Induction of Terminological Cluster Trees

Preliminaries, Model, Method and Perspectives

Giuseppe Rizzo, Claudia d'Amato,
Nicola Fanizzi, and Floriana Esposito

LACAM – Dipartimento di Informatica
Università degli Studi di Bari "Aldo Moro"
Campus – Via Orabona 4, 70125, Bari, Italy
http://lacam.di.uniba.it

Abstract. In this paper, we tackle the problem of clustering individual resources in the context of the *Web of Data*, that is characterized by a huge amount of data published in a standard data model with a well-defined semantics based on Web ontologies. In fact, clustering methods offer an effective solution to support a lot of complex related activities, such as ontology *construction*, *debugging* and *evolution*, taking into account the inherent incompleteness underlying the representation. Web ontologies already encode a hierarchical organization of the resources by means of the subsumption hierarchy of the classes, which may be expressed explicitly, with proper subsumption axioms, or it must be detected indirectly, by reasoning on the available axioms that define the classes (*classification*). However it frequently happens that such classes are sparsely populated as the hierarchy often reflect a view of the knowledge engineer prior to the actual introduction of assertions involving the individual resources. As a result, very general classes are often loosely populated, but this may happen also to specific subclasses, making it more difficult to check the types of a resource (*instance checking*), even through reasoning services. Among the large number of algorithms proposed in the *Machine Learning* literature, we propose a clustering method that is able to organize groups of resources hierarchically. Specifically, in this work, we introduce a conceptual clustering approach that combines a distance measure between individuals in a knowledge base in a divide-and-conquer solution that is intended to elicit *ex post* the underlying hierarchy based on the actual distributions of the instances.

1 Introduction and Motivation

With the growth of the Web of Data, along with the *Linked Data* initiative [12], a large number of datasets/vocabularies are being published on the ground of a standard data model and connected within a uniform semantic space exploiting RDF and the Web infrastructure. However, this huge quantity of data is also known as inherently uncertain, for being often *inconsistent/conflictual*, *vague* and, especially, *incomplete* [14].

Web ontologies already encode a hierarchical organization of the data by means of the subsumption hierarchy of the classes, which may be expressed explicitly, with proper subsumption axioms, or it has to be determined indirectly, by reasoning on the available axioms defining the various classes (*classification* task). However it frequently happens that such classes are sparsely populated as the hierarchy often reflects a view of the knowledge engineer prior to the actual introduction of assertions involving the individual resources. As a result, very general classes are often loosely populated, but this may happen also to specific subclasses, hindering or, at least, making it difficult to check the types of a resource (instance checking), even employing reasoning services. This may have a strong negative impact on services based on query answering, which often only rely on explicit assertions regarding the data.

Machine Learning methods can be employed to elicit implicit pieces of knowledge from the datasets, also in the face of such cases of inherent incompleteness, and provide *non-standard* inference services [17]. In particular, in this paper we will address *conceptual clustering* [18] as a means to exploit the data to detect further classes that may arise with the underlying hierarchical structure. Specifically, we introduce an approach that combines semantic distance measures over the space of individuals together with a divide-and-conquer solution that is intended to elicit in retrospection an additional class hierarchy to reflect the actual distributions of the instances.

Clustering is an unsupervised learning task aiming at partitioning a collection of objects into subsets or *clusters*, so that those within each cluster are more closely related to one another than to the objects assigned to different clusters [1]. In the Web of Data context, clustering can enable the definition of new emerging concepts (*concept formation*) on the grounds of those employed in a knowledge base; supervised methods can exploit these clusters to induce new concept definitions or to refine existing ones (*ontology evolution*); intensionally defined groupings may speed-up the task of search and discovery; clustering may also induce criteria for ranking the retrieved resources [10, 9].

An object is usually described by a fixed set of feature (*attribute values*) and the most common notion of similarity between the objects is expressed in terms of a distance function based on this set; for example, datasets made up of objects described by tuples of numeric features and (extensions of) the Euclidean distance are adopted to determine object and cluster similarity.

