TETI: a TimeML Compliant TimEx Tagger for Italian

Tommaso Caselli, Felice dell'Orletta and Irina Prodanof
ILC-CNR
Pisa, Italy
Email: firstName.secondName@ilc.cnr.it

***Abstract*—In this paper we will present an ongoing research on the development of a temporal expression tagger and normalizer for Italian, compliant with the TimeML specifications. Similarly to other existing temporal expression taggers, the system is rule-based and benefits from an extensive corpus study to identify the reserved time words. However, it differs from other systems since it implements WordNet-based semantic relations between temporal expressions in order to improve its accuracy. So far, the system reports an F-measure of 86.41% for the subtask of temporal expression detection and bracketing.**

I. Introduction

Recently, a renewed interest in temporal processing of real texts has spread in the NLP community, boosted by the presence of specific mark-up languages such as TimeML [1]. Moreover, a growing number of initiatives (CLEF[1], TERN[2], TempEval[3]) has been developed focussing on automatic extraction of temporal information from texts. This task is both challenging and extremely useful since it may improve semantic access to information from which many NLP applications, such as Information Extraction, (IE), and Question-Answering (both Domain-Specific and Open-Domain) can improve their performance.

A temporal expression, (timex), is a phrase which primarily denotes a temporal entity, i.e. an interval or an instant[4]. This definition of timex is a standard in the NLP community [3] and restricts the notion of timex to purely temporal items, preventing an overextension of this concept to semantically different words such as *presidency*, *pregnancy*, *school* etc., which have a contextually dependent temporal reading. Timexes do not convey only strict temporal information but they instantiate also temporal links, or anchors, both for locating an eventuality in time (event time stamping) and to put eventualities in order one with respect to each other (event temporal ordering). Thus, timexes have a primary role to perform temporal reasoning since they offer explicit temporal information.

The task of automatically extracting timexes can be divided into four subtasks:

- recognizing and bracketing the portion of text which denotes the timex;
- extracting the features (type of time unit, referential status, and presence of modifiers);
- computing the interval of reference on the time line;
- resolving the timex, i.e. normalize the value to a standard output format.

Our tagger so far is set to deal with the first two subtasks, i.e. recognition and bracketing, and feature extraction. With respect to most existing systems for timex recognition (and normalization) based on TIDES ([3], [4]), the system is based on the TimeML specifications [1]. The remaining of the paper is structured as follows: in Section II the markup of timexes in TimeML will be briefly described. Section III will illustrate the methodology followed in order to implement the system and the general architecture. Finally, in Section IV we report the results achieved by the system, summarising strenghts and limitations and some suggestions for its improvements. Conclusions and future works will be presented in Section V.

II. Timexes in TimeML: the TIMEX3 tag

The specifics of the TimeML tagset for annotating timexes do not simply extend or modify previous tags for this annotation task, namely the `TIMEX` tag in STAG [5] and the `TIMEX2` tag in TIDES, but present interesting differences which have been introduced in order to improve the representational and informational strength of the tag. Provided the fact that our tagger is not able at the moment to perform all the four subtasks, we will illustrate only those parts of the `TIMEX3` tag specifications which are relevant for the comprehension of the functioning of the tagger .

The `<TIMEX3>` tag is used to mark up any timex referring to:

(a.) Day times; e.g.: "*mezzogiorno*" [noon], *3:00*, "*la mattina*" [the morning], ...;

(b.) Dates of different granularity such as days (e.g. "*ieri*" [yesterday], "*8 Gennaio 1980*" [Jan, 8 1980], "*venerdì scorso*" [last Friday]), weeks (e.g. "*la prossima settimana*" [next week]), months ("*tra due mesi*" [in two month], "*il mese prossimo*" [next month]...), seasons or business quarters (e.g. "*la scorsa primavera*" [last spring], "*lo scorso semestre*" [last semester]...), years (e.g. "*1980*", "*l'anno scorso*" [last year]...), decades, centuries, millenium;

(c.) Durations; e.g.: "*due mesi*" [two month], "*un periodo di cinque ore*" [a 5-hour period], ...;

[1]http://www.clef-campaign.org/
[2]http://www.nist.gov/speech/tests/ace/2007/index.html
[3]http://www.timeml.org/tempeval/
[4]Following [2] we consider interval and instants as both temporal primitives.

(d.) Sets; e.g.: "*una volta al mese*" [one a month], "*ogni martedì*" [every Tuesday], ...

The surface-oriented approach to the tagging of expressions in TimeML implies that the annotation of timexes is based (i.) on the constituent structure and (ii.) on the granularity of the time units and, finally, (iii.) on their relations.

The span of the tag must correspond to one of the following categories:

- Noun Phrase
- Adjectival Phrase
- Adverbial Phrase
- Time/Date Patterns

A standard TIMEX3 tag will look like as in the following examples:

1.) *il pomeriggio.*
the afternoon.
`<TIMEX3>il pomeriggio</TIMEX3>`

2.) *01/12/80*
`<TIMEX3>01/12/80</TIMEX3>`

When a timex is introduced by a preposition or by subordinating conjunction, these parts-of-speech are not to be included into the TIMEX3 tag. Relevant temporal prepositions and other signals are marked with a tag of their own, namely the SIGNAL tag, thus:

3.) *nel pomeriggio.*
in the afternoon.
```
<SIGNAL>nel</SIGNAL>
<TIMEX3>pomeriggio</TIMEX3>
```

4.) *per l'autunno.*
by the fall.
```
<SIGNAL>per</SIGNAL>
<TIMEX3>l'autunno</TIMEX3>
```

The only exceptions are represented by the prepositions "*circa*", "*intorno a*" and "*verso*" which must be included into the extent of the tag because they have a role in the normalization of the timex.

