Combining the Best of Two Worlds:
A Hybrid Approach to Multilingual Coreference Resolution

Chen Chen and Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{yzcchen, vince}@hlt.utdallas.edu

Abstract
We describe our system for the CoNLL-2012 shared task, which seeks to model coreference in OntoNotes for English, Chinese, and Arabic. We adopt a hybrid approach to coreference resolution, which combines the strengths of rule-based methods and learning-based methods. Our official combined score over all three languages is 56.35. In particular, our score on the Chinese test set is the best among the participating teams.

1 Introduction
The CoNLL-2012 shared task extends last year's task on coreference resolution from a monolingual to a multilingual setting (Pradhan et al., 2012). Unlike the SemEval-2010 shared task on Coreference Resolution in Multiple Languages (Recasens et al., 2010), which focuses on coreference resolution in European languages, the CoNLL shared task is arguably more challenging: it focuses on three languages that come from very different language families, namely English, Chinese, and Arabic.

We designed a system for resolving references in all three languages. Specifically, we participated in four tracks: the closed track for all three languages, and the open track for Chinese. In comparison to last year's participating systems, our resolver has two distinguishing characteristics. First, unlike last year's resolvers, which adopted either a rule-based method or a learning-based method, we adopt a hybrid approach to coreference resolution, attempting to combine the strengths of both methods. Second, while last year's resolvers did not exploit genre-specific information, we optimize our system's parameters with respect to each genre.

Our decision to adopt a hybrid approach is motivated by the observation that rule-based methods and learning-based methods each have their unique strengths. As shown by the Stanford coreference resolver (Lee et al., 2011), the winner of last year's shared task, many coreference relations in OntoNotes can be identified using a fairly small set of simple hand-crafted rules. On the other hand, our prior work on machine learning for coreference resolution suggests that coreference-annotated data can be profitably exploited to (1) induce lexical features (Rahman and Ng, 2011a, 2011b) and (2) optimize system parameters with respect to the desired coreference evaluation measure (Ng, 2004, 2009).

Our system employs a fairly standard architecture, performing mention detection prior to coreference resolution. As we will see, however, the parameters of these two components are optimized jointly with respect to the desired evaluation measure.

In the rest of this paper, we describe the mention detection component (Section 2) and the coreference resolution component (Section 3), show how their parameters are jointly optimized (Section 4), and present evaluation results on the development set and the official test set (Section 5).

2 Mention Detection
To build a mention detector that strikes a relatively good balance between precision and recall, we employ a two-step approach. First, in the extraction step, we identify named entities (NEs) and employ language-specific heuristics to extract mentions...
from syntactic parse trees, aiming to increase our upper bound on recall as much as possible. Then, in the pruning step, we aim to improve precision by employing both language-specific heuristic pruning and language-independent learning-based pruning. Section 2.1 describes the language-specific heuristics for extraction and pruning, and Section 2.2 describes our learning-based pruning method.

### 2.1 Heuristic Extraction and Pruning

**English.** During extraction, we create a candidate mention from a contiguous text span \( s \) if (1) \( s \) is a PRP or an NP in a syntactic parse tree; or (2) \( s \) corresponds to a NE that is not a PERCENT, MONEY, QUANTITY or CARDINAL. During pruning, we remove a candidate mention \( m_k \) if (1) \( m_k \) is embedded within a larger mention \( m_j \) such that \( m_j \) and \( m_k \) have the same head, where the head of a mention is detected using Collins's (1999) rules; (2) \( m_k \) has a quantifier or a partitive modifier; or (3) \( m_k \) is a singular common NP, with the exception that we retain mentions related to time (e.g., "today").

**Chinese.** Similar to English mention extraction, we create Chinese mentions from all NP and QP nodes in syntactic parse trees. During pruning, we remove a candidate mention \( m_k \) if (1) \( m_k \) is embedded within a larger mention \( m_j \) such that \( m_j \) and \( m_k \) have the same head, except if \( m_j \) and \( m_k \) appear in a newswire document since, unlike other document annotations, Chinese newswire document annotations do consider such pairs coreferent; (2) \( m_k \) is a NE that is a PERCENT, MONEY, QUANTITY and CARDINAL; or (3) \( m_k \) is an interrogative pronoun such as "什么 [what]", "哪儿 [where]."

