A Global Relaxation Labeling Approach to Coreference Resolution

Emili Sapena, Lluís Padró and Jordi Turmo*
TALP Research Center
Universitat Politècnica de Catalunya
{esapena, padro, turmo}@lsi.upc.edu

Abstract

This paper presents a constraint-based graph partitioning approach to coreference resolution solved by relaxation labeling. The approach combines the strengths of groupwise classifiers and chain formation methods in one global method. Experiments show that our approach significantly outperforms systems based on separate classification and chain formation steps, and that it achieves the best results in the state of the art for the same dataset and metrics.

1 Introduction

Coreference resolution is a natural language processing task which consists of determining the *mentions* that refer to the same entity in a text or discourse. A mention is a noun phrase referring to an entity and includes named entities, definite noun phrases, and pronouns. For instance, "Michael Jackson" and "the youngest of Jackson 5" are two mentions referring to the same entity.

A typical machine learning-based coreference resolution system usually consists of two steps: (i) classification, where the system evaluates the *coreferentiality* of each pair or group of mentions, and (ii) formation of chains, where given the confidence values of the previous classifications the system forms the coreference chains.

Research supported by the Spanish Science and Innovation Ministry, via the KNOW2 project (TIN2009-14715-C04-04) and from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement number 247762 (FAUST)

Regarding the classification step, pioneer systems developed were based on *pairwise* classifiers. Given a pair of mentions, the process generates a feature vector and feeds it to a classifier. The resolution is done by considering each mention of the document as *anaphor*[1] and looking backward until the *antecedent* is found or the beginning of the document is reached (Aone and Bennett, 1995; McCarthy and Lehnert, 1995; Soon et al., 2001).

A first approach towards *groupwise* classifiers is the twin-candidate model (Yang et al., 2003). The model faces the problem as a competition between two candidates to be the antecedent of the anaphor into account. Each candidate mention is compared with all the others in a round robin contest. Following the *groupwise* approach, rankers consider all the possible antecedent mentions at once (Denis and Baldridge, 2008). Rankers can obtain more accurate results due to a more informed context where all candidate mentions are considered at the same time.

Coreference chains are formed after classification. Many systems form the chains by joining each positively-classified pair (i.e. *single-link*) or with simple improvements such as linking an anaphor only to its antecedent with maximum confidence value (Ng and Cardie, 2002).

Some works propose more elaborated methods than single-link for chain formation. The approaches used are Integer Linear Programming

[1] Typically a pair of coreferential mentions m_i and m_j ($i < j$) are called antecedent and anaphor respectively, though m_j may not be anaphoric.

(ILP) (Denis and Baldridge, 2007; Klenner and Ailloud, 2009; Finkel and Manning, 2008), graph partitioning (Nicolae and Nicolae, 2006), and clustering (Klenner and Ailloud, 2008). The main advantage of these types of *post-processes* is the enforcement of transitivity sorting out the contradictions that the previous classification process may introduce.

Although chain formation processes search for global consistency, the lack of contextual information in the classification step is propagated forward. Few works try to overcome the limitations of keeping classification and chain formation apart. Luo et al. (2004) search the most probable path comparing each mention with the partial-entities formed so far using a Bell tree structure. McCallum and Wellner (2005) propose a graph partitioning cutting by distances, with the peculiarity that distances are learned considering coreferential chains of the labeled data instead of pairs. Culotta et al. (2007) combine a groupwise classifier with a clustering process in a First-Order probabilistic model.

The approach presented in this paper follows the same research line of joining group classification and chain formation in the same step. Concretely, we propose a graph representation of the problem solved by a relaxation labeling process, reducing coreference resolution to a graph partitioning problem given a set of constraints. In this manner, decisions are taken considering the whole set of mentions, ensuring consistency and avoiding that classification decisions are independently taken. Our experimental results on the ACE dataset show that our approach outperforms systems based on separate classification and chain formation steps, and that it achieves the best results in the state of the art for the same dataset and metrics.

The paper is organized as follows. Section 2 describes the graph representation of the task. Section 3 explains the use of relaxation labeling algorithm and the machine learning process. Finally, experiments and results are explained in Section 4 before paper is concluded.

