CS224N Coreference Resolution

Jiayuan Ma
jiayuanm@stanford.edu

Xincheng Zhang
xinchen2@stanford.edu

November 7, 2013

1 Score Intepretation

Our implementation of `OneCluster` and `AllSingleton` gave MUC and B3 scores in Table 1. As indicated by their names, `AllSingleton` makes each mention a standalone cluster while `OneCluster` puts all mentions in one cluster. `OneCluster` should have 100% recall, since it has no links to add. On the other hand, `AllSingleton` behaves the opposite, 100% precision, because it requires no edges to add. This tells us both MUC and B3 scores can sometimes be tricked. Over-clustering/segmenting will bring us meaningless high recall/precision. In the case of MUC score, making large clusters (more coreference predictions) will lead to a higher score.

Algorithm	MUC (dev/test)			B3 (dev/test)		
	Precision	Recall	F1	Precision	Recall	F1
OneCluster	76.94/74.39	100.00/100.00	86.97/85.32	16.46/12.68	100.00/100.00	28.27/22.50
AllSingleton	100.00/100.00	0.0/0.0	0.0/0.0	100.00/100.00	24.66/27.32	39.57/42.92

Table 1: Baseline MUC and B3 scores on dev and test set

2 Rule Based Approach

2.1 Better Baseline

Before diving into the implementation of rule based system, we first look at the baseline code provided. The baseline code demonstrates the sample usage of the given library and implemented an exact match coreference system. In our better baseline, we implemented two rules to cluster mentions together. We found that simple head match and golden pair counting can already give us a relatively high score (see Table 2).

- **head match** If two mentions share the same head word, we make them into one cluster.
- **head pair appeared** We collect all golden pairs of head word matches in training phase. If two mentions have the same head word pair appeared during training, we make them into one cluster when testing.

2.2 Implementation Details

We make multiple passes over the data as suggested by the instruction.

Exact Match Same as baseline, but leave out all the first person and second person pronoun pair.

Head Match Same as better baseline, but leave out all the first person and second person pronoun pair.

Head Pair Appeared Same as better baseline, but leave out all the first person and second person pronoun pair.

Hobbs Candidate Filtering We ran Hobbs algorithm against all the sentences in a document. It will filter out a list of word pairs as matching candidates. If a mention matches a candidate pair, we will then check the candidate to see if one of the following conditions is met.

- if the two mentions agree on gender or number
- if the first mention is a human name and the second mention is not a neutral gender pronoun
- if the first mention is not a human name and the second mention did not have a clear gender

Quoted Match We made a special rule to deal with the first person speaker and second person speaker pronoun pairs. If both mentions are not quoted or they share the same speaker, then we simply cluster all first speaker and second speaker by exact match. Otherwise, we match the first speaker to the second speaker and vice versa.

2.3 Error Analysis

The rule based system does not perform very well when mentions are connected by their meanings instead of lexical features. Using rule based algorithms, we easily capture lexical structures, but it is hard for us to encode semantic relations into our rules. Here are some examples.

```
{the crew on the U.S. carrier} -->  !{the sailors remaining on board};
{the story} --> !{a book};
{the U.S.} -->  !{the United States};
{glamorous Princess Di} -->  !{Diana};
```

For all those examples, the lexical evidence is not sufficient enough to decide their coreferences. We need to incorporate semantic information to resolve these coreferences. Hobbs algorithm is quite good at detecting the following coreferent relations.

```
{the story} -->  {it};
{Yemen 's President} -->  {him}; {he}; {his};
{investigators} -->  {their};
```

This is because Hobbs algorithm examines the parsed tree and makes lexical rules accordingly. Also, the mentions that appear in the training set may also help us to filter out the following relations.

```
{the attack} -->  {the bombing of the "Cole"};
```

However, Hobbs algorithm also introduces some "false" matches as follows.

```
{Only ninteen per cent of Americans polled} -->  !{they};
{Your emails} -->  !{they}; !{them}; !{they}; {them};
```

Just by staring these matches without contexts, we may say these coreferent resolutions seem perfectly fine, but they are not in very specific contexts. This is the inherent flaw of all rule-based systems: there is very little (or almost no) universal rule that works under any cases. Human languages are sophiscated and evolving subjects that cannot be simply characterized by some rules.

