Clustering Algorithms for Noun Phrase Coreference Resolution

Roxana Angheluta, Patrick Jeuniaux, Rudradeb Mitra, Marie-Francine Moens

roxana.angheluta@law.kuleuven.ac.be, patrick.jeuniaux@law.kuleuven.ac.be, rudradeb_mitra@rediffmail.com, marie-france.moens@law.kuleuven.ac.be

Katholieke Universiteit Leuven, Belgium

Published in 7'es Journées Internationales d'Analyse Statistique des Donées Textuelles, Presses universitaires de Louvain, 2004, vol1, pp. 60-70
ISBN: 2-930344-49-0

Abstract

In this paper, we present four clustering algorithms for noun phrase coreference resolution. We developed two novel algorithms for this task, a fuzzy algorithm and its hard variant and evaluated their performance on two different sets of texts in comparison with an existing fuzzy and a hard clustering algorithm that are described in the literature. Our algorithms perform slightly better and do not rely on a predefined threshold distance value for cluster membership. In addition, our fuzzy clustering algorithm seems to perform better than a hard clustering on a pronoun resolution task.

Keywords: Coreference resolution, clustering.

1 Introduction

Most of the natural language processing applications that deal with meaning of discourse imply the completion of some reference resolution activity. The first kind of reference resolution that appears significant in this framework is noun phrase coreference resolution. This is the ability to relate each noun phrase in a text to its referent in the real world. Our coreference resolution focuses on detecting "identity" relationships (i.e. not on is-a or whole/part links for example). Two entities are considered as coreferents when they both refer to the same noun phrase in the situation described by the text (e.g., in the sentences: "Dan Quale met his wife in college. The Indiana senator married her shortly after he finished his studies": "his", "Indiana senator" and "he" all co-refer to "Dan Quale").

It is natural to view coreferencing as a partitioning or clustering of the set of entities. The idea is to gather coreferents into the same cluster, which is accomplished in two steps: 1) detection of the entities and extraction of a specific set of their features; 2) clustering of the entities. For the first subtask we use the same set of features as in [3]. We implemented two novel algorithms for the second step: a progressive fuzzy clustering algorithm and its hard variant. We also implemented the hard clustering algorithm presented in [3] and

Feature f	Weight w_f	$function_f$
Words	10.0	(#of mismatching words)/(# of words in longer NP)
Head Noun	1.0	1 if the head noun differs; else 0
Position	5.0	difference in position/maximum difference in document
Pronoun	5.0	1 if NP_i is pronoun and NP_j is not; else 0
Article	5.0	1 if NP_j is indefinite and not appositive; else 0
Words-Substring	$-\infty$	1 if NP_i subsumes NP_j
Appositive	$-\infty$	1 if NP_j is appositive and NP_i is its immediate predecessor; else 0
Number	∞	1 if they do not match in number; else 0
Proper Name	∞	1 if both are proper names but mismatch in every word; else 0
Gender	∞	1 if genders are contradictory; else 0
Semantic class	∞	1 if entities have different semantic classes; else 0

Table 1: Set of features and their weights used in coreference resolution

a fuzzy clustering algorithm as described in [2]. Our goal is to test the quality of the coreference resolution that is achieved by these four algorithms.

Coreference resolution is very valuable in text based applications including information retrieval, text summarization, information extraction and question answering, as it refines the representations made of the content of a text.

The next section explains the methods used for feature selection and the four clustering algorithms. We then describe our experiments and their results. In the discussion the four algorithms are critically evaluated.

2 Methods

2.1 Feature Selection

An entity is a noun phrase in the text. Each entity is represented as a set of 11 features (see table 1). Here follows a short description of their definition and mode of extraction:

Individual Words: The words of the noun phrase are used to measure the mismatching words between two entities and to see if one entity subsumes (includes totally as a substring) the other.
Head Noun: The noun in the phrase that is not the modifier of another noun.
Position: Each entity is indexed from 1 to n, according to their order of occurrence in the text.
Pronoun: A pronoun is recognized by its part-of-speech (POS) tag.
Article: A noun phrase can be either Indefinite (contains 'a' or 'an'), Definite (contains 'the') or None (without article).
Appositive: An entity is considered appositive if it is enclosed between "," and ",",";" or ".", it is a proper name or contains an article and it is immediately preceded by another noun phrase in the text.
Number: The number is detected using the POS tag.
Proper Name: Proper Names are identified by looking at the capitalization of the words.
Gender: A list [6] is used to distinguish the male and female first names. Short lists with words like *brother, mother, etc.* are used to assign gender to common nouns. The rest of the nouns remain unresolved.

