FCA-based Search for Duplicate objects in Ontologies

Dmitry A. Ilvovsky and Mikhail A. Klimushkin

National Research University Higher School of Economics, School of Applied Mathematics and Informational Science
11 Pokrovskiy boulevard, Moscow, Russia
dilv_ru@yahoo.com, klim.mikhail@gmail.com

Abstract. A new approach for detecting duplicates in ontology built on real redundant data is considered. This approach is based on transformation of initial ontology into a formal context and processing this context using methods of Formal Concept Analysis (FCA). As a part of a new method we also introduce a new index for measuring similarity between objects in formal concept. We study the new approach on randomly generated contexts and real ontology built for a collection of political news and documents.

1 Introduction

One of the most generic ways to represent structured data is an ontology [2]. A common way to build an ontology is its automatic or semi-automatic generation from unstructured data (usually text). The problem of this approach is data redundancy, because different sources of information often contain duplicated information. Detecting and elimination of redundancy directly at the ontology building (or extending) stage (for instance, via pairwise comparison of new objects with existing ones) is not very effective because such an approach significantly increases burden on the expert who makes final decisions. Moreover, in real world redundant data comes to ontology unevenly and it makes sense to eliminate redundancy not every time when an ontology renews but do it at longer intervals. The duration of intervals can be determined by features of a particular domain.

In this work we consider a new method for effective identification of redundancy in data represented by an ontology. This method can be either used as an automatic detector of a list of potential duplicate objects or as a recommendation system. Algorithm is realized via union of closed sets of objects and based on Formal Concept Analysis methods [1].

2 Basic definitions

Formal Concept Analysis (FCA) [1] is an applied branch of lattice theory. From data analysis point of view, methods used in Formal Concept Analysis belong

to methods of object-attribute clustering. FCA considers not clusters of objects without their attribute descriptions, but groups of objects and attributes strongly related with each other.

Definition 1. A formal context $\mathbb{K}$ *is defined as a triple* (G, M, I)*, where* G *is called a set of* objects*,* M *is called a set of* attributes*,* $I \subseteq G \times M$ *is a binary relation specifies which objects have which attributes.*

If $g \in G$, $m \in M$ and gIm, the object g has the attribute m.

Definition 2. *The derivation operators* $(.)'$ *are defined for* $A \subseteq G$ *and* $B \subseteq M$ *as follows:*

$$A' \triangleq \{m \in M \mid gIm \forall g \in A\}, B' \triangleq \{g \in G \mid gIm \forall m \in B\}$$

Definition 3. Formal concept *of the context* $\mathbb{K} = (G, M, I)$ *is a pair* (A, B)*, where* $A \subseteq G$*,* $B \subseteq M$*,* $A' = B$ *and* $B' = A$*. The set* A *is called the* extent*, and* B *is called the* intent *of the concept* (A, B)*.*

Definition 4. *A concept* (A, B) *is a subconcept of* (C, D) *if* $A \subset C$ *(equivalently* $D \subset B$*). In this case* (C, D) *is called a superconcept of* (A, B)*.*

The set of all concepts of K ordered by subconcept-superconcept relation forms a lattice, which is called the *concept lattice* $\beta(\mathbb{K})$.

3 Problem statement

The problem solved in this paper is to search for duplicate objects in an ontology, i.e objects describing the same object in the real world. The original problem was posed by analysts of Avicomp company. Their main interest is to search for duplicate objects describing people and companies in an ontologies built by the automatic semantic processing flow of news texts. Currently, this problem in Avicomp is solved by methods based on the Hamming distance and a variety of additional heuristics.

The input to the algorithm takes an ontology constructed from text sources. An ontology contains objects of different classes. Objects can involve relationships, appropriate to their classes. The number of detected features and links between object can vary greatly. Some objects describe the same object in the real world.

At the output the algorithm should return lists of objects that have been detected as duplicates. The algorithm must have high precision because the returning of two different objects as duplicates is more critical error than not detecting some of the duplicates of the object.

4 A method of duplicates detection

The algorithm of duplicates detection consists of several stages:

1. Transformation of a source ontology to a multi-valued context.
2. Scaling multi-valued context.
3. Building the set of formal concepts of the context.
4. Building the set of potential duplicate objects.

