A Multi-resolution Framework for Information Extraction from Free Text

Mstislav Maslennikov and Tat-Seng Chua
Department of Computer Science
National University of Singapore
{maslenni,chuats}@comp.nus.edu.sg

Abstract

Extraction of relations between entities is an important part of Information Extraction on free text. Previous methods are mostly based on statistical correlation and dependency relations between entities. This paper re-examines the problem at the multi-resolution layers of phrase, clause and sentence using dependency and discourse relations. Our multi-resolution framework ARE (Anchor and Relation) uses clausal relations in 2 ways: 1) to filter noisy dependency paths; and 2) to increase reliability of dependency path extraction. The resulting system outperforms the previous approaches by 3%, 7%, 4% on MUC4, MUC6 and ACE RDC domains respectively.

1 Introduction

Information Extraction (IE) is the task of identifying information in texts and converting it into a predefined format. The possible types of information include entities, relations or events. In this paper, we follow the IE tasks as defined by the conferences MUC4, MUC6 and ACE RDC: slot-based extraction, template filling and relation extraction, respectively.

Previous approaches to IE relied on co-occurrence (Xiao et al., 2004) and dependency (Zhang et al., 2006) relations between entities. These relations enable us to make reliable extraction of correct entities/relations at the level of a single clause. However, Maslennikov et al. (2006) reported that the increase of relation path length will lead to considerable decrease in performance. In most cases, this decrease in performance occurs because entities may belong to different clauses. Since clauses in a sentence are connected by clausal relations (Halliday and Hasan, 1976), it is thus important to perform discourse analysis of a sentence.

Discourse analysis may contribute to IE in several ways. First, Taboada and Mann (2005) reported that discourse analysis helps to decompose long sentences into clauses. Therefore, it helps to distinguish relevant clauses from non-relevant ones. Second, Miltsakaki (2003) stated that entities in subordinate clauses are less salient. Third, the knowledge of textual structure helps to interpret the meaning of entities in a text (Grosz and Sidner 1986). As an example, consider the sentences *"ABC Co. appointed a new chairman. Additionally, the current CEO was retired"*. The word 'additionally' connects the event in the second sentence to the entity 'ABC Co.' in the first sentence. Fourth, Moens and De Busser (2002) reported that discourse segments tend to be in a fixed order for structured texts such as court decisions or news. Hence, analysis of discourse order may reduce the variability of possible relations between entities.

To model these factors, we propose a multi-resolution framework ARE that integrates both discourse and dependency relations at 2 levels. ARE aims to filter noisy dependency relations from training and support their evaluation with discourse relations between entities. Additionally, we encode semantic roles of entities in order to utilize semantic relations. Evaluations on MUC4, MUC6 and ACE RDC 2003 corpora demonstrates that our approach outperforms the state-of-art systems mainly due to modeling of discourse relations.

The contribution of this paper is in applying discourse relations to supplement dependency relations in a multi-resolution framework for IE. The

Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 592–599,
Prague, Czech Republic, June 2007. ©2007 Association for Computational Linguistics

framework enables us to connect entities in different clauses and thus improve the performance on long-distance dependency paths.

Section 2 describes related work, while Section 3 presents our proposed framework, including the extraction of anchor cues and various types of relations, integration of extracted relations, and complexity classification. Section 4 describes our experimental results, with the analysis of results in Section 5. Section 6 concludes the paper.

2 Related work

Recent work in IE focuses on relation-based, semantic parsing-based and discourse-based approaches. Several recent research efforts were based on modeling relations between entities. Culotta and Sorensen (2004) extracted relationships using dependency-based kernel trees in Support Vector Machines (SVM). They achieved an F_1-measure of 63% in relation detection. The authors reported that the primary source of mistakes comes from the heterogeneous nature of non-relation instances. One possible direction to tackle this problem is to carry out further relationship classification. Maslennikov et al. (2006) classified relation path between candidate entities into simple, average and hard cases. This classification is based on the length of connecting path in dependency parse tree. They reported that dependency relations are not reliable for the hard cases, which, in our opinion, need the extraction of discourse relations to supplement dependency relation paths.

Surdeanu et al. (2003) applied semantic parsing to capture the predicate-argument sentence structure. They suggested that semantic parsing is useful to capture verb arguments, which may be connected by long-distance dependency paths. However, current semantic parsers such as the ASSERT are not able to recognize support verb constructions such as *"X conducted an attack on Y"* under the verb frame "attack" (Pradhan et al. 2004). Hence, many useful predicate-argument structures will be missed. Moreover, semantic parsing belongs to the intra-clausal level of sentence analysis, which, as in the dependency case, will need the support of discourse analysis to bridge inter-clausal relations.

