Yet Another Platform for Extracting Knowledge from Corpora

Francesca Fallucchi, Fabio Massimo Zanzotto

DISP - University of Rome "Tor Vergata" (Italy)
fallucchi@info.uniroma2.it, zanzotto@info.uniroma2.it

Abstract

The research field of "extracting knowledge bases from text collections" seems to be mature: its target and its working hypotheses are clear. In this paper we propose a platform, YAPEK, i.e., *Yet Another Platform for Extracting Knowledge from corpora*, that wants to be the base to collect the majority of algorithms for extracting knowledge bases from corpora. The idea is that, when many knowledge extraction algorithms are collected under the same platform, relative comparisons are clearer and many algorithms can be leveraged to extract more valuable knowledge for final tasks such as Textual Entailment Recognition. As we want to collect many knowledge extraction algorithms, YAPEK is based on the three working hypotheses of the area: the *basic hypothesis*, the *distributional hypothesis*, and the *point-wise assertion patterns*. In YAPEK, these three hypotheses define two spaces: the space of the *target textual forms* and the space of the *contexts*. This platform guarantees the possibility of rapidly implementing many models for extracting knowledge from corpora as the platform gives clear entry points to model what is really different in the different algorithms: the feature spaces, the distances in these spaces, and the actual algorithm.

1. Introduction

The research field of "extracting knowledge bases from text collections" seems to be mature. **Its target is clearly defined**: methods and systems want to extract, with the minimal supervision, equivalences or relations between relevant *textual elements*, i.e., words, word sequences, or generalized patterns. **Its main working hypotheses are stable**: (1) the *Basic Hypothesis* stating that similarities or relations between textual elements can be derived only looking at these textual elements (e.g., (Jacquemin, 2001)); (2) *Harris' Distributional Hypothesis* (Harris, 1964) that gives a way to induce similarity between textual elements looking at their contexts in a corpus; (3) the *Point-wise Assertion Pattern Exploitation Hypothesis* saying that specific textual patterns can be used to extract relevant relations between textual elements (firstly used in (Robison, 1970)). **Its main strategies for realizing algorithms are stable**: there are three strategies for going backward and forward between the two spaces, the space of the target textual elements and the space of the contexts, defined by the above *three working hypotheses*. Finally, **its principles and algorithms are used in many fields**: lexical acquisition (e.g., (Baldwin et al., 2005)), terminology extraction and structuring (e.g., (Jacquemin, 2001)), ontology learning in the area of the semantic web (e.g., (Medche, 2002)), and learning knowledge for knowledge-intensive applications (e.g., Question Answering, Information Extraction, Textual Entailment Recognition, etc.).

Despite all this, frontiers and challenges of the field are somewhat unclear and real innovations of knowledge acquisition models presented in these years are difficult to determine. Two important points are still missing. Firstly, the field is still looking for *a shared evaluation method that allows comparison between different approaches*. This evaluation method is particularly difficult to define as the failure of the Ontology Learning Challenge in the Pascal Network of Excellence[1] seems to demonstrate. Secondly, it is still unclear how to *effectively use all the extracted knowledge*: attempts to use extracted knowledge in tasks such as Textual Entailment Recognition (Bar Haim et al., 2006) had very limited impact on performance.

In this paper we propose a platform, YAPEK, i.e., *Yet Another Platform for Extracting Knowledge from corpora*, that wants to be the base to collect the majority of algorithms for extracting knowledge bases from corpora. The idea is that, when many knowledge extraction algorithms are collected under the same platform, relative comparisons are clearer and many algorithms can be leveraged to extract more valuable knowledge for final tasks such as Textual Entailment Recognition. As we want to collect many knowledge extraction algorithms, YAPEK is based on the three working hypotheses of the area: the *basic hypothesis*, the *distributional hypothesis*, and the *point-wise assertion patterns*. In YAPEK, these three hypotheses define two spaces: the space of the *target textual forms* and the space of the *contexts*. This platform guarantees the possibility of rapidly implementing many models for extracting knowledge from corpora as the platform gives clear entry points to model what is really different in the different algorithms: the feature spaces, the distances in these spaces, and the actual algorithm.

2. Spaces and working hypotheses

Maby knowledge extraction methods are based on the three working hypotheses: Basic Hypothesis (BH), Harris' Distributional Hypothesis (DH), and Point-wise Assertion Pattern exploitation hypothesis (PAP). As our aim is to define a platform that can be used to model many of these knowledge extraction methods, we need to clarify how these hypotheses interact for extracting equivalences or relations among textual forms, i.e., how they can be used to capture *language variability*.

The three hypotheses clearly define different spaces where *target textual forms* and *contexts* may be represented. These spaces are, respectively, the *space of the target textual forms* and the *space of the contexts*. These spaces are

[1] http://www.pascal-network.org

strictly interconnected and what is done in one space can be exploited in the other. Fig. 1 shows the two spaces and presents an example, that can clarify how the three hypotheses work together.