4.1 Transformation of a source ontology to the multi-valued context

The source (instance of) ontology is transformed to the multi-valued context as follows:

1. A set of **context objects** is a set of objects O of the ontology.
2. A set of **context attributes** is a set $M = L \cup C \cup R$, where:
   - L is a set of all features defined by all ontology classes,
   - C is a set of binary attributes, which characterize object classes,
   - R is a set of binary attributes, which describe relations between ontology objects. Any relation $p(x, y)$ between ontology objects x and y generates two binary attributes in the context: $p(x, \_)$ and $p(\_, y)$. They correspond to the relations p from x and p to y. An object z has an attribute $p(\_, y)$ in context iff there exists a relation p from z to y in the source ontology.
3. Each object is incident to the values of its source attributes. Also, each object gets a special value *null* for those attributes not incident to it or those whose incidence to the object is unknown. Also it gets binary attributes, corresponding to the object's class (and all of its superclasses) and relations between this object and other objects.

4.2 Scaling multi-valued context

The multi-valued context is converted to a binary context by means of *scaling* [1, 3]. Attribute sets C and R are binary. Attribute set L is scaled depending on the type of attribute. As a rule, many attributes describe nonquantitative attributes of objects (e.g., a persons name, a company name, etc.). Moreover, many quantitative or numerical data are such that an approximated similarity by these attributes does not mean the similarity of objects. By way of example, if two company objects have attributes 2005 and 2006 as their year of establishment, the proximity (failure to match) of these attributes does not make us sure that the objects describe the same company; they more likely produce the opposite effect. For such attributes, only matching of their values makes sense and if the values are different then the distance between them does not matter. Such attributes are scaled by a *nominal scale*, i.e., its own binary attribute corresponds to each attribute value. In other attributes, some other scaling types are used such as:

- *Interval type*: the transformation of an attribute A into a set of binary attributes of type $a \leq A < b$. In this case, the intervals $[a, b)$ can be both disjoint and overlapping.
- *Ordering*: an attribute A is transformed into a set of binary attributes of $A > b$ type.
- Other scaling types which, in an experts opinion, can characterize the similarity of duplicate objects in the best way.

The experiments on the generated data and a real ontology use only nominal scaling; however, this does not restrain the generality of the proposed approach.

4.3 Building set of formal concepts of the context

There are several effective methods for building sets of formal concepts of the formal context. In this work the AddIntent [7] algorithm was used.

4.4 Formal concepts filtering

After building the set of formal concepts it is necessary to distinguish formal concepts having an extent which includes only duplicates of the particular object. Building special indices to filter concepts we have to consider all main features of these concepts. First, *index must increase if the number of attributes which are different for objects in a extent decreases all other things being equal.* To take into account this property we used the following index:

$$I_1(A, B) = \frac{|A||B|}{\sum_{g \in A} |\{g\}'|} \tag{1}$$

The second feature that an index must fulfill is — *it must increase when the number of common attributes is increasing, all other things being equal.* As a result we used an index that has got this feature:

$$I_2(A, B) = \sum_{m \in B} \frac{|A|}{|\{m\}'|} \tag{2}$$

The final index DII is a combination of two indices described earlier. In our work we used the following combination variants:

1. Linear combination of indices:

$$DII_+ = k_1 I_1 + k_2 I_2 \tag{3}$$

2. Product of indices with power coefficients:

$$DII_* = I_1^{k_1} * I_2^{k_2} \tag{4}$$

Absolute values of the coefficients have an influence only to the value of a threshold and filtering quality depends on the coefficients correlation in the index formula. So we can specialize indices family without loss of the optimum and take 1 as a value for the one of the coefficients:

$$DII_+ = I_1 + kI_2, k > 0, \tag{5}$$

$$DII_* = I_1 * I_2^k, k > 0 \tag{6}$$

4.5 Forming the set of potential duplicate objects

The lists of objects that the algorithm returns as potential duplicates are obtained from those formal concept extents that have high value of index. The algorithm have two work modes: *automatic operation mode* and *semi-automatic operation mode with an expert assistance*.

In automatic mode the algorithm filters formal concepts by the threshold of the developed index. At this stage, various heuristics can be added that are hard to account for by means of the index. Then the algorithm prepares the lists of duplicate objects. We assume that the binary relation "be a duplicate" is transitive and it holds for objects in a selected formal concept. In this case to find lists of duplicates we should find connected components in the graph of objects with the "duplicate" relation.

