

Evolutionary Conceptual Clustering of Semantically Annotated Resources

Nicola Fanizzi, Claudia d'Amato, Floriana Esposito
Dipartimento di Informatica – Università degli studi di Bari
Via Orabona 4, 70125 Bari, Italy
{fanizzi, claudia.damato, esposito}@di.uniba.it

Abstract

A clustering method is presented which can be applied to knowledge bases storing semantically annotated resources. The method can be used to discover groupings of structured objects expressed in the standard concept languages employed in the Semantic Web. The method exploits effective language-independent semi-distance measures over the space of resources. These are based on their semantics w.r.t. a number of dimensions corresponding to a committee of features represented by a group of discriminating concept descriptions. We show how to obtain a maximally discriminating group of features through a feature construction procedure based on genetic programming. The evolutionary clustering algorithm employed is based on the notion of medoids applied to relational representations. It is able to induce an optimal set of clusters by means of a proper fitness function based on the defined distance and the discernibility criterion. An experimentation with some real ontologies proves the feasibility of our method.

1. Introduction and Motivation

In the applications related to 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, evolution, and retrieval. The automation is likely to be assisted by extensions of supervised or unsupervised learning methods that are capable to the specific representations of the field [7, 4, 6, 9, 13].

In this work, we investigate on unsupervised learning for knowledge bases storing annotated resources whose semantics is expressed in standard concept languages (RDF through OWL). In particular, we focus on the problem of clustering semantically annotated resources. Indeed the benefits of *conceptual clustering* [18] can be manifold. Clustering enables 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 refining 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.

Many existing clustering methods are essentially based on the application of similarity (or density) measures defined over a fixed set of attributes of the resources. Good clusters are considered those that exhibit low intercluster similarity (density) and high intracluster similarity (density). Yet, often these methods do not take into account any form of *background knowledge* in a form that is capable of conveying semantics by characterizing resource groups by means of global concepts and relations. This hinders the interpretation of the outcomes which is crucial in the SW perspective which foresees sharing and reusing knowledge enforcing semantic interoperability.

Conceptual clustering methods for expressive representations (usually) define intensionally groups of objects with simple conjunctive descriptions based on selected attributes [18]. In a SW setting, the expressivity of the cluster descriptions (concepts) is particularly important. So far the approaches employing semantically rich representations such as *Description Logics* (DLs) [1], have pursued language-specific solutions to the problem that are essentially based on logic inference [12, 7] which makes them error-prone in case of noisy knowledge. This motivates the investigation on similarity-based methods which can be more noise-tolerant, and as language-independent as possible.

We propose an extension of effective clustering techniques, tailored for the SW context, that is intended for grouping similar resources w.r.t. a semantic dissimilarity measure. Upgrading existing algorithms to work on richer representations, like the concept languages used in the SW, required specific measures that could grasp the underlying semantics. However, 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. Moreover, they have been conceived for assessing *concept* similarity, whereas, for in-

ductive tasks, a notion of similarity between *individuals* is likely to be required. Another theoretical problem which makes it difficult to extend existing approaches is the *Open World Assumption* (OWA) that has to be considered when reasoning in the intended semantics of these languages.

Recently, dissimilarity measures for specific DLs have been proposed [4]. Although they turned out to be quite effective in some inductive tasks, they were still partly based on structural criteria which makes them fail to fully grasp the underlying semantics and hardly extensible to more complex languages. Therefore, a family of totally semantic dissimilarity measures has been devised, which can overcome these limitations [5]. Following the criterion of semantic discernibility of individuals, we defined a measure that is suitable for a wide range of languages since it is merely based on the discernibility of the input individuals with respect to a fixed committee of features represented by concept definitions. As such the measure is not absolute, yet it depends on the knowledge base they are applied to. Thus, also the choice of the optimal feature sets deserves a preliminary feature construction phase, which may be performed by means of a randomized search procedure based on *genetic programming*, whose operators are borrowed from recent works on ontology refinement [13, 9].