2.1. The Basic Hypothesis

The well-known *Basic Hypothesis* (BH) is applied when the relation or the similarity between the textual forms, w_i and w_j, is determined looking only to w_i and w_j and to external knowledge repositories such as WordNet (Miller, 1995). The similarity or oriented relation between textual forms may be defined as $r_{BH}(w_i, w_j)$. The $r_{BH}(w_i, w_j)$ function works only in the space of the target textual forms. For example, an $r_{BH}(w_i, w_j)$ may detect the similarity between w_i and w_j, where w_i and w_j are lemmas, using similarity measures defined over WordNet as the ones collected in (Pedersen et al., 2004). Contexts are not used. This r_{BH} can then detect the similarity between $compose$ and $constitute$ (Fig. 1), i.e., $r_{BH}(compose, constitute)$, by looking at the two words and an external resource only. More complex ways of computing the similarity using the basic hypothesis have been proposed when the forms are word sequences such as terms (e.g., as in (Jacquemin, 2001)) or complete sentences (e.g., (Dolan et al., 2004; Burger and Ferro, 2005)).

2.2. The Distributional Hypothesis

The well-known *Distributional Hypothesis* (DH) (Harris, 1964) allows to determine whether or not two forms are in relation by looking at their contexts. These latter are found in a corpus. The hypothesis states that *two forms are similar if these are found in similar contexts*, i.e., $r_{DH}(w_i, w_j) \approx sim_{DH}(C(w_i), C(w_j))$ where $C(w_i)$ and $C(w_j)$ are the contexts of the forms w_i and w_j in a given corpus. In the example (Fig. 1), using the distributional hypothesis, the similarity between $compose$ and $constitute$ is determined as $C(constitute)$ and $C(compose)$ overlap. Then, $sim_{DH}(C(w_i), C(w_j))$ is high. The words $compose$ and $constitute$ are similar as they are found in similar contexts such as *the sun is constituted of hydrogen* and *the sun is composed of hydrogen*, i.e., the contexts containing both *sun* and *oxygen*. The $sim_{DH}(C(w_i), C(w_j))$ similarity is defined in the space of the contexts.

2.3. The Point-wise Assertion Pattern Exploitation Hypothesis

Finally, *point-wise assertion pattern exploitation hypothesis* (PAP) (first used in (Robison, 1970) and frequently used afterwards in (Hearst, 1992a; Szpektor et al., 2004; Pantel and Pennacchiotti, 2006)) allow to determine relations among relevant words using textual patterns, e.g. *X is constituted of Y* may be used to determine the *part-of* relation among X and Y. These point-wise assertion patterns are defined in textual form space and they are used in the context space to retrieve textual elements in relation. In the example (see Fig. 1), the pattern *X is constituted of Y* is used to find the *part-of* relation between the two words *sun* and *oxygen*. More generally, $C(constitute)$ contains, after some statistical filter, words that are in a *part-of* relation. Moreover, the equivalences determined in the textual form space, e.g., the equivalence between *constitute* and *compose*, can be used to further augment words that are in a given relation. Using the equivalence between $constitute$ and $compose$, elements in *part-of* relation can be found in $C(constitute) \cup C(compose)$.

3. Using the three hypotheses: classification of the learning algorithms

The three hypotheses and the related spaces (i.e, the context space and the target form space) offer a very interesting tool to investigate similarities among linguistic forms. Yet, alone these do not give a way to extract equivalence classes of forms from a corpus. In literature, different algorithms have been proposed. These algorithms can be divided in three main classes: *direct*, *indirect*, and *iterative* model. In the *direct* strategy, textual elements are previously selected and, then, similarities are sought in the context space (e.g., (Lin and Pantel, 2001)) (Sec. 4.1.). In the *indirect* strategy, contexts are previously selected and, then, equivalent textual elements emerge using clustering methods (e.g., (Hearst, 1992a)) (Sec. 4.2.). In the *iterative* strategy, the algorithms use seeds in one of the spaces and, going back and forward in the two spaces, incrementally augment equivalent textual elements (e.g., (Szpektor et al., 2004; Pantel and Pennacchiotti, 2006)) (Sec. 4.3.).

