Преглед НЦД 32 (2018), 1–26

Aleksandar Janjić
Faculty of Mechanical Engineering, University of Banjaluka

FRAMEWORK FOR FUZZY CLASSIFICATION OF DIGITIZED DOCUMENTS

Abstract. The classification of a text document with respect to a predefined set of classes is an assignment of one of the values 0 or 1 to each ordered pair (document, class), depending on whether the document belongs to the class or not. Fuzzy classification generalizes this notion by enabling the membership to be expressed by any real number between 0 and 1. In this paper, we show one possible method of fuzzy classification by using the existing formulas for calculating the distance of a document from a class. As an illustration, we use this method to form a fuzzy classification of a subset of documents from Ebart-hier corpus. After that, we briefly describe the current state of the National Center for Digitization virtual library and show by an example how fuzzy classification can be used to improve the organization of the Library data and extend the querying possibilities.

1 Introduction

Since Zadeh's Fuzzy Sets theory was formulated in mid-60-ties of the 20th century, as well as Codd's relational model of data in 1970, different approaches have been proposed to extend databases, especially relational ones, in such a way as to manage incomplete, vague, unknown, imprecise data, giving rise to different fuzzy database models. Still, implementation of fuzzy database management systems has not been a well-established practice yet, and applications in different areas that may benefit from such systems are still under exploration. Actually, development of fuzzy database management systems strongly depends on applications that may take advantage of flexible data management provided by fuzzy databases. This paper deals with such an application, namely – fuzzy classification of digitized documents.

In this paper different fuzzy database models are presented first - possibilistic models and similarity relationship models, with similarity and proximity relations as opposed to equality relation in the crisp relational model.

Then, document classification problem is introduced as a real-life problem that may take full advantage of flexible data treatment and fuzzy model in specific.

The distance between two document classes, satisfying metric conditions, is introduced for flat and hierarchically structured document corpora. For flat corpora classes may be equidistant but for hierarchically structured corpora a simplified variant of a Depth-dependent Measure classification error [4] is introduced as a sum of weights of all the edges on the path between the two classes (considering the classification tree as an undirected graph). Then some examples of similarity relations among two documents (or a document and a class) are presented, e.g., the one based on k nearest neighbors (kNN) and n-gram vectors of characters, bytes or words. Based on the similarity of documents and classes, a fuzzy membership relation (of a document and set of classes) is then defined.

Finally, a framework for fuzzy classification of a digitized archive (e.g., a hierarchically structured portion of the library of the National Center for Digitization, NCD library) will be presented and possibilities for fuzzy retrieval will be outlined.

2 Fuzzy relational database models

2.1 Relational database model. The relational database model was defined by Edgar F. Codd in 1970. It was based on the elements of set theory and the first order predicate calculus, making it the first database model with a firm mathematical foundation, which enabled it to quickly replace at that time popular models like the hierarchical or network model. Today it is still the most widespread database model in commercial applications, though the systems in which all of its features are fully implemented are very rare. We will expound briefly and informally some basic elements of the relational database model [9].

We can view a relational database as a set of *tables*, together with the constraints defined over them. They are composed of *rows* and *columns*, whose intersections are the *fields* that contain the data. Tables are used to represent sets of entities. Each row describes a particular entity and each column corresponds to one of the entity's properties. Let us consider, for example, a database containing the information about the students of a university. The table named Student could describe the characteristics like the first name, the last name, logbook number, date and place of birth, average grade, etc. Those characteristics would be represented by the columns of the table, while the rows would correspond to each particular student.

Each column has a *domain*, which is the set of all possible values in the fields of that column. Each field contains exactly one value from the corresponding column domain (e.g., one student cannot have two different birth dates). This condition is called *the first normal form* (1NF) and the tables that satisfy it are said to be *normalized*. In case a column is of such a type that it is possible for more than one value to correspond to the same entity, multiple rows will be created for such entities - one for each corresponding domain element. For example, in a table representing a set of musicians and containing, among others, the information about the instruments they play, it is likely that there will exist a musician that is proficient at playing two or more instruments. For a musician that plays two instruments, say guitar and violin, the table will contain two rows, one with the value 'guitar' in the corresponding column and another with the value 'violin'.

To illustrate these concepts, we will show two tables with the data about musicians and the instruments they play.

Name	Year of birth	Place of birth	Band
Roy Wood	1946	Birmingham	Wizzard
Jeff Lynne	1947	Birmingham	ELO
Bev Bevan	1944	Birmingham	The Move
Richard Tandy	1948	Birmingham	ELO
Trevor Burton	1944	Birmingham	The Move
Hugh McDowell	1953	London	ELO
Mike Burney	1938	Birmingham	Wizzard

Table 1. Musicians

Name	Instrument
Roy Wood	guitar
Jeff Lynne	guitar
Roy Wood	bass
Richard Tandy	bass
Bev Bevan	drums
Hugh McDowell	cello
Richard Tandy	keyboards
Jeff Lynne	piano
Roy Wood	drums
Roy Wood	saxophone
Mike Burney	saxophone
Mike Burney	clarinet

Table 2. Instruments

In Table 1, assuming there are no two musicians with the same name, knowing a musician's name suffices to also know all of his other characteristics. More precisely, the values in the column ‘Name’ unambiguously determine the content of all the other columns. A column or a set of columns with this property is called a *key*.

If we assume that in Table 2 only the names of the musicians from Table 1 can appear, then each name from Table 2 will correspond to one name from Table 1, where ‘Name’ is the key of Table 1. Such a column or a set of columns in Table 2 is called a *foreign key* referencing Table 1.

2.2 Fuzzy sets. *Fuzzy sets* were introduced by Lotfi Zadeh in his 1965 paper [25]. Apart from a full membership of an element in a set (usually represented with 1) and full non-membership (usually 0), they also allow partial membership, represented by the real numbers between 0 and 1. So, for example, the membership of the element a in the fuzzy set F can be 0.75, or 0.12. That is why fuzzy sets are a convenient way of representing various subjective or imprecise concepts where a clear border between membership and non-membership does not exist. Zadeh cites the set of all tall people as an example. It is clear that the classical set theory cannot make this notion precise in a satisfying way. If, for example, we choose to define the tall people as those who are taller than 180 cm, then those between 179 and 180 cm would not be the members of that set, even though they differ very little from some of its members. On the other hand, in the theory of fuzzy sets, to a 179 cm tall person, we could assign the membership degree of, say, 0.99, which would nicely illustrate its proximity to the full membership in the set.

A fuzzy set is characterised by its *membership function*, which is a generalization of the notion of the characteristic function from the classical set theory. Let the universal set $U = \{x_1, x_2, \dots, x_n\}$ be given. A fuzzy set *F* in the universe *U* is the set of ordered pairs

$$F = \{(x_1, \mu_F(x_1)), (x_2, \mu_F(x_2)), \dots, ((x_n, \mu_F(x_n))\},$$

where $\mu_F: U \to [0,1]$ is the membership function of the fuzzy set *F*. (Note: This definition assumes that *U* is a finite set, but with obvious modifications it applies to sets of arbitrary cardinality). In the literature, the fuzzy sets are often denoted by

$$F = \mu_1/x_1 + \mu_2/x_2 + \cdots + \mu_n/x_n,$$

where $\mu_i = \mu_F(x_i)$, "+" denotes the set union, and "/" the ordered pair of an element and its associated membership degree.

The set equality, subsets and set operations are defined in terms of the membership functions. All these operations can be defined in various ways [2], and here we will mention the most common ones.

Let the universe U be given and let fuzzy sets A and B in U be defined by their membership functions μ_A and μ_B, respectively. The fuzzy A is a subset of the fuzzy set *B* (*A*⊂*B*) if and only if $\mu_A(x) \leq \mu_B(x)$ for all *x* in *U*. The sets *A* and *B* are equal if and only if $A \subset B$ and $B \subset A$. The membership functions of the results of some of the most common set operations are:

$$\mu_{A \cup B}(x) = max\left(\mu_A(x), \mu_B(x)\right)$$

$$\mu_{A \cap B}(x) = min\left(\mu_A(x), \mu_B(x)\right)$$

$$\mu_{\acute{A}}(x) = 1 - \mu_A(x)$$

2.3 Fuzzy relational database models. Though the relational model itself doesn't put any constraints on the structure of the domains in the database, so that theoretically they can be of arbitrary complexity [9], in practice the domains are most often represented by "simple" predefined sets like *integer*, *string*, *char*, *date*, etc. The implementation of fuzzy concepts in databases requires different, more complex domains, as well as additional operators that would provide the necessary functionality in the representation and management of data. The difference between the traditional and fuzzy logic is most clearly seen when the results of a query are displayed. In traditional databases, a row of a table will either satisfy the query, in which case it is displayed on the screen, or it will not. In fuzzy databases the user requirements will almost always be satisfied at least to some small degree, so the problem of displaying the results is not trivial.

