Lightweight Multilingual Entity Extraction and Linking

Aasish Pappu
Yahoo Research
New York, NY
aasishkp@yahoo-inc.com

Roi Blanco
University of A Coruña A
Coruña, Spain
rblanco@udc.es

Yashar Mehdad
AirBnB
San Francisco, CA
yashar.mehdad@airbnb.com

Amanda Stent
Bloomberg L.P.
New York, NY
astent@bloomberg.net

Kapil Thadani
Yahoo Research
New York, NY
thadani@yahoo-inc.com

ABSTRACT

Text analytics systems often rely heavily on detecting and linking entity mentions in documents to knowledge bases for downstream applications such as sentiment analysis, question answering and recommender systems. A major challenge for this task is to be able to accurately detect entities in new languages with limited labeled resources. In this paper we present an accurate and lightweight[1] multilingual named entity recognition (NER) and linking (NEL) system. The contributions of this paper are three-fold: 1) Lightweight named entity recognition with competitive accuracy; 2) Candidate entity retrieval that uses search clicklog data and entity embeddings to achieve high precision with a low memory footprint; and 3) efficient entity disambiguation. Our system achieves state-of-the-art performance on TAC KBP 2013 multilingual data and on English AIDA-CONLL data.

1. INTRODUCTION

Key tasks for text analytics systems include *named entity recognition* (NER) – the identification of *mentions*, or text spans that identify the who, what and where of document content; and *named entity linking* (NEL) – the identification of the entity in a knowledge base (KB) to which a particular mention may refer. Some systems perform NER and NEL jointly *e.g.*, [13, 32, 46]. However, most approaches are sequential and involve (some of) the following steps [26, 34, 43]: **(1)** mention detection; **(2)** mention normalization (*e.g.*, through acronym expansion [41]); **(3)** candidate entity retrieval for each mention; **(4)** entity disambiguation for mentions with multiple candidate entities; and **(5)** mention clustering for mentions that do not link to any entity. As this step involves inter-document entity clustering, for scalability, we do not perform any inter-document operations. However, we plan to address this task in future work.

In this paper we describe an accurate and lightweight NER/NEL system that performs mention detection (Section 2), candidate entity retrieval (Section 3) and entity disambiguation (Section 4). We demonstrate the accuracy of our system on the multilingual (English/Spanish/Chinese) TAC KBP 2013 data and on a standard monolingual data set, AIDA. We demonstrate that our system is lightweight in terms of speed and memory footprint. Specific contributions of this work include:

- with very few features that are easy to extend to multiple languages, we can achieve competitive performance on mention detection,
- with meta-linguistic context, specifically click data from search logs, we can provide competitive performance for multilingual candidate entity retrieval from documents, and
- through efficient methods for entity disambiguation, we can get further improvements in NEL accuracy

We make available the source code and entity embeddings that we use for candidate entity retrieval and disambiguation. `https://github.com/yahoo/FEL`.

2. MENTION DETECTION

Mention detection typically consists of running a NER system over input text. NER is often performed using a sequence tagging method such as conditional random fields [28], trained on human-labeled data and using lexical, syntactic and semantic features which may become quite complex and language specific [50]. With their joint NER/NEL semi-Conditional Random Field (CRF) system including Brown clusters, WordNet clusters and dictionaries, Luo et al. [32] report state-of-the-art F1 of 91.20 for NER for English on the standard CoNLL-2003 data set [50]. Similar performance has recently been achieved using only word embeddings and character features, but with less-efficient neural methods [39, 29, 33]. By contrast, we use simple CRFs and only a few features that we can readily extend to additional languages. The performance of our system is close to that of state-of-the-art single-language NER systems while being easily extensible to other languages and more computationally efficient.

[1]By *lightweight*, we mean easily extensible to additional languages, with a low memory footprint, and fast.

WSDM 2017, February 06-10, 2017, Cambridge, United Kingdom

DOI: http://dx.doi.org/10.1145/3018661.3018724

Feature	Description
Tokens	w_i for i in $\{-2, ..., +2\}$, $w_i \& w_{i+1}$ for i in $\{-1, 0\}$
Embeddings	$emb[100]$ for i in $\{-2, ..., +2\}$
Morphological	$morpho_i$ for i in $\{-2, ..., +2\}$
POS	pos_i for i in $\{-2, ..., +2\}$, $pos_i \& pos_{i+1}$ for i in $\{-2, .., 1\}$

Table 1: NER features.

2.1 System Description

As in previous work, we treat NER as a sequence labeling problem. To train, we use CRFsuite [37] with L-BFGS [38]. Following Ratinov and Roth [42] and based on our own experiments, we use a BILOU label encoding scheme.

- B - 'beginning'
- I - 'inside'
- L - 'last'
- O - 'outside'
- U - 'unique'

The features we use are listed in Table 1. To ensure that NER is lightweight, we focus on efficient-to-compute features that will scale easily to multiple languages. Our main features are tokens and token embeddings, within a small window of the target token. We learn token embeddings for each language from Wikipedia; we use word2vec [36] with the Continuous-Bag-of-Words (CBOW) algorithm, 5 iterations, and a window size of 5. We do not tune on any development set. We also use morphological features, which consist of word shape and capitalization features, token prefixes and suffixes (up to length 4), numbers and punctuation. Finally, we experiment with language-specific part-of-speech (POS) tags; POS tagging adds minimal preprocessing and is available for over 40 languages.

Features	EN			ES			ZH		
	P	R	F1	P	R	F1	P	R	F1
Token + Embeddings	91	82	86	86	79	82	76	54	64
+ POS	90	87	88	86	80	83	77	54	65
+ Morphological	90	88	89	85	84	85	74	60	67
+ POS + Morphological	89	88	89	85	84	84	75	61	67

Table 2: Precision, recall and F_1 of NER for CoNLL-2003 English, CoNLL-2002 Spanish and Ontonotes Chinese test sets.

We use standard evaluation data: CoNLL 2003 for English (EN) [50], CoNLL 2002 for Spanish (ES) [51] and OntoNotes 4.0 (LDC2011T03) for Chinese (ZH). In each case we use the training data for training CRFs and test data (testb for CoNLL datasets) for evaluation.

