Distant Learning for Entity Linking with Automatic Noise Detection

Distant Learning for Entity Linking with Automatic Noise Detection

Phong Le[1] and **Ivan Titov**[1,2]
[1]University of Edinburgh [2]University of Amsterdam
lephong.xyz@gmail.com ititov@inf.ed.ac.uk

Abstract

Accurate entity linkers have been produced for domains and languages where annotated data (i.e., texts linked to a knowledge base) is available. However, little progress has been made for the settings where no or very limited amounts of labeled data are present (e.g., legal or most scientific domains). In this work, we show how we can learn to link mentions without having any labeled examples, only a knowledge base and a collection of unannotated texts from the corresponding domain. In order to achieve this, we frame the task as a multi-instance learning problem and rely on surface matching to create initial noisy labels. As the learning signal is weak and our surrogate labels are noisy, we introduce a noise detection component in our model: it lets the model detect and disregard examples which are likely to be noisy. Our method, jointly learning to detect noise and link entities, greatly outperforms the surface matching baseline. For a subset of entity categories, it even approaches the performance of supervised learning.

1 Introduction

Entity linking (EL) is the task of linking potentially ambiguous textual mentions to the corresponding entities in a knowledge base. Accurate entity linking is crucial in many natural language processing tasks, including information extraction (Hoffart et al., 2011) and question answering (Yih et al., 2015). Though there has been significant progress in entity linking recently (Ratinov et al., 2011; Hoffart et al., 2011; Chisholm and Hachey, 2015; Globerson et al., 2016; Yamada et al., 2017; Ganea and Hofmann, 2017; Le and Titov, 2018), previous work has focused on supervised learning. Annotated data necessary for supervised learning is available for certain knowledge bases and domains. For example, one can directly use webpages linking to Wikipedia to learn a Wikipedia linker. Similarly, there exist domain-specific sets of manually annotated documents (e.g., AIDA-CoNLL news dataset for YAGO (Hoffart et al., 2011)). However, for many ontologies and domains annotation is not available or limited (e.g., law). Our goal is to develop a method which does not rely on any training data besides unlabeled texts and a knowledge base.

In order to construct such a method, we use an insight from simple surface matching heuristics (e.g., Riedel et al. (2010)). Such heuristics choose entities from a knowledge base by measuring the overlap between the sets of content words in the mention and in the entity name. For example, in Figure 1, the entities BILL CLINTON (PRESIDENT) and PRESIDENCY OF BILL CLINTON both have two matching words with the mention *Bill Clinton*. Whereas we will see in our experiments that this method alone is not particularly accurate at selecting the best entity, the candidate lists it provides often include the correct entity. This implies that we can both focus on learning to select candidates from these lists and, less obviously, that we can leverage the lists as weak or distant supervision.

We frame this distance learning (DL) task as the multi-instance learning (MIL) problem (Dietterich et al., 1997). In MIL, each bag of examples is marked with a class label: the label indicates that the bag contains at least one example corresponding to that class. Relying on such labeled bags, MIL methods aim at learning classifiers for individual examples.

Our DL problem can be regarded as a binary version of MIL. For a list of entities (and importantly given the corresponding mention and its document context), we assume that we know if the list contains a correct entity or not. The 'positive lists' are essentially top candidates from the

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4081–4090
Florence, Italy, July 28 - August 2, 2019.

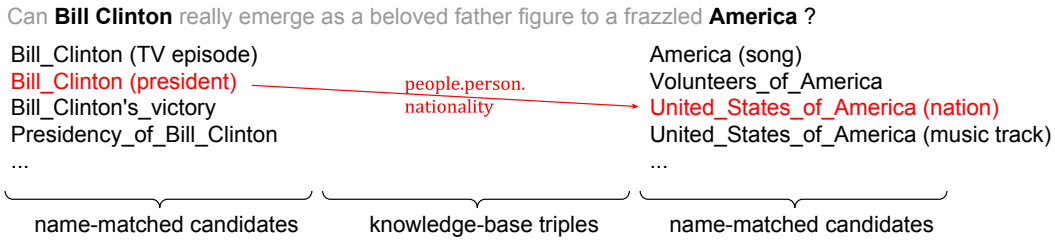

Figure 1: We annotate raw sentences using entity names and knowledge base triples. In training, we keep only red entities as positive candidates. In testing, we consider $|E^+| = 100$ name-matched candidates.

matching heuristic. For example, the four candidate entities for the mention 'Bill Clinton' in Figure 1 could be marked as a positive set. The 'negative lists' are randomly sampled sets of entities from the knowledge base. As with other MIL approaches, while relying on labeled lists, we learn to classify individual entities, i.e. to predict if an entity should be linked to the mention.

