Finding an appropriate generalization for a fuzzy thematic set in taxonomy

Dmitry Frolov
Dept. of Data Analysis and AI,
National Research University
"Higher School of Economics"
Moscow, Russian Federation
dmitsf@gmail.com

Boris Mirkin
Dept. of Data Analysis and AI,
National Research University
"Higher School of Economics"
Moscow, Russian Federation
Dept. of Computer Science,
Birkbeck University of London UK
bmirkin@hse.ru

Susana Nascimento
Dept. of CS and NOVA LINCS
Universidade Nova de Lisboa
Caparica, Portugal
snt@fct.unl.pt

Trevor Fenner
Dept. of Computer Science
Birkbeck University of London
London, UK
trevor@dcs.bbk.ac.uk

Abstract

This paper proposes a novel method, referred to as ParGenFS, for finding a most specific generalization of a query set, represented by a fuzzy set of topics assigned to leaves of the rooted tree of a taxonomy. This generalization lifts the query set to one or several "head subjects" in the higher ranks of the taxonomy. The head subject is supposed to "tightly" cover the query set, however dispersed that can be over branches of the tree, possibly bringing in some "gaps", that are taxonomy nodes covered by the head subject but irrelevant to the set. To balance that, we admit some "offshoots", that are nodes belonging to the query set but not covered by the head subject. The method globally minimizes the total number of head subjects and gaps and offshoots, differently weighted. Our algorithm is applied to the structural analysis and description of a collection of 17685 abstracts of research papers published in 17 Springer journals on data science for the 20-years period 1998-2017. Our taxonomy of Data Science (DST) is extracted from the Association of Computing Machinery Classification of Computing Subjects 2012 (ACM-CCS), a six-layer hierarchical taxonomy manually developed by a team of ACM experts. The DST also involves a number of additions detailing the leaves of the ACM-CCS taxonomy and added by ourselves. We find fuzzy clusters of leaf topics over the text collection, with a specially developed machinery. Three of the clusters are thematic indeed, relating to Data Science sub-areas: (a) learning, (b) information retrieval, and (c) clustering. These three clusters are lifted with ParGenFS in the DST, which allows us to make some conclusions of the tendencies of the developments in these areas.

Keywords: hierarchical taxonomy, parsimony, generalization, additive fuzzy cluster, spectral clustering, annotated suffix tree

Contents

1 Introduction

The issue of automation of structurization and interpretation of digital text collections is of ever-growing importance because of both practical needs and theoretical necessity. This paper concerns an aspect of this, the issue of generalization as a unique feature of human cognitive abilities. We use this term in the meaning pointed out by the Merriam-Webster dictionary, which defines "to generalize" as "to give a general form to" (1) or "to derive or induce (a general conception or principle) from particulars" (2a) (see https://www.merriam-webster.com/, last visited 28 November 2018).

The existing approaches to computational analysis of structure of text collections usually involve no generalization as a specific aim. The most popular tools for structuring text collections are cluster analysis and topic modelling. Both involve features of the same level of granularity as individual words or short phrases in the texts, thus no generalization as an explicitely stated goal.

Nevertheless, the hierarchical nature of the universe of meanings is reflected in the flow of publications on text analysis. We can distinguish between at least three directions at which the matter of generalization is addressed. First of all, one should mention activities related to developing taxonomies, especially those involving hyponymic/hypernymic relations (see, for example, [38, 39, 45], and references therein). A recent paper [42] should be mentioned here too, as that devoted to supplementing a taxonomy with newly emerging research topics.

Another direction is part of conventional activities in text summarization. Usually, summaries are created using a rather mechanistic approach of sentence extraction. There is, however, also an approach for building summaries as abstractions of texts by combining some templates such as subject-verb-object (SVO) triplets (see, for example, [17, 27]).

Yet one more field of activities is what can be referred to as operational generalization. In this direction, the authors use generalized case descriptions involving taxonomic relations between generalized states and their parts to achieve a tangible goal such as improving characteristics of text retrieval (see, for example, [26] and [44].)

This paper falls in neither of these approaches, as we do not attempt to change any taxonomy. We rather try to use a taxonomy for straightforwardly implementing the idea of generalization. According to the Merriam-Webster dictionary cited above, the term "generalization" refers to deriving a general conception from particulars. We assume that a most straightforward medium for such a derivation, a taxonomy of the field, is given to us. A taxonomy, in this setting, is a rooted tree over key concepts of a domain tagging the tree nodes. The tree bears a main hyponymic/hypernymic relation in the domain, so that an A-tagged node is the parent of a B-tagged node if relation “B is an A” is true. The situation of our concern is a case at which we are to generalize a fuzzy set of taxonomy leaves representing the essence of some empirically observed phenomenon. Specifically, one may be interested in patterns of novel research in a domain like Data Science. Data Science is a newly emerging area of Computer Science. The most popular Computer Science taxonomy is manually developed by the world-wide Association for Computing Machinery, a most representative body in the domain; the latest release of the taxonomy has been published in 2012 as the ACM Computing Classification System (ACM-CCS) [1]. We take its part related to Data Science, as presented in a slightly modified form by adding a few leaves in [24]. We add a few more leaves to better reflect the research papers being analyzed, and present it in the Appendix.

As an evidence of the research being conducted in Data Science, we take a collection of research papers published for a recent period in a set of journals representative of the domain of our concern. We are going to extract tight clusters of ACM-CCS topics according to the collection, each representing core tendencies of the development of the domain as reflected in the collection. It should be expected that the clusters are fuzzy, in accordance with the fuzzy nature of semantics. A major issue emerging then would be of interpretation of such a fuzzy cluster. Among various ways for addressing the challenge, we choose a most straightforward approach of generalization, prompted by the case illustrated in Figure 1 (a).

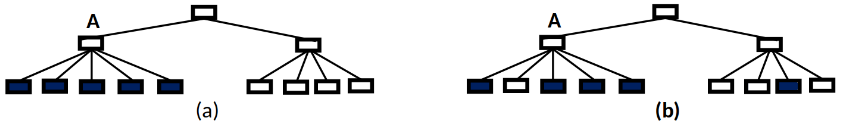


Figure 1: A taxonomy fragment with black boxes corresponding to a query set; a straightforward case (a) and a more complex case (b).

This Figure represents a taxonomy fragment with a leaf cluster fully covering all the children of the parental node A. Of course, the taxonomy concept assigned to A is a most natural generalization of the cluster in this case suggesting an interpretation of the cluster as all those topics that fall in the concept A. We are going to extend this approach to generalize less obvious cases such as that in Figure 1 (b) and see how this approach can be applied to real-world scenarios.

The rest of the paper is organized accordingly. Section 2 presents a mathematical formalization of the generalization problem as of parsimoniously lifting of a given query fuzzy leaf set to higher ranks of the taxonomy and provides a recursive algorithm leading to a globally optimal solution to the problem. Section 3 describes an application of this approach to deriving tendencies in development of the data science, that can be discerned from a set of about 18000 research papers published by the Springer Publishers in 17 journals related to data science for the past 20 years. Its subsections describe our approach to finding and generalizing fuzzy clusters of research topics. Specifically, the taxonomy of data science (DST) used in this paper is presented in Section 3a (see also Appendix); a method for developing a matrix of topic-to-paper relevance values based on automated processing of the text collection is described in Section 3b; Section 3c describes a spectral method for finding fuzzy clusters of research topics from DST using the relevance data; Section 3d presents results of lifting of three most homogeneous clusters out of six found in the DST taxonomy; and Section 3e presents our conclusions on the tendencies in the development of the corresponding parts of Data Science drawn from the lifting results. Section 4 concludes the paper.

2 Parsimoniously lifting a fuzzy thematic cluster in a taxonomy: model and method

2.1 Statement of the problem

Mathematically, a taxonomy is a rooted tree whose nodes are annotated by taxonomy topics. We consider the following problem. Given a fuzzy set S of taxonomy leaves, find a node $t(S)$ of higher rank in the taxonomy, that covers the set S as tight as possible. Such a "lifting" problem is a mathematical explication of the human facility for generalization, that is, "the process of forming aconceptualform" of a phenomenon represented, in this case, by a fuzzy leaf subset.

The problem is not as simple as it may seem to be. Consider, for the sake of simplicity, a hard set S shown with five black leaf boxes on a fragment of a tree in Figure 1 (b). Figure 2 illustrates the situation at which the set of black boxes is lifted to the root, which is shown by blackening the root box, and its offspring, too. If we accept that set S may be generalized by the root, this would lead to a number, four, white boxes to be covered by the root and, thus, in this way, falling in the same concept as S even as they do not belong in S. Such a situation will be referred to as a gap. Lifting with gaps should be penalized. Altogether, the number of conceptual elements introduced to generalize S here is 1 head subject, that is, the root to which we have assigned S, and the 4 gaps occurred just because of the topology of the tree, which imposes this penalty. Another lifting decision is illustrated in Figure 3: here the set is lifted just to the root of the left branch of the tree. We can see that the number of gaps has drastically decreased, to just 1. However, another oddity emerged: a black box on the right, belonging to S but not covered by the root of the left branch at which the set S is mapped. This type of error will be referred to as an offshoot. At this lifting, three new items emerge: one head subject, one offshoot, and one gap. This is less than the number of items emerged at lifting the set to the root (one head subject and four gaps, that is, five), which makes it more preferable. Of course, this conclusion holds only if the relative weight of an offshoot is less than the total relative weight of three gaps.

Therefore, at lifting a fuzzy leaf set to taxonomic higher ranks there may emerge two types of errors, gaps and offshoots. If one likens assigning a general concept to the process of classification then gaps correspond to false positives, and offshoots, to false negatives. Further on, we are going to formulate an algorithm for producing the most parsimonious lifting to involve arbitrary weights assigned to errors.

We are interested to see whether a fuzzy set S can be generalized by a node t from higher ranks of the taxonomy, so that S can be thought of as falling within the framework covered by the node t. If this is done straightforwardly, for example, by picking up the last common ancestor t of the nodes in S, the coverage of S by t would be unlikely as tight as it might be desirable. To make the general concept more specific, one should admit not only gaps, but also offshoots, that are leaves which do belong in S but are not covered by t. The goal of finding an interpretable pigeon-hole for S within the taxonomy can be formalized as that of finding one or more "head subjects" t to cover S with the minimum number of all the elements introduced at the generalization: head subjects, gaps, and offshoots. This goal realizes the principle of Maximum Parsimony (MP) in describing the phenomenon in question, and is popular in some domains such as Bioinformatics [33].

Consider a rooted tree T representing a hierarchical taxonomy so that its nodes are annotated with key phrases signifying various concepts. We denote the set of its *leaves* by I. The relationship between nodes in the hierarchy is conventionally expressed using genealogical terms: each node $t \in T$ is said to be the *parent* of the nodes immediately descending from t in T, its *children*. We use $\chi(t)$ to denote the set of children of t. Each *interior* node $t \in T - I$ is assumed to correspond to a concept that generalizes the topics corresponding to the leaves $I(t)$ descending from t, viz. the leaves of the subtree $T(t)$ rooted at t, which is conventionally referred to as the *leaf cluster of* t.

A *fuzzy set* on I is a mapping u of I to the non-negative real numbers that assigns a membership value, or support, $u(i) \geq 0$ to each $i \in I$. We refer to the set $S_u \subset I$, where $S_u = \{i \in I : u(i) > 0\}$, as the *base* of u. In general, no other assumptions are made about the function u, other than, for convenience, commonly limiting it to not exceed unity. Conventional, or *crisp*, sets correspond to binary membership functions u such that $u(i) = 1$ if $i \in S_u$ and $u(i) = 0$ otherwise.

Given a fuzzy query set u defined on the leaves I of the tree T, one can consider u to be a (possibly noisy) projection of a higher rank concept, u's "head subject", onto the corresponding leaf cluster. Under this assumption, there should exist a head subject node h among the interior nodes of the tree T such that its leaf cluster $I(h)$ more or less coincides (up to small errors) with S_u. This head subject is the generalization of u to be found. The two types of possible errors associated with the head subject if it does not cover the base precisely, are false positives and false negatives, referred to in this paper, as *gaps* and *offshoots*, respectively, are illustrated in Figures 2 and 3. Altogether, the total number of head subjects, gaps, and offshoots has to be as small as possible. To this end, we introduce a penalty for each of these elements. Assuming for the sake of simplicity, that the black box leaves on Figure 1 (b) have membership function values equal to unity, one can easily see that the total penalty at the head subject raised to the root (Figure 2) is equal to $1+4\gamma$ where 1 is the penalty for a head subject and γ, the penalty for a gap, since the lift on Figure 2 involves one head subject, the root, and four gaps, the blank box leaves. Similarly, the penalty for the lift on Figure 3 to the root of the left-side subtree is equal to $1+\gamma+\lambda$ where λ is the penalty for an offshoot, as there is one copy of each, head subject, gap, and offshoot, in Figure 3. Therefore, depending on the relationship between γ and λ either lift on Figure 2 or lift on Figure 3 is to be chosen. That will be the former, if $3\gamma < \lambda$, or the latter, if otherwise.

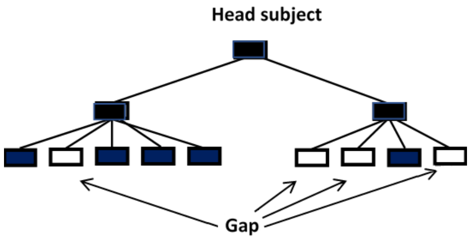


Figure 2: Generalization of the query set from Figure 1 (b) by mapping it to the root, with the price of four gaps emerged at the lift.