2 Graph Representation

Let $G = G(V, E)$ be an undirected graph where V is a set of vertices and E a set of edges. Let $\mathbf{m} = (m_1, ..., m_n)$ be the set of mentions of a document with n mentions to resolve. Each mention m_i in the document is represented as a vertex $v_i \in V$. An edge $e_{ij} \in E$ is added to the graph for pairs of vertices (v_i, v_j) representing the possibility that both mentions corefer. The list of adjacent vertices of a vertex v_i is $A(v_i)$.

Let C be our set of constraints. Given a pair of mentions (m_i, m_j), a subset of constraints $C_{ij} \subseteq C$ restrict the compatibility of both mentions. C_{ij} is used to compute the weight value of the edge connecting v_i and v_j. Let $w_{ij} \in W$ be the weight of the edge e_{ij}:

$$w_{ij} = \sum_{k \in C_{ij}} \lambda_k f_k(m_i, m_j) \quad (1)$$

where $f_k(\cdot)$ is a function that evaluates the constraint k. And λ_k is the weight associated to the constraint k (λ_k and w_{ij} can be negative).

In our approach, each vertex (v_i) in the graph is a variable (v_i) for the algorithm. Let L_i be the number of different values (labels) that are possible for v_i. The possible labels of each variable are the partitions that the vertex can be assigned. Note that the number of partitions (entities) in a document is unknown, but it is at most the number of vertices (mentions), because in a extreme case, each mention in a document could be referring to a different entity. A vertex with index i can be in the first i partitions (i.e. $L_i = i$).

Each combination of labelings for the graph vertices is a partitioning (Ω). The resolution process searches the partitioning Ω^* which optimizes the goodness function $F(\Omega, W)$, which depends on the edge weights W. In this manner, Ω^* is optimal if:

$$F(\Omega^*, W) \geq F(\Omega, W), \forall \Omega \quad (2)$$

The next section describes the algorithm used in the resolution process.

3 Relaxation Labeling

Relaxation labeling (Relax) is a generic name for a family of iterative algorithms which perform

function optimization, based on local information. The algorithm has been widely used to solve NLP problems such as PoS-tagging (Màrquez et al., 2000), chunking, knowledge integration, and Semantic Parsing (Atserias, 2006).

Relaxation labeling solves our weighted constraint satisfaction problem dealing with the edge weights. In this manner, each vertex is assigned to a partition satisfying as many constraints as possible. To do that, the algorithm assigns a probability for each possible label of each variable. Let $\mathbf{H} = (\mathbf{h}^1, \mathbf{h}^2, \ldots, \mathbf{h}^n)$ be the weighted labeling to optimize, where each $\mathbf{h}^i$ is a vector containing the probability distribution of v_i, that is: $\mathbf{h}^i = (h_1^i, h_2^i, \ldots, h_{L_i}^i)$. Given that the resolution process is iterative, the probability for label l of variable v_i at time step t is $h_l^i(t)$, or simply h_l^i when the time step is not relevant.

The support for a pair variable-label (S_{il}) expresses how compatible is the assignment of label l to variable v_i considering the labels of adjacent variables and the edge weights. Although several support functions may be used (Torras, 1989), we chose the following one, which defines the support as the sum of the edge weights that relate variable v_i with each adjacent variable v_j multiplied by the weight for the same label l of v_j:

$$S_{il} = \sum_{j \in A(v_i)} w_{ij} \times h_l^j \qquad (3)$$

where w_{ij} is the edge weight obtained in Equation 1. In our version of the algorithm, $A(v_i)$ is the list of adjacent vertices of v_i but only including the ones with an index $k < i$. Consequently, the weights only have influence in one direction which is equivalent to using a directed graph. Although the proposed representation is based on a general undirected graph, preliminary experiments showed that using directed edges yields higher perfomance in this particular problem.