When timexes are realized by multiword expressions, like "*per ora*" [for the moment], "*dopo domani*" [the day after tomorrow], "*fin ora*" [up to now], and similar the whole expression is considered as a single time unit ([6]). Consequently, all elements forming the timexes must be included into the tag, as illustrated in the following examples:

5.) *per ora.*
nowadays.
`<TIMEX3>per ora<TIMEX3>`

6.) *dopo domani.*
the day after tomorrow.
`<TIMEX3>dopo domani<TIMEX3>`

Modifiers must be included into the tag with the exception of postmodifiers denoting an eventuality. Appositive constructions are considered as post-modifiers, and thus they are included into the tag span. Nevertheless, if the appositive clause contains a lexical trigger for timexes, two distinct TIMEX3 tags must be created.

When two consecutive timexes are encountered, different rules apply for the tag span according to the type of relation which exists between the two timexes. In these cases:

- The timexes will be marked up in a single tag if:
  (i.) the two expressions belong to the same temporal unit or if they are related by a meronomical relation of *part_of*, or if they correspond to a clock time:
  7.) *venerdì sera.*
  Friday night.
  `<TIMEX3>venerdì sera</TIMEX3>`
  8.) *venerdì ore 11.*
  Friday, 11:00.
  `<TIMEX3>venerdì ore 11</TIMEX3>`
  9.) *martedì 26 giugno.*
  Tuesday, June 26.
  `<TIMEX3>martedì 26 giugno</TIMEX3>`
  10.) *alle 13 e 56.*
  at 13:56.
  ```
  <SIGNAL>alle</SIGNAL>
  <TIMEX3>13 e 56</TIMEX3>
  ```
  (ii.) the second timex is introduced by the prepositions *di* or *del* and it represents a definite time specification:
  11.) *la mattina del 20 giugno.*
  the morning if June, 20.
  ```
  <TIMEX3>
  la mattina del 20 giugno
  </TIMEX3>
  ```
  12.) *ottobre del 1963.*
  October 1963.
  `<TIMEX3>ottobre del 1963</TIMEX3>`
  13.) *alle 11 di ieri mattina.*
  at 11 yesterday morning.
  ```
  <SIGNAL>alle</SIGNAL>
  <TIMEX3>11 di ieri mattina</TIMEX3>
  ```
- Two tags must be created:
  (i.) when two timexes are in an anchoring relation with each other:
  14.) *due settimane da oggi.*
  two weeks from today.
  ```
  <TIMEX3>due settimane</TIMEX3>
  <SIGANL>da<SIGNAL>
  <TIMEX3>oggi</TIMEX3>
  ```
  15.) tre giorni prima di ieri.
  three days before yesterday.
  ```
  <TIMEX3>tre giorni</TIMEX3>
  <SIGNAL>prima di<SIGNAL>
  <TIMEX3>ieri</TIMEX3>
  ```
  (ii.) when the timexes are separated by an intervening element, like temporal prepositions (with the exception of *di*) or conjunctions:

16.) *venerdì sera alle 20.00.*
Friday night at 20:00.

```
<TIMEX3>venerdì sera</TIMEX3>
<SIGNAL>alle<SIGNAL>
<TIMEX3>20.00</TIMEX3>
```

Among non markable time expressions, together with those expressions which can have a temporal meaning but are not considered trigger words, are also included (non markable elements are in bold):

- Frequency expressions, when no time period is given;

  17.) *L' Italia è campione del mondo per **la quarta volta**.*
  Italy has become World Champion for the 4^{th} time.

- Sequencing and ordering expressions:

  18.) *Le indagini erano state **inizialmente** approvate dal presidente.*
  The investigations were initially approved by the president

- Manner adverbs:

  19.) ***Subito** soccorsa dai medici presenti nel villaggio.*
  Immediately helped by the doctors who were in the village.

- Non-quantifiable durations:

  20.) ***a breve termine**.*
  in the short term.

- Proper names that contain or comprise a time expression but denote named entities or similar.

III. System architecture and methodology

Previous systems for timex recognition and normalization ([7], [8], [9], [10] among others) have been implemented in large part by means of finite state transducers or rule-based taggers. These approaches are the most useful for this task. In fact, the relatively limited set of words which give rise to a timex suggests that rule-based systems can be implemented with a small effort and provide good results.

The general architecture of the tagger is illustrated in Figure 1. The tagging program takes in input a document which has been analyszed by a shallow parser [11][5]. The tagger relies on two main components: an identifier of timexes (TIMEX DETECTOR) and a grammar (TIMEX TAGGER). The grammar, which combines a general condition for activation and a set of local rules, is responsible both for the bracketing and for the identification of the features of the timexes. Both components are linked to two external resources: a dictionary of timex trigger words augmented with semantic relations (TimEx Trigger Dictionary) and a dictionary of modifiers (Modifier Dictionary).