Evaluation results for our system are shown in Table 2. For all three languages, the best performing models include token, embedding and morphological features. As shown in Table 3 our best model beats the state-of-the-art multilingual systems from the literature [1, 17]. The performance of our best model is close to that for state-of-the-art heavily-tuned single-language systems for English, Spanish and Chinese without using gazetteers or chunkers/parsers. Surprisingly, for English and Spanish POS features add nothing to overall performance when morphological features are included. In future work, we will investigate the use of character-based features and word cluster features.

Systems	EN	ES	ZH
This Work	88.6	84.6	67.2
Al-Rfou et al. [1]†	71.3	63.0	-
Stanford [17]*	86.3	81.1	64.1/69.5
Suzuki and Isozaki [48]	89.9	-	-
Che et al. [7]*	-	-	64.1/69.5
Lample et al. [29]$^+$	90.9	85.8	-
Ma and Hovy [33]$^+$	91.2	-	-
Luo et al. [32]*	91.2	-	-

Table 3: F1 for NER for CoNLL-2003 English, CoNLL-2002 Spanish and Ontonotes Chinese test sets. † indicates multilingual systems. * indicates systems using features hard to scale to multiple languages such as gazetteers, syntactic and semantic features. + indicates systems using neural methods.

3. CANDIDATE ENTITY RETRIEVAL

Candidate entry retrieval consists of identifying zero or more entities in an input knowledge base to which a mention may refer. We assume a KB has a *canonical form* (CF) for each entity. In our experiments we use the wikipedia page title corresponding to each KB entity as the canonical form. In typical NEL systems, candidate entities are retrieved if their canonical form or an alias is similar to the (expanded) mention text. For example, if a KB has wiki IDs then aliases may include substrings of the wiki ID, Wikipedia redirects and inlinks, especially from Wikipedia disambiguation pages [10, 15, 25, 31, 32]. Aliases may also include references to the wiki ID from non-Wikipedia pages [31], from search click logs [4, 41, 53], or from the output of coreference [10, 31].

3.1 Entity Embeddings

We developed an entity embedding approach akin to [30, 12] and used these embeddings for candidate entity retrieval and for entity disambiguation. Our training documents are preprocessed Wikipedia articles, where hyperlinks in the articles have been transformed to the CFs for their associated entities, and the article title is the CF for the target entity. We represent each article a as: (1) a sequence of the entities mentioned in it, $(ent_1, ent_2, \ldots, ent_n)$, where $ent_i \in Ent$, the set of all entities; and (2) sequence of the tokens it contains, $(w_1, w_2 \ldots, w_m)$, where $w_j \in W$, the set of all tokens. We aim to simultaneously learn D-dimensional representations of Ent and W in a common vector space. The context of an entity and the context of a token are modeled using the architecture in Figure 1, where entity vectors act not only as units to predict their surrounding entities, but also as the global context of word sequences contained within them. In this way, one layer models entity context, and the other layer models token context. We connect these two layers using the same technique that Quoc and Mikolov [30] used to train paragraph vectors. We use continuous skip-grams with 300 dimensions and a window size of 10 and we set negative sampling to 5 to train our entity embedding model. In order to improve our vectors while limiting the number of iterations to 5, we also use hierarchical sampling. For qualitative study of entity embeddings, we provided examples of nearest-neighbors in the vector space in Figure 2.

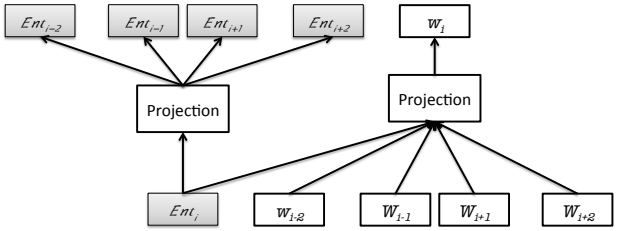

Figure 1: Architecture for training word and entity embeddings simultaneously. *Ent* represents entities, *W* represents their context words.

We use these entity embeddings for our candidate entity retrieval system, which is based on the Fast Entity Linker (FEL) from [4], and which we briefly present below. Although FEL is an efficient and precise candidate entity retriever [4], the entity embeddings alone contain considerable information. So we add as a baseline a k-nearest neighbors (KNN) method which takes a mention as input, retrieves the entity embedding for the mention, then performs a nearest neighbor search amongst the entity embeddings for the CFs of all KB entities using Euclidean distance. With both methods, we aim for precision, so that we can pass a concise set of candidate entities to disambiguation for greater efficiency.

3.2 Fast Entity Linking

Fast Entity Linker (FEL) is an unsupervised approach which selects the segmentation of an input sequence of words that maximizes the likelihood of all substrings linking to an entity in the Knowledge Base. The model requires to calculate the conditional probabilities of an entity given every substring of the input sequence, but avoids computing entity to entity joint dependencies, which makes the process very efficient. As a byproduct of this segmentation, the model selects the most likely entities that would be linked to each substring in a sequence of words.

To approximate entity likelihoods FEL makes use of anchor text in, and user queries leading to a click on, a Web page representing the entity, such as the entity's Wikipedia page. FEL imposes contextual dependencies by calculating the cosine distance between a candidate entity's embedding and the entity embeddings from the substrings of the input string, and including those as part of the final probability estimations. With this paper, we will release our implementation of FEL and the data packs (models) that it uses.

We built per-language data packs for FEL using query logs that spanned 12 months and Wikipedia anchor text extracted from Wikipedia dumps dated November 2015. We obtained access to anonymized search engine data consisting of queries for which a searcher clicks on a Wikipedia page, *e.g.*, *Barack* and *President Obama* map to *wiki/Barack_Obama*. Candidate entity retrieval precision for FEL is 73.6 for EN, 54.7 for ES and 81.7 for ZH.

To ensure that candidate entity retrieval is lightweight, all the strings embedded into the data packs are hashed using a minimal perfect hash function, which returns the identifier of a string in constant time and guarantees a zero collision rate between strings of the key set. However, to make the collision probability negligible a constant-sized signature is associated to each string, which is used to check whether the string being looked up was present in the key set. In order to hold counts for aliases and entities we use Elias-Fano integer coding techniques. We compress embedding vectors in a similar fashion: after quantizing their values the integers obtained are then encoded with Golomb codes. The encodings of each vector are concatenated into a single bit stream, and their starting positions are stored in an Elias-Fano monotone sequence [14].