The aim of the algorithm is to find a weighted labeling such that global consistency is maximized. Maximizing global consistency is defined as maximizing the average support for each variable. Formally, $\mathbf{H}^*$ is a consistent labeling if:

Initialize:
$H := H_0$,

Main loop:
repeat
For each variable v_i
For each possible label l for v_i
$S_{il} = \sum_{j \in A(v_i)} w_{ij} \times h_l^j$
End for
For each possible label l for v_i
$h_l^i(t+1) = \frac{h_l^i(t) \times (1 + S_{il})}{\sum_{k=1}^{L_i} h_k^i(t) \times (1 + S_{ik})}$
End for
End for
Until no more significant changes

Figure 1: Relaxation labeling algorithm

$$\sum_{l=1}^{L_i} h_l^{*i} \times S_{il} \geq \sum_{l=1}^{L_i} h_l^i \times S_{il} \quad \forall \mathbf{h}, \forall i \qquad (4)$$

A partitioning Ω is directly obtained from the weighted labeling $\mathbf{H}$ assigning to each variable the label with maximum probability. The supports and the weighted labeling depend on the edge weights (Equation 3). To satisfy Equation 4 is equivalent to satisfy Equation 2. Many studies have been done towards the demonstration of the consistency, convergence and cost reduction advantages of the relaxation algorithm (Rosenfeld et al., 1976; Hummel and Zucker, 1987; Pelillo, 1997). Although some of the conditions required by the formal demonstrations are not fulfilled in our case, the presented algorithm –that forces a stop after a number of iterations– has proven useful for practical purposes.

Figure 1 shows the pseudo-code of the relaxation algorithm. The process updates the weights of the labels in each step until convergence. The convergence is met when no more significant changes are done in an iteration. Specifically, when the maximum change in an update step ($max_{i,l}(|h_l^i(t+1) - h_l^i(t)|)$) is lower than a parameter ϵ, a small value (0.001 in our experiments), or a fixed number of iterations is reached (2000 in our experiments). Finally, the assigned label for a variable is the one with the highest weight. Figure 2 shows a representation.

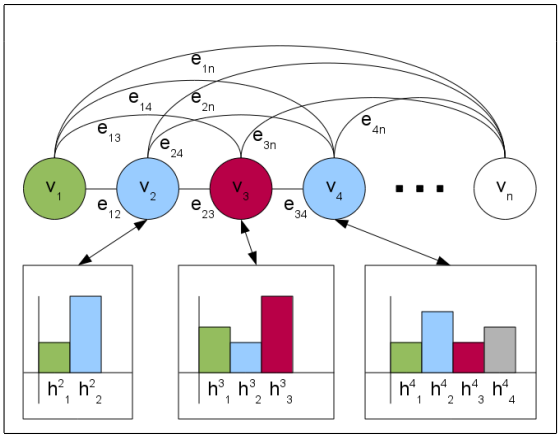

Figure 2: Representation of Relax. The vertices representing mentions are connected by weighted edges e_{ij}. Each vertex has a vector $\mathbf{h}^i$ of probabilities to belong to different partitions. The figure shows $\mathbf{h}^2$, $\mathbf{h}^3$ and $\mathbf{h}^4$.

DIST: Distance between m_i and m_j in sentences: number
DIST_MEN: Distance between m_i and m_j in mentions: number
APPOSITIVE: One mention is in apposition with the other: y,n
I/J_IN_QUOTES: $m_{i/j}$ is in quotes or inside a NP or a sentence in quotes: y,n
I/J_FIRST: $m_{i/j}$ is the first mention in the sentence: y,n
I/J_DEF_NP: $m_{i/j}$ is a definitive NP: y,n
I/J_DEM_NP: $m_{i/j}$ is a demonstrative NP: y,n
I/J_INDEF_NP: $m_{i/j}$ is an indefinite NP: y,n
STR_MATCH: String matching of m_i and m_j: y,n
PRO_STR: Both are pronouns and their strings match: y,n
PN_STR: Both are proper names and their strings match: y,n
NONPRO_STR: String matching like in Soon et al. (2001) and mentions are not pronouns: y,n
HEAD_MATCH: String matching of NP heads: y,n
NUMBER: The number of both mentions match: y,n,u
GENDER: The gender of both mentions match: y,n,u
AGREEMENT: Gender and number of both mentions match: y,n,u
I/J_THIRD_PERSON: $m_{i/j}$ is 3rd person: y,n
PROPER_NAME: Both mentions are proper names: y,n,u
I/J_PERSON: $m_{i/j}$ is a person (pronoun or proper name in a list): y,n
ANIMACY: Animacy of both mentions match (persons, objects): y,n
I/J_REFLEXIVE: $m_{i/j}$ is a reflexive pronoun: y,n
I/J_TYPE: $m_{i/j}$ is a pronoun (p), entity (e) or nominal (n)
NESTED: One mention is included in the other: y,n
MAXIMALNP: Both mentions have the same NP parent or they are nested: y,n
I/J_MAXIMALNP: $m_{i/j}$ is not included in any other mention: y,n
I/J_EMBEDDED: $m_{i/j}$ is a noun and is not a maximal NP: y,n
BINDING: Conditions B and C of binding theory: y,n
SEMCLASS: Semantic class of both mentions match: y,n,u (the same as Soon et al. (2001))
ALIAS: One mention is an alias of the other: y,n,u (only entities, else unknown)

