories here is much less problematic. Our category processing (described in the Training Data for Typing section) assigns this mention several categories including `Software`. Predicting `Software` in this context is relatively easy for a typing model given the indicative words in the context. Knowing this type is enough to disambiguate between these two entities independent of other clues, and notably, it is not really skewed by the relative rarity of the *Apache_Ant* title. The category information is shared across many entities, so we can expect that approximating the proper representation for the category information would be much more efficient than learning rare entities directly.

The example (b) adds another challenge since "Washington" can correspond to many different people or locations, some of which can occur in relatively similar contexts. Even a coarse-grained type inventory will distinguish between these mentions. However, more specific category information is needed to distinguish between *Washington_(state)* and *Washington,_D.C.* In this case, `States of the West Coast of the United States` would disambiguate between these, and context clues like "northwestern" can help identify this. This category is extremely fine-grained and we cannot assume an ability to predict it reliably; we discuss in the Training Data for Typing section how to get around this limitation by splitting categories into parts.

Finally, we note that a global linking system (Hoffart et al., 2011) can sometimes exploit relevant context information from related entities like Java (programming language). In this model, we focus on a purely local approach for simplicity and see how far this can go; this is also the most general for datasets like WikilinksNED where other reliable mentions may not be present close by.

Setup We focus on the entity linking (EL) task of selecting the appropriate Wikipedia entities for

[1]We use italics to denote *Wikipedia titles* and true type to represent `Wikipedia categories`.

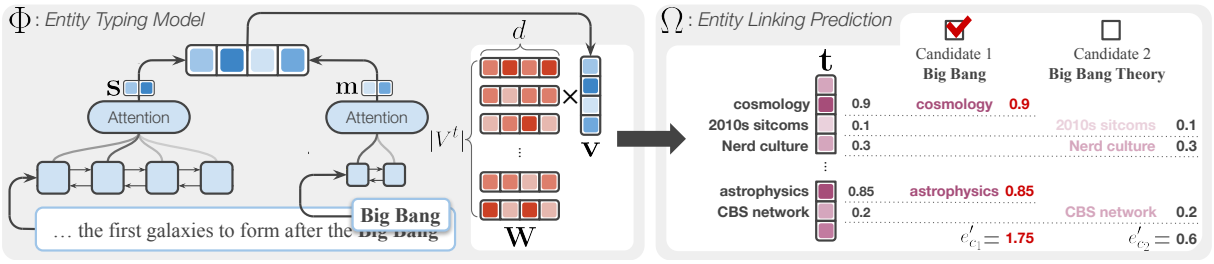

Figure 2: Entity typing for entity linking (ET4EL) model. Given a mention m and a sentence s, the entity typing model Φ computes the probability for each category. Then the entity linking predictor Ω makes the final prediction. In this example, the model chooses Big Bang over Big Bang Theory based on the scores (1.75 vs 0.6).

the mentions in context. We use m to denote a mention of an entity, s to denote a context sentence, e to denote a Wikipedia entity associated with the mention m, and C to denote a set of candidate entities. We also assume that we have access to a set of Wikipedia categories T corresponding to the Wikipedia entity e.

Suppose we have an entity linking dataset $\mathcal{D}_{\text{EL}} = \{(m, s, e, C)^{(1)}, \ldots, (m, s, e, C)^{(k)}\}$. In the standard entity linking setting, we train a model using the training set of $\mathcal{D}_{\text{EL}}$ and evaluate on the development/test sets of $\mathcal{D}_{\text{EL}}$. In our approach, we also have an entity typing dataset collected from hyperlinks in English Wikipedia $\mathcal{D}_{\text{Wiki}} = \{(m, s, T)^{(1)}, \ldots, (m, s, T)^{(l)}\}$. Since $\mathcal{D}_{\text{Wiki}}$ is derived from Wikipedia itself, this data contains a large number of common Wikipedia entities. This enables us to train a general entity typing model that maps the mention m and its context s to a set T of Wikipedia categories: $\Phi : (m, s) \rightarrow T$. Then, we use a scoring function Ω to make entity linking predictions based on the canidate set: $e = \Omega(\Phi(m, s), C)$. We evaluate our approach on the development/test sets of the existing entity linking data $\mathcal{D}_{\text{EL}}$. During training, we never assume access to labeled entity data $\mathcal{D}_{\text{EL}}$. Furthermore, by optimizing to predict Wikipedia categories T instead of an entity e, we can achieve a higher level of generalization across entities rather than simply memorizing our Wikipedia training data.