Distance-based clustering methods, descending from K-MEANS [10] are based on the notion of *centroids* as prototypes around which the other similar objects gather. This is typical of numeric or ordinal features representations. In our case, we deal with categorical features thus we recur to the notion of *medoids* [11] as central individuals in a cluster. In our evolutionary algorithm searches a space of genomes made up of strings of medoids with different lengths, optimizing a fitness function based on cluster separability. Medoids are computed according to the semantic measure induced with the method 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.

The paper is organized as follows. Sect. 2 recalls the family of measures and proposes a method for optimizing it. The clustering algorithm is presented and discussed in Sect. 3. After Sect. 4 concerning the related work, we report in Sect. 5 an experimental session aimed at assessing the validity of the method on ontologies available in the Web. Conclusions and extensions are finally examined in Sect. 6.

2. Semantic Distance Measures

Since our method is not intended for a particular representation, in the following we assume that resources, concepts and their relationships 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 handbook [1] for a thorough reference).

In this context, 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. \mathcal{A} contains assertions concerning the world state. The set of the individuals (resources) occurring in \mathcal{A} will be denoted with $\text{Ind}(\mathcal{A})$. Each individual can be assumed to be identified by its own URI. Sometimes, it could be useful to make the *unique names assumption* on such individuals.

As regards the inference services, like all other instance-based methods, our procedure may require performing *instance-checking*, which amounts to determining whether an individual, say a , belongs to a concept extension, i.e. whether $\mathcal{K} \models C(a)$ holds for a certain concept C .

2.1. A Semantic Metric for Individuals

For the clustering procedure, we have developed a new measure with a definition that totally depends on semantic aspects of the individuals in the knowledge base. For our purposes, we need a function to measure the (dis)similarity of individuals. However individuals do not have a syntactic structure that can be compared. On a semantic level, similar individuals should behave similarly with respect to the same concepts. Then, a way for assessing the similarity of individuals in a knowledge base can be based on the idea of comparing their semantics along a number of dimensions represented by a committee of concept descriptions. Following the ideas borrowed from metric learning in clausal spaces [17], we propose the definition of totally semantic distance measures for individuals in the context of a DL knowledge base.

The rationale of the measure is to compare them on the grounds of their behavior w.r.t. a given set of concept descriptions, say $F = \{F_1, F_2, \dots, F_m\}$, which stands as a group of discriminating *features* expressed in the language taken into account. A whole family of distance functions for individuals inspired to Minkowski's distances can be defined as follows [5]:

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

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

where $p > 0$ and $\forall i \in \{1, \dots, m\}$ the projection function π_i is defined: $\forall a \in \text{Ind}(\mathcal{A})$

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

The case of $\pi_i(a) = 1/2$ corresponds to the case when a reasoner cannot give the truth value for a certain membership query. This is due to the OWA normally made in this context.

It is easy to prove that these dissimilarity functions have the standard properties for semi-distances [5]:

Proposition 2.1 (semi-distance) *For a fixed feature set F and $p > 0$, given any three instances $a, b, c \in \text{Ind}(\mathcal{A})$. it holds that:*

1. $d_p^F(a, b) > 0$
2. $d_p^F(a, b) = d_p^F(b, a)$
3. $d_p^F(a, c) \leq d_p^F(a, b) + d_p^F(b, c)$

It cannot be proved that $d_p^F(a, b) = 0$ iff $a = b$. This is the case of *indiscernible* individuals with respect to the given set of hypotheses F .

Compared to other proposed distance (or dissimilarity) measures [3, 4], the presented functions do not depend on the constructors of a specific language, rather they require only (retrieval or) instance-checking for computing the projections through class-membership queries to the KB. Note also that the projections that determine the measure can be computed (or derived from maintained KB statistics) before the actual distance application, thus determining a speed-up in the computation of the measure. This is very important for algorithms that massively use this distance, such as all instance-based methods.