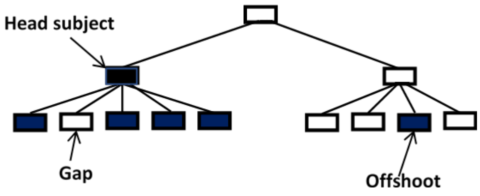


Figure 3: Generalization of the query set from Figure 1 (b) by mapping it to the root of the left branch, with the price of one gap and one offshoot emerged at this lift.

To properly define the concept of gap in general, let us first consider the *u-irrelevant* nodes in the tree T. A node $t \in T$ is referred to as *u-irrelevant* if its leaf-cluster $I(t)$ is disjoint from the base S_u. Obviously, if a node is u-irrelevant, all of its descendants are also u-irrelevant.

Consider a candidate node h in T and its meaning relative to fuzzy set u. An *h-gap* is a node g of $T(h)$, other than h, at which a *loss* of the meaning has occurred, that is, g is a maximal u-irrelevant node in the sense that its parent is not u-irrelevant. Conversely, establishing a node h as a head subject can be considered as a *gain* of the meaning of u at the node. The set of all h-gaps will be denoted by $G(h)$.

An *h-offshoot* is a leaf $i \in S_u$ which is not covered by h, i.e., $i \notin I(h)$. The set of all h-offshoots is $S_u - I(h)$.

Since no taxonomy perfectly reflects all the relevant real-world relations, some fuzzy topic sets u may refer to general concepts that are not captured in $T(h)$. In such a case, two or more head subjects may be needed to cover S_u accurately, rather than just one. This motivates the following definition.

Given a fuzzy topic set u over I, a set of nodes H will be referred to as a *u-cover* if: (a) H covers S_u, that is, $S_u \subseteq \bigcup_{h \in H} I(h)$, and (b) the nodes in H are unrelated, i.e. $I(h) \cap I(h') = \emptyset$ for all $h, h' \in H$ such that $h \neq h'$. The interior nodes of H will be referred to as *head subjects* and the leaf nodes as *offshoots*, so the set of offshoots in H is $H \cap I$. The set of *gaps* in H is the union of $G(h)$ over all head subjects $h \in H - I$.

We associate a penalty with H, so that only the most parsimonious generalizations are to be considered. A parsimonious generalization should have as small a number of head subjects, gaps and offshoots as possible. To reflect the relative importance of each of these, we use positive *penalty weights*, λ and γ, for gaps and offshoots, respectively. A head subject is assigned weight 1. The introduced concept of parsimonious generalization is a mathematical explication of the idea of most specific generalization.

Of course, the penalty value associated with a node $h \in H$ should take into account the u-membership values of all its offspring, not just the the leaf cluster $I(h)$: the smaller they are, the less their effect should be. Therefore, to properly define the overall penalty, we extend the u-membership values from I to all the nodes in T. The algorithm ParGenFS (Parsimonious Generalization of Fuzzy Sets) for a parsimonious generalization of u, described below, does not depend on the way the u-values are assigned, so any extension of the membership values to $u(t)$ for $t \in T - I$ is acceptable, provided the value of $u(t)$ is zero for all u-irrelevant nodes t. Although every gap is assigned with a membership value of zero, we may consider some gaps more important than others, depending on the membership values assigned to their parents. A gap is less significant if its parent's membership value is smaller. Therefore, a measure $v(g)$ of "gap importance" should also be defined, to be reflected in the penalty function. We suggest defining the *gap importance* as $v(g) = u(par(g))$, where $par(g)$ is the parent of g. An alternative definition would be to scale these values by dividing them by the number of children of $par(g)$. However, we note that the algorithm ParGenFS below works for any definition of gap importance.

We, therefore, define the penalty function $p(H)$ for a u-cover H as:

$$p(H) = \sum_{h \in H-I} u(h) + \sum_{h \in H-I} \sum_{g \in G(h)} \lambda v(g) + \sum_{h \in H \cap I} \gamma u(h). \tag{1}$$

2.2 Algorithm ParGenFS for finding a most specific generalization

The problem we address is to find a u-cover H that globally minimizes the penalty $p(H)$. Such a u-cover will be the parsimonious generalization of the query set u.

Before applying an algorithm to minimize the total penalty, one needs to execute a preliminary transformation of the tree by pruning it from all the non-maximal u-irrelevant nodes, i.e. descendants of gaps. Simultaneously, the sets of gaps $G(t)$ and the internal summary gap importance $V(t) = \sum_{g \in G(t)} v(g)$ in Eq. (1) can be computed for each interior node t. We note that the elements of S_u are in the leaf set of the pruned tree, and the other leaves of the pruned tree are precisely the gaps.

Assume that the tree T has already been pruned and all its nodes are annotated by the membership values $u(t)$. The sets $G(t)$, and the gap importance values, $v(t)$ and $V(t)$, are assigned as described above.

Now we can apply our lifting algorithm ParGenFS. For each node t, the algorithm ParGenFS computes two sets, $H(t)$ and $L(t)$, containing those nodes in $T(t)$ at which respectively gains and losses of head subjects occur (including offshoots). The associated penalty is computed as $p(t)$ described below.

An assumption of the algorithm is that no gain can happen after a loss. Therefore, $H(t)$ and $L(t)$ are defined assuming that the head subject has not been gained (nor therefore lost) at any of t's ancestors. The algorithm ParGenFS recursively computes $H(t)$, $L(t)$ and $p(t)$ from the corresponding values for the child nodes in $\chi(t)$.

Specifically, for each leaf node that is not in S_u, we set both $L(\cdot)$ and $H(\cdot)$ to be empty and the penalty to be zero. For each leaf node that is in S_u, $L(\cdot)$ is set to be empty, whereas $H(\cdot)$, to contain just the leaf node, and the penalty is defined as its membership value multiplied by the offshoot penalty weight γ.

To compute $L(t)$ and $H(t)$ for any interior node t, we analyze two possible cases: (a) when the head subject has been gained at t and (b) when the head subject has not been gained at t.

In case (a), the sets $H(\cdot)$ and $L(\cdot)$ at its children are not needed. In this case, $H(t)$, $L(t)$ and $p(t)$ are defined by:

$$\begin{aligned} H(t) &= \{t\} \\ L(t) &= G(t) \\ p(t) &= u(t) + \lambda V(t). \end{aligned} \tag{2}$$

In case (b), the sets $H(t)$ and $L(t)$ are just the unions of those of its children, and $p(t)$ is the sum of their penalties:

$$\begin{aligned} H(t) &= \bigcup_{w \in \chi(t)} H(w) \\ L(t) &= \bigcup_{w \in \chi(t)} L(w) \\ p(t) &= \sum_{w \in \chi(t)} p(w). \end{aligned} \tag{3}$$

To obtain a parsimonious lift, whichever case gives the smaller value of $p(t)$ is chosen.

When both cases give the same values for $p(t)$, we may choose arbitrarily – in the formulation of the algorithm below, we have chosen (a). The output of the algorithm consists of the values at the root, namely, H – the set of head subjects and offshoots, L – the set of gaps, and p – the associated penalty.

Algorithm ParGenFS

```
• INPUT: $u$, $T$

• OUTPUT: $H = H(root)$, $L = L(root)$, $p = p(root)$

I Base Case
  for each leaf $i \in I$
    if $u(i) > 0$
      $H(i) = \{i\}$
      $L(i) = \oslash$
      $p(i) = \gamma u(i)$
    else
      $H(i) = \emptyset$
      $L(i) = \emptyset$
      $p(i) = 0$
II Recursion
    if $u(t) + \lambda V(t) \leq \sum_{w \in \chi(t)} p(w)$
      $H(t) = \{t\}$
      $L(t) = G(t)$
      $p(t) = u(t) + \lambda V(t)$
    else
      $H(t) = \bigcup_{w \in \chi(t)} H(w)$
      $L(t) = \bigcup_{w \in \chi(t)} L(w)$
      $p(t) = \sum_{w \in \chi(t)} p(w)$
```

It is not difficult to see that the algorithm ParGenFS leads to an optimal lifting indeed, as stated in the following proposition.

Theorem 1. *Any u-cover H found by the algorithm ParGenFS is a (global) minimizer of the penalty p.*

Proof. We prove this result by induction over the number of nodes n in the tree. If $n = 1$, there is only one node i and, in the Base Case of ParGenFS, the definition of the sets $H(i)$ and $L(i)$ is such that the only possible nonempty set is $H(i) = \{i\}$, when $i \in S_u$. The penalty in this case is $\gamma u(i)$, which is clearly the correct, and minimum, penalty. When $i \notin S_u$, the penalty is obviously zero.

Let us now assume that the statement is true for all rooted trees with fewer than n nodes. Consider a rooted tree $T(t)$ with n nodes, where $n > 1$. Each child w of the root t is itself the root of a subtree $T(w)$ with fewer than n nodes.

If the head subject is not gained at t, then the optimal H- and L-sets at t are clearly the unions of the corresponding sets for the subtrees $T(w)$; this follows from the additive structure of the penalty function in (1). Clearly, the minimum penalty for the subtree $T(t)$ must be the smaller of the penalty values $p(t) = u(t)+\lambda V(t)$ and $p(t) = \sum_{w \in \chi(t)} p(w)$, as it is in the algorithm. The result now follows by induction on n. □

2.3 Illustrative examples

2.3.1 Computing a parsimonious lift (illustrative case)

Let us apply ParGenFS algorithm to the case presented in Fig. 4(a). This shows a three-layer tree whose nodes are labeled by letters A and B with extensions, and a fuzzy query set u having support $S_u = \{$A1, A2,B1, B2$\}$, with membership values shown in Fig. 4(b). The membership values are given in thousandths for convenience of reading and are normalized according to the quadratic condition defined further on in (15).

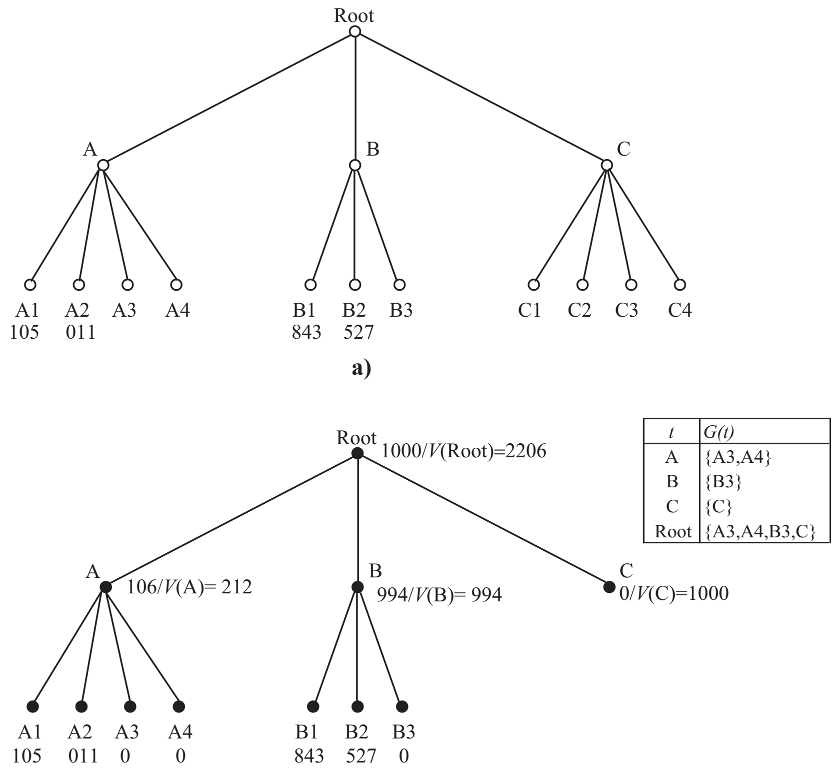
Fig. 4(b) shows the pruned tree, with the membership values extended to all nodes and the associated gap importance values $V(\cdot)$ shown at the higher rank nodes.

Tables 1 and 2 illustrate successive steps of the algorithm ParGenFS at the interior nodes A, B (Table 1) and the root (Table 2), with penalty weights $\lambda = 0.2$ and $\gamma = 0.9$. The computations leading to the smaller penalty values are highlighted in bold face.

Table 1: Computational results of ParGenFS at nodes A and B of the pruned tree for the fuzzy query set u in Fig. 4; the gap and offshoot penalty weights are $\lambda = 0.2$ and $\gamma = 0.9$.

i	$H(i)$	$L(i)$	$p(i)$	t		$H(t)$	$L(t)$	$p(t)$
$A1$	$\{A1\}$	$\emptyset$	$0.9 \times 0.105 = 0.095$		Gain	$\{A\}$	$\{A3, A4\}$	$0.106 + 0.2 \times 0.212 = 0.148$
$A2$	$\{A2\}$	$\emptyset$	$0.9 \times 0.0105 = 0.009$	A				
$A3$	$\emptyset$	$\emptyset$	0		**No Gain**	**{A1,A2}**	**{A3,A4}**	**0.095 + 0.009 = 0.104**
$A4$	$\emptyset$	$\emptyset$	0					
$B1$	$\{B1\}$	$\emptyset$	$0.9 \times 0.843 = 0.759$		**Gain**	**{B}**	**{B3}**	**0.994 + 0.2×0.994 = 1.193**
$B2$	$\{B2\}$	$\emptyset$	$0.9 \times 0.527 = 0.474$	B				
$B3$	$\emptyset$	$\emptyset$	0		No Gain	$\{B1, B2\}$	$\{B3\}$	$0.759 + 0.474 = 1.233$
C	$\emptyset$	$\emptyset$	0					

Table 2 shows that the “No Gain” solution at the root corresponds to the smaller penalty value 1.297. The query set u then leads to a u-cover with just node B being a head subject, leaving A1 and A2 as offshoots.



t	G(t)
A	{A3,A4}
B	{B3}
C	{C}
Root	{A3,A4,B3,C}

Figure 4: (a) Illustrative taxonomy T and a fuzzy query set with membership values assigned to leaves in thousandths; (b) T after pruning and annotating with membership values and gap importance values $V(\cdot)$.