An important difference among the various clustering techniques is related to the type of membership that is adopted. In the simplest (*crisp*) case, e.g. K-MEANS [16], cluster membership *can be exclusive*: each object belongs exactly to one cluster. Extensions, such as FUZZY C-MEANS [3] or EM [7], admit an overlap between the clusters and a degree of membership (*responsibility*) of each object to a cluster. Further extensions include non-flat clustering structures such as those induced via *hierarchical* clustering [1].

In this work, moving on to conceptual clustering, we will require that classes with an intensional definition may account for and define these clusters. To this purpose the algorithms should be able to exploit a background knowledge

that can be expressed using expressive representation languages[1], such as *Description Logics*. Again, in such methods, clustering requires the definition of (dis)similarity measure between the set of the objects to be clustered.

We propose a generalized solution based on *logical trees* [4], namely *Terminological Cluster Trees*, as an extension of terminological decision trees [8]. They adopt a pseudo-metric defined over the space of individuals as a criterion to separate groups of objects rather than *information gain* or other notions of *purity* devised for supervised concept learning and inductive classification methods, terminological decision trees. The proposed solution provides intensional definitions that can be used for describing the individuals in the cluster, and unlike other methods [13, 11], does not have to resort to complex approximations such as the *most specific concept* [2] as the representative of individuals on the conceptual level. Even more so, terminological cluster trees can determine autonomously the number of clusters to be generated, which is a required parameter for several other methods having a strong impact on the quality of the partitions obtained.

The rest of the paper is organized as follows: the next section illustrates the basic notions about the underlying Description Logics foundations for the intended representation and reasoning services; Sect. 3 introduces the problem of clustering individuals of a knowledge base while Sect. 4 presents the approach required for inducing terminological cluster trees. Finally, Sect. 5 illustrates the conclusions and some possible extensions.

2 Basics

In the following, we will borrow notation and terminology from *Description Logics* (DLs) [2] as the representation and reasoning services for the knowledge bases in the Web of Data ultimately rely on such a family of languages. Hence we will use the terms *concept* (*description*) and *role* as synonyms of *class* and *property*[2], respectively.

In these languages, a domain is modeled through atomic *concepts* (classes) N_C and *roles* (relations) N_R, which can be used to build complex descriptions regarding *individuals* (instances, objects), by using specific operators (complement, conjunction and disjunction between concepts) that depend on the adopted language. A *knowledge base* is a couple $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ where the *TBox* $\mathcal{T}$ contains axioms concerning concepts and roles (typically inclusion axioms such as $C \sqsubseteq D$) and the *ABox* $\mathcal{A}$ contains assertions, i.e. axioms regarding the individuals ($C(a)$, resp. $R(a, b)$). The set of individuals occurring in $\mathcal{A}$ is denoted by $\mathsf{Ind}(\mathcal{A})$.

The semantics of individuals, concepts, and roles is defined through interpretations. An *interpretation* is a couple $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is the *domain* of the interpretation and $\cdot^{\mathcal{I}}$ is a *mapping* such that, for each individual a, $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, for each concept C, $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and for each role R, $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The semantics

[1] In the past, various fragments of First-Order Logic have been adopted, such as Clausal Logics, especially in *Inductive Logic Programming*.

[2] Datatype properties, i.e. roles ranging on *concrete domains* will not be considered in this study.

of complex descriptions descends from the interpretation of the primitive concepts/roles and of the operators employed, depending on the adopted language. $\mathcal{I}$ satisfies an axiom $C \sqsubseteq D$ (C is subsumed by D) when $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and an assertion $C(a)$ (resp. $R(a,b)$) when $a^{\mathcal{I}} \in C^{\mathcal{I}}$ (resp. $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$). $\mathcal{I}$ is a *model* for $\mathcal{K}$ iff it satisfies each axiom/assertion α in $\mathcal{K}$, denoted with $\mathcal{I} \models \alpha$. When α is satisfied w.r.t. these models, we write $\mathcal{K} \models \alpha$.

We will be interested in the *instance-checking* inference service: given an individual a and a concept description C determine if $\mathcal{K} \models C(a)$. Due to the *Open World Assumption* (OWA), answering to a class-membership query is more difficult w.r.t. ILP settings where the closed-world reasoning is the standard form. Indeed, one may not be able to prove the truth of either $\mathcal{K} \models C(a)$ or $\mathcal{K} \models \neg C(a)$, as there may be possible to find different interpretations that satisfy either cases.