2.2. Measure Optimization

So far we made the assumption that F may represent a sufficient number of (possibly redundant) features that are able to discriminate really different individuals.

From preliminary experiments where the measure was employed at solving classification/retrieval problems (as in [6]), we could obtain good results by using the very set of both primitive and defined concepts found in the ontology. However, while redundancy may be beneficial (as discussed also in [17]) using too many features increases the computational effort required for the measure. The choice of the concepts to be included – *feature selection* – may be crucial. Therefore, we have devised a specific optimization algorithm founded in *genetic programming* which is able to find optimal choices of discriminating concept committees.

Various heuristics for optimizing the parameters for the measures can be foreseen. Among the possible sets of features, one would prefer those that are able to discriminate the individuals in the ABox.

Namely, since the function is very dependent on the concepts included in the committee of features F , two immediate heuristics can be derived:

- limit the number of concepts in the committee, including especially those that are able to actually discriminate individuals;
- search for optimal sets of discriminating features, by allowing the construction of new ones through the specific constructors of the language of choice.

These objectives can be accomplished by means of randomized optimization techniques especially when knowledge bases with large sets of individuals are available. Namely, part of the entire data (a *hold out set*) can be drawn in order to learn optimal F sets, in advance with respect to the successive usage for other specific purposes.

The randomized search in the space of potential optimal sets of features is performed by recurring to genetic programming. Essentially the algorithm (depicted in Fig. 1) searches the space of all possible feature committees starting from an initial guess (determined by `MAKEINITIALFS(K)`) based on the concepts (both primitive and defined) currently referenced in the knowledge base \mathcal{K} .

The outer loop gradually augments the cardinality of the candidate committees. It is repeated until employing larger feature committees would not yield a better fitness value with respect to the best fitness recorded in the previous iteration (with fewer features). If this sort of fix-point is not reached then the current committee is augmented with a new random concept by the `ENLARGEFS()`.

The inner loop is iterated for a number of generations until a stop criterion is met, based on the maximal value of generations `maxGenerations` or, alternatively, when a fitness value `fitnessThreshold` is reached by some feature set in the population, which can be returned.

The `MAXFITNESS()` routine computes the maximal fitness of the feature sets based on their *discernibility factor*. For instance, given the whole set of individuals (or just a hold out sample to be used to induce an optimal measure) $HS \subseteq \text{Ind}(\mathcal{A})$ the fitness function may be:

$$\text{DISCERNIBILITY}(F) = \sum_{(a,b) \in HS^2} \sum_{i=1}^{|F|} |\pi_i(a) - \pi_i(b)|$$

`GENERATEOFFSPRINGS()` constructs candidate sets of concepts to replace the current one; it was implemented by recurring to simple transformations of a feature set:

- choose $F_{sel} \in \text{currentFSs}$;
- randomly select $F_i \in F_{sel}$;
- replace F_i with $F'_i \in \text{RANDOMMUTATION}(F_i)$
- or
- replace F_i with one of its refinements $F'_i \in \text{REF}(F_i)$

```

FeatureSet OPTIMIZEFS( $\mathcal{K}$ , maxGenerations, fitnessThreshold)
input   $\mathcal{K}$ : current knowledge base
        maxGenerations: maximal number of generations
        fitnessThreshold: fitness threshold
output FeatureSet: FeatureSet

begin
currentBestFitness := 0;
formerBestFitness := 0;
currentFSs := MAKEINITIALFS( $\mathcal{K}$ );
formerFSs := currentFSs;
repeat
  generationNumber := 0;
  currentBestFitness := MAXFITNESS(currentFSs);
  while (currentBestFitness < fitnessThreshold) or
    (generationNumber < maxGenerations)
    begin
    offsprings := GENERATEOFFSPRINGS(currentFSs);
    currentFSs := SELECTFROMPOPULATION(offsprings);
    currentBestFitness := MAXFITNESS(currentFSs);
    ++generationNumber;
    end
  if (currentBestFitness > formerBestFitness) and
    (currentBestFitness < fitnessThreshold) then
    begin
    fitnessImproved := true;
    formerFSs := currentFSs;
    formerBestFitness := currentBestFitness;
    currentFSs := ENLARGIFS(currentFSs);
    end
  else
    fitnessImproved := false;
  end
until not fitnessImproved;
return BEST(formerFSs);
end

```

Figure 1. Feature set optimization algorithm.