Figure 3: Feature functions used

3.1 Constraints

The performance of the resolution process depends on the edge weights obtained by a set of weighted constraints (Equation 1). Any method or combination of methods to generate constraints can be used. For example, a set of constraints handwritten by linguist experts can be added to another automatically obtained set.

This section explains the automatic constraint generation process carried out in this work, using a set of feature functions and a training corpus. Màrquez et al. (2000) have successfully used similar processes to acquire constraints for constraint satisfaction algorithms.

Each pair of mentions (m_i, m_j) in a training document is evaluated by a set of feature functions (Figure 3). The values returned by these functions form a positive example when the pair of mentions corefer, and a negative one otherwise. Three specialized models are constructed depending on the type of anaphor mention (m_j) of the pair: pronoun, named entity or nominal.

For each specialized model, a decision tree (DT) is generated and a set of rules is extracted with C4.5 rule-learning algorithm (Quinlan, 1993). These rules are our set of constraints. The C4.5rules algorithm generates a set of rules for each path from the learnt tree. It then generalizes the rules by dropping conditions.

The weight assigned to a constraint (λ_k) is its precision over the training data (P_k), but shifted to be zero-centered: $\lambda_k = P_k - 0.5$.

3.2 Pruning

Analyzing the errors of development experiments, we have found two main error patterns that can be solved by a pruning process. First, the contribution of the edge weights for the resolution depends on the size of the document. And second, many weak edge weights may sum up to produce a bias in the wrong direction.

The weight of an edge depends on the weights assigned for the constraints which apply to a pair of mentions according to Equation 1. Each vertex is adjacent to all the other vertices. This produces that the larger the number of adjacencies, the smaller the influence of a constraint is. A consequence is that resolution for large and short documents has different results.

Many works have to deal with similar problems, specially the ones looking backward for antecedents. The larger the document, the more pos-

sible antecedents the system has to classify. This problem is usually solved looking for antecedents in a window of few sentences, which entails an evident limitation of recall.

Regarding the weak edge weights, it is notable that some kind of mention pairs are very weakly informative. For example, the pairs (pronoun, pronoun). Many stories have a few main characters which monopolize the pronouns of the document. This produces many positive training examples for pairs of pronouns matching in gender and person, which may lead the algorithm to produce large coreferential chains joining all these mentions even for stories where there are many different characters. For example, we have found in the results of some documents a huge coreference chain including every pronoun "he". This is because a pair of mentions ("he", "he") is usually linked with a small positive weight. Although the highest adjacent edge weight of a "he" mention may link with the correct antecedent, the sum of several edge weights linking the mention with other "he" causes the problem.

A pruning process is perfomed solving both problems and reducing computational costs from $O(n^3)$ to $O(n^2)$. For each vertex's adjacency list $A(v_i)$, only a maximum of N edges remain and the others are pruned. Concretely, the N/2 edges with largest positive weight and the N/2 with largest negative weight. The value of N is empirically chosen by maximizing performances over training data. On the one hand, the pruning forces the maximum adjacency to be constant and the contribution of the edge weights does not depend on the size of the document. On the other hand, most edges of the less informative pairs are discarded avoiding further confusion. There are no limitations in distance or other restrictions which may cause a loss of recall.