Webber et al. (2002) reported that discourse structure helps to extract anaphoric relations. However, their set of grammatical rules is heuristic. Our task needs construction of an automated approach to be portable across several domains. Cimiano et al. (2005) employed a discourse-based analysis for IE. However, their approach requires a predefined domain-dependent ontology in the format of extended logical description grammar as described by Cimiano and Reely (2003). Moreover, they used discourse relations between events, whereas in our approach, discourse relations connect entities.

3 Motivation for using discourse relations

Our method is based on Rhetorical Structure Theory (RST) by Taboada and Mann (2005). RST splits the texts into 2 parts: a) nuclei, the most important parts of texts; and b) satellites, the secondary parts. We can often remove satellites without losing the meaning of text. Both nuclei and satellites are connected with discourse relations in a hierarchical structure. In our work, we use 16 classes of discourse relations between clauses: *Attribution, Background, Cause, Comparison, Condition, Contrast, Elaboration, Enablement, Evaluation, Explanation, Joint, Manner-Means, Topic-Comment, Summary, Temporal, Topic-Change*. The additional 3 relations impose a tree structure: *textual-organization, span* and *same-unit*. All the discourse relation classes are potentially useful, since they encode some knowledge about textual structure. Therefore, we decide to include all of them in the learning process to learn patterns with best possible performance.

We consider two main rationales for utilizing discourse relations to IE. First, discourse relations help to narrow down the search space to the level of a single clause. For example, the sentence *"[<Soc-A1>**Trudeau**</>'s <Soc-A2>**son**</> told everyone], [their prime minister was his father], [who took him to a secret base in the arctic] [and let him peek through a window]."* contains 4 clauses and 7 anchor cues (key phrases) for the type Social, which leads to 21 possible variants. Splitting this sentence into clauses reduces the combinations to 4 possible variants. Additionally, this reduction eliminates the long and noisy dependency paths.

Second, discourse analysis enables us to connect entities in different clauses with clausal relations. As an example, we consider a sentence *"It's a dark comedy about a boy named <AT-A1>**Marshal**</> played by Amourie Kats who discovers all kinds of* *on and scary things going on in <AT-A2>**a seemingly quiet little town**</>”*. In this example, we need to extract the relation “At” between the entities “Marshal” and “a seemingly quiet little town”. The discourse structure of this sentence is given in Figure 1.

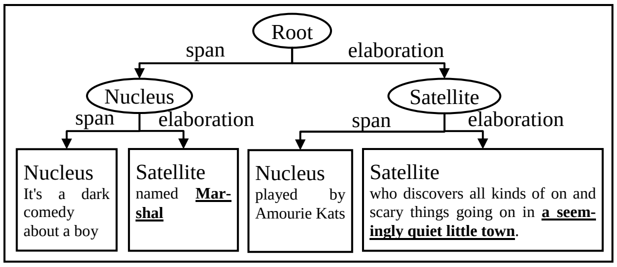

Figure 1. Example of discourse parsing

The discourse path *“Marshal <-elaboration- _ <-span- _ -elaboration-> _ -elaboration-> town”* is relatively short and captures the necessary relations. At the same time, prediction based on dependency path *“Marshal <–obj- _ <-i- _ <-fc- _ <-pnmod- _ <-pred- _ <-i- _ <-null- _ -null-> _ -rel-> _ -i-> _ -mod-> _ -pcomp-n-> town”* is unreliable, since the relation path is long. Thus, it is important to rely on discourse analysis in this example. In addition, we need to evaluate both the score and reliability of prediction by relation path of each type.

4 Anchors and Relations

In this section, we define the key components that we use in ARE: anchors, relation types and general architecture of our system. Some of these components are also presented in detail in our previous work (Maslennikov et al., 2006).

4.1 Anchors

The first task in IE is to identify candidate phrases (which we call anchor or anchor cue) of a predefined type (anchor type) to fill a desired slot in an IE template. The example anchor for the phrase “Marshal” is shown in Figure 2. Given a training set of sentences, we extract the anchor cues $A_{Cj} = [A_1, ..., A_{Nanch}]$ of type C_j using the procedures described in Maslennikov et al. (2006). The linguistic features of these anchors for the anchor types of Perpetrator, Action, Victim and Target for the MUC4 domain are given in Table 1.

