Evolutionary Clustering in Description Logics: Controlling Concept Formation and Drift in Ontologies

Nicola Fanizzi, Claudia d'Amato, and Floriana Esposito

Dipartimento di Informatica – Università degli Studi di Bari
Campus Universitario, Via Orabona 4, 70125 Bari, Italy
{fanizzi|claudia.damato|esposito}@di.uniba.it

Abstract. We present a method based on clustering techniques to detect concept drift or novelty in a knowledge base expressed in Description Logics. The method exploits an effective and language-independent semi-distance measure defined for the space of individuals, that is based on a finite number of dimensions corresponding to a committee of discriminating features (represented by concept descriptions). In the algorithm, the possible clusterings are represented as strings of central elements (medoids, w.r.t. the given metric) of variable length. The number of clusters is not required as a parameter; the method is able to find an optimal choice by means of the evolutionary operators and of a fitness function. An experimentation with some ontologies proves the feasibility of our method and its effectiveness in terms of clustering validity indices. Then, with a supervised learning phase, each cluster can be assigned with a refined or newly constructed intensional definition expressed in the adopted language.

1 Introduction

In the context of the Semantic Web (henceforth SW) there is an extreme need of automatizing those activities which are more burdensome for the knowledge engineer, such as ontology construction, matching and evolution. These phases can be assisted by specific learning methods, such as instance-based learning (and analogical reasoning) [5], case-based reasoning [7], inductive generalization [8, 21, 15] and unsupervised learning (clustering) [19, 12] crafted for knowledge bases (henceforth KBs) expressed in the standard representations of the field and complying with their semantics.

In this work, we investigate on the problem of conceptual clustering of semantically annotated resources. The benefits of *conceptual clustering* [24] in the SW context are manifold. Clustering annotated resources enables the definition of new emerging concepts (*concept formation*) on the grounds of the concepts defined in a KB; 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* [6]; a clustering may also suggest criteria for *ranking* the retrieved resources based on the distance from the centers. Approaches based on incremental learning [9] and clustering have also been proposed [23] to detect *novelties* or track the phenomenon of *concept drift* [25] over time. Most of the clustering methods are based on the application of similarity (or density) measures defined over a fixed set of attributes of the domain objects [16]. Classes of objects are taken as collections that exhibit low interclass similarity (density) and high intraclass similarity (density). These methods are rarely able to take into account some form of *background knowledge* that could characterize object configurations by means of global concepts and semantic relationships [24]. This hinders the interpretation of the outcomes of these methods that is crucial in the SW perspective which enforces sharing and reusing the produced knowledge to enable semantic interoperability across different KBs and applications.

Conceptual clustering methods can answer these requirements since they have been specifically crafted for defining groups of objects through descriptions based on selected attributes [24]. The expressiveness of the language adopted for describing objects and clusters is extremely important. Related approaches, specifically designed for terminological representations (*Description Logics* [1], henceforth DLs), have recently been introduced [19, 12]. They pursue logic-based methods for attacking the problem of clustering w.r.t. some specific DLs. The main drawback of these methods is that they are language-dependent and cannot scale to standard SW representations that are mapped on complex DLs. Moreover, purely logic methods can hardly handle noisy data.

These problems motivate the investigation on similarity-based clustering methods which can be more noise-tolerant and language-independent. In this paper, an extension of distance-based techniques is proposed. It can cope with the standard SW representations and profit by the benefits of a randomized search for optimal clusterings. The method is intended for grouping similar resources w.r.t. a notion of similarity, coded in a distance measure, which fully complies with the semantics KBs expressed in DLs. The individuals are gathered around cluster centers according to their distance. The choice of the best centers (and their number) is performed through an evolutionary approach [13, 20]. From a technical viewpoint, upgrading existing distance-based algorithms to work on multi-relational representations, like the concept languages used in the SW, requires similarity measures that are suitable for such representations and their semantics. A theoretical problem is posed by the *Open World Assumption* (OWA) that is generally made on the language semantics, differently from the *Closed World Assumption* (CWA) which is standard in other contexts. Moreover, as pointed out in a seminal paper on similarity measures for DLs [3], most of the existing measures focus on the similarity of atomic concepts within hierarchies or simple ontologies. Recently, dissimilarity measures have been proposed for some specific DLs [5]. Although they turned out to be quite effective for specific inductive tasks, they were still partly based on structural criteria which makes them fail to fully grasp the underlying semantics and hardly scale to more complex ontology languages. We have devised a family of dissimilarity measures for semantically annotated resources, which can overcome the aforementioned limitations [10]. Following the criterion of semantic discernibility of individuals, a family of measures is derived that is suitable for a wide range of languages since it is merely based on the discernibility of the input individuals w.r.t. a fixed committee of features represented by a set of concept definitions. In this setting, instead of the notion of *centroid* that characterizes the distance-based algorithms descending from K-MEANS [16], originally developed for numeric or ordinal features, we recur to the notion of *medoids* [18]. The proposed clustering algorithm employs genetic programming as a search schema. The evolutionary problem is modeled by considering populations made up of strings of medoids with different lengths. The medoids are computed according to the semantic measure mentioned above. On each generation, the strings in the current population are evolved by mutation and cross-over operators, which are also able to change the number of medoids. Thus, this algorithm is also able to suggest autonomously a promising number of clusters. Accordingly, the fitness function is based both on the optimization of a cluster cohesion index and on the penalization of lengthy medoid strings.