4. Basic notations

In this section we give the basic notations we are using. Trying to formalize, given a set of target words W in the target word space and a set of contexts Ctx in the context space drawn from a corpus C, we can define a function that associates the contexts to each subset of words:

$$\mathcal{C} : 2^W \rightarrow 2^{Ctx} \tag{1}$$

Contexts may be represented in a feature space $F = F_1 \times ... \times F_n$. As any vector space model, this space hosts also the intensional representation of sets of contexts $\mathcal{I}(Ctx')$ (e.g., their centroids). The function that computes the intensional representations is:

$$\mathcal{I} : 2^{Ctx} \rightarrow F \tag{2}$$

The similarity between two sets of contexts (or between their intensional representations) can then be computed over the space F. In the following we will use $w \in W$ and $c \in Ctx$ in the similarities to indicate, respectively, the singletons $\{w\}$ and $\{c\}$.

4.1. Discovering similarity using the *direct* approach

The **direct** approach is fairly studied and it is applicable when the corpus C is a-priori known. Given a set W of relevant words or linguistic structures, the corpus C is scanned for each element $w \in W$. Contexts for each w are gathered and represented in the feature space $\mathcal{F}$. Pairs (w', w'') in $W \times W$ are ranked according to $sim(w', w'')$ and these are retained as good pairs if their similarity is greater than a threshold α, i.e. $sim(w', w'') > \alpha$. Sometimes, to improve selectiveness this threshold depends on one of the two elements, i.e. $\alpha_{w'}$. The applicability to known corpora depends on two factors. The first is that the set W

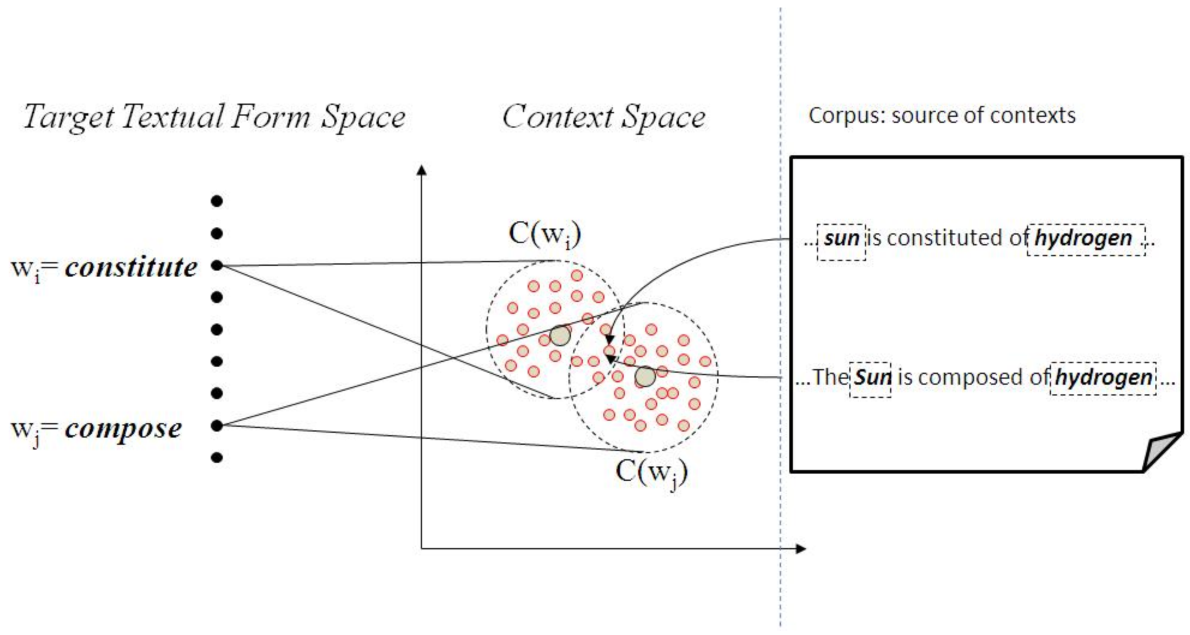
Target Textual Form Space
Context Space
Corpus: source of contexts
w_i= constitute
w_j= compose
$C(w_i)$
$C(w_j)$
...sun is constituted of hydrogen ...
...The Sun is composed of hydrogen...

Figure 1: Two spaces for the three hypotheses

has to be defined a-priori. This is generally done analysing the frequency, or similar indicators, of the words w in the corpus. The second applicability limitation depends on the fact that the contexts of each w have to be represented in the $\mathcal{F}$ space. This means that the corpus has to be completely scanned. Only after the computation of the similarity between different words can be done.

The general algorithm direct_learning(C), that directly applies the Distributional Hypothesis, is described in the following. It takes in input a corpus C and return a set of equivalence classes W_i containing words or linguistic structures:

```
direct_learning($C$) returns ($W_1$,...,$W_n$)
Given a corpus $C$ and the releted function $\mathcal{C}$:
  1. Let $W$ be the most important elements in $C$
  2. For each $w_i \in W$:
     (a) initialize $W_i = \{w_i\}$
     (b) retreive all the contexts $\mathcal{C}(w_i)$ and map them in the feature space $F$
     (c) eventually compute $\mathcal{I}(\mathcal{C}(w_i))$
  3. For each $(w_i, w_j) \in W \times W$:
     (a) comupute $sim_{ctx}(\mathcal{C}(w_i), \mathcal{C}(w_j))$
     (b) if $sim_{ctx}(\mathcal{C}(w_i), \mathcal{C}(w_j)) > \alpha_{w_i}$ then put $w_j$ in $W_i$
```

The above algorithm has been largely employed. As we already discussed in the previous sections, the methods differ in the target of the analysis and the feature space in which the similarity is computed.