In comparison with other rule-based systems, like [8], [9], which use as input the simple morphological analysis level, i.e. parts-of-speech, the choice of using as input chunked texts is strictly related to the fact that the textual extent of the `TIMEX3` tag corresponds to a limited and well defined set of phrases

[5]The chunker' s performance in terms of precision and recall is P = 90.65 and R = 91.62.

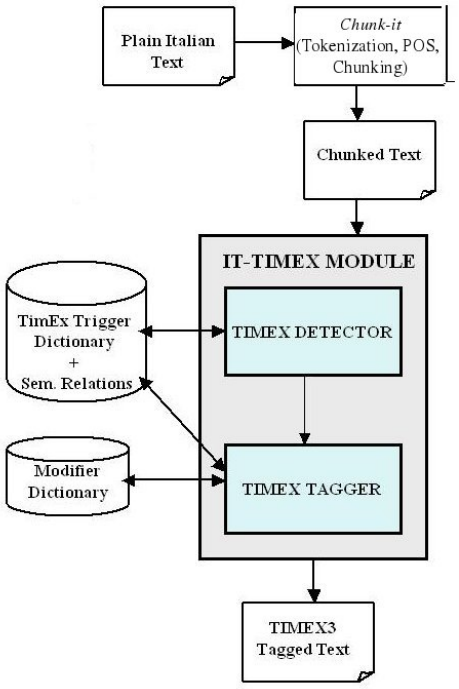

Figure 1. The architecture of the system.

and the output of a shallow parser represents an approximation of the target phrases, thus facilitating the writing of the rules.

In order to write the rules and adapt the `TIMEX3` specifications to Italian we have performed a corpus exploration. The corpus used is composed by 179 Italian newpaper articles and a total of 62 thousands words. The corpus has been analyzed by the shallow parser and then we have automatically extracted those constituents which could contain a timex as their head, including also prepositional chunks, for a total of five different chunk types (nominal, adverbial, adjectival, prepositional and *di*_chunks). The extracted chunks have been connected to a semantic lexical resource, SIMPLE/CLIPS ([12]), and augmented with ontological information from the resource, by associating the head noun of each chunk to its ontological type. By means of a simple query, all instances of timexes have been extracted by restricting the noun heads to the type "TIME", which, in SIMPLE/CLIPS, is defined as all nouns referring to timexes. This subsets of chunks has been manually checked to exclude instances of false positives. As a result we have identified a total of 1485 chunks of potential timexes. A first interesting result is the distribution of the constituents: more than 60% (968 out of 1485) is realized by prepositional chunks (including the class of *di*_chunk), followed by the class of nominal (254) and adverbial chunks (195), and finally by the class of adjectival chunks (68). The data show that the vast majority of timexes is introduced by a temporal preposition. This has an important consequence for the development of our rules, since, in order to be compliant with the TimeML specifics, the tagger must distinguish the real timexes from the class of signals which are not to be

considered as part of the TIMEX3 tag.

Analyzing the various chunks, we have identified four main patterns which may correspond to a TIMEX3 tag, namely:

- Pattern 1: a single chunk; e.g.: "*ieri*$_{ADV\_C}$" [yesterday], "*lunedì*$_{N\_C}$" [Monday], "*di domenica*$_{DI\_C}$" [on Sunday], "*il semestre*$_{N\_C}$" [the semester], "*nello stesso periodo*$_{P\_C}$" [at the same period], ...
- Pattern 2: a combination of two consecutive chunks; e.g.: "*sabato*$_{N\_C}$ *notte*$_{N\_C}$" [Saturday night], "*il mese*$_{N\_C}$ *scorso*$_{ADJ\_C}$" [last month], "*la mattina*$_{N\_C}$ *di venerdì*$_{DI\_C}$" [Friday morning], ...
- Pattern 3: a combination of three consecutive chunks; e.g.: "*gli ultimi*$_{ADJ\_C}$ *tre mesi*$_{N\_C}$ *dell'anno*$_{DI\_C}$" [the last three months of the year], ...
- Pattern 4: a combination of four consecutive chunks; e.g.: "*il primo semestre*$_{N\_C}$ *fiscale*$_{ADJ\_C}$ *dell' anno*$_{DI\_C}$ *scorso*$_{ADJ\_C}$" [the first fiscal semester of the last year], ...

In principle, it is also possible to have two further patterns which may be obtained by the composition of five or six consecutive chunks and which correspond to timexes of the kind "*gli ultimi*$_{ADJ\_C}$ *tre mesi*$_{N\_C}$ *del primo semestre*$_{DI\_C}$ *fiscale*$_{ADJ\_C}$ *dell' anno*$_{DI\_C}$" [the last three months of the first fiscal semester of the year] and "*gli ultimi*$_{ADJ\_C}$ *tre mesi*$_{N\_C}$ *del primo semestre*$_{DI\_C}$ *fiscale*$_{ADJ\_C}$ *dell' anno*$_{DI\_C}$ *scorso*$_{ADJ\_C}$" [the first three months of the first fiscal semester of the last year]. Such complex chunk patterns have not been identified in the corpus but though their existence cannot be excluded *a priori*, they have been considered as possible patterns as well.