3 Model

Our model consists of two parts: a learned entity typing model and a heuristic (untrained) entity link predictor that depends only on the types.

3.1 Entity Typing Model

Figure 2 summarizes the model architecture of the entity typing model Φ. We use the attention-based model (Onoe and Durrett, 2019) designed for the fine-grained entity typing tasks (Gillick et al., 2014; Choi et al., 2018). This model takes a mention span in a sentence context, use span attention over vector representations of the mention span and its context, then aggregate that information to predict types. We follow (Onoe and Durrett, 2019) for our entity typing model design and hyperparameter choices.

Encoder The mention m and the sentence s are converted into contextualized word vectors s' and m' using ELMo (Peters et al., 2018). The sentence vectors s' are concatenated with the location embedding ℓ and fed into a bi-LSTM encoder followed by a span attention layer (Lee et al., 2017; Choi et al., 2018): $\mathbf{s} = \text{Attention}(\text{bi-LSTM}([s'; \ell]))$, where $\mathbf{s}$ is the final representation of the sentence s. The mention vectors m' are fed into another bi-LSTM and summed by a span attention layer to obtain the word-level mention representation: $\mathbf{m}^{\text{word}} = \text{Attention}(\text{bi-LSTM}(m'))$. We also use a 1-D convolution (Collobert et al., 2011) over the characters of the mention to generate a character-level mention representation $\mathbf{m}^{\text{char}}$. The final representation of the mention and the sentence is a concatenation of the three vectors: $\mathbf{v} = [\mathbf{s}; \mathbf{m}^{\text{word}}; \mathbf{m}^{\text{char}}] \in \mathbb{R}^d$. Unlike Onoe and Durrett (2019), we do not include the contextualized word vectors of the mention headword.[2]

Decoder We use $|V^t|$ to denote the size of the category vocabulary. Following previous work (Choi et al., 2018; Onoe and Durrett, 2019), we assume independence between the categories; thus, this boils down to a binary classification problem for each of the categories. The decoder is a sin-

[2]Compared to other models we considered, such as BERT, this approach was more stable and more scalable to use large amounts of Wikipedia data.

gle linear layer parameterized with $\mathbf{W} \in \mathbb{R}^{|V^t| \times d}$. The probabilities for all categories in the vocabulary are obtained by $\mathbf{t} = \sigma(\mathbf{W}\mathbf{v})$, where $\sigma(\cdot)$ is an element-wise sigmoid operation. The probability vector $\mathbf{t}$ is the final output from the entity typing model Φ.

Learning The entity typing model Φ is learned on the training examples consisting of (m, s, T) triples. The loss is a sum of binary cross-entropy losses over all categories over all examples. That is, the typing problem is viewed as independent classification for each category, with the mention-context encoder shared across categories. Formally, we optimize a multi-label binary cross entropy objective:

$$\mathcal{L} = -\sum_i y_i \cdot \log t_i + (1 - y_i) \cdot \log(1 - t_i), \tag{1}$$

where i are indices over categories, t_i is a score of the i-th category, and y_i is a binary representation of the gold categories.

3.2 Entity Linking Prediction

Once the entity typing model Φ is trained, we use the model output $\mathbf{t}$ to make entity linking predictions. Assume we have an example from the test set of an entity linking dataset: $x = (m, s, C)$, where C is a set of the candidate entities. We perform the forward computation of Φ and obtain the probability vector $\mathbf{t} = \Phi(m, s)$. Then, we score all the candidates in C using a scoring function Ω. Our choice of Ω is defined as the sum of probabilities for each type exhibited by the selected entity:

$$\begin{aligned} e'_c &= \sum_i t_i \cdot \mathbb{1}_{T_c}\left(V_i^t\right) \\ e &= \arg\max_e \left(e'_1, \ldots, e'_{|C|}\right), \end{aligned} \tag{2}$$

where e'_c is a score for a candidate entity c, $\mathbb{1}_{T_c}(\cdot)$ is an indicator function that is activated when i-th category in the vocabulary V_i^t is in the set of categories of the candidate entity c, and e is a predicted entity.

