Probabilistic Ontology Learner in Semantic Turkey

Francesca Fallucchi, Noemi Scarpato,
Armando Stellato, and Fabio Massimo Zanzotto

Disp, University "Tor Vergata" Rome, Italy
{fallucchi,scarpato,stellato,zanzotto}@info.uniroma2.it

Abstract. In this paper we present the Semantic Turkey Ontology Learner (ST-OL), an incremental ontology learning system, that follows two main ideas: (1) putting final users in the learning loop; (2) using a probabilistic ontology learning model that exploits transitive relations for inducing better extraction models.

1 Introduction

Ontologies and knowledge repositories are important components in Knowledge Representation (KR) and Natural Language Processing (NLP) applications. Yet, to be effectively used, ontologies and knowledge repositories have to be large or, at least, adapted to specific domains. Even huge knowledge repositories such as WordNet [1] are extremely poor when used in specific domains such as the medical domain (see [2]). Studying methods and building systems for automatically creating, adapting, or extending existing knowledge repositories using domain texts is a very important and active area.

A large variety of methods have been proposed: ontology learning methods [3–5] in KR as well as knowledge harvesting methods in NLP such as [6, 7]. These learning methods use variants of the distributional hypothesis [8] or exploit some induced lexical-syntactic patterns (originally used in [9]). The task is generally seen as a classification (e.g., [10, 11]) or a clustering (e.g., [4]) problem. This allows the use of machine learning or probabilistic models.

Models for automatic creating knowledge repositories generally exploit existing structured knowledge such as existing thesauri. Methods based on the Hearst's work [6] use existing pairs of words in a given semantic relation to extract patterns from corpora. These patterns are then used to induce novel pairs of words that are in the same semantic relation. For example, the pair of words *Bush* and *New Haven* are known examples of the semantic relation *has_born_in*. These can be used to extract from corpora that *is the birthplace of* is a good pattern to induce other instances of the above relation. Yet, these models hardly exploit the formal properties of the target relation. Then, these models do not properly exploit information that can be indirectly derived for existing data.

Some semantic relations such as hyperonymy and part-of have an extremely important property that is transitiveness. Exploiting this property along with

existing knowledge repositories during the discovering phase may help in building better knowledge extraction and structuring models. Such an idea is explored in [11, 12].

Automatic models for extracting ontological knowledge from texts do not have the performance needed to extend existing ontologies with a high degree of accuracy. Then, resulting automatically expanded ontologies can become totally useless. Generally, systems for augmenting ontologies extracting information from texts foresee a manual validation for assessing the quality of ontology expansion. Yet, these systems do not use the manual validation for refining the information extraction model that proposes novel ontological information. The idea here is to prefer methods that can use decisions of final users to incrementally refine the model for extracting ontological information from texts, i.e., each decision of final users is exploited in refining the parameters of the extraction model. Including these new examples as training for a machine helps in augmenting the performances of the automatic extractor as shown in [13].

In this paper we present the Semantic Turkey Ontology Learner (ST-OL), an incremental ontology learning system, that follows two main ideas: (1) putting final users in the learning loop; (2) using a probabilistic ontology learning model that exploits transitive relations for inducing better extraction models.

The paper is organized as follows. We firstly review the related work (Sec. 2). We present the ideas behind our new ontology learning system (Sec. 3). We then introduce the system that follows the above principles (Sec. 4). Finally, we draw some conclusions (Sec. 5).

2 Related Work

Exploiting the above (and other) algorithms and techniques for inducing ontological structures from texts, different approaches have been devised, followed and applied regarding how to properly exploit the learned objects and translate them into real ontologies through dedicated editing tools. This is an aspect which is not trivially confined to importing induced data inside an existing (or empty) ontology, but identifies iterative processes that could beneficiate of properly assessed interaction steps with the user, giving life to novel ways of interpreting ontology development.

One of most notable examples of integration between ontology learning systems and ontology development frameworks are offered by Text-to-Onto [14], an ontology learning module for the KAON tool suite, which discovers conceptual structures from different kind of sources (ranging from free text to semi structured information sources such as dictionaries, legacy ontologies and databases) using knowledge acquisition and machine learning techniques; OntoLT [15], is a Protégé [16] plug-in able to extract concepts (classes) and relations (Protégé slots or Protégé OWL properties) from linguistically annotated text collections. It provides mapping rules, defined by use of a precondition language, that allow for a mapping between extracted linguistic entities and classes/slots.