The main goal of the fuzzy database research is to provide the possibility of making queries that are as close as possible to the natural language queries, where fuzzy concepts like "tall", "short", "close", "distant", "young", "old", "intelligent", "experienced", "capable" etc. are unavoidable. Different approaches provide different degrees of such functionality. Here we will mention a few of the approaches to the representation of data in fuzzy databases. Aside from the data representation, fuzzy queries, as well as the logical fuzzy database design, constitute the important research problems [2].

2.3.1 Fuzzy relation based model [3, 27]. In this model, the column domains are like in the traditional databases, but the membership of a row in a table can be partial. In practice this means that the table will have an additional column, usually denoted by μ, with values between 0 and 1, that would "measure" the membership of the row to the table. This model would be useful if, for example, we would like to modify the table with the musicians and the instruments to include the capability of a musician to play the corresponding instrument, from the degree 0 (doesn't play it at all) to 1 (plays it perfectly). The rows with μ value zero don't appear in the table. Table 3 is an example of such a model

Name	Instrument	μ
Roy Wood	guitar	0.92
Jeff Lynne	guitar	0.8
Roy Wood	bass	0.75
Richard Tandy	bass	0.7
Bev Bevan	drums	0.92
Hugh McDowell	cello	0.9
Richard Tandy	keyboards	0.9
Jeff Lynne	piano	0.75
Roy Wood	drums	0.64
Roy Wood	saxophone	0.69
Mike Burney	saxophone	0.92
Mike Burney	clarinet	0.88

*T*able 3 Instruments – fuzzy version

2.3.2 Similarity-based model [1,6]. The aforementioned model only added a new column to the table and didn't change the structure of the existing domains. In the similarity based model, the column domains are sets of sets, so that a field value doesn't have to be a single element, but an arbitrary subset of some set. Another way of looking at this is to assume that the domains are unchanged, but that the fields don't necessarily contain only one value from the corresponding domain, but their subsets (including, of course, singleton subsets), except for the empty set. In this case, the first normal form assumption doesn't hold anymore.

Another characteristic of this model is the existence of the *similarity* relation over the column domains. That relation provides the possibility, if needed, to group the elements of some domain together if they are close enough to one another, even though they are not truly equal. The similarity relations are usually given in the form of a symmetrical square matrix with ones on the main diagonal [24]. This is the consequence of the properties of reflexivity and symmetry of the similarity relation.

The third property of the similarity relation is the *max-min transitivity*: If by *s(a,c)* we denote the similarity between the elements *a* and *c*, then $s(a,c)\geq\min(s(a,b),s(b,c))$, for any element *b*. These three properties of the similarity relation *s* over some domain give the possibility of forming, for arbitrary $\alpha\in[0,1]$, an equivalence relation over that domain. This relation partitions the domain into disjoint sets such that within each of them the similarity of any two elements is larger than α. One possible similarity relation of musical instruments, formed according to their type, is given in Table 4.

Instrument	G	B	K	P	C	D	S	Cl
Guitar (G)	1	0.8	0.2	0.2	0.2	0	0	0
Bass (B)	0.8	1	0.2	0.2	0.2	0	0	0
Keyboards (K)	0.2	0.2	1	0.95	0.3	0	0	0
Piano(P)	0.2	0.2	0.95	1	0.3	0	0	0
Cello (C)	0.2	0.2	0.3	0.3	1	0	0	0
Drums (D)	0	0	0	0	0	1	0	0
Saxophone (S)	0	0	0	0	0	0	1	0.7
Clarinet (Cl)	0	0	0	0	0	0	.7	1

Table 4. Musical instruments similarity relation

For example, for α=0.6, the *merging* [16,24] of the rows based on the values of the Instrument column could lead to the creation of Table 5, which relates groups of instruments to the musicians that play them.

Name	Instruments
Roy Wood, Jeff Lynne, Richard Tandy	guitar, bass
Richard Tandy, Jeff Lynne	piano, keyboards
Hugh McDowell	cello
Roy Wood, Jeff Lynne, Bev Bevan	drums
Mike Burney, Roy Wood	saxophone, clarinet

Table 5. Instrument groups and musicians

2.3.3 The possibility based model [17, 23, 26]. This approach, similarly, deals with the representation of impreciseness within the individual table fields. The value of any field can be a *possibility distribution* over the corresponding domain. A possibility distribution is essentially a fuzzy set defined over the column domain, which represents the universal set.

As an example of this approach, we can take the data of Table 3 and represent them in another way. The data on any single musician would be put in a single row, where the rightmost column values would be the list of all instruments he plays, together with the corresponding values of the μ column:

Name	Instruments
Roy Wood	0.92/guitar+0.75/bass+0.64/drums+0.69/saxophone
Jeff Lynne	0.8/guitar+0.75/piano
Richard Tandy	0.7/bass+0.9/keyboards
Bev Bevan	0.92/drums
Hugh McDowell	0.9/cello
Mike Burney	0.92/saxophone+0.88/clarinet

Table 6. Instruments – possibility distribution

2.3.4 Extended possibility based model [7,19]. The extended possibility based model represents a generalization of the two previously mentioned models. Like in the previous model, fuzzy sets can exist within individual fields. Also, so-called *closeness* relation, satisfying the properties of reflexivity and symmetry, is defined over each column domain. The reason for this more general relation is that it's very hard to define a satisfying similarity relation for some very important domains, especially those that are linearly ordered. The problem lies in the max-min transitivity property, which sets very strict constraints on the values in the similarity matrix [16].

3 Text classification

The text classification has the task of classifying a text in natural language into one of the predefined classes.

In *single-label* classification, the classes are mutually disjoint, while in *multi-label* classification they are not, so that the document can belong to zero, one or more classes at the same time. Classification can also be *supervised*, when the information about the true corresponding class is provided, *unsupervised*, when no such information exists, and *semi-supervised*, when information about the corresponding classes of some of the documents comes from some external source.

Text classification can be performed manually, but that task is time-consuming and expensive. Given the wide availability, low cost and high speed of computers, automatic classification is becoming a standard in the efficient processing of documents, including classification. Among the methods of automated text classification, knowledge-based methods require the existence of appropriate knowledge bases, while so-called statistical classification methods [18], i.e. methods based on machine learning, require the existence of labeled training instances. Among the most used methods of machine learning applied to automated text classification are *k nearest neighbors* (kNN) and *support vector methods* (SVM).

Classification can be flat, when there is no relation that would define a class structure, and hierarchical, when such a relation exists. Hierarchical classification helps in searching the classes when their number significantly increases, and it is necessary when the nature of the classification problem is hierarchical itself.

Finally, classification can be *crisp*, when a document either belongs or doesn't belong to a class, and *fuzzy*, when belonging of an element to a set (of a document to a class) is characterized by a certain degree.

Formally, crisp text classification is a task that assigns a truth value to each pair $(d_j, C_i) \in D \times C$ where D is a set of documents, and $C = \{C_1, C_2, \dots, C_n\}$ a set of predefined classes. The value T (true) means that the document belongs to a class, the value F (false) that it does not. Actually, it is required to approximate an unknown goal function $\Phi: D \times C \to \{T, F\}$ that describes true classification by a function $\Phi': D \times C \to \{T, F\}$, that is called a classifier and which coincides with Φ as much as possible [20].

On the other hand, in fuzzy classification to a pair $(d_j, C_i) \in D \times C$ is assigned, instead of a truth value, a real number in the interval [0,1] as a degree of belonging of the document d_j to the class C_i and the classifier function is now a mapping $\Phi: D \times C \to [0,1]$, for example Φ'(*d*,*c*)=0.75.

4 Distance measures of documents and classes

4.1 Distance between classes. In this paper, our primary interest is the distance between documents and classes or between individual documents. However, it is also useful to consider defining the distance between two classes. One way of forming such a distance function is to consider their places in a hierarchy if one exists.