The RANDOMMUTATION() routine constructs new random concepts to replace other, when restarting the search in a different part of the search space. Refinement of concept descriptions is language specific. E.g. for the case of \mathcal{ALC} logic, refinement operators have been proposed in [13, 9]. Complete operators are to be preferred to ensure that any concept may in principle be selected for the committee.

After a number of new feature sets is generated, these offsprings are evaluated and the best are included in the new version of the currentFSs vector; the maximal fitness value for these feature sets is also computed. On exiting the while-loop, the current best fitness is compared with the best one computed for the former feature set; if an improvement is detected then the outer repeat-loop is continued, otherwise (one of) the former best feature set(s) is selected for being returned as the result of the algorithm.

3. Evolutionary Clustering Around Medoids

The conceptual clustering procedure consists of two phases: one that detects the clusters in the data and the other that constructs an intensional definition for the groups of individuals detected in the former phase.

The first clustering phase implements a genetic programming learning scheme. The designed representation for the competing genes is made up of strings (lists) of individuals of different lengths, where each individual stands as prototypical for one cluster. Thus, each cluster will be represented by its prototype recurring to the notion of *medoid* [11, 10] w.r.t. the distance measure. The algorithm performs a search in the space of possible clusterings of the individuals optimizing a fitness measure maximizing discernibility of the individuals of the different clusters (inter-cluster separation) and the intra-cluster similarity measured in terms of our metric.

The second phase is more language dependent. The various cluster can be considered as sets of 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 [12, 7].

3.1. The Clustering Algorithm

The proposed clustering algorithm can be considered as an extension of methods based on genetic programming, where the notion of cluster prototypical instance of centroid, typical of the numeric feature-vector data representations, is replaced by that of medoid [11]: each cluster is represented by one of the individuals in the cluster, the medoid, i.e., in our case, the one with the lowest average distance w.r.t. all the others individuals in the cluster. Namely, the medoid of a group of individuals is the individual that has the shortest distance w.r.t. the others. Formally, given a cluster $C = \{a_1, a_2, \dots, a_n\}$, the medoid is defined:

$$m = \text{medoid}(C) = \underset{a \in C}{\operatorname{argmin}} \sum_{j=1}^n d(a, a_j)$$

Considering medoids has two advantages: it allows for categorical attributes, and their choice is dictated by the location of a predominant fraction of individuals inside a cluster making it is less sensitive to the presence of outliers.

In the algorithm, a genome will be represented by a list of medoids $G = \{m_1, \dots, m_k\}$. On each generation, those maximizing a fitness function are selected for passing to the next generation. Note that the algorithm does not prescribe a fixed length of these lists (as, for instance in K-MEANS and its extensions [10]), hence it should be able to detect an optimal number of clusters for the data.

medoidVector ECM(individuals, maxGenerations, minGap)

input:

individuals: set of individuals;
maxGenerations: max number of iterations;
minGap: minimal gap for stopping the evolution;

output:

medoidVector: list of medoids

begin

generation = 0;

gap = MAX_REAL; // between the best and worst genome

currentPopulation := INITIALIZE(popLength);

while (generation ≤ maxGenerations) **and** (gap > minGap)

begin

offsprings := GENERATEOFFSPRINGS(currentPopulation);

fitnessVector := COMPUTEFITNESS(offsprings);

currentPopulation := SELECT(offsprings,fitnessVector);

gap := (fitnessVector[popLength] - fitnessVector[1]);

++generation;

end

return currentPopulation[1]; // best genome

end

Figure 2. The Clustering Algorithm.