Semantic Class: Here we use WordNet [7] to classify objects, human entities and pronouns into three different semantic classes: 0, 1 and 2 respectively (S_0, S_1, S_2).
In contrast with [3], we do not include the animacy feature, which indicates whether or not the entity is a living thing.

2.2 Distance Metric

The clustering algorithms use the following metric for computing the distance between two entities NP_i and NP_j:

$$dist(NP_i, NP_j) = \sum_{f \in F} w_f * function_f(NP_i, NP_j) \tag{1}$$

where F corresponds to the entity feature set, w_f is the weight of a feature, $function_f$ returns a value in [0,1] (see table 1). A weight of ∞ has priority over $-\infty$: if two entities mismatch on a feature which has a weight of ∞, then they have a distance equal to ∞ (so they are not considered as coreferents). The approach is consistent with [3].

3 The Clustering Methods

3.1 Hard Clustering Cardie et al. (HC-C)

In the algorithm of [3] each entity initially forms its own cluster (i.e. a singleton cluster). The algorithm starts from the end of the document and goes backwards, while each noun phrase cluster is compared with all preceding clusters. If the distance (1) between two noun phrases of the two compared clusters is less than a distance value (a threshold set by experiments), their clusters merged, provided all entities are compatible (i.e. don't have a pairwise distance of ∞).

We pick for each cluster a representative member (medoid), chosen as the first person name in the text, if it exists, otherwise as the first entity in the text that appears in the cluster.

The algorithm is very simple and fast, however it has also some weak points.

- The highly greedy character of this algorithm (as it considers the first match and not the best match) introduces errors which are further propagated as the algorithm advances. In the example *"Robert Smith lives with his wife ... Smith loves her"*., when clustering the entities *"Robert Smith"*, *"wife"*, *"Smith"* and *"her"*, the gender of *"Smith"* cannot be determined as it can be both male or female. *"Smith"* may come in the same cluster with *"her"*, if allowed by the distance threshold, but then *"Robert Smith"* will never be correctly resolved with the entity *"Smith"* because of the incompatibility between *"Robert Smith"* and *"her"*.
- The algorithm is very dependent on the threshold distance value for cluster membership. The value of this distance is determined experimentally and may be tuned for each document, which makes the algorithm corpus-dependent.

- The single pass algorithm is very dependent on the order when comparing clusters. Often there are different possibilities for merging clusters - especially when the distance threshold for cluster membership is set to a very high value - , each of them resulting in a different clustering.
- For a high threshold, the algorithm has the tendency to group all entities with semantic class 0 or semantic class 2 in one cluster.

3.2 Fuzzy Clustering Bergler et al. (FC-B)

Another promising approach found in the literature [2] considers noun phrase coreference resolution as a fuzzy clustering task because of the ambiguity typically found in natural language and the difficulty of solving the coreferents with absolute certainty. In this algorithm each entity initially forms its own cluster (whose medoid it is). Each other entity is assigned to all of the initial clusters by computing the distance (conform eq. 1 normalized) between it and the medoid of the cluster. [2] uses additional WordNet [7] dependent heuristics for computing the distance metric, which we did not include in our implementation. As a result each entity has a fuzzy membership with each cluster, forming a fuzzy coreference chain (a fuzzy set). Each fuzziness number is in [0,1]. Our implementation of fuzziness is based on the distance function and so it indicates how far we are from coreference. The medoid entity that originally formed the singleton cluster has a complete membership with itself or a distance of zero with itself. Then the chains are iteratively merged when their fuzzy set intersection is no larger than an a priori defined distance, in other words when there is at least one noun phrase in the fuzzy set of both clusters that has a distance smaller than a threshold with the respective medoids of the clusters. In the merged chain, the medoids of both original chains get complete membership and the membership of the other entities is updated by taking the minimum distance to the medoids. The merging continues until no chains can be merged anymore. The merging assumes that coreference resolution is symmetric and transitive. The algorithm has the effect of a single link hierarchical clustering. Once no more merging can be done, one can form hard clusters by defuzzification. We do that by assigning each entity to the cluster with whom it has the lowest fuzziness.