We propose the exploitation of the outcomes of the clustering algorithm for detecting the phenomena of concept drift or novelty from the data in the KB. Indeed ontologies evolve over the time (because new assertions are added or because new concepts are defined). Specifically, the occurrence of new assertions can provoke the introduction of new concepts (defined only by the extensions) or can transform existing concepts into more general or more specific ones. We consider the set of new assertions as a candidate cluster and we evaluate its nature w.r.t. the computed clustering model; namely we assess if the candidate cluster is a *normal* cluster, a *new* concept or a *drift* concept. Hence, new concepts could be induced and/or existing ones could be refined.

The remainder of the paper is organized as follows. Sect. 2 presents the basics of the target representation and the semantic similarity measure adopted with the clustering algorithm which is presented in Sect. 3. In Sect. 4 we report an experiment aimed at assessing the validity of the method on some ontologies available in the Web. The utility of clustering in the logic of ontology evolution is discussed in Sect. 5. Conclusions and extensions of the work are examined in Sect. 6.

2 Semantic Distance Measures

In the following, we assume that resources, concepts and their relationship may be defined in terms of a generic ontology language that may be mapped to some DL language with the standard model-theoretic semantics (see the DLs handbook [1] for a thorough reference). In the intended framework setting, a *knowledge base* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ contains a *TBox* $\mathcal{T}$ and an *ABox* $\mathcal{A}$. $\mathcal{T}$ is a set of concept definitions. The complexity of such definitions depends on the specific DL language constructors. $\mathcal{A}$ contains *assertions* (ground facts) on *individuals* (domain objects) concerning the current world state, namely: *class-membership* $C(a)$ which means that a is an instance of concept C; *relations* $R(a, b)$ which means that a is R-related to b. The set of the individuals referenced in the assertions ABox $\mathcal{A}$ will be denoted with $\mathsf{Ind}(\mathcal{A})$. The *unique names assumption* can be made on the ABox individuals[1] therein.

As regards the required inference services, the measure requires performing *instance-checking*, which amounts to determine whether an individual, say a, belongs to a concept extension, i.e. whether $C(a)$ holds for a certain concept C. Note that, differently from the standard DB settings, due to the OWA, the reasoner might be unable to provide a definite answer. Hence one has to cope with this form of uncertainty.

Following some techniques for distance induction in clausal spaces developed in ILP [22], we propose the definition of totally semantic distance measures for individuals in the context of a KB which is also able to cope with the OWA. The rationale of the new measure is to compare individuals on the grounds of their behavior w.r.t. a given set of

[1] Each individual can be assumed to be identified by its own URI, however this is not bound to be a one-to-one mapping.

features, that is a collection of concept descriptions, say $\mathsf{F} = \{F_1, F_2, \ldots, F_m\}$, which stands as a group of discriminating *features* expressed in the considered DL language. A family of dissimilarity measures for individuals inspired to the Minkowski's distances (L_p) can be defined as follows [10]:

Definition 2.1 (family of dissimilarity measures). *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a knowledge base. Given set of concept descriptions $\mathsf{F} = \{F_1, F_2, \ldots, F_m\}$, a family of functions $\{d_p^{\mathsf{F}}\}_{p \in \mathbb{N}}$ with $d_p^{\mathsf{F}} : \mathsf{Ind}(\mathcal{A}) \times \mathsf{Ind}(\mathcal{A}) \mapsto [0, 1]$ is defined as follows: $\forall a, b \in \mathsf{Ind}(\mathcal{A})$*

$$d_p^{\mathsf{F}}(a, b) := \frac{L_p(\pi(a), \pi(b))}{m} = \frac{1}{m} \left(\sum_{i=1}^{m} \mid \pi_i(a) - \pi_i(b) \mid^p \right)^{\frac{1}{p}}$$

where $p > 0$ and $\forall a \in \mathsf{Ind}(\mathcal{A})$ the projection function *π_i is defined by:*

$$\pi_i(a) = \begin{cases} 1 & \mathcal{K} \models F_i(a) \\ 0 & \mathcal{K} \models \neg F_i(a) \\ 1/2 & otherwise \end{cases}$$

The superscript F will be omitted when the set of features is fixed. The functions $\{d_p^{\mathsf{F}}\}_{p \in \mathbb{N}}$ are semi-distance measures (see [10] for more details). The case $\pi_i(a) = 1/2$ occurs when a reasoner cannot give the truth value for a certain membership query. This is due to the OWA normally made in this context. Differently from other DLs measures [4, 17], the presented measure is able to measure dissimilarity between individuals and moreover it does not depend on the constructors of a specific language. It requires only the instance-checking service that is used for deciding whether an individual that is asserted in the KB belongs to a concept extension. Such information can be also pre-computed in order to speed-up the computation of the dissimilarity values and consequently also the clustering process. The underlying idea in the measure definition is that similar individuals should exhibit the same behavior w.r.t. the concepts in F. Here, we make the assumption that the feature-set F represents a sufficient number of (possibly redundant) features that are able to discriminate really different individuals. Preliminary experiments, where the measure has been exploited for instance-based classification (*Nearest Neighbor* algorithm) and similarity search [26], demonstrated the effectiveness of the measure using the very set of both primitive and defined concepts found in the KBs.