Let $C = \{C_1, C_2, \dots, C_n\}$ be a class set. In case that all classes are mutually independent, i.e. there's no hierarchy, we can assume that any two classes are "maximally" distant. We can assign any numerical value to such "maximal" distance. Here we will use the value 1, in order to have a normalized distance function.

Definition 4.1 *Let $C = \{C_1, C_2, \dots, C_n\}$ be a set of mutually independent classes. We will call the value of the function*

$$d(C_i, C_j) = \begin{cases} 0, i = j \\ 1, i \neq j \end{cases}$$

the distance between classes C_i and C_j.

In case we want to represent different values of the distance between classes, it is natural to assume that classes that are close to one another have some mutual superclass that denotes their common properties. This way we get a tree structure in which individual classes are represented by the nodes of a tree. Thus, we can use some measure of the distance between the tree nodes as a formula for the distance between classes.

A convenient measure of the distance between nodes is the so-called *shortest path distance* [4]. It denotes the smallest number of branches on a path from one node to another. This measure uses the *depth* of a tree node. To get a normalized measure of the distance between classes, we will divide the distance between the corresponding nodes by the maximal distance between any two nodes, which equals twice the height of the tree.

Definition 4.2 *Let classes$C_1, C_2, \dots, C_n$ represent the nodes of a tree of height n, where$depth(C_i)$ denotes the depth of the node corresponding to the class C_i. The distance between the classes C_i and C_j is*

$$d(C_i, C_j) = \frac{depth(c_i) + depth(C_j) - 2depth\left(DCA(C_i, C_j)\right)}{2n}$$

where $DCA(C_i, C_j)$ is the deepest common ancestor of the nodes representing classes C_i and C_j.

In this definition, we assumed that each branch of the tree has equal weight. We can form many different distance functions in which, say, the branches near the root of the tree would have a larger weight than those closer to the leaves. That way, for example, the distance between two direct ancestors of the root would be higher than between a class and its subclass' subclass, despite the fact that the path between the classes in both cases has two branches.

Example 4.1 *We will consider a (hypothetical) class hierarchy of the documents of the National Center for Digitization [15] (Figure 1). It is represented by a balanced tree of height 2. The classes listed are 'Mathematics', 'Astronomy' and 'Computer Science' and some of their subclasses. 'Discrete mathematics' and 'Applied mathematics' (Figure 2) are the subclasses of the class 'Mathematics'. The path between the classes has the length 2, while the double height of the tree is 4. According to the formula in the definition 4.2, the distance between these two classes is 0.5.*

*The class 'Observational Astronomy' is a subclass of the class 'Astronomy', while 'Theoretical Computer Science' is a subclass of 'Computer Science'. These two classes are maximally distant – the length of the shortest path between them is 4, so that their normalized distance according to the formula is 1 (*Figure *3).*

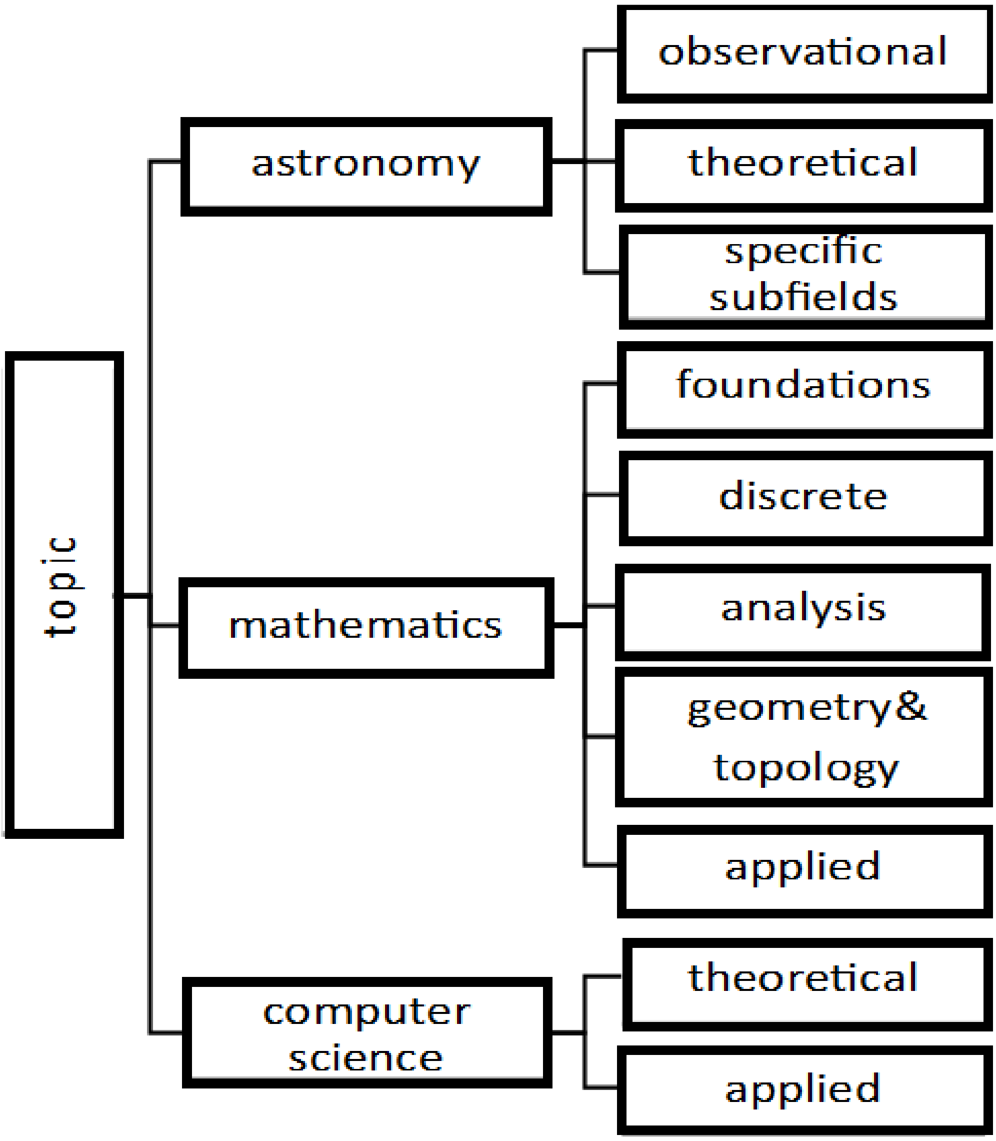

Figure 1. Class hierarchy of the documents of the National Center for Digitization

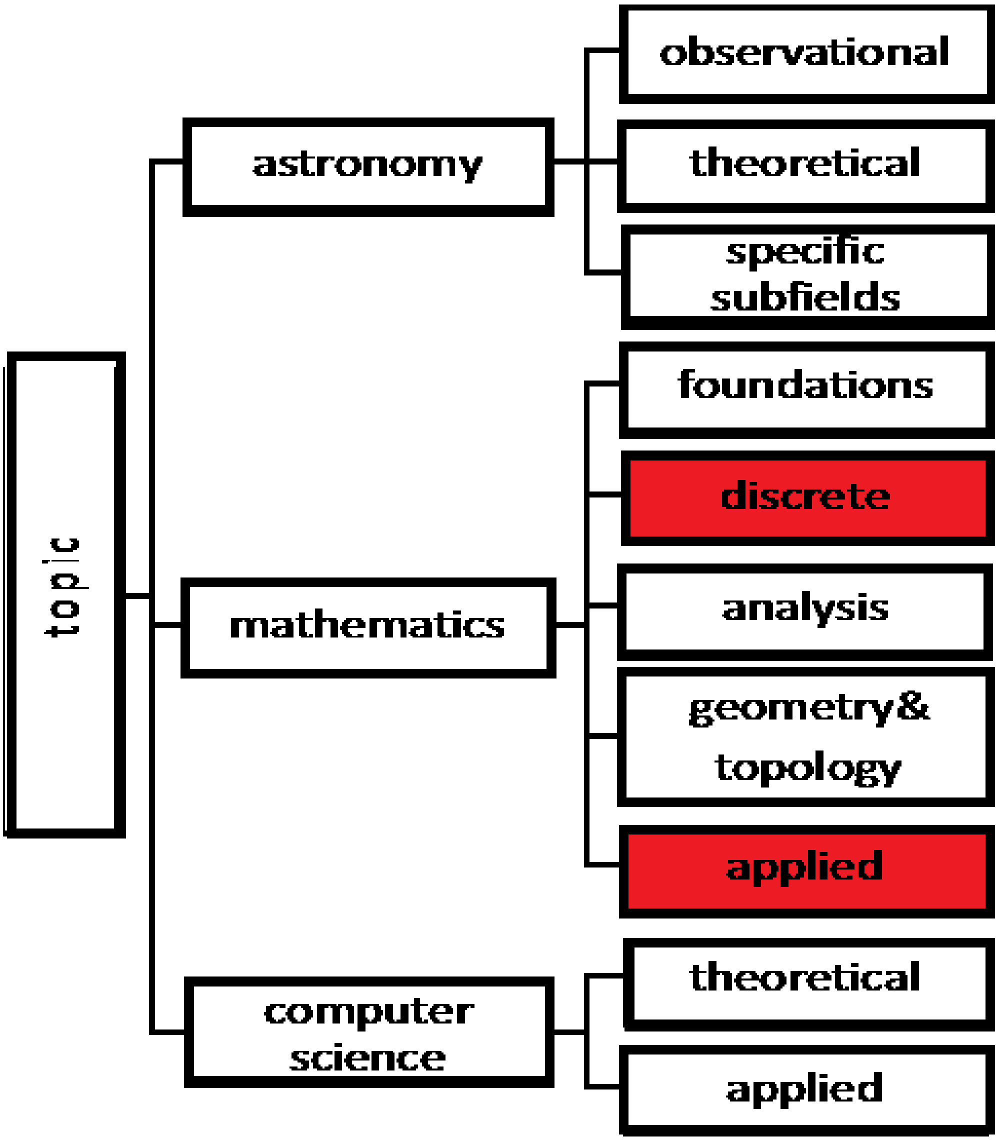
topic
astronomy
observational
theoretical
specific subfields
mathematics
foundations
discrete
analysis
geometry& topology
applied
computer science
theoretical
applied