4. ENTITY DISAMBIGUATION

Entity disambiguation is the task of figuring out to which candidate entity a mention refers. The task is complex because mentions may refer to different entities, depending on local context (*e.g. Jason Williams*^{a} the basketball player *plays* or *wins*, while *Jason Williams*^{b} the actor *directs* or *writes*), document context (*Jason Williams*^{a} co-occurs with entities like *NBA* and *Memphis Grizzlies*, while *Jason Williams*^{b} co-occurs with entities like *The Westside Theatre*), and world knowledge (*Jason Williams*^{a} gets more sports media coverage).

An intuitive representation for entity disambiguation is a graph with (weighted) edges linking mentions to mentions, mentions to entities, and entities to entities; the goal is to find dense subgraphs [25]. However, this is NP-hard. Researchers have attempted various approximate methods, including supervised ranking [5, 9, 41, 54], neural networks [24], and global ranking methods such as approximate dense subgraph computation [25], variations on PageRank [23, 21], clique partitioning [2], random walks [20], and loopy belief propagation [19]. In this work, we compare three general-purpose and efficient methods: the forward-backward algorithm, exemplar clustering, and label propagation. We describe these approaches below.

Features used for disambiguation include co-occurrence of tokens or entities between the Wikipedia entry for an entity and the input document, as well as Wikipedia link structure, entity category and mention type [4, 5, 10, 9, 16, 25, 31, 32, 41, 53]. They may also include "entity popularity": a prior on the likelihood of observing an entity, estimated *e.g.* by frequency of occurrence in Wikipedia, in the input documents, etc. [9, 16, 25, 31, 32, 41, 53]. Ceccarelli *et al.* and Fahrni *et al.* [6] have especially good descriptions of entity-entity and entity-mention relatedness features. Zhou et al. [53], Blanco et al. [4] and Dilek et al. [22] all use search click logs to approximate entity popularity; Shen et al. [44] uses user mentions of an entity on Twitter; and Chisholm and Hachey [9] use in-links to Wikipedia pages from the open web. In this work, of course, some of the features are captured by entity embeddings, while others are used in the FEL candidate entity retriever. In fact, a strength of our approach is that disambiguation features are used in candidate entity retrieval, increasing the precision at that stage and allowing for a shorter n-best list to passed through to disambiguation than in prior work(*e.g.,* [31, 41, 54]).

4.1 Forward Backward Algorithm

The forward-backward algorithm [3] (FwBw), listed in Algorithm 1, is quadratic in $O(|M| \times n)$ where $|M|$ is the number of mentions and n the number of candidate entities per mention. The input is $M = (m_1, m_2, ..., m_T)$, a list of mentions in the document, and $NB = ({e_1}^n, {e_2}^n, ..., {e_k}^n)$, a list of sets of candidate entities retrieved for each mention. The algorithm runs subroutine Forward twice – first on the list of mentions, then on the reversed list of mentions

(the "Backward" step). These steps return $fwd_{|M|\times n}$ and $bkwd_{|M|\times n}$ matrices respectively. The output contains cumulative likelihood values for each candidate entity for each mention. We then compute posterior marginals for each candidate entity and return the best candidate entity for each mention. Procedure FORWARD is similar to the popular Viterbi sequence decoding algorithm, except that it does not keep track of the best path. Another difference from the standard forward-backward algorithm is in line 18 of Algorithm 1; procedure JOINT_SIM computes lexical and vector (cosine) similarities between the mention text and the candidate entity's CF.

Algorithm 1 ForwardBackward

1: **Input**: $M \leftarrow$ mentions, $NB \leftarrow$ N-BestLinks,
2: $P \leftarrow$ Posterior probability from NB
3: **Output**: $\hat{L} \leftarrow$ 1-best Entities
4: **procedure** FWBW
5: $fwd \leftarrow$ FORWARD(NB, M)
6: $bkwd \leftarrow$ FORWARD(NB_{rev}, M_{rev})
7: **for** $i \leftarrow 1, 3, ..., |M|$ **do**
$\hat{L}_i \leftarrow \arg\max_k(fwd_{i,k} \cdot bkwd_{|M|-i,k})$
8: **end for**
return $\hat{L}_{1,2..,i,..|M|}$
9: **end procedure**
10: **procedure** JOINT_SIM(u,v)
11: $sem \leftarrow$ `semSim`(u, v), $lex \leftarrow$ `textSim`(u, v)
return $(\lambda \cdot sem + (1-\lambda) \cdot lex)$
12: **end procedure**
13: **procedure** FORWARD
14: **for** l_i in NB_1 **do**
$S_{i,1} \leftarrow$ JOINT_SIM(l_i, M_1)
$\theta_{0,i} = P(l_i, M_1) \cdot S_{l_i,M_1}$
15: **end for**
16: **for** $i \leftarrow 2, 3, ..., |M|$ **do**
17: **for each** `link` l_j **do**
18: $S_{M_i,l_j} \leftarrow$ JOINT_SIM(M_i, l_j)
$\theta_{j,i} \leftarrow \max_k(\theta_{k,i-1} \cdot S_{M_i,l_j} \cdot S_{l_k,l_j} \cdot P(M_i, l_k))$
19: **end for**
20: **end for**
return θ
21: **end procedure**

4.2 Exemplar Clustering

When an entity is mentioned several times in a document, later mentions are often abbreviated or shortened, *e.g.*, *Bill Clinton* may become *Clinton*. Although later mentions contain less lexical information for disambiguation, the document context may be used to disambiguate entity mentions. Specifically, we cluster the entity embeddings (see Section 3.2) of mentions and candidate entities' CFs. We use exemplar clustering [18], which lets us choose certain candidate entities as potential cluster centroids and assign mentions to these clusters. This choice can be initialized using a preference vector with higher (and positive) values set for candidate entities' entity embeddings and zeros for mentions' entity embeddings.