3.3 Initial State

The initial state of the vertices define the *a priori* probabilities for each vertex to be in each partition. There are several possible initial states. In the case where no prior information is available, a random or uniformly distributed state is commonly used. However, a well-informed initial state should drive faster the relaxation process to a better solution. This section describes the well-informed initial state chosen in our approach and the random one. Both are compared in the experiments (Section 4.2).

The well-informed initial state favors the creation of new chains. Variable v_i has $L_i = i$ possible values while variable v_{i+1} has $L_i + 1$. The probability distribution of v_{i+1} is equiprobable for values from 1 to L_i but it is the double for the probability to start a new chain $L_i + 1$.

$$h^i_l = \frac{1}{L_i+1}, \qquad \forall l = 1..L_i - 1$$
$$h^i_{L_i} = \frac{2}{L_i+1}$$

Pronouns do not follow this distribution but a totally equiprobable one, given that they are usually anaphoric.

$$h^i_l = \frac{1}{L_i}, \qquad \forall l = 1..L_i$$

This configuration enables the resolution process to determine as singletons the mentions for which little evidence is available. This small difference between initial probability weights is also introduced in order to avoid exceptional cases where all support values contribute with the same value.

The random initial state is also used in our experiments to test that our proposed configuration is better-informed than random. Given the equiprobability state, we add a random value to each probability to be in a partition:

$$h^i_l = \frac{1}{L_i} + \epsilon_{il}, \qquad \forall l = 0..L_i$$

where ϵ_{il} is a random value $\frac{-1}{2L_i} \leq \epsilon_{il} \leq \frac{1}{2L_i}$. These little random differences may help the algorithm to avoid local minima.

3.4 Reordering

The vertices of the graph would usually be placed in the same order as the mentions are found in the document (*chronological*). In this manner, v_i corresponds to m_i. However, as suggested by Luo (2007), there is no need to generate the model following that order. In our approach, the first variables have a lower number of possible labels. Moreover, an error in the first variables has more influence on the performance than an error in the later ones. Placing named entities at the beginning is reasonably to expect that is helpful for the algorithm, given that named entities are usually the most informative mentions.

	Tokens	Mentions	Entities
bnews train	66627	9937	4408
bnews test	17463	2579	1040
npaper train	68970	11283	4163
npaper test	17404	2483	942
nwire train	70832	10693	4297
nwire test	16772	2608	1137

Figure 4: Statistics about ACE-phase02

Suppose we have three mentions appearing in this order somewhere in a document: "A. Smith", "he", "Alice Smith". For proximity, mention "he" may tend to link with "A. Smith". Then, the third mention "Alice Smith" clearly is the whole name of "A. Smith" but the gender with "he" does not agree. Given that our implementation acts like a directed graph only looking backward (see Section 3), mention "he" won't change its tendency and it may cause a split in the "Alice Smith" coreference chain. However, having named entities in first place and pronouns at the end, enables the mention "he" to determine that "A. Smith" and "Alice Smith" having the same label are not good antecedents.

Reordering only affects on the number of possible labels of the variables and the list of adjacencies $A(v_i)$. The chronological order of the document is taken into account by the constraints regardless of the graph representation. Our experiments confirm (Section 4) that placing first named entity mentions, then nominal mentions and finally the pronouns, the precision increases considerably. Inside of each of these groups, the order is the same order of the document.

4 Experiments and Results

We evaluate our approach to coreference resolution using ACE-phase02 corpus, which is composed of three sections: Broadcast News (BNEWS), Newswire (NWIRE) and Newspaper (NPAPER). Each section is in turn composed of a training set and a test set. Figure 4 shows some statistics about this corpus.

In our experiments, we consider the *true mentions* of ACE. This is because our focus is on evaluating pairwise approach versus the graph partitioning approach and also comparing them to some state-of-the-art approaches which also use true mentions. Moreover, details on mention identifier systems and their performances are rarely published by the systems based on automatic identification of mentions and it difficults the comparison.