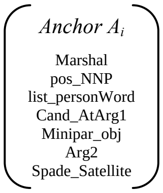

Figure 2. Example of anchor

Anchor types / **Feature**	**Perpetrator_Cue (A)**	**Action_Cue (D)**	**Victim_Cue (A)**	**Target_Cue (A)**
Lexical (Head noun)	terrorists, individuals, Soldiers	attacked, murder, Massacre	Mayor, general, priests	bridge, house, Ministry
Part-of-Speech	Noun	Verb	Noun	Noun
Named Entities	Soldiers (PERSON)	-	Jesuit priests (PERSON)	WTC (OBJECT)
Synonyms	Synset 130, 166	Synset 22	Synset 68	Synset 71
Concept Class	ID 2, 3	ID 9	ID 22, 43	ID 61, 48
Co-referenced entity	He -> terrorist, soldier	-	They -> peasants	-
Clausal type	Nucleus Satellite	Nucleus, Satellite	Nucleus, Satellite	Nucleus, Satellite
Argument type	Arg_0, Arg_1	Target, -, ArgM-MNR	Arg_0, Arg_1	Arg_1, ArgM-MNR

Table 1. Linguistic features for anchor extraction

Given an input phrase *P* from a test sentence, we need to classify if the phrase belongs to anchor cue type C_j. We calculate the entity score as:

$$Entity\_Score(P) = \sum \delta_i * Feature\_Score_i(P,C_j) \qquad (1)$$

where *Feature_Score(P,C_j)* is a score function for a particular linguistic feature representation of type C_j, and δ_i is the corresponding weight for that representation in the overall entity score. The weights are learned automatically using Expectation Maximization (Dempster et al., 1977). The *Feature_Score$_i$(P,C_j)* is estimated from the training set as the number of slots containing the correct feature representation type versus all the slots:

$$Feature\_Score_i(P,C_j) = \#(positive\ slots) / \#(all\ slots) \quad (2)$$

We classify the phrase *P* as belonging to an anchor type C_j when its *Entity_score(P)* is above an empirically determined threshold ω. We refer to this anchor as A_j. We allow a phrase to belong to multiple anchor types and hence the anchors alone are not enough for filling templates.

4.2 Relations

To resolve the correct filling of phrase *P* of type C_i in a desired slot in the template, we need to consider the relations between multiple candidate phrases of related slots. To do so, we consider several types of relations between anchors: discourse, dependency and semantic relations. These relations capture the interactions between anchors and are therefore useful for tackling the paraphrasing and alignment problems (Maslennikov et al., 2006). Given 2 anchors A_i and A_j of anchor types C_i and C_j, we consider a relation $Path_l = [A_i, Rel_1, ..., Rel_n, A_j]$ between them, such that there are no anchors between A_i and A_j. Additionally, we assume that the relations between anchors are represented in the form of a tree T_l, where $l = \{s, c, d\}$ refers to discourse, dependency and semantic relation types respectively. We describe the nodes and edges of T_l separately for each type, because their representations are different:

1) The nodes of discourse tree T_c consist of clauses $[Clause_1, ..., Clause_{Ncl}]$; and their relation edges are obtained from the Spade system described in Soricut and Marcu (2003). This system performs RST-based parsing at the sentence level. The reported accuracy of Spade is 49% on the RST-DT corpus. To obtain a clausal path, we map each anchor A_i to its clause in Spade. If anchors A_i and A_j belong to the same clause, we assign them the relation *same-clause*.
2) The nodes of dependency tree T_d consist of words in sentences; and their relation edges are obtained from Minipar by Lin (1997). Lin (1997) reported a parsing performance of Precision = 88.5% and Recall = 78.6% on the SUSANNE corpus.
3) The nodes of semantic tree T_s consist of arguments $[Arg_0, ..., Arg_{Narg}]$ and targets $[Target_1, ..., Target_{Ntarg}]$. Both arguments and targets are obtained from the ASSERT parser developed by Pradhan (2004). The reported performance of ASSERT is F_1=83.8% on the identification and classification task for all arguments, evaluated using PropBank and AQUAINT as the training and testing corpora, respectively. Since the relation edges have a form $Target_k$ -> Arg_l, the relation path in semantic frame contains only a single relation. Therefore, we encode semantic relations as part of the anchor features.

