A Deterministic Coreference System with Rich Syntactic Features and Semantic Knowledge

Heeyoung Lee
Department of Electrical Engineering
Stanford University

heeyoung@stanford.edu

Sudarshan Rangarajan
Department of Computer Science
Stanford University

sudarshn@cs.stanford.edu

Abstract

Effective Coreference Resolution is a must-have utility within many NLP pipelines. Machine translation, Text summarization, Parsing, Question answering are just a few NLP applications where such a utility provides the necessary confidence towards any decision of and within the system. In this project, we have implemented and investigated numerous enhancements and feature extensions to the baseline system [Haghighi and Klein] part of JavaNLP[1] , in addition to some important bug fixes. This report explains the need for and each such enhancement, besides providing a thorough analysis of the results obtained.

1 Introduction

NLP tasks contain numerous instances where incorrect decisions are arrived at, due to inadequate understanding of the text/corpus contents. A known cause for such inadequacy is either the absence of a good coreference system or the presence of a poor one. Coreference resolution aims to successfully recommend to a linguistic processing task that a given pair of lexical mentions correspond or refer to a similar entity. In other words, its purpose is to identify coreferent chains (chains of referring expressions) that contain mentions referring to the same entity in principle. This information is invaluable when it comes to making the right decisions on proper translations, question answering (lexically restructuring responses), text summarization, parsing refinements and so on.

Despite its simplicity and employing only a few rich features, the deterministic system proposed by Haghighi (baseline) was able to beat all other unsupervised machine learned systems [Bean and Riloff (2004); Poon and Domingos (2008)] and also come quite close to the supervised ones [Culotta et al. (2007); Bengston and Roth (2008); Finkel and Manning (2008)]. We performed numerous experiments on this system, some of them controlled, based on features we felt were key towards identifying coreference across English texts. In addition to providing a brief discussion about our work prior to this class, we also delve into greater detail on each of the numerous features and extensions we employed besides analysis of the results thus obtained. We also propose a few future directions to this project we might undertake post this quarter. Our implementation is entirely based out of Stanford's JavaNLP package, as one of our intentions was to also add a robust, effective and state-of-the-art coreference resolution system to JavaNLP's current pipeline.

The rest of our report is divided into eight sections. Section 2 provides an overview of the baseline system. Section 3 deals with our previous work for CS224U and our findings there. Section 4 provides an overview of the data sets we used in our work. Section 5 deals with all the new experiments we conducted over the course of this quarter, each with an analysis, of our observations. Section 6 deals with a summary of our findings with importance to significant experiments. Section 7 provides our conclusion and Section 8 contains a discussion on future extensions. Section 9 provides our references to works from literature.

2 Simple Coreference System

The baseline system [Haghighi and Klein] initially parses each sentence in a document using the Stanford Parser [Klein and Manning] with each mention in the sentence being mapped to its corresponding node in the parse tree. Using Hobb's tree distance as a measure of syntactic salience, possible coreferent mentions are chosen post enforcement of the i-within-i and other attribute constraints across the candidate set. This system also deterministically deems mentions that participate in ap-

[1] http://nlp.stanford.edu/javanlp/

positive and predicate nominative grammatical constructs to be coreferent. Their final phase, a bootstrapping algorithm, extracts semantically compatible mention pairs from the BLIPP and WIKI corpora that participate in either of the above mentioned grammatical constructs and uses them in future decisions.

3 Previous work

Prior to this quarter, for our CS224U course project, we began with a thorough error analysis of the results from applying the then existing coreference system over the MUC-6 corpus. This system lacked detection of appositives or predicate nominatives as well as use of semantic knowledge from the corpus. For our analysis, we used the gold annotations of MUC-6 corpus towards identifying and thus classifying a majority of these errors into seven categories [Fig 1] namely: Lack of Semantic knowledge, errors in head detection, lack of detection of appositions, predicate nominatives, synonymy, hypernymy and animacy.

