Joint Named Entity Recognition and Disambiguation

Gang Luo[1], **Xiaojiang Huang**[2], **Chin-Yew Lin**[2], **Zaiqing Nie**[2]
[1]Microsoft, California, USA
[2]Microsoft Research, Beijing, China
{gluo, xiaojih, cyl, znie}@microsoft.com

Abstract

Extracting named entities in text and linking extracted names to a given knowledge base are fundamental tasks in applications for text understanding. Existing systems typically run a named entity recognition (NER) model to extract entity names first, then run an entity linking model to link extracted names to a knowledge base. NER and linking models are usually trained separately, and the mutual dependency between the two tasks is ignored. We propose JERL, Joint Entity Recognition and Linking, to jointly model NER and linking tasks and capture the mutual dependency between them. It allows the information from each task to improve the performance of the other. To the best of our knowledge, JERL is the first model to jointly optimize NER and linking tasks together completely. In experiments on the CoNLL'03/AIDA data set, JERL outperforms state-of-art NER and linking systems, and we find improvements of 0.4% absolute F_1 for NER on CoNLL'03, and 0.36% absolute precision@1 for linking on AIDA.

1 Introduction

In applications of complex Natural Language Processing tasks, such as automatic knowledge base construction, entity summarization, and question answering systems, it is essential to first have high quality systems for lower level tasks, such as part-of-speech (POS) tagging, chunking, named entity recognition (NER), entity linking, and parsing among others. These lower level tasks are usually decoupled and optimized separately to keep the system tractable. The disadvantage of the decoupled approach is that each lower level task is not aware of other tasks and thus not able to leverage information provided by others to improve performance. What is more, there is no guarantee that their outputs will be consistent.

This paper addresses the problem by building a joint model for Entity Recognition and Disambiguation (ERD). The goal of ERD is to extract named entities in text and link extracted names to a knowledge base, usually Wikipedia or Freebase. ERD is closely related to NER and linking tasks. NER aims to identify named entities in text and classify mentions into predefined categories such as persons, organizations, locations, etc. Given a mention and context as input, entity linking connects the mention to a referent entity in a knowledge base.

Existing ERD systems typically run a NER to extract entity mentions first, then run an entity linking model to link mentions to a knowledge base. Such a decoupled approach makes the system tractable, and both NER and linking models can be optimized separately. The disadvantages are also obvious: 1) errors caused by NER will be propagated to linking and are not recoverable 2) NER can not benefit from information available used in entity linking; 3) NER and linking may create inconsistent outputs.

We argue that there is strong mutual dependency between NER and linking tasks. Consider the following two examples:

---

1. The New York Times (NYT) is an American daily newspaper.
2. Clinton plans to have more news conferences in 2nd term. WASHINGTON 1996-12-06

Example 1 is the first sentence from the Wikipedia article about "The New York Times". It is reasonable but incorrect for NER to identify "New York Times" without "The" as a named entity, while entity linking has no trouble connecting "The New York Times" to the correct entity.

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 879–888,
Lisbon, Portugal, 17-21 September 2015.

Example 2 is a news title where our NER classifies "WASHINGTON" as a location, since a location followed by a date is a frequent pattern in news articles it learned, while the entity linking prefers linking this mention to the U.S. president "George Washington" since another president's name "Clinton" is mentioned in the context. Both the entity boundaries and entity types predicted by NER are correlated to the knowledge of entities linked by entity linking. Modeling such mutual dependency is helpful in resolving inconsistency and improving performance for both NER and linking.

We propose JERL, Joint Entity Recognition and Linking, to jointly model NER and linking tasks and capture the mutual dependency between them. It allows the information from each task to improve the performance of the other. If NER is highly confident on its outputs of entity boundaries and types, it will encourage entity linking to link an entity which is consistent with NER's outputs, and vice versa. In other words, JERL is able to model how consistent NER and linking's outputs are, and predict coherent outputs. According to our experiments, this approach does improve the end to end performance. To the best of our knowledge, JERL is the first model to jointly optimize NER and linking tasks together completely .

Sil (2013) also proposes jointly conducting NER and linking tasks. They leverage existing NER/chunking systems and Freebase to over generate mention candidates and leave the linking algorithm to make final decisions, which is a reranking model. Their model captures the dependency between entity linking decisions and mention boundary decisions with impressive results. The difference between our model and theirs is that our model jointly models NER and linking tasks from the training phrase, while their model is a combined one which depends on an existing state-of-art NER system. Our model is more powerful in capturing mutual dependency by considering entity type and confidences information, while in their model the confidence of outputs is lost in the linking phrase. Furthermore, in our model NER can naturally benefit from entity linking's decision since both decisions are made together, while in their model, it is not clear how the linking decision can help the NER decision in return.

Joint optimization is costly. It increases the problem complexity, is usually inefficient, and requires the careful consideration of features of multiple tasks and mutual dependency, making proper assumptions and approximations to enable tractable training and inference. However, we believe that joint optimization is a promising direction for improving performance for NLP tasks since it is closer to how human beings process text information. Experiment result indicates that our joint model does a better job at both NER and linking tasks than separate models with the same features, and outperforms state-of-art systems on a widely used data set. We found improvements of 0.4% absolute F_1 for NER on CoNLL'03 and 0.36% absolute precision@1 for linking on AIDA. NER is a widely studied problem, and we believe our improvement is significant.

