Using Taxonomy Tree to Generalize a Fuzzy Thematic Cluster

Dmitry Frolov
Dept. of Data Analysis and AI,
HSE University
Moscow, Russian Federation
dfrolov@hse.ru

Susana Nascimento
Dept. of Computer Science
and NOVA LINCS,
Universidade Nova
de Lisboa
Lisbon, Portugal
snt@fct.unl.pt

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

Boris Mirkin
Dept. of Data Analysis and AI,
HSE University
Moscow, Russian Federation
Dept. of Computer Science,
Birkbeck University of London
London, UK
mirkin@dcs.bbk.ac.uk

***Abstract*—This paper presents an algorithm, ParGenFS, for generalizing, or "lifting", a fuzzy set of topics to higher ranks of a hierarchical taxonomy of a research domain. The algorithm ParGenFS finds a globally optimal generalization of the topic set to minimize a penalty function, by balancing the number of introduced "head subjects" and related errors, the "gaps" and "offshoots", differently weighted. This leads to a generalization of the topic set in the taxonomy. The usefulness of the method is illustrated on a set of 17685 abstracts of research papers on Data Science published in Springer journals for the past 20 years. We extracted a taxonomy of Data Science from the international Association for Computing Machinery Computing Classification System 2012 (ACM-CCS). We find fuzzy clusters of leaf topics over the text collection, lift them in the taxonomy, and interpret found head subjects to comment on the tendencies of current research.**

Keywords–generalization, gap-offshoot penalty, fuzzy cluster, annotated suffix tree

I. Introduction

We propose a method for structurization and generalization of research text collections over a taxonomy of the field, specifically, the ACM Computing Classification System (ACM-CCS 2012) [1]. The method, Parsimonious Generalization of Fuzzy Sets (ParGenFS), is applied to generalize fuzzy topic clusters extracted from a collection of 17685 abstract of research papers from Springer journals.

The existing approaches to computational analysis of structure of text collections are cluster analysis and topic modeling. Both approaches consider items of the same level of granularity as individual words or short phrases in the texts, involving no generalization as a specific goal. However, the hierarchical nature of the domain of text semantic analysis is reflected in some publications. We can distinguish, at least, three directions at which the matter of generalization is addressed.

First, there are activities related to developing and exploring hierarchical taxonomies which are indeed a form of knowledge engineering and become widely popular (e.g., Genome Ontology (GO) project [3], SNOMED CT project [4] and the like). Some papers explicitly explore hyponymic/hypernymic relations (e.g. [5], [6], and references therein). A recent work [7] is devoted to supplementing a taxonomy with newly emerging research topics. A second direction is part of conventional activities in text summarization. Usually, summaries are created using a rather mechanistic approach of sentence extraction. There is also an approach for building summaries as abstractions of texts by combining some templates such as Subject-Verb-Object (SVO) triplets, e.g. [8]. The third type of approach is what can be referred to as "operational" generalization. In this case, 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, e.g., [9], [10].

The present work presents a novel approach, to the best of our knowledge. We use the concept of taxonomy for straightforwardly modeling the concept of generalization. According to the Merriam-Webster dictionary, the term "generalization" refers to deriving a general conception from particulars. We assume that a most straightforward medium for such a derivation, a domain taxonomy, is given to us. We address the problem of generalize a fuzzy set of taxonomy leaves representing the essence of some empirically observed phenomenon over a domain taxonomy. We consider the most popular Computer Science taxonomy manually developed by the world-wide Association for Computing Machinery, 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 [11].

The methodology adopted in this work involves the following steps:

- preparing a scholarly text collection;
- preparing a taxonomy of the domain under consideration;
- developing a matrix of score values expressing the relevance between taxonomy leaf topics and research publications from the collection;
- finding fuzzy clusters of leaf topics according to the structure of score values to express hidden directions of research according to the text collection;
- lifting the clusters over the taxonomy to conceptualize them via generalization;
- interpreting the generalizations in the knowledge domain.

The rest of the paper is organized as follows: Section 2

978-1-5386-1728-1/19/$31.00 ©2019 IEEE

presents a mathematical formalization of the generalization problem as of parsimoniously lifting of a given fuzzy leaf set to higher ranks of the taxonomy. Section 3 provides a recursive algorithm, ParGenFS, whose output is a globally optimal solution to the problem. Section 4 describes an application of the algorithm for deriving tendencies in development of Data Science discerned from a collection of 17685 research papers published by the Springer Publishers. It describes a multi-step technology for finding and generalizing fuzzy clusters of research topics according to the collection. The paper concludes by describing the tendencies in the development of corresponding parts of Data Science derived from the generalization results.