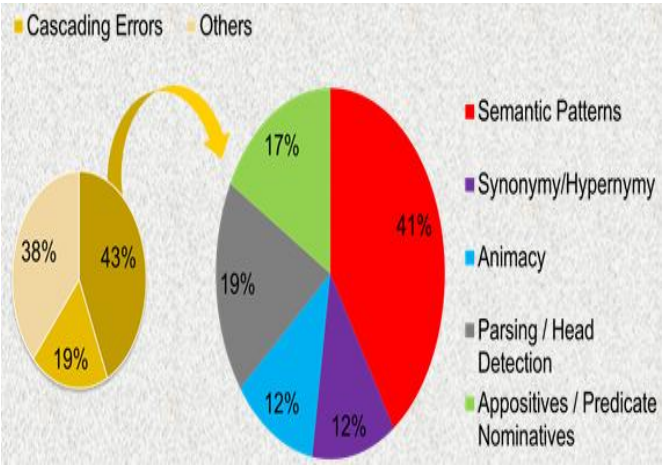

Figure 1– Preliminary Error Analysis

We then enhanced this system to account for most of the above errors. Fixing tokenization errors, i-within-i cases, inclusion of detection of appositions, predicate nominatives, improved gender and head detection, together brought us to a F1 score of 46.55 on the train set. We then implemented our double-pass bootstrapping algorithm to extract patterns and mention pairs over the Gigaword corpus (Figure 2). Prior to the start of this quarter, we were at an F1 score of 46.7 (MUC dev set), 63.7 (MUC test set) being just 3 points away from the baseline system's F1 score on the test set. Our future directions then were to include animacy detection, using a re-ranking system to rank the order of coreferent decisions made for each mention, and use a global knowledge system of previously identified coreferent chains for future decisions.

4 Data sets

• Test-set: To compare our system's results with the baseline system as well as towards final evaluation of our system, we used the MUC-6 test set [30 documents; 2,141 mentions].

• Development: We used the MUC-6 training data [195 documents; 9,853 mentions] as our development set for carrying out all our experiments and recording the results we got from each addition to our system. We used ACE dev set [68 documents and 4,536 mentions] for feature precision analysis in multi-pass system.

• Unlabeled: We used the AP and NYT newswire sections of the Gigaword corpus for extracting semantic knowledge, in contrast to the BLIPP and WIKI datasets used by Haghighi and Klein.

5 Our Experiments

Having a baseline implementation at 63.7 F1 on the test set meant that we had to identify as many possible cases of coreference patterns widely used in English writing styles and account for them in our system towards boosting our final F1 score. Simple stats of error distributions are in table 1.

Number of precision errors		Strategy
With pronoun	99	Animacy detection, attributes sharing, reranking,
Without pronoun	116	Discourse salience, considering modifiers
Number of recall errors		
With pronoun	130	Animacy detection, attributes sharing, reranking,
Without pronoun	437	Semantic knowledge

Table 1 – Preliminary error distribution

We can use different strategies to resolve errors in different categories. For example, semantic knowledge can reduce the number of recall errors without pronoun, but might not be very helpful for precision error with pronoun.
To this end, we selected numerous features and extensions to our baseline implementation, each of which is discussed in detail below.

5.1 SKES (Semantic Knowledge Extraction System)

Last quarter we were able to only extract patterns from seed pairs and use both towards coreference resolution. This quarter, we were able to obtain semantic mention pairs, that is, mention pairs that are extracted from intermediate patterns. We also spent considerable effort

refining the yield of this algorithm. Multiple iterations of refinements and filtration over our initial seed pairs set yielded us, a relatively cleaner collection of high-confidence coreferent semantic mention pairs and patterns from the bootstrapping SKES algorithm (SKES - Fig 2). These seed pairs were obtained by extracting mention pairs around appositive and predicate nominative constructs in the Gigaword corpus. (Table 2)

One hundred handpicked seed pairs, posts the first pass of SKES were then fed into the second pass of our algorithm. Further patterns in which these mention pairs participated across the corpus were extracted and were then used again to extract more mention pairs. To avoid noisy patterns, we tried various ranking methods based on 5 metrics (Table 5). We found that sorting by word coverage with minimum precision cutoff (0.005) gives the best list of patterns. Examples of patterns from SKES are shown in Table 3. The last pattern represents the dependency of this method on the corpus, with many instances of 'of America' occurring in between selected mention pairs (Example – 'president of America Obama' with 'president, Obama' being the seed mention pair.)