Figure 2

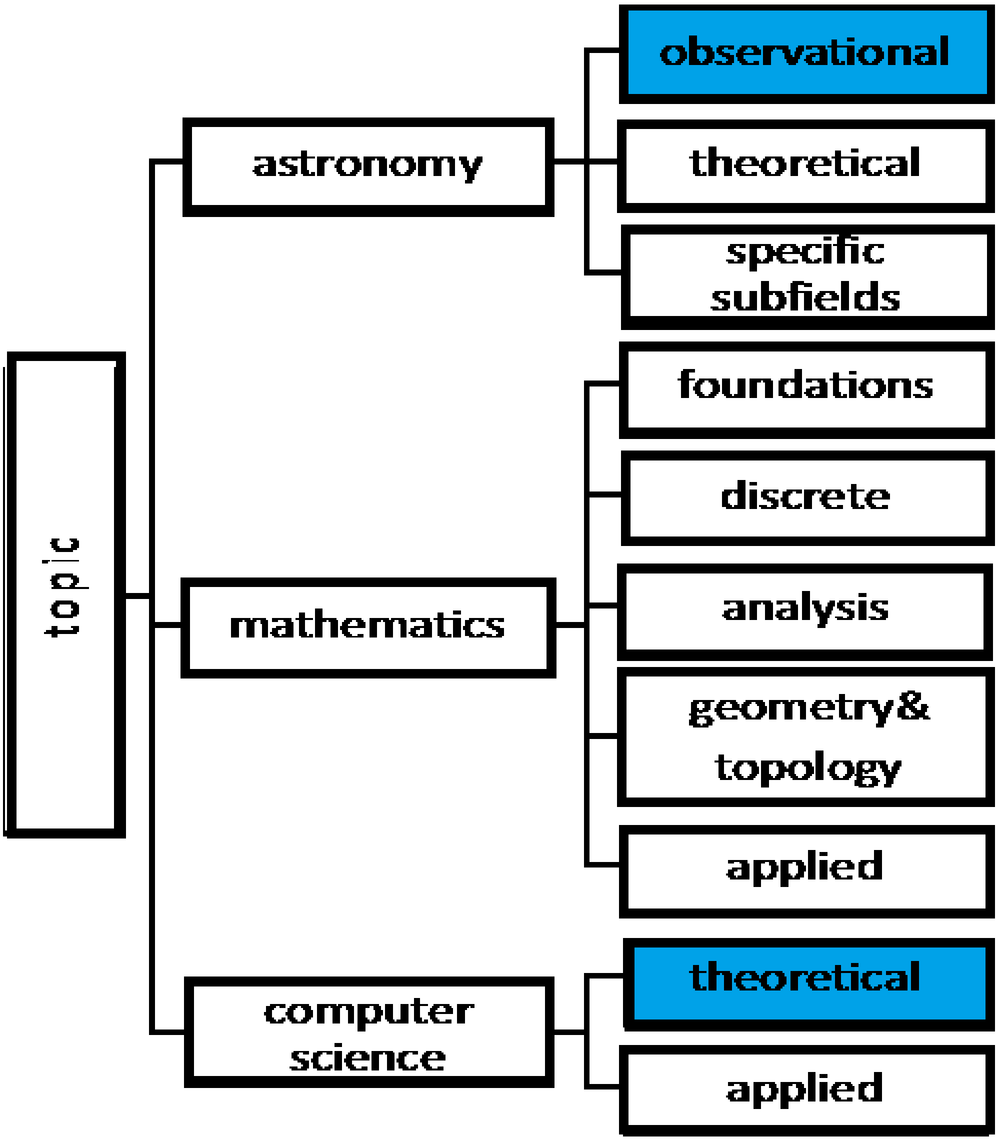

Figure 3

4.2 Distance between documents. A convenient method for formulating measures of the distance of two documents is defining document profiles based on the n-grams that appear in them and their relative frequencies [11,13]. After picking a value L for the profile length, the distance between the documents is measured by some of the dissimilarity functions of their profiles.

Definition 4.3 *Let a sequence of tokens $S = (s_1, s_2, \dots, s_{N+n-1})$ over an alphabet A be given, where N and n are positive integers. An n-gram of the sequence S is any n-long subsequence of consecutive tokens of S.*

It can be seen from the definition that the sequence *S* contains *N* n-grams.

Definition 4.4 *Let there be given a document $D = (s_1, s_2, \dots, s_{N+n-1})$ and an n-gram x of that document. The relative frequency of the n-gram x in the document D is the quotient $f_D(x) = \frac{p_D(x)}{N}$, where $p_D(x)$ is the number of appearances of x in D.*

Obviously, the values $f_D(x)$ lie in the interval [0,1].

Definition 4.5 *For a given document* $D = (s_1, s_2, \dots, s_{N+n-1})$ *and positive integers n and L, the profile of the document D is the set of ordered pairs of L most frequent n-grams of the document D and their relative frequencies, i.e.*

$$P_D = \big((x_1, f_1), (x_2, f_2), \dots, (x_n, f_n)\big)$$

where $f_i = f_D(x_i)$.

Definition 4.6 *Let **P** be a set of profiles. A dissimilarity measure d of the profiles of the set **P** is a function that assigns a nonnegative real number to each pair of profiles*$P_1, P_2 \in P \times P$.

The function d is not unique, but it's natural to require it to satisfy the following conditions:

- $d(P, P) = 0$ for all profiles $P \in \boldsymbol{P}$,
- $d(P_1, P_2) = d(P_2, P_1)$ for any two profiles $P_1, P_2 \in P$,
- If P_1 and P_2 are similar, $d(P_1, P_2)$ is small,
- If P_1 and P_2 are nor similar, $d(P_1, P_2)$ is large

The last two conditions are informal, since the notions "similar", "small" and "large" don't have precise definitions.

Tomović, Janičić and Kešelj [22] define 19 different measures that satisfy those conditions. In practice, the best results were shown by the measure

$$d_1(P_1, P_2) = \sum_{x \in P_1 \cup P_2} \left(\frac{2(f_1(x) - f_2(x))}{f_1(x) + f_2(x)}\right)^2$$

where $f_1(x)$ and $f_2(x)$ are the relative frequencies of an n-gram *x* in profiles P_1 and P_2, respectively.