An outdated overview of this kind of integrated tools (which is part of a complete survey on ontology learning methods and techniques) can be found in the public Deliverable 1.5 [17] of the OntoWeb project.

A more recent examples is offered by the Text2Onto [13] plug-in for the Neon toolkit [18], a renewed version of Text-To-Onto with improvements featuring ont-model independence (a *Probabilistic Ontology Model* is adopted as a replacement for any definite target ontology language), better user interaction and incremental learning. Lastly, in [19] the authors define a web browser extensions based on the Semantic Turkey Knowledge Acquisition Framework [20], offering two distinct learning modules: a relation extractor based on a light-weight and fast-to-perform version of algorithms for relation extraction defined in [7], and an ontology population module for harvesting data from html tables.

Most of the above define supervised cyclic *develop and refine* processes controlled by domain experts.

3 Incremental Ontology Learning

To efficiently set-up an incremental model for ontology learning, we need to address two issues:

- an efficient way to interact with final users
- an incremental learning model

The rest of the section shows how we obtain this using existing models and existing systems. We start from present the concept of incremental ontology learning (Sec. 3.1). Secondly, we describe the used ontology editor (Sec. 3.2). Finally, we introduce the used ontology learning methodology (Sec. 3.3).

3.1 The concept

The incremental ontology learning process we want to model leverages on the positive interaction between an automatic model for *ontology learning* and the final users. We obtain this positive interaction using one additional component: an *ontology editor*. The overall process is organized in two phases: (1) the *initialization step* and (2) the *learning loop*. In the *initialization step*, the user selects the initial ontology and selects the corpus. The system uses these two elements to generate the first model for learning ontological information from documents. In the *learning loop*, the machine learning component extract a ranked list of pairs (*candidate_concept*,*superconcept*). The user selects, among the first k pairs, the correct ones to be added to the ontology. We can then use these choices to generate both positive and negative training examples for the ontology learning component. When the new ontology extraction model is learnt from the corpus, the updated ontology, and the growing *non-ontology*, the process restarts from the beginning of the loop.

Given a selected corpus C, the initial ontology O_0, and the generic ontology O_i at the iteration i, we can see the incremental learning process as the sequence

of the resulting ontologies $O_0 \dots O_n$. The *transition* function leverage on the ontology learning model M and the interaction with the user UV. This function can be represented as follows:

$$M_C(O_i, \overline{O}_i) = \widehat{O}_{i+1} \overset{UV}{\rightsquigarrow} (O_{i+1}, \overline{O}_{i+1}) \tag{1}$$

where M_C is the model learnt from the corpus, O_i is the ontology at the step i and $\overline{O}_i$ are the negative choices of the users at the same step. This model outputs a ranked list of possible updates of the ontology $\widehat{O}_{i+1}$. The user validation UV on the first k possibilities produces the updated ontology O_{i+1} and the updated *non-ontology* $\overline{O}_{i+1}$. At the initial step, the process has O_0 and $\overline{O}_0 = \emptyset$. The *ontology learner* produces the model $M_C(O_i, \overline{O}_i)$ building feature vectors representing the contexts of the corpus C where we can find pairs of pairs (*candidate_concept*,*superconcept*). These pairs are extracted from the ontology O_i and the *non-ontology* $\overline{O}_i$.

3.2 Semantic Turkey

Semantic Turkey is a Knowledge Management and Acquisition system developed by the Artificial Intelligence Group of the University of Rome, Tor Vergata. Semantic Turkey (ST, from now on) had been initially developed [21] as a web browser extension (currently implemented for the popular Web Browser Mozilla Firefox) for *Semantic Bookmarking*, that is, the process of *eliciting* information from (web) documents, to *acquire* new knowledge and *represent* it through representation standards, while *keeping reference* to its original information sources.