However, the choice of the concepts to be included in the committee F is crucial and may be the object of a preliminary learning problem to be solved (*feature selection for metric learning*). We have devised specific optimization algorithms [11] founded in *genetic programming* and *simulated annealing* (whose presentation goes beyond the scope of this work) which are able to find optimal choices of discriminating concept committees. Differently from the goal of the this paper, in [11], the problem of managing novelties and concept drift in an ontology has not been considered. Since the measure is very dependent on the concepts included in the committee of features F, two immediate heuristics can be derived: 1) control the number of concepts of the committee, including especially those that are endowed with a real discriminating power; 2) finding optimal sets of discriminating features, by allowing also their composition employing the specific constructors made available by the DL of choice.

3 Evolutionary Clustering Procedure

Many similarity-based clustering algorithms [16] can be applied to semantically annotated resources stored in a KB, exploiting the measures discussed in the previous section even if, for the best of our knowledge, very few (conceptual) clustering algorithms for coping with DL representations have been proposed in the literature. We focussed on the techniques based on evolutionary methods which are able to determine also an optimal number of clusters, instead of requiring it as a parameter (although the algorithm can be easily modified to exploit this information that greatly reduces the search-space). Conceptual clustering requires also to provide a definition for the detected groups, which may be the basis for the formation of new concepts inductively elicited from the KB. Hence, the conceptual clustering procedure consists of two phases: one that detects the clusters in the data and the other that finds an intensional definition for the groups of individuals detected in the former phase. The first phase of the clustering process is presented in this section. The concept formation process is presented in Sect. 5.2.

The first clustering phase implements a genetic programming learning scheme, where the designed representation for the competing genomes is made up of strings (lists) of individuals of different lengths, with each gene standing as prototypical for a cluster. Specifically, each cluster will be represented by its prototype recurring to the notion of *medoid* [18, 16] on a categorical feature-space w.r.t. the distance measure previously defined. Namely, the medoid of a group of individuals is the individual that has the minimal distance w.r.t. the others. Formally. in this setting:

Definition 3.1 (medoid). *Given a cluster of individuals $C = \{a_1, a_2, \ldots, a_n\} \subseteq \mathsf{Ind}(\mathcal{A})$, the* medoid *of the cluster is defined:*

$$\mathrm{medoid}(C) := \operatorname*{argmin}_{a \in C} \sum_{j=1}^{n} d(a, a_j)$$

In the proposed evolutionary algorithm, the population will be made up of genomes represented by a list of medoids $G = \{m_1, \ldots, m_k\}$ of variable lengths. The algorithm performs a search in the space of possible clusterings of the individuals, optimizing a fitness measure that maximizes the discernibility of the individuals of the different clusters (inter-cluster separation) and the intra-cluster similarity measured in terms of the d_p^{F} pseudo-metric. On each generation those strings that are considered as best w.r.t. a fitness function are selected for passing to the next generation. Note that the algorithm does not prescribe a fixed length of the genomes (as, for instance in K-MEANS and its extensions [16]), hence it searches a larger space aiming at determining an optimal number of clusters for the data at hand. In the following, a sketch of the algorithm, named ECM, *Evolutionary Clustering around Medoids* is reported.

```
medoidVector ECM(maxGenerations)
input:  maxGenerations: max number of iterations;
output: medoidVector: list of medoids
static: offsprings: vector of generated offsprings
            fitnessVector: ordered vector of fitness values
            generationNo: generation number
```

```
INITIALIZE(currentPopulation,popLength)
generationNo = 0
while (generationNo < maxGenerations)
        begin
        offsprings = GENERATEOFFSPRINGS(currentPopulation)
        fitnessVector = COMPUTEFITNESS(offsprings)
        currentPopulation = SELECT(offsprings,fitnessVector)
        ++generationNo
        end
return currentPopulation[0] // fittest genome
```

After the call to the INITIALIZE() function returning (to currentPopulation) a randomly generated initial population of popLength medoid strings, the algorithm essentially consists of the typical generation loop of genetic programming, where a new population is computed and then evaluated for deciding on the best genomes to be selected for survival to the next generation. On each iteration, new offsprings of current best clusterings in currentPopulation are computed. This is performed by suitable genetic operators explained in the following. The fitnessVector recording the quality of the various offsprings (i.e. clusterings) is then updated, and then the best offsprings are selected for the next generation. The fitness of a single genome $G = \{m_1, \ldots, m_k\}$ is computed by distributing all individuals among the clusters ideally formed around the medoids in that genome. For each medoid m_i ($i = 1, \ldots, k$), let C_i be such a cluster. Then, the fitness is computed by the function:

$$\textsc{fitness}(G) = \left(\lambda(k) \sum_{i=1}^{k} \sum_{x \in C_i} d_p(x, m_i) \right)^{-1}$$

The factor $\lambda(k)$ is introduced to penalize those clusterings made up of too many clusters that could enforce the minimization in this way (e.g. by proliferating singletons). A suggested value is $\lambda(k) = \sqrt{k+1}$ which was used in the experiments (see Sect. 4).