Another interesting dissimilarity measure, which also showed good results in practice, was defined by Graovac [11]. Here the profiles are viewed only as sets of n-grams (without the relative frequencies) and the dissimilarity of two profiles is defined as the cardinal number of their symmetrical difference:

$$dSymmDif(P_1, P_2) = |P_1 \triangle P_2|$$

For the sum in the definition of measure d_1, the maximal possible number of addends is equal to the sum of the cardinal numbers of the profiles$P_1$and P_2. This number of addends will appear when comparing disjoint profiles. Also, we will get the maximal value of an addend in the sum if the n-gram *x* only belongs to one of the profiles. That means that one of the numbers$f_1(x)$ and $f_2(x)$ equals zero, and the other one will disappear when reducing the fraction, so that $2^2 = 4$ remains as the value of the addend. So, the maximal value of the function (1) is $4(|P_1| + |P_2|)$. If we assume that L is the largest possible profile cardinality, then the value of the measure (1) is not larger than 4(*L*+*L*) = 8*L*.

The function *dSymmDif* for a given profile length *L* reaches its maximal value if the symmetrical difference has the largest possible cardinality, i.e. if the profiles P_1 and P_2 are disjoint. Then the cardinal number of the symmetrical difference equals the cardinal number of their (disjoint) union, i.e. *2L*.

Now we can define the normalized versions of the measures d_1 and $dSymmDif$:

$$d_1n(P_1, P_2) = \frac{\sum_{x \in P_1 \cup P_2} \left(\frac{2(f_1(x) - f_2(x))}{f_1(x) + f_2(x)}\right)^2}{8L}$$

$$dSymmDifn = \frac{|P_1 \triangle P_2|}{2\mathrm{L}}$$

4.3 Distance between document and class. A document class is formed by concatenating a number of documents chosen on the basis of some of their common property. Since the class is itself a document, for calculating the distance of a document from a class we can use some of the earlier formulas from this chapter. In the next example we will use the distance formula $d_1(P_d, P_C) = \sum_{x \in P_1 \cup P_2} \left(\frac{2(f_d(x) - f_C(x))}{f_d(x) + f_C(x)}\right)^2$, where P_d and $f_d(x)$ (P_C and $f_C(x)$) denote, respectively, the profile of a document (class) and the relative frequency of an n-gram x in P_d (P_C).

4.4 Example: Ebart-hier corpus. The Ebart-hier corpus consists of eight document classes, grouped into four superclasses [14]. The results of the application of the measure d_1 on 16 randomly selected test documents (two from each class) are displayed in Tables 7, 8 and 9. The second column denotes the document's true class and the first - the document's number inside that class. The next four columns denote the distance of the document from the predefined superclasses (Table 7) and classes of documents (Tables 8 and 9). The boldface values represent the classification errors, i.e. values of the function $d_1(d, C)$, where C is a class different from the document's true class.

Doc	**Class**	C_1	C_2	C_3	C_4
1	C_{11}	197526.30	197686.67	197627.90	198222.53
3	C_{11}	**196679.83**	196580.65	197044.30	197882.69
5	C_{12}	199469.46	199490.48	**199414.84**	199858.13
100	C_{12}	197869.65	**197754.89**	198027.31	198427.90
4053	C_{21}	199284.71	199180.08	199349.88	199591.72
5555	C_{21}	199445.43	199340.71	199519.49	199657.91
380	C_{22}	199190.39	199076.64	199261.42	199633.64
77	C_{22}	193915.31	193906.35	194381.44	195164.80
500	C_{31}	195383.32	195782.36	195265.27	196432.24
4048	C_{31}	196216.48	196626.13	195664.58	197927.84
23	C_{32}	199885.21	199888.23	199877.48	199943.26
1	C_{32}	191629.43	191803.30	191431.15	193869.44
5	C_{41}	199375.44	199392.59	199412.21	199155.26
6	C_{41}	199257.12	199428.24	199295.36	197476.65
56	C_{42}	200038.47	200044.43	200035.24	200001.77
4	C_{42}	192565.55	192747.60	192710.45	191797.76

Table 7. Distances from superclasses C_1, C_2, C_3 and C_4

Doc	**Class**	C_{11}	C_{12}	C_{21}	C_{22}
1	C_{11}	197579.12	**197541.09**	197688.53	197774.46
3	C_{11}	196730.35	196742.50	**196580.77**	196909.91
5	C_{12}	199552.99	199420.07	199492.61	199724.13
100	C_{12}	197941.61	197798.84	**197749.02**	197974.85

4053	C_{21}	199306.98	199273.85	199177.70	199306.59
5555	C_{21}	199450.48	199448.20	199339.49	199408.84
380	C_{22}	199191.41	199202.95	199099.62	199063.89
77	C_{22}	193999.47	193976.00	193943.74	193929.73
500	C_{31}	195631.93	195305.76	195795.35	196114.68
4048	C_{31}	196430.36	196115.95	196634.45	196960.51
23	C_{32}	199880.44	199890.95	199888.11	199899.17
1	C_{32}	192218.81	**191249.30**	**191848.01**	**192029.40**
5	C_{41}	199393.54	199386.15	199394.43	199567.78
6	C_{41}	199405.28	199265.01	199418.94	199676.24
56	C_{42}	200042.79	200038.20	200040.32	200066.71
4	C_{42}	192761.71	192481.44	192740.83	193397.96

Table 8. Distances from classes C_{11}, C_{12}, C_{21} and C_{22}

Doc	**Class**	C_{31}	C_{32}	C_{41}	C_{42}
1	C_{11}	197629.63	198228.18	198944.91	198381.67
3	C_{11}	197049.71	198042.98	198929.61	198156.11
5	C_{12}	199422.67	200078.38	200367.89	199960.59
100	C_{12}	198031.16	198403.21	199275.78	198493.91
4053	C_{21}	199354.24	199586.94	199885.08	199645.15
5555	C_{21}	199520.16	199575.54	199890.79	199708.15
380	C_{22}	199265.68	199496.35	199948.70	199689.00
77	C_{22}	194396.00	195309.64	196520.87	195414.44
500	C_{31}	195266.54	196912.91	197898.93	196717.95
4048	C_{31}	195663.88	197742.70	199460.37	198305.71
23	C_{32}	199877.95	199876.40	200023.12	199945.91
1	C_{32}	**191457.52**	192159.19	197062.60	194345.89
5	C_{41}	199420.56	199824.41	199293.81	199371.69
6	C_{41}	199304.47	200517.90	197447.58	199435.99
56	C_{42}	200035.31	200133.96	200145.36	200006.86
4	C_{42}	192713.07	195050.54	195158.10	191996.76

Table 9. Distances from classes C_{31}, C_{32}, C_{41} and C_{42}

It can be seen from the tables that 11 of 16 documents were successfully classified, which constitutes an accuracy of around 70%. If we ignore the superclasses, i.e. if we view the set of eight classes as a flat corpus, the accuracy of the classification would be 75%. For two of the documents the superclass was correctly assigned, but not the class itself. For one of those two documents the assigned class belongs to the document's correct superclass.

This classification process was performed for fixed values *n=4* and *L=50000*. Also, the Ebart-hier corpus represents only a small part of the larger Ebart corpus and some classes have profile cardinalities significantly smaller than 50000. Taking into account the entire corpus and training the values n and L would lead to a more precise classification. For example, an experiment performed on the Ebart-hier corpus using the SSVM classification method, for different n-gram length values (between n=2 and n=7) and various tf-idf measures of the significance of a particular n-gram in the document, had the success rate (in the F1 measure) mostly around 89%, with the best result (90.43%) being achieved for flat classification with n-gram length $n=6$ and *boolean1* tf-idf measure [14].

5 Closeness measures between documents and classes

5.1 Fuzzy classification of documents in the document database. As we said in the first chapter, the fuzzy set theory allows a partial membership of an element to a set, usually measured by the real numbers from the interval [0,1]. Therefore, to measure a document's membership in a class, we need a function whose codomain would be the mentioned interval, where the values near 0 would denote weak membership and the values close to 1 strong membership. This function can be easily obtained from the functions of the distance between a document and a class from the preceding chapter, with the assumption that the membership of a document to a class increases as the distance between them decreases. If d is a normalized measure of the distance between a document and a class, a simple function that fulfills this condition is $c=1\text{-}d$.

Generally, if d is a normalized function of the distance between two documents, a document and a class, or between two classes, the closeness function should satisfy the following conditions:

- $c \to 0$ if $d \to 1$
- $c \to 1$ if $d \to 0$

The mentioned function c=1−d is the simplest one that satisfies these conditions, but it is possible to define an infinite number of other functions with the same properties (for example, $c = (1-p)^d, p > 0$. Depending on the case at hand, it is possible that one of these other functions would be a more convenient choice.

