Supervised Coreference Resolution with SUCRE

Hamidreza Kobdani and Hinrich Schütze
Institute for Natural Language Processing
University of Stuttgart, Germany
kobdani@ims.uni-stuttgart.de

Abstract

In this paper we present SUCRE (Kobdani and Schütze, 2010) that is a modular coreference resolution system participating in the CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNote (Pradhan et al., 2011). The SUCRE's modular architecture provides a clean separation between data storage, feature engineering and machine learning algorithms.

1 Introduction

Noun phrase coreference resolution is the process of finding markables (noun phrase) referring to the same real world entity or concept. In other words, this process groups the markables of a document into entities (equivalence classes) so that all markables in an entity are coreferent. Examples of applications of coreference resolution are Information Extraction, Question Answering and Automatic Summarization.

Coreference is an equivalence relation between two markables, i.e., it is reflexive, symmetric and transitive. The first solution that intuitively comes to mind is binary classification of markable pairs (links). Therefore at the heart of most existing approaches there is a binary classifier that classifies links to coreferent/disreferent. One can also use the transitive property of coreference relation to build the entities; this is done using a clustering method.

Our approach in this paper consist of the above mentioned steps, namely:

1. Classification of links to coreferent/disreferent.
2. Clustering of links which are classified as coreferent.

This paper is organized as follows. In Section 2, we present our feature engineering approach. Section 3 presents the system architecture. Data set is described in Section 4. Sections 5 and 6 present results and conclusions.

2 Feature Engineering

In recent years there has been substantial work on the problem of coreference resolution. Most methods present and report on the benchmark data sets for *English*. The feature sets they use are based on (Soon et al., 2001). These features consist of string-based features, distance features, span features, part-of-speech features, grammatical features, and agreement features.

We defined a comprehensive set of features based on previous coreference resolution systems for English, e.g. (Bengtson and Roth, 2008). In the common approach to coreference resolution we have chosen, features are **link features**, i.e., features are defined over a pair of markables. For link feature definition and extraction, the head words of markables are usually used, but in some cases the head word is not a suitable choice. For example, consider these two markables: *the book* and *a book*, in both cases *book* is the head word but to distinguish which markable is definite and which indefinite additional information about the markables has to be taken into account. Now consider these two markables: *the university students in Germany* and *the university students in France* in this case the head words and the first four words of each markable are the same but they cannot be coreferent, and this could be detected only by looking at the entire noun phrase. Some features require complex preprocessing or complex definitions. Consider the two markables *the members of parliament* and *the members of the European Union*. The semantic class of *members* is *person* in the first case and *country* in the second. To cover all such cases, we introduced a feature definition language (Kobdani et al., 2010). With the feature definition language we will be able to access all information that is connected to a markable, including the first, last and head words of the two markables; all other words of the two markables; and the two markables as atomic elements.

After defining new features (new definition from scratch or definition by combination of existing features), we have to evaluate them. In principle, we could use any figure of merit to evaluate the usefulness of a feature or to compare two similar features, including Gini coefficient, mutual information, and correlation coefficient. In our current system, expected information gain (IG) and information gain ratio (IGR) are used.

As an example, consider the following two features, which can be considered different attempts to formalize the same linguistic property:

1. The noun phrase has a subject role and is *definite* (e.g. markable begins with a *definite* article)

2. The noun phrase has a subject role and is *not indefinite* (e.g. markable begins with an *indefinite* article)

The information gain ratios of the above mentioned features are equal to 0.0026 for the first and 0.0051 for the second one – this shows that the second one is a better choice. We now define IG and IGR.

The change in entropy from a prior state to a state that takes some information is the expected information gain (Mitchell, 1997):

$$IG\left(f\right) = H\left(C\right) - H_f\left(C\right) \tag{1}$$

Where f is the feature value, C its corresponding class, and entropy is defined as follows:

$$H\left(C\right) = -\sum_i P\left(C_i\right) log_2 P\left(C_i\right) \tag{2}$$

$$H_f(C) = \sum_f \frac{|C_f|}{|C|} H\left(C_f\right) \tag{3}$$

If a feature takes a large number of distinct values, the information gain would not be a good measure for deciding its relevance. In such cases the information gain ratio is used instead. The information gain ratio for a feature is calculated as follows:

$$IGR\left(f\right) = \frac{IG\left(f\right)}{SInf\left(C\right)} \tag{4}$$

$$SInf(C) = -\sum_i \frac{|C_i|}{|C|} log_2 \frac{|C_i|}{|C|} \tag{5}$$

Equation (4) can be used as an indicator for which features are likely to improve classification accuracy.

3 System Architecture

The architecture of the system has two main parts: preprocessing and coreference resolution.