Semantic Bookmarks differ from their traditional cousins in that they abandon the purely partitive semantics of traditional links&folders bookmarking, and promote a new paradigm, aiming at "a clear separation between (acquired) knowledge data (the WHAT) and their associated information sources (the WHERE)". In practice, the user is able to select portions of text from web pages accessed from the browser, and to annotate them in a (user defined) ontology. A neat separation is maintained between ontological resources created through annotation, and the annotations themselves. This way, the user can easily organize its knowledge (by establishing relationships between ontology objects, categorizing them, better defining them through attributes etc...), while keeping multiple bookmarks in a separated space, pointing to ontology resources and carrying with them all information related to the taken annotations (such as the page where the annotation has been taken, its title, the text which was referring to the created/referenced ontology resource etc...). Easy-to-perform drag-and-drop operations were thought to optimize user interaction, by concentrating the different actions accompanying the creation of both the ontological resources and their related annotations in a few mouse clicks.

ST lately evolved [20] in a complete Knowledge Management and Acquisition System based on Semantic Web technologies: by introducing full support for ontology editing and by improving functionalities for annotation&creation,

ST explored a new dimension which has no predecessor in the field of Ontology Development or Semantic Annotation, and is unique to the process of building new knowledge while exploring the web to acquire it. ST new objective was thus reducing the impedance mismatch between domain experts and knowledge investigators on the one side, and knowledge engineers on the other, by providing them with a unifying platform for acquiring, building up, reorganizing and refining knowledge. It is upon this framework that the ontology learning module that we introduce here has been implemented and integrated.

3.3 Probabilistic Ontology Learner

We use the Probabilistic Ontology Learning (POL) [11, 12] to expand existing ontologies with new facts. In POL is possible to take into consideration both corpus-extracted evidences and the structure of the generated ontology. In the probabilistic formulation [11], the task of learning ontologies from a corpus is seen as a maximum likelihood problem. The ontology is seen as a set O of assertions R over pairs $R_{i,j}$. In particular we will consider the *is-a* relation. In this case if $R_{i,j}$ is in O, i is a concept and j is one of its generalization (i.e., the direct or the indirect generalization). For example, $R_{dog,animal} \in O$ describes that *dog* is an *animal* according to the ontology O.

The main probabilities are then: (1) the prior probability $P(R_{i,j} \in O)$ of an assertion $R_{i,j}$ to belong to the ontology O and (2) the posterior probability $P(R_{i,j} \in O|\overrightarrow{e}_{i,j})$ of an assertion $R_{i,j}$ to belong to the ontology O given a set of evidences $\overrightarrow{e}_{i,j}$ derived from the corpus. These evidences are derived from the contexts where the pair (i, j) is found in the corpus. The vector $\overrightarrow{e}_{i,j}$ is a feature vector associated with a pair (i, j). For example, a feature may describe how many times i and j are seen in patterns like *"i as j"* or *"i is a j"*. These among many other features are indicators of an Is-a relation between i and j (see [6]).

Given a set of evidences E over all the relevant word pairs, in [11, 12], the probabilistic ontology learning task is defined as the problem of finding an ontology $\widehat{O}$ that maximizes the probability of having the evidences E, i.e.:

$$\widehat{O} = \arg\max_{O} P(E|O)$$

In the original model [11, 12], this maximization problem is solved with a local search. In the incremental ontology learning model we propose, this maximization function is solved using also the information coming from final users.

In the user-less model, what is maximized at each step is the ratio between the likelihood $P(E|O')$ and the likelihood $P(E|O)$ where $O' = O \cup N$ and N are the relations added at each step. This ratio is called multiplicative change $\Delta(N)$ and is defined as follows:

$$\Delta(N) = P(E|O')/P(E|O) \tag{2}$$

The last important fact is that it is possible to demonstrate that

$$\Delta(R_{i,j}) = k \cdot \frac{P(R_{i,j} \in O|\overrightarrow{e}_{i,j})}{1 - P(R_{i,j} \in O|\overrightarrow{e}_{i,j})} =$$

$$= k \cdot odds(R_{i,j})$$

where k is a constant (see [11]) that will be neglected in the maximization process. We calculate the *odds* using the logistic regression.

