A Joint Framework for Coreference Resolution and Mention Head Detection

Haoruo Peng **Kai-Wei Chang** **Dan Roth**
University of Illinois, Urbana-Champaign
Urbana, IL, 61801
{hpeng7,kchang10,danr}@illinois.edu

Abstract

In coreference resolution, a fair amount of research treats mention detection as a pre-processed step and focuses on developing algorithms for clustering coreferred mentions. However, there are significant gaps between the performance on gold mentions and the performance on the real problem, when mentions are *predicted* from raw text via an imperfect Mention Detection (MD) module. Motivated by the goal of reducing such gaps, we develop an ILP-based joint coreference resolution and mention head formulation that is shown to yield significant improvements on coreference from raw text, outperforming existing state-of-art systems on both the ACE-2004 and the CoNLL-2012 datasets. At the same time, our joint approach is shown to improve mention detection by close to 15% F1. One key insight underlying our approach is that identifying and co-referring mention *heads* is not only sufficient but is more robust than working with complete mentions.

1 Introduction

Mention detection is rarely studied as a stand-alone research problem (Recasens et al. (2013) is one key exception). Most coreference resolution work simply mentions it in passing as a module in the pipelined system (Chang et al., 2013; Durrett and Klein, 2013; Lee et al., 2011; Björkelund and Kuhn, 2014). However, the lack of emphasis is not due to this being a minor issue, but rather, we think, its difficulty. Indeed, many papers report results in terms of gold mentions versus system generated mentions, as shown in Table 1. Current state-of-the-art systems show a very significant drop in performance when running on system generated mentions. These performance gaps are worrisome, since the real goal of NLP systems is to process raw data.

System	Dataset	Gold	Predict	Gap
Illinois	CoNLL-12	77.05	60.00	17.05
Illinois	CoNLL-11	77.22	60.18	17.04
Illinois	ACE-04	79.42	68.27	11.15
Berkeley	CoNLL-11	76.68	60.42	16.26
Stanford	ACE-04	81.05	70.33	10.72

Table 1: **Performance gaps between using gold mentions and predicted mentions for three state-of-the-art coreference resolution systems.** Performance gaps are always larger than 10%. Illinois's system (Chang et al., 2013) is evaluated on CoNLL (2012, 2011) Shared Task and ACE-2004 datasets. It reports an average F1 score of MUC, B^3 and $CEAF_e$ metrics using CoNLL v7.0 scorer. Berkeley's system (Durrett and Klein, 2013) reports the same average score on the CoNLL-2011 Shared Task dataset. Results of Stanford's system (Lee et al., 2011) are for B^3 metric on ACE-2004 dataset.

This paper focuses on improving end-to-end coreference performance. We do this by: 1) Developing a new ILP-based joint learning and inference formulation for coreference and mention head detection. 2) Developing a better mention head candidate generation algorithm. Importantly, we focus on heads rather than mention boundaries since those can be identified more robustly and used effectively in an end-to-end system. As we show, this results in a dramatic improvement in the quality of the MD component and, consequently, a significant reduction in the performance gap between coreference on gold mentions and coreference on raw data.

Existing coreference systems usually consider a pipelined system, where the mention detection step is followed by that of clustering mentions into coreference chains. Higher quality mention identification naturally leads to better coreference performance. Standard methods define mentions as *boundaries* of text, and expect *exact* boundaries as input in the coreference step. However, mentions have an intrinsic structure, in which mention heads carry the crucial information. Here, we define a mention head as the last token of a syntactic head, or the whole syntactic head for proper names.[1] For example, in "the

[1]Here, we follow the ACE annotation guideline. Note that

incumbent [Barack Obama]" and "[officials] at the Pentagon", "Barack Obama" and "officials" serve as mention heads, respectively. Mention heads can be used as auxiliary structures for coreference. In this paper, we first identify mention heads, and then detect mention boundaries based on heads. We rely heavily on the first, head identification, step, which we show to be sufficient to support coreference decisions. Moreover, this step also provides enough information for "understanding" the coreference output, and can be evaluated more robustly (since minor disagreements on mention boundaries are often a reason for evaluation issues when dealing with predicted mentions). We only identify the mention boundaries at the end, after we make the coreference decisions, to be consistent with current evaluation standards in the corefernce resolution community. Consider the following example[2]:

> [Multinational companies investing in [China]] had become so angry that [they] recently set up an anti-piracy *league* to pressure [the [Chinese] government] to take action. [Domestic manufacturers, [who] are also suffering], launched a similar body this month. [They] hope [the government] can introduce a new law increasing fines against [producers of fake goods] from the amount of profit made to the value of the goods produced.

Here, phrases in the brackets are mentions and the underlined simple phrases are mention heads. Moreover, mention boundaries can be nested (the boundary of a mention is inside the boundary of another mention), but mention heads never overlap. This property also simplifies the problem of mention head candidate generation. In the example above, the first "they" refers to "Multinational companies investing in China" and the second "They" refers to "Domestic manufacturers, who are also suffering". In both cases, the mention heads are sufficient to support the decisions: "they" refers to "companies", and "They" refers to "manufacturers". In fact, most of the features[3] implemented in existing coreference resolution systems rely solely on mention heads (Bengtson and Roth, 2008).