We observed that simply summing up the scores performed better than other options such as computing a log odds and averaging. Intuitively, summing benefits candidates with many categories, which biases the model towards more frequent entities in a beneficial way. It also rewards models with many correlated types, but we did not find other approaches that performed better.[3]

There are certain types of entities in Wikipedia whose categories do not mesh well with our prediction task. For example, the page *Employment* about the general concept only has the category `Employment`, making resolution of this concept challenging. In these cases, we back off to a mention-entity prior (see in the Preprocessing Evaluation Data section). We call our combined system ET4EL (entity typing for entity linking).

4 Training Data for Typing

To train the ET4EL model to cover a wide range of entities, we need access to a large set of entities labeled with types. We derive this data directly from Wikipedia: each hyperlinked occurrence of an entity on Wikipedia can be treated as a distantly supervised (Craven and Kumlien, 1999; Mintz et al., 2009) example from the standpoint of entity typing. The distant types for that mention are derived from the Wikipedia categories associated with the linked entity.

4.1 Data Construction

Annotation First, we collect all sentences that contain hyperlinks, internal links pointing to other English Wikipedia articles, from all articles on English Wikipedia. Our data is taken from the March 2019 English Wiki dump.[4] We divide up our entity typing training data by sentences. Given a sentence with a hyperlink, we use the hyperlink as a mention m, the whole sentence as a context sentence s, the destination of the hyperlink as an entity e, and the Wiki categories that are associated with e as a set of fine-grained types T. One sentence could have multiple hyperlinks. In this case, we create a tuple (m, s, e, T) for each of the hyperlinks. This process results 88M examples. Importantly, our training examples for typing are tuples of (m, s, T) since the ET4EL model is optimized towards the gold Wiki categories T and do not rely on the gold entity e.

Category Set The original Wikipedia categories are mostly fine-grained and lack general categories. For example, the Wiki entity *New_York_City* has fine-grained categories such as `Cities in New York (state)` and `Populated`

[3] We explored notions of type embeddings or creating a set of basis types using a singular value decomposition over type cooccurences, but did not find this helpful.

[4] https://dumps.wikimedia.org/enwiki/

Model	Input	Training Data	Supervision
ET4EL (this work)	mention, context	Wiki	mention-categories
Gupta et al. (2017) CDTE	document	Wiki	mention-entity
Lazic et al. (2015) Plato, sup	document	Wiki	mention-entity
Lazic et al. (2015) Plato, semi-sup	document	Wiki + 50M Web pages	mention-entity
Le and Titov (2019)	document	Wiki + 30k RCV1 docs	mention-entity
Standard EL Systems (local)	mention, context	domain-specific training set	mention-entity
Standard EL Systems (global)	document	domain-specific training set	mention-entity

Table 1: Assumptions and resources for different entity linking systems. Our model only requires supervision from Wikipedia and trains using typing supervision (from categories) only.

`places established in 1624`, but there are no general categories (e.g. `Cities`) that potentially useful to distinguish between two obviously different entities (e.g. location vs person). To add more general types, we expand the original categories if they contain prepositions.[5] We split the original category at the location of the first-occurring preposition. We chunk the left side into words and add to the category set. We add the right side, a prepositional phrase, to the category set without modification. Retaining the preposition helps to keep the relation information. We also retain the original category. For the two original categories above, the new categories `Cities, in New York (state)`, `Populated, places, established, in 1624` would be added to the category set.[6]

Training the Typing Model Since the total number of Wikipedia categories is very large (over 1 million), we train on a subset of the categories for efficiency. For a given test set, we only need access to categories that might possibly occur. We therefore restrict the categories to the most common n categories occurring with candidates in that dataset; note that this does not assume the existence of labeled target-domain data, only unlabeled.

To create the training set, we randomly select 6M examples that have at least one Wikipedia category from the restricted category vocabulary. We select 10k examples for the development set using the same procedure. The encoder may specialize somewhat to these types, but as we show later, it can handle large type sets and recognize diverse entity types (see the Results and Discussion section).

[5]We use 'in', 'from', 'for', 'of', 'by', 'for', 'involving.'

[6]Other splits are possible, e.g. extracting *20th century* from *20th century philosophers*. However, these are more difficult to reliably identify.