In (Glickman and Dagan, 2003), the direct_learning has been used to extract similarity between verbs from a single corpus. The elements represented in the feature space F were then intensional representations of verb contexts. The actual features were determined by the pair $(VerbArg, ArgFiller)$ where $VerbArg$ is one of the possible verb arguments (e.g., the $subject, object, modifier - from, modifier - in, ...$) and $ArgFiller$ is a word sequence. The value of this feature was related to the frequency and the inverse document frequency.

In (Lin and Pantel, 2001), the algorithm has been used to discover equivalence relations between verbal linguistic expressions connecting two arguments X and Y, e.g. *X solved Y* $\approx$ *X found a solution to Y*. Each of these verbal linguistic expressions is called pattern p. The idea was to represent in a feature space the possible filler of the slots X and Y. The feature space represents intensionally a set of contexts of each pattern. The features were (s, w) where s is the slot X or Y and w is a possible word filling the slot. Given a set of contexts where the pattern pat has been found (the set will be called pat as the pattern), the feature values of the $\mathcal{I}(pat)$ are computed as follows:

$$\mathcal{I}_{(s,w)}(pat) = mi(pat, s, w) \quad (3)$$

where $mi(pat, s, w)$ is the point-wise mutual information (Church and Hanks, 1989):

$$mi(pat, s, w) = log\frac{p(pat, s, w)}{p(pat, s)p(s, w)} \quad (4)$$

Probabilities are estimated with the maximum likelihood principle over the corpus. We define the vectors I_s (where s is the slot X or Y) as the vectors having 1 in the feature (s, w) and 0 otherwise. The similarity $sim(p_1, p_2, s)$

between p_1 and p_2 according to the slot s is defined as follows :

$$sim(p_1, p_2, s) = \frac{(\mathcal{I}(p_1) + \mathcal{I}(p_2))I(p_1, p_2)I_s^T}{\mathcal{I}(p_1)I_s^T + \mathcal{I}(p_2)I_s^T} \quad (5)$$

where $I(p_1, p_2)$ is a matrix whose elements on the diagonal are:

$$I_{(s,w),(s,w)}(p_1, p_2) = \begin{cases} 1 \quad \texttt{if} \quad \mathcal{I}_{(s,w)}(p_1) > 0 \ \texttt{and} \\ \qquad\qquad \mathcal{I}_{(s,w)}(p_2) > 0 \\ 0 \qquad\qquad \texttt{otherwise} \end{cases}$$

and elements out of the diagonal are 0. The similarity $sim_p(p_1, p_2)$ between two patterns is then computed as follows:

$$sim_p(p_1, p_2) = \sqrt{sim(p_1, p_2, X) \times sim(p_1, p_2, Y)} \quad (6)$$

4.2. The *indirect* approaches: discovering similarities starting from the context space

The **indirect** approaches are not so popular but they are extremely important. The main idea here is that now we already have some aggregation points in the context space. These aggregation points can be seen as prototypes for the equivalence classes of elements in the space of the target forms. The first and important example of the *indirect* approach is the extraction of equivalent forms that express the relation *isa* (see (Hearst, 1992b)). The method is very intuitive. It assumes a pre-existing *isa* hierarchy $\mathcal{H}$ among words, e.g., a dictionary or a lexical knowledge base. Word pairs $(w_i, w_j) \in \mathcal{H}$ indicate that w_i *isa* w_j. Then, if we find w_i and w_j in the same context, it is very likely that this context is a lexicalization of the *isa* relation. For example, let's assume that $(venus, planet) \in \mathcal{H}$. If we scan the web for contexts where these two words are together, it is likely to come across sentences like *Venus is the second planet from the Sun*. This indicates that the target form *is the second* is likely to be a lexicalization of elements in the *isa* equivalence class. The pair $(venus, planet)$ is then one of the prototypical contexts of the *isa* equivalence class in the target form space. In a bag-of-word context feature space, it can be seen as the vector where the two features, *venus* and *planet*, are active.