One important detail is that the classifier must not have access to information which and how many words match between the mention and the entity name. If it would know this, it would easily figure out which entity set is a candidate list and which one consists of randomly generated entities based solely on this information. Instead, by hiding it from the classifier, we force the classifier to extract features of the mention and its context predictive of the entity properties (e.g., an entity type), and hence ensure generalization.

Unfortunately, our supervision is noisy. The positive lists will often miss the correct entity for the given mention. This confuses the MIL model. In order to address this issue, we, jointly with the MIL model, learn a classifier which detects potentially problematic candidate lists. In other words, the classifier predicts how likely a given list is noisy (i.e., how much we should trust it). The probability is then used to weight the corresponding term in the objective function of the MIL model. By jointly training the MIL model and the noise detection classifier, we effectively let the MIL model choose which examples to use for training. As we will see in our experimental analysis, this joint learning method leads to a substantial improvement in performance. We also confirm that the noise detection model is generally able to identify and exclude wrong candidate lists by comparing its predictions to the gold standard.

DL is the mainstream approach to learning relation extractors (RE) (Mintz et al., 2009; Riedel et al., 2010), a problem related to entity linking. However, the two instantiations of the DL framework are very different. For RE, a bag of sentences is assigned to a categorical label (a relation). For EL, we assign a bag of entities, conditioned on the mention, to a positive class (correct) or a negative class (incorrect).

We evaluate our approach on the news domain for English as, having gold standard annotation (AIDA CoNLL), we can both assess performance and compute the upper bound, given by supervised learning. Nevertheless, we expect that our methodology is applicable to a wider range of knowledge bases, as long as unlabeled texts can be obtained for the corresponding domain. We plan to verify this claim in future work. In addition, we restrict ourselves to sentence-level modeling and, unlike state-of-the-art supervised methods, (Yamada et al., 2017; Ganea and Hofmann, 2017; Le and Titov, 2018) ignore interaction between linking decisions in the document. Again, it would be interesting to see if such global modeling would be beneficial in the distance learning setting.

Our contributions can be summarized as follows

- we show how the entity linking problem can be framed as a distance learning problem, namely as a binary MIL task;
- we construct a model for this task;
- we introduce a method for detecting noise in the automatic annotation;
- we demonstrate the effectiveness of our approach on a standard benchmark.

2 Entity linking as MIL

For each entity mention m with context c, we denote E^+ and E^- lists of positive candidates

and negative candidates: E^+ should have a high chance of containing the correct entity e, while E^- should include only incorrect entities. As standard in MIL, this will be the only supervision the model receives at training time. When using this supervision, the model will need to learn to decide which entity e in E^+ is most likely to correspond to the mention-context pair (m, c). At test time, the model with be provided with the list E^+ and will need to select an entity from this list.

Performing entity linking in two stages, candidate selection (generating candidate lists) and entity disambiguation (choosing an entity from the list), is standard in EL, with the first stage usually handled with heuristics and the second one approached with statistical modeling (Ratinov et al., 2011; Hoffart et al., 2011).

However, in our DL setting both stages change substantially. The candidate selection stage relies primarily on a surface matching heuristic, as described in Section 4. Whereas supervised learning for the disambiguation stage (e.g., Hoffart et al. (2011)) is replaced with MIL learning as described below in Section 3.[1]

To make the following sections clear, we introduce the following terms.

Definition 1. *A data point is a tuple* $\langle m, c, E^+, E^- \rangle$ *of mention* m*, context* c*, positive set* E^+*, and negative set* E^-*. In testing,* $E^- = \emptyset$*.*

Definition 2. *A data point* $\langle m, c, E^+, E^- \rangle$ *is* noisy *if* E^+ *does not contain the correct entity for mention* m*. If a data point is not noisy, we will refer to it as* valid*.*

3 Models

We introduce two approaches. The first one directly applies MIL, disregarding the fact that many data points are noisy. The second one addresses this shortcoming by integrating a noise detection component.