**Arabic.** We employ as candidate mentions all the NPs extracted from syntactic parse trees, removing those that are PERCENT, MONEY, QUANTITY or CARDINAL.

### 2.2 Learning-Based Pruning

While the heuristic pruning method identifies candidate mentions, it cannot determine which candidate mentions are likely to be coreferent. To improve pruning (and hence the precision of mention detection), we employ learning-based pruning, where we employ the training data to identify and subsequently discard those candidate mentions that are not likely to be coreferent with other mentions.

| Language | Recall | Precision | F-Score |
|----------|--------|-----------|---------|
| English  | 88.59  | 40.56     | 55.64   |
| Chinese  | 85.74  | 42.52     | 56.85   |
| Arabic   | 81.49  | 21.29     | 33.76   |

Table 1: Mention detection results on the development set obtained prior to coreference resolution.

Specifically, for each mention \( m_k \) in the test set that survives heuristic pruning, we compute its mention coreference probability, which indicates the likelihood that the head noun of \( m_k \) is coreferent with another mention. If this probability does not exceed a certain threshold \( t_C \), we will remove \( m_k \) from the list of candidate mentions. Section 4 discusses how \( t_C \) is jointly learned with the parameters of the coreference resolution component to optimize the coreference evaluation measure.

We estimate the mention coreference probability of \( m_k \) from the training data. Specifically, since only non-singleton mentions are annotated in OntoNotes, we can compute this probability as the number of times \( m_k \)'s head noun is annotated (as a gold mention) divided by the total number of times \( m_k \)'s head noun appears. If \( m_k \)'s head noun does not appear in the training set, we set its coreference probability to 1, meaning that we let it pass through the filter. In other words, we try to be conservative and do not filter any mention for which we cannot compute the coreference probability.

Table 1 shows the mention detection results of the three languages on the development set after heuristic extraction and pruning but prior to learning-based pruning and coreference resolution.

### 3 Coreference Resolution

Like the mention detection component, our coreference resolution component employs heuristics and machine learning. More specifically, we employ Stanford's multi-pass sieve approach (Lee et al., 2011) for heuristic coreference resolution, but since most of these sieves are unlexicalized, we seek to improve the multi-pass sieve approach by incorporating lexical information using machine learning techniques. As we will see below, while different sieves are employed for different languages, the way we incorporate lexical information into the sieve approach is the same for all languages.
3.1 The Multi-Pass Sieve Approach
A sieve is composed of one or more heuristic rules. Each rule extracts a coreference relation between two mentions based on one or more conditions. For example, one rule in Stanford’s discourse processing sieve posits two mentions as coreferent if two conditions are satisfied: (1) they are both pronouns; and (2) they are produced by the same speaker.

Sieves are ordered by their precision, with the most precise sieve appearing first. To resolve a set of mentions in a document, the resolver makes multiple passes over them: in the \(i\)-th pass, it attempts to use only the rules in the \(i\)-th sieve to find an antecedent for each mention \(m_k\). Specifically, when searching for an antecedent for \(m_k\), its candidate antecedents are visited in an order determined by their positions in the associated parse tree (Haghighi and Klein, 2009). The partial clustering of the mentions created in the \(i\)-th pass is then passed to the \(i+1\)-th pass. Hence, later passes can exploit the information computed by previous passes, but a coreference link established earlier cannot be overridden later.

3.2 The Sieves
3.2.1 Sieves for English
Our sieves for English are modeled after those employed by the Stanford resolver (Lee et al., 2011), which is composed of 12 sieves. Since we participated in the closed track, we re-implemented the 10 sieves that do not exploit external knowledge sources. These 10 sieves are listed under the "English" column in Table 2. Specifically, we leave out the Alias sieve and the Lexical Chain sieve, which compute semantic similarity using information extracted from WordNet, Wikipedia, and Freebase.

3.2.2 Sieves for Chinese
Recall that for Chinese we participated in both the closed track and the open track. The sieves we employ for both tracks are the same, except that we use NE information to improve some of the sieves in the system for the open track. To obtain automatic NE annotations, we employ a NE model that we trained on the gold NE annotations in the training data.