Beside the fuzzy representation, there are two main differences with [3]: 1) the chaining effect is larger because two clusters can be merged even without checking any pairwise incompatibilities of cluster objects; 2) in contrast with [3], the algorithm is independent of the order in which the clusters are merged.

3.3 Progressive Fuzzy Clustering (FC-P)

It seemed to us that the previous algorithms exhibit some good points and that a fuzzy clustering is appropriate for the noun phrase coreference task, but the fuzziness could be exploited in such a way that - in contrast with [2] - uncertainty of cluster membership of all fuzzy members plays a role in cluster merging. In addition, an algorithm that does not rely on a corpus-dependent distance threshold appears more practical.

The algorithm [9] is summarized in figure 1.

Our clustering algorithm is neither pure hard nor pure fuzzy. Initially, all the entities with semantic class 0 or 1 ($\in S_0$, $\in S_1$) form medoids of singleton clusters. Thus the number of clusters is equal to the number of entities $\in (S_0 \cup S_1)$. For each of the other entities ($\in S_2$, pronouns), a fuzzy membership value is calculated using the distance between the entity and the cluster. The fuzzy set of each cluster is used for merging two clusters. The idea behind the merging is that two clusters that corefer should have quite similar fuzzy sets. The possible noise from two or three entities in the fuzzy set must be smoothed by the rest of them.

```
MAIN

  1. Set each entity with semantic class 0 or 1 as the medoid of a singleton cluster.
  2. Merge those medoids which have appositive distance -∞.
  3. For each entity $NP_i$ with semantic class 2
         For each cluster $C_j$
             Compute $fuzziness(NP_i, C_j)$ (conform eq. 2).
  4. Repeat
       • Find the most similar clusters $C_i, C_j$.
           – clusters whose medoids have a distance of 0 or $-\infty$ or
           – clusters that have minimum output of $SIMILAR(C_i, C_j)$ and for which
             $ALL\_NPs\_COMPATIBLE(C_i, C_j)$
       • Merge the most similar clusters.
       • For each $NP_i$ with semantic class 2 ($S_2$)
             For each cluster $C_j$
                 Recompute $fuzziness(NP_i, C_j)$.
  5. Until cannot merge anymore.
  6. Defuzzify and assign to each entity of a cluster the coreference represented by the central medoid.

FUZZINESS($NP_i$,$C_j$)

  $$fuzziness(NP_i, C_j) = \frac{dist1(NP_i, C_j)}{\sum_k dist1(NP_i, C_k)}$$   (2)

  where
  $dist1(NP_i, C_j) =$
  $\sum_{f \in F \setminus position} w_f * function_f(NP_i, NP_k) +$
  $+ w_{position} * function_{position}(NP_i, NP_t)$
  where $NP_k$ is the central medoid of $C_j$ and, considering the feature text position, $NP_t$ is the closest entity
  to $NP_i$ which belongs to cluster $C_j$.

SIMILAR($C_i$,$C_j$)

  • $d = 0$
  • For each entity $NP_k$ with semantic class 2 for which
    $fuzziness(NP_k, C_i) < \infty$ and
    $fuzziness(NP_k, C_j) < \infty$

        $d = d + fuzziness(NP_k, C_i) - fuzziness(NP_k, C_j)$   (3)

  • Return $d$ or $\infty$ if no $NP_k$ has fuzziness less than $\infty$ with both $C_i$ and $C_j$.

MERGE($C_i$,$C_j$)

  • Add all medoids from $C_j$ to $C_i$.
  • Set the central medoid of the new $C_i$ as the central medoid of the two merging clusters with the
    lowest entity index, except when a proper name with a lower index appears in any of the clusters.
    In the last case, set it to the mentioned proper name.
  • Delete $C_j$.

DEFUZZIFY

  • For each entity $NP_i$ with semantic class 2 set
    $cluster(NP_i) = C_j$
    where $C_j = argmin_{C_k}(fuzziness(NP_i, C_k))$.

ALL_NPs_COMPATIBLE($C_i$,$C_j$)

  1. For all $NP_a \in C_i$
         For all $NP_b \in C_j$
             If $dist(NP_a, NP_b) = \infty$
             then return FALSE.
  2. Return TRUE.
```

Figure 1: Progressive Fuzzy Clustering Algorithm

To improve the performance two special cases are included in the algorithm.