3 Conceptual Clustering for DL Knowledge Bases

As we are targeting conceptual clustering for DL Knowledge Bases, the problem, in a simple formulation, may be formalized as follows:

Definition 3.1 (conceptual clustering – flat case).

Given:

- *a knowledge base* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$
- *a set of training individuals* $\mathsf{TI} \subseteq \mathsf{Ind}(\mathcal{A})$

Find:

- *a partition of* TI *in* n *pairwise disjoint clusters* $\{\mathbf{C}_1, \ldots, \mathbf{C}_n\}$
- *for each* $i = 1, \ldots, n$, *a concept description* D_i *that accounts for* $\mathbf{C}_i$, *i.e. such that* $\forall a \in \mathbf{C}_i$:
  1. $\mathcal{K} \models D_i(a)$
  2. $\mathcal{K} \models \neg D_j(a) \quad \forall j \in \{1, \ldots, n\}, j \neq i$

Note that in this setting the number of clusters (n) is not required as a parameter. Condition *2.* may be relaxed (e.g. $\mathcal{K} \not\models D_j(a)$) to allow some *overlap* between the clusters/concepts that may be further extended towards probabilistic clustering methods and models [1].

This problem can be regarded as a recursive one, as each cluster, in its turn, might yield its internal partitioning and each sibling sub-cluster would be characterized intensionally by its sub-class.

The decision on whether to partition recursively a given cluster or not generally depends on *cohesion* metrics, assessing a measure of *intra-cluster* similarity (within the cluster) and the w.r.t. the *inter-cluster* dissimilarity (w.r.t. the sibling partitions).

4 Terminological Cluster Trees

The notion of terminological cluster tree extends *logical clustering trees* introduced in [6] and learned through C0.5, a system derived from TILDE [4]. The induction of the model combines elements of logical decision trees induction (i.e. the approach based on recursive partitioning and exploiting of refinement operator for specializing concept descriptions) with other elements of *instance-based learning* (i.e. the employment of a distance measure over the instance space).

Definition 4.1 (Terminological Cluster Trees). *Given a knowledge base $\mathcal{K}$, a* terminological cluster tree *(TCT) is a (binary) logical tree where:*

- *each leaf node stands for a cluster of individuals,* $\mathbf{C}$
- *each node contains a concept description D (over the signature of $\mathcal{K}$);*
- *each edge from an internal node corresponds to the outcome of the membership test of individuals w.r.t. the concept installed in the node*[3]*.*

Hence, a tree-node can be represented by a quadruple $\langle D, \mathbf{C}, T_{left}, T_{right} \rangle$, indicating the two subtrees connected by either edge.

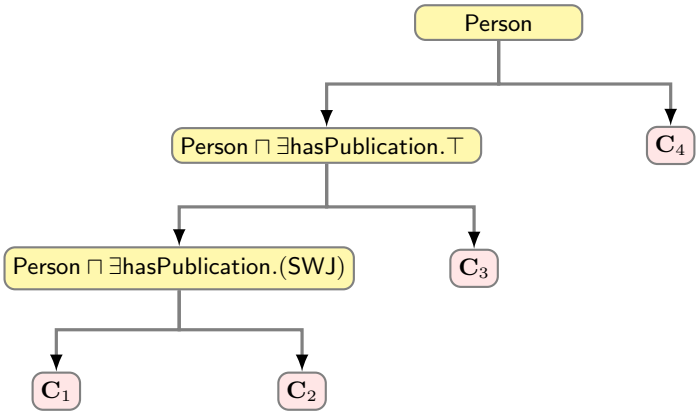

Fig. 1. A simple TCT for describing individuals involved in the Semantic Web research community

Example 4.1 (a TCT). Fig. 1 illustrates a simple example of a TCT for describing individuals in the academic domain. The root node of the tree contains the concept Person. Two edges depart from this node: the left branch is used to denote a positive membership of an individual w.r.t. the concept Person, while

[3] By convention the left branch is for positive instances and the right one is for negative instances.

the right branch denote a negative membership w.r.t. the concept. On one hand, the right child of this node is a leaf that contains a cluster $\mathbf{C}_4$ composed by individuals that are not instances of the concept Person. On the other hand, the left child of the root node contains a further complex concept description. Again, there are two edges departing from this node: the right edge links the node to another leaf containing the cluster $\mathbf{C}_3$ of individuals denoting individuals that have no publication. □

The details of the algorithms for (a) growing a TCT and (b) deriving intensional definitions are reported in the sequel.