Table 2: Computational results at the root for the fuzzy query set u in Fig. 4, following the computations shown in Table 1; the gap and offshoot penalty weights are $\lambda = 0.2$ and $\gamma = 0.9$.

t	$H(t)$	$L(t)$	$p(t)$	t		$H(Root)$	$L(Root)$	$p(Root)$
A	$\{A1, A2\}$	$\{A3, A4\}$	0.104		Gain	$\{Root\}$	$\{A3, A4, B3, C\}$	$1 + 0.2 \times 2.206 = 1.441$
B	$\{B\}$	$\{B3\}$	1.193	$Root$	**No Gain**	**{A1,A2,B}**	**{B3}**	**0.104 + 1.193 = 1.297**
C	$\emptyset$	$\emptyset$	0.00					

2.3.2 Computing a parsimonious lift (real-world case)

Consider the following fragment of fuzzy cluster R, described later in Section 3.5, related to Computer Vision in DST taxonomy:

- image representations: 0.262,
- shape representations: 0.246,
- appearance and text representations: 0.177,
- hierarchical representations: 0.144,
- video segmentation: 0.209,
- image segmentation: 0.196.

At penalty values $\lambda = 0.1$ and $\gamma = 0.9$, the lifting results are presented in Figure 5; the head subject is Computer vision. The penalty values involved are presented in Table 3.

The cumulative penalty at the head subject is $p(H) = 0.514 + 0.1 * (8 \cdot 0.287) + 0.9 \cdot |\varnothing| = 0.743$. If we do not lift the query set, the penalty would be greater: $p(H) = |\varnothing| + 0.1 \cdot |\varnothing| + 0.9 \cdot (0.262 + 0.246 + 0.177 + 0.144 + 0.209 + 0.196) = 1.214$.

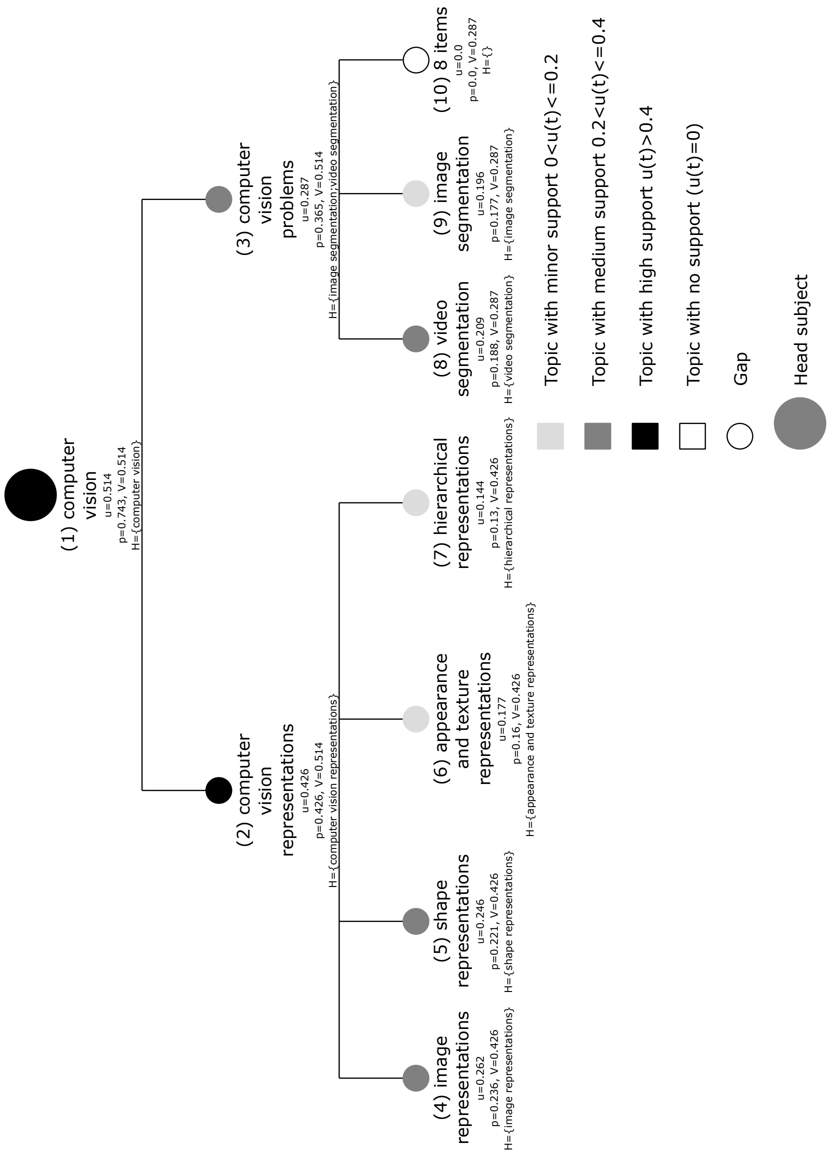

(1) computer vision
u=0.514
p=0.743, V=0.514
H={computer vision}
(2) computer vision representations
u=0.426
p=0.426, V=0.514
H={computer vision representations}
(3) computer vision problems
u=0.287
p=0.365, V=0.514
H={image segmentation;video segmentation}
(4) image representations
u=0.262
p=0.236, V=0.426
H={image representations}
(5) shape representations
u=0.246
p=0.221, V=0.426
H={shape representations}
(6) appearance and texture representations
u=0.177
p=0.16, V=0.426
H={appearance and texture representations}
(7) hierarchical representations
u=0.144
p=0.13, V=0.426
H={hierarchical representations}
(8) video segmentation
u=0.209
p=0.188, V=0.287
H={video segmentation}
(9) image segmentation
u=0.196
p=0.177, V=0.287
H={image segmentation}
(10) 8 items
u=0.0
p=0.0, V=0.287
H={}
Topic with minor support 0<u(t)<=0.2
Topic with medium support 0.2<u(t)<=0.4
Topic with high support u(t)>0.4
Topic with no support (u(t)=0)
Gap
Head subject

Figure 5: Lifting of the Computer Vision sub-cluster. Nodes are numbered as in Table 3.

Table 3: Nodes and variable values for lifting the Computer Vision fuzzy subcluster.

	Node name	$u(t)$	$p(t)$	$v(t)$	$H(t)$	$V(t)$	$L(t)$
1	computer vision	0.514	0.743	0.514	computer vision	interest point and salient region detections, shape inference, object detection, object recognition, object identification, tracking, reconstruction, matching	interest point and salient region detections, shape inference, object detection, object recognition, object identification, tracking, reconstruction, matching
2	computer vision representations	0.426	0.426	0.514	computer vision representations		
3	computer vision problems	0.287	0.365	0.514	image segmentation, video segmentation	interest point and salient region detections, shape inference, object detection, object recognition, object identification, tracking, reconstruction, matching	
4	image representations	0.262	0.236	0.426	image representations		
5	shape representations	0.246	0.221	0.426	shape representations		
6	appearance and texture representations	0.177	0.160	0.426	appearance and texture representations		
7	hierarchical representations	0.144	0.130	0.426	hierarchical representations		
8	video segmentation	0.209	0.188	0.287	video segmentation		
9	image segmentation	0.196	0.177	0.287	image segmentation		
10	interest point and salient region detections, shape inference, object detection object recognition, object identification, tracking, reconstruction, matching	0.000 0.000	0.000 0.000	0.287 0.287			

2.3.3 Effects of changing weights: modeling prejudices

It is well known that cognitive generalizations can be much biased because a cognitive system may involve prejudices preventing that from an adequate assessment of facts. Our model of generalization can imitate this phenomenon.

To illustrate this, let us return to the tree in Fig. 4 and the query set u presented in its part (b). The output of ParGenFS depends not on the membership values only, but also on the interrelation between the topology of the taxonomy tree and the penalty weights, too.

Table 4 presents the resulting u-covers for a number of different combinations of membership values and penalty weights, leading to different interpretations for the taxonomy T presented in Fig. 4(a). For the convenience of reading, the membership values in the table are not normalized. The membership values are extended to the interior nodes using the quadratic normalization condition in (15) further on.

Table 4 exhibits all possible u-cover outcomes at the very same fuzzy set u. Case 1 demonstrates a situation in which all the elements of the optimal u-cover are offshoots, rarely a desirable outcome. By increasing the membership value $u(B2)$ from 0.01 to 0.1, we arrive at Case 2. The optimal u-cover now includes B as a head subject. This case clearly demonstrates how much the structure of the tree may affect the result. Indeed, the bulk of the membership values is clearly in the subtree rooted at A, yet A is not a head subject in this case! The slight asymmetry of the pruned tree and the fact that B has fewer children than A are the causes of this change.

Table 4: Different membership values and penalty weights that lead to all possible descriptions in rows 1 to 5 for the illustrative taxonomy of Fig. 4(a). The algorithm is run using the membership values normalized and extended according to the quadratic condition in (15) further on.

#	$u(A1)$	$u(A2)$	$u(B1)$	$u(B2)$	γ	λ	H(Root)	L(Root)	P(Root)
1	0.8	0.5	0.1	0.01	0.9	0.2	$\{A1, A2, B1, B2\}$	$\{\}$	1.34
2	0.8	0.5	0.1	0.1	0.9	0.2	$\{A1, A2, B\}$	$\{B3\}$	1.40
3	0.8	0.5	0.1	0.01	0.9	0.1	$\{A, B1, B2\}$	$\{A3, A4\}$	1.30
4	0.8	0.5	0.1	0.1	1.1	0.2	$\{A, B\}$	$\{A3, A4, B3\}$	1.56
5	0.8	0.5	0.1	0.1	0.9	0.1	$\{Root\}$	$\{A3, A4, B3, C\}$	1.31
6	0.1	0.01	0.8	0.5	0.9	0.1	$\{A1, A2, B\}$	$\{B3\}$	1.20
7	0.1	0.1	0.8	0.5	0.9	0.1	$\{Root\}$	$\{A3, A4, B3, C\}$	1.23
8	0.1	0.01	0.8	0.5	0.9	0.2	$\{A1, A2, B\}$	$\{B3\}$	1.30

Case 2 can be transformed into either Case 4 or Case 5 by changing just one or other of the weights. By changing γ from 0.9 to 1.1 (Case 4), we obtain a u-cover that includes both A and B as head subjects. This seems somewhat odd – because A and B are the only relevant children of the root, one might think that here the root should be a head subject in an optimal u-cover. Indeed, to achieve this "unnatural" result, we had to make γ greater than unity, the penalty weight for a head subject. By reducing λ from 0.2 in Case 2 to 0.1, we obtain Case 5, for which the root is the sole head subject. A different change to the parameters in Case 1, now changing the weight λ from 0.2 to 0.1 rather than changing the membership values, leads to Case 3, in which A is now the sole head subject (i.e. Case 3 is like Case 5, but reverting to the original membership values in Case 1).

Cases 6, 7 and 8 illustrate results that can be obtained by rearranging the membership values so that the high values are moved from A1 and A2 to B1 and B2, and vice versa. Case 6 leads to a u-cover in which B is the only head subject. Small changes to Case 6, by increasing either the membership value of A2 (Case 7) or the weight λ (Case 8), may or may not lead to changes in the result. The optimal u-cover makes the root a head subject in Case 7, but there is no change from Case 6 to Case 8. Using the quadratic normalization condition in (15) further on. Case 8 is just the familiar fuzzy query set u illustrated in Fig. 4 and Tables 1 and 2.

3 Applying ParGenFS to structuring and conceptualizing a collection of research papers

Being confronted with the problem of structuring and interpreting a set of research publications in a domain, one can think of either of the following two pathways to take. One is so-to-speak empirical and the other theoretical. The first pathway tries to discern main categories from the texts, the other, from knowledge of the domain. The first approach is exemplified by the LDA-based topic modeling (see, for example, [6, 43]); the second approach, by using an expert-driven taxonomy such as ACM-CCS [1] (see, for example, [14, 28]).

This paper follows the second pathway by moving, in sequence, through the following stages:

- preparing a scholarly text collection;
- preparing a taxonomy of the domain under consideration;

- developing a matrix of relevance values between taxonomy leaf topics and research publications from the collection;
- finding fuzzy clusters according to the structure of relevance values;
- lifting the clusters over the taxonomy to conceptualize them via generalization;
- making conclusions from the generalizations.

Each of the items is covered in a separate subsection further on.

3.1 Scholarly text collection

Because of a generous offer from the Springer Publisher, we were able to download a collection of 17685 research papers together with their abstracts published in 17 journals related to Data Science, in our opinion, for 20 years from 1998-2017. We take the abstracts to these papers as a representative collection. The list of the journals is in Table 5.

Table 5: List of Springer journals related to Data Science used as the source for our text collection. Some journals start not in 1998, but later because of unrelated issues.