The contributions of this paper are as follows:
1. We identify the mutual dependency between NER and linking tasks, and argue that NER and linking should be conducted together to improve the end to end performance.
2. We propose the first completely joint NER and linking model, JERL, to train and inference the two tasks together. Efficient training and inference algorithms are also presented.
3. The JERL outperforms the best NER record on the CoNLL'03 data set, which demonstrates how NER could be improved further by leveraging knowledge base and linking techniques.

The remainder of this paper is organized as follows: the next section discusses related works on NER, entity linking, and joint optimization; section 3 presents our Joint Entity Recognition and Linking model in detail; section 4 describes experiments, results, and analysis; and section 5 concludes.

2 Related Work

The NER problem has been widely addressed by symbolic, statistical, as well as hybrid approaches. It has been encouraged by several editions of evaluation campaigns such as MUC (Chinchor and Marsh, 1998), the CoNLL 2003 NER shared task (Tjong Kim Sang and De Meulder, 2003) and ACE (Doddington et al., 2004). Along with the improvement of Machine Learning techniques, statistical approaches have become a major direction for research on NER, especially after Conditional Random Field is proposed by Lafferty et al. (2001). The well known state-of-art NER systems are Stanford NER (Finkel et al., 2005) and UIUC

NER (Ratinov and Roth, 2009). Liang (2005) compares the performance of the 2nd order linear chain CRF and Semi-CRF (Sarawagi and Cohen, 2004) in his thesis. Lin and Wu (2009) cluster tens of millions of phrases and use the resulting clusters as features in NER reporting the best performance on the CoNLL'03 English NER data set. Recent works on NER have started to focus on multi-lingual named entity recognition or NER on short text, e.g. Twitter.

Entity linking was initiated with Wikipedia-based works on entity disambiguation (Bunescu and Pasca, 2006; Cucerzan, 2007). This task is encouraged by the TAC 2009 KB population task[1] first and receives more and more attention from the research community (Hoffart et al., 2011; Ratinov et al., 2011; Han and Sun, 2011). Linking usually takes mentions detected by NER as its input. Stern et al. (2012) and Wang et al. (2012) present joint NER and linking systems and evaluate their systems on French and Chinese data sets. Sil and Yates (2013) take a re-ranking based approach and achieve the best result on the AIDA data set. In 2014, Microsoft and Google jointly hosted "Entity Recognition and Disambiguation Challenge" which focused on the end to end performance of linking system [2].

Joint optimization models have been studied at great length. E.g. Dynamic CRF (McCallum et al., 2003) has been proposed to conduct Part-of-Speech Tagging and Chunking tasks together. Finkel and Manning (2009) show how to model parsing and named entity recognition together. Yu et al. (2011) work on jointly entity identification and relation extraction from Wikipedia. Sil's (2013) work on jointly NER and linking is described in the introduction section of this paper. It is worth noting that joint optimization does not always work. The CoNLL 2008 shared task (Surdeanu et al., 2008) was intended to encourage jointly optimize parsing and semantic role labeling, but the top performing systems decoupled the two tasks.

3 Joint Entity Recognition and Linking

Named entity recognition is usually formalized as a sequence labeling task, in which each word is classified to not-an-entity or entity labels. Conditional Random Fields (CRFs) is one of the popular models used. Most features used in NER are word-level (e.g. a word sequence appears at position i or whether a word contains exactly four digits). It is hard, if not impossible, to encode entity-level features (such as "entity length" and "correlation to known entities") in traditional CRF. Entity linking is typically formalized as a ranking task. Features used for entity linking are at entity-level inherently (such as entity prior probability; whether there are any related entity names or discriminative keywords occurring in the context).

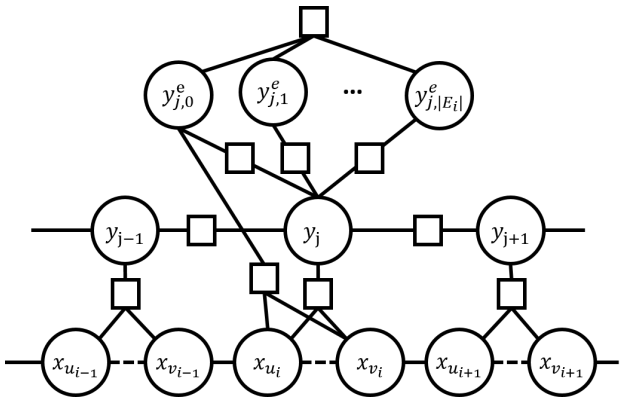

Figure 1: The factor graph of JERL model

The main challenges of joint optimization between NER and linking are: how to combine a sequence labeling model and a ranking model; and how to incorporate word-level and entity-level features. In a linear chain CRF model, each word's label is assumed to depend on the observations and the label of its previous word. Semi-CRF carefully relaxes the Markov assumption between words in CRF, and models the distribution of segmentation boundaries directly. We further extend Semi-CRF to model entity distribution and mutual dependency over segmentations, and name it Joint Entity Recognition and Linking (JERL). The model is described below.