In this work we specifically use the affinity propagation flavor of exemplar clustering as implemented in scikitlearn [40]. The clustering algorithm is listed in Algorithm 2. As before, the inputs are M and NB. We construct a preference vector $pref$ of size $n = |M| + |NB|$. We initialize this preference vector with the posterior probability of each candidate entity for a given mention and 0 for all mention vectors. When candidate entities are in the n-best list for multiple mentions, we pick the highest posterior value associated with that entity. $X_{n\times d}$ is a matrix of d dimensional entity embeddings for each mention and for each candidate entity's CF (lines 2–3). Exemplar clustering is primarily a message-passing algorithm that allows datapoints to communicate their candidature for becoming an exemplar. To facilitate this communication, we use two matrices, an availability matrix (A) and a response matrix (R). A carries messages sent from exemplars to potential cluster members to show the appropriateness of all potential exemplar points. R carries messages from cluster members to a candidate exemplar to show their cluster membership potential given the candidate being an exemplar. These matrices are initialized with zeros. The message passing process also involves measuring similarity between any two datapoints. To this end, we construct a similarity matrix $S_{n\times n}$ with pairwise similarities for all the rows in X as shown in line 5 of Algorithm 2. On line 6, we incorporate our exemplar preferences by adding the preference vector to the diagonal of S. The iterative part of the algorithm begins at line 8 and runs until we reach convergence *i.e.*, matrices R and A do not change, OR we hit $T \geq max\_iterations$. Convergence also depends on *damping factor*[2] which is often used to discourage severe oscillations while updating R and A. In lines 9–11, the algorithm updates the response and availability matrices R and A. Once the message matrices have converged, we select exemplars as shown in line 13. At this point all the datapoints in X have been assigned to clusters and the exemplars CI represent cluster centroids. We iterate over mentions and assign their cluster centroid (exemplar) as their entity link, as shown in line 14. Drawbacks of exemplar clustering are its runtime complexity $O(|M|^2T)$ and that it may not converge if the preference values or the damping factor are too low.

Algorithm 2 Exemplar Clustering

Input: M, NB, $pref_{1\times n} \leftarrow$ Posterior probability from N-BestLinks
2: **Output**: $\hat{L} \leftarrow$ 1-best Entities
$X_{n\times d} \leftarrow embeddings(M) \oplus embeddings(NB)$
4: $S_{n\times n} \leftarrow pairwiseSim(X)$
$R_{n\times n}, A_{n\times n} \leftarrow zeros, zeros$
6: $diag(S) \leftarrow diag(S) + pref$
λ is *damping factor* to discourage oscillations
8: **while** convergence `OR` $T \leq max\_iterations$ **do**
$R_{i,k} \leftarrow S_{i,k} - \max_{k'\neq k}\{A_{i,k'} + S_{i,k'})\}$
10: $A_{i,k} \leftarrow \min\left(0, A_{k,k} + \sum_{i'\notin\{i,k\}} max(0, R_{i',k})\right)$
$A_{k,k} \leftarrow \sum_{i'\neq k} max(0, R_{i',k})$
12: **end while**
$I \leftarrow R_{i,i} + A_{i,i} > 0$
14: $CI = \arg\max_{k\in I} S_{k,k}$
return $\hat{L} \leftarrow \left(\forall_{k\in|CI|} CI_k\right)$

4.3 Label Propagation

Label propagation is an umbrella term for a family of graph-based semi-supervised algorithms. A label propaga-

[2]The details on *damping factor* can be found in [18]. It is applied to lines 9–11 in Algorithm 2.

tion algorithm propagates labels to unlabeled nodes in a graph, starting with a few labeled nodes. These algorithms have been successful in several text processing tasks such as sentiment analysis [47], gloss-finding [11], and word sense disambiguation [52]. In this work, we apply a variant of label propagation proposed by [49] called modified adsorption (MAD). This is a transductive learning algorithm that operates under noisy-label assumption and aims to relabel labeled examples for coherency across the graph. First we construct an undirected graph G whose nodes V correspond to all mentions M in a document. The nodes are connected by weighted edges E_w where the weight is the value returned by the JOINT_SIM procedure from Algorithm 1 for the mention texts. After we construct the adjacency matrix for this graph $G \leftarrow (V, E_w)$, we inject seed labels L on a few nodes. In our case, for nodes (mentions) V' with entity candidates of high posterior values based on a threshold tuned on a development set, we assign a label distribution $\{l_1 : p_1, l_2 : p_2, \ldots, l_n : p_n\}$. Along with $\{L, G\}$, MAD takes three hyperparameters $\{\mu_1, \mu_2, \mu_3\}$ as input, which control the behavior of a random walk on the graph. These hyperparameters correspond to `inject`, `continue`, and `abandon` actions in the random walk. Once the random walk begins, with probability p^{inj} it may stop and return the seed label distribution L. Alternatively, abandon the labeling and return all-zeros vector with probability p^{abnd}. Or it would continue the random walk from the current node to one of its neighbors with probability p^{cont}. For every node these probabilities sum to unity. The transition probability between a pair of nodes is directly proportional to normalized edge weight between the nodes. On convergence, MAD generates a ranked list of labels for each node in V without modifying the labels of nodes in V'. We pick the highest ranked label for each node in V as the final candidate. The algorithm complexity is $O(|V|T)$ where T is number of iterations and $|V|$ number of nodes in the graph. This makes the algorithm highly scalable for our task.

5. EVALUATION

There are only a few publicly available data sets for NER/NEL, and they cover only a few languages. We evaluated on data from the cross-lingual TAC KBP 2013 shared task. Because of issues with this data set, including a small KB with many missing entities and considerable variation in settings in prior work [31, 15], we also evaluated using the monolingual AIDA-CONLL 2003 dataset. For the TAC KBP data, the reference KB contains a subset of English Wikipedia entities mapped to unique KBP identifiers. The reference KB contains mappings from Wikipedia English entities to the TAC KBP reference ID. We further bootstrapped this KB to include Spanish (4467) and Chinese (3060) Wikipedia entities to improve coverage for entities used in the evaluation corpus for each language. Since the AIDA annotations use wiki IDs, no KB-mapping was required. Statistics about the datasets used in this work are given in Table 4.

For the TAC KBP data, we ran mention detection (Section 2); the AIDA data comes with gold mentions, on which results are reported in previous work. Before running mention detection on the TAC KBP data, we removed HTML tags from the source documents and sentence split and tokenized using an in-house preprocessor.