In semi-automatic operation mode with an expert assistance the algorithm consequently offers expert estimate concepts in descending order of DII values. The lists of duplicates are built as soon as an expert gives answers. Before asking an expert to estimate the concept the algorithm searches for all lists of duplicate objects with non-empty extent intersections with this object. If this extent is included in one of the lists then this concept is not offered to the expert. So the algorithm gets the list of objects corresponding to the current mark up made by the expert. Furthermore, the expert can stop the estimating process at each step and receive the formed lists of duplicate objects.

5 Experiments on artificially generated formal contexts

Basic experiments were conducted on artificially generated data with the duplicates known in advance in order to obtain statistical evaluation of the quality of the developed algorithm. Thus, this enables us to evaluate the quality of the method based on a large scope of input data and qualitatively compare it with the most widespread alternative approaches. Along with this, in the data generation, the features of ontology were taken into account to extrapolate the obtained results onto real data.

5.1 Input data generation

Various quality metrics on artificially generated contexts were used in order to evaluate the method quality. The generated formal contexts in this case have the properties of the contexts that were obtained from ontologies.

First, the generated contexts must contain a large number of objects and attributes. It is assumed that the objects will be measured in *tens of thousands*. The number of binary attributes is comparable to the number of objects, since many objects contain unique or rare attributes.

In this case, each object has a relatively small number of attributes. Their quantity does not exceed *several tens*. Therefore, the context is strongly scattered and despite its large dimension it has a relatively small number of formal concepts: from 5000 to 30000.

Second, the number of object attributes varies considerably and, as a rule, satisfies the Mandelbrot law, i.e., the number of attributes is in reverse proportion to the object range among the objects that are ordered by the number of their attributes.

The third property that is regarded in the context generation is an irregular distribution of attribute frequencies. Commonly, the attribute frequency is inversely proportional to its range in the sequence that is ordered by the frequency of the appearance of the attribute in the context objects. Upon generating unique objects, an input context was generated. An object in the context was generated for each object in the following way: each object attribute was added with a certain probability to the set of object attributes in the context. For some initial objects, several objects were thus generated. The obtained objects were taken as the duplicates of the same object.

5.2 Comparative analysis of the methods for detecting duplicates

As alternative methods we considered methods of pairwise comparison based on *Hamming distance* and *absolute similarity*. Also we considered the method which is similar to ours: the difference was in the use of *extensional stability* index [4, 8] instead of our index.

The method based on the concept extensional stability S. Kuznetsov was the first to introduce the formal concept stability in 1990 [4]. Later, in work [8], it was proposed to distinguish extensional and intentional stability. In our work, we deal with extensional stability since we assume that the objects that are considered as duplicates must be strongly related to a large number of attributes and have a small number of individual attributes. A formal concept they generate must be stable to the elimination of individual attributes.

The algorithm for searching for duplicates is similar to the basic method, viz. the most (extensionally) stable concepts are deleted from the set of formal concepts. Then it is assumed that the objects from the extent of the stable formal concept are the duplicates of the same object. The relationship R "to be duplicates" is built by the set of the chosen formal concepts. Then connectivity components for this relation are sought. The obtained components are given to the input as the lists of duplicate objects.

The method based on absolute similarity This method is based on the pairwise comparison of objects. Duplicate objects are assumed to have a large number of general attributes. Therefore, the number of general attributes serves the criterion of object similarity. The indicator that is based on this measure is the threshold of the quantity of general attributes.

The algorithm receives an incoming square similarity matrix $A : A[i][j] = k \Leftrightarrow$ the ith and jth objects have k general binary attributes and the threshold $t(N)$.

The matrix A and the threshold are used to build an adjacency matrix B : $A[i][j] > t \Rightarrow B[i][j] = 1$.

The adjacency matrix (similarly to the ingress matrix) is symmetrical and describes the similarity relationship R. Proceeding from the fact that the relationship "to be a duplicate" is an equivalence relationship and possesses transitivity, its transitive closure R^* is built using the obtained relationship R. The equivalence classes in R^* correspond to the object groups that are the duplicates of the same object. The same result can be obtained by detecting all the connectivity components of the relationship R.