In summary, in order to raise our recall, we need to find more candidate pairs by using algorithms like Hobbs algorithm to find the connection from one word to another by link made by pronoun. However, introduce such links may lower the precision since it is difficult to rule out false positives. Our result shows that by solely using the head match and head pair appears among traing set rules, we already had a high F1 score for both MUC and B3.

From a machine learning perspective, rule-based approach is a good heuristic-based model that has relatively high bias but low variance. We can observe the data and make complex rules to bring down the model bias. However, it is almost impossible to enumerate and hardcode every possibility in natural languages. Also, it is also difficult to decide which rules to believe when several rules "fire" together. Rule-based approach, in that sense, is limited.

3 Classifier Based Approach

3.1 Features

One subset of our features are generated by converting rules we used in our rule-based system to indicator features in our classifier-based system. We also added the following additional features.

- **Sentence distance**: number of sentences between two mentions.
- **Mention distance**: number of other mentions between two mentions.
- **Pronoun type pair**: A pair of string indicating the pronoun type of the mention.
- **Name Indicator pair**: A pair of boolean indicating if the onPrix/candidate word is a Name string
- **Parsing tree path**: the path between two mentions on the parsing tree.
- **Gender Match**: If the mentions have gender and if the gender matches
- **Number Match**: If the mentions have number and if the number matches

We printed the weight of each feature to understand what features influence the decision.

```
weight sorted by absolute value from large to small
1.792 [true] (CandidatePron(WE), CandidatePron(OUR))
-1.779 [true] Path(No Path)
1.666 [true] (CandidatePron(I), CandidatePron(MY))
1.628 [true] (CandidatePron(HE), CandidatePron(HIS))
-1.580 [true] Path(/PRP$/NP/NN)
1.515 [true] (CandidatePron(HE), CandidatePron(HE))
...
```

We can see the pronoun pairs which have the same gender and same number is a very strong signal that two mentions belong to the same cluster. Also, the head word match and a parsing path from "Noun Phrase" to "Pronoun Phrase" are also very good features for clustering two mentions together.

3.2 Error Analysis

Since we use an additive linear model for our classification task (coreferent v.s. non-coreferent), all features jointly influence the classification results. Although we include all rules from our rule-based systems as binary indicator features, we cannot guarantee that a single "fire" of these features can 100% affect the result like Hobbs algorithm did in rule-based systems. Therefore,it is interesting to observe errors of what Hobbs algorithm already discovered in our rule-based system.

```
{Yemen's President} -->  !{him}; !{he}; !{his};
```

In this example, we produced a false "negative" as we didn't observe many paths of this pattern during training.

Although parsing path is a strong feature, it is not guaranteed that a certain path will definitely produce a coreference mention pair. For instance, the following is a good example to show. For some really common paths that appear many times in our training data (like `KMT` and `he` in this case), our classifier would be likely to produce a false "positive" in some really special context.

```
{KMT} --> !{He}; !{he};
```

Since classifier-based system gives the result depending on all the features it is given, it cannot rule out a wrong mention clustering by a single feature. The advantage of this joint additive effects is that classifier-based system has high precision because cumulative "good" evidences always lead to a coreference with higher confidence. On the other hand, we are still faced with the same problem as in rule-based systems: we lack the semantic clues to do coreference resolution. Possible solutions to solve this problem can be adding semantic information from some knowledge bases or doing joint semantic parsing and coreference resolution at the same time.

4 Performance Summary

The results in Table 2 were obtained by running our coreferance systems on dev and test (only once) sets using default number (100) of training documents. Both our rule-based's and classifier-based's MUC and B3 scores are on-par or beyond the given figures in the writeup.

Algorithm	MUC (dev/test)			B3 (dev/test)		
	Precision	Recall	F1	Precision	Recall	F1
Better Baseline	78.07/78.76	74.41/69.91	76.19/74.07	66.06/73.99	67.37/66.51	66.81/70.05
RuleBased	79.99/80.80	75.15/69.60	77.50/74.78	65.90/75.83	68.93/66.68	67.38/70.96
ClassifierBased	85.37/84.05	72.39/65.29	78.35/73.49	80.72/85.20	64.69/62.62	71.82/72.19

Table 2: MUC and B3 scores of more sophisticated algorithms on dev and test set