Fig. 2 reports a sketch of the clustering algorithm. After the call to the initialization procedure INITIALIZE() returning a randomly generated initial population of medoid strings (currentPopulation) in a number of popLength, it essentially consists of the typical generation loop of genetic programming. On each iteration it computes new offsprings of current best clusterings stored in currentPopulation. 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, which is used to select the best offsprings that survive, passing to the next generation.

The loop condition is controlled by two factors the maximal number of generation (the maxGenerations parameter) and the difference (gap) between the fitness of best and of the worst selected genomes in currentPopulation (which is supposed to be sorted in ascending order, 1 thru popLength). Thus another stopping criterion is met when this gap becomes less than the minimal gap minGap passed as a parameter to the algorithm, meaning that the algorithm has reached a (local) minimum.

In the GENERATEOFFSPRINGS procedure three mutation operators and one crossover operators are implemented:

MUTATE2NEIGHBOR(G) randomly select $m \in G$ and replace it with $m' \notin G$ such that: $\forall m'' \notin G : d(m, m') \leq d(m, m'')$

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

MUTATEBYLENGTHENING(G) select $m \notin G$ that is added to $G: G := G \cup \{m\}$

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 number of the offsprings may be set as another parameter of the ECM algorithm.

The fitness of a genome $G = \{m_1, \dots, m_k\}$ is computed by distributing all individuals among the clusters ideally formed around the medoids in that genome: each individual is assigned to the cluster with the closest medoid. Per each medoid $m_i, i = 1, \dots, k$, let C_i be such a cluster. Then, the fitness is computed:

$$\text{FITNESS}(G) = \lambda \sum_{i=1}^k \sum_{x \in C_i} \frac{d(x, m_i)}{|C_i|}$$

The factor $\lambda = 1/\sqrt{k+1}$ is introduced in order to penalize those clusterings made up of too many small clusters that could enforce the maximization in this way. This simple function could be replaced with more sophisticated clustering quality indices proposed in the literature [2, 10].

3.2. The Supervised Learning Phase

Each cluster may be labeled with an intensional concept definition which characterizes the individuals in the given cluster while discriminating those in other clusters [12, 7]. Labeling clusters requires solving a number of supervised learning problems in the specific DL representation targeted in our setting.

A straightforward solution may be found, for DLs (such as \mathcal{ALC}) that allow for the computation of (an approximation of) the *most specific concept* (msc) of an individual w.r.t. \mathcal{A} (i.e. the most specific concept the individual is an instance of) and *least common subsumer* (lcs, minimal generalization of the concepts in the input set) [1]. This involves the following steps:

Given a cluster of individuals C_j

- **for each** individual $a_i \in C_j$ **do**
 - compute $M_i := \text{msc}(a_i)$ w.r.t. \mathcal{A} ;
- **let** $\text{MSC}_{S_j} := \{M_i \mid a_i \in C_j\}$;
- **return** $\text{lcs}(\text{MSC}_{S_j})$

Alternatively, algorithms for learning concept descriptions expressed in DLs may be employed [13, 9]. Indeed, concept formation can be cast as a supervised learning problem: once disjoint clusters have been formed, the members of a cluster are considered as positive examples and the members of the other as negative ones (and vice versa). Then any concept learning method which can deal with this representation may be utilized for this new task.

4. Related Work

4.1. Relational Similarity Measures

As mentioned in the first section, various attempts to define semantic (dis)similarity measures for concept languages have been made, yet they have still a limited applicability to simple languages [3] and they often depend also on the structure of the descriptions [4].