Table 2: Sieves for English and Chinese (listed in the order in which they are applied).

The Chinese resolver is composed of 9 sieves, as shown under the "Chinese" column of Table 2. These sieves are implemented in essentially the same way as their English counterparts except for a few of them, which are modified in order to account for some characteristics specific to Chinese or the Chinese coreference annotations. As described in detail below, we introduce a new sieve, the Chinese Head Match sieve, and modify two existing sieves, the Precise Constructs sieve, and the Pronouns sieve.

1. **Chinese Head Match sieve**: Recall from Section 2 that the Chinese newswire articles were coreference-annotated in such a way that a mention and its embedding mention can be coreferent if they have the same head. To identify these coreference relations, we employ the Same Head sieve, which posits two mentions \(m_j\) and \(m_k\) as coreferent if they have the same head and \(m_k\) is embedded within \(m_j\). There is an exception to this rule, however: if \(m_j\) is a coordinated NP composed of two or more base NPs, and \(m_k\) is just one of these base NPs, the two mentions will not be considered coreferent (e.g., 查尔斯和戴安娜 [Charles and Diana] and 戴安娜 [Diana]).

2. **Precise Constructs sieve**: Recall from Lee et al. (2011) that the Precise Constructs sieve posits two mentions as coreferent based on information such as whether one is an acronym of the other and whether they form an appositive or copular construction. We incorporate additional rules to this sieve to handle specific cases of abbreviations in Chinese: (a) Abbreviation of foreign person names, e.g., 萨达姆·侯赛因 [Saddam Hussein] and 萨达姆 [Saddam]. (b) Abbreviation of Chinese person names, e.g.,
We can then apply these word lists to determine the grammatical attribute values of mentions in a test text. Due to the small size of these word lists, and with the goal of improving precision, we consider two mentions to be grammatically incompatible if for one of these three attributes, one mention has an Unknown value whereas the other has a known value.

As seen in Table 2, our Chinese resolver does not have the Relaxed String Match sieve, unlike its English counterpart. Recall that this sieve marks two mentions as coreferent if the strings after dropping the text following their head words are identical (e.g., Michael Wolf, and Michael Wolf, a contributing editor for "New York"). Since person names in Chinese are almost always composed of a single word and that heads are seldom followed by other words in Chinese, we believe that Relaxed Head Match will not help identify Chinese coreference relations. As noted before, cases of Chinese person name abbreviation will be handled by the Precise Constructs sieve.

3.2.3 Sieves for Arabic
We only employ one sieve for Arabic, the exact match sieve. While we experimented with additional sieves such as the Head Match sieve and the Pronouns sieve, we ended up not employing them because they do not yield better results.

3.3 Incorporating Lexical Information
As mentioned before, we improve the sieve approach by incorporating lexical information.

To exploit lexical information, we first compute lexical probabilities. Specifically, for each pair of mentions $m_j$ and $m_k$ in a test text, we first compute two probabilities: (1) the string-pair probability (SP-Prob), which is the probability that the strings of the two mentions, $s_j$ and $s_k$, are coreferent; and (2) the head-pair probability (HP-Prob), which is the probability that the head nouns of the two mentions, $h_j$ and $h_k$, are coreferent. For better probability estimation, we preprocess the training data and the two mentions by (1) downcasing (but not stemming) each English word, and (2) replacing each Arabic word $w$ by a string formed by concatenating $w$ with its lemmatized form, its Buckwalter form, and its vocalized Buckwalter form. Note that SP-Prob($m_j,m_k$) (HP-
\( \text{Prob}(m_j, m_k) \) is undefined if one or both of \( s_j (h_j) \) and \( s_k (h_k) \) do not appear in the training set.