3.1 JERL

Let $\boldsymbol{x} = \{x_i\}$ be a word sequence containing $|\boldsymbol{x}|$ words. Let $\boldsymbol{s} = \{s_j\}$ be a segmentation assignment over $\boldsymbol{x}$, where segment $s_j = (u_j, v_j)$ consist of a start position u_j and an end position v_j. All segments have a positive length and are adjacent to each other, so every (u_j, v_j) always satisfies $1 \leq u_j \leq v_j \leq |\boldsymbol{x}|$ and $u_{j+1} = v_j + 1$. Let $\boldsymbol{y} = \{y_j\}$ be labels in a fixed label alphabet $\mathcal{Y}$ over a segmentation assignment $\boldsymbol{s}$. Here $\mathcal{Y}$ is the set of types NER to predict. $x_{s_j} = (x_{u_j} \dots x_{v_j})$

[1] http://www.nist.gov/tac/2014/KBP/

[2] http://web-ngram.research.microsoft.com/ERD2014/

is the corresponding word sequence to s_j, and $\mathcal{E}_{s_j} = \{e_{j,k}\}$ is a set of entities in the knowledge base (KB), which may be referred by word sequence x_{s_j} in the entity linking task. Each entity $e_{j,k}$ is associated with a label $y^e_{j,k} \in \{0, 1\}$. Label $y^e_{j,k}$ takes 1 iff x_{s_j} referring to entity $e_{j,k}$, and 0 otherwise. If x_{s_j} does not refer to any entity in the KB, $y^e_{j,0}$ takes 1, which is analogous to the NIL[3] identifier in entity linking.

Based on the preliminaries and notations, Figure 1 shows the factor graph (Kschischang et al., 2001) of JERL. There are similar factor nodes for every $(u_j, v_j, y^e_{j,k})$, we only show the first one $(u_j, v_j, y^e_{j,0})$ for clarity.

Given $\boldsymbol{x}$, let $\boldsymbol{a} = (\boldsymbol{s}, \boldsymbol{y}, \boldsymbol{y^e})$ be a joint assignment, and $\boldsymbol{g}(\boldsymbol{x}, j, \boldsymbol{a})$ be local functions for x_{s_j}, namely features, each of which maps an assignment $\boldsymbol{a}$ to a measurement $g^k(x, j, a) \in \Re$. Then $\boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a}) = \sum_{j=1}^{|s|} \boldsymbol{g}(\boldsymbol{x}, j, \boldsymbol{a})$ is the factor graph defining a probability distribution of assignment $\boldsymbol{a}$ conditioned on word sequence $\boldsymbol{x}$.

Then JERL, conditional probability of $\boldsymbol{a}$ over $\boldsymbol{x}$, is defined as:

$$P(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{Z(\boldsymbol{x})} e^{\boldsymbol{w} \cdot \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a})} \quad (1)$$

where $\boldsymbol{w}$ is the weight vector corresponding to $\boldsymbol{G}$ will be learned later, and $Z(\boldsymbol{x})$ is the normalization factor $Z(\boldsymbol{x}) = \sum_{\boldsymbol{a} \in A} e^{\boldsymbol{w} \cdot \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a})}$, in which A is the union of all possible assignments over $\boldsymbol{x}$.

JERL is a probabilistic graphical model. More specificly, as shown in Figure 1, there are three groups of local functions and one constrain introduced. Each of them take a different role in JERL, as described below:

Features defined on $\boldsymbol{x}$, s_j, y_j, y_{j-1} are written as $\boldsymbol{g}^{ner}(\boldsymbol{x}, s_j, y_j, y_{j-1})$. These functions model segmentation and entity types' distribution over $\boldsymbol{x}$. Actually, every local features used in NER can be formulated in this way, and thus can be included in JERL. We thus refer to them as "NER features".

Features defined on $\boldsymbol{x}$, s_j, $y^e_{j,k}$ are written as $\boldsymbol{g}^{el}(\boldsymbol{x}, s_j, y^e_{j,k})$ and are called "linking features". These features model joint probabilities of word sequence x_{s_j} and linking decisions $y^k_{j,k} = 1(0 \leq k \leq |\mathcal{E}_{s_j}|)$ given context $\boldsymbol{x}$. JERL incorporates all linking features in this way.

Features defined on y_j, $y^e_{j,k}$ are written as $\boldsymbol{g}^{cr}(y_j, y^e_{j,k})$. These features model "mutual dependency" between NER and linking's outputs. For each entity $e_{j,k}$, there is additional information available in the knowledge base, e.g. categories information, popularity and relationship to other entities. These features encourage predicting coherent outputs for NER and linking.

There is one constrain for each $\boldsymbol{y^e_j}$ that the corresponding x_{s_j} can refer to only one entity $e_{j,k} \in \mathcal{E}_{sj}$ or NIL. This is equivalent to $\sum_{k=0}^{|\mathcal{E}_{s_j}|} y^e_{j,k} = 1$.