The loop condition is controlled by the maximal number of generation (the maxGenerations parameter) ensuring that eventually it may end even with a suboptimal solution to the problem. Besides other parameters can be introduced for controlling the loop based on the best fitness value obtained so far or on the gap between the fitness of best and of the worst selected genomes in currentPopulation. Eventually, the best genome of the vector (supposed to be sorted by fitness in descending order) is returned.

It remains to specify the nature of the GENERATEOFFSPRINGS procedure and the number of such offsprings, which may as well be another parameter of the ECM algorithm. Three mutation and one crossover operators are implemented:

DELETION(G) drop a randomly selected medoid: $G := G \setminus \{m\}, m \in G$

INSERTION(G) select $m \in \mathsf{Ind}(\mathcal{A}) \setminus G$ that is added to G: $G := G \cup \{m\}$

REPLACEMENTWITHNEIGHBOR(G) randomly select $m \in G$ and replace it with $m' \in \mathsf{Ind}(\mathcal{A}) \setminus G$ s.t. $\forall m'' \in \mathsf{Ind}(\mathcal{A}) \setminus G \; d(m, m') \leq d(m, m'')$: $G' := (G \setminus \{m\}) \cup \{m'\}$

CROSSOVER(G_A, G_B) select subsets $S_A \subset G_A$ and $S_B \subset G_B$ and exchange them between the genomes: $G_A := (G_A \setminus S_A) \cup S_B$ and $G_B := (G_B \setminus S_B) \cup S_A$

The representation of centers by means of medoids has two advantages. First, it presents no limitations on attributes types, and, second, the choice of medoids is dictated by the location of a predominant fraction of points inside a cluster and, therefore, it is less sensitive to the presence of outliers. In K-MEANS case a cluster is represented by its centroid, which is a mean (usually weighted average) of points within a cluster. This works conveniently only with numerical attributes and can be negatively affected even by a single outlier.

A (10+60) selection strategy has been implemented, with the numbers indicating, resp., the number of parents selected for survival and the number of their offsprings generated employing the mutation operators presented above.

4 Evaluation

The feasibility of the clustering algorithm has been evaluated with an experimentation on KBs selected from standard repositories. For testing our algorithm we preferred using populated ontologies (which may be more difficult to find) rather than randomly generating assertions for artificial individuals, which might have biased the procedure.

4.1 Experimental Setup

A number of different OWL ontologies, selected from various sources[2], have been considered for the experimentation: FSM, SURFACEWATERMODEL, TRANSPORTATION, NEWTESTAMENTNAMES, and FINANCIAL. Table 1 summarizes details concerning such ontologies. Of course, the number of individuals gives only a partial indication of the number of assertions concerning them which affects both the complexity of reasoning and distance assessment.

Table 1. Ontologies employed in the experiments.

Ontology	DL lang.	#concepts	#obj.prop.	#data prop.	#individuals
FSM	$\mathcal{SOF}(D)$	20	10	7	37
SURFACEWATERMODEL	$\mathcal{ALCOF}(D)$	19	9	1	115
TRANSPORTATION	$\mathcal{ALC}$	44	7	0	331
NEWTESTAMENTNAMES	$\mathcal{SHIF}(D)$	47	27	8	676
FINANCIAL	$\mathcal{ALCIF}$	60	16	0	1000

In the computation of the distances between individuals all concepts in the KB have been used for the committee of features, thus guaranteeing meaningful measures with high redundancy. The PELLET reasoner[3] was employed to perform the instance-checking that were necessary to compute the projections.

[2] See the Protégé library: `http://protege.stanford.edu/plugins/owl/owl-library` and the website: `http://www.cs.put.poznan.pl/alawrynowicz/financial.owl`

[3] `http://pellet.owldl.com`

The experimentation consisted of 10 runs of the algorithm per knowledge base. The indexes which were chosen for the experimental evaluation were: the *generalized* R-Squared (modRS), the *generalized* Dunn's index, the average Silhouette index, and the number of clusters obtained. We will consider a generic partition $P = \{C_1, \ldots, C_k\}$ of n individuals in k clusters. The indexes are formally defined as follows.