5.2 Closeness measure for classes.

Definition 5.1 *Let a set of classes* $C = \{C_1, C_2, \dots, C_n\}$ *be given. If* $d(C_i, C_j)$ *is a distance measure between classes* C_i *and* C_j*, then by the closeness of these classes we will refer to the value*

$$c(C_i, C_j) = 1 - d(C_i, C_j), i, j = 1,2, \dots, n$$

Example 5.1 *For pairs of classes from the example in the preceding chapter we get the following closeness measure values:*

c(Discrete Maths, Applied Maths)=1-0.5=0.5
c(Observational Astronomy, Theoretical Computer Science)=0.

5.3 Closeness measure of document and class. Fuzzy document classification.

Definition 5.2 *Let a document d and a set of classes* $C = \{C_1, C_2, \dots, C_n\}$ *be given. By fuzzy C-classification of document d, we will denote the set of ordered pairs*

$$c_C(d) = \{\left(C_1, \mu_{C_1}(d)\right), \left(C_2, \mu_{C_2}(d)\right), \dots, \left(C_n, \mu_{C_n}(d)\right)\}$$

where $\mu_{C_i}(d)$ is the membership degree of the document d to the class C_i.

The degree of membership of the document d to the class $C_i \in C$ can be calculated from any distance measure between a document and a class. The simplest way to do this is to use a normalized measure, whose value we subtract from 1. For example, for the distance d_1 the membership degree of the document d to the class C_i would be $\mu_{C_i}(d) = 1 - d_1n(d, C_i)$. However, the next example shows that this formula is not good enough.

Example 5.2 *For the (correctly classified) document 4053 of the class 2.1 we get the following values of the measure (1):*

$$d_1(d, C_{11}) = 199306.98$$
$$d_1(d, C_{12}) = 199273.85$$
$$d_1(d, C_{21}) = 199177.70$$
$$d_1(d, C_{22}) = 199306.59$$
$$d_1(d, C_{31}) = 199354.24$$
$$d_1(d, C_{32}) = 199586.94$$
$$d_1(d, C_{41}) = 199885.08$$
$$d_1(d, C_{42}) = 199645.15$$

Using the previous formula, we get the following membership degrees:

$$\mu_{C_{11}}(d) = 1 - \frac{199306.98}{400000} = 0.5017$$
$$\mu_{C_{12}}(d) = 0.5018$$
$$\mu_{C_{21}}(d) = 0.5021$$
$$\mu_{C_{22}}(d) = 0.5017$$
$$\mu_{C_{31}}(d) = 0.5016$$
$$\mu_{C_{32}}(d) = 0.5010$$
$$\mu_{C_{41}}(d) = 0.5003$$
$$\mu_{C_{42}}(d) = 0.5009$$

The required conditions that all the values belong to the interval [0,1] and that the highest membership degree is reached precisely for the document's true class (or, in general, to the class whose distance d_1 from the document has the smallest value) are fulfilled. However, all values are very close to one another and start to differ only in the third decimal place. Thus, the informal but still important condition that the membership to the true class is close to 1 and the membership to distant classes close to 0 is not fulfilled.

To get more useful values of the measure μ_{C_i}, we will assume that a document has to have full membership to at least one of the classes. That, of course, will be the class whose distance from the document d is the smallest. Let M denote the maximal value of the distance measure of a document d from the classes of the set $C = \{C_1, C_2, \dots, C_n\}$, i.e. $M = \max_{i=\overline{1,n}} d_1(d, C_i)$. Similarly, let $m = \min_{i=\overline{1,n}} d_1(d, C_i)$. Now, by the membership degree of the document d to the class Ci of the set C we will denote the value

$$\mu_{C_i}(d) = \frac{M - d_1(d, C_i)}{M - m}.$$

It can be seen from the definition that this measure has values in the interval [0, 1]. Also, the membership degree to the closest class from the set C will be 1 (because$d_1(d, C_i) = m$), and to the farthest - 0.

Example 5.3 *For the test document 4053 of the class 2.1* $M = d_1(d, C_{41}) = 199885.08$ *and m=199177.70. We get the following values of the measure μ:*

$$\mu_{C_{11}}(d) = \frac{M - d_1(d, C_{11})}{M - m} = 0.82$$

$$\mu_{C_{12}}(d) = 0.86$$

$$\mu_{C_{21}}(d) = 1.00$$

$$\mu_{C_{22}}(d) = 0.82$$

$$\mu_{C_{31}}(d) = 0.75$$

$$\mu_{C_{32}}(d) = 0.42$$

$$\mu_{C_{41}}(d) = 0.00$$

$$\mu_{C_{42}}(d) = 0.34$$

So, for the class set $C = \{C_{11}, C_{12}, C_{21}, C_{22}, C_{31}, C_{32}, C_{41}, C_{42}\}$ *and the document 4053 of the test class 2.1 (denoted by d) we get the following fuzzy C-classification:*

$$c_C(d) = \{(C_{11}, 0.82), (C_{12}, 0.86), (C_{21}, 1.00), (C_{22}, 0.82), (C_{31}, 0.75), (C_{32}, 0.42), (C_{41}, 0.00), (C_{42}, 0.34)\}$$

In such a classification process an important role, aside from the individual values of the distance measure d_1, is played by the class set C itself. Considering, for example, some additional classes, i.e. by replacing the set C with a set $K \supset C$, it is possible that the minimal and maximal values of the measure d_1 would be changed. That would lead to the change of all the values of the measure μ, even for those classes that belong to the set C. Similar thing would happen, of course, by eliminating some of the classes.

In practice it can happen that the number of classes is very large, for example a few thousand. In that case, the document classifications would have very large cardinalities. There are two simple ways to reduce those cardinal numbers, if needed. The first is to choose a smaller number *k* (e.g. 4 or 5) and to consider for the classification only the *k* ordered pairs with the largest μ values. The other way is to choose a real number α∈[0,1] (e.g. α=0.7 or α=0.8) and to consider only ordered pairs for which μ≥α.

6 Framework for fuzzy classification of digitized documents of NCD library

The National Centre for Digitization digital library [28] currently contains approximately 4000 documents, divided into 12 communities: Arts and Humanities, Candidates thesis, CD Library, Croatian editions, etc. Some of these communities are further. Some of these communities are further divided into collections and subcollections (Books, Scientific Works,…), whose list can be obtained by clicking the community name on the front page. There are also the options of searching and displaying the documents by the author name, publication year, title and topic. However, this last option is not functional.

This makes searching the documents by topic significantly harder. If we would want, for example, to find all documents related to astronomy, we would have to search each community separately, with all its collections and sub-communities. However, even then we would be unable to obtain the list of all relevant documents in one place.

There are additional problems with the current hierarchical organization of the Library. The criterion for defining the communities is unclear. Some of them, for example, are determined by their topic (Arts and Humanities, Mathematical Sciences,...) and some by language(Macedonian Editions, Slovenian Editions,...). The number of documents in the communities varies significantly, from Macedonian Editions, which contains only one document, to Mathematical Sciences, which has over 2000 of them. Finally, the difference between a subcommunity and a collection is also unclear. For example, the community Mathematical Sciences contains Books as a subcommunity, while in Arts and Humanities Books are listed as one of the collections. Moreover, the only subcommunity of the Arts and Humanities community is called Collections.

If the digital library is to fulfill its purpose in a satisfying way, after scanning and storing documents we need to provide for their more efficient searching and a more convenient way to display them. We will briefly describe one of the possible ways to achieve this.

1. It is required to perform the automatic classification of the documents based on some chosen class hierarchy [15]. This procedure would be performed in a few distinct steps, most of which would be automated (converting the documents from the .pdf format into .txt with the help of an OCR program, n-gram extraction, training of the chosen classification method, etc.)

2. A crisp classification method that would use one of the previous formulas for the distance between documents and classes would also make possible the automated fuzzy classification, as described in the previous chapter.

3. Relational databases are a convenient technology for displaying the data about the documents and their searching. The *view* mechanism would provide different types of access to different users if required. For example, the data about the documents from the Internal Documents class could be completely hidden from the users without an eLibrary account.

4. The NCD documents database would be very simple. The most natural design would contain three tables (though other approaches are possible). All the columns would have simple numerical or textual domains, so it wouldn't be necessary to create a large number of additional user operators which are sometimes required if the system doesn't have a good domain support. The basic database structure is shown in Figure 4.