The indirect approach wants an a-priori knowledge K = $\{K_1,...,K_n\}$. This should describe some prototypes in the context space that represent an equivalence class in the target form space. Given K, an indirect approach will then produce the set of equivalence forms (W_1,...,W_n). The algorithm is the following:

```
**indirect_learning**($C$,$K_1, ..., K_n$) returns ($W_1$,...,$W_n$)
Given a corpus $C$ and the related function $\mathcal{C}$:
  1. Given an $i$, for each $k \in K_i$:
     (a) retreive all the contexts $\mathcal{C}^{-1}(k)$ that is all $c \in C$ such that $sim(k, c) > \tau$
     (b) extract recurrent forms from $\mathcal{C}^{-1}(k)$ and add them in $W_i$
```

The indirect_learning is very simple. Methods can be very different as they can vary the similarity function $sim(k, c)$ and the way to extract the recurrent forms from $\mathcal{C}^{-1}(k)$.

4.3. The *iterative* approaches: Anchor-based algorithms

The **iterative** approaches (e.g. (Ravichandran and Hovy, 2002; Szpektor et al., 2004)) have been proposed to apply the Distributional Hypothesis and the Point-wise assertion pattern when the corpus C is a-priori not known (e.g. the Web that is potentially infinite). These methods go back and forward in the context space and in the target form space. The problem is that the set W can't be computed in advance. If W were somehow given, the more similar words to a word w can be found only when $sim_w(w, w')$ has been computed for each $w' \in W$. This is generally unfeasible due to the large size of the corpus. The pursued idea is the following. Given a seeding set of words or linguistic structures W_S considered similar or realising a unique semantic relation (e.g., the *is-a* relation such as in (Hearst, 1992b)), a set C_S of prototypical contexts are extracted. Each element c in the set C_S that has some important property is called *anchor*. An anchor highly characterises the contexts where the set of words W_S appears. For each element c, it is then possible to derive the set of words:

$$W_c = \mathcal{C}^{-1}(\{c' \in CTX | sim_{ctx}(c, c') > \alpha\}) \quad (7)$$

These sets can be used to enrich the original set of word W with similar words. The similarity is always estimated using the similarity between contexts.

```
**iterative_learning**($C$,($W'_1$,...,$W'_N$))
returns ($W_1$,...,$W_N$)
Given a corpus $C$ and the releted function $\mathcal{C}$:
  1. For each $W'_i$:
     (a) $W_i = W'_i$:
     (b) until $|W_i|$ augments
         i. select a relevant set $W' \subseteq W_i$
         ii. compute $I_{W'} = \mathcal{I}(\mathcal{C}(W'))$
         iii. extract $W'' = \mathcal{C}^{-1}(\{c' \in CTX | sim_{ctx}(c, I_{W'}) > \alpha\})$
         iv. select the most relevant $W''' \subseteq W''$
         v. $W_i = W_i \cup W'''$
```

Ravichandran and Hovy (Ravichandran and Hovy, 2002) explored the power of surface text patterns for open-domain Question Answering systems. They recognized that in several Q/A systems specific types of answer are expressed by using characteristic phrases (that could be described in regular expressions). They described a pattern-learning algorithm and focused on scaling relation extraction to the Web. In fact with their algorithm it is possible to infer surface patterns from a small set of instance seeds (the anchor in this

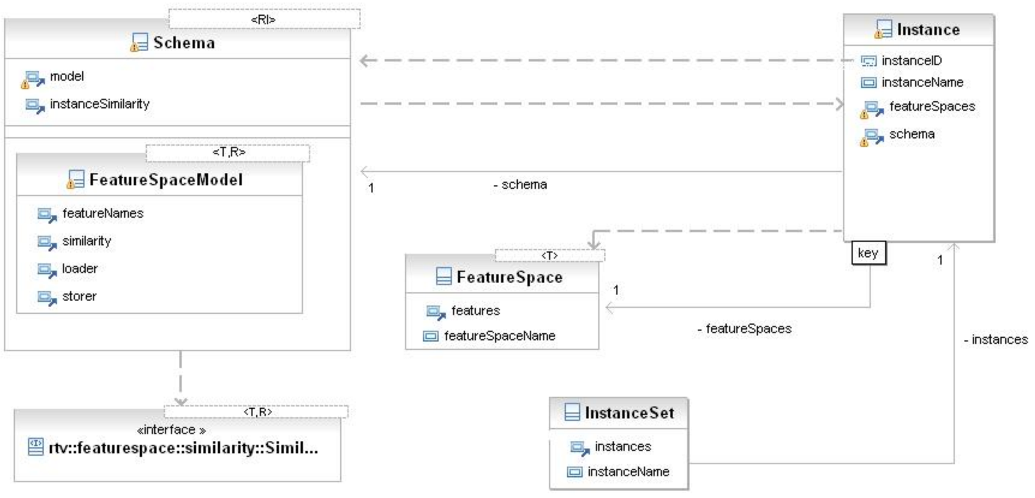

Figure 2: Partial classes diagram of the feature space objects.