Based on the above description, $\boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a})$ in equation 1 is the sum of conjunction $(\boldsymbol{g}^{ner}, \boldsymbol{g}^{el}, \boldsymbol{g}^{cr})$ over $\boldsymbol{s}$, and can be rewritten as,

$$\begin{aligned} \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a}) = \textstyle\sum_{j=1}^{|\boldsymbol{s}|} (\quad & \boldsymbol{g}^{ner}(\boldsymbol{x}, s_j, y_j, y_{j-1}) \\ & , \textstyle\sum_{k=0}^{|\mathcal{E}_j|} \boldsymbol{g}^{el}(\boldsymbol{x}, s_j, y^e_{j,k}) \\ & , \textstyle\sum_{k=0}^{|\mathcal{E}_j|} \boldsymbol{g}^{cr}(\boldsymbol{x}, y_j, y^e_{j,k}) \ ) \end{aligned}$$

In summary, JERL jointly models the NER and linking, and leverages mutual dependency between them to predict coherent outputs. Previous works (Cucerzan, 2007; Ratinov et al., 2011; Sil and Yates, 2013) on linking argued that entity linking systems often suffer because of errors involved in mention detection phrase, especially false negative errors, and try to mitigate it via overgenerating mention candidates. From the mention generation perspective, JERL actually considers every possible assignment and is able to find the optimal $\boldsymbol{a}$.

3.2 Parameter Estimation

We describe how to conduct parameter estimation for JERL in this section. Given independent and identically distributed (i.i.d.) training data $\boldsymbol{T} = \{(\boldsymbol{x_t}, \boldsymbol{a}_t)\}_{t=1}^N$, the goal of parameter estimation is to find optimal $\boldsymbol{w^*}$ to maximize the joint probability of the assignments $\{\boldsymbol{a_t}\}$ over $\{\boldsymbol{x_t}\}$.

$$\boldsymbol{w}^* = argmax_{\boldsymbol{w} \in \Re^{|\boldsymbol{G}|}} \prod_{t=1}^{N} P(\boldsymbol{a_t}|\boldsymbol{x_t}, \boldsymbol{w})$$

We use conditional log likelihood with ℓ_2 norm as the objective function in training,

$$\mathcal{L}(\boldsymbol{T}, \boldsymbol{w}) = \ \textstyle\sum_t logP(\boldsymbol{a_t}|\boldsymbol{x_t}, \boldsymbol{w}) - \frac{1}{2\sigma^2}||\boldsymbol{w}||_2^2$$

The above function is concave, adding regularization to ensure that it has exactly one global optimum. We adopt a limited-memory quasi-Newton method (Liu and Nocedal, 1989) to solve the optimization problem.

[3] In the entity linking task, if a given mention refers to an entity which is not in the knowledge base, linking system should return a special identifier "NIL".

The gradient of $\mathcal{L}(\boldsymbol{T}, \boldsymbol{w})$ is derived as,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \sum_t (\boldsymbol{G}(\boldsymbol{x_t}, \boldsymbol{a_t}) - \sum_{a'} \boldsymbol{G}(\boldsymbol{x_t}, \boldsymbol{a'}) P(\boldsymbol{a'}|\boldsymbol{x_t}, \boldsymbol{w})) - \frac{\boldsymbol{w}}{\sigma^2} \quad (2)$$

As shown in Figure 1, our model's factor graph is a tree, which means the calculation of the gradient is tractable.

Inspired by the forward backward algorithm (Sha and Pereira, 2003) and Semi-CRF (Sarawagi and Cohen, 2004), we leverage dynamic programming techniques to compute the normalization factor $Z_{\boldsymbol{w}}$ and marginal probability $P(\boldsymbol{a'_j}|\boldsymbol{x_t}, \boldsymbol{w})$ when $\boldsymbol{w}$ is given.(Sutton and McCallum, 2006) The parameter estimation algorithm is abstracted in Algorithm 1.

Algorithm 1: JERL parameter estimation

input : training data $\boldsymbol{T} = \{(\boldsymbol{x}_t, \boldsymbol{a}_t)\}_{t=1}^N$
output: the optimal $\boldsymbol{w}$

$\boldsymbol{w} \leftarrow \boldsymbol{0}$;
while *weight* $\boldsymbol{w}$ *is not converged* **do**
  $Z \leftarrow 0$;
  $\boldsymbol{w'} \leftarrow \boldsymbol{0}$;
  for $t \leftarrow 1$ **to** N **do**
    calculate $\boldsymbol{\alpha_t}, \boldsymbol{\beta_t}$ according to eq.3;
    calculate Z_t according to eq.4
    calculate $\boldsymbol{w'_t}$ according to eq.2, 5;
    $Z \leftarrow Z + Z_t$;
    $\boldsymbol{w'} \leftarrow \boldsymbol{w'} + \boldsymbol{w'_t}$;
  end
  update $\boldsymbol{w}$ to maximize log likelihood $\mathcal{L}(\boldsymbol{T}, \boldsymbol{w})$ under $(Z, \boldsymbol{w'})$ via L-BFGS;
end

Let $\alpha_{i,y}$ $(i \in [0, |x|], y \in \mathcal{Y})$ be the sum of potential functions of all possible assignments over $(x_1 \dots x_i)$ whose last segmentation's labels are y. Then $\alpha_{i,y}$ can be calculated recursively from $i = 0$ to $i = |\boldsymbol{x}|$ as below.