3.1 Model 1: MIL

Encoding context Context c is the entire l-word sentence $w_1, ..., w_l$ which also includes the mention $m = (w_h, ..., w_k), 1 \leq h \leq k \leq l$. We use a BiLSTM to encode sentences. The input to the BiLSTM is a concatenation $\mathbf{w}_i^* = [\mathbf{w}_i, \mathbf{p}_i]$ where $\mathbf{p}_i \in \mathbb{R}^{d_p}$ is position embedding and $\mathbf{w} \in \mathbb{R}^{d_w}$ is from GloVe[2] (Pennington et al., 2014). Forward $\mathbf{f}_i$ and backward $\mathbf{b}_i$ states of BiLSTM are fed into the classifier described below.

Entity embeddings In this work, we use a simple and scalable approach which involves computing entity embeddings on the fly using associated types. For instance, the TV episode BILL CLINTON is associated with several types including BASE.TYPE_ONTOLOGY.NON_AGENT and TV.TV_SERIES_EPISODE. Specifically, in order to produce an entity embedding, each type t is assigned a vector $\mathbf{t} \in \mathbb{R}^{d_t}$. We then compute a vector for entity e as

$$\mathbf{e} = \text{ReLU}(\mathbf{W}_e \frac{1}{|T_e|} \sum_{t \in T_e} \mathbf{t} + \mathbf{b}_e),$$

where T_e is the set of e's types, and $\mathbf{W}_e \in \mathbb{R}^{d_e \times d_t}$, $\mathbf{b} \in \mathbb{R}^{d_e}$ are a weight matrix and a bias vector.

More sophisticated approaches to producing entity embeddings (e.g., using relational graph convolutional networks (Schlichtkrull et al., 2018)) are likely to yield further improvements.

Scoring a candidate We use a one-hidden layer feed forward NN to compute score compatibility between a context-mention pair (m, c) and an entity e:

$$g(e, m, c) = \text{FFN}_g([\mathbf{e}, \mathbf{f}_{h-1}, \mathbf{b}_{h-1}, \mathbf{f}_k, \mathbf{b}_k])$$

If e^* is the correct entity, we want $g(e^*, m, c) > g(e, m, c)$ for any entity $e \neq e^*$.

Training Recall that for each mention-context pair (m, c), we have a positive set E^+ and a negative set E^-. We want to train the model to score at least one candidate in E^+ higher than any candidate in E^-. We use the max-margin loss to achieve this. Let

$$l(m, c) = [\max_{e \in E^-} g(e, m, c) + \delta - \max_{e \in E^+} g(e, m, c)]_+$$

$$L_1 = \sum_{(m,c) \in D} l(m, c)$$

where δ is a margin and $[x]_+ = x$ if $x > 0$ else 0; D is the training set. We want to minimize L_1 with respect to the model parameters. We rely on Adam optimizer and employ early stopping.

[1] Supervised learning is equivalent to assuming that E^+ are singletons containing only the gold-standard entity.

[2] http://nlp.stanford.edu/data/glove.840B.300d.zip

3.2 Model 2: MIL with Noise Detection (MIL-ND)

The model 1 ignores the fact that many data points are noisy, i.e. E^+ may not contain the correct entity. We address this by integrating a binary noise detection (ND) classifier which predicts if a data point is noisy. Intuitively, data points classified as noisy need to be discarded from training of the EL model. In practice, we weight them with the confidence of the ND classifier. As discussed below, we train the ND classifier jointly with the EL model.

Representation for E^+ The ND classifier needs to decide if there is at least one entity in the list E^+ corresponding to the mention-context pair (m, c). The question is now how to represent E^+ to make classification as easy as possible. One option is to use mean pooling, but this would result in uninformative representations, especially for longer candidate lists. Another option is max pooling, but it would not take into account which mention-context pair (m, c) is currently considered, so also unlikely to yield informative features of E^+. Instead we use attention, with the attention weight computed as a function of (m, c):

$$\mathbf{e}_{E^+} = \sum_{e \in E^+} \alpha_e \mathbf{e}$$

where α_e are attention weights

$$\alpha_e = \frac{\exp\{g'(e, m, c)/T\}}{\sum_{e' \in E^+} \exp\{g'(e', m, c)/T\}},$$

where g' is a score function. Instead of learning a separate attention function for the ND classifier, we reuse the one from the EL model, i.e. $g = g'$. This will reduce the number of parameters and make the method less prone to overfitting. Maybe more importantly, we expect that the better the entity disambiguation score function is, the better the ND classifier is, so tying the two together may provide an appropriate inductive bias. T is temperature, controlling how sharp α_e should be. We found that a small $T = 1/3$ stabilizes the learning.