Data	Docs	Entities	Unique entities	Mentions
KBP-EN	1820	1183	349	150144
KBP-ES	1175	1305	583	6321
KBP-ZH	1224	1229	159	15092
AIDA-all	1392	37922	5598	50758

Table 4: Statistics for our evaluation datasets.

5.1 Evaluation Setup

We ran all our experiments on a Redhat 6.4 machine with 24GB memory and a 4-core Xeon CPU. In our experiments, we tuned all model hyper-parameters on the 2012 TAC KBP English dataset. Both of our candidate generation methods generate n-best ($N = 10$) candidates for every entity mention. We specify the following hyper-parameters for the disambiguation algorithms:

- In **FwBw**, λ is set to 0.5.
- In **Exemplar**, `max_iterations` = 300, *damping factor* (λ) = 0.5 and the algorithm stops if the message passing matrices do not change for 50 iterations.
- For **LabelProp**, $\mu_1 = 1$, $\mu_2 = 1e-2$, $\mu_3 = 1e-2$, $\beta = 2$ and the maximum number of iterations is 100. These are default experimental values provided in the implementation of [49][3] that we validated on our development set.

5.2 Evaluation Results and Analysis

5.2.1 TAC KBP Evaluation Results

We applied our three disambiguation algorithms (Section 4) on the output of FEL, comparing with our KNN baseline (Section 3). For the TAC KBP data we use the official scorer[4] and report the `strong_link_match` measure to compare our system with previous work, disregarding NIL links to entities which are not in the official KB. In 2013 only two systems were officially evaluated on all three languages (EN, ES, ZH): HITS [15] and BasisTech [35]. However, HITS uses separate Wikipedia dumps as a KB, and does not separate out NIL clustering from non-NIL entity linking, so we compare our system against BasisTech. We note that BasisTech performed inter-document entity clustering, whereas for scalability, we do not perform any inter-document operations.

In Table 5, we show F_1 scores for the combinations of candidate entity retrieval and entity disambiguation methods. In addition to showing results with a 10-best list of candidate entities, we show results for 1-best candidate entity retrieval. FEL candidate entity retrieval outperforms KNN across all languages and disambiguation settings. On the English portion of the TAC KBP data, FEL combined with FwBw disambiguation achieves the best result (61.0) among all our disambiguation methods and outperforms BasisTech (56.5). On the Chinese cross-lingual NER/NEL task, FEL 1-best outperforms BasisTech, while BasisTech maintains an advantage over our systems on the Spanish cross-lingual task.

[3]https://github.com/parthatalukdar/junto
[4]https://github.com/wikilinks/neleval

Dataset	1-best		FwBw		Exemplar		LabelProp		BasisTech
	KNN	FEL	KNN	FEL	KNN	FEL	KNN	FEL	
KBP-EN	32.0	50.6	29.1	**61.0**	52.6	52.8	29.8	53.6	56.5
KBP-ES	31.3	50.8	27.7	46.7	24.0	50.5	28.5	48.3	**61.2**
KBP-ZH	17.0	**67.3**	7.5	54.7	9.8	57.5	12.3	49.8	62.1

Table 5: `strong_link_match` F_1% of our methods and basistech on monolingual entity linking (English) and cross-lingual linking (Spanish and Chinese) on the TAC KBP 2013 test partition.

5.2.2 Analysis

Table 6 shows the precision, recall and F_1 scores for the three TAC KBP 2013 datasets and their constituent subgroups. For the Spanish and Chinese tasks, disambiguation via exemplar clustering typically yields the highest `strong_link_match` precision among our proposed systems. However, the 1-best FEL result achieves higher recall, and thereby F_1, for non-NIL entity links. This trend of high precision for disambiguation approaches being offset by higher recall for the 1-best system remains consistent across all entity types and document genres in the Spanish and Chinese data. We hypothesize that the performance of our systems on these cross-lingual tasks is limited by a smaller number of in-language entities in our training data when compared to English (*e.g.*, 1.1M Spanish entities vs. 4.9M English entities). The state-of-the-art system on this dataset may use a larger in-house database of entities. Our NEL performance could thus likely be improved by increasing the coverage of entities in our non-English data packs.

On monolingual linking in English, where entity coverage is more comprehensive, FwBw disambiguation achieves a balance of strong precision and recall to yield state-of-the-art performance. We attribute this result in part to the Markov independence assumptions implicit in the forward-backward algorithm. Although conditioning on global context is generally accepted to be valuable in entity linking, techniques like Exemplar or LabelProp can also be sensitive to large numbers of noisy mentions in the document, which are likely to occur in the extremely large discussion forum and newsgroup documents present in KBP-EN. Interestingly, linking precision is observed to be weakest overall on the newsgroups subset rather than the longer discussion forums, likely owing to its emphasis on sports conversations in which mentions of athletes are often mistakenly linked to other personalities (*e.g.*, "Becks" [David Beckham] linked to the musician Beck) and team names to locations (*e.g.*, "Madrid" [Real Madrid C.F.] linked to the city of Madrid).

We show in Table 7 the variation in document statistics for documents with correctly-linked entities with respect to those over the full dataset.[5] This illustrates that linking in English is more accurate on documents with fewer words and fewer entity mentions, likely reflecting the poor precision over the newsgroups portion. Although the number of words or mentions in the document do not appear to influence linking consistently, we do observe a clear association between *mention density* (measured as #mentions per word) and accurate linking for all languages and systems. Furthermore, we also note that LabelProp consistently produces more accurate links on shorter documents across all languages, leading us to conjecture that similar graph-based disambiguation approaches may be preferable when little document context is available (*e.g.*, for tweets).

[5]Note that these measures are not dominated by entity-dense documents as most entities (94% for EN and ES, 100% for ZH) come from documents with only 1-2 labeled entities.