1) *Appositive merging*: Appositives have much higher preferences than the other features. Thus all the appositives are merged immediately after the formation of the initial clusters.
2) *Restriction on pronoun coreferencing*: According to this restriction, no pronoun can corefer to an element of a cluster whose central medoid is occurring after it. This restriction however prohibits cataphoric references (e.g. *"After he saw the danger, Quayle got scared"*), but they appear quite rarely in texts.

The main resemblances and differences with the foregoing algorithms are:

- *Progressive nature*: As in [2], our fuzzy algorithm progressively updates the fuzzy membership after each merging of clusters. However it updates it differently, i.e., not by taking the minimum fuzziness of an entity in the merged clusters, but by recomputing the fuzzy membership of an entity in the new cluster (see eq. 2). This is necessary as it is not always possible to correctly resolve some features (e.g. the gender) of a name. Initially the pronouns (he and she) are assigned a fuzzy membership to the clusters. As clusters are merged the feature of the entity may be resolved and thus the fuzzy set membership may change completely.
- *Merging of clusters*: In contrast to [2] and [3], our criteria for merging clusters are different by restricting the merging of chains that have a non-pronoun phrase as medoid and by considering the similarity of the current fuzzy sets of the clusters.
- Conform to [3] but unlike [2], our algorithm does not merge clusters when members of the new cluster would be incompatible (except when their central medoids have a distance of 0 or $-\infty$).
- *Search for the best match*: Conform to [2], but unlike [3], the algorithm iteratively searches for the best match instead of the nearest match for merging clusters. Thus for the first example given in section 3.1: *"Robert Smith"* would be resolved to *"Smith"* and *"her"* would never be integrated with *"Smith"*.
- *Corpus-independent*: Unlike [3] and [2], the algorithm is corpus-independent as no threshold distance is used.

3.4 The Hard Variant (HC-V)

We also implemented the hard variant of the above progressive fuzzy clustering algorithm. The algorithm is summarized in figure 2.

We assign the entities of semantic class 2 (i.e. the pronouns) to their closest cluster at the completion of the merging process and not at its start, because we hope to be able to assign them more correctly in case of missing features (e.g., gender) of some of the entities. The difference of this hard algorithm with the foregoing fuzzy algorithm lies in the computation of the similarity between clusters when merging the most similar clusters. In our fuzzy algorithm we use the complete fuzzy set of the clusters in the computation of the similarity (eq. 2); in the hard algorithm we merge clusters with a group average hierarchical clustering scheme until no more clusters can be merged based

```
MAIN

1. Set each entity with semantic class 0 or 1 as the
   medoid of a singleton cluster.
2. Merge those medoids which have appositive distance -∞.
3. Repeat
     • Find the most similar clusters $C_i, C_j$.
         – clusters whose medoids have a distance
           of 0 or $-\infty$ or
         – clusters that have minimum output of
           $SIMILAR(C_i, C_j)$ and for which
           $ALL\_NPs\_COMPATIBLE(C_i, C_j)$.
     • Merge the most similar clusters.
4. Until cannot merge anymore.
5. For each $NP_i$ with semantic class 2 ($S_2$)
     For each cluster $C_j$
       Compute $fuziness^a(NP_i, C_j)$ conf. eq. 2
     $cluster(NP_i) = C_j$ where
     $C_j = argmin_{C_k}(fuziness(NP_i, C_k))$
6. Assign to each entity of a cluster the coreference
   represented by the central medoid.

SIMILAR($C_i$,$C_j$)

• $d = 0$, $count = 0$
• For each entity $NP_k$ of $C_i$
    For each entity $NP_l$ of $C_j$
      $d = d + dist(NP_k, NP_l)$ (conform eq. 3)
      $count = count + 1$
• Return $d/count$ (or $\infty$ if $d = \infty$)
```

[a]The name *fuziness* is used here for the sake of consistency with the previous apparition of the same formula and does not imply any reference to a fuzzy character of the algorithm.