5 Experiments

We evaluate our approach on the development/test sets of the CoNLL-YAGO (Hoffart et al., 2011) dataset, which is a widely used entity linking benchmark. The CoNLL data consists of news documents and covers relatively narrow domains. Additionally, we test our model in a much harder setting where the mention-entity pairs are unseen during test time. We create the training, development, and test sets from the WikilinksNED dataset (Eshel et al., 2017), which contains a diverse set of ambiguous entities spanning more domains than the CoNLL data. We call this dataset Unseen-Mentions. The domain-specific training set is only used for the baseline models. The ET4EL model is still trained on the Wikipedia data. Unlike the CoNLL data, the examples in the Unseen-Mentions dataset are essentially single-sentence, meaning that resolution has to happen with limited context.

5.1 Preprocessing Evaluation Data

Candidate Selection For the CoNLL data, we use the publicly available candidate list, PPRforNED (Pershina et al., 2015) that gives 99% gold recall on the testa (development) and the testb (test) sets.

For the Unseen-Mentions data, we use a mention-entity prior $\hat{p}(e|m)$ to select candidate entities (Ganea and Hofmann, 2017). We compute $\hat{p}(e|m)$ using the count statistics from the Wiki dump. We rank the candidate entities based on $\hat{p}(e|m)$ and clip low frequency entities with a threshold 0.05. On average, this produces around 30 candidates per example and gives 88% gold recall on the development and test sets.

Category Vocabulary To reduce more than 1 million total Wikipedia categories to a more tractable number for a given dataset, we use count statistics from the candidates of the training ex-

amples in that data set. Note that this process uses this data in a totally unlabeled way. For each category, we count the number of associated unique mentions. We rank all the categories by the counts and select the top 60k categories as the category vocabulary.

5.2 Baselines

MOST FREQUENT ENTITY Given a mention m, we choose an entity with the highest mention-entity prior $\hat{p}(e|m)$. We compute $\hat{p}(e|m)$ using the count statistics from the Wiki dump.

COSINE SIMILARITY This baseline selects an entity with the highest cosine similarity between the context and entity vectors using the pretrained `word2vecf` (Levy and Goldberg, 2014). The context vector is obtained by mean pooling over the input word vectors.

GRU-ATTN Our implementation of the attention-based model introduced in Mueller and Durrett (2018). This model achieves state-of-the-art performance on the WikilinksNED dataset in the standard supervised setting. See Mueller and Durrett (2018) for more details.

CBOW+WORD2VEC This simpler baseline model[7] uses the pretrained embeddings and a simple bag-of-words mention-context encoder. That is, the encoder is unordered bag-of-words representations of the mention, the left context, and the right context. For each of three, the words are embedded and combined using mean pooling to give context representations. Similar to Eshel et al. (2017), we use `word2vecf` to initialize entity embeddings. We compare the context representations and the entity representation by following Mueller and Durrett (2018). The final representation is fed into a two-layer multilayer perceptron (MLP) with ReLU activations, batch normalization (Ioffe and Szegedy, 2015), and dropout (Srivastava et al., 2014).

Training Data for CoNLL Baselines To train our baselines in a comparable fashion, we create training examples (m, s, e) from the Wikipedia data $\mathcal{D}_{\text{Wiki}}$ to use for our learning-based baselines (GRU-ATTN and CBOW+WORD2VEC). We use the same mention-entity prior $\hat{p}(e|m)$, explained in the previous section, to select candidates for each training example.

We consider two variants of this training data. First, we train these baselines on a training set sampled uniformly from all of Wikipedia. Second, we give these baselines a more favorable transductive setting where the training entities from Wikipedia are restricted to only include entities that are candidates in the domain-specific training data. The CoNLL training set contains 2k unique entities. We collect 1.4M training examples from $\mathcal{D}_{\text{Wiki}}$ that cover these 2k CoNLL entities are covered; this training set should specialize these models to CoNLL fairly substantially, though they maintain our setting of not considering the training labels.

Training Data for Unseen-Mentions Baselines To ensure that all mentions in the development and test sets do not appear in the training set, we split the WikilinksNED training set into train, development, and test sets by unique mentions (15.5k for train, 1k for dev, and 1k for test). This results 2.2M, 10k, and 10k examples[8] respectively. Our learning-based baselines (GRU-ATTN and CBOW+WORD2VEC) are trained on the 2.2M training examples, which do not share any entities with the dev or test sets.