Next, we exploit these lexical probabilities to improve the resolution of \( m_j \) and \( m_k \) by presenting two extensions to the sieve approach. The first extension aims to improve the precision of the sieve approach. Specifically, before applying any sieve, we check whether \( \text{SP-Prob}(m_j, m_k) \leq t_{\text{SPL}} \) or \( \text{HP-Prob}(m_j, m_k) \leq t_{\text{HPL}} \) for some thresholds \( t_{\text{SPL}} \) and \( t_{\text{HPL}} \). If so, our resolver will bypass all of the sieves and simply posit \( m_j \) and \( m_k \) as not coreferent. In essence, we use the lexical probabilities to improve precision, specifically by positing two mentions as not coreferent if there is "sufficient" information in the training data for us to make this decision. Note that if one of the lexical probabilities (say \( \text{SP-Prob}(m_j, m_k) \)) is undefined, we only check whether the condition on the other probability (in this case \( \text{HP}(m_j, m_k) \leq t_{\text{HPL}} \)) is satisfied. If both of them are undefined, this pair of mentions will survive this filter and be processed by the sieve pipeline.

The second extension, on the other hand, aims to improve recall. Specifically, we create a new sieve, the Lexical Pair sieve, which we add to the end of the sieve pipeline and which posits two mentions \( m_j \) and \( m_k \) as coreferent if \( \text{SP-Prob}(m_j, m_k) \geq t_{\text{SPU}} \) or \( \text{HP-Prob}(m_j, m_k) \geq t_{\text{HPU}} \). In essence, we use the lexical probabilities to improve recall, specifically by positing two mentions as coreferent if there is "sufficient" information in the training data for us to make this decision. Similar to the first extension, if one of the lexical probabilities (say \( \text{SP-Prob}(m_j, m_k) \)) is undefined, we only check whether the condition on the other probability (in this case \( \text{HP}(m_j, m_k) \geq t_{\text{HPU}} \)) is satisfied. If both of them are undefined, the Lexical Pair sieve will not process this pair of mentions.

The four thresholds, \( t_{\text{SPL}}, t_{\text{HPL}}, t_{\text{SPU}}, \) and \( t_{\text{HPU}} \), will be tuned to optimize coreference performance on the development set.

### 4 Parameter Estimation

As discussed before, we learn the system parameters to optimize coreference performance (which, for the shared task, is \( U_{\text{avg}} \), the unweighted average of the three commonly-used evaluation measures, MUC, B^3, and CEAF_e) on the development set. Our system has two sets of tunable parameters. So far, we have seen one set of parameters, namely the five lexical probability thresholds, \( t_C, t_{\text{SPL}}, t_{\text{HPL}}, t_{\text{SPU}}, \) and \( t_{\text{HPU}} \). The second set of parameters contains the rule relaxation parameters. Recall that each rule in a sieve may be composed of one or more conditions. We associate with condition \( i \) a parameter \( \lambda_i \), which is a binary value that controls whether condition \( i \) should be removed or not. In particular, if \( \lambda_i = 0 \), condition \( i \) will be dropped from the corresponding rule. The motivation behind having the rule relaxation parameters should be clear: they allow us to optimize the hand-crafted rules using machine learning. This section presents two algorithms for tuning these two sets of parameters on the development set.

Before discussing the parameter estimation algorithms, recall from the introduction that one of the distinguishing features of our approach is that we build genre-specific resolvers. In other words, for each genre of each language, we (1) learn the lexical probabilities from the corresponding training set; (2) obtain optimal parameter values \( \Theta_1 \) and \( \Theta_2 \) for the development set using parameter estimation algorithms 1 and 2 respectively; and (3) among \( \Theta_1 \) and \( \Theta_2 \), take the one that yields better performance on the development set to be the final set of parameter estimates for the resolver.

#### Parameter estimation algorithm 1.

This algorithm learns the two sets of parameters in a sequential fashion. Specifically, it first tunes the lexical probability thresholds, assuming that all the rule relaxation parameters are set to one. To tune the five probability thresholds, we try all possible combinations of the five probability thresholds and select the combination that yields the best performance on the development set. To ensure computational tractability, we allow each threshold to have the following possible values. For \( t_C \), the possible values are \(-0.1, 0, 0.05, 0.1, \ldots, 0.3; \) for \( t_{\text{SPL}} \) and \( t_{\text{HPL}} \), the possible values are \(-0.1, 0, 0.05, 0.15, \ldots, 0.45; \) and for \( t_{\text{SPU}} \) and \( t_{\text{HPU}} \), the possible values are \( 0.55, 0.65, \ldots, 0.95, 1.0 \) and \( 1.1 \). Note that the two threshold values \(-0.1 \) and \( 1.1 \) render a probability threshold useless. For example, if \( t_C = -0.1 \), that means all mentions will survive learning-based pruning in the mention detection component. As another example, if \( t_{\text{SPU}} \) and \( t_{\text{HPU}} \) are both \( 1.1 \), it means that the String Pair sieve
will be useless because it will not posit any pair of
tions as coreferent.