Furthermore, consider the possible mention candidate "league" (italic in the text). It is not chosen as a mention because the surrounding context is not focused on "anti-piracy league". So, mention detection can be viewed as a global decision problem, which involves considering the relevance of a mention to its context. The fact that the coreference decision provides a way to represent this relevance, further motivates considering mention detection and coreference jointly. The insight here is that a mention candidate will be more likely to be valid when it has more high confidence coreference links.

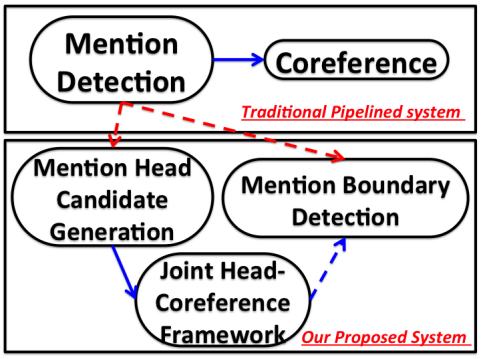

Figure 1: **Comparison between a traditional pipelined system and our proposed system.** We split up mention detection into two steps: mention head candidate generation and (an optional) mention boundary detection. We feed mention heads rather than complete mentions into the coreference model. During the joint head-coreference process, we reject some mention head candidates and then recover complete mention boundaries after coreference decisions are made.

This paper develops a joint coreference resolution and mention head detection framework as an Integer Linear Program (ILP) following Roth and Yih (2004). Figure 1 compares a traditional pipelined system with our proposed system. Our joint formulation includes decision variables both for coreference links between pairs of mention heads, and for all mention head candidates, and we simultaneously learn the ILP coefficients for all these variables. During joint inference, some of the mention head candidates will be rejected (that is, the corresponding variables will be assigned '0'), contributing to improvement both in MD and in coreference performance. The aforementioned joint approach builds on an algorithm that generates mention head candidates. Our candidate generation process consists of a statistical component and a component that makes use of existing resources, and is designed to ensure high recall on head candidates.

Ideally, after making coreference decisions, we extend the remaining mention heads to complete mentions; we employ a binary classifier, which shares all features with the mention head detection model in the joint step.

Our proposed system can work on both ACE and OntoNotes datasets, even though their styles of annotation are different. There are two main differ-

the CoNLL-2012 dataset is built from OntoNotes-5.0 corpus.

[2]This example is chosen from the ACE-2004 corpus.

[3]All features except for those that rely on modifiers.

ences to be addressed. First, OntoNotes removes singleton mentions, even if they are valid mentions. This causes additional difficulty in learning a good mention detector in a pipelined framework. However, our joint framework can adapt to it by rejecting those singleton mentions. More details will be discussed in Sec. 2. Second, ACE uses shortest denotative phrases to identify mentions while OntoNotes tends to use long text spans. This makes identifying mention boundaries unnecessarily hard. Our system focuses on mention heads in the coreference stage to ensure robustness. As OntoNotes does not contain head annotations, we preprocess the data to extract mention heads which conform with the ACE style.

Results on ACE-2004 and CoNLL-2012 datasets show that our system[4] reduces the performance gap for coreference by around 25% (measured as the ratio of performance improvement over performance gap) and improves the overall mention detection by over 10 F1 points. With such significant improvements, we achieve the best end-to-end coreference resolution results reported so far.

The main contributions of our work can be summarized as follows:

1. We develop a new, end-to-end, coreference approach that is based on a joint learning and inference model for mention heads and coreference decisions.
2. We develop an improved mention head candidate generation module and a mention boundary detection module.
3. We achieve the best coreference results on predicted mentions and reduce the performance gap compared to using gold mentions.

The rest of the paper is organized as follows. We explain the joint head-coreference learning and inference framework in Sec. 2. Our mention head candidate generation module and mention boundary detection module are described in Sec. 3. We report our experimental results in Sec. 4, review related work in Sec. 5 and conclude in Sec. 6.

2 A Joint Head-Coreference Framework

This section describes our joint coreference resolution and mention head detection framework. Our work is inspired by the latent left-linking model in Chang et al. (2013) and the ILP formulation from Chang et al. (2011). The joint learning and inference model takes as input mention head candidates (Sec. 3) and jointly (1) determines if they are indeed mention heads and (2) learns a similarity metric between mentions. This is done by simultaneously learning a binary mention head detection classifier and a mention-pair coreference classifier. The mention head detection model here is mainly trained to differentiate valid mention heads from invalid ones. By learning and making decisions jointly, it also serves as a singleton mention head classifier, building on insights from Recasens et al. (2013). This joint framework aims to improve performance on both mention head detection and on coreference.

We first describe the formualtion of the mention head detection and the ILP-based mention-pair coreference separately, and then propose the joint head-coreference framework.

2.1 Mention Head Detection

The mention head detection model is a binary classifier $g_m = w_1^\top \varphi(m)$, in which $\varphi(m)$ is a feature vector for mention head candidate m and w_1 is the corresponding weight vector. We identify a candidate m as a mention head if $g_m > 0$. The features utilized in the vector $\varphi(m)$ consist of: 1) Gazetteer features 2) Part-Of-Speech features 3) Wordnet features 4) Features from the previous and next tokens 5) Length of mention head. 6) Normalized Pointwise Mutual Information (NPMI) on the tokens across a mention head boundary 7) Feature conjunctions. Altogether there are hundreds of thousands of sparse features.