approach) by extracting substrings relating seeds in corpus sentences. The presence of any variant of the answer term causes the same treatment as for the original answer term. Nevertheless the patterns cannot handle long-distance dependencies. The approach has been tested on several relations providing good results for specific relations (such as birthdate) whereas lower precision was revealed for generic and frequent ones (as is-a and part-of). Also this algorithm is in the line of learning the extracting approach.
Szpektor et al. (Szpektor et al., 2004) defined a fully unsupervised learning algorithm for web-based extraction of entailment relations. By assuming that the same meaning can be expressed in a text in a huge variety of surface forms, they focused on acquiring paraphrase patterns that represent different forms in which a certain meaning can be expressed. This approach has been applied to acquire entailment relations from the Web. Paraphrase acquisition results in finding linguistic structures (called templates) sharing the same lexical elements describing the context of a sentence. These lexical elements are used as anchors. Templates extracted from different sentences, while connecting the same anchors, are assumed to paraphrase each other. We recognized a structuring model for which we distinguish between syntactic anchors (such as subject, object, verb) and context anchors (such as prepositional phrase). Main problems relate to both finding matching anchors and identifying template structure. Specific algorithms have been developed for both problems. A broad range of semantic relations varying from synonymy to more complex entailment have been extracted.

5. YAPEK: Yet Another Platform for Extracting Knowledge from corpora

Given the analysis of the previous section, the definition of a knowledge extraction platform is straightforward. Knowledge extraction methods are largely based on the above three working hypotheses. The differences depend on four important aspects:

- *the target information*: the target forms and the relations they want to extract
- *features and feature values*: what are the spaces that are defined by the three hypotheses
- *distance or similarity*: what are the similarity functions that needs to be defined in these spaces
- *how these two spaces are used*: the knowledge-extraction methods backward and forward between the two spaces using different strategies

In YAPEK, it is possible to control all the four aspects using clear APIs. This gives the possibility of implementing many of the approaches for extracting knowledge from corpora. The platform gives APIs for defining: the target textual forms defined in the target textual form space, the feature vectors in the context space, and the similarity functions, $r_{BH}(w_i, w_j)$ and $sim_{DH}(C(w_i), C(w_j))$, that needs to be defined in these spaces.
The realized system allows to implement different approaches for extracting knowledge. The system analyzes the given corpus according to the strategy used for realizing knowledge extraction methods. Moreover it allows to implement the above three working hypotheses thanks to the general definition of the components of the system.

5.1. The feature space

The three working hypotheses define different spaces where *target textual forms* and *contexts* may be represented.
The relevant aspect is that each instance can be completely described by referring to a single space that includes all the possible feature subspaces related to the considered instance.
For each instance we can get specific information according to the selected feature space.
The partial class diagram is shown in Fig. 2.
The principal feature space objects are:

- Schema: is the model to define the feature space

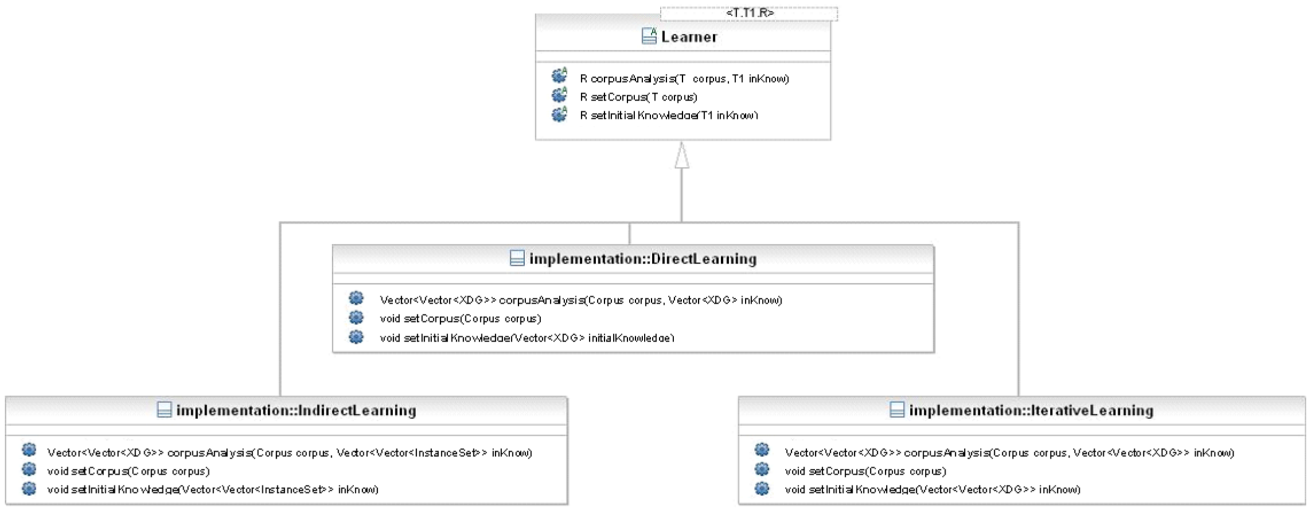
<T,T1,R>
Learner
R corpusAnalysis(T corpus, T1 inKnow)
R setCorpus(T corpus)
R setInitialKnowledge(T1 inKnow)
implementation::DirectLearning
Vector<Vector<XDG>> corpusAnalysis(Corpus corpus, Vector<XDG> inKnow)
void setCorpus(Corpus corpus)
void setInitialKnowledge(Vector<XDG> initialKnowledge)
implementation::IndirectLearning
Vector<Vector<XDG>> corpusAnalysis(Corpus corpus, Vector<Vector<InstanceSet>> inKnow)
void setCorpus(Corpus corpus)
void setInitialKnowledge(Vector<Vector<InstanceSet>> inKnow)
implementation::IterativeLearning
Vector<Vector<XDG>> corpusAnalysis(Corpus corpus, Vector<Vector<XDG>> inKnow)
void setCorpus(Corpus corpus)
void setInitialKnowledge(Vector<Vector<XDG>> inKnow)