We first define base cases as $\alpha_{0,y} = 1|_{\{y \in \mathcal{Y}\}}$. When $i \in (0, |x|]$:

$$\alpha_{i,y} = \sum_{d=1}^{L} \sum_{y' \in \mathcal{Y}} \alpha_{i-d,y'} \psi^{ner}_{i-d+1,i,y,y'} \left(\sum_{\boldsymbol{y_j^e} \in Y_j^{e*}} \psi^{el.cr}_{i-d+1,i,y,\boldsymbol{y_j^e}} \right) \quad (3)$$

where L is the max segmentation length in Semi-CRF, and Y_j^{e*} is all valid assignments for $\boldsymbol{y_j^e}$ which satisfies $\sum_{k=0}^{|\mathcal{E}_{s_j}|} y^e_{j,k} = 1$. The $\psi^{ner}_{u_j,v_j,y_j,y_{j-1}}$ and $\psi^{el.cr}_{u_j,v_j,y_j,\boldsymbol{y_j^e}}$ are precomputed ahead as below,

$$\psi^{ner}_{u_j,v_j,y_j,y_{j-1}} = e^{\boldsymbol{w}^{ner} \cdot \boldsymbol{g}^{ner}(\boldsymbol{x}, s_j, y_j, y_{j-1})}$$

$$\psi^{el.cr}_{u_j,v_j,y_j,\boldsymbol{y_j^e}} = \prod_{k=0}^{|\mathcal{E}_j|} e^{\boldsymbol{w}^{el} \boldsymbol{g}^{el}(\boldsymbol{x}, s_j, y^e_{j,k}) + \boldsymbol{w}^{cr} \boldsymbol{g}^{cr}(y_j, y^e_{j,k})}$$

where $\boldsymbol{w}^{ner}$, $\boldsymbol{w}^{el}$ and $\boldsymbol{w}^{cr}$ are weights for $\boldsymbol{g}^{ner}$, $\boldsymbol{g}^{el}$ and $\boldsymbol{g}^{cr}$ in $\boldsymbol{w}$ accordingly.

The value of $Z_{\boldsymbol{w}}$ can then be written as

$$Z_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_y \alpha_{|\boldsymbol{x}|,y} \quad (4)$$

Define $\beta_{i,y}$ $(i \in [0, |x|], y \in \mathcal{Y})$ as the sum of potential functions of all possible assignments over $(x_{i+1} \dots x_{|\boldsymbol{x}|})$ whose first segmentation's labels are y. $\beta_{i,y}$ is calculated in a similar way, except they are calculated from $i = |x|$ to left $i = 0$.

Once we get $\{\alpha_{i,j}\}$ and $\{\beta_{i,j}\}$, the marginal probability of arbitrary assignment $a_j = (s_j, y_j, \boldsymbol{y_j^e})$, where $s_j = (u_j, v_j)$, can be calculated as below:

$$P(s_j, y_j|\boldsymbol{x}, \boldsymbol{w}) = \frac{(\sum_{y' \in \mathcal{Y}} \alpha_{u_j - 1, y'} \psi^{ner}_{u_j,v_j,y_j,y'}) \beta_{v_j,y_j}}{Z_{\boldsymbol{w}}(\boldsymbol{x})}$$

and

$$P(a_j|\boldsymbol{x_t}, \boldsymbol{w}) = P(s_j, y_j|\boldsymbol{x}, \boldsymbol{w}) \frac{\psi^{el.cr}_{u_j,v_j,y_j,\boldsymbol{y_j^e}}}{\sum_{\boldsymbol{y_{j'}^e} \in Y_j^{e*}} \psi^{el.cr}_{u_j,v_j,y_j,\boldsymbol{y_{j'}^e}}} \quad (5)$$

3.3 Inference

Given a new word sequence $\boldsymbol{x}$ and model weights $\boldsymbol{w}$ trained on a training set, the goal of inference is to find the best assignment, $\boldsymbol{a^*} = argmax_{\boldsymbol{a}} P(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{w})$ for $\boldsymbol{x}$. We extend the Viterbi algorithm to exactly infer the best assignment. The inference algorithm is shown in Algorithm 2.

Let $\phi(u_j, v_j, y_j, y_{j-1})$ be the product of potentials depending on (s_j, y_j, y_{j-1}) as,

$$\phi(u_j, v_j, y_j, y_{j-1}) = \psi^{ner}_{u_j,v_j,y_j,y_{j-1}} \left(\sum_{\boldsymbol{y_j^e} \in Y_j^{e*}} \psi^{el.cr}_{u_j,v_j,y_j,\boldsymbol{y_j^e}} \right) \quad (6)$$

Algorithm 2: JERL inference

input : one word sequence $\boldsymbol{x}$ and weights $\boldsymbol{w}$
output: the best assignment $\boldsymbol{a}$ over $\boldsymbol{x}$