To evaluate our system we use CEAF (Luo, 2005) and B^3 (Bagga and Baldwin, 1998). CEAF is computed based on the best one-to-one map between key coreference chains and response ones. We use the mention-based similarity metric which counts the number of common mentions shared by key coreference chains and response ones. As we are using *true mentions* for the experiments, precision, recall and F_1 are the same value and only F_1 is shown. B^3 scorer is used for comparison reasons. B^3 algorithm looks at the presence/absence of mentions for each entity in the system output. Precision and recall numbers are computed for each mention, and the average gives the final precision and recall numbers.

MUC scorer (Vilain et al., 1995) is not used in our experiments. Although it has been widely used in the state of the art, we consider the newer metrics have overcome some MUC limitations (Bagga and Baldwin, 1998; Luo, 2005; Klenner and Ailloud, 2008; Denis and Baldridge, 2008).

Our preprocessing pipeline consists of FreeLing (Atserias et al., 2006) for sentence splitting and tokenization, SVMTool (Gimenez and Marquez, 2004) for part of speech tagging and BIO (Surdeanu et al., 2005) for named entity recognition and classification. No lemmatization neither syntactic analysis are used.

4.1 Baselines

4.1.1 DT with automatic feature selection

The baseline developed in our work is based on Soon et al. (2001) with the improvements of Ng and Cardie (2002), which uses a Decision Tree (DT). Many research works use the same references in order to evaluate possible improvements done by their new models or by the incorporation of new features.

The features used in the baseline are the same than those used in our proposed system (Figure 3). However, some features are noisy and many others have redundancy which causes low performances using DTs. In order to select the best set

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Model	F_1	F_1	F_1	F_1	P	R	F_1
DT	60.6	57.8	60.5	59.7	61.0	74.1	66.9
DT Hill	67.8	61.6	65.0	**64.8**	74.7	69.8	**72.2**

Table 1: Results ACE-phase02. Comparing baselines based on Decision Trees.

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Model	F_1	F_1	F_1	F_1	P	R	F_1
DT	60.6	59.5	64.7	61.7	63.3	74.7	68.5
DT + ILP	62.8	60.3	63.7	**62.5**	72.4	69.2	**70.7**
DT Hill	67.8	63.2	67.2	**66.5**	76.8	71.0	**73.8**
DT Hill + ILP	67.6	63.5	66.7	66.3	80.0	68.3	73.7
Relax	69.5	68.3	73.0	**70.4**	86.5	67.9	**76.1**

Table 2: Results on documents shorter than 200 mentions of ACE-phase02

of features a Hill Climbing process has been performed doing a five-fold cross-validation over the training corpus. A similar feature selection process has been done by Hoste (2005).

The Hill Climbing process starts using the whole set of features. A cross-validation is done (un)masking each feature. The (un)masked feature with more improvement is (added to) removed from the set. The process is repeated until an iteration without improvements is reached.

Note that this optimization process is biased by the metric used to evaluate each feature combination. We use CEAF in our experiments, which encourages precision and consistency.

4.1.2 Integer Linear Programming

The second baseline developed forms the coreference chains given the output of the pair classification of the first baseline. A set of binary variables (x_{ij}) symbolize whether pairs of mentions (m_i, m_j) corefer ($x_{ij} = 1$) or not ($x_{ij} = 0$). An objective function is defined as follows:

$$min \sum_{i<j} -log(Pc_{ij})x_{ij} - log(1 - Pc_{ij})(1 - x_{ij})$$

where Pc_{ij} is the confidence value of mentions m_i and m_j to corefer obtained by the pair classifier. The minimization of the objective function is done by Integer Linear Programming (ILP) in a similar way to (Klenner, 2007; Denis and Baldridge, 2007; Finkel and Manning, 2008). In order to keep consistency in the results, which is the goal of this *post-process*, a set of *triangular* constraints is required. For each three mentions with indexes $i < j < k$ the corresponding variables have to satisfy three constraints:

- $x_{ik} \geq x_{ij} + x_{jk} - 1$
- $x_{ij} \geq x_{ik} + x_{jk} - 1$
- $x_{jk} \geq x_{ij} + x_{ik} - 1$

This implies that this model needs, for a document with n mentions, $\frac{1}{2}n(n-1)$ variables and $\frac{1}{2}n(n-1)(n-2)$ constraints to assure consistency[2]. This is an important limitation with a view to scalability. In our experiments only documents shorter than 200 mentions can be solved by this baseline due to its computational cost.