II. A METHOD FOR PARSIMONIOUSLY LIFTING FUZZY CLUSTER IN A DOMAIN TAXONOMY

A taxonomy is a rooted tree, a hierarchy in which each node has only one parent, whose nodes are annotated by taxonomy topics.

Given a fuzzy set S of taxonomy leaves, we are interested in finding a node $t(S)$ of a higher rank in the taxonomy, that covers the set S 'tightly'. Such a "lifting" approach is a mathematical explication of the idea of generalization for a phenomenon represented by a fuzzy leaf subset.

This problem is not that simple. Consider, for the sake of simplicity, a crisp set S shown with five black leaf nodes on a fragment of a tree as illustrated in Fig. 1. Fig. 1 shows the situation at which the set of black leaf nodes is lifted to the root, which is highlighted with black nodes of the root and its offspring, too. In this case, there are four white nodes covered by the root and, thus, falling in the same concept as S even as they do not belong in S. Such a mismatch is referred to as a *gap*. Gaps should be penalized. Altogether, the number of conceptual elements introduced to generalize S here is 1 head subject (the root), and 4 gaps occurring due to the topology of the tree.

Another lifting scenario is illustrated in Fig. 2: here the set S of leaf nodes is lifted just to the root of the left branch of the tree. In this case the number of gaps drastically decreases, to just 1. However, another type of mismatch occurs: a black node on the right, belonging to S but not covered by the root of the left branch. 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 (see Fig. 1) which makes it preferable. This conclusion, however, holds only if the relative penalty for an offshoot is less than the total relative penalty for three gaps.

The goal of finding a pigeon-hole for S within the taxonomy can be formalized as that of finding one or more "head subjects" 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) which is popular in Bioinformatics and some other domains.

Let T be a rooted tree representing a hierarchical taxonomy so that its nodes are annotated with key phrases signifying various concepts, and let I denote the set of all its leaves. Each node $t \in T$ is said to be the *parent* of the nodes immediately descending from t in T, its *children*. Let $\chi(t)$ 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 referred to as the *leaf cluster of* t.

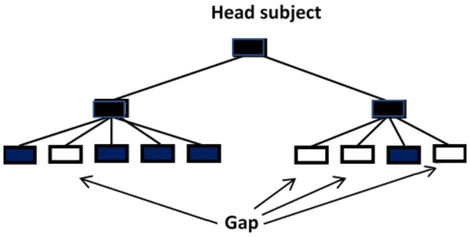

Figure 1. Generalization of a query set defined by the black leaf nodes, by lifting it to the root, with the price of four gaps emerged at the lift.

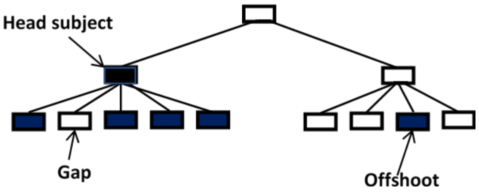

Figure 2. Generalization of a query set defined by the black leaf nodes, by lifting it to the root of the left branch, with the price of one gap and one offshoot emerged at this lift.

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 $i \in I$. The set $S_u \subset I$, where $S_u = \{i \in I : u(i) > 0\}$, is referred as the *base* of u. As is commonly assumed, function u does not exceed unity.

Given a fuzzy set u defined on the leaves I of the tree T, 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. They are illustrated in Fig. 1 and in Fig. 2. Given a head subject node h, a gap is a node t covered by h but not belonging to u, so that $u(t) = 0$. In contrast, an offshoot is a node t belonging to S_u, so that $u(t) > 0$ but not covered by the h.

To warrant that the total number of head subjects, gaps, and offshoots are as small as possible we introduce a penalty for each of these elements. Assuming that the black node leaves on Fig. 1 have membership function values equal to unity, one can easily see that the total penalty at the head subject raised to the root on Fig. 1 is equal to $1 + 4\lambda$ where 1 is the penalty for a head subject and λ, the penalty for a gap. Similarly, the penalty for the lift on Fig. 2 to the root of the left-side subtree is equal to $1 + \gamma + \lambda$ where γ is the penalty for an offshoot. Therefore, depending on the relationship between γ and $3 * \lambda$ either lift on Fig. 1 or lift on Fig. 2 is to be chosen.