The R-Squared index [14] is a measure of cluster separation, ranging in [0,1]. Instead of the cluster means, we generalize the measure by computing it w.r.t. their medoids, namely:

$$RS(P) := \frac{SS_b(P)}{SS_b(P) + SS_w(P)}$$

where SS_b is the *between clusters Sum of Squares* defined as $SS_b(P) := \sum_{i=1}^{k} d(\overline{m}, m_i)^2$ where $\overline{m}$ is the medoid of the whole dataset and SS_t is the *within cluster Sum of Squares* that is defined as $SS_w(P) := \sum_{i=1}^{k} \sum_{a \in C_i} d(a, m_i)^2$

The generalized Dunn's index is a measure of both compactness (within clusters) and separation (between clusters). The original measure is defined for numerical feature vectors in terms of centroids and it is known to suffer from the presence of outliers. To overcome these limitations, we adopt a generalization of Dunn's index [2] that is modified to deal with medoids. The new index can be defined:

$$V_{GD}(P) := \min_{1 \leq i \leq k} \left\{ \min_{\substack{1 \leq j \leq k \\ i \neq j}} \left\{ \frac{\delta_p(C_i, C_j)}{\max_{1 \leq h \leq k} \{\Delta_p(C_h)\}} \right\} \right\}$$

where δ_p is the Hausdorff distance for clusters derived[4] from d_p, while the cluster diameter measure Δ_p is defined as $\Delta_p(C_h) := \frac{2}{|C_h|} \sum_{c \in C_h} d_p(c, m_h)$. It is more noise-tolerant w.r.t. the original measure. It ranges in $[0, +\infty[$ and has to be maximized.

The average Silhouette index [18] is a measure ranging in the interval [-1,1], thus suggesting an absolute best value for the validity of a clustering. For each individual x_i, $i \in \{1, \ldots, n\}$, the average distance to other individuals within the same cluster C_j, $j \in \{1, \ldots, k\}$, is computed: $a_i := \frac{1}{|C_j|} \sum_{x \in C_j} d_p(a_i, x)$ Then the average distance to the individuals in other clusters is also computed: $b_i := \frac{1}{|C_j|} \sum_{x \in C_h}^{h \neq j} d_p(a_i, x)$ Hence, the Silhouette value for the considered individual is obtained as follows:

$$s_i := \frac{(b_i - a_i)}{\max(a_i, b_i)}$$

The average Silhouette value s for the whole clustering is computed: $s := \frac{1}{k} \sum_{1=1}^{k} s_i$

We also considered the average number of clusters resulting from the repetitions of the experiments on each KB. A stable algorithm should return almost the same number of clusters on each repetition. It is also interesting to compare this number to the one of the primitive and defined concepts in each ontology (see Tab. 1), although this is not a hierarchical clustering method.

[4] δ_p is defined $\delta_p(C_i, C_j) := \max\{d_p(C_i, C_j), d_p(C_j, C_i)\}$, where $d_p(C_i, C_j) := \max_{a \in C_i}\{\min_{b \in C_j}\{d_p(a, b)\}\}$.

4.2 Results

The experiment consisted in 10 runs of the evolutionary clustering procedure with an optimized feature set (computed in advance). Each run took from a few minutes to 41 mins on a 2.5GhZ (512Mb RAM) machine. These timings include the pre-processing phase needed to compute the distance values between all couples of individuals. The elapsed time for the core clustering algorithm is actually very short (max 3 minutes). The outcomes of the experiments are reported in Tab. 2. For each KB and for each index, the average values observed along the various repetitions is considered. The standard deviation and the range of minimum and maximum values are also reported.

Table 2. Results of the experiments: for each index, average value (±standard deviation) and [min,max] interval of values are reported.

Ontology	R-Squared	Dunn's	Silhouette	#clusters
FSM	.39 (±.07)	.72 (±.10)	.77 (±.01)	4 (±.00)
	[.33,.52]	[.69,1.0]	[.74,.78]	[4,4]
SURFACEWATERMODEL	.45 (±.15)	.99 (±.03)	.999 (±.000)	12.9 (±.32)
	[.28,.66]	[.9,1.0]	[.999,.999]	[12,13]
TRANSPORTATION	.33 (±.04)	.67 (±.00)	.975 (±.004)	3 (±.00)
	[.26,.40]	[.67,.67]	[.963,.976]	[3,3]
NEWTESTAMENTNAMES	.46 (±.08)	.79 (±.17)	.985 (±.008)	29.2 (±2.9)
	[.35,.59]	[.5,1.0]	[.968,.996]	[25,32]
FINANCIAL	.37 (±.06)	.88 (±1.16)	.91 (±.03)	8.7 (±.95)
	[.29,.45]	[.57,1.0]	[.87,.94]	[8,10]

The R-Squared index values denotes an acceptable degree of separation between the various clusters. We may interpret the outcomes observing that clusters present a higher degree of compactness (measured by the SS_w component). It should also pointed out that flat clustering penalizes separation as the concepts in the knowledge base are not necessarily disjoint. Rather, they naturally tend to form subsumption hierarchies. Observe also that the variation among the various runs is very limited.

Dunn's index measures both compactness and separation; the rule in this case is *the larger the better*. Results are good for the various bases. These outcomes may serve for further comparisons to the performance of other clustering algorithms. Again, note that the variation among the various runs is very limited, so the algorithm was quite stable, despite its inherent randomized nature.