// shrink JERL graph to a Semi-CRF graph;
for $u \leftarrow 1$ **to** $|\boldsymbol{x}|$ **do**
  for $v \leftarrow u+1$ **to** $|\boldsymbol{x}|$ **do**
    for $(y, y') \in \mathcal{Y} \times \mathcal{Y}$ **do**
      calculate $\phi_{u,v,y,y'}$ // see eq.6;
    end
  end
end
// infer the best assignment of $(\boldsymbol{s}^*, \boldsymbol{y}^*)$;
for $i \leftarrow 1$ **to** $|\boldsymbol{x}|$ **do**
  for $y \in \mathcal{Y}$ **do**
    calculate $V_{i,y}$ // see eq.7;
  end
end
$(\boldsymbol{s}^*, \boldsymbol{y}^*) \leftarrow argmax(V_{i,y})$;
// infer the best assignment of $\{\boldsymbol{y}_{\boldsymbol{j}}^{\boldsymbol{e}}\}$;
for $j \leftarrow 1$ **to** $|\boldsymbol{s}^*|$ **do**
  $\boldsymbol{y}_{\boldsymbol{j}}^{\boldsymbol{e}} \leftarrow argmax(P(|\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{s}_{\boldsymbol{j}}^*, \boldsymbol{y}_{\boldsymbol{j}}^*))$
end
$\boldsymbol{a}^* \leftarrow (\boldsymbol{s}^*, \boldsymbol{y}^*, \boldsymbol{y}^{\boldsymbol{e}*})$;

and let $V(i, y)$ denotes the largest value of $(\boldsymbol{w} \cdot \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{a'}))$ where $\boldsymbol{a'}$ could be any possible partial assignment starting from x_1 to x_i. The best $(\boldsymbol{s}^*, \boldsymbol{y}^*)$ are derived during the following recursive calculation,

$$V_{i,y} = \begin{cases} max_{y' \in \mathcal{Y}, d \in [1,L]} \\ (V_{i-d,y'} + \phi(i-d+1, i, y, y')) & i > 0 \\ 0 & i = 0 \\ -\infty & i < 0 \end{cases} \quad (7)$$

where L is the maximum segmentation length for Semi-CRF.

Once $(\boldsymbol{s}^*, \boldsymbol{y}^*)$ are found, the corresponding $\boldsymbol{y}_{\boldsymbol{j}}^{\boldsymbol{e}*} = argmax_{\{\boldsymbol{y}_{\boldsymbol{l}}^{\boldsymbol{e}} \in Y_j^{e*}\}}(\psi^{el.cr}_{u_j^*, v_j^*, y_j^*, \boldsymbol{y}_{\boldsymbol{l}}^{\boldsymbol{e}}})$ is also the optimal one. Then $\boldsymbol{a}^* = (\boldsymbol{s}^*, \boldsymbol{y}^*, \boldsymbol{y}^{\boldsymbol{e}*})$ is the best assignment for the given $\boldsymbol{x}$ and $\boldsymbol{w}$.

4 Experiments

In our experiments, we first construct two baseline models JERL$_{ner}$ and JERL$_{el}$, which use exact NER and EL feature sets used in JERL. Then evaluate JERL and the two baseline models against several state-of-art NER and linking systems. After that, we evaluate JERL under different feature settings to analysis the contributions of each features set, and show some examples we find. We also compare the training speed under different settings.

CoNLL'03	Training	Dev set	Test
Articles	946	216	231
Sentences	14,987	3,466	3,684
Tokens	203,621	51,362	46,435
Entities	23,499	5,942	5,648
NIL Entities	4,857	1,129	1,133

Table 1: Overview of CoNLL'03/AIDA data set

4.1 Data set

We take the CoNLL'03/AIDA English data set to evaluate the performance of NER and linking systems. CoNLL'03 is extensively used in prior work on NER evaluation (Tjong Kim Sang and De Meulder, 2003). The English data is taken from Reuters news articles published between August 1966 and August 1997. Four types of entities persons (PER), organizations (ORG), locations (LOC), and miscellaneous names (MISC) are annotated. Hoffart et al. (2011) hand-annotated all proper nouns with corresponding entities wiht YAGO2, Freebase and Wikipedia IDs. This data is referenced as AIDA here. To the best of our knowledge, this data set is the biggest data set which has been labeled for both NER and linking tasks. It becomes a really good starting point for our work. Table 1 contains of an overview of the CoNLL'03/AIDA data set.

For entity linking, we take Wikipedia as the referent knowledge base. We use a Wikipedia snapshot dumped in May 2013, which contains around 4.8 million articles. We also align our Wikipedia dump with additional knowledge bases, Freebase and Satori (a Microsoft internal knowledge base), to enrich the information of these entities.

4.2 Evaluation Metrics

We follow the CoNLL'03 metrics to evaluate NER performance by precision, recall, and F_1 scores, and follow Hoffart's (2011) experiment setting to evaluate linking performance by micro precision@1. Since the linking labels of CONLL'03 were annotated in 2011, it is not completely consistent with the Wikipedia dump we used in the case. We only consider mention entity pairs where the ground truth are known, and ignore around 20% of NIL mentions in the ground truth.

Category	Features
NER	Word unigram / bigram
	Lower cased unigram / bigram
	Word shape unigram / bigram
	Stemmed unigram / bigram
	POS unigram / bigram
	Chunk unigram / bigram
	Words in the 4 left/right window
	Character n-grams, $n \leq 4$
	Brown clusters
	WordNet clusters
	Dictionaries
Linking	Alternative names
	Entity priors
	Entity name priors
	Entity priors over names
	Context scores
	Geo distance
	Related entities
Mutual	Type-category correlation

Table 2: JERL feature list

4.3 JERL Implementation

Table 2 shows features used in our models. JERL uses all features in the three categories, while JERL$_{ner}$ and JERL$_{el}$ use only one corresponding category. All three models are trained on the train and development set, and evaluated on the test set of CoNLL'03/AIDA.