#	Title	Volumes	Years
1	Pattern Analysis and Applications	1–20	1998–2017
2	Journal of Mathematical Imaging and Vision	14–29	2001–2017
3	World Wide Web	1–20	1998–2017
4	Artificial Intelligence Review	22–48	2004–2017
5	Annals of Mathematics and Artificial Intelligence	23–80	1998–2017
6	Pattern Analysis and Applications	1–20	1998–2017
7	Knowledge and Information Systems	1–52	1999–2017
8	Machine Learning	30–106	1998–2017
9	Swarm Intelligence	1–11	2007–2017
10	Applied Intelligence	14–47	1998–2017
11	Neural Processing Letters	7–45	1998–2017
12	Data Mining and Knowledge Discovery	2–31	1998–2017
13	Machine Vision and Applications	15–28	2004–2017
14	Social Network Analysis and Mining	1–7	2011–2017
15	International Journal on Document Analysis and Recognition	1–20	1998–2017
16	International Journal of Multimedia Information Retrieval	1–6	2012–2017
17	Pattern Recognition and Image Analysis	16–27	2006–2017

3.2 DST Taxonomy

Taxonomy is a form of knowledge engineering which is getting more and more popular. Most known are taxonomies within the bioinformatics Genome Ontology project (GO) [12], health and medicine SNOMED CT project [16] and the like.

Mathematically, a taxonomy is a rooted tree, a hierarchy, whose all nodes are labeled by main concepts of a domain. The hierarchy corresponds to a relation of inclusion: the fact that node A is the parent of B means that B is part, or a special case, of A. An important characteristic of a rooted tree is that each node has only one parent.

There are two major approaches for developing a domain taxonomy: automatic and manual. The latter currently is by far more mature and developed. However, even this approach suffers of deficiences summarized as follows: "Taxonomy design decisions regarding the used classification structures, procedures and descriptive bases are usually not well described and motivated." [41]. The automatic approach can exploit a multitude of digital resources and methods for semantic analysis. A rather comprehensive attempt is described in [41]. Our preferences, at this moment, are with taxonomies manually established by a representative body of specialists. Indeed, such a taxonomy does not depend on purely empirical data utilized by automatic methods.

Moreover, a manual taxonomy usually balances the theoretical insight and practical experience accumulated in the community represented by the body issuing it. Such is the Computing Classification System (ACM-CCS 2012) by the world-wide Association for Computing Machinery [1]. The subdomain of our choice is Data Science, comprising such areas as machine learning, data mining, data analysis, etc. We take that part of the ACM-CCS 2012 taxonomy, which is related to Data Science, and add a few leaves related to more recent Data Science developments. A major extract from the taxonomy of Data Science is published in [24].

The higher ranks of the taxonomy are presented in Table 6 and its full version, in Table 6 in the Appendix. The leaves added by the authors are labeled with a star "*".

Table 6: ACM Computing Classification System (ACM-CCS) 2012 higher rank subjects related to Data Science.

Subject index	Subject name
1.	Theory of computation
1.1.	Theory and algorithms for application domains
2.	Mathematics of computing
2.1.	Probability and statistics
3.	Information systems
3.1.	Data management systems
3.2.	Information systems applications
3.3.	World Wide Web
3.4.	Information retrieval
4.	Human-centered computing
4.1.	Visualization
5.	Computing methodologies
5.1.	Artificial intelligence
5.2.	Machine learning

3.3 Evaluation of relevance between texts and key phrases

After ground-breaking discoveries of methods for automatically developing sets of topics relevant to collections of documents [5, 6], it has become popular to concentrate on key-words taken from the documents being analyzed. We, however, prefer using topics produced manually by committees of experts because of the obvious advantages: comprehensiveness and stability. Therefore, our list of key-phrases comprises lower-rank ACM-CCS taxonomy topics related to data science. To take into account specifics of the structure of the collection of texts being analyzed, one should focus on the relevance between taxonomy key-phrases and the texts. Most popular and well established approaches to scoring keyphrase-to-document relevance include the so-called vector-space approach [34] and probabilistic text model approach [33]. These, however, rely on individual words and text pre-processing.

We utilize a method, first developed by R. Pampapathi et al [30] and further advanced by Chernyak and Mirkin [9, 10], the AST method for evaluating keyphrase-to-text relevance score using purely string frequency information. An advantage of the method is that it requires no manual work, but works rather reliably, as claimed by these authors.

An Annotated Suffix Tree (AST) is a weighted rooted tree used for storing text fragments and their frequencies. To build an AST for a text string, all suffixes from this string are extracted.

A k-suffix of a string $x = x_1x_2 \dots x_N$ of length N is a continuous end fragment $x^k = x_{N-k+1}x_{N-k+2} \dots x_N$. For example, a 3-suffix of string $INFORMATION$ is substring ION, and a 5-suffix, $ATION$. Each AST node is assigned a symbol and the so-called annotation (frequency of the substring corresponding to the path from the root to the node including the symbol at the node). The root node of AST has no symbol or annotation (see Figure 3.3). An algorithm for building an AST for any given string $x = x_1x_2 \dots x_N$ is described below.

1. Initialize an AST to consist of a single node, the root: T.
2. Find all the suffixes of the given string:
   $\{x^k = x_{N-k+1}x_{N-k+2} \dots x_N | k = 1, 2, \dots, N\}$.
3. For each suffix x^k find its maximal overlap, that is, a path from the root in T coinciding with its beginning fragment $x^{k_{max}}$. At each node of the path for $x^{k_{max}}$ add 1 to the annotation. If the length of the overlap $x^{k_{max}}$ is less than k, the path is extended by adding new nodes corresponding to symbols from the remaining part of this suffix. Annotations of all the new nodes are set to be 1.

To accelerate the working of the method, one should use efficient versions of algorithms utilising suffix trees and suffix arrays (see, for example, [13]).

Having an AST T built, we can score the string-to-document relevance over the AST. To do this, we follow [21] by computing the conditional probability of node u in T:

$$p(u) = \frac{f(u)}{f(parent(u))}. \tag{4}$$

For all the immediate offspring of the root (R), formula has the following form:

$$p(u) = \frac{f(u)}{\sum\limits_{v \in T: parent(v) = R} f(v)}, \tag{5}$$

where $f(u)$ is the frequency annotation of the node u. Using the formula above, one can calculate the probability of node u relative to all its siblings.

For each suffix x^k of string x the relevance score $s(x^k, T)$ is defined as:

$$s(x^k, T) = \frac{1}{k_{max}} \sum_{i=1}^{k_{max}} p(x_i^k). \tag{6}$$

The AST relevance score of string x and text T is defined as the sum of all the suffix scores:

$$S(x, T) = \frac{1}{N} \sum_{k=1}^{N} s(x^k, T). \tag{7}$$

In practical computations, we split any document into a set of strings (usually consisting of 2-3 consecutive words), create an empty AST for the document and add these strings in the AST in sequence, by using the algorithm above.

To lessen the effects of frequently occurring general terms, the scoring function is modified by five-fold decreasing the weight of stop-words. The list of stop-words includes: "learning, analysis, data, method" and a few postfixes: "s/es, ing, tion". After an AST for a document has been built, the time complexity of calculating the string-to-document relevance score is $O(m^2)$ where m is the length of the query string. This does not depend on the document length, in contrast to the popular Levenstein-distance based approaches.

Let us consider, for example, a "document" consisting of a string 'inference'. We start from an AST consisting of a single node (Root), split the document into short strings and sequentially add suffixes of these strings to the AST. In the example, we should add all the suffixes of the only string to the AST: from 'inference' to 'nference' to 'ference', etc., and to 'e' (see Figures 6 and 7).

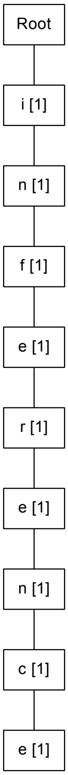


Figure 6: AST for string ‘inference’: step 1.

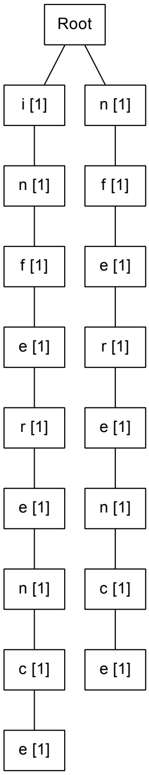


Figure 7: AST for string ‘inference’: step 2.

The final AST for string ‘inference’ is shown in Figure 8.

Let us calculate the relevance score of a string ‘fence’ according to the AST found above. The string has five suffixes: ‘fence’, ‘ence’, ‘nce’, ‘ce’, ‘e’. First of all, one needs to calculate relevance scores for these suffixes according to formula 6. These scores are presented in Table 7. After that, according to formula 7, we calculate score for the whole string:

$$S(fence, T) = \frac{1}{5} \cdot (0.5555 + 0.7083 + 0.5740 + 0.5555 + 0.3333) \approx 0.545$$

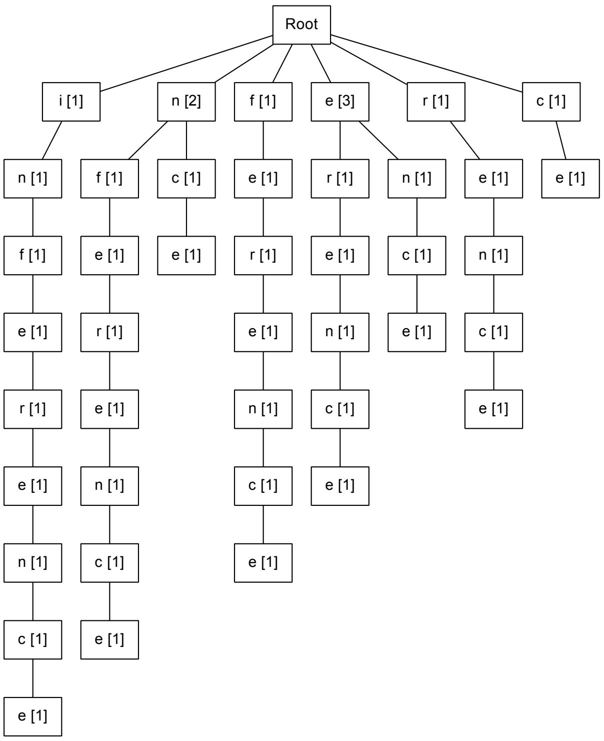


Figure 8: Final AST for string ‘inference’.

Consider another example, relevance score calculations for string ‘since’ (see Table 8). For the whole string, we have:

$$S(since, T) = \frac{1}{5} \cdot (0 + 0.5555 + 0.5740 + 0.5555 + 0.3333) \approx 0.404$$

Table 7: Computing the relevance scores for suffixes of a string ‘fence’.

Suffix	Match	Score
‘fence’	‘fe’	$\frac{1}{2} \cdot \left(\frac{1}{9} + \frac{1}{1}\right) \approx 0.5555$
‘ence’	‘ence’	$\frac{1}{4} \cdot \left(\frac{3}{9} + \frac{1}{2} + \frac{1}{1} + \frac{1}{1}\right) \approx 0.7083$
‘nce’	‘nce’	$\frac{1}{3} \cdot \left(\frac{2}{9} + \frac{1}{2} + \frac{1}{1}\right) \approx 0.5740$
‘ce’	‘ce’	$\frac{1}{2} \cdot \left(\frac{1}{9} + \frac{1}{1}\right) \approx 0.5555$
‘e’	‘e’	$\frac{1}{1} \cdot \left(\frac{3}{9}\right) \approx 0.3333$

Table 8: Computing the scores for suffixes of string ‘since’.

Suffix	Match	Score
‘since’	‘’	0
‘ince’	‘in’	$\frac{1}{2} \left(\frac{1}{9} + \frac{1}{1}\right) \approx 0.5555$
‘nce’	‘nce’	$\frac{1}{3} \left(\frac{2}{9} + \frac{1}{2} + \frac{1}{1}\right) \approx 0.5740$
‘ce’	‘ce’	$\frac{1}{2} \left(\frac{1}{9} + \frac{1}{1}\right) \approx 0.5555$
‘e’	‘e’	$\frac{1}{1} \left(\frac{3}{9}\right) \approx 0.3333$

3.4 Defining and computing fuzzy clusters of taxonomy topics

Clusters of topics should reflect co-occurrence of topics: the greater the number of texts to which both topics t and t' are relevant, the greater the interrelation between t and t', the greater the chance for topics t and t' to fall in the same cluster.

We have tried several popular clustering algorithms. Unfortunately, no satisfactory results have been found. Therefore, we present here results obtained with the FADDIS algorithm from [21] developed specifically for finding thematic clusters.

This algorithm implements assumptions that are relevant to the task:

LN Laplacian Normalization: Similarity data transformation modeling – to an extent – heat distribution and, in this way, making the cluster structure sharper.

AA Additivity: Thematic clusters behind the texts are additive so that similarity values are sums of contributions by different hidden themes.

AN Non-Completeness: Clusters do not necessarily cover all the key phrases available as the text collection under consideration may be irrelevant to some of them.