Given the two stochastic variables Y and X, we can define as p the probability of Y to be 1 given that X=x, i.e.:

$$p = P(Y = 1|X = x)$$

The distribution of the variable Y is a Bernoulli distribution, i.e.:

$$Y \sim Bernoulli(p)$$

Given the definition of the *logit* as:

$$logit(p) = \ln\left(\frac{p}{1-p}\right) \tag{3}$$

and given the fact that Y is a Bernoulli distribution, the logistic regression foresees that the logit is a linear combination of the values of the regressors, i.e.,

$$logit(p) = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k \tag{4}$$

where $\beta_0, \beta_1, ..., \beta_k$ are called *regression coefficients* of the variables $x_1, ..., x_k$ respectively.

The remaining problem is how to estimate the regression coefficients. This estimation is done using the maximal likelihood estimation to prepare a set of linear equations using the above *logit* definition and, then, solving a linear problem using pseudo-inverse matrix [12]. We will called the logit vector with l, then we estimate the regression coefficients as following

$$\widehat{\beta} = X^T l \tag{5}$$

In the user-oriented incremental ontology learning we propose, the above maximization is done including final users in the loop. In our task we do not find the ontology that maximizes the likelihood of having the evidences E. We calculate the probabilities step by step. Then we present an ordered set of choices to final users that will make the final decision on what to use on the next iteration. The order set is obtained using the logit function as it is equivalent to the order given by the probabilities. For this reason, in the following we will operate directly on the logit rather than on the probabilities. It is possible to calculate the logit vector to the i-th iteration using both equations (3) and (5) and obtained

$$XX^+ \; l_i = \widehat{l}_{i+1} \overset{UV}{\rightsquigarrow} l_{i+1} \tag{6}$$

In each iteration, we calculate the logit vector using the logit vector of the previous iteration. After then the logit vector is changed in the user validation (UV). When the user accepts a new relation its probability is set to 0.99 while

when the user discards a relation, its probability is set to 0.01. The matrix XX^+ is constant for each iteration. Here we have found a matrix XX^+ that is the constant model M_C of the equation (1). The matrix XX^+ only depends on the corpus C and not on the initial ontology. The logit vector l represents both the current ontology O_i and the negative ontology $\overline{O_i}$ as it includes the logit of both probabilities, i.e., 0.99 and 0.01.

4 Semantic Turkey-Ontology Learner (ST-OL)

The model described in previous section has been implemented and integrated in a Semantic Turkey extension called ST Ontology Learner (ST-OL). ST-OL provides a graphical user interface and a human-computer interaction work-flow supporting the incremental learning loop of our learning theory. If the user has loaded an ontology in ST, he can to improve it by adding new classes and new instances using ST-OL. The interaction process is achieved through the following steps:

- an *initialization phase* where the user selects the initial ontology O and the bunch of documents C where to extract new knowledge
- an *iterative phase* where the user launch the learning and validates the proposals of ST-OL

Thus, starting from the initial ontology O and a bunch of documents C, he has the possibility to use an incremental ontology learning model.

For the *initialization phase* (c.f., Sec. 3.1), the User Interface (UI) of ST-OL allows users to select the initial set of documents C (corpus), and to send both the ontology O and the corpus C to the learning module. To start this stage of the process, the user selects *"Initialize POL"* on the ST-OL panel (see Fig. 1). The probabilistic ontology learner analyzes the corpus, finds the contexts for each ontological pair, computes the first extraction model, and, finally, proposes the pairs that are in is-a relation. This first analysis is the more expensive one as it computes the matrix XX^+. Yet, this computation is done only once in the iterative process.

Once this initialization finishes, the *iterative phase* starts. ST-OL enables the button labeled *"Proposed Ontology"*. The effect of this button is to show the initial ontology extended with the pairs proposed by POL. Figure 1 shows an example of an enriched initial ontology.

The main goal of ST-OL is to help in focusing the attention to the good added information. The user has the possibility of selecting the pairs he wants to add among the proposed pairs. To drive the attention towards the good pairs, we use different brightness of red for the different probabilities. More intense tonalities of red represent higher probabilities.