4.3.1 NER

Features in the NER category are relevant to NER. We considered the most commonly used features in literatures (Finkel et al., 2005; Liang, 2005; Ratinov and Roth, 2009). We collect several known name lists, like popular English first/last names for people, organization lists and so on from Wikipedia and Freebase. UIUC NER's lists are also included. In addition, we extract entity name lists from the knowledge base we used for entity linking, and construct 655 more lists. Although those lists are noisy, we find that statistically they do improve the performance of our NER baseline by a significant amount.

4.3.2 Linking

Features in linking category are relevant to entity linking. An entity can be referred by its canonical name, nick names, alias, and first/last names. Those names are defined as alternative names for this entity. We collect all alternative names for all known entities and build a name to entity index. This index is used to select entity candidates for any word sequence, also known as surface form. Following previous work by Han and Sun (2011), we calculate entity priors and entity name priors from Wikipedia. Context scores are calculated based on discriminative keywords. Geo distance and related entities capture the relatedness among entities in the given context.

4.3.3 Mutual

Features in this category capture the mutual dependency between NER and linking's outputs. For each entity in a knowledge base, there is category information available. We aggregate around 1000 distinct categories from multiple sources. One entity can have multiple categories. For example, London is connected to 29 categories. We use all combinations between NER types and categories as features in JERL, and let the model learn the correlation of each combination. This encourages coherent NER and EL decisions, which is one of the key contributions of our work.

4.3.4 Non-local features

Features capturing long distance dependency between hidden labels are classified as non-local features. Those features are very helpful in improving NER system performance but are costly. Since this is not the focus of this paper, we take a simple approach to incorporate non-local features. We cache history results of previous sentences in a 1000 words window, and adopt several heuristic rules for personal names. This approach contributes 0.2 points to the final NER F_1 score. Non-local features are also considered in linking (Ratinov et al., 2011; Han et al., 2011). We try several features, which has been proved to be helpful in TAC data set. However, the gain on CoNLL'03/AIDA data set is not obvious, we do not optimize linking globally.

Lastly, based on preliminary studies and experiments, we set the maximum segmentation length to 6 and max candidate count per segmentation to 5 for efficient training and inference.

4.4 State-of-Art systems

We take three state-of-art NER systems: NereL (Sil and Yates, 2013), UIUC NER (Ratinov and Roth, 2009) and Stanford NER (Finkel et al., 2005). NereL firstly over generates mentions and decomposes them to sets of connected compo-

Dataset	System	Prec.	Recall	F_1
CoNLL'03	Stanford	**95.1**	78.3	85.9
	UIUC	91.2	90.5	90.8
	NereL	86.8	89.5	88.2
	$JERL_{ner}$	90.0	89.9	89.9
	$JERL$	91.5	**91.4**	**91.2**

Table 3: NER evaluation results

Dataset	System	Precision@1
CoNLL'03	Kul09	76.74
	Hof11	81.91
	Nerel	84.22
	JERL_{el}	81.49
	JERL	**84.58**

Table 4: Linking evaluation results

#	Feature set description	NER F_1
0	JERL_{ner} (baseline)	89.9
1	+ candidate	88.7
2	+ candidate + linking	89.9
3	+ candidate + mutual	90.6
4	+ candidate + mutual + linking	91.2

Table 5: JERL features analysis

nents, then trains a maximum-entropy model to re-rank different assignments. UIUC NER uses a regularized averaged perceptron model and external gazetteers to achieve strong performance. In Addition, NereL also uses UIUC NER to generate mentions. Stanford NER uses Conditional Random Fields and Gibbs sampling to incorporate non-local features into its model.

For entity linking systems, NereL, Kul09 (Kulkarni et al., 2009) and Hof11 (Hoffart et al., 2011) are compared with our models. NereL achieves the best precision@1. Kul09 formulates the local compatibility and global coherence in entity linking, and optimizes the overall entity assignment for all entities in a document via a local hill-climbing approach. Hof11 unifies the prior probability of an entity being mentioned, the similarity between context and entity, and the coherence between entity candidates among all mentions in a dense graph.

4.5 Results

Table 3 shows the performance of different NER systems on the CoNLL'03 testb data set. We refer the numbers of state-of-art systems reported by Sil and Yates (2013). Stanford NER achieves the best precision, but its recall is low. UIUC reports the (almost) best recorded F_1. JERL_{ner} considers features only in the NER category, which could be treated as a pure NER system implemented in Semi-CRF. Actually CRF-based implementation with a similar feature set has comparable performance. Our baseline JERL_{ner} is strong enough. We argue that that it is mainly because of the additional dictionaries derived from the knowledge base. JERL further pushes the F_1 to 91.2, which outperforms UIUC by 0.4 points in F_1 score. To the best of our knowledge, it is the best F_1 on CoNLL'03 since 2009. The reason our model can outperform state-of-art systems is that, it has more knowledge about entities via incorporate entity linking techniques. If an entity can be linked to a well known entity via entity linking in high confidence, its mention boundary and entity type are confirmed implicitly.

Table 4 shows the performance of different entity linking systems on the AIDA test set. Kul09 and Hof11 use only the correct mentions detected by the Stanford NER as input, and thus their recall is bound by the recall of NER. NereL uses its overgeneration techniques to generate mention candidates, and outperforms Hoff11 in both precision and recall. Our baseline model JERL_{el} is also evaluated on Stanford NER generated mentions, which has comparable performance with Kul09 and Hof11. JERL achieves precision@1 84.58 which is better than NereL.