The asymptotic complexity of the algorithm by time is $O(n^2 * m)$, where n is the number of objects in the formal context and m is the number of attributes.

The method based on Hamming distance The algorithm for detecting duplicates is based on the pairwise comparison of objects. The Hamming distance serves as the metric of similarity. First, a square matrix of the distances between objects is built. Then, using the obtained matrix A and a specified threshold $t(N)$ the matrix B of the relationship R "to be a duplicate" is built: $R : A[i][j] > t \Rightarrow B[i][j] = 1, (x_i, x_j) \in R$. The obtained relationship will be reflective and symmetrical. The connectivity components are sought by this relationship. The objects that enter the same connectivity component are considered as the duplicates of the same object.

The asymptotic complexity of the algorithm is similar to that of the algorithm based on absolute similarity, viz. $O(n^2 * m)$, where n is the number of objects in a formal context and m is the number of attributes.

5.3 The results

We used a few quality metrics for comparison of methods: recall, precision, average value of recall when precision is 100%, Mean Average Precision (MAP):

$$MAP(K) = \frac{\sum_{i=1}^{|K|} AveP(K_i)}{|K|} \tag{7}$$

$$AveP(k) = \frac{\sum_{c \in C_k} (P(c))}{|C_k|}, \tag{8}$$

where K is set of contexts, C_k is set of relevant formal concepts of the context k, $P(c)$ is number of the relevant concepts between all of the concepts having range (value of index) not lower than the concept c.

For the evaluation of the new method optimal coefficients for the index were primarily chosen. The coefficient was chosen using one of the generated contexts. The network on the positive real line was taken and the MAP index was maximized on it. Therefore, the coefficients for the used variants of the index DII were obtained:

$$DII_+ = I_1 + 0.25I_2, \tag{9}$$

$$DII_* = I_1 * I_2^{0.18} \tag{10}$$

The algorithm with this index was compared with the alternative methods for searching for duplicates. In order to build the function of algorithm precision versus its recall, *several tens* of different thresholds were specified and then for each threshold, the recall and the precision were calculated.

The method based on extensional stability demonstrates good results at a high index threshold. At a threshold above 0.5 only formal concepts that have duplicates are chosen. At a threshold below 0.5, the algorithm precision drops on average to 10%, since a large number of formal concepts with stability 0.5 are one attribute concepts that do not characterize duplicate objects.

The algorithm for searching for duplicates using Hamming distance has shown relatively low results. The Hamming distance takes into consideration only distinctions in attributes rather than the quantity of general attributes

The algorithm based on absolute similarity proved to be the most efficient among the considered alternative algorithms. In most cases, a large number of common attributes in a pair of objects means that these objects are duplicates. The disadvantage of the index is that it disregards the distinctions between objects.

The algorithm based on the new index demonstrated better results than the alternatives considered. The main distinct feature of the new method is small decrease of precision (down to 90%) while recall increases up to 70%. The results for DII_+ and DII_* are very similar. The difference is in that the behavior of DII_* is less stable, viz., while sometimes making errors at a large threshold the algorithm did not make errors at a low threshold and detected 42% of the duplicates

Table 1. Max. recall with abs. precision

Algorithm	Max. recall with 100% precision
Abs. similarity	6.22%
Hamming distance	0.56%
e-stability index	22.44%
DII_+ index	21.78%
DII_* index	9.49%

6 Expirements on a real ontology

The ontology for tests was built by Avicomp. This ontology was built and extended automatically by semantic analysis of several political news sites. The OntosMiner [13] programming tool set was used. The ontology contains **12006**

Table 2. Mean Average Precision

Algorithm	MAP
e-stability index	0.4992
DII_+index	0.9352
DII_* index	0.9382

Table 3. Optimal thresholds and search quality

Algorithm	Threshold	Recall	Precision
Abs. similarity	3.5	19.35%	98.82%
Hamming distance	0.5	34.37%	86.32%
e-stability index	0.5	22.44%	100%
DII_+ index	1.15	40.09%	99.58%
DII_* index	0.9	31.8%	99.55%

objects of different classes. We used our algorithm for detecting duplicates with objects belonging to classes "Person" and "Company". The ontology contains **9821** such objects. Though we searched for duplicates only in two classes we used all classes and relations between objects and classes in ontology as attributes of objects in these classes.