Noise detection We use a binary classifier to detect noisy data points. The probability that a data point is noisy is defined as

$$p_N(1|m, c, E^+) = \sigma\left(\frac{\text{FFN}_f([\mathbf{e}_{E^+}, \mathbf{f}_{h-1}, \mathbf{b}_{h-1}, \mathbf{f}_k, \mathbf{b}_k])}{T}\right),$$

σ is the logistic sigmoid function. For simplicity, we use the same T as above.

Training Our goal is to down-weight potentially noisy data points. Our new loss is

$$L_2 = \sum_{(m,c) \in D} p_N(0|m, c, E^+) l(m, c) + \eta \times \text{KL}(\frac{\sum_{(m,c) \in D} p_N(\cdot|m, c, E^+)}{|D|} | p_N^*),$$

where p_N^* is a prior distribution indicating our beliefs about the proportion of noisy data points; η is a hyper-parameter. We optimize the objective with respect to the parameters of both ND and EL models. The second term is necessary, as without it the loss can be trivially minimized by the ND classifier predicting that all data points are noisy with the probability of 1. This would set the first term to exactly zero.

Intuitively, when using the second term, the model can disregard certain data points but disregarding too many of them incurs a penalty. Which data points are likely to be disregarded? Presumably the ones less consistent with the predictions of the EL model. In other words, joint training of EL and ND models encourages learning an entity-linking scoring function consistent with a large proportion of the data set but not necessarily with the entire data set. As we will see in the experimental section, the ND classifier indeed detects noisy data points rather than chooses some random subset of the data.[3]

We use the same optimization procedure as for the model 1. The second term is estimated at the mini-batch level.

Testing Differently from model 1, with model 2 we have two options on how to use it at test time:

- ignoring the ND classifier, thus doing entity disambiguation the same way as for model 1, or
- using the ND classifier as a mechanism to decide if the test data point should be classified as 'undecidable' or not. Specifically, if $p_N(1|m, c, E^+) > \tau$, model 2 will not output an entity for this data point. This should increase precision, as at test time E^+ also may not contain the correct entity.

[3]The second term is similar to that used in posterior regularization (Ganchev et al., 2010) and generalized expectation criteria method (Mann and McCallum, 2010).

Set	# sentences	# mentions
Train	170,000	389,989
Dev	2,275	4,603
Test	2,414	4,286

Table 1: The statistics of the proposed dataset.

We call the two versions MIL-ND and τMIL-ND, respectively.

4 Dataset

We describe how we create our dataset. We use Freebase[4], though our approach should be applicable to many other knowledge bases. Brief statistics of the dataset are shown in Table 1.

4.1 Training set

We took raw texts from the New York Times corpus, tagged them with the CoreNLP named entity recognizer[5] (Manning et al., 2014). We then selected only sentences that contain at least two entity mentions. We did this because on the one hand in most applications of EL we care about relations between entities (e.g., relation extraction), on the other hand, it provides us with an opportunity to prune the candidate list effectively, as discussed below. Note that we do it only for training.

For each mention m we carried out candidate selection as follows. First, we listed all entities which names contain all words of m. For instance, "America" (Figure 1) can be both the nation UNITED STATES OF AMERICA and Simon & Garfunkel's song AMERICA. We ranked these chosen entities by the entity ordering in the knowledge base (i.e., the one that appears first in the knowledge base would be ranked first); for Freebase this order is correlated with prominence.

Second, for each mention (e.g., "Bill Clinton"), we kept only entities which participate in a relation with one of the candidate entities for another mention in the sentence. For example, BILL CLINTON (PRESIDENT) is kept because it is in the PERSON.PERSON.NATIONALITY relation with the entity UNITED STATES OF AMERICA (NATION).

Last, to keep candidate lists manageable, we selected only $|E^+| = 100$ candidates from step 2 for each mention as positive candidates. During training, we sampled $|E^-| = 10$ candidates from the rest of the knowledge base as negative candidates.