Our measure is mainly based on Minkowski's measure and on a method for distance induction developed by Sebag [17] in the context of machine learning (for clausal logics). It is shown that the induced measure could be accurate when employed for classification tasks even though set of features (hypotheses) to be used were not the optimal ones (or they were redundant). Besides the language representation (and its semantics) the original method also differs for the clauses induced for defining the distance measure regard a single concept, that is the target concept to be learned. In our distance induction method we generalize this to any kind of feature, trying to optimize a discernibility criterion.

Indeed, a source of inspiration was *rough sets* theory [16] which aims at the definition of vague concepts by means of approximations determined by an indiscernibility relationship. Hopefully, these methods developed in this context will help solving the open points of our framework (see the next section) and suggest new ways to treat uncertainty.

Another related metric was defined [15] for the Herbrand interpretations of logic clauses as induced from a metric on ground atoms. Specifically, it may be employed to assess the dissimilarity of individuals by deriving a related *most specific concept* description accounting for them.

4.2. Clustering Procedures

To the best of our knowledge, no other method can be found in the literature which is able to work on complex representations, such as DL.

Our algorithm can be considered as a descendant of the K-MEANS and K-MEDOIDS clustering procedures [10] adapted to the relational representations. Specifically, in K-MEDOIDS a cluster is represented by its medoid (rather than a centroid that works conveniently only with numerical attributes), which is a mean of the points within a cluster. This is more robust w.r.t. to the presence of single outliers.

PAM (*Partitioning Around Medoids*) [11] is an early version of K-MEDOID. PAM is an iterative optimization method that combines relocation of points between potential clusters with re-nominating the points as potential medoids. The process is guided by an objective function, which may be computationally expensive. A PAM algorithm identifies clusters by the medoids. Medoids are robust representations of the cluster centers that are less sen-

sitive to outliers than other cluster profiles, such as the cluster means of K-MEANS. This robustness is particularly important in the common context that many elements do not belong exactly to any cluster, which may be the case of the membership in DL knowledge bases, due to the OWA. In addition, a PAM algorithm can comply with any specified metric, allowing a flexible definition of similarity. Many clustering algorithms allow only Euclidean distance in current implementations.

Two noteworthy descendants of PAM are CLARA [11] and CLARANS [14]. CLARA uses several samples to be subjected to PAM. The whole dataset is assigned to resulting medoids, the objective function is computed, and the best system of medoids is retained. In CLARANS a graph is considered whose nodes are sets of k medoids and an edge connects two nodes if they differ by exactly one medoid. While CLARA compares very few neighbors corresponding to a fixed small sample, CLARANS uses random search to generate neighbors by starting with an arbitrary node and randomly checking a number of neighbors. If a neighbor represents a better partition, the process continues with this new node. Otherwise a local minimum is found, and the algorithm restarts until a certain number of local minima is found. The best node (i.e. a set of medoids) is returned for the formation of a resulting partition.

Our evolutionary algorithm extends these procedures also for it autonomously detect a good estimate of the number of clusters k , which must fixed beforehand in the standard clustering methods. This may be difficult when scarce knowledge about the domain is available. As an alternative, hierarchical partitioning methods may be used, iteratively dividing bad clusters until a threshold value for cluster *quality* (many measures have been proposed in the literature [2, 10]) is passed making further partitions useless.

Further comparable clustering methods are those based on an indiscernibility relationship [8]. While in our method this idea is embedded in the semi-distance measure (and the choice of the committee of concepts), these algorithms are based on an iterative refinement of an equivalence relationship which induces clusters as equivalence classes.

5. Experimental Validation

For assessing the validity of the method, a number of populated OWL ontologies have been selected, namely: FSM, SURFACE-WATER-MODEL, TRANSPORTATION and NEWTESTAMENTNAMES from the Protégé library¹, and the FINANCIAL ontology² a testbed for the PELLET reasoner (the ABox was limited to assertions for a sample of 1000 individuals belonging to the various classes).