In later parts of this paper, we consider only discourse and dependency relation paths $Path_l$, where $l=\{c, d\}$.

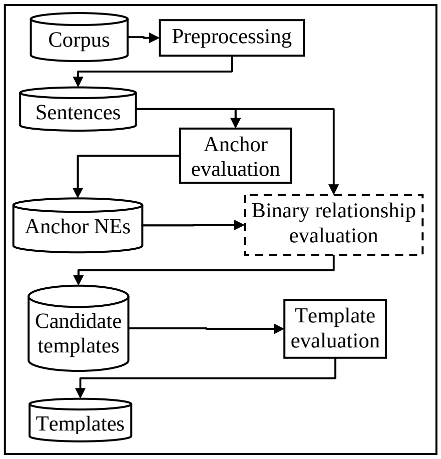

Figure 3. Architecture of the system

4.3 Architecture of ARE system

In order to perform IE, it is important to extract candidate entities (anchors) of appropriate anchor types, evaluate the relationships between them, further evaluate all possible candidate templates, and output the final template. For the case of relation extraction task, the final templates are the same as an extracted binary relation. The overall architecture of ARE is given in Figure 3.

The focus of this paper is in applying discourse relations for binary relationship evaluation.

5 Overall approach

In this section, we describe our relation-based approach to IE. We start with the evaluation of relation paths (single relation ranking, relation path ranking) to assess the suitability of their anchors as entities to template slots. Here we want to evaluate given a single relation or relation path, whether the two anchors are correct in filling the appropriate slots in a template. This is followed by the integration of relation paths and evaluation of templates.

5.1 Evaluation of relation path

In the first stage, we evaluate from training data the relevance of relation path $Path_l = [A_i, Rel_1, ..., Rel_n, A_j]$ between candidate anchors A_i and A_j of types C_i and C_j. We divide this task into 2 steps. The first step ranks each single relation $Rel_k \in Path_l$; while the second step combines the evaluations of Rel_k to rank the whole relation path $Path_l$.

Single relation ranking

Let Set_i and Set_j be the set of linguistic features of anchors A_i and A_j respectively. To evaluate Rel_k, we consider 2 characteristics: (1) the direction of relation Rel_k as encoded in the tree structure; and (2) the linguistic features, Set_i and Set_j, of anchors A_i and A_j. We need to construct multiple single relation classifiers, one for each anchor pair of types C_i and C_j, to evaluate the relevance of Rel_k with respect to these 2 anchor types.

(a) Construction of classifiers. The training data to each classifier consists of anchor pairs of types C_i and C_j extracted from the training corpus. We use these anchor pairs to construct each classifier in four stages. First, we compose the set of possible patterns in the form $P^+ = \{ P_m = <S_i -Rel-> S_j> \mid S_i \in Set_i , S_j \in Set_j \}$. The construction of P_m

conforms to the 2 characteristics given above. Figure 4 illustrates several discourse and dependency patterns of P^+ constructed from a sample sentence.

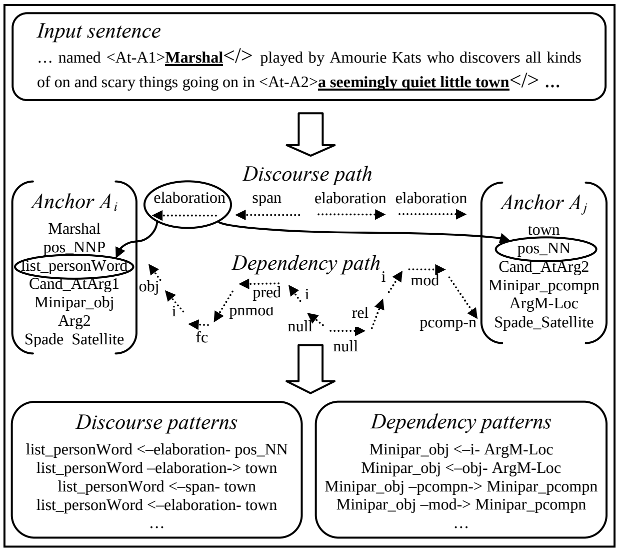
Input sentence
… named <At-A1>Marshal</> played by Amourie Kats who discovers all kinds of on and scary things going on in <At-A2>a seemingly quiet little town</> …
Discourse path
elaboration
span
elaboration
elaboration
Anchor A_i
Marshal
pos_NNP
list_personWord
Cand_AtArg1
Minipar_obj
Arg2
Spade Satellite
Anchor A_j
town
pos_NN
Cand_AtArg2
Minipar_pcompn
ArgM-Loc
Spade_Satellite
Dependency path
obj
i
fc
pnmod
pred
i
null
null
rel
i
mod
pcomp-n
Discourse patterns
list_personWord <–elaboration- pos_NN
list_personWord –elaboration-> town
list_personWord <–span- town
list_personWord <–elaboration- town
…
Dependency patterns
Minipar_obj <–i- ArgM-Loc
Minipar_obj <–obj- ArgM-Loc
Minipar_obj –pcompn-> Minipar_pcompn
Minipar_obj –mod-> Minipar_pcompn
…