2.2 ILP-based Mention-Pair Coreference

Let M be the set of all mentions. We train a coreference model by learning a pairwise mention scoring function. Specifically, given a mention-pair (u,v) ($u,v \in M$, u is the antecedent of v), we learn a left-linking scoring function $f_{u,v} = w_2^\top \phi(u,v)$, where $\phi(u,v)$ is a pairwise feature vector and w_2 is the weight vector. The inference algorithm is inspired by the best-left-link approach (Chang et al., 2011), where they solve the following ILP problem:

$$\begin{aligned} &\arg\max_{y} \sum_{u<v,u,v\in M} f_{u,v} y_{u,v}, \\ &\text{s.t.} \sum_{u<v} y_{u,v} \leq 1, \quad \forall v \in M, \\ &\quad y_{u,v} \in \{0,1\} \quad \forall u,v \in M. \end{aligned} \tag{1}$$

Here, $y_{u,v} = 1$ iff mentions u,v are directly linked. Thus, we can construct a forest and the mentions in the same connected component (i.e., in the same tree) are co-referred. For this mention-pair coreference model $\phi(u,v)$, we use the same set of features used in Bengtson and Roth (2008).

[4] Available at http://cogcomp.cs.illinois.edu/page/software_view/Coref

2.3 Joint Inference Framework

We extend expression (1) to facilitate joint inference on mention heads and coreference as follows:

$$\begin{aligned}
&\arg\max_{y} \sum_{u<v,u,v\in M} f_{u,v}y_{u,v} + \sum_{m\in M} g_m y_m, \\
&\text{s.t.} \sum_{u<v} y_{u,v} \le 1, \quad \forall v \in M', \\
&\sum_{u<v} y_{u,v} \le y_v, \quad \forall v \in M', \\
&y_{u,v} \in \{0,1\}, \quad y_m \in \{0,1\} \quad \forall u,v,m \in M'.
\end{aligned}$$

Here, M' is the set of all mention head candidates. y_m is the decision variable for mention head candidate m. $y_m = 1$ if and only if the mention head m is chosen. To consider coreference decisions and mention head decisions together, we add the constraint $\sum_{u<v} y_{u,v} \le y_v$, which ensures that if a candidate mention head v is not chosen, then it will not have coreference links with other mention heads.

2.4 Joint Learning Framework

To support joint learning of the parameters w_1 and w_2 described above, we define a joint training objective function $C(w_1,w_2)$ for mention head detection and coreference, which uses a max-margin approach to learn both weight vectors. Suppose we have a collection of documents D, and we generate n_d mention head candidates for each document d ($d \in D$). We use an indicator function $\delta(u,m)$ to represent whether mention heads u,m are in the same coreference cluster based on gold annotations ($\delta(u,m) = 1$ iff they are in the same cluster). Similarly, $\Omega(m)$ is an indicator funtion representing whether mention head m is valid in the gold annotations.

For simplicity, we first define

$$\begin{aligned}
u' &= \arg\max_{u<m} (w_2^\top \phi(u,m) - \delta(u,m)), \\
u'' &= \arg \max_{u<m,\delta(u,m)=1} w_2^\top \phi(u,m)\Omega(m).
\end{aligned}$$

We then minimize the following joint training objective function $C(w_1,w_2)$.

$$\begin{aligned}
C(w_1,w_2) = \frac{1}{|D|}\sum_{d\in D}\frac{1}{n_d}\sum_m (C_{coref,m}(w_2) \\
+C_{local,m}(w_1) + C_{trans,m}(w_1)) + R(w_1,w_2).
\end{aligned}$$

$C(w_1,w_2)$ is composed of four parts. The first part is the loss function for coreference, where we have

$$\begin{aligned}
C_{coref,m}(w_2) = -w_2^\top \phi(u'',m)\Omega(m) \\
+ (w_2^\top \phi(u',m) - \delta(u',m))(\Omega(m) \vee \Omega(u')).
\end{aligned}$$

It is similar to the loss function for a latent left-linking coreference model[5]. As the second component, we have the quadratic loss for the mention head detection model,

$$C_{local,m}(w_1) = \frac{1}{2}(w_1^\top \varphi(m) - \Omega(m))^2.$$

Using the third component, we further maximize the margin between valid and invalid mention head candidates when they are selected as the best-left-link mention heads for any valid mention head. It can be represented as

$$C_{trans,m}(w_1) = \frac{1}{2}(w_1^\top \varphi(u') - \Omega(u'))^2\Omega(m).$$

The last part is the regularization term

$$R(w_1,w_2) = \frac{\lambda_1}{2}||w_1||^2 + \frac{\lambda_2}{2}||w_2||^2.$$

2.5 Stochastic Subgradient Descent for Joint Learning

For joint learning, we choose stochastic subgradient descent (SGD) approach to facilitate performing SGD on a per mention head basis. Next, we describe the weight update algorithm by defining the subgradients.