We run 15 trials for both NER and linking's experiments and report the average numbers above. The standard deviations are 0.11% and 0.08% for NER and linking separately, which pass the standard t-test with confidence level 5%, demonstrating the significance of our results.

In order to investigate how different features contribute to the overall gain. We compare JERL_{ner} with four different feature sets. Table 5 summaries the results. In the trial "+candidate", JERL expands every possible segmentation with corresponding entity list and builds its factor graph without any linking and mutual features. This version's F_1 drops to 88.7 which indicates the created structure is quite noisy. In the "+candidate +linking" trial, only linking features are enabled and the F_1 is comparable to the baseline. On the other side, in the "+candidate +mutual" trial when mutual features are enabled the F_1 increases to 90.6. If we combine both linking and mutual features,

Category	PER	LOC	ORG	Other
people.person	3.65	0.817	1.260	-1.782
location.city	-0.187	0.712	0.491	-0.188
sports.team	-0.180	2.382	3.595	-2.019

Table 6: Learned mutual dependency

Setting		NER	Linking	Training
MSL	MRC	F_1	prec@1	time (min)
4	5	87.9	76.74	195
5	5	90.8	84.01	234
6	1	90.8	80.13	37
6	3	91.0	83.21	109
6	5	91.2	84.58	280

Table 7: Training time under different settings

JERL achieves the reported performance. The result indicates that mutual features are the determining factor to the performance gain.

Table 6 shows weights of learned mutual dependency of three categories "people.person", "location.city", and "sports.team". The bigger a weight is, the more consistent this combination would be. From the weights, we find several interesting things. If an entity belongs to any of the three categories, it is less likely to be predicted as non-an-entity by NER. If an entity belongs to the category of "people.person", it more likely to be predicted as PER. When an entity belongs to the category "location.city" or "sports.team", NER may predict it as ORG or LOC. This is because in the CoNLL'03/AIDA data set, there are many sports teams mentioned by their city/country names. JERL successfully models such unexpected mutual dependency.

Table 7 compares the performance and training time under different settings of max segmentation length (MSL) and max referent count (MRC). We use machines with Intel Xeon E5620 @ 2.4GHz CPU (8 cores / 16 logical processors) and 48GB memory. We run every setting 10 times and report the averages. As MSL and MRC increasing, the performance is slightly better, but the training time increased a lot. MSL has linear impact on training time, while MRC affects training time more.

5 Conclusion and Future Work

In this paper, we address the problem of joint optimization of named entity recognition and linking. We propose a novel model, JERL, to jointly train and infer for NER and linking tasks. To the best of our knowledge, this is the first model which trains two tasks at the same time. The joint model is able to leverage mutual dependency of the two tasks, and predict coherent outputs. JERL outperforms the state-of-art systems on both NER and linking tasks on the CoNLL'03/AIDA data set.

For future works, we would like to study how to leverage existing partial labeled data, either for NER or for linking only, in joint optimization, and incorporate more NLP tasks together for multi-tasks joint optimization.

Acknowledgments

This work was performed when the first author was working at Microsoft Research. The first author is sponsored by Microsoft Bing Core Relevance team. Thanks Shuming Shi, Bin Gao, and Yohn Cao for their helpful guidance and valuable discussions. Additionally, we would like to thank the three anonymous reviewers for their insightful suggestions and detailed comments.

References

Razvan C Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, volume 6, pages 9–16.

Nancy Chinchor and Elaine Marsh. 1998. Muc-7 information extraction task definition. In *Proceeding of the seventh message understanding conference (MUC-7), Appendices*, pages 359–367.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, volume 7, pages 708–716.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *LREC*.

Jenny Rose Finkel and Christopher D Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334. Association for Computational Linguistics.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Xianpei Han and Le Sun. 2011. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 945–954. Association for Computational Linguistics.

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

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 Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Frank R Kschischang, Brendan J Frey, and H-A Loeliger. 2001. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. 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: Volume 2-Volume 2*, pages 1030–1038. Association for Computational Linguistics.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Andrew McCallum, Khashayar Rohanimanesh, and Charles Sutton. 2003. Dynamic conditional random fields for jointly labeling multiple sequences. In *nips workshop on syntax, semantics and statistics*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. 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-Volume 1*, pages 1375–1384. Association for Computational Linguistics.

Sunita Sarawagi and William W Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, pages 1185–1192.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics.

Avirup Sil and Alexander Yates. 2013. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2369–2374. ACM.

Rosa Stern, Benoît Sagot, and Frédéric Béchet. 2012. A joint named entity recognition and entity linking system. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 52–60. Association for Computational Linguistics.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The conll-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177. Association for Computational Linguistics.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128.

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-Volume 4*, pages 142–147. Association for Computational Linguistics.

Longyue Wang, Shuo Li, Derek F Wong, and Lidia S Chao. 2012. A joint chinese named entity recognition and disambiguation system. *CLP 2012*, page 146.

Xiaofeng Yu, Irwin King, and Michael R Lyu. 2011. Towards a top-down and bottom-up bidirectional approach to joint information extraction. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 847–856. ACM.