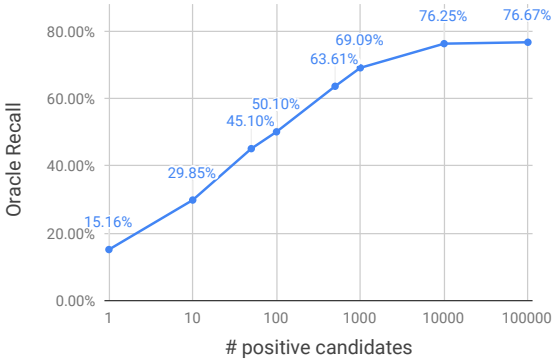

Figure 2: Oracle recall as a function of $|E^+|$ (the number of positive candidates) on the development set.

4.2 Development and test sets

We took manually annotated AIDA-A and AIDA-B as development and test sets (Hoffart et al., 2011). We turned the ground truth Wikipedia links in these sets to Freebase entities, thanks to the mapping available in Freebase.[6]

Candidate selection was done in the same way as for training, except for not filtering out sentences with only 1 entity (i.e. no step 2 from Section 4.1). The oracle recall for surface name matching (i.e. step 1 from Section 4.1) is 77%. It goes down to 50% if we restrict $|E^+| = 100$ (see Figure 2). We believe that there are straightforward ways to improve the selection heuristic (e.g., modifying the string matching heuristic or using word embeddings to match words in entity names and words in the mention) but we leave this for future work.

Note that because AIDA CoNLL dataset is based on Reuters newswire articles, these development and test sets do not overlap with the training set.

5 Experiments

We evaluated the models above using the data from Section 4. The source code and the data are available at https://github.com/lephong/dl4el

We ran each model five times and report mean and 95% confidence interval of three metrics:

[4] https://developers.google.com/freebase/. Freebase is chosen because it contains the largest set of entities among available knowledge bases (Färber and Rettinger, 2018).

[5] https://stanfordnlp.github.io/CoreNLP/

[6] We could not handle NIL cases here because the knowledge base used to annotate AIDA-CoNLL is different from the one we use.

(micro) precision, (micro) recall, and (micro) F1 (Cornolti et al., 2013) under two settings:

- 'All': all mentions are taken into account,
- 'In E^+': only mentions with E^+ containing the correct entity are considered.

The latter, though not realistic, is interesting as it lets us concentrate on the contribution of the disambiguation model, and ignore cases which are hopeless with the considered candidate selection method.

Note that, for system outputting exactly one entity for each mention (e.g., MIL model 1), precision and recall are equal.

5.1 Systems

We compared our models against 'Name matching'. It was proposed by Riedel et al. (2010) for RE: a mention is linked to an entity if it matches the entity's name. For tie cases, we chose the first matched entity appearing in Freebase. For instance, "America" is linked to the song instead of the nation. To our knowledge, name matching is the only method tried in previous work for our setting (i.e. with no annotated texts).

We also compared with a supervised version of model 1. We used the same method in Section 4.2 to convert AIDA CoNLL training set, with E^+ being singletons consisting of the correct entity provided by human annotators. This system can be considered as an upper-bound of our two models because: (i) it is trained in supervised rather than MIL setting with gold standard labels rather than weak supervision, and (ii) the training set is in the same domain (i.e. Reuter) with the test set. Although it uses only entity types but no other entity-related information for entity disambiguation, in Appendix B we show that this system performs on par with Hoffart et al. (2011) when evaluated in their setting.

Note that comparison with supervised linkers proposed in previous work is not possible as they require Wikipedia (see Section 6) for candidate selection, as a source of supervision, and often for learning entity embeddings.

We tuned hyper-parameters on the development set. Details are in Appendix A. Note that, in model 2 (both MIL-ND and τMIL-ND), we set the prior $p_N^*(1)$ to 0.9, i.e. requiring 90% of training data points should be ignored.[7] We experimented with $|E^+| = 100$ for both training and testing. For training, we set $|E^-| = 10$.

5.2 Results

Table 2 shows results on the test set. 'Name matching' is far behind the two models. Many entities in the knowledge base have similar or even identical names, so relying only on the surface form does not result in an effective method.[8]

MIL-ND achieves higher precision, recall, and F1 than MIL, this suggests that the ND classifier helped to eliminate bad data points during training. Using its confidence at test time (τMIL-ND, 'All' setting) was also beneficial in terms of precision and F1 (it cannot possibly increase recall). Because all the test data points are valid for the 'In E^+' setting, using the ND classifier had a slight negative effect on F1.