Our development efforts concentrated on writing rules based on these patterns. One of the main advantages of working with chunks and their possible combinations is the reduced number of rules for tagging timexes since the longer the pattern, the fewer the types of chunks involved.

A consequence of the corpus analysis is the creation of the two external resources: the timex trigger dictionary, TimEx Trigger Dictionary, and the modifier dictionary, Modifier Dictionary. In the next section we will illustrate their structure and the information they make available to the tagger.

A. The external lexical resources

The two dictionaries represent two key elements for the correct functioning of the tagger. Both dictionaries have been created in a semi-automatic way and then manually postprocessed to check for wrong or missing information.

The TimEx Trigger Dictionary is composed by 157 lexical entries corresponding to a comprehensive list of words denoting timexes, including proper names of national holidays and festivities. The dictionary is not simply a list of lemmas but it represents a repository of information on timexes. Every entry in the dictionary has the following structure:

- the lemma and its associated part-of-speech;
- the absolute reference type of the trigger word, in particular if the trigger word is an absolute timex or a relative one;
- the default type according to the TimeML specifications, i.e. whether the timex corresponds to a calendar date (DATE), a clock time or a part of the day (TIME), a duration (DURATION), or a set of times (SET);
- the basic time format and value according to the ISO 8601 standard which is associated to the trigger word on the basis of its semantics; for instance a trigger word like "*domani*" [tomorrow] has an associated time format corresponding to that of a calendar date, i.e. YYYY-MM-DD where the values for the year, month and day remain underspecified since it is not possible to associate a specific value to any of them in a principled way. Notice the difference with a trigger word like "*Natale*" [Christmas] which has the same time format as "*domani*" [tomorrow] but can be associated with a more specific value where only the year is underspecified e.g. YYYY-12-25;
- a description of the semantic of the trigger word, when possible, by means of a metalanguage;
- the associated granularity level of the time unit expressed by the trigger word, i.e. whether the time word denotes a hour (**TH**), a day (**D**), a part or time of the day (**TOD**), a day of the week (**DOW**), a year (**Y**), a decade (**DE**), a century (**C**) and so on an so forth
- ontological information, expressed by the "*is_a*" relation whose primitives are represented by the value *interval* and *instant*;
- a set of 8 semantic relations automatically obtained from the combination of ItalWordNet and SIMPLE/CLIPS, comprehending both classical lexical semantics relations like synonymy, merony, holonymy, hyponymy and hyperonymy, temporal relations, like *after* and *before*, and a general semantic relation, i.e. *fuzzynym*. This set of relations connects the lemmas of temporal trigger words with each other and extends the relations to all the other features which are associated with that lemma, thus forming a rich semantic network which offers important information both for the bracketing task and the normalization process. For instance, knowing that a trigger word like "*sera*" [evening] stands in a *part_of* relation with "*venerdì*" [Friday] implies that their time units stand in the same relation as well thus avoiding that the two trigger words are assigned to two different TIMEX3 tag labels. Moreover, this set of relations facilitates the creation of normalization rules.

An instance of how a dictionary entry looks like is reported in example 21. For clarity's sake, when presenting the semantic relations we have not listed all the lemmas with which the entry is connected to but only a reduced sample:

21.) lemma = LUNEDI' [
part of speech: NN
absolute reference: relative
default type: DATE
basic time format and value: YYYY-MM-DD
semantic description: WEEK $\subset$ (DAY_1)

time unit granularity level: DOW
is_a: INTERVAL
is_a_part_of: settimana, mese, anno, ...; *has_hyperonym*: giorno; *has_fuzzynym*: oggi, domani, vigilia, ieri, Natale, ...; *has_as_part*: mattina, pomeriggio, alba, ora, ...; *before*: martedì; *after*: domenica
]

The modifiers' dictionary is composed by 63 lexical entries. It contains two classes of modifiers: those whose semantics is essential in order to assign an absolute value to relative timexes, like "*scorso*" [last], *passato* [past/last], "*in corso*" [current] and similar, and those modifiers which code a vague quantification over the timexes, like "*circa*" [about], "*non più di*" [no more than], "*non meno di*" [less than], "*verso*" [towards] and similar. This second type of modifiers has a special role in determining the meaning of timexes, since it introduces fuzziness in the intended values, in particular, with respect to when the denoted time period starts and ends. For instance, a timex like "*i primi anni Sessanta*" [the early Sixties] is rather vague with respect to what part of the decade is referring to. It could be a period ranging from 1961 to 1965, or even a smaller one, from 1961 to 1963. Due to the fact that an absolute calendar date or duration cannot be reliably assigned to these expressions, it has been decided, since the development of `TIMEX2` tag, to express this intrinsic vagueness by means of a dedicated attribute, `mod`, which is implemented in the `TIMEX3` tag as well . In addition to this, due to the very limited set of vague modifiers, they have been associated with standard values. Modifiers are certainly more important in the normalization phase rather then in the bracketing one, but observing the various patterns (**Pattern 1 - 4**) we can notice that the chunker output does not always clusters all modifiers into the same chunk. Consequently, even in the bracketing phase being aware of the fact that the head of a chunk represents a modifier of a timex is necessary in order to be compliant to the TimeML specifications.