3.4.1 Co-relevance topic-to-topic similarity score

Given a keyphrase-to-document matrix R of relevance scores, it is converted to a keyphrase-to-keyphrase similarity matrix A or scoring the "co-relevance" of keyphrases according to the text collection structure. The similarity score $a_{tt'}$ between topics t and t' can be computed as the inner product of vectors of scores $r_t = (r_{tv})$ and $r_{t'} = (r_{t'v})$ where $v = 1, 2, \ldots, V = 17685$. The inner product is moderated by a natural weighting factor assigned to texts in the collection. The weight of text v is defined as the ratio of the number of topics n_v relevant to it and n_{max}, the maximum n_v over all v = 1,2,...,V. A topic is considered relevant to v if its relevance score is greater than 0.2 (a threshold found experimentally, see [9]). The numbers of texts in our collection, that are relevant to 0, 1, or more topics, are presented in Table 9.

Table 9: The numbers of relevant topics in our text collection.

Number of texts	Number of relevant topics
1237	0
2353	1
7114	2,3, or 4
6124	5–11
857	12 or more

The accepted topic-to-topic similarity measure has the following properties [21]:

1. The similarity matrix is positive semi-definite.
2. The similarity between two topics can be positive if and only if there is at least one text in which both are engaged.
3. The greater the individual relevance scores, the greater the similarity.
4. Given a pair of topics, the greater the set of texts relevant to both of them, the greater the similarity.

3.4.2 Additive fuzzy spectral clustering

Let us denote the total set of leaf topics by T and assume that a fuzzy cluster over T is represented by a fuzzy membership vector $\vec{u} = (u_t)$, $t \in T$, such that $0 \leq u_t \leq 1$ for all $t \in T$, and an intensity $\mu > 0$, a scale coefficient to relate the membership scores to the similarity scores. For T being a set of research topics and $\vec{u} = (u_t)$, $t \in T$, a membership values vector representing the a semantic substructure of a corpus of research papers under consideration, the product $(\mu u_t)(\mu u_{t'}) = \mu^2 u_t u_{t'}$ can be considered as the contribution by the research direction represented by the cluster under consideration to the total similarity score $a_{tt'}$ between topics t and t'. The additive fuzzy clustering model in [21] states that the entries in the topic-to-topic similarity matrix A can be considered as resulting from additive contributions of K fuzzy clusters, up to small errors to be minimized:

$$a_{tt'} = \sum_{k=1}^{K} \mu_k^2 u_{kt} u_{kt'} + e_{tt'}, \tag{8}$$

where $\vec{u}_k = (u_{kt})$ is the membership vector of cluster k, and μ_k its intensity. These assumptions require that clusters are extracted according to an additive model. A method developed in [21], FADDIS, finds clusters in (8) one-by-one, which accords with the assumptions above. Paper [21] provides some theoretical and experimental computation results to demonstrate that FADDIS is competitive over other fuzzy clustering approaches.

A fuzzy cluster $(\vec{u},\ \mu)$ is extracted to minimize the one-cluster least-squares criterion

$$E = \sum_{t,t' \in T} (w_{tt'} - \xi u_t u_{t'})^2 \tag{9}$$

with respect to unknown positive $\xi = \mu^2$ and fuzzy membership vector $\vec{u} = (u_t)$ for a given similarity matrix $W = (w_{tt'})$.

Initially, $W = A$. Then the matrix W is changed by subtracting from it that part of the similarities that is accounted for by the found cluster, $W \leftarrow W - \mu^2 \vec{u}\vec{u}'$, where μ and $\vec{u}$ are the intensity and membership vector of the cluster.

In this way, A is step-by-step additively decomposed in accordance with the model (8), so that the number of clusters K can be determined during the process. The first-order optimality condition for the criterion E in (9) states that an optimal ξ satisfies equation:

$$\xi = \frac{\sum_{t,t' \in T} w_{tt'} u_t u_{t'}}{\sum_{t \in T} u_t^2 \sum_{t' \in T} u_{t'}^2},$$

or, in matrix terms,

$$\xi = \frac{\vec{u}^T W \vec{u}}{(\vec{u}^T \vec{u})^2}. \tag{10}$$

By substituting this value of ξ in (9), we obtain

$$E = S(W) - \xi^2 \left(\vec{u}^T \vec{u}\right)^2,$$

where $S(W) = \sum_{t,t' \in T} w_{tt'}^2$ is the similarity data scatter.

Denoting the last term by $G(\vec{u})$, we have

$$G(\vec{u}) = \xi^2 \left(\vec{u}^T \vec{u}\right)^2 = \left(\frac{\vec{u}^T W \vec{u}}{\vec{u}^T \vec{u}}\right)^2, \tag{11}$$

which provides for a decomposition of the similarity data scatter into explained and unexplained parts, according to the equation above:

$$S(W) = G(\vec{u}) + E. \tag{12}$$

Therefore, to minimize E, one must maximize $G(\vec{u})$ in (11) or its square root,

$$g(\vec{u}) = \xi \vec{u}^T \vec{u} = \frac{\vec{u}^T W \vec{u}}{\vec{u}^T \vec{u}}, \tag{13}$$

The value $g(\vec{u})$ in (13) is the so-called Rayleigh quotient whose maximum is the maximum eigenvalue of W reached at the corresponding eigenvector. This brings forward the celebrated spectral clustering approach [37]. According to this approach, one first solves the unconstrained problem of maximizing $g(\vec{u})$ with respect to any u, to obtain the normalized eigenvector $\vec{z}$ corresponding to the maximum eigenvalue of W. Then its projection $\vec{u}$ onto the set of nonnegative fuzzy membership vectors is found:

$$u_t = \begin{cases} 0, & \text{if } z_t \leq 0; \\ z_t, & \text{if } 0 < z_t \leq 1. \end{cases} \tag{14}$$

As the criterion in (13) involves division over the squared Euclidean norm of vector u, the Euclidean normalization condition is most natural for the solution, so that the extracted cluster u is normalized accordingly:

$$\vec{u}^T \vec{u} = 1. \tag{15}$$

The process of one-by-one extracting fuzzy clusters stops when any of the conditions below is satisfied:

1. The value of ξ given by (10) at the current step is negative.
2. The contribution of a single extracted cluster is too low. For example, a cluster should contribute at least as much as an average entity, so that the threshold $1/N$ should be considered a well-defined conservative value.
3. The residual scatter E becomes smaller than, say, 5% of the original similarity data scatter.
4. A pre-specified number $K_{\max}$ of clusters is reached.

To make the hidden cluster structure in similarity data sharper, we apply the so-called Laplacian normalization [18, 37]. This normalization usually applies with the so-called minimum normalized cut criterion, and, thus, brings forward the minimum eigenvalue, whereas FADDIS uses the maximum one. This is why this normalization is modified in [21] to involve the inverse eigen-values. Specifically, the so-called Laplacian pseudo-inverse transformation (Lapin, in short) is applied. Given a similarity matrix W, the $N \times N$ diagonal matrix D is computed, with (t, t)-th entry equal to $d_t = \sum_{t' \in T} w_{tt'}$, the sum of row t of W, and the Laplacian is defined as

$L_n = I - D^{-1/2}WD^{-1/2}$. Then the Laplace Pseudo INverse transformation (Lapin) is defined as

$$L_n^+ = Z\tilde{\Lambda}^{-1}Z^T,$$

where Z is the matrix of eigenvectors corresponding to non-zero eigenvalues, from the spectral decomposition $L_n = Z\Lambda Z^T$, and the diagonal matrix $\tilde{\Lambda}$ is obtained from Λ by removing zero eigenvalues and replacing each non-zero eigenvalue λ by $1/\lambda$.

Lastly, the eigenvector corresponding to the maximum eigenvalue of L_n^+ is taken to generate the fuzzy cluster for the model (9).

3.4.3 FADDIS thematic clusters

After computing the 317 × 317 topic-to-topic co-relevance matrix, converting in to a topic-to-topic Lapin transformed similarity matrix, and applying FADDIS clustering, we sequentially obtained 6 clusters, of which three clusters seem especially homogeneous. We denote them using letters L, for 'Learning'; R, for 'Retrieval'; and C, for 'Clustering'. These clusters are presented in Tables 10, 11, and 12, respectively.

Table 10: Cluster L "Learning": topics with membership values greater than 0.15.

$u(t)$	Code	Topic
0.300	5.2.3.8.	rule learning
0.282	5.2.2.1.	batch learning
0.276	5.2.1.1.2.	learning to rank
0.217	1.1.1.11.	query learning
0.216	5.2.1.3.3.	apprenticeship learning
0.213	1.1.1.10.	models of learning
0.203	5.2.1.3.5.	adversarial learning
0.202	1.1.1.14.	active learning
0.192	5.2.1.4.1.	transfer learning
0.192	5.2.1.4.2.	lifelong machine learning
0.189	1.1.1.8.	online learning theory
0.166	5.2.2.2.	online learning settings
0.159	1.1.1.3.	unsupervised learning and clustering

Table 11: Cluster R "Retrieval": topics with membership values greater than 0.15.

$u(t)$	Code	Topic
0.211	3.4.2.1.	query representation
0.207	5.1.3.2.1.	image representations
0.194	5.1.3.2.2.	shape representations
0.194	5.2.3.6.2.1	tensor representation
0.191	5.2.3.3.3.2	fuzzy representation
0.187	3.1.1.5.3.	data provenance
0.173	2.1.1.5.	equational models
0.173	3.4.6.5.	presentation of retrieval results
0.165	5.1.3.1.3.	video segmentation
0.155	5.1.3.1.2.	image segmentation
0.154	3.4.5.5.	sentiment analysis

Table 12: Cluster C "Clustering": topics with membership values greater than 0.15.

$u(t)$	Code	Topic
0.327	3.2.1.4.7	biclustering
0.286	3.2.1.4.3	fuzzy clustering
0.248	3.2.1.4.2	consensus clustering
0.220	3.2.1.4.6	conceptual clustering
0.192	5.2.4.3.1	spectral clustering
0.187	3.2.1.4.1	massive data clustering
0.159	3.2.1.7.3	graph based conceptual clustering
0.151	3.2.1.9.2.	trajectory clustering

3.5 Results of lifting clusters L, R, and C within DST

All obtained clusters are lifted in the DST taxonomy using ParGenFS algorithm with the gap penalty $\lambda = 0.1$ and off-shoot penalty $\gamma = 0.9$.

The results of lifting Cluster L are shown in Figure 9. The cluster has received three head subjects: machine learning, machine learning theory, and learning to rank. These represent the structure of the general concept "Learning" according to our text collection. The list of gaps obtained is less instructive, reflecting probably a relatively modest coverage of the domain by the publications in the collection (see in Table 13).

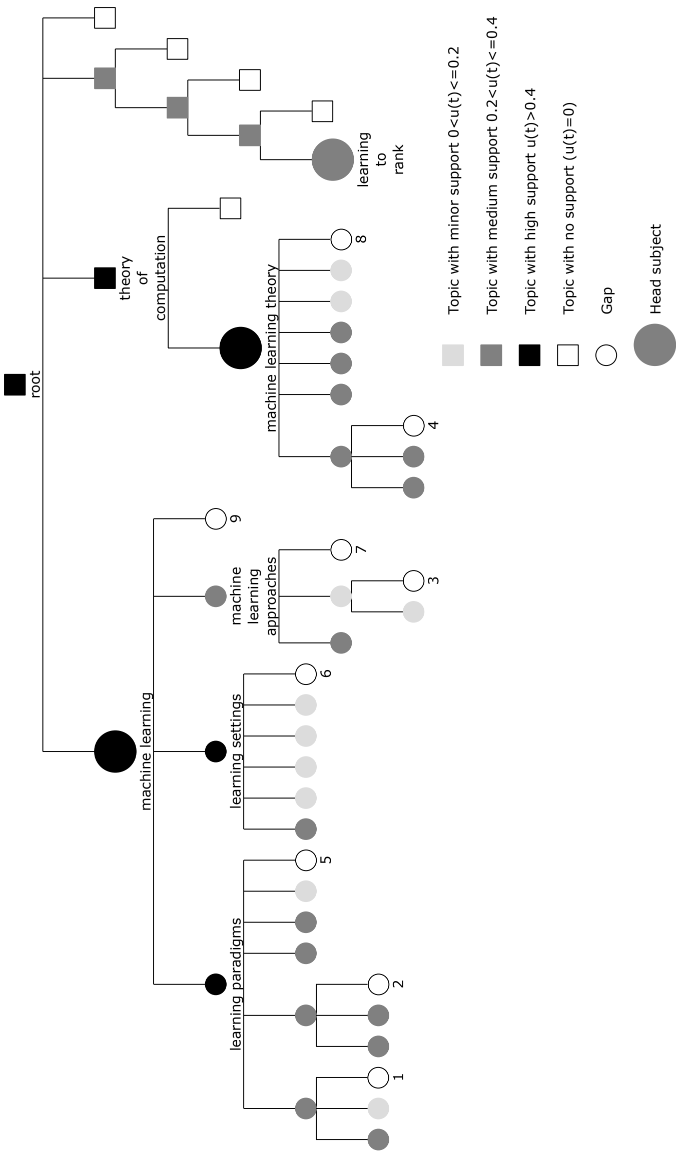

root
theory of computation
learning to rank
machine learning theory
machine learning
machine learning approaches
learning settings
learning paradigms
1
2
3
4
5
6
7
8
9
Topic with minor support 0<u(t)<=0.2
Topic with medium support 0.2<u(t)<=0.4
Topic with high support u(t)>0.4
Topic with no support (u(t)=0)
Gap
Head subject

Figure 9: Lifting results for Cluster L: Learning. Gaps are numbered, see Table 13.

Table 13: Gaps at the lifting of Cluster L.