Figure 4. Examples of discourse and dependency patterns

Second, we identify the candidate anchor *A*, whose type matches slot *C* in a template. Third, we find the correct patterns for the following 2 cases: 1) A_i, A_j are of correct anchor types; and 2) A_i is an action anchor, while A_j is a correct anchor. Any other patterns are considered as incorrect. We note that the discourse and dependency paths between anchors A_i and A_j are either correct or wrong simultaneously.

Fourth, we evaluate the relevance of each pattern $P_m \in P^+$. Given the training set, let $PairSet_m$ be the set of anchor pairs extracted by P_m; and $PairSet^+(C_i, C_j)$ be the set of correct anchor pairs of types C_i, C_j. We evaluate both precision and recall of P_m as

$$Precision(P_m) = \frac{\| PairSet_m \text{ I } PairsSet^+(C_i, C_j) \|}{\| PairSet_m \|} \quad (3)$$

$$Recall(P_m) = \frac{\| PairSet_m \text{ I } PairsSet^+(C_i, C_j) \|}{\| PairsSet^+(C_i, C_j) \|} \quad (4)$$

These values are stored and used in the training model for use during testing.

(b) Evaluation of relation. Here we want to evaluate whether relation *InputRel* belongs to a path between anchors $InputA_i$ and $InputA_j$. We employ the constructed classifier for the anchor types $InputC_i$ and $InputC_j$ in 2 stages. First, we find a subset $P^{(0)} = \{ P_m = <S_i \text{–}InputRel\text{->} S_j> \in P^+ \mid S_i \in InputSet_i,\ S_j \in InputSet_j \}$ of applicable patterns. Second, we utilize $P^{(0)}$ to find the pattern $P_m^{(0)}$ with maximal precision:

$$Precision(P_m^{(0)}) = argmax_{Pm \in P(0)}\ Precision\ (P_m) \quad (5)$$

A problem arises if $P_m^{(0)}$ is evaluated only on a small amount of training instances. For example, we noticed that patterns that cover 1 or 2 instances may lead to *Precision=1*, whereas on the testing corpus their accuracy becomes less than 50%. Therefore, it is important to additionally consider the recall parameter of $P_m^{(0)}$.

Relation path ranking

In this section, we want to evaluate relation path connecting template slots C_i and C_j. We do this independently for each relation of type discourse and dependency. Let $Recall_k$ and $Precision_k$ be the recall and precision values of Rel_k in $Path = [A_i, Rel_1, \ldots, Rel_n, A_j]$, both obtained from the previous step. First, we calculate the average recall of the involved relations:

$$W = (1/Length_{Path}) * \Sigma_{Relk \in Path}\ Recall_k \quad (6)$$

W gives the average recall of the involved relations and can be used as a measure of reliability of the relation *Path*. Next, we compute a combined score of average $Precision_k$ weighted by $Recall_k$:

$$Score = 1/(W*Length_{Path})*\Sigma_{Relk \in Path}\ Recall_k*Precision_k \quad (7)$$

We use all $Precision_k$ values in the path here, because omitting a single relation may turn a correct path into the wrong one, or vice versa. The combined score value is used as a ranking of the relation path. Experiments show that we need to give priority to scores with higher reliability *W*. Hence we use *(W, Score)* to evaluate each *Path*.