For the average Silhouette measure, that has a precise range of values, the performance of our algorithm is generally very good, with a degradation with the increase of individuals taken into account. Besides, the largest KB (in terms of its population) is also the one with the maximal number of concepts which provided the features for the metric. Thus in the resulting search space there is more freedom in the choice of the ways to make one individual discernible from the others. Surprisingly, the number of clusters is limited w.r.t. the number of concepts in the KB, suggesting that many individuals gather around a restricted subset of the concepts, while the others are only complementary (they can be used to discern the various individuals). Such subgroups may be detected extending our method to perform hierarchical clustering.

As regards the overall stability of the clustering procedure, we may observe that the main indices (and the number of clusters) show very little variations along the repetitions (see the standard deviation values), which suggests that the algorithm tends to converge towards clusterings of comparable quality with generally the same number of clusters. As such, the optimization procedure does not seem to suffer from being caught in local minima. However, the case needs a further investigation.

Other experiments (whose outcomes are not reported here) showed that sometimes the initial genome length may have an impact to the resulting clustering, thus suggesting the employment of different randomized search procedures (e.g. again simulated annealing or tabu search) which may guarantee a better exploration of the search space.

5 Automated Concept Evolution in Dynamic Ontologies

In this section we illustrate the utility of clustering in the process of the automated evolution of dynamic ontologies. Namely, clustering may be employed to detect the possible evolution of some concepts in the ontology as reflected by new incoming resources as well as the emergence of novel concepts. These groups of individuals may be successively employed by supervised learning algorithms to induce the intensional description of revised or newly invented concepts.

5.1 Incrementality and Automated Drift and Novelty Detection

As mentioned in the introduction, conceptual clustering enables a series of further activities related to dynamic settings: 1) concept drift [25]: i.e. the change of known concepts w.r.t. the evidence provided by new annotated individuals that may be made available over time; 2) novelty detection [23]: isolated clusters in the search space that require to be defined through new emerging concepts to be added to the knowledge base.

The algorithms presented above are suitable for an online unsupervised learning implementation. Indeed as soon as new annotated individuals are made available these may be assigned to the *closest* clusters (where closeness is measured as the distance to the cluster medoids or to the minimal distance to its instances). Then, new runs of the evolutionary algorithm may yield a modification of the original model (clustering) both in the clusters composition and in their number.

Following [23], the model representing the starting concepts is built based on the clustering algorithm. For each cluster, the maximum distance between its instances and the medoid is computed. This establishes a decision boundary for each cluster. The union of the boundaries of all clusters is the global decision boundary which defines the current model. A new unseen example that falls inside this global boundary is consistent with the model and therefore considered *normal*; otherwise, a further analysis is needed. A single such individual should not be considered as novel, since it could simply represent noise. Due to lack of evidence, these individuals are stored in a short-term memory, which is monitored for the formation of new clusters that might indicate two conditions: novelty and concept drift. Using the clustering algorithm on individuals in the short-term memory generales candidate clusters. For a candidate cluster to be considered valid, i.e. likely a concept in our approach, the following algorithm can be applied.

```
(decision,NewClustering) DRIFT_NOVELTY_DETECTION(Model, CCluster)
input:  Model: current clustering; CandCluster: candidate cluster;
output: (decision, NewClustering);

$m_{\mathsf{CC}} := \mathrm{medoid}(\mathsf{CandCluster})$;
for each $C_j \in \mathsf{Model}|$ do $m_j := \mathrm{medoid}(\mathsf{C_j})$;
$d_{\mathsf{overall}} := \frac{1}{|\mathsf{Model}|} \sum_{C_j \in \mathsf{Model}} \left( \frac{1}{|C_j|} \sum_{a \in C_j} d(a, m_j) \right)$;
$d_{\mathsf{candidate}} := \frac{1}{|\mathsf{CandCluster}|} \sum_{a \in \mathsf{CCluster}} d(a, m_{\mathsf{CC}})$;
if $d_{\mathsf{overall}} \geq d_{\mathsf{candidate}}$ then // valid candidate cluster
    begin
    $\overline{m} := \mathrm{medoid}(\{m_j \mid C_j \in \mathsf{Model}\})$; // global medoid
    $d_{\mathsf{max}} := \max_{m_j \in \mathsf{Model}} d(\overline{m}, m_j)$;
    if $d(\overline{m}, m_{\mathsf{CC}}) \leq d_{\mathsf{max}}$ then
        return (drift, replace(Model,CandCluster))
        else return (novelty, Model ∪ CandCluster)
    end
else return (normal, integrate(Model,CandCluster))
```

The candidate cluster CandCluster is considered valid[5] for drift or novelty detection when the average mean distance between medoids and the respective instances for all clusters of the current model is greater than the average distance of the new instances to the medoid of the candidate cluster. Then a threshold for distinguishing between concept drift and novelty is computed: the maximum distance between the medoids of the model and the global one[6]. When the distance between overall medoid and the medoid of the candidate cluster exceeds the maximum distance then the case is of concept drift and the candidate cluster is merged with the current model. Otherwise (novelty case) the clustering is simply extended. Finally, when the candidate cluster is made up of normal instances these can be integrated by assigning them to the closest clusters.