Number	Topics
1	ranking, supervised learning by classification, structured outputs
2	sequential decision making in practice, inverse reinforcement learning in practice
3	statistical relational learning
4	sequential decision making, inverse reinforcement learning
5	unsupervised learning
6	learning from demonstrations, kernel approach
7	classification and regression trees, kernel methods, neural networks, learning in probabilistic graphical models, learning linear models, factorization methods, markov decision processes, stochastic games, learning latent representations, multiresolution, support vector machines
8	sample complexity and generalization bounds, boolean function learning, kernel methods, boosting, bayesian analysis, inductive inference, structured prediction, markov decision processes, regret bounds
9	machine learning algorithms

Similar comments can be made with respect to results of lifting of Cluster R: Retrieval (see Figure 10). The obtained head subjects: Information Systems and Computer Vision show the structure of "Retrieval" in the set of publications under considerations. Gaps obtained for Cluster R are listed in Table 14.

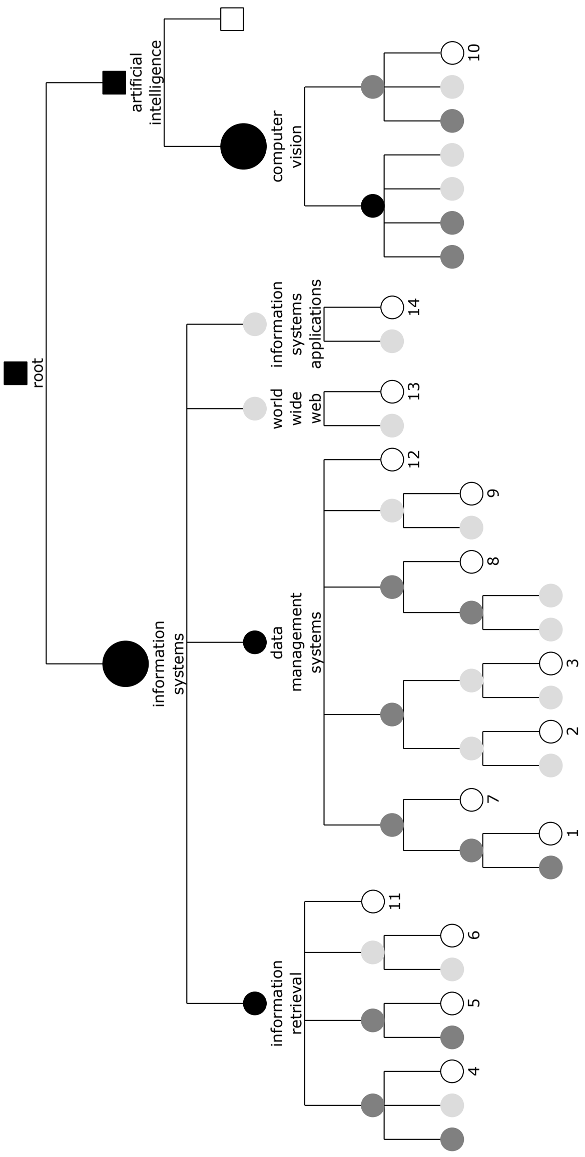


Figure 10: Lifting results for Cluster R. Gaps are numbered, see Table 14. The meaning of elements in the Figure is explained in the legend to Fig. 9.

Table 14: Gaps at the lifting of Cluster R.

Number	Topics
1	data streams, incomplete data, temporal data, uncertainty, inconsistent data
2	multidimensional range search, point lookups, unidimensional range search, proximity search
3	data compression, record and block layout
4	query intent, query suggestion, query reformulation
5	test collections, relevance assessment, retrieval effectiveness, retrieval efficiency
6	question answering, document filtering, recommender systems, information extraction, expert search, clustering and classification, summarization, business intelligence
7	relational database model, entity relationship models, hierarchical data models, network data models, physical data models
8	relational database query languages, mapreduce languages, call level interfaces
9	data exchange, data cleaning, mediators and data integration, entity resolution, data warehouses
10	interest point and salient region detections, shape inference, object detection, object recognition, object identification, tracking, reconstruction, matching
11	document representation, users and interactive retrieval, retrieval models and ranking, specialized information retrieval
12	database management system engines
13	site wrapping, data extraction and integration, traffic analysis, knowledge discovery
14	data cleaning, collaborative filtering, association rules, clustering, data stream mining, graph mining, process mining, text mining, data mining tools, sequence mining

We decided to show no drawing for the results of lifting of Cluster C, because the corresponding taxonomy fragment is too large, whereas the lifting results are too fragmentary. There are 16 (!) head subjects here:

- clustering,
- graph based conceptual clustering, item trajectory clustering,
- clustering and classification,
- unsupervised learning and clustering,
- spectral methods,
- document filtering,
- language models,

- music retrieval,
- collaborative search,
- database views,
- stream management,
- database recovery,
- mapreduce languages,
- logic and databases,
- language resources.

As one can see, the core clustering subjects are supplemented by methods and environments in the cluster – this shows that the ever increasing role of clustering activities perhaps should be better reflected in the taxonomy.

3.6 Making conclusions

We can see that the topic clusters found with the text collection do highlight areas of soon-to-be developments. Three clusters under consideration closely relate, in respect, to the following processes:

- theoretical and methodical research in learning, as well as merging the subject of learning to rank within the mainstream;
- representation of various types of data for information retrieval, and merging that with visual data and their semantics; and
- various types of clustering in different branches of the taxonomy related to various applications and instruments.

In particular, one can see from the "Learning" head subjects (see Figure 9 and comments to it) that main work here still concentrates on theory and method rather than applications. A good news is that the field of learning, formerly focused mostly on tasks of learning subsets and partitions, is expanding currently towards learning of ranks and rankings. Of course, there remain many subareas to be covered: these can be seen in and around the list of gaps in Table 13.

Moving to the lifting results for the information retrieval cluster R (see Figure 10 and Table 11), we can clearly see the tendencies of the contemporary stage of the process. Rather than relating the term "information" to texts only, as it was in the previous stages of the process of digitalization, visuals are becoming parts of the concept of information. There is a catch, however. Unlike the multilevel granularity of meanings in texts, developed during millennia of the process of communication via languages in the humankind, there is no comparable hierarchy of meanings for images. One may only guess that the elements of the R cluster related to segmentation of images and videos, as well as those related to data management systems, are those that are going to be put in the base of a future multilevel system of meanings for images and videos. This is a direction for future developments clearly seen from the picture in Figure 10.

Regarding the "clustering" cluster C with its 16 (!) head subjects, one may conclude that, perhaps, a time moment has come or is to come real soon, when the subject of clustering must be raised to a higher level in the taxonomy to embrace all these "heads". At the beginning of the Data Science era, a few decades ago, clustering was usually considered a more-or-less auxiliary part of machine learning, the unsupervised learning. Perhaps, soon we are going to see a new taxonomy of Data Science, in which clustering is not just an auxiliary instrument but rather a model of empirical classification, a big part of the knowledge engineering. When discussing the role of classification as a knowledge engineering phenomenon, one encounters three conventional aspects of classification:

- structuring the phenomena;
- relating different aspects of phenomena to each other;
- shaping and keeping knowledge of phenomena.

Each of them can make a separate direction of research in knowledge engineering.

4 Conclusion

This paper presents a formalization of the concept of generalization, an important part of the human ability for conceptualization. According to Collins Dictionary, conceptualization is "formation (of a concept or concepts) out of observations, experience, data, etc." We assume that such an operation may require a coarser granularity of the domain structuring. This is captured by the idea of lifting a query set to higher ranks in a hierarchical taxonomy of the domain.

The hierarchical structure of taxonomy brings in possible inconsistencies between a query set and the taxonomy structure. These inconsistencies can be of either of two types, gaps or offshoots, potentially emerging at the coarser "head subject" to cover the query set. A gap is such a node of the taxonomy, that is covered by the head subject but does not belong in the query set. An offshoot is a node of the taxonomy, that does belong in the query set but is not covered by the head subject. The higher the rank of a candidate for the conceptual head subject, the larger the number of gaps. The lower is the rank of the head subject, the larger the number of offshoots. Our algorithm ParGenFS allows to find a globally optimal lifting to balance the numbers of head subjects, gaps, and offshoots depending on relative penalties for these types of inconsistencies.

We illustrate usefulness of this approach on the set of 17685 abstracts of research papers published by the Springer Publishers for 20 years, 1998–2017, in 17 journals related to Data Science. We use this set to obtain six fuzzy clusters of taxonomy topics according to their co-relevance. We can easily interpret only three out of the found clusters, which probably reflects the inherent randomness of research processes and presence in it of an agenda not clearly manifested so far. It should be pointed out that finding interpretable clusters of taxonomy topics over the textual data requires using rather sophisticated methods involving spectral clustering, weighting publications, Laplacian transformation and the like. This complexity perhaps comes, at least partly, from the way we estimate the similarity between concepts – by considering them as just strings and, thus, by removing any imposed pre-structuring coming from pre-selected keywords or NLP constructions. Lifting of these clusters brings in several general conclusions over the current research in Data Science. These conclusions, even if not entirely unexpected, give as a glimpse into the hidden research processes, as captured by the authorship of the Springer journals.

The proposed approach to generalization can be used in a number of similar tasks such as positioning of a research project, interpretation of a concept which is not present in the taxonomy, annotation of a set of research articles. These all are parts of the processes of long-term research analysis and planning at which our approach should be positioned.

Among major issues requiring further development in this direction, two of the most relevant are:

- Taxonomy developments
- Specifying penalty weights

The former needs more attention both from research communities and planning committees. Specifically, most urgent directions for development here are:

- developing better methods to automate the process of taxonomy making (an interesting approach to this is described in [15]) ; and
- open discussion of the taxonomies at conferences and meetings of research communities and committees(the need in this is pointed out in a recent paper [25].

As to the latter, a reasonable computational progress over penalty weights can be achieved, in our view, by replacing the criterion of maximum parsimony by the criterion of maximum likelihood if each node of the taxonomy can be assigned probabilities of "gain" and "loss" of topic events. We intend pursuing this approach for the DST taxonomy in the near future.

Acknowledgment. Dmitry Frolov and Boris Mirkin acknowledge support by the Basic Research Program at the National Research University Higher School of Economics (HSE) and by the International Decision Choice and Analysis Laboratory (DECAN) NRU HSE in the framework of a subsidy granted to the HSE by the Government of the Russian Federation for the implementation of the Global Competitiveness Program.

5 References

[1] The 2012 ACM Computing Classification System. [Online]. Available:*http://www.acm.org* /about/class/2012 (Accessed 2018, 30 April).

[2] J. Ashraf, E. Chang, O. K. Hussain, F. K. Hussain, "Ontology usage analysis in the ontology lifecycle: A state-of-the-art review," *Knowledge-Based Systems*, vol. 80, pp. 34-47, 2015.

[3] D. Beneventano, N. Dahlem, S. El Haoum, A. Hahn, D. Montanari, M. Reinelt, "Ontology-driven semantic mapping," *Enterprise Interoperability III*, Part IV, Springer, pp. 329-341, 2008.

[4] J.C. Bezdek, R.J. Hathaway, M.P. Windham, "Numerical comparisons of the RFCM and AP algorithms for clustering relational data," *Pattern Recognition*, 24, pp. 783-791, 1991.

[5] Blei D.M., Ng D.M., Jordan M.I. "Latent Dirichlet allocation," *The Journal of Machine Learning Research*, 3, pp. 993–1022, 2003.

[6] D. Blei, "Probabilistic topic models," *Communications of the ACM*, 55 (4), pp. 77–84, 2012.

[7] R.K. Brouwer, "A method of relational fuzzy clustering based on producing feature vectors using FastMap," *Information Sciences*, 179, pp. 3561-3582, 2009.

[8] P. Buche, J. Dibie-Barthelemy, L. Ibanescu, "Ontology mapping using fuzzy conceptual graphs and rules," *ICCS Supplement*, pp. 17-24, 2008.

[9] Chernyak, Ekaterina. "An approach to the problem of annotation of research publications." *Proceedings of the eighth ACM international conference on web search and data mining*, ACM, 429-434, 2015.

[10] E. Chernyak, B. Mirkin, "Refining a Taxonomy by Using Annotated Suffix Trees and Wikipedia Resources," *Annals of Data Science*, 2(1), pp. 61-82, 2015.

[11] M.J. Gacto, R. Alcalá, F. Herrera, "Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures," *Information Sciences*, vol. 181, pp. 4340-4360, 2011.

[12] Gene Ontology Consortium, "Gene ontology consortium: going forward," *Nucleic Acids Research*, vol. 43 D1049-D1056, 2015.

[13] R. Grossi and J.S. Vitter, "Compressed suffix arrays and suffix trees with applications to text indexing and string matching," *SIAM Journal on Computing*, 35, 2 pp.378-407, 2005.

[14] Kapur, S., Ayman O. F., and R. E. Chatwin. "Method and apparatus for representing text using search engine, document collection, and hierarchal taxonomy." U.S. Patent No. 7,580,926. 2009.

[15] Klavans, Richard, and Kevin W. Boyack, "Which type of citation analysis generates the most accurate taxonomy of scientific and technical knowledge?", *Journal of the Association for Information Science and Technology*, 68(4), pp. 984-998, 2017.

[16] D. Lee, R. Cornet, F. Lau, N. De Keizer, "A survey of SNOMED CT implementations," *Journal of Biomedical Informatics*, vol. 46, no. 1, pp. 87-96, 2013.