Figure 3: Partial classes diagram of the learning algorithm

- InstanceSet: contains all instances extracted from the corpus;
- Instance: contains a single instance and its feature space.
- FeatureSpace: contains the space that includes all feature subspaces.

Each new feature space implements the class *FeatureSpace* and it is necessary to add it to the created instance of *Schema*. Then it is possible to add feature subspaces to the new feature space.
It is also necessary to define the functions, e.g. the similarity, that operate in this space.
The similarity implements the class *Similarity* and it is possible to define different similarities according to the kind of the feature space and the working hypothesis.

5.2. The eXtended Dependency Graph

To model both target textual forms and context we use also the computational version of the eXtended Dependency Graph (XDG) (Basili and Zanzotto, 2002). An XDG is a dependency graph whose nodes C are *constituents* and whose edges D are the *grammatical relations* among the constituents, i.e. $\mathcal{XDG} = (C, D)$.
Constituents (i.e. $c \in C$) are classical syntactic trees with explicit *syntactic heads*, $h(c)$, and *potential semantic governors*, $gov(c)$. Constituents can be represented as feature structures, having as relevant features:

- the $head$ and the gov, having as domain $\mathcal{C}$ (the set of trees and subtrees derived from C), and representing respectively *syntactic heads* and *potential semantic governors*;
- the $type$ representing the syntactic label of the constituent and having as domain Λ.

Moreover, a constituent can be either *complex* or *simple*. A *complex constituent* is a tree containing other constituents as sons (which are expressed by the feature $subConstituents$). A *simple constituent* represents a leaf node, i.e., a token span in the input sentence, that carries information about lexical items through the following features:

- $surface$, representing the actual form found in the token span,
- $lemma$, taking values in the lexicon $\mathcal{L}$ and representing the canonical form of the target surface,
- $morphology$, representing the morphological features of the inflected form.

On the other hand, dependencies in $(h, m, T) \in D$ represent typed (where T is the type) and ambiguous relations among a constituent, the $head$ h, and one of its $modifiers$ m. The ambiguity is represented using $plausibility$, a real value ranging between 0 and 1, where 1 stands for unambiguous. Then, D is defined as a subset of $C \times C \times \Gamma \times (0, 1]$, where the sets represent respectively the domains of the features $head$, $modifier$, $type$, and $plausibility$.

5.3. The algorithm hierarchy

The realized system gives the possibility of implementing many of the approaches for extracting knowledge from corpus. In the Fig. 3 it is shown the partial classes diagram of the knowledge extraction algorithm.
Each class that extends the *Learner* class implements one of the three strategies for realizing knowledge extraction methods. These classes override the methods inherited from the respective super class and they specify input and output parameters types according to the implemented strategies.
The method *corpusAnalysis()* is the principal method used to extract knowledge from a given corpus. On the base of what has been discussed according to the adopted approach the method has different input parameters type whereas