The partial subgradient w.r.t. mention head m for the head weight vector w_1 is given by

$$\begin{aligned}
&\nabla_{w_1,m} C(w_1,w_2) = \\
&\frac{1}{|D|n_d}(\nabla C_{local,m}(w_1) + \nabla C_{trans,m}(w_1)) + \lambda_1 w_1, \quad (2)
\end{aligned}$$

where

$$\begin{aligned}
\nabla C_{local,m}(w_1) &= (w_1^\top \varphi(m) - \Omega(m))\varphi(m), \\
\nabla C_{trans,m}(w_1) &= (w_1^\top \varphi(u') - \Omega(u'))\varphi(u')\Omega(m).
\end{aligned}$$

The partial subgradient w.r.t. mention head m for the coreference weight vector w_2 is given by

$$\begin{aligned}
&\nabla_{w_2,m} C(w_1,w_2) = \lambda_2 w_2 + \\
&\begin{cases}
\phi(u',m) - \phi(u'',m) & \text{if } \Omega(m) = 1, \\
\phi(u',m) & \text{if } \Omega(m) = 0 \text{ and } \Omega(u') = 1, \\
0 & \text{if } \Omega(m) = 0 \text{ and } \Omega(u') = 0.
\end{cases} \quad (3)
\end{aligned}$$

Here λ_1 and λ_2 are regularization coefficients which are tuned on the development set. To learn the mention head detection model, we consider two different parts of the gradient in expression (2). $\nabla C_{local,m}(w_1)$ is exactly the local gradient of mention head m while we add $\nabla C_{trans,m}(w_1)$ to represent

[5] More details can be found in Chang et al. (2013). The difference here is that we also consider the validity of mention heads using $\Omega(u), \Omega(m)$

the gradient for mention head u', the mention head chosen by the current best-left-linking model for m. This serves to maximize the margin between valid mention heads and invalid ones. As invalid mention heads will not be linked to any other mention head, ∇_{trans} is zero when m is invalid. When training the mention-pair coreference model, we only consider gradients when at least one of the two mention heads m, u' is valid, as shown in expression (3). When mention head m is valid ($\Omega(m) = 1$), the gradient is the same as local training for best-left-link of m (first condition in expression (3)). When m is not valid while u' is valid, we only demote the coreference link between them (second condition in expression (3)). We consider only the gradient from the regularization term when both m, u' are invalid.

As mentioned before, our framework can handle annotations with or without singletion mentions. When the gold data contains no singleton mentions, we have $\Omega(m) = 0$ for all singleton mention heads among mention head candidates. Then, our mention head detection model partly serves as a singleton head detector, and tries to reject singletons in the joint decisions with coreference. When the gold data contains singleton mentions, we have $\Omega(m) = 1$ for all valid singleton mention heads. Our mention head detection model then only learns to differentiate invalid mention heads from valid ones, and thus has the ability to preserve valid singleton heads.

Most of the head mentions proposed by the algorithms described in Sec. 3 are positive examples. We ensure a balanced training of the mention head detection model by adding sub-sampled invalid mention head candidates as negative examples. Specifically, after mention head candidate generation (described in Sec. 3), we train on a set of candidates with precision larger than 50%. We then use Illinois Chunker (Punyakanok and Roth, 2001)[6] to extract more noun phrases from the text and employ Collins head rules (Collins, 1999) to identify their heads. When these extracted heads do not overlap with gold mention heads, we treat them as negative examples.

We note that the aforementioned joint framework can take as input either complete mention candidates or mention head candidates. However, in this paper we only feed mention heads into it. Our experimental results support our intuition that this provides better results.

[6] http://cogcomp.cs.illinois.edu/page/software_view/Chunker

3 Mention Detection Modules

This section describes the module that generates our mention head candidates, and then how the mention heads are expanded to complete mentions.

3.1 Mention Head Candidate Generation

The goal of the mention head candidate generation process is to acquire candidates from multiple sources to ensure high recall, given that our joint framework acts as a filter and increases precision. We view the sources as independent components and merge all mention heads generated. A sequence labelling component and a named entity recognition component employ statistical learning methods. These are augmented by additional heads that we acquire from Wikipedia and a "known heads" resource, which we incorporate utilizing string matching algorithms.

3.1.1 Statistical Components

Sequence Labelling Component We use the following notations. Let $O =< o_1, o_2, \cdots, o_n >$ represent an input token sequence over an alphabet Ω. A mention is a substring of consecutive input tokens $m_{i,j} =< o_i, o_{i+1}, \cdots, o_j >$ for $1 \leq i \leq j \leq n$. We consider the positions of mentions in the text: two mentions with an identical sequence of tokens that differ in position are considered different mentions.

The sequence labeling component builds on the following assumption:

Assumption *Different mentions have different heads, and heads do not overlap with each other. That is, for each $m_{i,j}$, we have a corresponding head $h_{a,b}$ where $i \leq a \leq b \leq j$. Moreover, for another head $h_{a',b'}$, we have the satisfying condition $a - b' > 0$ or $b - a' < 0$ $\forall h_{a,b}, h_{a',b'}$.*

Based on this assumption, the problem of identifying mention heads is a sequential phrase identification problem, and we choose to employ the *BILOU*-representation as it has advantages over traditional *BIO*-representation, as shown, e.g. in Ratinov and Roth (2009). The *BILOU*-representation suggests learning classifiers that identify the **B**eginning, **I**nside and **L**ast tokens of multi-token chunks as well as **U**nit-length chunks. The problem is then transformed into a simple, but constrained, 5-class classification problem.