We refer to node $t \in T$ as to *u-irrelevant* if its leaf-cluster $I(t)$ is disjoint from the base S_u. Naturally, if a node is u-irrelevant, all of its descendants are also u-irrelevant. Consider a candidate head subject 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 are denoted by $G(h)$.

A gap is less significant if its parent's membership value is smaller. We define a measure of *gap importance* as $v(g) = u(par(g))$, where $par(g)$ is the parent of g. In fact, the algorithm ParGenFS below works for any definition of gap importance. Also, we define a summary gap importance: $V(h) = \sum_{g \in G(h)} v(g)$.

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 $S_u - I(h)$.

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

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

The problem is to find a u-cover H that globally minimizes the penalty $p(H)$. Such a u-cover is a most appropriate generalization of the fuzzy set u.

III. The ParGenFS algorithm for parsimonious generalization of fuzzy sets

Before applying an algorithm to minimize the total penalty, let us prune the tree 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 (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 $u(t)$. Our algorithm does not depend on the way the u-values are assigned, provided the value of $u(t)$ is zero for all u-irrelevant nodes t. In practical computations, we aggregate u-values on the leaves according to the fuzzy membership constraints [11].

After this, the Parsimonious Generalization of Fuzzy Sets (ParGenFS) algorithm applies, as follows. 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 $p(t)$ is computed too as 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. 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.

A pseudo-code description of the algorithm, as well as a demonstration that the algorithm leads to an optimal lifting, are in [11].

IV. Application to the analysis of a collection of papers over the ACM-CCS taxonomy

A set of 17685 research papers published in 17 Springer journals related to Data Science for 20 years (1998-2017) were collected for this study [11]. Specifically, the abstracts to these papers were the object of our analysis. 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. This taxonomy of Data Science, over 317 leaves, can be found in [11]. We find fuzzy clusters of leaf topics according to their relevance to the texts.

Most popular and well established approaches to scoring keyphrase-to-document relevance include the so-called vector-space approach [12] and probabilistic text model approach [13]. These, however, rely on individual words and manual text pre-processing. We apply a different approach, the Annotated Suffix Tree (AST) method by [14], [15] by using purely string frequency information.

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

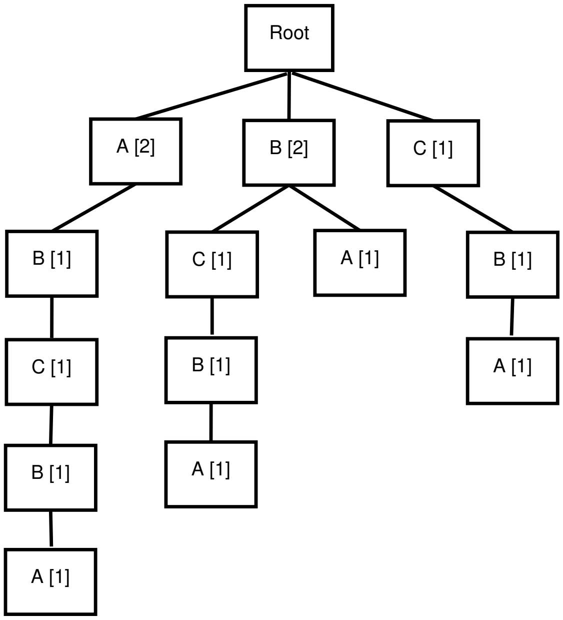

Figure 3. An example: AST for string ‘ABCBA’.

extracted. A k-suffix of a string $x = x_1 x_2 \dots x_N$ of length N is a continuous end fragment $x^k = x_{N-k+1} x_{N-k+2} \dots x_N$. Each AST node is annotated by a symbol and the frequency of the substring corresponding to the path from the root to the node including the symbol at the node, referred annotation. The root node of AST has no symbol or annotation. The used algorithm [14], [15] for building an AST for any given string $x = x_1 x_2 \dots x_N$ can be described as follows:

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.

An example of AST built for a string ‘ABCBA’ is presented at Figure 3.

Having an AST T built, the string-to-document relevance is scored over the AST by combining the conditional probabilities of nodes u in T as in [15]:

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

For all the immediate offspring of the root (R):

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

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 mean (or sum) of all the suffix scores:

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

Let us calculate the relevance score of a string ‘ABDC’ according to the AST for string ‘ABCBA’ in Figure 3. The string has four suffixes: ‘ABDC’, ‘BDC’, ‘DC’, ‘C’. First of all, one needs to calculate relevance scores for these suffixes according to formula (6). These scores are presented in Table I. After that, according to formula (7), we calculate score for the whole string:

$$S(\text{‘ABDC’}, T) = 1/4 \cdot (0.7 + 0.4 + 0 + 0.2) = 0.325$$

TABLE I. COMPUTING THE RELEVANCE SCORES FOR SUFFIXES OF A STRING ‘ABDC’.