Measure	Lang	1-best	FwBw	Exemplar	LabelProp
#words	EN	-5.2	-4.9	-5.9	-41.3
	ES	6.0	-4.8	2.5	-3.9
	ZH	3.8	-2.9	6.5	-19.8
#mentions	EN	-1.2	-2.7	-1.6	-44.8
	ES	16.0	12.3	12.7	1.1
	ZH	-1.3	-4.3	8.9	-22.2
#mentions per word	EN	16.5	15.9	15.1	37.8
	ES	13.1	22.6	18.5	15.7
	ZH	6.0	12.1	16.2	4.8

Table 7: Percentage variation in document statistics for correctly-linked entities with respect to all entities over TAC KBP 2013 data. Negative values imply that linking accuracy is negatively correlated with the measure and vice versa.

Dataset	Precision		Recall		F_1	
	KNN	FEL	KNN	FEL	KNN	FEL
KBP-EN	**55.4**	45.3	50.1	**63.1**	52.6	**52.8**
├ News	**53.6**	44.9	49.2	**66.8**	51.3	**53.7**
├ Forums	**65.5**	54.9	53.5	**60.4**	**58.7**	57.5
└ Newsgroups	**34.3**	26.7	41.5	**55.9**	**37.5**	36.2

Table 8: Precision, recall and F_1 percentages for systems using Exemplar clustering disambiguation over the English TAC KBP 2013 data and its subgroups.

Finally, turning to candidate entity retrieval, FEL yields dramatically stronger candidates than KNN across all languages and disambiguation strategies. The one exception to this is KNN + Exemplar on the English task, which remains competitive with FEL + Exemplar. Table 8 compares precision, recall and F_1 for the two candidate generation systems in this scenario. Intriguingly, this result appears to be driven in part by the presence of *longer* documents (*i.e.*, the newsgroups and discussion forums mentioned previously) on which the precision advantage of KNN outweighs the stronger recall of FEL. We conjecture that a large amount of training data for these mention vectors leads to a reliable clustering of mentions around their entity vectors. In such a scenario, Exemplar disambiguation can effectively exploit document context to accurately link entities even without search logs and click data for the target language.

Dataset	#docs	#words per doc	Precision				Recall				F_1			
			1-best	FwBw	Exemplar	LabelProp	1-best	FwBw	Exemplar	LabelProp	1-best	FwBw	Exemplar	LabelProp
KBP-EN	1820	3118.1	42.5	**62.1**	45.3	59.9	62.6	59.9	**63.1**	48.5	50.6	**61.0**	52.8	53.6
├ News	924	300.6	42.8	**64.4**	44.9	62.6	66.3	61.7	**66.8**	61.7	52.0	**63.0**	53.7	62.1
├ Forums	607	7555.9	50.1	**64.3**	54.9	61.3	**60.4**	58.5	**60.4**	31.9	54.8	**61.3**	57.5	41.9
└ Newsgroups	288	2813.8	24.6	**46.5**	26.7	45.4	53.4	**56.8**	55.9	50.0	33.7	**51.2**	36.2	47.6
KBP-ES	1175	168.7	60.5	67.4	**71.0**	62.5	**43.8**	35.7	39.2	39.4	**50.8**	46.7	50.5	48.3
├ Spanish news	775	160.5	58.1	**65.3**	64.5	60.3	**37.9**	30.6	26.9	32.4	**45.9**	41.7	37.9	42.2
└ English news	397	180.7	64.3	70.8	**74.8**	65.5	**56.3**	46.3	42.8	53.9	**60.0**	56.0	54.4	59.1
KBP-ZH	1224	752.5	74.3	75.5	**77.1**	61.8	**61.5**	42.9	45.8	41.7	**67.3**	54.7	57.5	49.8
├ Newsgroups	415	1215.9	78.6	74.4	**81.0**	59.1	**66.6**	43.7	48.9	38.9	**72.1**	55.0	61.0	46.9
├ Chinese news	406	323.0	70.5	76.9	**82.6**	55.2	**51.7**	40.2	48.0	33.7	59.7	52.8	**60.8**	41.9
├ English news	230	217.9	71.4	71.4	52.7	**71.7**	**69.6**	43.5	30.0	60.4	**70.5**	54.1	38.2	65.6
└ Blogs	173	1360.0	75.9	81.0	**85.5**	65.8	**61.5**	46.6	54.0	42.0	**67.9**	59.1	66.2	51.2

Table 6: Precision, recall and F_1 percentages for FEL systems over the three TAC KBP 2013 datasets and distinct subsets grouped by document genre. Small subgroups with fewer than 4 documents are omitted.

System	A_{macro}	A_{micro}
1-best	83.48	81.07
FwBw	**83.63**	80.98
Exemplar	83.50	81.08
Alhelbawy and Gaizauskas [2]	82.80	**86.10**
Cucerzan [10]	43.74	51.03
Kulkarni et al. [27]	76.74	72.87
Hoffart et al.[25]	81.91	81.82
Shirakawa et al. [45]	83.02	82.29
He et al. [24]	83.37	84.82

Table 9: Performance on the AIDA data.

5.2.3 AIDA Evaluation

On the AIDA data [25], we compare our methods against previous work using micro-accuracy (A_{micro}) and macro-accuracy (A_{macro}) as defined in [2]. We do not tune the system on any part of the AIDA data and we ran our entity linking methods on the entire dataset (train, testA, testB) (cf. [2]).

Our results are reported in Table 9. Our FWBW disambiguation method beats the state-of-the-art system [2] on A_{macro} and comes close to [25] on A_{micro}. This is an encouraging result given that we did not tune our system to the AIDA data. Luo et al. [32] used *train* and *testA* for training and validation respectively. Our performance on *testB* ($Pr@1 = 79.7$) is close to their $JERL_{el}$ ($Pr@1 = 81.4$) without tuning on the data.

5.3 Runtime performance

Our methods are lightweight because:

- they are fast (Table 11),
- they have a small memory footprint (Table 10),
- and they use features (primarily word and entity embeddings) that are easy to extend to more languages

Our best candidate entity retrieval (FEL) and disambiguation (FWBW) methods are implemented in Java 8 and have been profiled for document throughput (Table 11). Our best method (FEL+FwBw) is 2.5 times faster than a recent Belief-Propagation based Entity Linking System [19] on the AIDA dataset (178 vs 445.56 milli-secs/doc).[6]

	# Entities	Data pack	# Vectors	Wiki
EN	4.9M	1.6GB	1.5GB	45GB
ES	1.10M	114M	877MB	9.8GB
ZH	870K	272MB	864MB	5.3GB

Table 10: Size of the data pack for each language and size of the original Wikipedia dumps.