5. The table DOCUMENTS contains information about the documents themselves. Its primary key is the IDD# column, while other columns represent the types of communities or collections that we wish to preserve (language, document type,…), as well as all the other properties that are interesting enough to be represented in the database (for example, title, number of pages, publication year,…). The AUTHORS table, with the primary key IDA#, contains a minimum of information on the authors of the documents (first and last name). In an actual database, this table would probably have a few additional columns. These two tables are connected by the table AUTHORSHIP. Its only two columns, IDA# and IDD#, together constitute the primary key, while each of them separately is a foreign key relating it to the table DOCUMENTS (column IDD#) or

AUTHORS (IDA#).

Among others, the following queries over such a database would be possible:

1. List the titles of all astronomy books in Serbian.

```
SELECT title
FROM documents
WHERE type='book' AND language='Serbian'
AND category='Astronomy'
```

2. List the titles and authors of all doctoral dissertations in mathematics in the period between 1970 and 2000.

```
SELECT documents.title, authors.first_name, authors.last_name
FROM documents, authorship, authors
WHERE documents.idd#=authorship.idd#
AND authorship.ida#=authors.ida#
AND documents.type='doctoral disertation'
AND documents.category='mathematics'
AND documents.year BETWEEN 1970 AND 2000
```

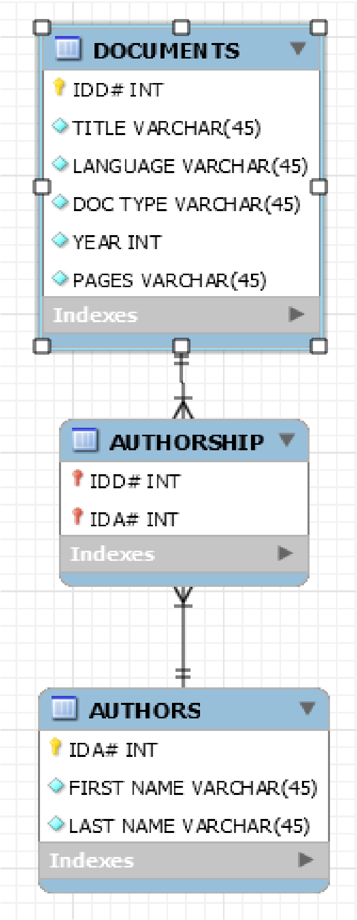

Figure 4

6.1 NCD example. Suppose that a digital library user is interested in older Serbian language popular science books about astronomy. We will show, in a very simplified way, how the previously described database design could be used to get a response to such a query.

Let the database contain the following elements:

- For documents, authors and the link between them – the previously described tables DOCUMENTS, AUTHORS and AUTHORSHIP, together with the fuzzy set *old*, defined over the column Year, which denotes the degree to which a book published in that year can be called "old";
- For the document classification – the CLASSIFICATION table, with the columns IDD#, Subject and Degree. The rows of this table show, for each particular document and each existing subject (mathematics, astronomy, etc.) the degree to which the document belongs to that subject;
- Fuzzy functions *classhigh* and *classlow*, defined over the column Degree, which show, for a given value of membership of the document to the corresponding subject, the degree in which such a membership could be called "high" or "low".

The required tables and operators, as well as the query, will be defined in the *Rel* software [29], which is an implementation of the *Tutorial D* relational language [9,10]. To use fuzzy querying over a larger database it would be best to use specialized fuzzy database software, with an implementation of one of the fuzzy extensions to a query language.

Rel has been in the development for approximately a decade, with the current version (at the time of this writing) being published in the mid-January 2018. Currently, it supports almost all features of the relational model, as described by Date and Darwen [10]. Most important to us is a very good support for the user-defined data types and operators (which constitute an extension of the original Codd's definition of the relational model). With these features, we can make very good approximations of many different kinds of fuzzy queries.

The following design is not optimal and its purpose is strictly the illustration of the query we wish to make. In a real application, each table would have additional columns, as well as a number of constraints over those columns, which won't be considered here. Moreover, to simplify the query, we will assume that the CLASSIFICATION table has a row for each (IDD#, Subject) combination, even if the membership of the document to that subject is zero. Of course, in a real application, such rows would not be in the table.

The content of the database is displayed in Tables 10, 11, 12 and 13. For this example, we filled the table with the real data from the NCD library. Only the fuzzy classification was arbitrarily chosen – in reality, it could be performed automatically.

IDD#	TITLE	PAGES	LANGUAGE	TYPE	YEAR
1	Sateliti i kosmicki brodovi	100	Serbian	Book	1965
2	Teorija polja	331	Serbian	Book	1952
3	Reseni problemi iz tenzorskog racuna…	350	Serbian	Book	1973
4	Mond – Modification of Newtonian Dynamics	173	English	Book	2017
5	Fiksne tacke preslikavanja	112	Serbian	Doctoral disertation	2012

Table 10. The DOCUMENTS table

IDA#	FIRST NAME	LAST NAME
1	Milivoj	Jugin
2	L	Landau
3	E	Lifsic
4	Marko	Leko
5	Milan	Plavsic
6	Veljko	Vujicic
7	Natasa	Babacev

Table 11. The AUTHORS table

IDD#	TOPIC	DEGREE
1	Astronomy	1.0
1	Physics	0.0
1	Mathematics	0.05
1	Mechanics	0.0
2	Astronomy	0.6
2	Physics	1.0

2	Mathematics	0.8
2	Mechanics	0.9
3	Astronomy	0.2
3	Physics	0.95
3	Mathematics	0.9
3	Mechanics	1.0
4	Astronomy	1.0
4	Physics	1.0
4	Mathematics	0.85
4	Mechanics	0.0
5	Astronomy	0.0
5	Physics	0.0
5	Mathematics	1.0
5	Mechanics	0.0

Table 12. The CLASSIFICATION table

IDD#	**IDA#**
1	1
2	2
2	3
3	4
3	5
4	6
5	7

Table 13. The AUTHORSHIP table

Function *old*:

$$old(year) = \begin{cases} 1, year < 1970 \\ \frac{2000-year}{30}, 1970 \leq year \leq 2000 \\ 0, godina > 2000 \end{cases}$$

Functions *classhigh* and *classlow*:

$$classhigh(degree) = \begin{cases} 0, degree < 0.5 \\ \frac{degree-0.5}{0.3}, 0.5 \leq degree \leq 0.8 \\ 1, degree > 0.8 \end{cases}$$

$$classlow(degree) = \begin{cases} 1, degree < 0.2 \\ \frac{0.5-degree}{0.3}, 0.2 \leq degree \leq 0.5 \\ 0, degree > 0.5 \end{cases}$$

These functions are defined in Rel by way of the user-defined operators. As an example, we show only the definition of the function *old*:

```
OPERATOR old(year INTEGER) RETURNS RATIONAL;
case ;
when god < 1970 then return 1.0 ;
when god >= 2000 then return 0.0 ;
else return CAST_AS_RATIONAL ( 2000 - year ) / 30.0 ;
end case ;
end operator ;
```

Now we can formulate the corresponding Tutorial D query. The essence of the popular science literature is to make its subject accessible to an average reader. Therefore, such books should contain as few mathematical formulas and calculus as possible. Having this in mind, the required condition could be interpreted as a request to list all books whose classification into Astronomy is high and into Mathematics and Physics low. We will also list the name(s) of the author(s) of the book and the degree to which the book satisfies the query requirements.

```
(1) with (K1:= CLASSIFICATION rename {SUBJECT as SUBJECT1, DEGREE as ASTRODEG},
(2) K2:= CLASSIFICATION rename {SUBJECT as SUBJECT2, DEGREE as          MATHDEG},
(3): K3:= CLASSIFICATION rename {SUBJECT as SUBJECT3, DEGREE as         PHYSDEG},
(4) T1:=K1 join K2 join K3,
(5) T2:= T1 where SUBJECT1='Astronomy' and SUBJECT2='Mathematics' and SUBJECT3='Physics',
(6) T3:= T2 {IDD#, ASTRODEG, MATHDEG, PHYSDEG},
(7) T4:=extend SUBJECT3: {ASTROHIGH:= classhigh(ASTRODEG),     MATHLOW:=class_low(MATHDEG),
         PHYSLOW:=classlow(PHYSDEG)},
(8) T5:= T4 {IDD#, ASTROHIGH, MATHLOW, PHYSLOW},
(9) T6:= T5 where ASTROHIGH>0.0 and MATHLOW>0.0 and PHYSLOW>0.0,
(10) D1:= DOCUMENTS{IDD#, TITLE, LANGUAGE, TYPE, YEAR},
(11) D2:= extend D1: {OLD_YEAR:=old(YEAR)},
(12) D3:= D2 {IDD#, TITLE, LANGUAGE, TYPE, OLD_GODINA},
(13) D4:= D3 where LANGUAGE='Serbian' and TYPE='book' and OLD_YEAR>0.0,
(14) ALL1:=D4 join T6 join AUTHORSHIP join AUTHORS,
(15) ALL2:= ALL1 {TITLE, OLD_YEAR, ASTROHIGH, MATHLOW, PHYSLOW, FIRST_NAME, LAST_NAME},
(16) ALL3:= extend ALL2: {DEG:=min{OLD_YEAR, ASTROHIGH, MATHLOW, PHYSLOW}}):
(17) ALL3 {TITLE, FIRST NAME, LAST NAME, DEGREE}
```