5.2 Integration of different relation path types

The purpose of this stage is to integrate the evaluations for different types of relation paths. The input to this stage consists of evaluated relation paths $Path_C$ and $Path_D$ for discourse and dependency relations respectively. Let $(W_l, Score_l)$ be an evaluation for $Path_l$, $l \in [c, d]$. We first define an integral path $Path_I$ between A_i and A_j as: 1) $Path_I$ is enabled if at least one of $Path_l$, $l \in [c, d]$, is enabled; and 2) $Path_I$ is correct if at least one of $Path_l$ is correct. To evaluate $Path_I$, we consider the average recall W_l of each $Path_l$, because W_l estimates the reliability of $Score_l$. We define a weighted average for $Path_I$ as:

$$W_I = W_C + W_D \quad (8)$$

$$Score_I = 1/W_I * \Sigma_l\ W_l * Score_l \quad (9)$$

Next, we want to determine the threshold score $Score_I^O$ above which $Score_I$ is acceptable. This score may be found by analyzing the integral paths on the training corpus. Let $S_I = \{ Path_I \}$ be the set of integral paths between anchors A_i and A_j on the training set. Among the paths in S_I, we need to define a set function $S_I(X) = \{ Path_I \mid Score_I(Path_I) \geq X \}$ and find the optimal threshold for X. We find the optimal threshold based on F_1-measure, because precision and recall are equally important in IE. Let $S_I(X)^+ \subset S_I(X)$ and $S(X)^+ \subset S(X)$ be sets of correct path extractions. Let $F_I(X)$ be F_1-measure of $S_I(X)$:

$$P_I(X) = \frac{\| S_I(X)^+ \|}{\| S_I(X) \|} \quad (10) \qquad R_I(X) = \frac{\| S_I(X)^+ \|}{\| S(X)^+ \|} \quad (11)$$

$$F_I(X) = \frac{2 * P_I(X) * R_I(X)}{P_I(X) + R_I(X)} \quad (12)$$

Based on the computed values $F_I(X)$ for each X on the training data, we determine the optimal threshold as $Score_I^O = argmax_X\ F_I\ (X)$, which corresponds to the maximal expected F_1-measure of anchor pair A_i and A_j.

5.3 Evaluation of templates

At this stage, we have a set of accepted integral relation paths between any anchor pair A_i and A_j. The next task is to merge appropriate set of anchors into candidate templates. Here we follow the methodology of Maslennikov et al. (2006). For each sentence, we compose a set of candidate templates T using the extracted relation paths between each A_i and A_j. To evaluate each template $T_i \in T$, we combine the integral scores from relation paths between its anchors A_i and A_j into the overall $Relation\_Score_T$:

$$Relation\_Score_T(T_i) = \frac{\sum_{1 \le i,j \le K} Score_I(A_i, A_j)}{M} \quad (13)$$

where K is the number of extracted slots, M is the number of extracted relation paths between anchors A_i and A_j, and $Score_I(A_i, A_j)$ is obtained from Equation (9).

Next, we calculate the extracted entity score based on the scores of all the anchors in T_i:

$$Entity\_Score_T(T_i) = \sum_{1 \le k \le K} Entity\_Score(A_k) / K \quad (14)$$

where $Entity\_Score(A_i)$ is taken from Equation (1).

Finally, we obtain the combined evaluation for a template:

$$Score_T(T_i) = (1 - \lambda) * Entity\_Score_T(T_i) + \lambda * Relation\_Score_T(T_i) \quad (15)$$

where λ is a predefined constant.

In order to decide whether the template T_i should be accepted or rejected, we need to determine a threshold $Score_T^O$ from the training data. If anchors of a candidate template match slots in a correct template, we consider the candidate template as correct. Let $TrainT = \{ T_i \}$ be the set of candidate templates extracted from the training data, $TrainT^+ \subset TrainT$ be the subset of correct candidate templates, and $TotalT^+$ be the total set of correct templates in the training data. Also, let $TrainT(X) = \{ T_i \mid Score_T(T_i) \geq X,\ T_i \in TrainT \}$ be the set of candidate templates with score above X and $TrainT^+(X) \subset TrainT(X)$ be the subset of correct candidate templates. We define the measures of precision, recall and F_1 as follows:

$$P_T(X) = \frac{\| TrainT^+(X) \|}{\| TrainT(X) \|} \quad (16) \qquad R_T(X) = \frac{\| TrainT^+(X) \|}{\| TotalT^+ \|} \quad (17)$$

$$F_T(X) = \frac{2 * P_T(X) R_T(X)}{P_T(X) + R_T(X)} \quad (18)$$

Since the performance in IE is measured in F_1-measure, an appropriate threshold to be used for the most prominent candidate templates is:

$$Score_T^O = argmax_X\ F_T\ (X) \quad (19)$$

The value $Score_T^O$ is used as a training model. During testing, we accept a candidate template $InputT_i$ if $Score_T(InputT_i) > Score_T^O$.