Datasets	Docs	Average mentions	Sec/doc
AIDA-all	1392	36.43	0.178
KBP-EN	1820	82.4	0.473
KBP-ES	1175	5.37	0.004
KBP-ZH	1224	14.54	0.013

Table 11: runtime of FEL + FwBw on different datasets.

Our models are also memory-efficient as shown in Table 10, *e.g.*, 4.9M English entities are compressed in a 1.6GB datapack. In contrast, Wikifier, a popular English entity linker [8] relies on Lucene indexes and gazetteers that have a memory footprint of 6.8GB. The code and the entity embeddings are available at `https://github.com/yahoo/FEL`.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have described an efficient and accurate multilingual NER/NEL system. Our NER implementation is outperformed only by NER systems that use much more complex feature engineering and/or modeling methods. Our compact and efficient candidate entity retrieval method, FEL, has high precision; with the efficient FWBW disambiguation method, we obtain state-of-the-art performance on English NEL on the TAC KBP 2013 and AIDA data

[6]This comparison is indirect; we could not run their system and they did not report hardware specifications for their experiments.

sets. Aspects of our approach that contribute to this strong performance are compact entity embeddings that capture some of the features commonly used for entity disambiguation, and the use of information from search click logs.

In future work, we plan to improve the performance of our system for other languages, by expanding the pool of entities for which we have information since we noticed that candidate entity retrieval in Spanish is relatively poor compared to English and Chinese. We also plan to expand our use of entity embeddings to cover entity aliases as well as CFs and perform mention clustering for mentions do not link to any entity. Finally, we plan to experiment with increasing the size of the n-best list input to entity disambiguation, with the goal of increasing recall while holding precision high.

7. REFERENCES

[1] R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena. Polyglot-NER: Massive multilingual named entity recognition. In *Proc. ICDM*, 2015.

[2] A. Alhelbawy and R. Gaizauskas. Collective named entity disambiguation using graph ranking and clique partitioning approaches. In *Proc. COLING*, 2014.

[3] S. Austin, R. Schwartz, and P. Placeway. The forward-backward search algorithm. In *Proc. ICASSP*, 1991.

[4] R. Blanco, G. Ottaviano, and E. Meij. Fast and space-efficient entity linking for queries. In *Proc. WSDM*, 2015.

[5] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proc. EACL*, 2006.

[6] D. Ceccarelli et al. Learning relatedness measures for entity linking. In *Proc. CIKM*, 2013.

[7] W. Che, M. Wang, C. D. Manning, and T. Liu. Named entity recognition with bilingual constraints. In *Proc. HLT-NAACL*, 2013.

[8] X. Cheng and D. Roth. Relational inference for wikification. In *Proc. EMNLP*, 2013.

[9] A. Chisholm and B. Hachey. Entity disambiguation with web links. *Trans. of the ACL*, 3:145–156, 2015.

[10] S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *Proc. EMNLP*, 2007.

[11] B. Dalvi, E. Minkov, P. Talukdar, and W. Cohen. Automatic gloss finding for a knowledge base using ontological constraints. In *Proc. WSDM*, 2015.

[12] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 248–255, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

[13] G. Durrett and D. Klein. A joint model for entity analysis: Coreference, typing, and linking. *Trans. of the ACL*, 2:477–490, 2014.

[14] P. Elias. Efficient storage and retrieval by content and address of static files. *Journal of the ACM*, 21(2):246–260, 1974.

[15] A. Fahrni, B. Heinzerling, T. Göckel, and M. Strube. HITS' monolingual and cross-lingual entity linking system at TAC 2013. In *Proc. TAC*, 2013.

[16] N. Fernandez Garcia, J. Arias Fisteus, and L. Sanchez Fernandez. Comparative evaluation of link-based approaches for candidate ranking in link-to-wikipedia systems. *Journal of Artificial Intelligence Research*, 49:733–773, 2014.

[17] J. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proc. ACL*, 2005.

[18] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[19] O.-E. Ganea et al. Probabilistic bag-of-hyperlinks model for entity linking. In *Proc. WWW*, 2016.

[20] Z. Guo and D. Barbosa. Robust entity linking via random walks. In *Proc. CIKM*, 2014.

[21] B. Hachey, W. Radford, and J. R. Curran. Graph-based named entity linking with Wikipedia. In *Proc. WISE*, 2011.

[22] D. Hakkani-Tür et al. Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding. In *Proc. INTERSPEECH*, 2014.

[23] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *Proc. SIGIR*, 2011.

[24] Z. He et al. Learning entity representation for entity disambiguation. In *Proc. ACL*, 2013.

[25] J. Hoffart et al. Robust disambiguation of named entities in text. In *Proc. EMNLP*, 2011.

[26] H. Ji, J. Nothman, and B. Hachey. Overview of TAC-KBP2014 entity discovery and linking tasks. In *Proc. TAC*, 2014.

[27] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of Wikipedia entities in web text. In *Proc. KDD*, 2009.

[28] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.

[29] G. Lample et al. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[30] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proc. ICML*, 2014.

[31] X. Ling, S. Singh, and D. Weld. Design challenges for entity linking. *Trans. of the ACL*, 3:315–328, 2015.

[32] G. Luo, X. Huang, C.-Y. Lin, and Z. Nie. Joint named entity recognition and disambiguation. In *Proc. EMNLP*, 2015.

[33] X. Ma and E. Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*, 2016.

[34] E. Meij, K. Balog, and D. Odijk. Entity linking and retrieval tutorial. http://ejmeij.github.io/entity-linking-and-retrieval-tutorial/, 2014.

[35] Y. Merhav et al. Basis Technology at TAC 2013 entity linking. In *Proc. TAC*, 2013.

[36] T. Mikolov et al. Distributed representations of words and phrases and their compositionality. In *Proc. NIPS*, 2013.

[37] N. Okazaki. CRFsuite: a fast implementation of conditional random fields (CRFs). http://www.chokkan.org/software/crfsuite/, 2007.