Figure 2: Hard Clustering Variant

on incompatibility of members. The difference with the fuzzy clustering of [2] lies in the merging of entities that only belong to S_2 and the use of a group average hierarchical clustering scheme instead of a single linkage hierarchical clustering scheme in [2].

Preliminary tests showed that pronouns like *it*, *I*, *me*, *our* are very difficult to be resolved and they harm the performance. So in all the above clustering algorithms we considered them alone in singleton clusters, which is not a correct solution, but the results will be affected equally for all the algorithms.

4 Corpora and Evaluation

We used two corpora: one from the Document Understanding Conference 2002 [5], the other from the Message Understanding Conference 6 [8]. The DUC documents were selected from the category "biographies" and they are small texts (on average 3KB each) which contain many entities to be resolved (pronominal and non pronominal entities). We chose randomly ten documents from this set, parsed them in order to extract the entities (as the smallest noun phrases which do not contain embedded noun phrases) and annotated them manually for coreference. The MUC-6 proposed a training set of 30 documents (the so-called "dryrun set") and a test set of 30 documents (the "formaltest set"). They are all annotated with coreference information. In this corpus, both the smallest noun phrases and the ones which contain them are considered as entities. The features are extracted slightly differently for the two corpora, because of the different nature of the entities (e.g. the proper name feature is reduced to the capitalization of the head noun in the MUC corpora, while in the DUC subcorpus we look at two consecutive capitalized words).

The MUC-6 corpora contains few pronouns, which are the entities that are the most interesting for our algorithm. That's why the DUC subcorpus is useful for the evaluation, especially for the pronoun resolution. The DUC documents have an average of 18 pronouns comprised in the set *"he", "she", "him", "her", "they", "them"*, while the two MUC-6 corpora have only a mean of 2.97 and 4.86 respectively.

We computed automatically the precision and recall and combined them into the F-measure. Two algorithms were initially implemented to perform the evaluation: the one of Vilain et al. (referred in [1], employed in MUC) and the B-CUBED algorithm, described in [1].

In Vilain's algorithm, the recall is computed as follows:

$$R = \frac{\sum_i (|C_i| - |p(C_i)|)}{\sum (|C_i| - 1)}$$

where C_i is a manual cluster and $p(C_i)$ is a partition of C_i relative to the automatic clusters ($|p(C_i)|$ measures how many of the automatic clusters contain entities from the C_i). The precision is computed by inverting the roles of the manual and automatic clusters.

In the BCUBED algorithm, the recall is computed as follows:

$$R = \frac{\sum_i R_i}{n}$$

where R_i is the recall obtained for entity i and n is the total number of entities. The recall for entity i is defined as:

$$R_i = \frac{|HC_i \cap CC_i|}{|HC_i|}$$

where HC_i is the manual cluster in which entity i appears and CC_i is the automatic cluster in which entity i appears. The precision is computed by inverting the roles of the manual and automatic clusters.

F-measure combines equally the precision with the recall: $F = \frac{2 * P * R}{P + R}$

As observed by [1], the algorithm of Vilain yields unintuitive results for some cases, because it does not give any credit for separating out singletons and it considers all types of errors as equal. The algorithm favors big clusters, which is confirmed by the results of our second baseline, which clusters all entities in one cluster (see below) (Vilain F-measure is 86% for the dryrun set, 87% for the formaltest set and 60% for the DUC subcorpus). Because of this, we decided not to use the results obtained with the Vilain's algorithm and we considered only the BCUBED algorithm.

We separately evaluated pronoun coreference, by selecting as entities only pronouns and their immediate antecedents in the manual files. For certain text processing tasks, the correct resolution of pronouns is important. With the Vilain and BCUBED evaluation it is still possible to obtain reasonably good results when a set of pronouns are clustered together, while their antecedent is missing or is wrongly assigned in the cluster. So we computed also the accuracy of the antecedent resolution.

$$accuracy = \frac{\text{number of pronouns correctly resolved}}{\text{total number of pronouns}}$$

A pronoun is considered correctly resolved when its main coreferent in the automatic cluster (the medoid of the cluster) is in the same manual cluster as the pronoun itself and when this coreferent is not another pronoun (case appearing in HC-C and FC-B).