In preprocessing the text corpus is converted to a relational data model. The main purpose of the relational model in our system is the use of a feature definition language (Kobdani et al., 2010). After modeling the text corpus, coreference resolution can be performed.

The main steps of the system are presented as follows.

3.1 Preliminary text conversion

In this step, tokens are extracted from the corpus. In the CoNLL-2011 Shared Task this step is as simple as reading each line of the input data set and extracting its corresponding token.

3.2 Atomic attributes of tokens

Atomic features of the tokens are extracted in this step. The extracted atomic features are: part of speech, number, pronoun person (first, second and third), pronoun type (subjective,,predeterminer,reflexive,objective and possessive), WordNet semantic class and gender.

We use a rather simple method to extract semantic class of each token from WordNet. We look at the synonyms of the token and if one of them is in the predefined keyword set, we take it as its corresponding semantic class. The example of the keywords are person, time, abstraction, device, human action, organization, place and animal.

3.3 Markable Detection

In this step all noun phrases from the parse tree are extracted. After clustering step all markables which are not included in a chain are deleted from the list of markables. In other word we will not have any cluster with less than 2 members.

Figure 1 presents the simple markable detection method which we used in the SUCRE.

3.4 Atomic attributes of markables

In this step, the atomic attributes of the markables are extracted. In the data set of the CoNLL-2011 shared task the named entity property of a markable can be used as its atomic attribute.

3.5 Link Generator

For training, the system generates a positive training instance for an adjacent coreferent markable pair (m, n) and negative training instances for the markable m and all markables disreferent with m that occur before n (Soon et al., 2001). For decoding it generates all the possible links inside a window of 100 markables.

3.6 Link feature definition and extraction

The output of the link generator, which is the list of the generated links, is the input to the link feature extractor for creating train and test data sets. To do this, the feature definitions are used to extract the feature values of the links (Kobdani et al., 2011).

3.7 Learning

For learning we implemented a decision tree classifier (Quinlan, 1993). To achieve state-of-the-art performance, in addition to decision tree we also tried support vector machine and maximum entropy that did not perform better than decision tree.

3.8 Classification and Clustering

In this part, the links inside one document are classified then the coreference chains are created. We use *best-first clustering* for this purpose. It searches for the best predicted antecedent from right to left starting from the end of the document. For the documents with more than a predefined number of markables we apply a limit for searching. In this way, in addition to better efficiency, the results also improve.

```
Markable_Detection_PSG_A ($W_1, W_2, \ldots, W_n$)
    1. A markable $M$ is presented by a set of three words:
       Begin ($M_b$), End ($M_e$) and Head ($M_h$).
    2. Let $DM$ be the set of detected markables.
    3. Let $T_i$ be the node $i$ in the parse tree with label $L_i$
       (if node is a word then $L_i$ is equal to $W_i$).
    4. Start from parse tree root $T_r$:
       Find_Markables($T_r$,$L_r$,$DM$)

Find_Markables($T$,$L$,$DM$)
    1. If $L$ is equal to noun phrase, then extract the markable $M$:
        (a) Set the begin word of the markable:
            $M_b$ = Noun_Phrase_Begin($T$,$L$)
        (b) Set the end word of the markable:
            $M_e$ = Noun_Phrase_End($T$,$L$)
        (c) Set the head word of the markable:
            $M_h$ = Noun_Phrase_Head($T$,$L$)
        (d) Add the markable $M$ to the set of detected markables $DM$.
    2. Repeat for all $T_i$ the daughters of $T$:
       Find_Markables($T_i$,$L_i$,$DM$)

Noun_Phrase_Begin($T$,$L$)
    If $T$ has no daughter then return $L$;
    else set $T_b$ to the first daughter of $T$ and return
    Noun_Phrase_Begin($T_b$,$L_b$).

Noun_Phrase_End($T$,$L$)
    If $T$ has no daughter then return $L$;
    else set $T_b$ to the last daughter of $T$ and return
    Noun_Phrase_End($T_b$,$L_b$).

Noun_Phrase_Head($T$,$L$)
    If $T$ has no daughter then return $L$;
    else set $T_h$ to the biggest noun phrase daughter
    of $T$ and return Noun_Phrase_Head($T_h$,$L_h$).
```

Figure 1: Markable Detection from Parse Tree (all possible markables) .

	Automatic			Gold		
	Rec.	Prec.	F_1	Rec.	Prec.	F_1
MD	60.17	60.92	60.55	62.50	61.62	62.06
MUC	54.30	51.84	53.06	57.44	53.15	55.21
B^3	71.39	64.68	67.87	74.07	64.39	68.89
$CEAF_M$	46.36	46.36	46.36	47.07	47.07	47.07
$CEAF_E$	35.38	37.26	35.30	35.19	38.44	36.74
BLANC	65.01	64.93	64.97	66.23	65.16	65.67

Table 1: Results of SUCRE on the development data set for the automatically detected markables. MD: Markable Detection.