The entries in the modifier dictionary have a common structure but differ from each other according to their type as we have described above. Thus, every modifiers has:

- lemma and part of speech;
- information on its position, i.e. if it is only a premodifier or a postmodifier, or both;

Then the entries differentiate for other information. Modifiers which are essential to the identification of the absolute value of the temporal trigger word have information necessary for the normalization process in terms of general rules and, when possible, the associated temporal relation with respect to the anchor. An instance of an entry of modifiers of this kind is represented in example 22.

22.) lemma = SCORSO [
position: PREMODIFIER — POSTMODIFIER
normalization value: CurrentTimEx_granularity - 1_anchorValue_granularity
temporal relations: BEFORE_anchor
]

The meaning of the normalization value is an approximation of the semantics of the modifier. In this case, it means that the presence of "*scorso*" requires that the value of the associated timex can be obtained by subtracting 1 to the anchoring timex at the time unit granularity level. For instance, if we have a timex like "*lo scorso anno*" [last year], from the timex trigger dictionary we know that the time unit of this timex has granularity **Y**ear and a time format of type `YYYY`. Once the anchor is identified, which is usually an absolute timex, the value of the timex is obtain by subtracting 1 to the anchor value at the same time unit granularity level, which in this case is **Y**.

Vague modifiers have the associated standard value which has to be added to the dedicated attribute in the `TIMEX3` tag. In example 23 we illustrate the structure of an entry of vague modifiers:

23.) lemma = META' [
position: PREMODIFIER
mod attribute value: MID
]

As for numbers, only ordinal numbers have been introduced in the dictionary. We have restricted the set of ordinal numbers to the first four, since these are the most used in timexes. These types of modifiers are twofolded: on the one hand, they can contribute to the identification of the absolute value of a relative expression, like in "*il primo giorno dell'anno*" [the first day of the year], and, on the other hand, they indicate a particular period of time, as in "*il primo trimestre*" [the first quarter]. In addition, when they occur in the plural, they assume the status of vague modifiers, as in "*i primi mesi dell'anno*" [the first months of the year], and are associated with a default value of the `mod` attribute. Introducing these modifiers as two distinct entries is not the best way to deal with them, since it will complicate the dictionary lookup mechanisms creating unnecessary complexities for the system. Thus we have opted for a different solution, that is to remain silent in the dictionary about this double values of this set of modifiers and shift the issue of values' assignment to the normalization phase. Thus, ordinal numbers are present in the dictionary but the set of information associated with them is restricted to the lemma, part-of-speech and position.

B. Detecting and tagging timexes and signals

The recognition of the chunk patterns corresponding to `TIMEX3` tags is performed by means of local rules which work, as previously stated, on a limited number of chunks, that is those which have as their head a timex trigger word. The TIMEX DETECTOR component analyzes the chunked text which it receives as input and identifies both the timex trigger words and the modifiers by means of a lookup in the dedicated dictionaries. When a positive match is found, it marks the chunk head with this information. In case the chunk head is a temporal trigger word the detector extracts all the additional information necessary for the bracketing phase, such as the granularity value of the time unit and the default type of the trigger word. This information represents the input for

the second component, the TIMEX TAGGER which applies the recursive rules.

The TIMEX TAGGER has two types of conditions which must be satisfied in order to activate the grammar rules responsible for the bracketing and the creation of the corresponding `TIMEX3` tag. The first is a general condition which states that the head of the chunk in analysis must correspond to a temporal trigger word. The second are local conditions on the type of chunk which contains the trigger word, and the head of the immediately previous and immediately following chunks. The local rules are activated only if the general condition has a positive match. If the local conditions are true, the tagger activates the corresponding grammar rules for the bracketing, otherwise it looks for other local conditions and related grammar rules. The final output of the tagger is a a chunked text with an additional layer of annotation represented by the `TIMEX3` tag. In case no positive match for the local conditions can be identified, the tagger will fail and no `TIMEX3` tag is assigned. However, even in case of failure, a partial output is always provided and is represented by the output of the detector component. In this way, it is possible to check which local conditions or rules are missing and integrate them into the tagger.

To illustrate in details the functioning of the tagger we will present the rule for recognizing timexes belonging to **Pattern 1**, i.e. timexes corresponding to a single chunk, like "*sabato*" [Saturday], "*ieri*" [yesterday] and similar. The rule is illustrated in example 24.

24.)
```
COND
  (POTGOV_lemma equals timexTrigger)
  (and
    (or (POTGOV_CHUNK equals N_C)
        (POTGOV_CHUNK equals ADV_C)
        (POTGOV_CHUNK equals ADJ_C))
    (not (POTGOV_CHUNK has PREMODIF)
    (not (POTGOV_lemma CHUNK-1
           equals modifTrigger))
    (or (not (POTGOV_lemma CHUNK+1
                 equals timexTrigger))
        (not ( POTGOV_lemma CHUNK+1
                equals modifTrigger)))
     )
     then
      CREATE TIMEX3_tag
      (and (BEGIN_AT B_CHUNK)
             (END_AT E_CHUNK) )
```

As already explained, the tagger takes as input a chunked text, as illustrated in Figure 2, and then activates the detector component which identifies the temporal trigger words and modifiers.