In order to focus only on possibly good pairs, ST-OL only shows pairs above a threshold τ of probabilities. For example, in Fig. 1, the relation, i.e., the pair, between "truck" and "container" is more probable than the relation between "spreader" and "container". Then different red tones are used. At this point, the

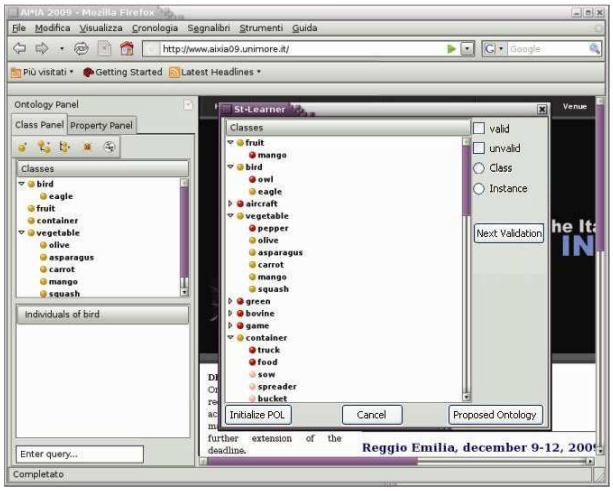

Fig. 1. Initial Ontology extended with the pairs proposed by the POL System

user can accept or reject the information. After acceptance, the new information is stored in the ST ontological repository and can be browsed as usual. through the ontology panel on the Firefox sidebar. Fig. 2 shows what happened when the user accepted two proposed pairs: "mango" as instance of "fruit" and "pepper" as subclass of "vegetable".

The above activity enables the incremental model as it builds an upgraded probability vector. When the user accepts a new pair, ST-OL updates its probability to 0.99. When the user discards the pair, its probability is set to 0.01. These new values are used for the next iteration of the leaning process. After some manual evaluation, the user can decide to update the proposed ontology. Given the probabilistic ontology learning model presented in 3.3, this new evaluation is just a simple multiplication of the existing matrix XX^+ and the new vector. To force the recompilation, the user can use the "*Proposed Ontology*" button.

5 Conclusion

In this paper we presented a computational model POL and a system ST-OL for incremental ontology learning. POL is basically an incremental probabilistic model to learn ontological information from texts and it is designed to positively exploit a probabilistic ontology learning method within a learning loop that includes final users. ST-OL, being developed and integrated as an extension for the Knowledge Management and Acquisition platform Semantic Turkey, has inherited all of the facilities that the main application is providing for ontology development, as well as those exposed by the hosting Web Browser (which enabled, for example, to rapidly integrate a web spider into the application and

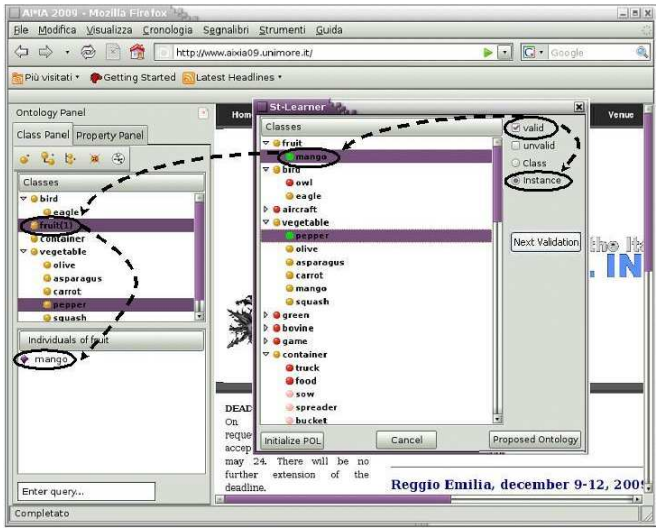

Fig. 2. Manual validation of new resources added to the ontology

use it to provide corpora for learning probabilistic models and/or for inducing new ontology contributions). ST-OL (and Semantic Turkey as its founding technology) has thus proven to be the right environment for embodying this kind of process, providing the crossroads between Users, Web and Knowledge