From SKES, we got semantically compatible mention pairs and coreferent patterns. One of our extensions from Haghighi's paper was directly applying both semantically compatible pair and semantic pattern on this system instead of using only mention pairs. Coreference pattern matching did not help much even though we had quite a good list of semantic patterns. The reason is that only a few semantic pattern matches actually occurred when we did the actual coreference resolution. When we used 100 semantic patterns, pattern matching occurred only 17 times on MUC train set, 15 of them correct, 4 of them incorrect, owing to parser or annotation errors. (Precision of this strategy was good, but it affected only small portions of the corpus)

When we applied mention pairs to our system, F1 score actually dropped. We think that we need to carefully apply semantic knowledge even if we do have a good list of semantically compatible mentions pairs. The correctness of applying semantic knowledge also depends on the contexts. For example, 'currency' and 'dollar' are compatible, but 'currency' can also be deemed coreferent with 'yen' depending on their context. Towards a more careful and robust application of semantic knowledge, we will combine semantic knowledge with attribute-sharing and multi-pass system later.

Mention 1	Mention 2
capital	kabul
capital	pristina
capital	colombo
association	group
hiv	virus
leader	khamenei
house	duma
commander	mladic

Table 2- Examples of seed mention pairs

Pattern
[(, , ,)]
[(which,WDT), (be,VB)]
[(know,VB), (as,IN)]
[(who,WP), (be,VB)]
[(have,VB), (become,VB)]
[(be,VB), (know,VB), (as,IN)]
[(be,VB), (consider,VB)]
[(call,VB), (be,VB)]
[(be,VB), (name,VB)]
[(of,IN), (america,NNP)]

Table 3 – Examples of semantic patterns from SKES

5.2 Role appositives and Relative pronoun

The next feature inclusion was the identification of role appositives in the corpus. Haghighi and Klein term 'Role appositives' to be constructs, where an NP term occurring immediately to the left of other NP term(s) with entity type 'PERSON', denotes their role. An example they highlight is 'painter Picasso'. Here, Picasso is a mention with entity type 'PERSON', and so 'painter' would be deemed coreferent with Picasso, as it is a likely role of Picasso. Some examples deemed coreferent by our system after this feature inclusion were:

Secretary	Danny Sullivan
Rep.	Fortney Stark
President	Owen Bieber
chief negotiator	William Fisher

Another feature that we also added was checking for relative pronouns. In the MUC and ACE corpora, there are many relative pronoun mentions. This is another exception of the i-within-i constraint. For example, if we have a sentence "a Democrat who was running for the Senate", 'Democrat' and 'who' are coreferent to each other. This feature gives high feature precision of 0.982 on the ACE dev set. Including these two features boosts F1 by 2% (Table 4)

Without role appositive /relative pronoun	F1 = 0.536, P = 0.649, R = 0.457
With role appositive /relative pronoun	F1 = 0.556, P = 0.640, R = 0.492