[17] Lloret, E., Boldrini, E., Vodolazova, T., Martnez-Barco, P., Munoz, R., & Palomar, M. "A novel concept-level approach for ultra-concise opinion summarization", *Expert Systems with Applications*, 42(20), pp. 7148-7156, 2015.

[18] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, pp. 395-416, 2007.

[19] J.P. Mei, Y. Wang, L. Chen, and C. Miao, "Large scale document categorization with fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, 25(5), 1239-1251, 2017.

[20] B. Mirkin, *Clustering: A Data Recovery Approach*, Chapman and Hall/CRC Press, 2012.

[21] Mirkin B. G., Chernjak E. L., Chugunova O. N. *Metod annotirovannogo suffiksnogo dereva dlja ocenki stepeni vhozhdenija strok v tekstovye dokumenty*. [Annotated Suffix Tree as a Way of String-To-Document Score Evaluating]. Business Informatics, 3 (21), pp. 31–41, 2012 (in Russian).

[22] B. Mirkin, S. Nascimento, "Analysis of Community Structure, Affinity Data and Research Activities using Additive Fuzzy Spectral Clustering," Technical Report 6, School of Computer Science, Birkbeck University of London, 2009.

[23] B. Mirkin, S. Nascimento, "Additive spectral method for fuzzy cluster analysis of similarity data including community structure and affinity matrices," *Information Sciences*, vol. 183, no. 1, pp. 16-34, 2012.

[24] B. Mirkin, M. Orlov, "Three aspects of the research impact by a scientist: measurement methods and an empirical evaluation." In *Optimization, Control, and Applications in the Information Age*, Springer, Cham, pp. 233-259, 2015.

[25] F. Murtagh, M. Orlov, & B. Mirkin, "Qualitative judgement of research impact: Domain taxonomy as a fundamental framework for judgement of the quality of research." *Journal of Classification*, 35(1), pp. 5-28, 2018.

[26] Mueller G, Bergmann R. "Generalization of Workflows in Process-Oriented Case-Based Reasoning,", In FLAIRS Conference, pp. 391-396, 2015.

[27] D. Nallaperuma & D. De Silva, "A participatory model for multi-document health information summarisation," *Australasian Journal of Information Systems*, 21.

[28] S. Nascimento, T. Fenner, B. Mirkin, "Representing research activities in a hierarchical ontology," in *Procs. of 3rd International Workshop on Combinations of Intelligent Methods and Applications* (CIMA 2012), Montpellier, France, August, 28, pp. 23-29, 2012.

[29] F. Osborne, A. Salatino, A. Birukou, E. Motta, "Automatic Classification of Springer Nature Proceedings with Smart Topic Miner," In: Groth P. et al. (Eds) *The Semantic Web* – ISWC 2016. *Lecture Notes in Computer Science*, vol 9982, pp. 383-399. Springer, Cham, 2016.

[30] Pampapathi R., Mirkin B., Levene M. "A suffix tree approach to anti-spam email filtering," *Machine Learning*, 65(1), pp. 309–338, 2006.

[31] A. M. Rinaldi, "An ontology-driven approach for semantic information retrieval on the Web," *ACM Trans. on Internet Technologies*, vol. 9, no.3, article 10, 2009.

[32] S. Robertson and H. Zaragoza, “The Probabilistic Relevance Framework: BM25 and Beyond,” *Journal Foundations and Trends in Information Retrieval*, 3(4), pp. 333–389, 2009.

[33] P. Robinson and S. Bauer, *Introduction to Bio-Ontologies*, Chapman and Hall/CRC Press, 2011.

[34] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing and Management*, vol. 25, no 5, pp. 513–523, 1998.

[35] A.P. Santos and F. Rodrigues, “Multi-Label Hierarchical Text Classification Using the ACM Taxonomy,” Proceedings of 14th Portuguese Conference on Artificial Intelligence, (Aveiro, Portugal, October 12–15), pp. 553-564, 2010.

[36] F. Sebastiani, “Machine learning in automated text categorization,” *Journal of ACM Computing Surveys*, 1, 34, pp. 1-42, 2002.

[37] R.N. Shepard and P. Arabie, “Additive clustering: representation of similarities as combinations of discrete overlapping properties,” *Psychological Review*, vol. 86, pp. 87- 123, 1979.

[38] R. Snow, D. Jurafsky, and A.Y. Ng, Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 801-808, 2006.

[39] Song, Y., Liu, S., Wang, H., Wang, Z., & Li, H. “Automatic taxonomy construction from keywords,” U.S. Patent No. 9,501,569. Washington, DC: U.S. Patent and Trademark Office, 2016.

[40] A.R. de Soto, “A hierarchical model of a linguistic variable,” *Information Sciences*, vol. 181, pp. 4394-4408, 2011.

[41] M. Usman, R. Britto, J. Boerstler, and E. Mendes, “Taxonomies in software engineering: A Systematic mapping study and a revised taxonomy development method,” *Information and Software Technology*, 85, pp. 43-59, 2017.

[42] N. Vedula, P.K. Nicholson, D. Ajwani, S. Dutta, A. Sala, and S. Parthasarathy, "Enriching Taxonomies With Functional Domain Knowledge," In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, ACM, pp. 745-754, 2018.

[43] C. Wang, D.M. Blei, "Collaborative topic modeling for recommending scientific articles," In*Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 448-456, 2011.

[44] J. Waitelonis, C. Exeler, and H. Sack, "Linked data enabled generalized vector space model to improve document retrieval," In *Proceedings of NLP & DBpedia 2015 workshop in conjunction with 14th International Semantic Web Conference (ISWC)*, CEUR-WS, vol. 1486, 2015.

[45] Wang, C., He, X., & Zhou, A. "A Short Survey on Taxonomy Learning from Text Corpora: Issues, Resources and Recent Advances," In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1190-1203, 2017.

6 APPENDIX: Taxonomy of Data Science according to CCM-CCS 2012

The leaves added by the authors are labeled with a star "*".

Index	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
1.	Theory of computation					
1.1.		Theory and algorithms for application domains				
1.1.1.			Machine learning theory			
1.1.1.1.				Sample complexity and generalization bounds		
1.1.1.2.				Boolean function learning		
1.1.1.3.				Unsupervised learning and clustering		
1.1.1.4.				Kernel methods		
1.1.1.4.1.					Support vector machines	
1.1.1.4.2.					Gaussian processes	
1.1.1.4.3.*					Modelling	
1.1.1.5.				Boosting		
1.1.1.6.				Bayesian analysis		
1.1.1.7.				Inductive inference		
1.1.1.8.				Online learning theory		
1.1.1.9.				Multi-agent learning		
1.1.1.10.				Models of learning		
1.1.1.11.				Query learning		
1.1.1.12.				Structured prediction		
1.1.1.13.				Reinforcement learning		
1.1.1.13.1.					Sequential decision making	
1.1.1.13.2.					Inverse reinforcement learning	
1.1.1.13.3.					Apprenticeship learning	
1.1.1.13.4.					Multi-agent reinforcement learning	
1.1.1.13.5.					Adversarial learning	
1.1.1.14.				Active learning		
1.1.1.15.				Semi-supervised learning		
1.1.1.16.				Markov decision processes		
1.1.1.17.				Regret bounds		
1.1.2.			Database theory			
1.1.2.1.				Data exchange		
1.1.2.2.				Data provenance		
1.1.2.3.				Data modeling		
1.1.2.4.				Database query languages (principles)		
1.1.2.5.				Database constraints theory		
1.1.2.6.				Database interoperability		
1.1.2.7.				Data structures and algorithms for data management		

1.1.2.8.				Database query processing and optimization (theory)		
1.1.2.9.				Data integration		
1.1.2.10.				Logic and databases		
1.1.2.11.				Theory of database privacy and security		
1.1.2.12.				Incomplete, inconsistent, and uncertain databases		
2.	Mathematics of computing					
2.1.		Probability and statistics				
2.1.1.			Probabilistic representations			
2.1.1.1.				Bayesian networks		
2.1.1.2.				Markov networks		
2.1.1.3.				Factor graphs		
2.1.1.4.				Decision diagrams		
2.1.1.5.				Equational models		
2.1.1.6.				Causal networks		
2.1.1.7.				Stochastic differential equations		
2.1.1.8.				Nonparametric representations		
2.1.1.8.1.					Kernel density estimators	
2.1.1.8.2.					Spline models	
2.1.1.8.3.					Bayesian nonparametric models	
2.1.2.			Probabilistic inference problems			
2.1.2.1.				Maximum likelihood estimation		
2.1.2.2.				Bayesian computation		
2.1.2.3.				Computing most probable explanation		
2.1.2.4.				Hypothesis testing and confidence interval computation		
2.1.2.5.				Density estimation		
2.1.2.5.1.					Quantile regression	
2.1.2.6.				Max marginal computation		
2.1.3.			Probabilistic reasoning algorithms			
2.1.3.1.				Variable elimination		
2.1.3.2.				Loopy belief propagation		
2.1.3.3.				Variational methods		
2.1.3.4.				Expectation maximization		
2.1.3.5.				Markov-chain Monte Carlo methods		
2.1.3.5.1.					Gibbs sampling	
2.1.3.5.2.					Metropolis-Hastings algorithm	
2.1.3.5.3.					Simulated annealing	
2.1.3.5.4.					Markov-chain Monte Carlo convergence measures	
2.1.3.6.				Sequential Monte Carlo methods		
2.1.3.7.				Kalman filters and hidden Markov models		

2.1.3.7.1.*					Factorial HMM	
2.1.3.8.				Resampling methods		
2.1.3.8.1.					Bootstrapping	
2.1.3.8.2.					Jackknifing	
2.1.3.9.				Random number generation		
2.1.4.			Probabilistic algorithms			
2.1.5.			Statistical paradigms			
2.1.5.1.				Queueing theory		
2.1.5.2.				Contingency table analysis		
2.1.5.3.				Regression analysis		
2.1.5.3.1.					Robust regression	
2.1.5.4.				Time series analysis		
2.1.5.5.				Survival analysis		
2.1.5.6.				Renewal theory		
2.1.5.7.				Dimensionality reduction		
2.1.5.8.				Cluster analysis		
2.1.5.9.				Statistical graphics		
2.1.5.10.				Exploratory data analysis		
2.1.6.			Stochastic processes			
2.1.6.1.				Markov processes		
2.1.7.			Nonparametric statistics			
2.1.8.			Distribution functions			
2.1.9.			Multivariate statistics			
3.	Information systems					
3.1.		Data management systems				
3.1.1.			Database design and models			
3.1.1.1.				Relational database model		
3.1.1.2.				Entity relationship models		
3.1.1.3.				Graph-based database models		
3.1.1.3.1.					Hierarchical data models	
3.1.1.3.2.					Network data models	
3.1.1.4.				Physical data models		
3.1.1.5.				Data model extensions		
3.1.1.5.1.					Semi-structured data	
3.1.1.5.2.					Data streams	
3.1.1.5.3.					Data provenance	
3.1.1.5.4.					Incomplete data	
3.1.1.5.5.					Temporal data	
3.1.1.5.6.					Uncertainty	
3.1.1.5.7.					Inconsistent data	
3.1.2.			Data structures			
3.1.2.1.				Data access methods		
3.1.2.1.1.					Multidimensional range search	
3.1.2.1.2.					Data scans	
3.1.2.1.3.					Point lookups	
3.1.2.1.4.					Unidimensional range search	
3.1.2.1.5.					Proximity search	

3.1.2.2.				Data layout		
3.1.2.2.1.					Data compression	
3.1.2.2.2.					Data encryption	
3.1.2.2.3.					Record and block layout	
3.1.3.			Database management system engines			
3.1.3.1.				DBMS engine architectures		
3.1.3.2.				Database query processing		
3.1.3.2.1.					Query optimization	
3.1.3.2.2.					Query operators	
3.1.3.2.3.					Query planning	
3.1.3.2.3.					Join algorithms	
3.1.3.3.				Database transaction processing		
3.1.3.3.1.					Data locking	
3.1.3.3.2.					Transaction logging	
3.1.3.3.3.					Database recovery	
3.1.3.4.				Record and buffer management		
3.1.3.5.				Parallel and distributed DBMSs		
3.1.3.5.1.					Key-value stores	
3.1.3.5.2.					MapReduce-based systems	
3.1.3.5.3.					Relational parallel and distributed DBMSs	
3.1.3.6.				Triggers and rules		
3.1.3.7.				Database views		
3.1.3.8.				Integrity checking		
3.1.3.9.				Distributed database transactions		
3.1.3.9.1.					Distributed data locking	
3.1.3.9.2.					Deadlocks	
3.1.3.9.3.					Distributed database recovery	
3.1.3.10.				Main memory engines		
3.1.3.11.				Online analytical processing engines		
3.1.3.12.				Stream management		
3.1.4.			Query languages			
3.1.4.1.				Relational database query languages		
3.1.4.1.1.					Structured Query Language	
3.1.4.2.				XML query languages		
3.1.4.2.1					XPath	
3.1.4.2.2.					XQuery	
3.1.4.3.				Query languages for non-relational engines		
3.1.4.3.1.					MapReduce languages	
3.1.4.4.				Call level interfaces		
3.1.5.			Information integration			
3.1.5.1.				Deduplication		
3.1.5.2.				Extraction transformation and loading		
3.1.5.3.				Data exchange		
3.1.5.4.				Data cleaning		
3.1.5.5.				Wrappers (data mining)		
3.1.5.6.				Mediators and data integration		
3.1.5.7.				Entity resolution		
3.1.5.8.				Data warehouses		