5 Results and Discussion

We tested the algorithms with the two corpora. We tried two baselines : every entity in a singleton cluster (BL1) and all entities in one cluster (BL2). For the hard clustering we used four different threshold values, determined experimentally: 8, 11.5, 16 and 20 (a small threshold corresponds to a conservative behavior: small clusters). For the fuzzy clustering, we used a threshold value of 0.2 and 0.5 (again, a small threshold is conservative).

5.1 All entities

The results using all the entities are summarized in table 2.

a)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.25	0.40
BL2	0.13	1.00	0.22
FC-P	0.81	0.52	0.62
HC-V	0.85	0.51	0.63
HC-C thrs.8	0.83	0.51	0.62
HC-C thrs 11.5	0.57	0.58	0.56
HC-C thrs. 16	0.52	0.61	0.55
HC-C thrs. 20	0.49	0.62	0.54
FC-B thrs. 0.2	0.83	0.51	0.62
FC-B thrs. 0.5	0.39	0.66	0.47

b)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.24	0.38
BL2	0.19	1.00	0.30
FC-P	0.78	0.48	0.58
HC-V	0.82	0.45	0.57
HC-C thrs.8	0.85	0.43	0.56
HC-C thrs 11.5	0.67	0.52	0.57
HC-C thrs. 16	0.60	0.54	0.55
HC-C. thrs. 20	0.58	0.56	0.56
FC-B thrs. 0.2	0.79	0.46	0.56
FC-B thrs. 0.5	0.40	0.65	0.46

c)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.56	0.72
BL2	0.07	1.00	0.13
FC-P	0.76	0.77	0.76
HC-V	0.85	0.72	0.78
HC-C thrs.8	0.79	0.67	0.72
HC-C thrs 11.5	0.45	0.73	0.55
HC-C thrs. 16	0.42	0.74	0.53
HC-C. thrs. 20	0.39	0.78	0.52
FC-B thrs. 0.2	0.88	0.67	0.76
FC-B thrs. 0.5	0.26	0.78	0.38

Table 2: Precision, recall and F-measure obtained considering all entities, using the different algorithms for a) dryrun subcorpus of the MUC-6 corpora, b) formaltest subcorpus of the MUC-6 corpora and c) DUC subcorpus.

Here follow a few remarks about the results:

1) The progressive fuzzy algorithm and its hard variant are among the best in terms of F-measure on all corpora.

2) For the HC-C and FC-B, the general tendency is to obtain better results for lower values of the threshold. A more conservative algorithm induces a higher precision, influencing positively the F-measure (although the recall decreases).

3) The difficulty to set an appropriate threshold can be observed in the MUC-6 corpora, since in the training corpus (dryrun), the best results were not obtained with the same threshold as in the test corpus (formaltest).

4) In contrast with precision, which is generally good, the recall values are quite low. We identified three types of errors responsible for the values and we analyzed more deeply the first one (see subsection *Experiment*).

- Wrong assignment of the semantic class. Pronouns (e.g. "he") come in the same clusters with entities like *Bentonville* or *Institute*, which are wrongly considered as person names. This type of errors might be resolved using a name entity recognition tool, able to semantically classify the entities.
- Acronym resolution. The algorithms are not able to relate acronyms with their corresponding long form (e.g. *"International Business Machines Corp." and "IBM"*). This problem is more frequent in MUC corpora (1.8% of the entities in the formaltest subcorpus are acronyms which can not be detected with Word-Substring feature or appositive - like in the case *"the Congressional Black Caucus (CBC)"*). We need to integrate an acronym resolution tool in order to correct this type of errors.
- Discourse structure. The texts contain a lot of direct speech, where the pronouns are very difficult to be resolved. For example, in the following phrase *"He spends most of his time talking to associates and customers," Shinkle said, "and he always comes back with many ideas from them"* (DUC subcorpus), the pronouns *"he"* and *"his"* are wrongly resolved as coreferring with *Shinkle*. This problem is more frequent in the DUC subcorpus (6.52% of the sentences are similar to the above example). A number of discourse specific heuristics could be added to the algorithm.
- Other errors. A number of other errors are due to different causes: lack of synonyms/hypernyms detection (*"Coca-Cola"*, *"Coke"*, *"million"*, *"cents"*), lack of knowledge of the world (*"the U.S."*, *"the country"*). These errors are more difficult to resolve.