The *BILOU*-classifier shares all features with the mention head detection model described in Sec. 2.1 except for two: length of mention heads and NPMI over head boundary. For each instance, the feature

vector is sparse and we use sparse perceptron (Jackson and Craven, 1996) for supervised training. We also apply a two layer prediction aggregation. First, we apply a baseline *BILOU*-classifier, and then use the resulting predictions as additional features in a second level of inference to take interactions into account in an efficient manner. A similar technique has been applied in Ratinov and Roth (2009), and has shown favorable results over other "standard" sequential prediction models.

Named Entity Recognition Component We use existing tools to extract named entities as additional mention head candidates. We choose the state-of-the-art "Illinois Named Entity Tagger" package[7]. It uses distributional word representations that improve its generalization. This package gives the standard Person/Location/Organization/Misc labels and we take all output named entities as candidates.

3.1.2 Resource-Driven Matching Components

Wikipedia Many mention heads can be directly matched to a Wikipedia title. We get 4,045,764 Wikipedia titles from Wikipedia dumps and use all of them as potential mention heads. The Wikipedia matching component includes an efficient hashing algorithm implemented via a DJB2 hash function[8]. One important advantage of using Wikipedia is that it keeps updating. This component can contribute steadily to ensure a good coverage of mention heads. We first run this matching component on training documents and compute the precision of entries that appear in the text (the probability of appearing as mention heads). We then get the set of entries with precision higher than a threshold α, which is tuned on the development set using F1-score. We use them as candidates for mention head matching.

Known Head Some mention heads appear repeatedly in the text. To fully utilize the training data, we construct a known mention head candidate set and identify them in the test documents. To balance between recall and precision, we set a parameter $\beta > 0$ as a precision threshold and only allow those mention heads with precision larger than β on the training set. Please note that threshold β is also tuned on the development set using F1-score.

We also employ a simple word variation tolerance algorithm in our matching components, to generalize over small variations (plural/singular, etc.).

[7] http://cogcomp.cs.illinois.edu/page/software_view/NETagger

[8] http://www.cse.yorku.ca/~oz/hash.html

3.2 Mention Boundary Detection

Once the joint learning and inference process determines the set of mention heads (and their coreference chains), we extend the heads to complete mentions. Note that this process may not be necessary, since in many applications, the head clusters often provide enough information. However, for consistency with existing coreference resolution systems, we describe below how we expand the heads to complete mentions.

We learn a binary classifier to expand mentions, which determines if the mention head should include the token to its left and to its right. We follow the notations in Sec. 2.1. We construct positive examples as $(o_p, h_{a,b}, dir)$, $\forall m_{i,j}(h_{a,b})$. Here $p \in \{i, i+1, \cdots, a-1\} \cup \{b+1, b+2, \cdots, j\}$ and when $p = i, i+1, \cdots, a-1$, $dir = L$; when $p = b+1, b+2, \cdots, j$, $dir = R$. We construct negative examples as $(o_{i-1}, h_{a,b}, L)$ and $(o_{j+1}, h_{a,b}, R)$. Once trained, the binary classifier takes in the head, a token and the direction of the token relative to the head, and decides whether the token is inside or outside the mention corresponding to the head. At test time, this classifier is used around each confirmed head to determine the mention boundaries. The features used here are similar to the mention head detection model described in Sec. 2.1.

4 Experiments

We present experiments on the two standard coreference resolution datasets, ACE-2004 (NIST, 2004) and OntoNotes-5.0 (Hovy et al., 2006). Our approach results in a substantial reduction in the coreference performance gap between gold and predicted mentions, and significantly outperforms existing stat-of-the-art results on coreference resolution; in addition, it achieves significant performance improvement on MD for both datasets.

4.1 Experimental Setup

Datasets The ACE-2004 dataset contains 443 documents. We use a standard split of 268 training documents, 68 development documents, and 106 testing documents (Culotta et al., 2007; Bengtson and Roth, 2008). The OntoNotes-5.0 dataset, which is released for the CoNLL-2012 Shared Task (Pradhan et al., 2012), contains 3,145 annotated documents. These documents come from a wide range of sources which include newswire, bible, transcripts, magazines, and web blogs. We report results on the test documents for both datasets.

	MUC	B^3	$CEAF_e$	AVG
$Gold_{M/H}$	78.17	81.64	78.45	79.42
$Stanford_M$	63.89	70.33	70.21	68.14
$Predicted_M$	64.28	70.37	70.16	68.27
H-M-$Coref_M$	65.81	71.97	71.14	69.64
H-Joint-M_M	**67.28**	**73.06**	**73.25**	**71.20**
$Stanford_H$	70.28	73.93	73.04	72.42
$Predicted_H$	71.35	75.33	74.02	73.57
H-M-$Coref_H$	71.81	75.69	74.45	73.98
H-Joint-M_H	**72.74**	**76.69**	**75.18**	**74.87**

Table 2: **Performance of coreference resolution for all systems on the ACE-2004 dataset.** Subscripts ($_M$, $_H$) indicate evaluations on (mentions, mention heads) respectively. For gold mentions and mention heads, they yield the same performance for coreference. Our proposed *H-Joint-M* system achieves the highest performance. Parameters of our proposed system are tuned as $\alpha = 0.9$, $\beta = 0.8$, $\lambda_1 = 0.2$ and $\lambda_2 = 0.3$.