¹<http://protege.stanford.edu/plugins/owl/owl-library>

²<http://www.cs.put.poznan.pl/alawrynowicz/financial.owl>

Knowledge Base	DL	#concepts	#object prop.	#data prop.	#individuals
FSM	$SO\mathcal{F}(D)$	20	10	7	37
S.-W.-M.	$ALCO\mathcal{F}(D)$	19	9	1	115
TRANSPORTATION	$AL\mathcal{C}$	44	7	0	250
NTN	$SH\mathcal{I}\mathcal{F}(D)$	47	27	8	676
FINANCIAL	$AL\mathcal{C}\mathcal{I}\mathcal{F}$	60	16	0	1000

Table 1. Ontologies employed in the experiments.

Table 1 summarizes important details concerning the ontologies employed in the experimentation. The complexity of the clustering problems also depends on the number of assertions per individual available in the knowledge base, which varies from an individual to another. The PELLET reasoner (ver. 1.4) was employed to compute the projections. An overall experimentation of 10 repetitions on a dataset took from a few minutes to 41 mins on a 2.5GhZ (512Mb RAM) Linux machine.

Two measures are employed for evaluating the results of the method: the *squared sum error* (SSE) and the *silhouette* index [11]. For each ontology, the experiments have been repeated for 10 times. The average values for each index on the repetitions is considered. Moreover, the standard deviation and the range of minimum and maximum values observed during the experiments are also reported. The outcomes of the experiments are reported in Tab. 2.

The SSE index is a a measure of cohesion; its values are not very informative in themselves. However, one may consider its normalization over the (squared) average number of individuals in a cluster. This emerges also by considering the ratio of overall number of individuals (in Tab. 2) for the knowledge base over the number of clusters (4th column). If we divide the SSE by the square of this ratio we got a normalized measure of the average cluster compactness.

Conversely, the average silhouette index has a precise range of values ([0,1]): it combines cohesion and separation for individual points, as well as clusters and clusterings. It can be observed in the table that the performance of our algorithm is generally very good (from .927 to 1.0) with a degradation of the performance with the increase of individuals to take into account.

Besides, note that the hardest knowledge base (FINANCIAL) 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 any other. 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).

As regards the stability of the clustering procedure, we

may observe the main indexes (and the number of clusters) vary very little 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. Indeed, the optimization performed by the clustering procedure is two-fold: it does not optimize the choice of the clustering medoids but also their number, which is normally considered as a fixed parameter for other algorithms (see Sect. 4). Other experiments (not reported here) showed that sometimes the initial genome lengths may have an impact to the resulting clustering, thus suggesting the employment of different randomized search procedures (e.g. simulated annealing, tabu search) which may guarantee a better exploration of the search space.

6. Conclusions and Future Work

We presented a clustering method for relational representations which are typical in the SW field. It can be used to discover interesting groupings of semantically annotated resources in a wide range of concept languages. Besides, we propose a way for inducing new concepts that can account for the individuals in a cluster from an intensional viewpoint. The method exploits a novel dissimilarity measure that is based on a committee of features, i.e. a group of discriminating concept descriptions. The algorithm, is an evolutionary extension of clustering around medoids procedures adapted for dealing with complex representations.

Better fitness functions may be investigated for both the distance optimization procedure and the clustering one. For instance the very indexes that we employed for assessing the clustering validity can be exploited in the algorithm for better measures of compactness and separation. The distance optimization procedure may suffer from the well-known problem of local minima. To this purpose we are currently investigating the application of a randomized search procedure based on *simulated annealing* or *tabu search*. Another promising research line, for extensions to matchmaking, retrieval and classification, is *retrieval by analogy* [6]:

Knowledge Base	SSE index	Silhouette index	#clusters
FSM	30.254 (± 11.394) [14.344,41.724]	.998 (± 0.005) [.985,1.000]	12 (± 0.0) [12,12]
S.-W.-M.	11.971 (± 11.394) [7.335,13.554]	1.000 (± 0.000) [1.000,1.000]	12.9 (± 0.32) [12,13]
TRANSPORTATION	46.812 (± 5.944) [39.584,57.225]	.976 (± 0.000) [.976,.976]	21 (± 0.0) [21,21]
NEWTTESTAMENTNAMES	96.155 (± 24.992) [64.756,143.895]	.986 (± 0.007) [.974,.996]	29.3 (± 2.4) [26,33]
FINANCIAL	130.863 (± 24.117) [99.305,163.259]	.927 (± 0.034) [.861,.951]	8.4 (± 0.84) [8,10]

Table 2. Results of the experiments: average value (\pm standard deviation) and [min,max] range.