4.2 Experiments

Four experiments have been done in order to evaluate our proposed approach. This section describes and analyzes the results of each experiment. Finally, our performances are compared with the state of the art.

The first experiment compares the performances of our baselines (Table 1). "DT" is the system based on Decision Tree using all the features of Figure 3 and "DT+Hill" is a DT using the features selected by the Hill Climbing process (Section 4.1.1). There is a significant improvement in the performances (5.1 points with CEAF, 5.3 with B^3) after the automatic feature selection process is done.

[2] $\frac{1}{6}n(n-1)(n-2)$ for each one of the three triangular constraints

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Model	F1	F1	F1	F1	P	R	F1
Relax	67.3	64.4	69.5	67.2	**88.4**	62.7	73.3
Relax pruning	68.6	65.2	70.1	68.0	82.3	**66.9**	73.8
Relax pruning & reorder	**69.5**	**67.3**	**72.1**	**69.7**	85.3	66.8	**74.9**
Relax random IS	68.2	66.1	71.0	68.5	83.5	66.7	74.2
MaxEnt+ILP (Denis, 2007)	-	-	-	66.2	81.4	65.6	72.7
Rankers (Denis, 2007)	65.7	65.3	68.1	67.0	79.8	66.8	72.7

Table 3: Results ACE-phase02.

In the second experiment the ILP chain formation process is applied using the output of both DTs. Results are shown in Table 2. Note that ILP only applies to documents shorter than 200 mentions due to its excessive computational cost (Section 4.1.2). Results for Relax applied to the same documents are also included for comparison. ILP forces consistency of the results producing an increase in precision score with B^3 metric in both cases. However, "DT+Hill" has been optimized for CEAF metric which encourages precision and consistency. For this, a post-process forcing consistency seems unnecessary for a classifier already optimized. Relax significantly outperforms all the baselines.

The third experiment shows the improvements achieved by the use of pruning and reordering techniques (Sections 3.2 and 3.4). Table 3 shows the results. Pruning improves performances with both metrics. B^3 precision is decreased but the global F_1 is increased due to a considerably improvement of recall. Reordering recovers the precision lost by the pruning without loosing recall, which achieves the best performances of 69.7 with CEAF and 74.9 with B^3.

The fourth experiment evaluates the influence of the initial state. A comparison is done with the proposed initial state (Section 3.3) and the random one. The results shown in Table 3 for random initial state are the average of 3 executions. The system called "Relax random IS" is using the same values for pruning and reordering techniques than the best result of previous experiment: "Relax pruning & reorder". As expected, results with a well-informed initial state outperform the random ones.

Finally, Relax performances are compared with the best scores we have found using the same corpora and metrics. We compare our approach with specialized Rankers –groupwise classifier–, and a system using ILP not only forcing consistency but also using information about anaphoricity and named entities. Relax outperforms both systems with both metrics (Table 3).

5 Conclusion

The approach for coreference resolution presented in this paper is a constraint-based graph partitioning solved by relaxation labeling.

The decision to join or not a set of mentions in the same entity is taken considering always the whole set of previous mentions like in groupwise classifiers. Contrarily to the approaches where variables are the linkage of each pair of mentions, in this model consistency is implicitly forced. Moreover, the influence of the partial results of the other mentions at the same time avoids that decisions are independently taken.

The capacity to easily incorporate constraints from different sources and using different knowledge is also remarkable. This flexibility gives a great potencial to the approach. Anaphoricity filtering is not needed given that the necessary knowledge can be also introduced by constraints.

In addition, three tecniques to improve results have been presented: reordering, pruning and feature selection by Hill Climbing. The experiments confirm their utility.

The experimental results clearly outperform the baselines with separate classification and chain formaiton. The approach also outperforms others in the state of the art using same corpora and metrics.

References

Aone, C. and S.W. Bennett. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 33rd annual meeting on ACL*, pages 122–129.

Atserias, J., B. Casas, E. Comelles, M. González, L. Padró, and M. Padró. 2006. Freeling 1.3: Syntactic and semantic services in an open-source nlp library. In *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC 2006), ELRA*. Genoa, Italy.