	Automatic			Gold		
	Rec.	Prec.	F_1	Rec.	Prec.	F_1
MUC	58.63	87.88	70.34	60.48	88.25	71.78
B^3	57.91	86.47	69.36	59.21	86.25	70.22
$CEAF_M$	59.81	59.81	59.81	60.91	60.91	60.91
$CEAF_E$	70.49	36.43	48.04	71.09	37.73	49.30
BLANC	69.67	76.27	72.34	70.34	76.01	72.71

Table 2: Results of SUCRE on the development data set for the true markables (i.e. no singletone is included).

4 Data Sets

OntoNotes has been used for the CoNLL-2011 shared task. The OntoNotes project [1] is to provide a large-scale, accurate corpus for general anaphoric coreference. It aims to cover entities and events (i.e. it is not limited to noun phrases or a limited set of entity types) (Pradhan et al., 2007).

For training we used 4674 documents containing a total of 1909175 tokens, 190700 markables and 50612 chains.

SUCRE participated in the closed track of the shared task. Experiments have been performed for the two kind of documents, namely, the automatically preprocessed documents and the gold preprocessed documents. In this paper, we report only the scores on the development data set using the official scorer of the shared task. The automatically preprocessed part consists of 303 documents containing a total of 136257 tokens, 52189 automatically detected markables, 14291 true markables and 3752 chains. The gold preprocessed part consists of 303 documents containing a total of 136257 tokens, 52262 automatically detected markables, 13789 true markables and 3752 chains.

5 Results

We report recall, precision, and F_1 for MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), $CEAF_M$/$CEAF_E$ (Luo, 2005) and BLANC (Recasens et al., 2010).

Table 1 presents results of our system for the automatically detected markables. It is apparent from this table that the application of the gold preprocessed documents slightly improves the performance (MD-F1: +1.51; MUC-F1: +2.15; B^3-F1: +1.02; $CEAF_M$-F1: +0.71; $CEAF_E$-F1: +1.44; BLANC-F1: +0.70).

Table 2 presents results of our system for the true markables that were all and only part of coreference chains. Again the results show that the application of gold preprocessed documents slightly improves the performance (MUC-F1: +1.44; B^3-F1: +0.86; $CEAF_M$-F1: +1.1; $CEAF_E$-F1: +1.26; BLANC-F1: +0.37).

Comparing the results of tables 1 and 2, there is a significant difference between the scores on the automatically detected markables and the scores on the true markables (e.g. for the automatically preprocessed documents: MUC-F1: +17.28; $CEAF_M$-F1: +13.45; $CEAF_E$-F1: +12.74; BLANC-F1: +7.37). No significant improvement in B^3 is seen (automatic: +1.49; gold: +1.33). We suspect that this is partly due to the very sensitive nature of B^3 against the singleton chains. Because in the implementation of scorer for the CoNLL-2011 shared task the non-detected key markables are automatically included into the response as singletons.

6 Conclusion

In this paper, we have presented our system SUCRE participated in the CoNLL-2011 shared task. We took a deeper look at the feature engineering of SUCRE. We presented the markable detection method we applied.

We showed that the application of the gold preprocessed documents improves the performance. It has been demonstrated that the availability of the true markables significantly improves the results. Also it has been shown that the singletons have a large impact on the B^3 scores.

[1] http://www.bbn.com/ontonotes/

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *LREC Workshop on Linguistics Coreference '98*, pages 563–566.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *EMNLP '08*, pages 294–303.

Hamidreza Kobdani and Hinrich Schütze. 2010. Sucre: A modular system for coreference resolution. In *SemEval '10*, pages 92–95.

Hamidreza Kobdani, Hinrich Schütze, Andre Burkovski, Wiltrud Kessler, and Gunther Heidemann. 2010. Relational feature engineering of natural language processing. In *CIKM '10*. ACM.

Hamidreza Kobdani, Hinrich Schütze, Michael Schiehlen, and Hans Kamp. 2011. Bootstrapping coreference resolution using word associations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, ACL '11. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *HLT '05*, pages 25–32.

Tom M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, New York.

Sameer Pradhan, Lance Ramshaw, Ralph Weischedel, Jessica MacBride, and Linnea Micciulla. 2007. Unrestricted Coreference: Identifying Entities and Events in OntoNotes. In *in Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*, September 17-19.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL 2011)*, Portland, Oregon, June.

J. Ross Quinlan. 1993. *C4.5: Programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

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 *SemEval '10*, pages 70–75.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. In *CL '01*, pages 521–544.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *MUC6 '95*, pages 45–52.