Given the optimal set of probability thresholds, we
tune the rule relaxation parameters. To do so, we ap-
ply the backward elimination feature selection al-
gorithm, viewing each condition as a feature that can be
removed from the "feature set". Specifically, all the
parameters are initially set to one, meaning that all
the conditions are initially present. In each iteration
of backward elimination, we identify the condition
whose removal yields the highest score on the de-
velopment set and remove it from the feature set. We
repeat this process until all conditions are removed,
and identify the subset of the conditions that yields
the best score on the development set.

Parameter estimation algorithm 2. In this algo-
rithm, we estimate the two sets of parameters in an
interleaved, iterative fashion, where in each iter-
ation, we optimize exactly one parameter from one
of the two sets. More specifically, (1) in iteration
$2n$, we optimize the \((n \mod 5)\)-th lexical proba-
bility threshold while keeping the remaining param-
eters constant; and (2) in iteration $2n+1$, we optimize
the \((n \mod m)\)-th rule relaxation parameter while
keeping the remaining parameters constant, where
\(n = 1, 2, \ldots,\) and \(m\) is the number of rule relax-
ation parameters. When optimizing a parameter in a
given iteration, the algorithm selects the value that,
when used in combination with the current values of
the remaining parameters, optimizes the $U_{avg}$ value
on the development set. We begin the algorithm by
initializing all the rule relaxation parameters to one;
\(t_C, t_{SPL}\) and \(t_{HPL}\) to $-0.1$; and \(t_{SPU}\) and \(t_{HPU}\)
to $1.1$. This parameter initialization is equivalent to
the configuration where we employ all and only the
hand-crafted rules as sieves and do not apply learn-
ing to perform any sort of optimization at all.

5 Results and Discussion

The results of our Full coreference resolver on the
development set with optimal parameter values are
shown in Table 3. As we can see, both the men-
tion detection results and the coreference results (ob-
tained via MUC, $B^3$, and CEAF$_c$) are expressed in
terms of recall ($R$), precision ($P$), and F-measure ($F$).
In addition, to better understand the role played by
the two sets of system parameters, we performed ab-
limation experiments, showing for each language-track
combination the results obtained without tuning (1)
the rule relaxation parameters ($-\lambda_i$’s); (2) the proba-
bility thresholds ($-t_j$’s); and (3) any of these param-
eters ($-\lambda_i$’s & $t_j$’s). Note that (1) we do not have any
rule relaxation parameters for the Arabic resolver
owing to its simplicity; and (2) for comparison pur-
poses, we show the results of the Stanford resolver
for English in the row labeled "Lee et al. (2011)".

A few points regarding the results in Table 3 de-
serve mention. First, these mention detection re-
results are different from those shown in Table 1: here,
the scores are computed over the mentions that ap-
pear in the non-singleton clusters in the coreference
partitions produced by a resolver. Second, our re-
implementation of the Stanford resolver is as good
as the original one. Third, parameter tuning is com-
paratively less effective for Chinese, presumably be-
cause we spent more time on engineering the sieves
for Chinese than for the other languages. Fourth,
our score on Arabic is the lowest among the three
languages, primarily because Arabic is highly inflec-
tional and we have little linguistic knowledge of the
language to design effective sieves. Finally, these
results and our official test set results (Table 4), as
well as our supplementary evaluation results on the
test set obtained using gold mention boundaries (Ta-
ble 5) and gold mentions (Table 6), exhibit similar
performance trends.

Table 7 shows the optimal parameter values ob-
tained for the Full resolver on the development set.
Since there are multiple genres for English and Chi-
inese, we show in the table the probability thresholds
averaged over all the genres and the corresponding
standard deviation values. For the rule relaxation
parameters, among the 36 conditions in the English
sieves and the 61 conditions in the Chinese sieves,
we show the number of conditions being removed
(when averaged over all the genres) and the corre-
ponding standard deviation values. Overall, differ-
ent conditions were removed for different genres.