MIL-ND significantly outperforms MIL: the 95% confidence intervals for them do not overlap. However, this is not the case for MIL-ND and τMIL-ND. We therefore conclude that the ND classifier is clearly helpful for training and potentially for testing.

5.3 Analysis

Error types In Table 3 we classified errors according to named entity types thanks to the annotation from Tjong Kim Sang and De Meulder (2003). PER is the easiest type for all systems. Even name matching, without any learning, can correctly predict in half of the cases.

For LOC, it turns out that candidate selection is a bottleneck: when candidate selection was flawless, the models made only about 12% errors, down from about 57%. For MISC a similar conclusion can be drawn.

Can the ND classifier detect noise? From the training set, we collected 100 data points and manually checked if a data point is valid (i.e., E^+ contains the correct entity). We then checked how the accuracy changes depending on the threshold τ (Figure 3), the accuracy is defined as

$$\frac{\text{\# valid data points with } p_N < \tau}{\text{\# all data points with } p_N < \tau}$$

[7] Section 5.3 shows that 90% is too high, but it helps the model to rely only on those entity disambiguation decisions that are very certain.

[8] For instance, there are 36 entities named BILL CLINTON, and 248 entities having 'Bill Clinton' in their name.

	All			In E^+		
System	P	R	F1	P	R	F1
Name matching	15.03	15.03	15.03	29.13	29.13	29.13
MIL (model 1)	35.87	35.87	35.87 $\pm$0.72	69.38	69.38	69.38 $\pm$1.29
MIL-ND (model 2)	37.42	**37.42**	37.42 $\pm$0.35	72.50	**72.50**	**72.50** $\pm$0.68
τMIL-ND (model 2)	**38.91**	36.73	**37.78** $\pm$0.26	**73.19**	71.15	72.16 $\pm$0.48
Supervised learning	42.90	42.90	42.90 $\pm$0.59	83.12	83.12	83.12 $\pm$1.15

Table 2: Results on the test set under two settings. 95% confidence intervals of F1 scores are shown.

	All				In E^+			
System	LOC	ORG	PER	MISC	LOC	ORG	PER	MISC
Name matching	96.26	89.48	57.38	96.60	92.32	76.87	47.40	76.29
MIL	57.09	**76.30**	41.35	93.35	11.90	**47.90**	27.60	53.61
MIL-ND	57.15	77.15	35.95	92.47	12.02	49.77	20.94	47.42
τMIL-ND	**55.15**	76.56	**34.03**	**92.15**	**11.14**	51.18	**20.59**	**40.00**
Supervised learning	55.58	61.32	24.98	89.96	8.80	14.95	7.40	29.90

Table 3: % errors on the development set for different named entity types under two settings. (Smaller is better.)

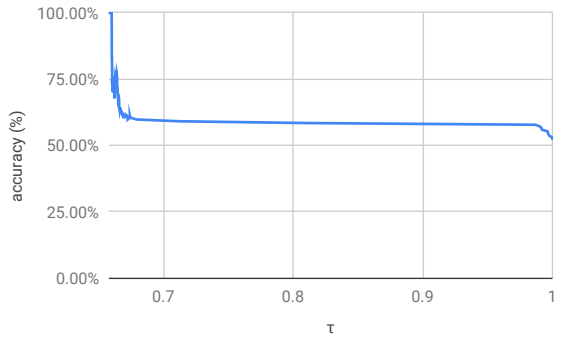

Figure 3: Accuracy vs τ. There are large plateaus between $\tau \in (0.7, 0.95)$ and $\tau < 0.6$ because the ND classifier hardly used these ranges. We hence set $\tau = 0.75$ in τMIL-ND.

Figure 4: F1 (top:'All'; bottom:'In E^+') on the development set with different numbers of positive candidates.

As expected, the smaller τ is, the higher the chance is that the chosen data point is valid (i.e., not noise). Hence, we can use the ND classifier to select high quality data points by adjusting τ.

For a further examination, from the training set, we collected all 47,213 data points (i.e. 27.8%) with $p_N(1|m, c, E^+) > \tau = 0.75$, and randomly chose 100 data points. We found that 89% are indeed noisy. This further confirms that the ND classifier is sufficiently accurate. Some examples are given in Table 4.