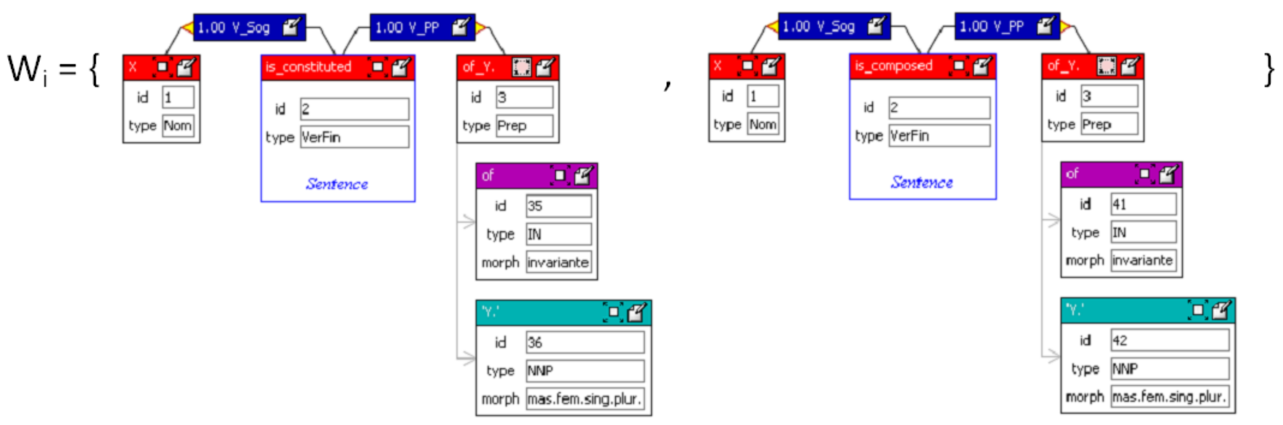

Figure 4: Example of a equivalence class

the output parameter type is the vector of the equivalence classes for all approaches.
Fig.4 shows an example of the equivalence class of the forms *X is constituted of Y* and *X is composed of Y*. The space of these forms are reported in Fig. 1.
The equivalence class W_i, in this example, contains the linguistic structure of the two contexts containing both X and Y . The XDGs of these two forms is obtained and they represent the grammatical relation among constituents.
The constituents *is_constituted* and *is_composed* are the principal verb constituents of the two forms. These verbs have a grammatical relation `V_Sog` with the first variable X and a grammatical relation `V_PP` with a constituent that contains the second variable Y.
In *DirectLearnig*, *corpusAnalysis()* has in input the corpus and a vector of the most important elements in the corpus ($Vector < XDG >$) and returns a set of equivalence classes. In *IndirectLearning* the method has in input the corpus and a vector of a-priori knowledge. Each K_i is a vector of *InstanceSet* i.e. the vector contains all instances and its feature space. In *IterativeLearnig* the input consists of the corpus and a vector of a-priori equivalence classes that we represented with a vector of XDG.

6. Conclusions and future work

In this paper we propose a platform, YAPEK, i.e., *Yet Another Platform for Extracting Knowledge from corpora*, that wants to incorporate many of the algorithms for extracting knowledge bases from corpora. This is a very interesting computational model that can help in better comparing knowledge acquisition algorithms. We plan to include in this platform many of the existing algorithms.

7. References

Timothy Baldwin, Anna Korhonen, and Aline Villavicencio, editors. 2005. *ACL-SIGLEX Workshop on Deep Lexical Acquisition*, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The II PASCAL RTE challenge. In *PASCAL Challenges Workshop*, Venice, Italy.

Roberto Basili and Fabio Massimo Zanzotto. 2002. Parsing engineering and empirical robustness. *Natural Language Engineering*, 8/2-3.

John Burger and Lisa Ferro. 2005. Generating an entailment corpus from news headlines. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 49–54, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vancouver, Canada.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of Coling 2004*, pages 350–356, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Oren Glickman and Ido Dagan. 2003. Identifying lexical paraphrases from a single corpus: A case study for verbs. In *Proceedings of the International Conference Recent Advances of Natural Language Processing (RANLP-2003)*, Borovets, Bulgaria.

Zellig Harris. 1964. Distributional structure. In Jerrold J. Katz and Jerry A. Fodor, editors, *The Philosophy of Linguistics*, New York. Oxford University Press.

Marti A. Hearst. 1992a. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 15th International Conference on Computational Linguistics (CoLing-92)*, Nantes, France.

Marti A. Hearst. 1992b. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 15th CoLing*, Nantes, France.

Christian Jacquemin. 2001. *Spotting and Discovering Terms through Natural Language Processing*. Massachusetts Institue of Technology, Cambrige, Massachussetts, USA.

Dekang Lin and Patrick Pantel. 2001. DIRT: discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, pages 323–328.

Alexander Medche. 2002. *Ontology Learning for the Se-*

mantic Web, volume 665 of *Engineering and Computer Science*. Kluwer International.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120, Sydney, Australia, July. Association for Computational Linguistics.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *Proc. of 5th NAACL*, Boston, MA.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th ACL Meeting*, Philadelphia, Pennsilvania.

Harold R. Robison. 1970. Computer-detectable semantic structures. *Information Storage and Retrieval*, 6(3):273–288.

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcellona, Spain.