We also train the learning-based baselines on the Wikipedia data described in the Training Data for Typing section. Similar to the Unseen-Mentions data, we use a mention-entity prior $\hat{p}(e|m)$ to select candidate entities. We obtain 2.5M training examples that have at least 2 candidate entities.

Comparison with other systems Table 1 compares assumptions and resources for different systems. The ET4EL model is a so-called local entity linking model, as it only uses a single mention and context, rather than a global model which does collective inference over the whole document (Ratinov et al., 2011; Hoffart et al., 2011). However, this allows us to easily support entity linking with little context, as is the case for WikilinksNED.

The chief difference from other models is that the ET4EL model is trained on data derived from Wikipedia, where the supervision comes from categories attached to entity mentions. Moreover, we

[7] This model shows comparable performance to GRU-ATTN, achieving 76.0 accuracy on the original test set of the WikilinksNED data. Our reimplementation of GRU-ATTN matches the original performance reported in Mueller and Durrett (2018) of 75.8.

[8] Development and test are subsampled from their "raw" sizes of 130k token-level examples.

Model	Dev	Test
MOST FREQUENT ENTITY	57.7	57.3
COSINE SIMILARITY	45.5	42.8
GRU+ATTN (Mueller and Durrett, 2018)	67.5	63.2
GRU+ATTN (Transduction)	82.0	75.3
CBOW+WORD2VEC	70.1	67.3
CBOW+WORD2VEC (Transduction)	84.6	77.5
ET4EL (this work)	**88.1**	**85.9**
Gupta et al. (2017) CDTE	84.9	82.9
Lazic et al. (2015) Plato, sup	-	79.7
Lazic et al. (2015) Plato, semi-sup†	-	86.4
Le and Titov (2019)‡	-	89.7

Table 2: Accuracy on the CoNLL development set (testa) and the CoNLL test set (testb). †: trained with additional large unlabeled data. ‡: uses in-domain unlabeled data (RCV1). Our model outperforms the baselines and models using similar data from prior work.

Amount of Context	Dev
Sentence	83.8
Sentence + left & right 50 tokens	84.5
Sentence + 1st doc sentence	**88.1**

Table 3: Accuracy on the CoNLL development set (testa) with different inputs. Sentence: no additional context. Sentence + left & right 50 tokens: added the left and right 50 tokens of the mention span. Sentence + 1st Sentence: added the first sentence of the document.

Category Size	1k	5k	30k	60k
Dev	85.1	85.6	87.1	**88.1**

Table 4: Accuracy on the CoNLL development set (testa) with the different category sizes.

only use Wikipedia as a source of training data; some other work like Le and Titov (2019) uses unlabeled data from the same domain as the CoNLL-YAGO test set.

6 Results and Discussion

6.1 CoNLL-YAGO

Table 2 shows accuracy of our model and baselines. Our model outperforms all baselines by a substantial margin. The MOST FREQUENT ENTITY baseline performs poorly on both development and test set. Interestingly, the simpler CBOW+WORD2VEC model is the strongest baseline here, outperforming the GRU+ATTN model in both general and transductive settings. Our model achieves the strongest performance on both the dev and test data. Interestingly, our model also has a much smaller drop from dev to test (only losing 2 points) compared to the transductive models (which drop by 7 points). The CoNLL testb set is slightly "out-of-domain" for the training set with respect to the time period it was drawn from, indicating that our method may have better generalization than more conventional neural models in the transductive setting.

Model	Training	Test
MOST FREQUENT ENTITY	Wiki	54.1
COSINE SIMILARITY	Wiki	21.7
GRU+ATTN (Mueller and Durrett, 2018)	in-domain	41.2
GRU+ATTN	Wiki	43.4
CBOW + WORD2VEC†	in-domain	43.0
CBOW + WORD2VEC	Wiki	38.0
ET4EL (this work)	Wiki	**62.2**

Table 5: Accuracy on the Unseen-Mentions test set. Our model substantially outperforms neural entity linking models in this setting.

We also list the state-of-the-art domain independent entity linking systems. Our model outperforms the full CDTE model of Gupta et al. (2017), as well as Plato in the supervised setting (Lazic et al., 2015), which is the same setting as ours. Our model is competitive with Plato in the semi-supervised setting, which additionally uses 50 million documents as unlabeled data. Le and Titov (2019)'s setting is quite different from ours that their model is a global model (requires document input) and trained on Wikipedia and 30k newswire documents from the Reuters RCV1 corpus (Lewis et al., 2004). Their model is potentially trained on domain-specific data since the CoNLL-YAGO dataset is derived from the RCV1 corpus.