Suffix	Match	Score
‘ABDC’	‘AB’	$\frac{1}{2} \cdot \left(\frac{2}{5} + \frac{1}{1}\right) = 0.7$
‘BDC’	‘B’	$\frac{1}{1} \cdot \left(\frac{2}{5}\right) = 0.4$
‘DC’	‘’	0
‘C’	‘C’	$\frac{1}{1} \cdot \left(\frac{1}{5}\right) = 0.2$

In practice, we split any document into a set of strings consisting of 2-3 consecutive words, create an empty AST for the document, and add these strings in the AST one-by-one 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 such as “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. An advantage of the AST method is that it does not depend on the document length, in contrast to the popular Levenstein-distance based approaches.

Let us denote topic-to-text relevance 17645×317 matrix by $P = (p_{ij})$. Each (p_{ij}) is an AST relevance score between the document i and the topic j.

Fuzzy topic clusters should reflect co-occurrence of topics: the greater the number of texts to which both t and t' topics 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. Therefore, a 317×317 topic-to-topic co-relevance matrix is defined as $P'MP$ where M is a diagonal matrix with (i, i)-th entry equal to the proportion of topics whose relevance value to i-th text is 0.2 or greater (a threshold found experimentally).

To find leaf-topic clusters, we applied an additive fuzzy spectral method, FADDIS [17], specifically developed for this.

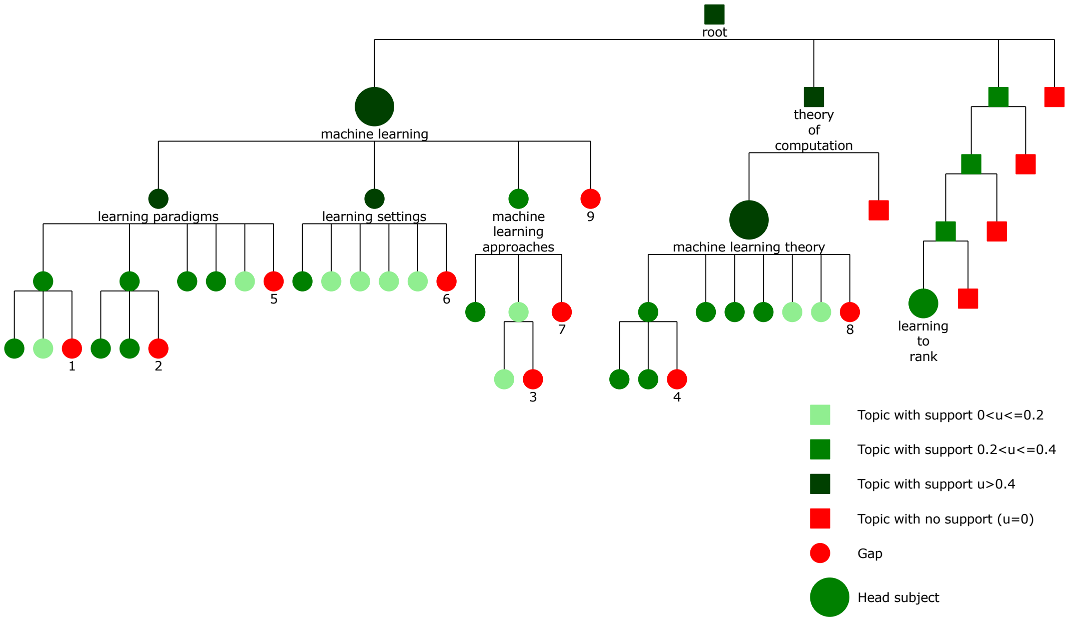

Figure 4. Lifting results for Cluster L: Learning. Gaps are numbered, see Table III.

The method operates on an extension of the spectral decomposition of a square similarity matrix and extracts clusters one by one, which allows us to draw several natural criteria for halting the process of extraction, thus appropriately defining the number of clusters. The FADDIS algorithm implements aspects that are relevant to this task. Specifically: (a) Laplacian Pseudo-Inverse Normalization (Lapin): Similarity data transformation, modeling – to an extent – heat distribution and, in this way, making the cluster structure sharper; (b) Additivity: Thematic clusters behind the texts are additive, so that similarity values are sums of contributions by different hidden themes; and (c) 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.