Table 4 – Effect of role appositive/relative pronoun

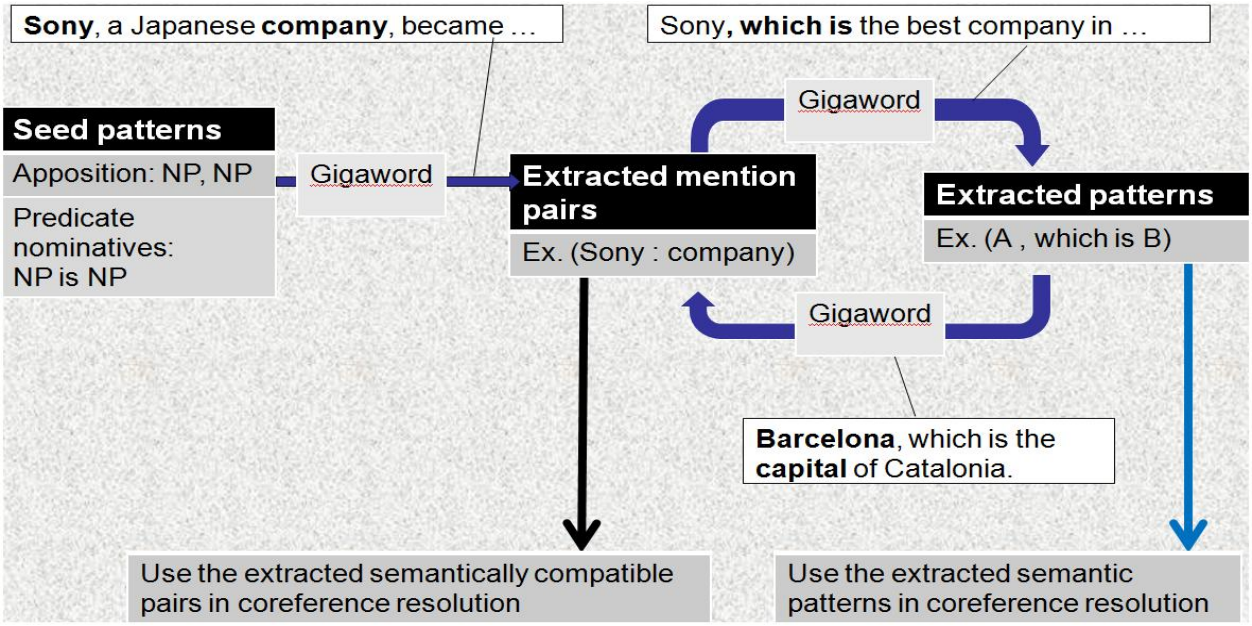

Figure 2 –Semantic Knowledge Extraction System (SKES - Bootstrapping Algorithm)

Metric	Meaning
Hit	Number of occurrences of pattern with all seed pairs
Pair coverage	Number of seed pairs which occur around this pattern
Word Coverage	Number of words which occur around this pattern
Precision	Hit / Total number of occurrences of pattern in corpus
Precision * log(hit)	Precision * log (hit)

Table 5 – Metrics used in selecting semantic patterns from SKES

5.3 Using gender, animacy and number lists

One of our future courses of action prior to the start of this quarter was to include animacy detection in our system, as mentioned in section 3. Our preliminary error analysis indicated lack of animacy detection to be somewhat influential. When going through the corpus, we observed both erroneous and ambiguous annotations for attributes such as gender, number as well. So, we used Shane Bergsma's lists [Bergsma and Lin, 2006] for gender and number, and Dekang-Lin's for animacy [Ji and Lin, 2009], which are lists of mentions with their corresponding gold-standard attribute annotations, to try to eradicate mistaken annotations of gender and number across the corpus.

The examples of mentions that were assigned new attributes are shown in table 6 and effect of this feature is shown in table 7.

Animacy	
pay	Inanimate
Representatives	Animate
Gender	
Doublas	Male
Sony	Neutral
Number	
OPEC	Singular
Most	Plural

Table 6 – Examples of mentions assigned new attributes and their new attributes.

Original System	F1 = 0.670, P = 0.776, R = 0.589
Better animacy	**F1 = 0.682, P = 0.791, R = 0.600**
Better animacy + gender	F1 = 0.680, P = 0.785, R = 0.600
Better animacy + number	F1 = 0.681, P = 0.786, R = 0.600
Better animacy + gender + number	F1 = 0.678, P = 0.781, R = 0.599

Table 7 – Effect of using lists (on MUC6 test)

As shown in table 7, using animacy list helped in increasing both precision and recall, but other lists didn't help. The reason is that we already had decent estimates of numbers and genders, but did not have animacy estimation in original system.

5.4 Stanford NER versus existing NE annotation

So far, we have used gold named entity annotation from the data set. Instead of gold NE annotation, we changed the system to use Stanford NER tagger (Finkel et al., 2005) for named entity label. As we can guess, this change hurts the performance of our system as shown in table 8.

ACE dev	F1	P	R
Gold NE	0.574	0.652	0.513
Stanford NER	0.556	0.64	0.492
MUC6 test			
Gold NE	0.684	0.803	0.596
Stanford NER	0.67	0.777	0.588

Table 8 – Effect of NER on ACE dev and MUC6 test

5.5 Discourse Salience

A mention with an indefinite article is most likely being introduced initially at that point where observed. However, a mention preceded by a definite article is most likely to have been introduced elsewhere. The same applies to definite/indefinite pronouns and adjectives.