The main differences from the original method [23], lie in the different representational setting (simple numeric tuples were considered) which allows for the use of off-the-shelf clustering methods such as k-MEANS [16] based on a notion of centroid which depend on the number of clusters required as a parameter. In our categorical setting, medoids substitute the role of medoids and, more importantly, our method is able to detect an optimal number of clusters autonomously, hence the influence of this parameter is reduced.

[5] This aims at choosing clusters whose density is not lower than that of the model.

[6] Clusters which are closer to the boundaries of the model are more likely to appear due to a drift occurred in the normal concept. On the other hand, a validated cluster appearing far from the normal concept may represent a novel concept.

5.2 Conceptual Clustering for Concept Formation

The next step may regard the refinement of existing concepts as a consequence of concept drift or the invention of new ones to account for emerging clusters of resources. The various cluster can be considered as training examples for a supervised algorithm aimed at finding an intensional DL definition for one cluster against the counterexamples, represented by individuals in different clusters [19, 12].

Each cluster may be labeled with an intensional concept definition which characterizes the individuals in the given cluster while discriminating those in other clusters [19, 12]. Labeling clusters with concepts can be regarded as a number of supervised learning problems in the specific multi-relational representation targeted in our setting [15]. As such it deserves specific solutions that are suitable for the DL languages employed. A straightforward solution may be found, for DLs that allow for the computation of (an approximation of) the *most specific concept* (msc) and *least common subsumer* (lcs) [1] (such as $\mathcal{ALC}$). The first operator, given the current knowledge base and an individual, provides (an approximation of) the most specific concept that has the individual as one of its instances. This would allow for lifting individuals to the concept level. The second operator computes minimal generalizations of the input concept descriptions. Indeed, concept formation can be cast as a supervised learning problem: once the two clusters at a certain level have been found, where the members of a cluster are considered as positive examples and the members of the dual cluster as negative ones. Then any concept learning method which can deal with this representation (and semantics) may be utilized for this new task. Given these premises, the learning process can be described through the following steps:

```
let $C_j$ be a cluster of individuals
1. for each individual $a_i \in C_j$
   do compute $M_i := \mathsf{msc}(a_i)$ w.r.t. $\mathcal{A}$;
2. let $\mathsf{mscs}_j := \{M_i \mid a_i \in C_j\}$;
3. return $\mathsf{lcs}(\mathsf{mscs}_j)$
```

As an alternative, more complex algorithms for learning concept descriptions expressed in DLs may be employed such as YinYang [15] or other systems based on refinement operators [21]. Their drawback is that they cannot deal with the most complex DL languages. The concepts resulting from conceptual clustering can be used for performing weak forms of abduction that may be used to update the ABox; namely, the membership of an individual to a cluster assessed by means of the metric, may yield new assertions that do not occur in the ABox may be added (or presented to the knowledge engineer as candidates to addition). Induced assertions coming for newly available individuals may trigger further supervised learning sessions where concepts are refined by means of the aforementioned operators.

6 Conclusions and Extensions

This work has presented a framework for evolutionary conceptual clustering that can be applied to standard relational representations for KBs in the SW context. Its intended usage is for discovering interesting groupings of semantically annotated resources and can be applied to a wide range of concept languages. Besides, the induction of new concepts may follow from such clusters, which allows for accounting for them from an intensional viewpoint. The method exploits a dissimilarity measure that is based on the undelying resource semantics w.r.t. a committee of features represented by a group of concept descriptions in the chosen language. A preliminary learning phase, based on randomized search, can be used to optimize the choice of the most discriminating features. The evolutionary clustering algorithm is an extension of distance-based clustering procedures employing medoids as cluster prototypes so to deal with complex representations of the target context. Variable-length strings of medoids yielding different partitions are searched guided by a fitness function based on cluster separation. The algorithm can also determine the length of the list, i.e. an optimal number of clusters.

As for the metric induction part, a promising research line, for extensions to matchmaking, retrieval and classification, is *retrieval by analogy* [5]: a search query may be issued by means of prototypical resources; answers may be retrieved based on local models (intensional concept descriptions) for the prototype constructed (on the fly) based on the most similar resources. The presented algorithm may be the basis for the model construction activity. The distance measure may also serve as a ranking criterion. The natural extensions of the clustering algorithm that may be foreseen are towards incrementality and hierarchical clustering. The former may be easily achieved by assigning new resources to their most similar clusters, and restarting the whole algorithm when some validity measure crosses a given threshold. The latter may be performed by wrapping the algorithm within a level-wise procedure starting with the whole dataset and recursively applying the partitive method until a criterion based on quality indices determines the stop. This may produce more meaningful concepts during the next supervised phase. Better fitness functions may be also investigated for both distance optimization and clustering. For instance, some clustering validity indices can be exploited in the algorithm as measures of compactness and separation.

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook.* Cambridge University Press, 2003.

[2] J.C. Bezdek and N.R. Pal. Some new indexes of cluster validity. *IEEE Transactions on Systems, Man, and Cybernetics*, 28(3):301–315, 1998.