As an additional remark, we note that domains MUC4, MUC6 and ACE RDC 2003 are significantly different in the evaluation methodology for the candidate templates. While the performance of the MUC4 domain is measured for each slot individually; the MUC6 task measures the performance on the extracted templates; and the ACE RDC 2003 task evaluates performance on the matching relations. To overcome these differences, we construct candidate templates for all the domains and measure the required type of performance for each domain. Our candidate templates for the ACE RDC 2003 task consist of only 2 slots, which correspond to entities of the correct relations.

6 Experimental results

We carry out our experiments on 3 domains: MUC4 (Terrorism), MUC6 (Management Succession), and ACE-Relation Detection and Characterization (2003). The MUC4 corpus contains 1,300 documents as training set and 200 documents (TST3 and TST4) as official testing set. We used a modified version of the MUC6 corpus described by Soderland (1999). This version includes 599 documents as training set and 100 documents as testing set. Following the methodology of Zhang et al. (2006), we use only the English portion of ACE RDC 2003 training data. We used 97 documents for testing and the remaining 155 documents for training. Our task is to extract 5 major relation types and 24 subtypes.

Case (%)	P	R	F_1
GRID	52%	62%	57%
Riloff'05	46%	51%	48%
ARE (2006)	58%	61%	60%
ARE	65%	61%	63%

Table 2. Results on MUC4

To compare the results on the terrorism domain in MUC4, we choose the recent state-of-art systems GRID by Xiao et al. (2004), Riloff et al. (2005) and ARE (2006) by Maslennikov et al. (2006) which does not utilize discourse and semantic relations. The comparative results are given in Table 2. It shows that our enhanced ARE results in 3% improvement in F_1 measure over ARE (2006) that does not use clausal relations. The improvement was due to the use of discourse relations on long paths, such as "X *distributed leaflets claiming responsibility for* murder *of* Y". At the same time, for many instances, it would be useful to store the extracted anchors for another round of learning. For example, the extracted features of discourse pattern "murder –same_clause-> HUM_PERSON" may boost the score for patterns that correspond to relation path "X <-span- _ -Elaboration-> murder". In this way, high-precision patterns will support the refinement of patterns with average recall and low precision. This observation is similar to that described in Ciravegna's work on (LP)^2 (Ciravegna 2001).

Case (%)	P	R	F_1
Chieu et al.'02	75%	49%	59%
ARE (2006)	73%	58%	65%
ARE	73%	70%	72%

Table 3. Results on MUC6

Next, we present the performance of our system on MUC6 corpus (Management Succession) as shown in Table 3. The improvement of 7% in F_1 is mainly due to the filtering of irrelevant dependency relations. Additionally, we noticed that 22% of testing sentences contain 2 answer templates, and entities in many of such templates are intertwined. One example is the sentence *"Mr. Bronczek who is 39 years old succeeds Kenneth Newell 55 who was named to the new post of senior vice president"*, which refers to 2 positions. We therefore we need to extract 2 templates *"PersonIn: Bronczek, PersonOut: Newell"* and *"PersonIn: Newell, Post: senior vice president"*. The discourse analysis is useful to extract the second template, while rejecting another long-distance template *"PersonIn: Bronczek, PersonOut: Newell, Post: seniour vice president"*. Another remark is that it is important to assign 2 anchors of 'Cand_PersonIn' and 'Cand_PersonOut' for the phrase "Kenneth Newell".

The characteristic of the ACE corpus is that it contains a large amount of variations, while only 2% of possible dependency paths are correct. Since many of the relations occur only at the level of single clause (for example, most instances of relation *At*), the discourse analysis is used to eliminate long-distance dependency paths. It allows us to significantly decrease the dimensionality of the problem. We noticed that 38% of relation paths in ACE contain a single relation, 28% contain 2 relations and 34% contain ≥ 3 relations. For the case of ≥ 3 relations, the analysis of dependency paths alone is not sufficient to eliminate the unreliable paths. Our results for general types and specific subtypes are presented in Tables 6 and 7, respectively.

Case (%)	P	R	F_1
Zhang et al.'06	77%	65%	70%
ARE	79%	66%	73%

Table 4. Results on ACE RDC'03, general types

Based on our results in Table 4, discourse and dependency relations support each other in different situations. We also notice that multiple instances require modeling of entities in the path. Thus, in our future work we need to enrich the search space for relation patterns. This observation corresponds to that reported in Zhang et al. (2006).