	MUC	B^3	$CEAF_e$	AVG
$Gold_{M/H}$	82.03	70.59	66.76	73.12
$Stanford_M$	64.62	51.89	48.23	54.91
$HotCoref_M$	70.74	58.37	55.47	61.53
$Berkeley_M$	71.24	58.71	55.18	61.71
$Predicted_M$	69.63	57.46	53.16	60.08
H-M-$Coref_M$	70.95	59.11	54.98	61.68
H-Joint-M_M	**72.22**	**60.50**	**56.37**	**63.03**
$Stanford_H$	68.53	56.68	52.36	59.19
$HotCoref_H$	72.94	60.27	57.53	63.58
$Berkeley_H$	73.05	60.39	57.43	63.62
$Predicted_H$	72.11	60.12	55.68	62.64
H-M-$Coref_H$	73.22	61.42	56.21	63.62
H-Joint-M_H	**74.83**	**62.77**	**57.93**	**65.18**

Table 3: **Performance of coreference resolution for all systems on the CoNLL-2012 dataset.** Subscripts ($_M$, $_H$) indicate evaluations on (mentions, mention heads) respectively. For gold mentions and mention heads, they yield the same performance for coreference. Our proposed *H-Joint-M* system achieves the highest performance. Parameters of our proposed system are tuned as $\alpha = 0.9$, $\beta = 0.9$, $\lambda_1 = 0.25$ and $\lambda_2 = 0.2$.

The ACE-2004 dataset is annotated with both mention and mention heads, while the OntoNotes-5.0 dataset only has mention annotations. Therefore, we preprocess Ontonote-5.0 to derive mention heads using Collins head rules (Collins, 1999) with gold constituency parsing information and gold named entity information. The parsing information[9] is only needed to generate training data for the mention head candidate generator and named entities are directly set as heads. We set these extracted heads as gold, which enables us to train the two layer *BILOU*-classifier described in Sec. 3.1.1. The non-overlapping mention head assumption in Sec. 3.1.1 can be verified empirically on both ACE-2004 and OntoNotes-5.0 datasets.

Baseline Systems We choose three publicly available state-of-the-art end-to-end coreference systems as our baselines: *Stanford* system (Lee et al., 2011), *Berkeley* system (Durrett and Klein, 2014) and *HOTCoref* system (Björkelund and Kuhn, 2014).

Developed Systems Our developed system is built on the work by Chang et al. (2013), using Constrained Latent Left-Linking Model (CL^3M) as our mention-pair coreference model in the joint framework[10]. When the CL^3M coreference system uses gold mentions or heads, we call the system *Gold*; when it uses predicted mentions or heads, we call the system *Predicted*. The mention head candidate generation module along with mention boundary detection module can be grouped together to form a complete mention detection system, and we call it *H-M-MD*. We can feed the predicted mentions from *H-M-MD* directly into the mention-pair coreference model that we implemented, resulting in a traditional pipelined end-to-end coreference system, namely *H-M-Coref*. We name our new proposed end-to-end coreference resolution system incorporating both the mention head candidate generation module and the joint framework as *H-Joint-M*.

Evaluation Metrics We compare all systems using three popular metrics for coreference resolution: MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and Entity-based CEAF ($CEAF_e$) (Luo, 2005). We use the average F1 scores (AVG) of these three metrics as the main metric for comparison. We use the v7.0 scorer provided by CoNLL-2012 Shared Task[11]. We also evaluate the mention detection performance based on precision, recall and F1 score. As mention heads are important for both mention detection and coreference resolution, we also report results evaluated on mention heads.

4.2 Performance for Coreference Resolution

Performance of coreference resolution for all systems on the ACE-2004 and CoNLL-2012 datasets is shown in Table 2 and Table 3 respectively.[12] These results show that our developed system *H-Joint-M*

[9] No parsing information is needed at evaluation time.

[10] We use Gurobi v5.0.1 as our ILP solver.

[11] The latest scorer is version v8.01, but MUC, B^3, $CEAF_e$ and CoNLL average scores are not changed. For evaluation on ACE-2004, we convert the system output and gold annotations into CoNLL format.

[12] We do not provide results from *Berkeley* and *HOTCoref* on ACE-2004 dataset as they do not directly support ACE input. Results for *HOTCoref* are slightly different from the results reported in Björkelund and Kuhn (2014). For *Berkeley* system, we use the reported results from Durrett and Klein (2014).

shows significant improvement on all metrics for both datasets. Existing systems only report results on mentions. Here, we also show their performance evaluated on mention heads. When evaluated on mention heads rather than mentions[13], we can always expect a performance increase for all systems on both datasets. Even though evaluating on mentions is more common in the literature, it is often enough to identify just mention heads in coreference chains (as shown in the example from Sec. 1). *H-M-Coref* can already bring substantial performance improvement, which indicates that it is helpful for coreference to just identify high quality mention heads. Our proposed *H-Joint-M* system outperforms all baselines and achieves the best results reported so far.