In this small excerpt the only trigger word is "*sabato*" [Saturday]. The tagger is activated and first checks if the general condition is true, i.e. if the head of the chunk is a temporal trigger words. This corresponds to the second line

```
38 B- P_C  PREP:   nel in PREP_A MS!!!!!
39 I- P_C  POTGOV: villaggio villaggio NN MS!!!!P
40 B- PUNCT_C      POTGOV: " " PUNCT !!!!!!!
41 B- N_C  POTGOV: kartibubbo kartibubbo NN_P !!!!!!!
42 B- PUNCT_C      POTGOV: " " PUNCT !!!!!!!
43 B- DI_C PREP:   di di PREP !!!!!!!
44 I- DI_C POTGOV: campobello campobello NN_P !!!!!!!
45 B- DI_C PREP:   di di PREP !!!!!!!
46 I- DI_C POTGOV: mazara mazara NN_P !!!!!!!
47 B- P_C  PREP:   nel in PREP_A MS!!!!!
48 I- P_C  POTGOV: trapanese trapanese NN _S!!!!P
49 B- PUNCT_C      POTGOV: , , PUNCT !!!!!!!
50 B- FV_C AUX:    è essere V_FIN !S3PI!!
51 I- FV_C POTGOV: morta morire V_PP FS!RP!!
52 B- N_C  POTGOV: sabato sabato NN MS!!!!P
53 B- P_C  PREP:   per per PREP !!!!!!!
54 I- P_C  DET:    una una ART_I FS!!!!!
55 I- P_C  POTGOV: scarica scarica NN FS!!!!P
56 B- ADJ_C        POTGOV: elettrica elettrico ADJ FS!!!!P
57 B- SUBORD_C     POTGOV: mentre mentre CONJ !!!!!!!
58 B- FV_C POTGOV: faceva fare V_FIN !S3II!!
59 B- N_C  DET:    una una ART_I FS!!!!!
60 I- N_C  POTGOV: doccia doccia NN FS!!!!P
```

Figure 2. Chunked input for temporal tagger.

```
38 B- P_C PREP: nel in PREP_A MS!!!!!
39 I- P_C POTGOV: villaggio villaggio NN MS!!!!P
40 B- PUNCT_C POTGOV: " " PUNCT !!!!!!!
41 B- N_C POTGOV: kartibubbo kartibubbo NN_P !!!!!!!
42 B- PUNCT_C POTGOV: " " PUNCT !!!!!!!
43 B- DI_C PREP: di di PREP !!!!!!!
44 I- DI_C POTGOV: campobello campobello NN_P !!!!!!!
45 B- DI_C PREP: di di PREP !!!!!!!
46 I- DI_C POTGOV: mazara mazara NN_P !!!!!!!
47 B- P_C PREP: nel in PREP_A MS!!!!!
48 I- P_C POTGOV: trapanese trapanese NN _S!!!!P
49 B- PUNCT_C POTGOV: , , PUNCT !!!!!!!
50 B- FV_C AUX: è essere V_FIN !S3PI!!
51 I- FV_C POTGOV: morta morire V_PP FS!RP!!
52 B- N_C POTGOV: sabato sabato NN MS!!!!P B-TIMEX3
53 B- P_C PREP: per per PREP !!!!!!!
54 I- P_C DET: una una ART_I FS!!!!!
55 I- P_C POTGOV: scarica scarica NN FS!!!!P
56 B- ADJ_C POTGOV: elettrica elettrico ADJ FS!!!!P
57 B- SUBORD_C POTGOV: mentre mentre CONJ !!!!!!!
58 B- FV_C POTGOV: faceva fare V_FIN !S3II!!
59 B- N_C DET: una una ART_I FS!!!!!
```

Figure 3. Final output of the *TETI* Tagger.

in the example 24, i.e. `COND ( POTGOV_lemma equals timexTrigger )`. Once a positive match is found, the local rules are activated, and, as illustrated in example 24, the tagger checks that (i.) the type of chunk in which the temporal trigger words is located which could be either a nominal chunk, or an adjectival one or an adverbial one; that (ii.) the head of the previous chunk (`POTGOV_lemma CHUNK-1`) is not a modifier, and that (iii.) the head of the following chunk (`POTGOV_lemma CHUNK+1`) is neither a modifier or another temporal trigger word. If all local rules are true, then it creates the `TIMEX3` tag, which, in this case, coincides with the temporal trigger chunk ((`BEGIN_AT B_CHUNK`) and (`END_AT E_CHUNK`))[6]. The final output is illustrated in Figure 3.

[6]In all the rules `B_CHUNK` stands for the beginning and `E_CHUNK` for the end of the temporal trigger chunk.

The system does not limit itself to the identification of timexes, but it recognizes and marks signals as well. In fact, the input provided by the chunks allows to detect the temporal prepositions that introduce the timex triggers, which are annotated with their corresponding tag, i.e. `SIGNAL`, according to the TimeML specifications.

The tagger works with a set of 33 rules, including three rules for timexes realized by time or date patterns, which are retrieved by means of regular expressions and a special rule which checks the second following chunk to deal with timexes belonging to Pattern 4 and, possibly, to Pattern 5 and Pattern 6, i.e. five or six consecutive chunks which form a unique `TIMEX3` tag.