Number of positive candidates We also experimented with different values of $|E^+|$ (10, 50, 100) on the development set (Figure 4).

First, MIL-ND and τMIL-ND are always better than MIL. This is more apparent in the 'In E^+' settings: with this evaluation regime, we zoom in on cases where our models can predict correct entities (of course, all models equally fail for examples outside E^+).

Using the ND classifier at test time to decide to predict any entity or skip (τMIL-ND) is helpful in

Correctly detected as noise: * Small-market teams , like Milwaukee and San Diego , even big-market clubs , like Boston , Atlanta and the two [Chicago] franchises , trumpet them . Candidates: CHICAGO (music single), BOSTON TO CHICAGO (music track) * The politically powerful [Green] movement in Germany has led public opposition to genetic technology research and production . Candidates: THE GREEN PRINCE (movie), THE GREEN ARCHER (movie), GREEN GOLD (movie)
Incorrectly detected as noise: * Everything Forrest remains unaffected by , [Jenny] self-indulgently , self-destructively drowns in : radical politics , drug abuse , promiscuity . Candidates: JENNY CURRAN (book/film character)

Table 4: Examples of 100 randomly chosen sentences from the training set whose $p_N(1|m, c, E^+) > \tau = 0.75$. The first two examples are correctly detected as noise by our ND classifier. The last one is incorrectly detected.

the more realistic 'All' setting. The difference between τMIL-ND and MIL-ND is less pronounced for larger E^+. This is expected as the proportion of valid data points is higher, and hence the ND classifier is less necessary at test time. For 'in E^+' setting, τMIL-ND performs worse than MIL-ND, as we expected, because there are no noisy data points at test time.

What is wrong with the candidate selector? The above results show that candidate selection is a bottleneck and that the used selector is far from perfect. We found two cases where the selector is problematic: (i) the mention or the entity name is in an abbreviated form, such as 'U.N.' rather than 'United Nations', (ii) the mention and the entity's name only fuzzily match, such as '[English] county' and ENGLAND (country). We can overcome these problems via extending our surface matching as in Charton et al. (2014); Usbeck et al. (2014) or using word embeddings.

Even in some cases when the selector does not have any problems with surface matching, the number of candidates may be too large. For instance, consider '[Simpson] killed his wife...', there are more than 1,500 entities in the knowledge base containing the word 'Simpson'. It is unlikely that our entity disambiguation model can deal with such large lists. We may need a stronger mechanism for reducing the number of candidates. For example, we could use document-level information to discard highly unlikely entities.

6 Related work

High performance approaches to EL, such as Ratinov et al. (2011); Chisholm and Hachey (2015); Globerson et al. (2016); Yamada et al. (2017); Ganea and Hofmann (2017), are two-stage methods: candidate generation is followed by selecting an entity for the candidate lists. We follow the same paradigm but with some important differences discussed below.

Most approaches use alias-entity maps, i.e. weighted sets of (mention, entity) pairs created from anchors in Wikpedia. For example, one can count how many times phrase "the president" refers to BILL CLINTON to assign the weight to the corresponding pair. However, the method requires large annotated datasets, and it cannot deal with less prominent entities. As we do not have access to links, we use surface matching instead.

To choose an entity from a candidate list, two main disambiguation frameworks (Ratinov et al., 2011) are introduced: *local* which resolves mentions independently, and *global* which makes use of coherence modeling at the document level. Though we experimented with local models, the local-global distinction is largely orthogonal as we can directly integrate coherence modeling components in our DL approach.

Different types of supervision have been considered in previous work: full supervision (Yamada et al., 2017; Ganea and Hofmann, 2017; Le and Titov, 2018), using combinations of labeled and unlabeled data (Lazic et al., 2015), and even distant supervision (Fan et al., 2015). The approach of Fan et al. (2015) is heavily Wikipedia-based: they rely on a heuristic mapping from Freebase entities to Wikipedia entities, and learn features from Wikipedia articles. Unlike ours, their approach cannot be generalized to set-ups where no documents are available for entities.

7 Conclusions

We introduced the first approach to entity linking which neither uses annotated texts, nor assumes that entities are associated with textual documents (e.g., Wikipedia articles).