3.1.5.9.				Federated databases		
3.2.		Information systems applications				
3.2.1.			Data mining			
3.2.1.1.				Data cleaning		
3.2.1.2.				Collaborative filtering		
3.2.1.2.1.*					Item-based	
3.2.1.2.2.*					Scalable	
3.2.1.3.				Association rules		
3.2.1.3.1.*					Types of association rules	
3.2.1.3.2.*					Interestingness	
3.2.1.3.3.*					Parallel computation	
3.2.1.4.				Clustering		
3.2.1.4.1.*					Massive data clustering	
3.2.1.4.2.*					Consensus clustering	
3.2.1.4.3.*					Fuzzy clustering	
3.2.1.4.4.*					Additive clustering	
3.2.1.4.5.*					Feature weight clustering	
3.2.1.4.6.*					Conceptual clustering	
3.2.1.4.7.*					Biclustering	
3.2.1.5.				Nearest-neighbor search		
3.2.1.6.				Data stream mining		
3.2.1.7.*				Graph mining		
3.2.1.7.1.*					Graph partitioning	
3.2.1.7.2.*					Frequent graph mining	
3.2.1.7.3.*					Graph based conceptual clustering	
3.2.1.7.4.*					Anomaly detection	
3.2.1.7.5.*					Critical nodes detection	
3.2.1.8.*				Process mining		
3.2.1.11.*				Text mining		
3.2.1.11.1.*					Text categorization	
3.2.1.11.2.*					Key-phrase indexing	
3.2.1.10.*				Data mining tools		
3.2.1.9.*				Sequence mining		
3.2.1.9.1.*					Rule and pattern discovery	
3.2.1.9.2.*					Trajectory clustering	
3.2.1.9.3.*					Market graph	
3.2.1.12.*				Formal concept analysis		
3.3.		World Wide Web				
3.3.1.			Web mining			
3.3.1.2.				Site wrapping		
3.3.1.3.				Data extraction and integration		
3.3.1.3.1					Deep web	
3.3.1.3.2.					Surfacing	
3.3.1.3.3.					Search results deduplication	
3.3.1.4.				Web log analysis		
3.3.1.5.				Traffic analysis		
3.3.1.6.*				Knowledge discovery		
3.4.		Information retrieval				
3.4.1.			Document representation			
3.4.1.1.				Document structure		

3.4.1.2.				Document topic models		
3.4.1.3.				Content analysis and feature selection		
3.4.1.4.				Data encoding and canonicalization		
3.4.1.5.				Document collection models		
3.4.1.6.				Ontologies		
3.4.1.7.				Dictionaries		
3.4.1.8.				Thesauri		
3.4.2.			Information retrieval query processing			
3.4.2.1.				Query representation		
3.4.2.2.				Query intent		
3.4.2.3.				Query log analysis		
3.4.2.4.				Query suggestion		
3.4.2.5.				Query reformulation		
3.4.3.			Users and interactive retrieval			
3.4.3.1.				Personalization		
3.4.3.2.				Task models		
3.4.3.3.				Search interfaces		
3.4.3.4.				Collaborative search		
3.4.4.			Retrieval models and ranking			
3.4.4.1.				Rank aggregation		
3.4.4.2.				Probabilistic retrieval models		
3.4.4.3.				Language models		
3.4.4.4.				Similarity measures		
3.4.4.5.				Learning to rank		
3.4.4.6.				Combination fusion and federated search		
3.4.4.7.				Information retrieval diversity		
3.4.4.8.				Top-k retrieval in databases		
3.4.4.9.				Novelty in information retrieval		
3.4.5.			Retrieval tasks and goals			
3.4.5.1.				Question answering		
3.4.5.2.				Document filtering		
3.4.5.3.				Recommender systems		
3.4.5.4.				Information extraction		
3.4.5.5.				Sentiment analysis		
3.4.5.6.				Expert search		
3.4.5.7.				Near-duplicate and plagiarism detection		
3.4.5.8.				Clustering and classification		
3.4.5.9.				Summarization		
3.4.5.10.				Business intelligence		
3.4.6.			Evaluation of retrieval results			
3.4.6.1.				Test collections		
3.4.6.2.				Relevance assessment		
3.4.6.3.				Retrieval effectiveness		
3.4.6.4.				Retrieval efficiency		
3.4.6.5.				Presentation of retrieval results		
3.4.7.			Specialized information retrieval			

3.4.7.1.				Structure and multilingual text search		
3.4.7.1.1.					Structured text search	
3.4.7.1.2.					Mathematics retrieval	
3.4.7.1.3.					Chemical and biochemical retrieval	
3.4.7.1.4.					Multilingual and cross-lingual retrieval	
3.4.7.2.				Multimedia and multimodal retrieval		
3.4.7.2.1.					Image search	
3.4.7.2.2.					Video search	
3.4.7.2.3.					Speech / audio search	
3.4.7.2.4.					Music retrieval	
3.4.7.3.				Environment-specific retrieval		
3.4.7.3.1.					Enterprise search	
3.4.7.3.2.					Desktop search	
3.4.7.3.3.					Web and social media search	
4.	Human-centered computing					
4.1.		Visualization				
4.1.2.			Visualization techniques			
4.1.2.1.				Treemaps		
4.1.2.2.				Hyperbolic trees		
4.1.2.3.				Heat maps		
4.1.2.4.				Graph drawings		
4.1.2.5.				Dendrograms		
4.1.2.6.				Cladograms		
4.1.2.7.*				Elastic maps		
4.1.3.			Visualization application domains			
4.1.3.1.				Scientific visualization		
4.1.3.2.				Visual analytics		
4.1.3.3.				Geographic visualization		
4.1.3.4.				Information visualization		
4.1.4.			Visualization systems and tools			
4.1.4.1.				Visualization toolkits		
4.1.5.			Visualization theory concepts and paradigms			
4.1.6.			Empirical studies in visualization			
4.1.7.			Visualization design and evaluation methods			
5.	Computing methodologies					

5.1.		Artificial intelligence				
5.1.1.			Natural language processing			
5.1.1.2.				Information extraction		
5.1.1.3.				Machine translation		
5.1.1.4.				Discourse dialogue and pragmatics		
5.1.1.5.				Natural language generation		
5.1.1.6.				Speech recognition		
5.1.1.7.				Lexical semantics		
5.1.1.7.1.*					Wikipedia based semantics	
5.1.1.8.				Phonology / morphology		
5.1.1.9.				Language resources		
5.1.2.			Knowledge representation and reasoning			
5.1.2.1.				Description logics		
5.1.2.2.				Semantic networks		
5.1.2.3.				Nonmonotonic, default reasoning and belief revision		
5.1.2.4.				Probabilistic reasoning		
5.1.2.5.				Vagueness and fuzzy logic		
5.1.2.6.				Causal reasoning and diagnostics		
5.1.2.7.				Temporal reasoning		
5.1.2.8.				Cognitive robotics		
5.1.2.9.				Ontology engineering		
5.1.2.10.				Logic programming and answer set programming		
5.1.2.11.				Spatial and physical reasoning		
5.1.2.12.				Reasoning about belief and knowledge		
5.1.3.			Computer vision			
5.1.3.1.				Computer vision problems		
5.1.3.1.1.					Interest point and salient region detections	
5.1.3.1.2.					Image segmentation	
5.1.3.1.3.					Video segmentation	
5.1.3.1.4.					Shape inference	
5.1.3.1.5.					Object detection	
5.1.3.1.6.					Object recognition	
5.1.3.1.7.					Object identification	
5.1.3.1.8.					Tracking	
5.1.3.1.9.					Reconstruction	
5.1.3.1.10.					Matching	
5.1.3.2.				Computer vision representations		
5.1.3.2.1.					Image representations	
5.1.3.2.1.1.*						2D PCA
5.1.3.2.2.					Shape representations	
5.1.3.2.3.					Appearance and texture representations	
5.1.3.2.4.					Hierarchical representations	
5.2.		Machine learning				
5.2.1.			Learning paradigms			
5.2.1.1.				Supervised learning		

5.2.1.1.1.					Ranking	
5.2.1.1.2.					Learning to rank	
5.2.1.1.3.					Supervised learning by classification	
5.2.1.1.4.					Supervised learning by regression	
5.2.1.1.5.					Structured outputs	
5.2.1.1.6.					Cost-sensitive learning	
5.2.1.2.				Unsupervised learning		
5.2.1.2.1.					Cluster analysis	
5.2.1.2.2.					Anomaly detection	
5.2.1.2.3.					Mixture modeling	
5.2.1.2.4.					Topic modeling	
5.2.1.2.5.					Source separation	
5.2.1.2.6.					Motif discovery	
5.2.1.2.7.					Dimensionality reduction and manifold learning	
5.2.1.2.7.1.*						Graph embedding
5.2.1.2.7.2.*						Supervised dimesionality reduction
5.2.1.3.				Reinforcement learning		
5.2.1.3.1.					Sequential decision making	
5.2.1.3.2.					Inverse reinforcement learning	
5.2.1.3.3.					Apprenticeship learning	
5.2.1.3.4.					Multi-agent reinforcement learning	
5.2.1.3.5.					Adversarial learning	
5.2.1.4.				Multi-task learning		
5.2.1.4.1.					Transfer learning	
5.2.1.4.2.					Lifelong machine learning	
5.2.1.4.3.					Learning under covariate shift	
5.2.2.			Learning settings			
5.2.2.1.				Batch learning		
5.2.2.2.				Online learning settings		
5.2.2.3.				Learning from demonstrations		
5.2.2.4.				Learning from critiques		
5.2.2.5.				Learning from implicit feedback		
5.2.2.6.				Active learning settings		
5.2.2.7.				Semi-supervised learning settings		
5.2.2.7.1.*					Kernel approach	
5.2.3.			Machine learning approaches			
5.2.3.1.				Classification and regression trees		
5.2.3.1.1.*					Parallel implementation	
5.2.3.1.2.*					Splitting criteria	
5.2.3.1.3.*					Model trees	
5.2.3.2.				Kernel methods		
5.2.3.2.1.*					Kernel support vector machines	
5.2.3.2.1.1.*						Dynamic
5.2.3.2.2.					Gaussian processes	
5.2.3.2.3.*					Kernel Matrix	
5.2.3.2.4.*					Kernel Independent components	

5.2.3.2.5.*					Kernel-based clustering	
5.2.3.3.				Neural networks		
5.2.3.3.1.*					Self organized map	
5.2.3.3.2.*					Training approaches	
5.2.3.3.2.1.*						Evolutionary approach
5.2.3.3.3.*					Representation	
5.2.3.3.3.1.*						Rule-based netwok archirtecture
5.2.3.3.3.2.*						Fuzzy representation
5.2.3.3.4.*					Evolving NN	
5.2.3.3.5.*					Ensembling	
5.2.3.4.				Logical and relational learning		
5.2.3.4.1.					Inductive logic learning	
5.2.3.4.2.					Statistical relational learning	
5.2.3.5.				Learning in probabilistic graphical models		
5.2.3.5.1.					Maximum likelihood modeling	
5.2.3.5.2.					Maximum entropy modeling	
5.2.3.5.3.					Maximum a posteriori modeling	
5.2.3.5.4.					Mixture models	
5.2.3.5.5.					Latent variable models	
5.2.3.5.6.					Bayesian network models	
5.2.3.5.7.*					Markov network models	
5.2.3.6.				Learning linear models		
5.2.3.6.1.					Perceptron algorithm	
5.2.3.6.2.*					Linear Discriminant Analysis	
5.2.3.6.2.1.*						Tensor representation
5.2.3.7.				Factorization methods		
5.2.3.7.1.					Non-negative matrix factorization	
5.2.3.7.2.					Factor analysis	
5.2.3.7.3.					Principal component analysis	
5.2.3.7.3.1.*						2D PCA
5.2.3.7.3.2.*						Sparse PCA
5.2.3.7.4.					Canonical correlation analysis	
5.2.3.7.6.					Latent Dirichlet allocation	
5.2.3.7.8.*					Independent Component Analysis	
5.2.3.7.9.*					Nonlinear Principal Components	
5.2.3.7.10.*					Multidimentional scaling	
5.2.3.7.10.1.*						Least moduli
5.2.3.8.				Rule learning		
5.2.3.8.1.*					Neuro-fuzzy approach	
5.2.3.9.				Instance-based learning		
5.2.3.10.				Markov decision processes		
5.2.3.11.				Partially-observable Markov decision processes		
5.2.3.12.				Stochastic games		
5.2.3.13.				Learning latent representations		

5.2.3.13.1.					Deep belief networks	
5.2.3.14.*				Multiresolution		
5.2.3.15.*				Support vector machines		
5.2.4.			Machine learning algorithms			
5.2.4.1.				Dynamic programming for Markov decision processes		
5.2.4.1.1.					Value iteration	
5.2.4.1.2.					Q-learning	
5.2.4.1.3.					Policy iteration	
5.2.4.1.4.					Temporal difference learning	
5.2.4.1.5.					Approximate dynamic programming methods	
5.2.4.2.				Ensemble methods		
5.2.4.2.1.					Boosting	
5.2.4.2.2.					Bagging	
5.2.4.2.3.*					Fusion of classifiers	
5.2.4.3.				Spectral methods		
5.2.4.3.1.*					Spectral clustering	
5.2.4.4.				Feature selection		
5.2.4.5.				Regularization		
5.2.4.5.1.*					Generalized eigenvalue	
5.2.5.			Cross-validation			