C. Semantic relations: a strategy to improve reliability

The major novelty introduced in this work is represented by a cluster of semantic restrictions which must be satisfied during the bracketing phase in order to be compliant to the `TIMEX3` specifications. To illustrate how these restrictions work, consider the following examples:

25.) il venerdì$_{N_C}$ sera$_{N_C}$.
Friday evening.

26.) il periodo$_{N_C}$ 93$_{N_C}$ - 94$_{N_C}$.
the period 93 - 94.

According to TimeML `TIMEX3` specifications, in the example 25 we have two timexes which are in a *part_of* relation one with each other. Consequently, **Pattern 2** applies and we have to create a unique `TIMEX3` tag as the following, i.e `<TIMEX3> il venerdì sera </TIMEX3>`. On the contrary in the example 26, we have three autonomous timexes, and **Pattern 1** applies. In absence of rules which include the contribution of semantic relations between the timex trigger words, we will have only one correct tagging, namely:

- if the rules allow the application of **Pattern 2** to two consecutive N_Cs, then only example 25 would be correctly tagged, while example 26 will result as wrong: `*<TIMEX3>il periodo 93</TIMEX3> - <TIMEX3> 94 </TIMEX3>`;
- if the rules don't allow the application of **Pattern 2** to two consecutive N_Cs, then only example 26 would be correctly tagged, while example 25 will result as wrong: `*<TIMEX3>il venerdì</TIMEX3> <TIMEX3>sera</TIMEX3>`.

As a general procedure, when the tagger identifies the presence of two consecutive temporal trigger words, first it checks for the chunk type of the two trigger words, and if they are of the same type, then it applies rules which take into account the semantic relations between the two trigger words. Since these relations are projected over the entire set of information which forms an entry in the TimEx Dictionary, it is sufficient to further lookup in the dictionary for the relevant semantic relation between the two trigger words to obtain the correct tagging.

IV. System evaluation

In order to verify the reliability of the system we have done an evaluation session. We have manually annotated[7] with the TimeML specifications a subset of 42 articles (16 thousand words) from the Italian treebank ([14]), containing a total of 367 timexes. Due to the fact that the normalization phase has not been implemented yet, the system has been evaluated both with respect to the general task of recognition and bracketing of timexes and for the subtask of modifier recognition. The evaluation of the bracketing comprehends also an evaluation of the system with respect to the tagging of the `SIGNAL` tag. In fact due to the input format, a wrong tagging for signals, corresponds to a wrong bracketing of the timexes. In Table I we report the results obtained by the system.

The columns COR and MISS and INC report, respectively, the number of items correctly identified, those not recognized by the system but present in the corpus and, finally, the number of items both incorrectly annotated and false positives. The overall evaluation of the system is computed in terms of precision (P), recall (R) and F-measure (F). Precision and recall are computed at the tag level, this means that for every instance of a timex correctly bracketed we have assigned one point, in case of partial identification we have assigned zero points.

As for Italian, to our knowledge, four systems have been developed for the task of recognition and normalizations of timexes and three of them implement rule-based methods. Unfortunately, these systems implement the `TIMEX2` specifications which differ from TimeML `TIMEX3` ones with respect to the bracketing of timexes, thus preventing a comparison of the systems' performance. The results obtained show a good overall performance of the system. The error analysis, however, shows that there is still room for improvements. The relatively low number of missing timexes is only in a minimal part a result of missing entries in the TimEx Dictionary. A source of errors is the presence of elliptic phrase heads as in *almeno 4*$_{N\_C}$ *o 5 giorni*$_{N\_C}$, "at least 4 or 5 days", where the first chunk misses the time word trigger. Another source of errors is represented by the relative high number of false positives. These are instances of numeric expressions which have a pattern similar to calendar and time timexes. These errors are mainly due to the fact that we have used regular expressions to identify these kinds of timexes. A possible strategy to avoid these errors could be represented by a more fine-grained POS tagset which could distinguish between bare numeric data and numeric data which correspond to timexes. Apparent dates, i.e. proper names with a timex composing it, like in *Il Sole - 24 Ore*, are not always recognized by the chunker as named entity and, thus, are incorrectly tagged as timexes. The error analysis has also indicated that some rules need refinements, in particular when date patterns and semantic relations co-occur. In fact, the former are missing from the TimEx Dictionary, and this calls for a strategy to insert them. Finally, the incorrect identification of the SIGNAL

[7]The annotation tool used is *Callisto* from MITRE.

Table I
TETI EVALUATION RESULTS.

Tag	TOT.	COR	MISS	INC	P	R	F
TIMEX3	367	321	35	66	82.95	90.17	86.41
TIMEX3:modifier	90	55	12	23	82.09	70.51	75.86

tag has contributed to worsening the results preventing a correct bracketing. Most of them are multiwords like "*fino a*" [up to], or "*rispetto a*" [(with) respect to], which are not recognized as such by the chunker.