How much context information should we add? On the CoNLL dataset, sentences containing entity mentions are often quite short, but are embedded in a larger document. We investigate the most effective amount of context information to add to our typing model. Table 3 compares accuracy for the different amount of context. We test the context sentence only, the left and right 50 tokens of the mention span, and the first sentence of the document. Adding the left and right 50 tokens of the mention span improves the accuracy over the context sentence only, but the gap is marginal. Adding the first sentence of the document improves the accuracy over the context sentence only (no additional context) by 4 points.[9] Since the documents are news articles, the first sentence usually has meaningful information about the topics. This is especially useful when

[9] Our baselines use this setting as well since we found it to work the best.

	Total			1-100			101-500			501-10000			10001+		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
ET4EL (this work)	76.2	46.1	57.5	79.7	61.3	69.3	79.0	39.9	53.0	76.2	40.1	52.5	76.5	37.0	49.9

Table 6: Macro-averaged P/R/F1. Entity typing performance on the categories grouped by frequency. (1-100) is the most frequent group, and (10001+) is the least frequent group.

the document is a list of sports results, and a sentence does not have rich context. For example, one sentence is "Michael Johnson (U.S.) 20.02", which is highly uninformative, but the first sentence of the document is "ATHLETICS - BERLIN GRAND PRIX RESULTS." Our model correctly predicts *Michael_Johnson_(sprinter)* after giving more context information about the sports.

Does the category vocabulary size matter? We show the performance on the development set with different category sizes. As we can see in Table 4, the development accuracy monotonically increases as the category size goes up. Even the 1k most frequent category set can achieve reasonable performance, 85% accuracy. The model is able to make use of even very fine-grained categories to make correct predictions.

6.2 Unseen-Mentions

Table 5 compares accuracy of our model and baselines on this dataset. Our model achieves the best performance in this setting, better than all baselines. Notably, the GRU+ATTN model, which achieves state-of-the-art performance on WikilinksNED, performs poorly, underperforming the MOST FREQUENT ENTITY baseline. The simpler CBOW+WORD2VEC model with the frozen entity embeddings shows slightly better performance than the GRU+ATTN model, implying that the model suffers from overfitting to the training data. The poor performance by these two models trained on the domain-specific data suggests that dealing with unseen mention-entity pairs is challenging even for these vector-based approaches trained with similar domain data, indicating that entity generalization is a major factor in entity linking performance. In addition, the GRU+ATTN model trained on the Wikipedia data also performs poorly.

The baseline models trained on the domain-specific data even make mistakes in easy cases such as disambiguating between `PERSON` and `LOCATION` entities. For example, a mention spans is **[Kobe]**, and an associated entity could be *Kobe_Bryant*, a former basketball player, or *Kobe*, a city in Japan. Those baseline models guess *Kobe_Bryant* correctly but get confused with *Kobe*. Our model predict both entities correctly; the context is usually indicative enough.

6.3 Typing Analysis

In the Training Data for Typing section, we described how we added more general types to the category set. We compare the original Wikipedia category set and the expanded category set on the CoNLL development set. Using 30k categories in both settings, the original set and expanded set achieve accuracies of 84.4 and 87.1 respectively, showing that our refined type set helps substantially.

Table 6 shows the typing performance on the 60k categories grouped by frequency. The first group (1-100) consists of the 100 most frequent categories. The forth group (10001+) is formed with the least frequent categories. Precision is relatively high for all groups. The first group (1-100) achieves the highest precision, recall, and F1, possibly leveraging the rich training examples. Recall drastically decreases between the first group (1-100) and the subsequent groups, which suggests the model has some difficulty accounting for the imbalanced nature of the classification of rare tags.

We further look at the performance of selected individual categories. We observe that having rich training examples, in general, leads the high performance. For example, `births` occurs with more than 2k unique mentions in the training set and achieves P:99/R:89/F1:93.7. However, `history` has more than 900 unique mentions in the training set but only achieves P:76.9/R:6.1/F1:11.4. This might be related to the purity of mentions (and entities). Most of the mentions for `births` are person entities, and this category is consistently applied. On the other hand, `history` may not denote a well-defined semantic type.