A rather simple heuristical constraint was added in the algorithm based on the new index DII (the DII_+ variant was used): we filtered out concepts which contained objects having different values of attribute Name or Last_name. The algorithm detected **905** group of objects. Group size ranged from 2 to 41 objects. The largest groups found by the algorithm described such people as Benjamin Netanyahu (41 objects), Julia Tymoshenko (35 objects), Vladimir Putin (34 objects), Dmitry Medvedev (33 objects), Steve Jobs (31 objects) etc. However, the main part of the detected groups contains 2 to 4 objects.

With experts assistance we estimated the precision of our algorithm. We could be sure that 98% of the detected groups consist of duplicates. Very often we can see groups, where attributes Name and Last_name are not common, but other attributes and relations let the algorithm place these objects in one group. For instance, the algorithm detected 7 objects, describing Ksenia Sobchak and having only 1 common ontology attribute but brought together because of same relations with other objects.

It is necessary to point out that attribute weights in index I_2 let algorithm detect large groups of objects describing Putin, Tymoshenko, Medvedev etc. The key feature of these objects is that all of them have a lot of attributes and relations that differ from other objects.

7 Conclusions

In this work a new algorithm for the detection of duplicate objects was introduced. The algorithm is based on methods of Formal Concepts Analysis. In particular a index for ranking formal concepts was proposed. The index allows one to select the set of concepts containing only duplicate objects with high accuracy. The proposed method was compared with other approaches to the solution of the problem of data duplication on randomly generated data and real ontology data. Experiments demonstrated the effectiveness of the new index. Further work will consist of estimating recall of the new method on a real ontology.

Acknowledgments

The results of the project "Mathematical Models, Algorithms, and Software Tools for Intelligent Analysis of Structural and Textual Data", carried out within the framework of the Basic Research Program at the National Research University Higher School of Economics in 2012, are presented in this work.

References

1. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. In: Springer, 1999
2. Maedche, A., Zacharias, V.: Clustering Ontology-based Metadata in the Semantic Web. In: Proc. of 6th European Conference on Principles of Data Mining and Knowledge Discovery.—2002.—P. 348 - 360
3. Prediger S.: Logical scaling in formal concept analysis. In: ICCS, Lecture Notes in Computer Science.—1997.—Vol. 1257. Springer.—P. 332 - 341
4. Kuznetsov S.O.: Stability as an Estimate of the Degree of Substantiation of Hypotheses on the Basis of Operational Similarity. In: Nauchno-Tekhnicheskaya Informatsiya, Seriya 2, Vol. 24, No. 12, pp. 21-29, 1990
5. Kuznetsov, S.O.: On stability of a formal concept. In: Annals of Mathematics and Artificial Intelligence.—2007.—Vol. 49.—P. 101–115
6. Kuznetsov, S.O.: A Fast Algorithm for Computing All Intersections of Objects in a Finite Semi-Lattice. In: Automatic Documentation and Mathematical Linguistics 27(5), 11-21, 1993
7. Merwe, D., Obiedkov, S., Kourie, D.: AddIntent: a new incremental algorithm for constructing concept lattices. In: LNCS, Springer.—2004.—P. 205 - 206
8. Roth, C., Obiedkov, S., Kourie, D.: On Succinct Representation of Knowledge Community Taxonomies with Formal Concept Analysis. In: IJFCS (Intl Journal of Foundations of Computer Science).—2008.—P. 383 - 404
9. Rudolf Wille.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Ordered Sets: Dordrecht/Boston, Reidel. - 1982. P. 445-470
10. Kuznetsov, S.O.: Mathematical Aspects of Concept analysis. In: Journal of Mathematical Sciences, Vol. 80, Issue 2, P. 1654 - 1698, Springer.—1996
11. Kuznetsov, S., Obiedkov, S., Roth, C.: Reducing the representation complexity of lattice-based taxonomies. In: 15th Intl Conf on Conceptual Structures, ICCS 2007.—Sheffield, UK.—LNCS/LNAI. Vol. 4604. Springer.—2007

12. Klimushkin, M.A., Obiedkov, S.A., Roth, C.: Approaches to the selection of relevant concepts in the case of noisy data. In: 8th International Conference, ICFCA2010, Morocco.—Springer.—2010
13. http://www.ontos.com/?page_id=630