5.1.1 Experiment

In order to quantify errors caused by the wrong assignment of the semantic class, we manually corrected this field and rerun the experiments. The new results are in table 3. The recall increases in all cases. The precision usually decreases for HC-C, which favors big clusters of entities with semantic class 0 (the number of such entities increases by correcting the semantic class), decreasing also the F-measure. For the other algorithms, F-measure usually increases. The influence on the results is larger for the MUC corpora than for the DUC subcorpus, because the number of entities with semantic class wrongly assigned was higher (20.98% for the dryrun and 19.29% for the formaltest comparing with 8.42% for the DUC subcorpus).

5.2 Pronouns

Evaluating just the pronouns, the results are summarized in table 4.

a)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.25	0.40
BL2	0.13	1.00	0.22
FC-P	0.77	0.55	0.62
HC-V	0.81	0.54	0.64
HC-C thrs. 8	0.81	0.56	0.65
HC-C thrs. 11.5	0.45	0.67	0.53
HC-C thrs. 16	0.40	0.70	0.50
HC-C thrs. 20	0.38	0.73	0.49
FC-B thrs. 0.2	0.79	0.55	0.63
FC-B thrs. 0.5	0.35	0.76	0.47

b)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.24	0.38
BL2	0.19	1.00	0.30
FC-P	0.78	0.55	0.62
HC-V	0.83	0.50	0.61
HC-C thrs. 8	0.83	0.47	0.59
HC-C thrs. 11.5	0.56	0.59	0.56
HC-C thrs. 16	0.50	0.61	0.54
HC-C. thrs. 20	0.49	0.65	0.54
FC-B thrs. 0.2	0.81	0.48	0.59
FC-B thrs. 0.5	0.42	0.72	0.51

c)

Algorithm	Precision	Recall	F-measure
BL1	1.00	0.56	0.72
BL2	0.07	1.00	0.13
FC-P	0.76	0.80	0.78
HC-V	0.86	0.72	0.78
HC-C thrs. 8	0.78	0.67	0.72
HC-C thrs. 11.5	0.38	0.76	0.51
HC-C thrs. 16	0.36	0.75	0.48
HC-C. thrs. 20	0.33	0.80	0.46
FC-B thrs. 0.2	0.88	0.67	0.76
FC-B thrs. 0.5	0.27	0.78	0.40

Table 3: Precision, recall and F-measure obtained considering all entities, using the different algorithms, starting with correct semantic classes for a) dryrun subcorpus of the MUC-6 corpora, b) formaltest subcorpus of the MUC-6 corpora and c) DUC subcorpus.

Here follow a few remarks about the pronoun resolution:

1) The results obtained for the DUC subcorpus are more representative than the ones obtained for the MUC-6 corpora, since the former contains more pronouns.

2) HC-C and FC-B work now better for higher values of the threshold. This can be explained by the fact that in all corpora most of the pronouns referred to the same person and so they should come out in the same cluster, which rewards big clusters.

3) Since in FC-P we give special attention to the pronouns (by iteratively recomputing the fuzzy vectors), we assume that this algorithm should outperform its hard variant on pronoun resolution. This assumption seems to be verified considering the F-measure, especially in the DUC subcorpus, which is the most representative for this evaluation, but is somehow contradicted by the accuracy results. We need additional tests to sustain our claim.

4) In the DUC subcorpus: although the F-measure is higher for HC-C with threshold 20 than for FC-P, the accuracy measure is lower. We believe that accuracy is a more fair measure for pronoun resolution, since correctly resolving the antecedent of the pronoun is more important than correctly grouping pronouns in the same cluster.