To get a better sense of the usefulness of the
probability thresholds, we show in Tables 8 and
9 some development set examples of cor-
rectly and incorrectly identified/pruned mentions
and coreferent/non-coreferent pairs for English and
Chinese, respectively. Note that no Chinese ex-
amples for $t_C$ are shown, since its tuned value cor-
Table 3: Results on the development set with optimal parameter values.

| Language | Track | System | Mention Detect. | MUC | B-CUBED | CEAFe | Avg |
|----------|-------|--------|-----------------|-----|---------|-------|-----|
|          |       |        | R   | P   | F   | R   | P   | F   | R   | P   | F   | R   | P   | F   | R | P | F | Avg |
| English  | Closed | Full   | 74.8 | 75.6 | 75.2 | 65.6 | 67.3 | 66.4 | 69.1 | 74.7 | 71.8 | 49.8 | 47.9 | 48.8 | 62.3 |
|          |       | − λ's | 75.2 | 73.4 | 74.3 | 64.6 | 65.8 | 65.2 | 68.5 | 74.1 | 71.2 | 48.8 | 47.6 | 48.2 | 61.5 |
|          |       | − t_j's | 76.4 | 73.0 | 74.7 | 65.1 | 65.3 | 65.2 | 68.6 | 73.8 | 71.1 | 48.6 | 48.3 | 48.4 | 61.6 |
|          |       | − λ's & t_j's | 75.2 | 72.8 | 74.0 | 64.2 | 64.8 | 64.5 | 68.0 | 73.4 | 70.6 | 47.8 | 47.1 | 47.5 | 60.8 |
|          |       | Lee et al. (2011) | 74.1 | 72.5 | 73.3 | 64.3 | 64.9 | 64.6 | 68.2 | 73.1 | 70.6 | 47.0 | 46.3 | 46.7 | 60.6 |
| Chinese  | Closed | Full   | 72.2 | 72.7 | 72.4 | 62.4 | 65.8 | 64.1 | 70.8 | 77.7 | 74.1 | 52.3 | 48.9 | 50.5 | 62.9 |
|          |       | − λ's | 71.3 | 72.8 | 71.9 | 61.8 | 66.7 | 64.2 | 70.2 | 78.2 | 74.0 | 52.2 | 47.6 | 49.9 | 62.6 |
|          |       | − t_j's | 72.7 | 71.1 | 71.9 | 62.3 | 64.8 | 63.5 | 70.7 | 77.1 | 73.8 | 51.2 | 48.8 | 50.0 | 62.4 |
|          |       | − λ's & t_j's | 71.7 | 71.4 | 71.5 | 61.5 | 65.1 | 63.3 | 70.0 | 77.6 | 73.6 | 51.3 | 47.9 | 49.5 | 62.1 |
| Arabic   | Closed | Full   | 56.2 | 64.5 | 60.3 | 40.4 | 42.8 | 41.6 | 58.9 | 62.7 | 60.7 | 40.4 | 37.8 | 39.1 | 47.1 |
|          |       | − t_j's | 52.0 | 64.3 | 57.5 | 33.1 | 40.2 | 36.3 | 53.4 | 67.9 | 59.8 | 41.9 | 39.2 | 37.6 | 44.6 |

Table 4: Official results on the test set.

| Language | Track | System | Mention Detect. | MUC | B-CUBED | CEAFe | Avg |
|----------|-------|--------|-----------------|-----|---------|-------|-----|
|          |       |        | R   | P   | F   | R   | P   | F   | R   | P   | F   | R   | P   | F   | R | P | F | Avg |
| English  | Closed | Full   | 75.1 | 72.6 | 73.8 | 63.5 | 64.0 | 63.7 | 66.6 | 71.5 | 69.0 | 46.7 | 46.2 | 46.4 | 59.7 |
|          |       | Closed | 71.1 | 72.1 | 71.6 | 59.9 | 64.7 | 62.2 | 69.7 | 77.8 | 73.6 | 53.4 | 48.7 | 51.0 | 62.2 |
|          |       | Full   | 71.5 | 73.5 | 72.4 | 62.5 | 67.1 | 64.7 | 71.2 | 78.4 | 74.6 | 53.6 | 49.1 | 51.3 | 63.5 |
| Arabic   | Closed | Full   | 56.2 | 64.0 | 59.8 | 38.1 | 40.0 | 39.0 | 60.6 | 62.5 | 61.5 | 41.9 | 39.8 | 40.8 | 47.1 |