If we know that a mention is newly appeared in the text, we can stop to find coreferent mentions to the mention, which can increase the performance. We tried three heuristics: 1) definite/indefinite article – 'a' vs 'the', 2) indefinite pronoun – 'some' or 'another' are not coreferent to antecedent with high probability, 3) indefinite adjective – 'another option' or 'other person' will not be coreferent to antecedent. This feature increase precision little bit, but drops recall and F1.

Without discourse salience	F1 = 0.682, P = 0.786, R = 0.603
With discourse salience	F1 = 0.677, P = 0.791, R = 0.592

Table 9 – Effect of discourse salience

5.6 Using uncompatible modifiers

There are many errors due to same heads.

For example, 'Gord team' cannot corefer 'Bush team', and 'Stanford University' and 'Harvard University' cannot be coreferent. To avoid these errors, we considered modifier of mentions. If two mentions has exactly same words except one word whose named entity are same, then those two mentions are highly uncoreferent. This feature increases the precision of head matching (Table 12).

5.7 Using Acronyms

We also include a feature for acronym. If one of the mentions has multiple capitalized words (International Business Machines), and the other mention is a single word that's all uppercase (IBM), then check for the capital letters of the previous mention (I, B, M) and see if it matches with the second mention. Data set has only few acronyms, so does not help much (only 21 acronyms in MUC, 3 acronyms in ACE dev)

5.8 Coreference clusters (chains) and Attribute sharing

We have done coreference resolution between two mentions. However, as more mentions are done by system, there is more information in the coreference cluster. For example, if we have 3 mentions coref cluster, 'Bush'-'President'-'he', then it gives us more combined information such as Bush is male. Before we start to run the system, we do not know the gender of 'Bush' and 'President', but after we get this cluster, we can know both of them are male. This global information should help our coref task, especially when we do it in multi-pass (section 5.9). Sometimes the attributes of mentions can be conflict each other. For instance, if 'union' and '2 firms' are in a same cluster, the cluster has two numbers 'Singular' from 'union', 'Plural' from '2 firms'. In that case, we make attributes set {'singular', 'plural'}, so we preserve the information that a mention can be both. There are 4 attributes: number, gender, animacy, named entity type.

Without attributes sharing	F1 = 0.682, P = 0.786, R = 0.603
With attributes sharing	F1 = **0.688**, P = 0.809, R = 0.598

Table 10 – Effect of attributes sharing

5.9 Multi-pass system

One thing we need to be careful when we apply attributes sharing is error propagation. If there is an error in a cluster, the attributes of the wrong mention propagate to the entire mentions in the same cluster. To make it more robust, we introduced multi-pass system. The idea of multi-pass system is that to identify coreference in one pass and use that cumulative knowledge across subsequent passes.

Our passes were based on controlled singleton and combined experiments considering various features of a pair of candidate mentions like exact head match, exact string match, entity match, individual and collective match of attributes (number, gender, animacy and type), and our stand-alone system otherwise.

We make clusters in the order that we are more confident about the correctness. For example, if the apposition gives higher precision than head matching, we first find appositions, and then do head matching with more information we get from clusters after first phase. To make this multi-pass system works, we need to know the precision of each feature (feature precision). The precision of various feature are shown in table 11.

Feature	**Precision**
Predicate nominatives	1 (= 12/12)
Relative pronoun	0.982 (= 56/57)
Exact string match	0.965 (= 2084/2160)
Apposition	0.892 (= 83/93)
Head/attributes match considering modifier	0.737 (= 3256/4449)
Head/attributes match	0.717 (= 3220/4494)
Role apposition	0.691 (= 85/123)
Acronym	0.667 (= 2/3)
Pronoun	0.612 (= 916/1497)
Single pass system with above features	0.644 (= 6347/9854)

Table 11 – Feature precision (on ACE dev)

6 Summary of Results

The summary of the effects of each feature and results of single/multi-pass system are shown in table 12 and 13.