References

1. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM **38**(11) (November 1995) 39–41
2. Toumouth, A., Lehireche, A., Widdows, D., Malki, M.: Adapting wordnet to the medical domain using lexicosyntactic patterns in the ohsumed corpus. In: AICCSA '06: Proceedings of the IEEE International Conference on Computer Systems and Applications, Washington, DC, USA, IEEE Computer Society (2006) 1029–1036
3. Medche, A.: Ontology Learning for the Semantic Web. Volume 665 of Engineering and Computer Science. Kluwer International (2002)
4. Cimiano, P., Hotho, A., Staab, S.: Learning concept hierarchies from text corpora using formal concept analysis. Journal of Artificial Intelligence research **24** (2005) 305–339
5. Navigli, R., Velardi, P.: Learning domain ontologies from document warehouses and dedicated web sites. Comput. Linguist. **30**(2) (2004) 151–179
6. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 15th International Conference on Computational Linguistics (CoLing-92), Nantes, France (1992)
7. Pantel, P., Pennacchiotti, M.: 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, Sydney, Australia, Association for Computational Linguistics (July 2006) 113–120
8. Harris, Z.: Distributional structure. In Katz, J.J., Fodor, J.A., eds.: The Philosophy of Linguistics, New York, Oxford University Press (1964)
9. Robison, H.R.: Computer-detectable semantic structures. Information Storage and Retrieval **6**(3) (1970) 273–288
10. Pekar, V., Staab, S.: Taxonomy learning: factoring the structure of a taxonomy into a semantic classification decision. Proceedings of the Nineteenth Conference on Computational Linguistics **2** (2002) 786–792
11. Snow, R., Jurafsky, D., Ng, A.Y.: Semantic taxonomy induction from heterogenous evidence. In: In ACL. (2006) 801–808
12. Fallucchi, F., Zanzotto, F.M.: SVD feature selection for probabilistic taxonomy learning. In: Proceedings of the Workshop on Geometrical Models of Natural Language Semantics, Athens, Greece, Association for Computational Linguistics (March 2009) 66–73
13. Cimiano, P., Vlker, J.: Text2onto - a framework for ontology learning and data-driven change discovery. In Montoyo, A., Munoz, R., Metais, E., eds.: Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB). Volume 3513 of Lecture Notes in Computer Science., Alicante, Spain, Springer (June 2005) 227–238
14. Maedche, A., Volz, R.: Icdm workshop on integrating data mining and knowledge management. In: The Text-To-Onto Ontology Extraction and Maintenance Environment, San Jose, California, USA. (2001)
15. Buitelaar, P., Olejnik, D., Sintek, M.: A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. In: 1st European Semantic Web Symposium (ESWS), Heraklion, Greece (May 2004)
16. Gennari, J., Musen, M., Fergerson, R., Grosso, W., Crubzy, M., Eriksson, H.: The evolution of Protégé-2000: An environment for knowledge-based systems development. International Journal of Human-Computer Studies **58**(1) (2003) 89123
17. Gómez-Pérez, A., Manzano-Macho, D.: Deliverable 1.5: A survey of ontology learning methods and techniques. Technical report (May 2003)
18. Haase, P., Lewen, H., Studer, R., Tran, D.T., Erdmann, M., d'Aquin, M., Motta, E.: The neon ontology engineering toolkit. In: In WWW 2008 Developers Track. (April, 2008)
19. Bagni, D., Cappella, M., Pazienza, M.T., Pennacchiotti, M., Stellato, A.: Harvesting relational and structured knowledge for ontology building in the wpro architecture. In Basili, R., Pazienza, M.T., eds.: AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, 10th Congress of the Italian Association for Artificial Intelligence, Rome, Italy, September 10-13, 2007, Proceedings. Lecture Notes in Computer Science. Volume 4733., Springer (2007) 157–169
20. Griesi, D., Pazienza, M.T., Stellato, A.: Semantic turkey - a semantic bookmarking tool (system description). In Franconi, E., Kifer, M., May, W., eds.: 4th European Semantic Web Conference (ESWC 2007). Volume The Semantic Web: Research and Applications, 4519 of Lecture Notes in Computer Science., Innsbruck, Austria, Springer (June 3-7 2007) 779–788 Innsbruck, Austria, June 3-7.
21. Griesi, D., Pazienza, M.T., Stellato, A.: Gobbleing over the web with semantic turkey. In: Semantic Web Applications and Perspectives, 3rd Italian Semantic Web Workshop (SWAP2006), Scuola Normale Superiore, Pisa, Italy, (2006) 18-20 December.