4.3 Performance for Mention Detection

The performance of mention detection for all systems on the ACE-2004 and CoNLL-2012 datasets is shown in Table 4. These results show that our developed system exhibits significant improvement on precision and recall for both datasets. *H-M-MD* mainly improves on recall, indicating, as expected, that the mention head candidate generation module ensures high recall on mention heads. *H-Joint-M* mainly improves on precision, indicating, as expected, that the joint framework correctly rejects many of the invalid mention head candidates during joint inference. Our joint model can adapt to annotations with or without singleton mentions. Based on training data, our system has the ability to preserve true singleton mentions in ACE while rejecting many singleton mentions in OntoNotes[14]. Note that we have better mention detection results on ACE-2004 dataset than on OntoNotes-5.0 dataset. We believe that this is due to the fact that extracting mention heads in the OntoNotes dataset is somewhat noisy.

4.4 Analysis of Performance Improvement

The improvement of our *H-Joint-M* system is due to two distinct but related modules: the mention head candidate generation module ("Head") and the joint learning and inference framework ("Joint").[15] We evaluate the effect of these two modules in terms of *Mention Detection Error Reduction* (MDER) and *Performance Gap Reduction* (PGR) for coreference. MDER is computed as the ratio of performance improvement for mention detection over the original mention detection error rate, while PGR is computed as the ratio of performance improvement for coreference over the performance gap for coreference. Results on the ACE-2004 and CoNLL-2012 datasets are shown in Table 5.[16]

The mention head candidate generation module has a bigger impact on MDER compared to the joint framework. However, they both have the same level of positive effects on PGR for coreference resolution. On both datasets, we achieve more than 20% performance gap reduction for coreference.

Systems	Precision	Recall	F1-score
ACE-2004			
Predicted$_M$	75.11	73.03	74.06
H-M-MD$_M$	77.45	92.97	83.90
H-Joint-M$_M$	85.34	91.73	**88.42**
Predicted$_H$	76.84	86.99	79.87
H-M-MD$_H$	80.82	93.45	86.68
H-Joint-M$_H$	88.85	92.27	**90.53**
CoNLL-2012			
Predicted$_M$	65.28	63.41	64.33
H-M-MD$_M$	70.09	76.72	73.26
H-Joint-M$_M$	78.51	75.52	**76.99**
Predicted$_H$	76.38	74.02	75.18
H-M-MD$_H$	77.73	83.99	80.74
H-Joint-M$_H$	85.07	82.31	**83.67**

Table 4: **Performance of mention detection for all systems on the ACE-2004 and CoNLL-2012 datasets.** Subscripts ($_M$, $_H$) indicate evaluations on (mentions, mention heads) respectively. Our proposed *H-Joint-M* system dramatically improves the MD performance.

5 Related Work

Coreference resolution has been extensively studied, with several state-of-the-art approaches addressing this task (Lee et al., 2011; Durrett and Klein, 2013; Björkelund and Kuhn, 2014; Song et al., 2012). Many of the early rule-based systems like Hobbs (1978) and Lappin and Leass (1994) gained considerable popularity. The early designs were easy to understand and the rules were designed manually. Machine learning approaches were introduced in many works (Connolly et al., 1997; Ng and

[13]Here, we treat mention heads as mentions. Thus, in the evaluation script, we set the boundary of a mention to be the boundary of its correponding mention head.

[14]Please note that when evaluating on OntoNotes, we eventually remove all singleton mentions from the output.

[15]"Joint" rows are computed as "H-Joint-M" rows minus "Head" rows. They reflect the contribution of the joint framework to mention detection (by rejecting some mention heads).

[16]We use bootstrapping resampling (10 times from the test data) with signed rank test. All the improvements shown are statistically significant.

ACE-2004	MDER	PGR(AVG)
Head_M	37.93	12.29
Joint_M	17.43	13.99
H-Joint-M_M	55.36	26.28
Head_H	34.00	7.01
Joint_H	19.22	15.21
H-Joint-M_H	53.22	22.22
CoNLL-2012	MDER	PGR(AVG)
Head_M	25.04	12.16
Joint_M	10.45	10.44
H-Joint-M_M	35.49	22.60
Head_H	22.40	10.58
Joint_H	11.81	13.75
H-Joint-M_H	34.21	24.33

Table 5: **Analysis of performance improvement in terms of *Mention Detection Error Reduction* (MDER) and *Performance Gap Reduction* (PGR) for coreference resolution on the ACE-2004 and CoNLL-2012 datasets.** "Head" represents the mention head candidate generation module, "Joint" represents the joint learning and inference framework, and "H-Joint-M" indicates the end-to-end system.

Cardie, 2002; Bengtson and Roth, 2008; Soon et al., 2001). The introduction of ILP methods has influenced the coreference area too (Chang et al., 2011; Denis and Baldridge, 2007). In this paper, we use the Constrained Latent Left-Linking Model (CL^3M) described in Chang et al. (2013) in our experiments.