Discourse parsing is very important to reduce the amount of variations for specific types on ACE

RDC'03, as there are 48 possible anchor types.

Case (%)	P	R	F_1
Zhang et al.'06	64%	51%	57%
ARE	67%	54%	61%

Table 5. Results on ACE RDC'03, specific types

The relatively small improvement of results in Table 5 may be attributed to the following reasons: 1) it is important to model the commonality relations, as was done by Zhou et al. (2006); and 2) our relation paths do not encode entities. This is different from Zhang et al. (2006), who were using entities in their subtrees.

Overall, the results indicate that the use of discourse relations leads to improvement over the state-of-art systems.

7 Conclusion

We presented a framework that permits the integration of discourse relations with dependency relations. Different from previous works, we tried to use the information about sentence structure based on discourse analysis. Consequently, our system improves the performance in comparison with the state-of-art IE systems. Another advantage of our approach is in using domain-independent parsers and features. Therefore, ARE may be easily portable into new domains.

Currently, we explored only 2 types of relation paths: dependency and discourse. For future research, we plan to integrate more relations in our multi-resolution framework.

References

P. Cimiano and U. Reyle. 2003. Ontology-based semantic construction, underspecification and disambiguation. *In Proc of the Prospects and Advances in the Syntax-Semantic Interface Workshop*.

P. Cimiano, U. Reyle and J. Saric. 2005. Ontology-driven discourse analysis for information extraction. *Data & Knowledge Engineering*, 55(1):59-83.

H.L. Chieu and H.T. Ng. 2002. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. *In Proc of AAAI-2002*.

F. Ciravegna. 2001. Adaptive Information Extraction from Text by Rule Induction and Generalization. *In Proc of IJCAI-2001*.

A. Culotta and J. Sorensen J. 2004. Dependency tree kernels for relation extraction. *In Proc of ACL-2004*.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38.

B. Grosz and C. Sidner. 1986. Attention, Intentions and the Structure of Discourse. *Computational Linguistics*, 12(3):175-204.

M. Halliday and R. Hasan. 1976. *Cohesion in English*. Longman, London.

D. Lin. 1997. Dependency-based Evaluation of Minipar. *In Workshop on the Evaluation of Parsing systems*.

M. Maslennikov, H.K. Goh and T.S. Chua. 2006. ARE: Instance Splitting Strategies for Dependency Relation-based Information Extraction. *In Proc of ACL-2006*.

E. Miltsakaki. 2003. *The Syntax-Discourse Interface: Effects of the Main-Subordinate Distinction on Attention Structure*. PhD thesis.

M.F. Moens and R. De Busser. 2002. First steps in building a model for the retrieval of court decisions. International *Journal of Human-Computer Studies*, 57(5):429-446.

S. Pradhan, W. Ward, K. Hacioglu, J. Martin and D. Jurafsky. 2004. Shallow Semantic Parsing using Support Vector Machines. *In Proc of HLT/NAACL-2004*.

E. Riloff, J. Wiebe, and W. Phillips. 2005. Exploiting Subjectivity Classification to Improve Information Extraction. *In Proc of AAAI-2005*.

S. Soderland. 1999. Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning*, 34:233-272.

R. Soricut and D. Marcu. 2003. Sentence Level Discourse Parsing using Syntactic and Lexical Information. *In Proc of HLT/NAACL*.

M. Surdeanu, S. Harabagiu, J. Williams, P. Aarseth. 2003. Using Predicate Arguments Structures for Information Extraction. *In Proc of ACL-2003*.

M. Taboada and W. Mann. 2005. Applications of Rhetorical Structure Theory. *Discourse studies*, 8(4).

B. Webber, M. Stone, A. Joshi and A. Knott. 2002. Anaphora and Discourse Structure. *Computational Linguistics*, 29(4).

J. Xiao, T.S. Chua and H. Cui. 2004. Cascading Use of Soft and Hard Matching Pattern Rules for Weakly Supervised Information Extraction. *In Proc of COLING-2004*.

M. Zhang, J. Zhang, J. Su and G. Zhou. 2006. A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features. *In Proc of ACL-2006*.

G. Zhou, J. Su and M. Zhang. 2006. Modeling Commonality among Related Classes in Relation Extraction. *In Proc of ACL-2006*.