[3] A. Borgida, T.J. Walsh, and H. Hirsh. Towards measuring similarity in description logics. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Working Notes of the International Description Logics Workshop*, volume 147 of *CEUR Workshop Proc.*, Edinburgh, UK, 2005.

[4] C. d'Amato, N. Fanizzi, and F. Esposito. A dissimilarity measure for $\mathcal{ALC}$ concept descriptions. In *Proceedings of the 21st Annual ACM Symposium of Applied Computing, SAC2006*, volume 2, pages 1695–1699, Dijon, France, 2006. ACM.

[5] C. d'Amato, N. Fanizzi, and F. Esposito. Reasoning by analogy in description logics through instance-based learning. In G. et al. Tummarello, editor, *Proc. of Workshop on Semantic Web Applications and Perspectives, SWAP2006*, volume 201 of *CEUR*, 2006.

[6] C. d'Amato, S. Staab, N. Fanizzi, and F. Esposito. Efficient discovery of services specified in description logics languages. In *Proc. of the ISWC Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web.*, 2007.

[7] M. d'Aquin, J. Lieber, and A. Napoli. Decentralized case-based reasoning for the Semantic Web. In Y. et al. Gil, editor, *Proc. of the 4th Int. Semantic Web Conf., ISWC2005*, number 3279 in LNCS, pages 142–155. Springer, 2005.

[8] F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, and G. Semeraro. Knowledge-intensive induction of terminologies from metadata. In F. van Harmelen, S. McIlraith, and D. Plexousakis, editors, *ISWC2004, Proceedings of the 3rd International Semantic Web Conference*, volume 3298 of *LNCS*, pages 441–455. Springer, 2004.

[9] F. Esposito, S. Ferilli, N. Fanizzi, T.M.A. Basile, and N. Di Mauro. Incremental learning and concept drift in INTHELEX. *Jour. of Intelligent Data Analysis*, 8(1/2):133–156, 2004.

[10] N. Fanizzi, C. d'Amato, and F. Esposito. Induction of optimal semi-distances for individuals based on feature sets. In D. Calvanese et al., editor, *Working Notes of the 20th International Description Logics Workshop, DL2007*, volume 250 of *CEUR*, 2007.

[11] N. Fanizzi, C. d'Amato, and F. Esposito. Randomized metric induction and evolutionary conceptual clustering for semantic knowledge bases. In M. J. Silva et al., editor, *Proc. of the 16th ACM Conf. on Information and Knowledge Management*, pages 51–60. ACM, 2007.

[12] N. Fanizzi, L. Iannone, I. Palmisano, and G. Semeraro. Concept formation in expressive description logics. In J.-F. et al. Boulicaut, editor, *Proc. of the 15th Europ. Conf. on Machine Learning, ECML2004*, volume 3201 of *LNAI*, pages 99–113. Springer, 2004.

[13] A. Ghozeil and D.B. Fogel. Discovering patterns in spatial data using evolutionary programming. In John R. Koza et al., editor, *Genetic Programming 1996: Proc. of the 1st Annual Conf.*, pages 521–527. MIT Press, 1996.

[14] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[15] L. Iannone, I. Palmisano, and N. Fanizzi. An algorithm based on counterfactuals for concept learning in the semantic web. *Applied Intelligence*, 26(2):139–159, 2007.

[16] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[17] K. Janowicz. Sim-dl: Towards a semantic similarity measurement theory for the description logic $\mathcal{ALCNR}$ in geographic information retrieval. In R. Meersman et al., editor, *Proc. of SeBGIS 2006, OTM Workshops*, volume 4278 of *LNCS*, pages 1681–1692, 2006.

[18] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

[19] J.-U. Kietz and K. Morik. A polynomial approach to the constructive induction of structural knowledge. *Machine Learning*, 14(2):193–218, 1994.

[20] C.-Y. Lee and E. K. Antonsson. Variable length genomes for evolutionary algorithms. In L. Whitley et al., editor, *Proc. of the Genetic and Evolutionary Computation Conference, GECCO00*, page 806. Morgan Kaufmann, 2000.

[21] J. Lehmann. Concept learning in description logics. Master's thesis, Dresden University of Technology, 2006.

[22] M. Sebag. Distance induction in first order logic. In S. Džeroski and N. Lavrač, editors, *Proceedings of the 7th International Workshop on Inductive Logic Programming, ILP97*, volume 1297 of *LNAI*, pages 264–272. Springer, 1997.

[23] E.J. Spinosa, A. Ponce de Leon Ferreira de Carvalho, and J. Gama. OLINDDA: A cluster-based approach for detecting novelty and concept drift in data streams. In *Proc. of the Annual ACM Symposium of Applied Computing*, volume 1, pages 448–452. ACM, 2007.

[24] R. E. Stepp and R. S. Michalski. Conceptual clustering of structured objects: A goal-oriented approach. *Artificial Intelligence*, 28(1):43–69, Feb. 1986.

[25] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[26] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search – The Metric Space Approach*. Advances in database Systems. Springer, 2007.