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 (w.r.t. some (dis)similarity measure). The presented algorithm may be the basis for the model construction activity. The distance measure may also serve as a ranking criterion.

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. Bezdek and N. Pal. Some new indexes of cluster validity. *IEEE Transactions on Systems, Man, and Cybernetics*, 28(3):301–315, 1998.
- [3] A. Borgida, T. 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 Proceedings*, Edinburgh, UK, 2005.
- [4] C. d’Amato, N. Fanizzi, and F. Esposito. Reasoning by analogy in description logics through instance-based learning. In G. Tummarello, P. Bouquet, and O. Signore, editors, *Proceedings of Semantic Web Applications and Perspectives, 3rd Italian Semantic Web Workshop, SWAP2006*, volume 201 of *CEUR Workshop Proceedings*, Pisa, Italy, 2006.
- [5] N. Fanizzi, C. d’Amato, and F. Esposito. Induction of optimal semi-distances for individuals based on feature sets. In D. Calvanese, E. Franconi, V. Haarslev, D. Lembo, B. Motik, A.-Y. Turhan, and S. Tessaris, editors, *Working Notes of the International Description Logics Workshop, DL2007*, volume 250 of *CEUR Workshop Proceedings*, Bressanone, Italy, 2007.
- [6] N. Fanizzi, C. d’Amato, and F. Esposito. Instance-based retrieval by analogy. In Y. Cho, R. Wainwright, H. Haddad, S. Shin, and Y. W. Koo, editors, *Proceedings of the 22nd Annual ACM Symposium of Applied Computing, SAC2007*, pages 1398–1402, Seoul, Korea, 2007. ACM.
- [7] N. Fanizzi, L. Iannone, I. Palmisano, and G. Semeraro. Concept formation in expressive description logics. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, editors, *Proceedings of the 15th European Conference on Machine Learning, ECML2004*, volume 3201 of *LNAI*, pages 99–113. Springer, 2004.
- [8] S. Hirano and S. Tsumoto. An indiscernibility-based clustering method. In X. Hu, Q. Liu, A. Skowron, T. Y. Lin, R. Yager, and B. Zhang, editors, *2005 IEEE International Conference on Granular Computing*, pages 468–473. IEEE, 2005.
- [9] 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.
- [10] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.
- [11] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990.
- [12] J.-U. Kietz and K. Morik. A polynomial approach to the constructive induction of structural knowledge. *Machine Learning*, 14(2):193–218, 1994.
- [13] J. Lehmann and P. Hitzler. A refinement operator based learning algorithm for the *ALC* description logic. In *Proceedings of the 17th International Conference on Inductive Logic Programming, ILP2007*, 2007.
- [14] R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In *Proceedings of the 20th Conference on Very Large Databases, VLDB94*, pages 144–155, 1994.
- [15] S.-H. Nienhuys-Cheng. Distances and limits on Herbrand interpretations. In D. Page, editor, *Proceedings of the 8th International Workshop on Inductive Logic Programming, ILP98*, volume 1446 of *LNAI*, pages 250–260. Springer, 1998.
- [16] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning About Data*. Kluwer Academic Publishers, 1991.
- [17] 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.
- [18] R. E. Stepp and R. S. Michalski. Conceptual clustering of structured objects: A goal-oriented approach. *Artificial Intelligence*, 28(1):43–69, Feb. 1986.