In this query, we used the ‘*with’* statement for the naming of the results of the application of various operations over other tables. Each assignment operator (:=) gives a name to a new table (on the left side of the assignment), which is the result of the application of operations over the tables on the right side. In the first 9 rows we created a table T6, which contains the identifiers of the documents with high Astronomy membership (ASTROHIGH>0.0) and low Mathematics and Physics membership (MATHLOW>0.0, PHYSLOW>0.0). Then (steps 10-13), from the DOCUMENTS table we picked those documents that satisfy the conditions for language (Serbian) and document type (book), as well as for the age of the document, by using the operator *old*. Finally (steps 14-15), we performed a natural join of these tables with the tables AUTHORSHIP and AUTHORS, to collect the required data on the first and last name of the book author(s). The penultimate row (16) calculates the overall satisfaction degree of the query. Since it is represented as a conjunction of conditions, some of which are fuzzy, we used the *min* function, as the simplest measure of the truth value of fuzzy conjunction. The last row (17) projects the table on the required columns (book title, the first and last name of the author, and the degree of membership of the book to the popular science category). The user who wishes to obtain more detailed data, for example on the degree of fulfillment of each individual condition, can simply leave out this final projection.

The query result is a singleton and it’s displayed in Table 14.

TITLE	FIRST NAME	LAST NAME	DEGREE
Sateliti i kosmicki brodovi	Milivoj	Jugin	1.0

Table 14. The query result

Rel is a program that looks very convenient for the implementation of fuzzy concepts into the traditional databases (in the absence of specialized software), because of the firm theoretical foundation on which rests its query language, as well as a good support to the user-defined data types and operators. However, from the technical point of view, there is much left to do to make it useful for the needs of the NCD library. Currently, its main purpose is teaching the concepts of the (object-) relational databases, while technical details are of secondary importance. So far it doesn’t have any kind of query optimization, except for some simple cases of the natural join operation. Further, there are no user applications which would enable the interactive object creation and querying. Also, ODBC drivers that would provide the possibility of access to the databases through some server language like PHP currently don’t exist.

7 Conclusion

In this paper, we presented a possible method of fuzzy classification of text documents. Fuzzy classification of a document, relative to a set of *n* predefined classes, is a set of *n* ordered pairs that give information about its membership degree to each individual class.

The membership degrees to individual classes are calculated by using a simple formula on the values of some of the dissimilarity measures [11,13].

After that, using a small subset of the digitized documents from the NCD library [28], we have shown some advantages of such a classification. With a convenient document database, fuzzy classification enables more complex and informative queries.

We also briefly listed the facilities needed for creating and populating such databases.

References

[1] M. Anvari and G. F. Rose, *Fuzzy relational databases*, in J. C. Bezdek, ed., Analysis of Fuzzy Information, Taylor Francis, Boca Raton, 1987, 203–212.

[2] Michal Baczinsky and Balasubramaniam Yajaram, *Fuzzy Implications*, Springer, 2008.

[3] J. F. Baldwin and S. Q. Zhou, *A fuzzy relational inference language*, Fuzzy Sets and Systems, 14 (1984), 155–174.

[4] H. Blockeel, M. Bruynooghe, S. Džeroski, J. Ramon and J. Struyf, *Hierarchical multi-classification*, Proceedings of the ACM SIGKDD 2002 workshop on multi-relational data mining *(MRDM 2002)*, 2002, 21–35.

[5] P. Bosc and O. Pivert, *SQLf: a relational database language for fuzzy querying*, IEEE Transactions on Fuzzy Systems, 3 (1995), 1–17.

[6] Billy P. Buckles and Frederick E. Petry, *A fuzzy representation of data for relational databases*, Fuzzy Sets and Systems, 7 (1982), 213–226.

[7] G. Q. Chen, J. Vandenbulcke, E. E. Kerre, *A step towards the theory of fuzzy relational database design*, in B. Lowen and M. Roubens, eds., Proceedings of IFSA'91 World Congress, 1991, 44–47.

[8] E. F. Codd, *A relational model of data for large shared data banks*,CACM, 13 (1970), 377–387.

[9] C. J. Date, *An Introduction to Database Systems*, Sixth Edition, Addison-Wesley, 1994.

[10] C. J. Date and Hugh Darwen, *Databases, Types and the Relational Model:* The Third Manifesto, Third Edition, Pearson, 2006.

[11] Jelena Graovac, *A variant of n-gram based language-independent text categorization*, Intelligent Data Analysis**,** 18 (2014), 677–695

[12] Etienne E. Kerre and Guoqing Chen, An *overview of fuzzy data models*, in P. Bosc and J. Kacprzyk, eds., Fuzziness in Database Management Systems, Springer-Verlag, 1995, 23–41.

[13] Vlado Kešelj, Fuchung Peng, Nick Cercone, and Calvin Thomas, *N-gram-based author profiles for authorship attribution*, Proceedings of the conference pacific association for computational linguistics*, PACLING* 3(2003), 255–264.

[14] Jovana Kovačević, Jelena Graovac, *Application of a structural support vector machine method to n-gram based text classification in Serbian*, INFOtheca, 16(2016), http://infoteka.bg.ac.rs/index.php/en/2016-1-2/infoteka-2016-16-1-2-1X

[15] Jovana Kovačević, Jelena Graovac, *Prospective automated hierarchical classification of digitized documents*, Pregled NCD, 29 (2016), 42–51

[16] Frederick E. Petry, *Fuzzy Databases: Principles and Applications*, Kluwer Academic Publishers, 1996.

[17] Henri Prade and Claudette Testemale, *Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries*, Information Sciences, 34 (1983), 115–143.

[18] P. Raghavan, C.D. Manning and H. Schutze, *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, 1(2008).

[19] E. A. Rundensteiner, L. W. Hawkes, W. Bandler, *On nearness measures in fuzzy relational data models,* International Journal of Approximate Reasoning*,* 3(1989), pp. 267–298.

[20] F. Sebastiani, *Machine learning in automated text categorization*, ACM Computing *Surveys (CSUR),* 34:1 (2002), 1–47.

[21] Sujeet Shenoi and Austin Melton, *Proximity relations in the fuzzy relational database model*, Fuzzy Sets and Systems, 31 (1989), 285–296.

[22] Andrija Tomović, Predrag Janičić, Vlado Kešelj, *N-gram based classification and unsupervised hierarchical clustering of genome sequences,* Computer Methods and Programs in Biomedicine, 81 (2006), 137–153

[23] M. Umano, *Retrieval from fuzzy databases by fuzzy relational algebra*, in Sanchez and Gupta, eds., Fuzzy Information Knowledge Representation and Decision Analysis, Pergamon Press, 1983, 1–6

[24] Adnan Yazici and Roy George, *Fuzzy Database Modeling*, Springer-Verlag Berlin Heidelberg, 1999.

[25] L. A. Zadeh*, Fuzzy set*s, Information and Control, 8 (1965), 338–353

[26] M. Zemankova and A. Kandel, *Fuzzy Relational Database - A Key to Expert Systems*, Verlag TUV Rheinland, 1984.

[27] A. Zvieli, *A fuzzy relational calculus*, in L. Kerschberg, ed., Expert Database Systems, Benjamin Cummings Pub. Co., 1986, 311–326.

[28] http://elibrary.matf.bg.ac.rs/

[29] http://reldb.org

aleksandar.janjic@unibl.rs