6 Conclusion and Future Work

In this paper we compared four clustering methods for coreference resolution: one progressive fuzzy, its hard variant and another two algorithms, taken from the literature. We evaluated them on two kinds of corpora, a standard one used in the coreference resolution task and another one containing more pronominal entities. Our algorithms are on top when all the entities are considered. For the pronoun resolution, our fuzzy algorithm oba)

Algorithm	Precision	Recall	F-measure	Acc
BL1	1.00	0.35	0.51	0.00
BL2	0.39	1.00	0.51	0.32
FC-P	0.93	0.47	0.62	0.09
HC-V	0.94	0.47	0.61	0.14
HC-C thrs. 8	0.94	0.40	0.55	0.00
HC-C thrs. 11.5	0.87	0.41	0.56	0.00
HC-C thrs. 16	0.87	0.41	0.56	0.00
HC-C. thrs. 20	0.84	0.46	0.58	0.09
FC-B thrs. 0.2	0.93	0.41	0.56	0.00
FC-B thrs. 0.5	0.81	0.43	0.55	0.00

b)

Algorithm	Precision	Recall	F-measure	Acc
BL1	1.00	0.27	0.41	0.00
BL2	0.51	1.00	0.62	0.36
FC-P	0.88	0.43	0.55	0.17
HC-V	0.90	0.42	0.54	0.20
HC-C thrs. 8	0.93	0.34	0.48	0.00
HC-C thrs. 11.5	0.89	0.38	0.51	0.01
HC-C thrs. 16	0.88	0.39	0.52	0.01
HC-C. thrs. 20	0.86	0.47	0.58	0.13
FC-B thrs. 0.2	0.91	0.34	0.48	0.00
FC-B thrs. 0.5	0.77	0.40	0.50	0.00

c)

Algorithm	Precision	Recall	F-measure	Acc
BL1	1.00	0.18	0.31	0.00
BL2	0.46	1.00	0.62	0.46
FC-P	0.87	0.55	0.67	0.41
HC-V	0.92	0.45	0.59	0.42
HC-C thrs. 8	0.91	0.32	0.47	0.00
HC-C thrs. 11.5	0.87	0.48	0.62	0.00
HC-C thrs. 16	0.86	0.51	0.63	0.06
HC-C thrs. 20	0.82	0.62	0.70	0.22
FC-B thrs. 0.2	0.91	0.32	0.47	0.00
FC-B thrs. 0.5	0.78	0.53	0.62	0.06

Table 4: Precision, recall and F-measure obtained considering just the pronouns, using the different algorithms for a) dryrun subcorpus of the MUC-6 corpora, b) formaltest subcorpus of the MUC-6 corpora and c) DUC subcorpus.

tains competitive or better F-measure results compared with the two algorithms from the literature and outperforms both of them in term of accuracy. These results are obtained despite the fact that our algorithms do not rely on a threshold distance value for cluster membership, which makes them corpus-independent. In the future we plan to perform more experiments with different types of texts and to enlarge the feature set based on current linguistic theories. We also plan to integrate the noun phrase coreference tool in our text summarization system.

References

[1] Baldwin B. et al. (1998) *Description of the UPENN CAMP System as Used for Coreference.* In Proceedings of the Seventh Message Understanding Conference (MUC-7), April 29-May 1, Fairfax, VA, USA. http://www.icl.pku.edu.cn/bswen/nlp/www.muc.saic.com/muc_7_toc.html.

[2] Bergler S. et al. (2003) *Using Knowledge-poor Coreference Resolution for Text Summarization.* In Proceedings of Document Understanding Conference 2003, May 31-June 1, NIST, USA, 85-92.

[3] Cardie C. and Wagstaff K. (1999). *Noun Phrase Coreference as Clustering.* In Proceedings of the Joint Conference on Empirical Methods in NLP and Very Large Corpora. http://citeseer.nj.nec.com/article/cardie99noun.html.

[4] Charniak E. (1999). *A Maximum-Entropy Inspired Parser.* Technical Report CS-99-12, Brown University, August. http://citeseer.nj.nec.com/ charniak99maximumentropyinspired.html.

[5] Document Understanding Conference *http://www-nlpir.nist.gov/projects/duc/guidelines/2002.html.*

[6] *http://www.census.gov/genealogy/names.*

[7] Miller G.A. et al. (1990) *Introduction to WordNet: An On-line Lexical Database.* International Journal of Lexicography (special issue), 3(4), 235-312.

[8] Message Understanding Conference *http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html.*

[9] Mitra R. et al. (2003) *Progressive fuzzy clustering for noun phrase coreference resolution.* In Proceedings of the 4'th Dutch-Belgian Information Retrieval Workshop, December 8-9, Amsterdam, The Nederlands, 19-30.