As for modifiers the results are less satisfying. We have identified only 90 timex with at least a modifier in it. Only 55 of them have been identified by the system, thus suggesting that some rules which deal with modifiers need to be refined. Moreover, most modifiers are not identified by the systems (column INC) resulting in an incorrect tagging for this subtask. The missing modifiers correspond to numeric values, namely cardinal numbers expressed by words. A possible, though time consuming, solution could be to insert them in the modifiers'dictionary. On the other hand, we claim that the best solution is to exploit the information provided by the chunked text, in particular their position in the chunk, which corresponds to the premodifier position, and the part of speech, which clearly state that these words are numbers, thus improving the tagging.

V. Conclusion and Future Work

The system described in this work is a module of a wider reasearch project on temporal processing of texts in the perspective of Open Domain Question Answering, which includes event detection and identification of the temporal links between the relevant entities, i.e. timexes - timexes, event - event and event - timexes. Though the results are quite good, improvements are needed in order to reduce the number of false positives, which may bias further processes of temporal analysis, like the identification of the temporal relations between an eventuality and a timex. Further research is needed to complete the tagger, in particular for the normalization phase. Provided the good results obtained by using rules for detecting and bracketing, we suggest that this methods could be useful for the normalization phase as well. In order to obtained a good normalizer some issues must be resolved, in particular as far as the identification of the correct anchor is concerned. In this sense previous studies on anaphoric definites represent a starting point for the identification of reliable heuristics for anchor detection.

A further improvement for the system is represented by the creation of a GUI which will allow the realise of the system, once completed, as a web-based tool.

References

[1] J. Pustejovsky, J. Castao, R. Ingria, R. Saurì, R. Gaizauskas, A. Setzer, and G. Katz, "Timeml: Robust specification of event and temporal expressions in text," in *Fifth International Workshop on Computational Semantics (IWCS-5)*, 2003.

[2] J. R. Hobbs and F. Pan, "An ontology of time in the semantic web," *ACM Transaction on Asian Language Information Processing*, vol. 3, no. 1, pp. 66–85, 2004.

[3] L. Ferro, I. Mani, B. Sundheim, and G. Wilson, "Tides temporal annotation guidelines, v. 1.0.2," MITRE, MITRE Technical report MTR01W0000041, 2001.

[4] L. Ferro, G. L., I. Mani, B. Sundheim, and G. Wilson, *Instruction Manual for the Annotation of Temporal Expressions*, MITRE, Washington C3 Center, McLean, Virginia, 2002.

[5] A. Setzer, "Temporal information in newswire articles: an annotation scheme and a corpus study," Ph.D. dissertation, University of Sheffield, 2001.

[6] A. Lavelli, B. Magnini, M. Negri, E. Pianta, M. Speranza, and R. Sprugnoli, "Italian content annotation bank (i-cab): Temporal expressions (v.1.0)," ITC-irst, Trento, Tech. Rep., June 2005.

[7] G. Wilson, I. Mani, B. Sundheim, and L. Ferro, "A multilingual approach to annotating and extracting temporal information," in *Proceedings of the ACL-EACL 2001 Workshop for Temporal and Spatial Information Processing*, 2001, pp. 81–87.

[8] P. Martínez-Barco, E. Saquete, and R. Muñoz, "A grammar-based system to solve temporal expressions in spanish texts," in *Advances in Natural Language Processing, Third International Conference, PorTAL 2002, Proceedings*, E. Ranchod and N. J. Mamede, Eds. Springer, 2002, pp. 53–62.

[9] M. Negri and L. Marseglia, "Recognition and normalization of time expressions itc-irst at tern 2004," ITC-irst, Trento, Tech. Rep., 2004.

[10] F. Schilder, "Extracting meaning from temporal nouns and temporal prepositions," *ACM Transaction on Asian Language Information Processing*, vol. 3, no. 1, pp. 33–50, 2004.

[11] A. Lenci, S. Montemagni, and V. Pirrelli, "*Chunk-it*: An italian shallow parser for robust syntatic annotation," *Linguistica Computazionale, Computational Linguistics in Pisa, special Issue*, vol. XVIII-XIX, pp. 353–86, 2003.

[12] N. Ruimy, M. Monachini, E. Gola, N. Calzolari, M. D. Fiorentino, M. Ulivieri, and S. Rossi, "A computational semantic lexicon of italian: *SIMPLE*," *Linguistica Computazionale, Computational Linguistics in Pisa, special Issue*, vol. XVIII-XIX, pp. 821–64, 2003.

[13] W. Croft and D. Cruse, *Cognitive Linguistics*. CUP, 2005.

[14] S. Montemagni, F. Barsotti, M. Battista, N. Calzolari, O. Corazzar, A. Lenci, V. Pirelli, A. Zampolli, F. Fanciulli, M. Massetani, R. Raffaelli, R. Basili, M. T. Pazienza, D. Saracino, F. Zanzotto, N. Mana, F. Pianesi, and R. Delmonte, "The syntactic-semantic treebank of italian. an overview." *Linguistica Computazionale, Computational Linguistics in Pisa, special Issue*, vol. XVIII-XIX, pp. 461–93, 2003.

[15] I. Mani and B. Schiffman, "Temporally anchoring and ordering events in news," in *Time and Event Recognition in Natural Language*, J. Pustejovsky and R. Gaizauskas, Eds. John Benjamins Publishing, 2007.