Wiki Entities	En	Es	Zh
Pizza	calzone, Meatball pizza, sandwiches, Chicago-style pizza, pepperoni, hamburger, pastrami, List of pizza varieties by country, Sicilian pizza, Pizza in the United States	Pizza marinera, Salchicha (*Sausage*), Hamburguesa *Burger*, Pizza caprichosa, alitas de pollo (*Chicken Wings*), hamburguesa con queso (*Cheese Burger*), Calzone, tiras de pollo (*Chicken Strips*), King Taco, Salsa	脆皮奶酥蛋糕 (*Mexican Burritos*), 宜宾燃面 (*Yíbīn rán miàn*), 焖子 (*Mèn zi*), 番茄沙司 (*Tomato Sauce*), 肉馅饼 (*Meat Pie*), 凉拌卷心菜 (*Cole Saw*), 辣子鸡 (*Spicy Chicken*), 比薩餅歷史 (*Pizza History*), 鸡蛋饼 (*Egg cake*)
Brad Pitt	Angelina Jolie, Ryan Gosling, Leonardo DiCaprio, Julia Roberts, Matt Damon, George Clooney, Michael Pitt, Kate Hudson, Brad Pitt filmography, Jennifer Aniston	Matt Damon, Angelina Jolie, Kevin Spacey, George Clooney, Tom Cruise, Sean Penn, Bradley Pitt, Leonardo DiCaprio, Chris O'Donnell, Alec Baldwin	布莱德彼特 (*Brad Pitt*), Brad Pitt, 畢彼特 (*Brad Pitt*), 布萊德·比特 (*Brad Pitt*), 布萊德·彼特 (*Brad Pitt*), 畢比特 (*Bi Bits*), 布萊德·彼特 (*Brad Pitt*), 毕拉斯普巨猿 (*Bì lā sī pǔ jù yuán*), 布拉德·彼特 (*Brad Pitt*), 保羅·碧坦尼 (*Bǎoluó·bì tǎn ní*)
Tokyo	Tokyo Japan, Osaka, Nagoya, Yokohama, Setagaya Tokyo, Koto Tokyo, Tokyo Tokyo, Bunkyo Tokyo, Ota Tokyo, Fukuoka	Osaka, Nagoya, Fukuoka, Nagoya, Yokohama, Prefectura de Tokio, Kōbe, Hiroshima, Kobe, Rascacielos en Tokio (*Skyscrapers in Tokyo*)	東京都 (*Tokyo*), 多摩地域 (*Western Tokyo*), 關東地方 (*Kanto region*), 青島村 (日本) (*Qingdao village in Japan*), 千葉市 (*Chiba*), 新宿區 (*Shinjuku City*), 神津島村 (*Kōzushima*), 利島村 (*To-shima*), 南關東 (*South Kanto*), 台東區 (*Taito*)
Dog	Cat, domestic dog, hunting dog, dog breed, herding dog, terrier, companion dog, dog type, hound	Gato (*Cat*), Canis lupus familiaris, Obesidad en perros (*Obesity in Dogs*), Gato domestico (*Domestic Cat*), Felis silvestris catus, Pequeño perro león (*Lōwchen*), Perros de aguas (*Spaniels*), Perro cobrador (*Retriever*), Guepardo (*Cheetah*), Perro guardián de ganado (*Guardian dog*)	狗更犬 (*Terrier*), 比格犬 (*Beagle*), 高砂犬 (*Takasago dog*), 台灣土狗 (*Taiwan dog*), 米格魯獵兔犬 (*Beagle-Harrier*), 狗子 (*Gouzi*), 鐵利亞 (*Terrier*), 米格魯 (*Beagle*), 狗食 (*Dog Food*), 獒 (*Molosser*)

Figure 2: Examples of top-10 nearest neighbors for Wikipedia entities in different languages from our entity embeddings. Translations / Transliterations for Spanish and Chinese examples are generated using Google Translate. Wikipedia entities are learnt as low-dimensional vectors, using a distributed memory model from the content and context of the Wikipedia page associated with each entity. We incorporate both lexical context and entity relationships into our entity embeddings. More details are provided in Section 3.1. In Chinese, the `Brad Pitt` example is particularly interesting because a foreign person name can have different surface forms (in Mandarin) and different Wikipedia pages, but they are close to each other in the embedding space. In the `Pizza` example, local/regional food is in the proximity of `Pizza` in Spanish and Chinese.

[38] N. Okazaki and J. Nocedal. Liblbfgs: a library of limited-memory broyden-fletcher-goldfarb-shanno (l-bfgs). *URL http://www. chokkan. org/software/liblbfgs*, 2010.

[39] A. Passos, V. Kumar, and A. McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367*, 2014.

[40] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[41] D. Rao, P. McNamee, and M. Dredze. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, Multilingual Information Extraction and Summarization*, pages 93–115. Springer, 2013.

[42] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Proc. CoNLL*, 2009.

[43] D. Roth, H. Ji, M.-W. Chang, and T. Cassidy. Wikification and beyond: The challenges of entity and concept grounding. *Proc. ACL*, 2014.

[44] W. Shen, J. Wang, P. Luo, and M. Wang. Linking named entities in tweets with knowledge base via user interest modeling. In *Proc. KDD*, 2013.

[45] M. Shirakawa et al. Entity disambiguation based on a probabilistic taxonomy. Technical Report MSR-TR-2011-125, Microsoft Research, 2011.

[46] A. Sil and A. Yates. Re-ranking for joint named-entity recognition and linking. In *Proc. CIKM*, 2013.

[47] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proc. EMNLP*, 2011.

[48] J. Suzuki and H. Isozaki. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proc. ACL-HLT*, 2008.

[49] P. P. Talukdar and K. Crammer. New regularized algorithms for transductive learning. In *Proc. ECML PKDD*, 2009.

[50] E. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. HLT-NAACL*, 2003.

[51] E. F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proc. CoNLL*, 2002.

[52] M. Yu, S. Wang, C. Zhu, and T. Zhao. Semi-supervised learning for word sense disambiguation using parallel corpora. In *Proc. FSKD*, 2011.

[53] Y. Zhou et al. Resolving surface forms to Wikipedia topics. In *Proc. COLING*, 2010.

[54] Z. Zuo, G. Kasneci, T. Gruetze, and F. Naumann. BEL: Bagging for entity linking. In *Proc. COLING*, 2014.