4.1 A Method for Inducing Terminological Cluster Trees

A terminological cluster tree T is induced by means of a recursive strategy, which follows the same schema proposed for terminological decision trees (TDTs) [8]. The sketch is reported in Alg. 1.

Algorithm 1 Routines for inducing a TCT

```
1  const ν, δ: thresholds
2
3  function induceTCT(I, C): TCT
4  input I: set of individuals
5        C: concept description
6  begin
7     T ← new TCT
8     if stopCondition(ν, I) then
9        T ← ⟨null, I, null, null⟩
10    else
11       S ← ρ(C)  {set of candidate specializations}
12       E* ← selectBestConcept(S, I)
13       ⟨I_left, I_right⟩ ← split(E*, I)
14       T_left ← induceTCT(I_left, E*)
15       T_right ← induceTCT(I_right, ¬E*)
16       T ← ⟨E*, I, T_left, T_right⟩
17    return T
18 end
19
20 function split(E, I): pair of sets of individuals
21 input E: concept description
22          I: set of individuals
23 begin
24       ⟨P, N⟩ ← retrievePosNeg(E, I, δ)
25       b ← getPrototype(P)
26       c ← getPrototype(N)
27       I_left ← ∅
28       I_right ← ∅
29       for each a ∈ I
30          if d(a, b) ≤ d(a, c) then
31             I_left ← I_left ∪ {a}
32          else
33             I_right ← I_right ∪ {a}
34
35    return ⟨I_left, I_right⟩
36 end
```

The main routine is to be invoked as INDUCETCT(TI, C), where C may be $\top$ or any other general concept the individuals in TI are known to belong to by default.

In this recursive algorithm, the base case depends on a test (via STOPCONDITION) on a threshold over the heuristics employed for growing the tree, measuring the cohesion of the cluster of individuals routed to the node in terms of a given metric. If this value exceeds ν then the branch is marked as completed and the cluster is stored in a leaf node. Further details about the heuristics and the stop condition will be reported later on.

In the inductive step, the current (parent) concept description C has to be specialized by means of an refinement operator (ρ) exploring the search space of specializations of C. A set $\mathbf{S}$ of candidate specializations $\rho(C)$ is obtained. For each $E \in \mathbf{S}$, the set of positive and negative individuals, denoted, resp., by $\mathbf{P}$ and $\mathbf{N}$ are retrieved.

A tricky situation may occur (line 24) when either $\mathbf{N}$ or $\mathbf{P}$ is empty for a given concept (e.g. due to a total lack of disjointness axioms). In such a case, the algorithm can re-assign individuals $\mathbf{I}$ to $\mathbf{N}$ (resp. $\mathbf{P}$) based on the distance between them and the prototype of $\mathbf{P}$ (resp. $\mathbf{N}$) when this goes beyond a given threshold δ.

For $\mathbf{P}$ and $\mathbf{N}$, a representative element is determined as a prototype, i.e. their *medoid*, a central element in a cluster, having a minimal average distance w.r.t. the other elements. Then, function SELECTBESTCONCEPT evaluates the specializations of the concept according to the closeness w.r.t. the medoids, determined according to a given distance measure (discussed later). The best concept $E^* \in \mathbf{S}$ is the one for which the distance between the medoids for positive and negative instance set is maximized. Then E^* is installed in the current node.

After the assessment of the best concept E^*, the individuals are partitioned by SPLIT to be routed along the left or right branch. Differently from TDTs, the routine does not decide the branch where the individuals will be sorted according to a concept membership test (instance check): rather it decides to split individuals according to the distance w.r.t. the prototypes of positive and negative membership w.r.t. E^*, i.e. the medoids of $\mathbf{P}$ and $\mathbf{N}$. This divide-and-conquer strategy is applied recursively until the instances routed to a node satisfy the stop condition. Note that, the number of the clusters is not required as an input but it depends on the number of paths generated in the growing phase: the algorithm is able to determine it naturally following the data distribution.