The task of mention detection is closely related to *Named Entity Recognition* (NER). Punyakanok and Roth (2001) thoroughly study phrase identification in sentences and propose three different general approaches. They aim to learn several different local classifiers and combine them to optimally satisfy some global constraints. Cardie and Pierce (1998) propose to select certain rules based on a given corpus, to identify base noun phrases. However, the phrases detected are not necessarily mentions that we need to discover. Ratinov and Roth (2009) present detailed studies on the task of named entity recognition, which discusses and compares different methods on multiple aspects including chunk representation, inference method, utility of non-local features, and integration of external knowledge. NER can be regarded as a sequential labeling problem, which can be modeled by several proposed models, e.g. Hidden Markov Model (Rabiner, 1989) or Conditional Random Fields (Sarawagi and Cohen, 2004). The typical BIO representation was introduced in Ramshaw and Marcus (1995); OC representations were introduced in Church (1988), while Finkel and Manning (2009) further study nested named entity recognition, which employs a tree structure as a representation of identifying named entities within other named entities.

The most relevant study on mentions in the context of coreference was done in Recasens et al. (2013); this work studies distinguishing single mentions from coreferent mentions. Our joint framework provides similar insights, where the added mention decision variable partly reflects if the mention is singleton or not.

Several recent works suggest studying coreference jointly with other tasks. Lee et al. (2012) model entity coreference and event coreference jointly; Durrett and Klein (2014) consider joint coreference and entity-linking. The work closest to ours is that of Lassalle and Denis (2015), which studies a joint anaphoricity detection and coreference resolution framework. While their inference objective is similar, their work assumes gold mentions are given and thus their modeling is very different.

6 Conclusion

This paper proposes a joint inference approach to the end-to-end coreference resolution problem. By moving to identify mention heads rather than mentions, and by developing an ILP-based, joint, online learning and inference approach, we close a significant fraction of the existing gap between coreference systems' performance on gold mentions and their performance on raw data. At the same time, we show substantial improvements in mention detection. We believe that our approach will generalize well to many other NLP problems, where the performance on raw data (the result that really matters) is still significantly lower than the performance on gold data.

Acknowledgments

This work is partly supported by NSF grant #SMA 12-09359 and by DARPA under agreement number FA8750-13-2-0008. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

References

A. Bagga and B. Baldwin. 1998. Algorithms for scoring coreference chains. In *In The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*.

E. Bengtson and D. Roth. 2008. Understanding the value of features for coreference resolution. In *EMNLP*.

A. Björkelund and J Kuhn. 2014. Learning structured Perceptrons for coreference resolution with latent antecedents and non-local features. In *ACL*.

C. Cardie and D. Pierce. 1998. Error-driven pruning of Treebanks grammars for base noun phrase identification. In *Proceedings of ACL-98*.

K. Chang, R. Samdani, A. Rozovskaya, N. Rizzolo, M. Sammons, and D. Roth. 2011. Inference protocols for coreference resolution. In *CoNLL*.

K.-W. Chang, R. Samdani, and D. Roth. 2013. A constrained latent variable model for coreference resolution. In *EMNLP*.

K. Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *ANLP*.

M. Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing.* Ph.D. thesis, Computer Science Department, University of Pennsylvenia.

D. Connolly, J. Burger, and D. Day. 1997. A machine learning approach to anaphoric reference. In *New Methods in Language Processing*.

A. Culotta, M. Wick, and A. McCallum. 2007. First-order probabilistic models for coreference resolution. In *NAACL*.

P. Denis and J. Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *NAACL*.

G. Durrett and D. Klein. 2013. Easy victories and uphill battles in coreference resolution. In *EMNLP*.

G. Durrett and D. Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking.

J. Finkel and C. Manning. 2009. Nested named entity recognition. In *EMNLP*.

J. R. Hobbs. 1978. Resolving pronoun references. *Lingua*.

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of HLT/NAACL*.

J. Jackson and M. Craven. 1996. Learning sparse perceptrons. *Proceedings of the 1996 Advances in Neural Information Processing Systems*.

S. Lappin and H. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational linguistics*.

E. Lassalle and P. Denis. 2015. Joint anaphoricity detection and coreference resolution with constrained latent structures.

H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the CoNLL-2011 Shared Task*.

H. Lee, M. Recasens, A. Chang, M. Surdeanu, and D. Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *EMNLP*.

X. Luo. 2005. On coreference resolution performance metrics. In *EMNLP*.

V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *ACL*.

NIST. 2004. The ACE evaluation plan.

S. Pradhan, A. Moschitti, N. Xue, O. Uryupina, and Y. Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *CoNLL*.

V. Punyakanok and D. Roth. 2001. The use of classifiers in sequential inference. In *NIPS*.

L. R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*.

L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Annual Workshop on Very Large Corpora*.

L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*.

M. Recasens, M.-C. de Marneffe, and C. Potts. 2013. The life and death of discourse entities: Identifying singleton mentions. In *NAACL*.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *CoNLL*.

S. Sarawagi and W. Cohen. 2004. Semi-Markov conditional random fields for information extraction. In *NIPS*.

Y. Song, J. Jiang, W.-X. Zhao, S. Li, and H. Wang. 2012. Joint learning for coreference resolution with markov logic. In *Proceedings of the 2012 Joint Conference of EMNLP-CoNLL*.

W. M. Soon, H. T. Ng, and D. C. Y. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*

M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th conference on Message understanding*.