7 Related Work

The internal link information in Wikipedia as supervision has been studied extensively in the field of entity linking and named entity disambiguation in the past decade (Bunescu and Paşca, 2006; Mihalcea and Csomai, 2007; Nothman et al., 2008; McNamee et al., 2009). Another approach utilizes manually annotated domain-specific data. Pre-neural systems used link structure (Milne and Witten, 2008) and tf-idf (Ratinov et al., 2011) vectors to represent entities. Recent approaches have used neural techniques (He et al., 2013; Sun et al., 2015; Francis-Landau et al., 2016), various joint models (Durrett and Klein, 2014; Nguyen et al., 2016). Learning pretrained entity representations from knowledge bases has also been studied for entity linking (Hu et al., 2015; Yamada et al., 2016, 2017; Eshel et al., 2017).

8 Conclusion

In this paper, we presented an entity typing approach that addresses the issue of overfitting to the entity distribution of a specific domain. Our approach does not rely on labeled entity linking data in the target domain and models fine-grained entity properties. With the domain independent setting, our approach achieves strong results on the CoNLL dataset. In a harder setting of unknown entities derived from the WikilinksNED dataset, our approach generalizes better than a state-of-the-art model on the dataset.

Acknowledgments

This work was partially supported by NSF Grant IIS-1814522, NSF Grant SHF-1762299, a Bloomberg Data Science Grant, and an equipment grant from NVIDIA. The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources used to conduct this research. Results presented in this paper were obtained using the Chameleon testbed supported by the National Science Foundation.

References

Razvan Bunescu and Marius Paşca. 2006. Using Encyclopedic Knowledge for Named entity Disambiguation. In *Proceedings of ACL*.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of ACL*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of ISMB*.

Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of EMNLP-CoNLL*.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *TACL*, 2:477–490.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of CoNLL*.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing semantic similarity for entity linking with convolutional neural networks. In *Proceedings of NAACL-HLT*.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of EMNLP*.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *CoRR*, abs/1412.1820.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity Linking via Joint Encoding of Types, Descriptions, and Context. In *Proceedings of EMNLP*.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning Entity Representation for Entity Disambiguation. In *Proceedings of ACL*.

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 EMNLP*.

Zhiting Hu, Poyao Huang, Yuntian Deng, Yingkai Gao, and Eric P. Xing. 2015. Entity Hierarchy Embedding. In *Proceedings of ACL*.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of ICML*.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *TACL*, 3:503–515.

Phong Le and Ivan Titov. 2019. Boosting entity linking performance by leveraging unlabeled documents. In *Proceedings of ACL*.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of EMNLP*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of ACL*.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

Paul McNamee, Mark Dredze, Adam Gerber, Nikesh Garera, Tim Finin, James Mayfield, Christine Piatko, Delip Rao, David Yarowsky, and Markus Dreyer. 2009. HLTCOE Approaches to Knowledge Base Population at TAC 2009. In *Proceedings of the 2009 Text Analysis Conference*.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of CIKM*.

David Milne and Ian Witten. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceeding of AAAI Wikipedia and AI Workshop*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL*.

David Mueller and Greg Durrett. 2018. Effective use of context in noisy entity linking. In *Proceedings of EMNLP*.

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 ACL*.

Thien Huu Nguyen, Nicolas Fauceglia, Mariano Rodriguez Muro, Oktie Hassanzadeh, Alfio Massimiliano Gliozzo, and Mohammad Sadoghi. 2016. Joint learning of local and global features for entity linking via neural networks. In *Proceedings of COLING*.

Joel Nothman, James R. Curran, and Tara Murphy. 2008. Transforming Wikipedia into Named Entity Training Data. In *Proceedings of ALTA Workshop*.

Yasumasa Onoe and Greg Durrett. 2019. Learning to Denoise Distantly-Labeled Data for Entity Typing. In *Proceedings of NAACL-HLT*.

Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized page rank for named entity disambiguation. In *Proceedings of NAACL-HLT*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*.

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

Lev Ratinov, Dan Roth, Doug Downey, and Mile Anderson. 2011. Local and Global Algorithms for Disambiguation to Wikipedia. In *Proceedings of ACL*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling Mention, Context and Entity with Neural Networks for Entity Disambiguation. In *Proceedings of IJCAI*.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of CoNLL*.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *TACL*, 5:397–411.