Refinement Operator The proposed approach relies on a downward refinement operator that can generate the concepts to be installed in child-nodes performing a specialization process on the concept, say C, installed in a parent-node:

ρ_1 by adding a concept atom (or its complement) as a conjunct: $C' = C \sqcap (\neg)A$;
ρ_2 by adding a general existential restriction (or its complement) as a conjunct: $C' = C \sqcap (\neg)(\exists)R.\top$;

ρ_3 by adding a general universal restriction (or its complement) as a conjunct: $C' = C \sqcap (\neg)(\forall)R.\top$;

ρ_4 by replacing a sub-description C_i in the scope of an existential restriction in C with one of its refinements: $\exists R.C'_i \in \rho(\exists R.C_i) \wedge C'_i \in \rho(C_i)$;

ρ_5 by replacing a sub-description C_i in the scope of a universal restriction with one of its refinements: $\forall R.C'_i \in \rho(\forall R.C_i) \wedge C'_i \in \rho(C_i)$.

Note that the cases of ρ_4 and ρ_5 are recursive.

Prototypes Despite the common schema of the algorithms employed for growing TCTs and TDTs, the latter are obtained by selecting the best test in terms of information gain maximization [8], while the clustering procedure for TCTs resorts to a distance-based criterion on the individuals in the knowledge base. Specifically, the heuristic adopted for selecting the best concept description that will be installed as new node can be defined as follows:

$$E^* = \arg\max_{D \in \rho(C)} d\left(p(\mathbf{P}), p(\mathbf{N})\right)$$

where $\mathbf{P}$ and $\mathbf{N}$ are sub-clusters obtained from splitting $\mathbf{I}$ w.r.t. D, $d(\cdot,\cdot)$ is a distance measure between individuals and $p(\cdot)$ is a function which maps a set of individuals to its prototype. As previously mentioned, the algorithm computes the *medoids* of both the set of positive and negative instances w.r.t. the test.

Distance Measure The computation of medoids requires a (possibly) language-independent measure for individuals whose definition should capture aspects of their semantics in the context of the knowledge base. However individuals don't have an algebraic structure that can be exploited directly. In the TCT induction algorithm a language-independent dissimilarity measure [5] has been adopted.

Given a knowledge base $\mathcal{K}$, the idea is to use the behavior of an individual w.r.t. a set of concepts $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ that is dubbed *context* or *committee* of features. After $\mathcal{C}$ has been chosen, a *projection function* for each $C_i \in \mathcal{C}$ can be defined as a mapping $\pi_i : \mathsf{Ind}(\mathcal{A}) \to \{0, \frac{1}{2}, 1\}$ such that

$$\forall\, a \in \mathsf{Ind}(\mathcal{A}) \qquad \pi_i(a) = \begin{cases} 1 & \text{if } \mathcal{K} \models C_i(a) \\ 0 & \text{if } \mathcal{K} \models \neg C_i(a) \\ \frac{1}{2} & \text{otherwise} \end{cases}$$

For the value of π_i associated to the uncertain membership case (owing to the OWA) we adopted a uniform prior probability. It can be set more accurately according to the membership to C_i, $\pi_i : \mathsf{Ind}(\mathcal{A}) \to [0,1]$, with $x \in \mathsf{Ind}(\mathcal{A}) \mapsto \pi_i(x) = \mathbb{P}[\mathcal{K} \models C_i(x)]$, if such a value can be estimated. For example, for densely populated ontologies this may be estimated as $|r_{\mathcal{A}}(C_i)|/|\mathsf{Ind}(\mathcal{A})|$, where $r_{\mathcal{A}}()$ indicates the *retrieval* of the argument concept w.r.t. $\mathcal{A}$, i.e. the set of individuals in that are known to belong to the concept [2]: $r_{\mathcal{A}}(C) = \{a \in \mathsf{Ind}(\mathcal{A}) \mid \mathcal{K} \models C(a)\}$. For largely populated concept C, this may be estimated on the ground of the assertions contained in $\mathcal{A}$ avoiding the bottleneck of reasoning.