Atserias, J. 2006. *Towards Robustness in Natural Language Understanding*. Ph.D. Thesis, Dept. Lenguajes y Sistemas Informáticos. Euskal Herriko Unibertsitatea. Donosti. Spain.

Bagga, A. and B. Baldwin. 1998. Algorithms for scoring coreference chains. *Proceedings of the Linguistic Coreference Workshop at LREC*, pages 563–566.

Culotta, A., M. Wick, and A. McCallum. 2007. First-Order Probabilistic Models for Coreference Resolution. *Proceedings of NAACL HLT*, pages 81–88.

Denis, P. and J. Baldridge. 2007. Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. *Proceedings of NAACL HLT*, pages 236–243.

Denis, P. and J. Baldridge. 2008. Specialized models and ranking for coreference resolution. *Proceedings of the EMNLP, Hawaii, USA*.

Denis, P. 2007. *New Learning Models for Robust Reference Resolution*. Ph.D. dissertation, University of Texas at Austin.

Finkel, J.R. and C.D. Manning. 2008. Enforcing transitivity in coreference resolution. In *Proceedings of the 46th Annual Meeting of the ACL HLT: Short Papers*, pages 45–48. Association for Computational Linguistics.

Gimenez, J. and L. Marquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 43–46.

Hoste, V. 2005. *Optimization Issues in Machine Learning of Coreference Resolution*. PhD thesis.

Hummel, R. A. and S. W. Zucker. 1987. On the foundations of relaxation labeling processes. pages 585–605.

Klenner, M. and É. Ailloud. 2008. Enhancing Coreference Clustering. In *Proceedings of the Second Workshop on Anaphora Resolution*. WAR II.

Klenner, M. and E. Ailloud. 2009. Optimization in Coreference Resolution Is Not Needed: A Nearly-Optimal Algorithm with Intensional Constraints. In *Proceedings of the 12th Conference of the EACL*.

Klenner, M. 2007. Enforcing consistency on coreference sets. In *Recent Advances in Natural Language Processing (RANLP)*, pages 323–328.

Luo, X., A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proceedings of 42nd ACL*, page 135.

Luo, X. 2005. On coreference resolution performance metrics. *Proc. of HLT-EMNLP*, pages 25–32.

Luo, X. 2007. Coreference or not: A twin model for coreference resolution. In *Proceedings of NAACL HLT*, pages 73–80.

Màrquez, L., L. Padró, and H. Rodríguez. 2000. A machine learning approach for pos tagging. *Machine Learning Journal*, 39(1):59–91.

McCallum, A. and B. Wellner. 2005. Conditional models of identity uncertainty with application to noun coreference. *Advances in Neural Information Processing Systems*, 17:905–912.

McCarthy, J.F. and W.G. Lehnert. 1995. Using decision trees for coreference resolution. *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pages 1050–1055.

Ng, V. and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 104–111.

Nicolae, C. and G. Nicolae. 2006. Best Cut: A Graph Algorithm for Coreference Resolution. *Proceedings of the 2006 Conference on EMNLP*, pages 275–283.

Pelillo, M. 1997. The dynamics of nonlinear relaxation labeling processes. *Journal of Mathematical Imaging and Vision*, 7(4):309–323.

Quinlan, J.R. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Rosenfeld, R., R. A. Hummel, and S. W. Zucker. 1976. Scene labelling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6(6):420–433.

Soon, W.M., H.T. Ng, and D.C.Y. Lim. 2001. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4):521–544.

Surdeanu, M., J. Turmo, and E. Comelles. 2005. Named Entity Recognition from Spontaneous Open-Domain Speech. In *Ninth European Conference on Speech Communication and Technology*. ISCA.

Torras, C. 1989. Relaxation and neural learning: Points of convergence and divergence. *Journal of Parallel and Distributed Computing*, 6:217–244.

Vilain, M., J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. *Proceedings of the 6th conference on Message understanding*, pages 45–52.

Yang, X., G. Zhou, J. Su, and C.L. Tan. 2003. Coreference resolution using competition learning approach. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 176–183.