We learn the model using the MIL paradigm, and introduce a novel component, a noise detecting classifier, estimated jointly with the EL model. The classifier lets us disregard noisy labels, resulting in a more accurate entity linking model. Experimental results showed that our models substantially outperform the heuristic baseline, and, for certain categories, they approach the model estimated with supervised learning.

In future work we will aim to improve candidate selection (including different strategies to select candidate lists E^+, E^-). We will also use extra document information and jointly predict entities for different mentions in the document. Besides, we will consider additional knowledge bases (e.g. YAGO and Wikidata).

Acknowledgments

We would like to thank anonymous reviewers for their suggestions and comments. The project was supported by the European Research Council (ERC StG BroadSem 678254), the Dutch National Science Foundation (NWO VIDI 639.022.518), and an Amazon Web Services (AWS) grant.

References

Eric Charton, Marie-Jean Meurs, Ludovic Jean-Louis, and Michel Gagnon. 2014. Improving entity linking using surface form refinement. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*.

Andrew Chisholm and Ben Hachey. 2015. Entity disambiguation with web links. *Transactions of the Association for Computational Linguistics*, 3:145–156.

Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. 2013. A framework for benchmarking entity-annotation systems. In *Proceedings of the 22nd international conference on World Wide Web*, pages 249–260. ACM.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71.

Miao Fan, Qiang Zhou, and Thomas Fang Zheng. 2015. Distant supervision for entity linking. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 79–86.

Michael Färber and Achim Rettinger. 2018. Which knowledge graph is best for me? *CoRR*, abs/1809.11099.

Kuzman Ganchev, Jennifer Gillenwater, Ben Taskar, et al. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11(Jul):2001–2049.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringaard, and Fernando Pereira. 2016. Collective entity resolution with multi-focal attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 621–631, Berlin, Germany. Association for Computational Linguistics.

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.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604. Association for Computational Linguistics.

Gideon S Mann and Andrew McCallum. 2010. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of machine learning research*, 11(Feb):955–984.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference*

on Natural Language Processing of the AFNLP, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

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*, pages 142–147.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Michael Röder, Daniel Gerber, Sandro Athaide Coelho, Sören Auer, and Andreas Both. 2014. Agdistis-graph-based disambiguation of named entities using linked data. In *International semantic web conference*, pages 457–471. Springer.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *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 1321–1331, Beijing, China. Association for Computational Linguistics.

A Hyper-parameters

The used values for the hyper-parameters are shown in Table 5.

Hyper-parameters	Model	Value
learning rate (Adam)	1, 2	0.001
mini-batch size	1, 2	50
number of epochs	1, 2	20
d_w (word emb. dim.)	1, 2	300
d_p (position emb. dim.)	1, 2	5
d_t (type emb. dim.)	1, 2	50
d_e (entity emb. dim.)	1, 2	100
BiLSTM hidden dim.	1, 2	100
FFN_g hidden dim.	1, 2	300
FFN_h hidden dim.	2	300
δ (margin)	1, 2	0.1
T (temperature)	1, 2	$1/3$
η (KL coefficient)	2	5
$p_I^*(1)$ (prior)	2	0.9
τ (threshold)	2	0.75

Table 5: Values of hyper-parameters.

System	Micro accuracy
Ours	81.47 $\pm$1.27
Hoffart et al. (2011)	81.91

Table 6: Micro accuracy on AIDA CoNLL testb of our supervised system and (Hoffart et al., 2011).

B Supervised learning system

Our supervised learning system is a supervised version of model 1. To examine how good this system is, we tested it on the AIDA CoNLL dataset. Because this system uses entity types for entity disambiguation (and uses no other information related to entities), we made use of the map between Freebase entities and Wikipedia entities. We compared it with Hoffart et al. (2011). Note that we did not tune the system: it used the same values of hyper-parameters with model 1 (see Table5).

Because Wikipedia and YAGO are often used for candidate selection and/or for additional supervision, we here also used them for candidate selection and for computing $p(e|m)$ as a feature as in most existing systems (such as in Hoffart et al. (2011); Globerson et al. (2016); Ganea and Hofmann (2017)). For the candidate section, for each mention we kept maximally 20 candidates.

We ran our system five times and report mean and 95% confidence interval. Table 6 shows micro accuracy (in knowledge-base). Our system performs on par with Hoffart et al. (2011).