Through the projection function, it is possible to define a family of distance measures $\{d_p^{\mathcal{C}}\}_{p\in\mathbb{N}}$ for individuals as follows: $d_p^{\mathcal{C}} : \mathsf{Ind}(\mathcal{A}) \times \mathsf{Ind}(\mathcal{A}) \to [0,1]$ such that

$$d_p^{\mathcal{C}}(a,b) = \sqrt[p]{\sum_{i=1}^{m} w_i \left[1 - \pi_i(a)\pi_i(b)\right]^p}$$

In previous papers, e.g. [9], we also used the form: $\sqrt[p]{\sum_i w_i |[\pi_i(a) - \pi_i(b)|^p}$. The vector of weights $\boldsymbol{w}$ can be set according to the entropy of the considered concepts computed over the individuals occurring in $\mathcal{K}$ [10].

To speed up the algorithm, the projection functions can be pre-computed (once) before the learning phase.

Stop Condition The growth of a TCT can be stopped by resorting to a heuristic that is similar to the one employed for selecting the best concept description. This can be made by introducing a threshold $\nu \in [0,1]$, for the value of $d(\cdot,\cdot)$. If the value is lower than the threshold, the branch growth is stopped.

4.2 Extracting Concepts from TCTs

Alg. 2 reports the sketch of the function for deriving the concept descriptions describing the clusters obtained through a TCT. Given a TCT T, essentially

Algorithm 2 Routines for deriving concept definitions from a TCT

```
1  function DeriveConcepts(C, T): set of concepts
2  input C: concept name
3        T: TCT
4  begin
5     let T = ⟨D, I, T_left, T_right⟩
6     if T_left = T_right = null then    { leaf }
7        return {C}
8     else
9        CS_left ← DeriveConcepts(C ⊓ D, T_left)
10       CS_right ← DeriveConcepts(¬C ⊓ ¬D, T_right)
11       return (CS_left ∪ CS_right)
12 end
```

function DERIVECONCEPTS traverses the tree structure to collect the concept descriptions that are used as parents of the leaf-nodes. In this phase, it generates a set of concept descriptions **CS**.

Example 4.2. The set **CS** that can be obtained from the tree reported in Fig. 1 contains the following concept descriptions (corresponding to the four clusters):

$$\begin{aligned}\mathbf{CS} = \{\ D_1, D_2, D_3, D_4\ \} = \\ \{\ &\mathsf{Person} \sqcap \exists \mathsf{hasPublication}.\top \sqcap \exists \mathsf{hasPublication}.(\mathsf{SWJ}),\\ &(\mathsf{Person} \sqcap \exists \mathsf{hasPublication}.\top) \sqcap \neg(\mathsf{Person} \sqcap \exists \mathsf{hasPublication}.(\mathsf{SWJ})),\\ &\mathsf{Person} \sqcap \neg(\mathsf{Person} \sqcap \exists \mathsf{hasPublication}.\top),\\ &\neg\mathsf{Person}\ \}\end{aligned}$$

Such concepts might be also simplified before being output. □

Note that also the internal nodes contain concept descriptions that account for the individuals routed to such nodes. Hence it is straightforward to extend the extraction procedure, in order to produce a list of subsumption axioms between couples of the node concepts which might be submitted to a knowledge engineer for validation.

5 Conclusions, Ongoing and Future Work

In this work, we have proposed an extension of terminological decision trees [8], which were originally employed for (supervised) concept learning, in order to solve (unsupervised) conceptual clustering problems in the context of the datasets belonging to the Web of Data.

The algorithm essentially adopts a divide-and-conquer strategy that generates concept descriptions to be installed in the inner nodes via a refinement operator and selects the most promising ones to represent clusters of similar individuals using a suitable distance measure.

This preliminary work can be extended along various possible directions (some extensions are already being carried out), which can be listed as follows:

- *a comparison to other clustering methods* in order to understand the feasibility of the proposed solution;
- *new distance measures between the individuals*: in this work we adopted a language-independent distance measure between the individuals of a knowledge base. In this perspective, it may be interesting to investigate other distance measures (e.g. less computationally expensive functions);
- *new refinement operators*: we adopted a refinement operator used for solving supervised learning problems. Further extensions of this work may consider new refinement operators that can be borrowed from other machine learning algorithms, e.g. DL-Learner [15] or other methods devised for the specific method;
- *new heuristics*: the approach installs concept descriptions so that the overlap between the corresponding sets of positive and negative individuals is minimized. It may be interesting to investigate a different heuristic that allows to quantify the degree of overlap between the two set of individuals;
- *discovery of disjointness axioms* in order to enrich a knowledge base through the induction of terminological cluster trees and evaluation of the approach;
- *strategies to obtain simpler trees*: we plan to investigate the effectiveness of post-pruning procedures
- *inducing different kind of clusters*: we focused on crisp clustering in this work. Another possible extension concerns the integration of theories for uncertainty management such as probabilistic models or the *Dempster-Shafer theory* allowing overlapping clusters;
- *scalability*: another extension is to adopt solutions to make the proposed method scalable. Some possible solutions span from the implementation of

distributed version of TCT induction algorithm and the employment of approximate reasoners in order to cope with the computational costs related to the underlying reasoning services adopted by the algorithm.

References

1. Aggarwal, C.C., Reddy, C.K.: Data Clustering: Algorithms and Applications. Chapman & Hall/CRC, 1st edn. (2013)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press, 2nd edn. (2007)
3. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Kluwer Academic Publishers (1981)
4. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artif. Intell. 101(1-2), 285–297 (1998)
5. d'Amato, C., Fanizzi, N., Esposito, F.: Query Answering and Ontology Population: An Inductive Approach. In: Bechhofer, S., et al. (eds.) Proceedings of ESWC 2008. LNCS, vol. 5021, pp. 288–302. Springer (2008)
6. De Raedt, L., Blockeel, H.: Using logical decision trees for clustering. In: Lavrač, N., Džeroski, S. (eds.) Proceedings of ILP 1997. LNAI, vol. 1297, pp. 133–140. Springer (1997)
7. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B 39(1), 1–38 (1977)
8. Fanizzi, N., d'Amato, C., Esposito, F.: Induction of Concepts in Web Ontologies through Terminological Decision Trees. In: Balcázar, J., et al. (eds.) Proceedings of ECML/PKDD2010. LNAI, vol. 6321, pp. 442–457. Springer (2010)
9. Fanizzi, N., d'Amato, C.: A Hierarchical Clustering Method for Semantic Knowledge Bases. In: Apolloni, B., Howlett, R.J., Jain, L.C. (eds.) Proceedings of KES 2007, Part III. LNCS, vol. 4694, pp. 653–660. Springer (2007)
10. Fanizzi, N., d'Amato, C., Esposito, F.: Evolutionary Conceptual Clustering Based on Induced Pseudo-Metrics. Int. J. Semantic Web Inf. Syst. 4(3), 44–67 (2008)
11. Fanizzi, N., Iannone, L., Palmisano, I., Semeraro, G.: Concept Formation in Expressive Description Logics. In: Boulicaut, J., et al. (eds.) Proceedings of ECML 2004. LNAI, vol. 3201, pp. 99–110. Springer (2004)
12. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web, Morgan & Claypool Publishers (2011)
13. Kietz, J.U., Morik, K.: A Polynomial Approach to the Constructive Induction of Structural Knowledge. Mach Learn 14, 193–217 (1994)
14. Laskey, K., Costa, P., Kokar, M., Martin, T., Lukasiewicz, T.: Uncertainty Reasoning for the World Wide Web. Tech. rep., URW3 W3C Incubator Group (2008), `http://www.w3.org/2005/Incubator/urw3/XGR-urw3-20080331`
15. Lehmann, J.: DL-Learner: Learning Concepts in Description Logics. Journal of Machine Learning Research (JMLR) 10, 2639–2642 (2009)
16. MacQueen, J.B.: Some Methods for Classification and Analysis of MultiVariate Observations. In: Le Cam, L.M., Neyman, J. (eds.) Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability. vol. 1, pp. 281–297. University of California Press (1967)
17. Rettinger, A., Lösch, U., Tresp, V., d'Amato, C., Fanizzi, N.: Mining the Semantic Web - Statistical learning for next generation knowledge bases. Data Min. Knowl. Discov. 24(3), 613–662 (2012)

18. Stepp, R.E., Michalski, R.S.: Conceptual Clustering: Inventing Goal Oriented Classifications of Structured Objects. In: Machine Learning: An Artificial Intelligence Approach, Vol II. Morgan Kaufmann (1986)