Feature	F1 gain	P gain	R gain
Semantic knowledge (SK)	-0.016	-0.01	-0.02
Role appositives /relative pronoun (RA/RP)	+0.02	-0.009	+0.035
Attributes lists (AL)	+0.012	+0.015	+0.011
Change to NER from gold NE (NER)	-0.018	-0.012	-0.021
Discourse salience (DS)	-0.005	+0.005	-0.011
Uncompatible modifier	-0.012	+0.002	-0.020
Acronym	+0.000	+0.000	+0.000
Attributes sharing (AS)	+0.006	+0.023	-0.005

Table 12 – Summary: effects of features

Figure 3. Summary: effects of features

Except NER, the drops in semantic knowledge, discourse salience, and modifier is the opposite results from our expectation. Even though F1 decreased in discourse salience and modifier, the precision increased, so we might get some advantages based on them or refined version of them in multi-pass system, which requires high precision features.

	F1	P	R
Single pass	0.548	0.644	0.476
Multi pass	0.504	0.548	0.467

Table 13 – Results of single/multi pass system (with same feature sets)

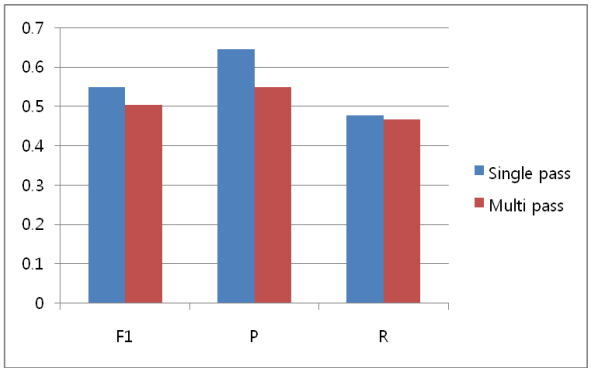

Figure 4. Results of single/multi pass system (with same feature sets)

Pass	Precision of each pass
1. Exact string match, apposition, relative pronoun, predicate nominatives	0.961
2. headmatch	0.492
3. role appositives	0.213
4. pronoun	0.490

Table 14 – Precision of each pass

The performance (especially precision) of multi-pass system is much worse than our original single-pass system. It seems that the precision drops a lot when it goes through later passes. The multi-pass system is more sensitive to error propagation (because we merge clusters, not mentions), so more careful cluster merging is required. We are now investigating how to resolve this problem (future work).

7 Conclusion

Our numerous experiments helped us gain a lot of insight into which features are useful and which aren't. Despite lower F1 scores, we were able to identify a broad range of features which would help us develop a re-ranker on our coreference decisions in future, Our interest lies heavily with both the re-ranker and the multi-pass system which we believe if constructed and used with the right pass orders would actually yield us better and very interesting results. Right now, our score stands highest at 68.8 F1 post attribute sharing and with an improved re-ranking and multi-pass system, we should be able to confidently expect a better score.

8 Future work / Extensions

We have a few more experiments in mind to conduct before we begin work on our EMNLP submission. Those experiments include coming up with an efficient re-ranking scheme which lets us associate confidence levels to our co-reference decisions, improving on our current multi-pass system through fine-tuning our existing passes as and including newer ones, better utilization of semantic knowledge in decision making. If the last one goes well, we also plan to incorporate better semantic knowledge utilization into both our re-ranker and multi-pass systems.

9 References

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

Eric Bengston and Dan Roth. 2008. Understanding the value of features for coreference resolution. *In Empirical Methods in Natural Language Processing.*

Jenny Finkel and Christopher Manning. 2008. Enforcing transitivity in coreference resolution. *In Association of Computational Linguists (ACL).*

David Bean and Ellen Riloff. 2004. Unsupervised Learning of Contextual Role Knowledge for Coreference Resolution. *In Proceedings of HLT/NAACL 2004.*

Hoifung Poon and Pedro Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. *In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing.*

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.*

D. Klein and C. Manning. 2003. Accurate unlexicalized parsing. *In Association of Computational Linguists (ACL).*

Heng Ji and Dekang Lin. 2009. Gender and Animacy Knowledge Discovery from Web-Scale N-Grams for Unsupervised Person Mention Detection. *Proc. PACLIC 2009.*

Shane Bergsma and Dekang Lin, Bootstrapping Path-Based Pronoun Resolution, *In Proceedings of the Conference on Computational Linguistics / Association for Computational Linguistics (COLING/ACL-06), Sydney, Australia, July 17-21, 2006*