Table 5: Supplementary results on the test set obtained using gold mention boundaries and predicted parse trees.

| Language | Track | System | Mention Detect. | MUC | B-CUBED | CEAFe | Avg |
|----------|-------|--------|-----------------|-----|---------|-------|-----|
|          |       |        | R   | P   | F   | R   | P   | F   | R   | P   | F   | R   | P   | F   | R | P | F | Avg |
| English  | Closed | Full   | 74.8 | 75.2 | 75.0 | 63.3 | 66.8 | 65.0 | 65.4 | 73.6 | 69.2 | 48.8 | 44.9 | 46.8 | 60.3 |
|          |       | Closed | 82.0 | 79.0 | 80.5 | 70.8 | 72.1 | 71.4 | 74.4 | 79.9 | 77.0 | 58.0 | 56.4 | 57.2 | 68.6 |
|          |       | Full   | 82.4 | 80.1 | 81.2 | 73.5 | 74.3 | 73.9 | 76.3 | 80.5 | 78.3 | 58.2 | 57.3 | 57.8 | 70.0 |
| Arabic   | Closed | Full   | 57.2 | 62.6 | 59.8 | 38.7 | 39.2 | 39.0 | 61.5 | 61.8 | 61.7 | 41.6 | 40.9 | 41.2 | 47.3 |

Table 6: Supplementary results on the test set obtained using gold mention boundaries and predicted parse trees.

| Language | Track | t_C | t_HPL | t_SPL | t_HPU | t_SPU | Rule Relaxation |
|----------|-------|-----|-------|-------|-------|-------|-----------------|
| English  | Closed | −0.06 | 0.27 | 0.06 | 1.01 | 0.06 | 6.63 |
| Chinese  | Closed | −0.10 | 0.00 | −0.08 | 0.10 | 0.27 | 1.63 |
| Chinese  | Open   | −0.10 | 0.00 | −0.08 | 0.10 | 0.27 | 1.63 |
| Arabic   | Closed | 0.05 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

Table 7: Optimal parameter values.

6 Conclusion

We presented a multilingual coreference resolver designed for the CoNLL-2012 shared task. We adopted
a hybrid approach to coreference resolution, which combined the advantages of rule-based methods and learning-based methods. Specifically, we proposed two extensions to Stanford's multi-pass sieve approach, which involved the incorporation of lexical information using machine learning and the acquisition of genre-specific resolvers. Experimental results demonstrated the effectiveness of these extensions, whether or not they were applied in isolation or in combination.

In future work, we plan to explore other ways to combine rule-based methods and learning-based methods for coreference resolution, as well as improve the performance of our resolver on Arabic.

Acknowledgments

We thank the two anonymous reviewers for their comments on the paper. This work was supported in part by NSF Grants IIS-0812261 and IIS-1147644.

References

Michael John Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.
Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 1152-1161.
Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task, pages 28–34.
Vincent Ng. 2004. Learning noun phrase anaphoricity to improve conference resolution: Issues in representation and optimization. In Proceedings of the 42nd Meeting of the Association for Computational Linguistics, pages 151–158.
Vincent Ng. 2009. Graph-cut-based anaphoricity determination for coreference resolution. In Proceedings of the 2009 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 575–583.
Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes, In Proceedings of the Sixteenth Conference on Computational Natural Language Learning.
Altaf Rahman and Vincent Ng. 2011a. Coreference resolution with world knowledge. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 814–824.
Altaf Rahman and Vincent Ng. 2011b. Narrowing the modeling gap: A cluster-ranking approach to coreference resolution. Journal of Artificial Intelligence Research, 40:469–521.
Marta Recasens, Lluís Màrquez, Emili Sapena, M. Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio, and Yannick Versley. 2010. Semeval-2010 task 1: Coreference resolution in multiple languages. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 1–8.