After computing the 317×317 topic-to-topic co-relevance matrix, converted to a topic-to-topic Lapin transformed similarity matrix, and applying FADDIS clustering, we sequentially obtained six clusters, of which three clusters are obviously homogeneous, relating, in respect, to 'Learning' (L), 'Retrieval' (R), and 'Clustering' (C). These clusters are presented in Table II.

The clusters L, R, and C are lifted in the DST taxonomy using ParGenFS algorithm with the gap penalty $\lambda = 0.1$ and off-shoot penalty $\gamma = 0.9$. These values reflect the desired limits to the gap / head-subject rate.

The results of lifting of Cluster L are shown in Fig. 4. There are three head subjects: Machine Learning, Machine Learning Theory, and Learning to Rank. These represent the structure of the general concept "Learning" according to the text collection under consideration. One can see from the "Learning" head subjects (see Fig. 4 and comments to it) that main work here still concentrates on theory and method rather than applications. An interesting aspect 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. However, there remain many sub-areas to be covered as seen in the list of gaps in Table III.

TABLE II. CLUSTERS L, R, C: TOPICS WITH LARGEST MEMBERSHIP VALUES.

$u(t)$	Code	Topic Cluster L
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
…	…	…

$u(t)$	Code	Topic Cluster R
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
…	…	…

$u(t)$	Code	Topic Cluster C
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
…	…	…

At lifting of Cluster R: Retrieval, we obtained head subjects: Information Systems and Computer Vision. They show the structure of "Retrieval" in the set of publications under considerations, reflecting the ever increasing role of images and their elements in the subject. 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

TABLE III. 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

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.

The lifting of Cluster C leads to 16 head subjects: clustering, graph based conceptual clustering, 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 at clustering – this shows that the ever increasing role of clustering activities should be better reflected in the taxonomy by inserting a corresponding node in the higher ranks. 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 knowledge engineering.

It should be pointed out that research in tendencies of science development is carried out by several groups using co-citation data, especially in dynamics (see, for example, a review in [2]). That approach leads to conclusions involving individual, rather than general, authors and/or papers, and, therefore, is complementary to our approach.

ACKNOWLEDGMENT

D.F. and B.M. acknowledge continuing support by the Academic Fund Program at the National Research University Higher School of Economics (grant 19-04-019 in 2018-2019) 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 the Russian Academic Excellence Project “5-100”. S.N. acknowledges the support by FCT/MCTES, NOVA LINCS (UID/CEC/04516/2019).

REFERENCES

[1] The 2012 ACM Computing Classification System. [Online]. Available:*http://www.acm.org* /about/class/2012 (Accessed 2019, 18 January).

[2] C. Chen, “Science mapping: A systematic review of the literature”, Journal of Data and Information Science, vol. 2, no. 2, pp. 1–40, 2017.

[3] Gene Ontology Consortium, “Gene ontology consortium: going forward,” Nucleic Acids Research, vol. 43 D1049-D1056, 2015.

[4] D. Lee, R. Cornet, F. Lau, and N. De Keizer, “A survey of SNOMED CT implementations,” Journal of Biomedical Informatics, 46(1), pp. 87-96, 2013.

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

[6] C. Wang, X. He, and A. Zhou, “A Short Survey on Taxonomy Learning from Text Corpora: Issues, Resources and Recent Advances,” in Procs. of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 1190-1203.

[7] 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, 2018, pp. 745-754.

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

[9] G. Mueller, and R. Bergmann, “Generalization of Workflows in Process-Oriented Case-Based Reasoning,” in Procs. of The Twenty-Eighth International Flairs (FLAIRS) Conference, AAAI Pub., 2015, pp. 391-396.

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

[11] D. Frolov, B. Mirkin, S. Nascimento, and T. Fenner, “Finding an appropriate generalization for a fuzzy thematic set in taxonomy,” Working paper WP7/2018/04, Moscow, Higher School of Economics Publ. House, 2018, 60 p. (see https://wp.hse.ru/en/wp7en).

[12] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” Information Processing and Management, 25(5), pp. 513–523, 1998.

[13] D. Blei, “Probabilistic topic models,” Communications of the ACM, 55(4), pp. 77–84, 2012.

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

[15] 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.

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

[17] B. Mirkin, S. Nascimento, “Additive spectral method for fuzzy cluster analysis of similarity data including community structure and affinity matrices,” Information Sciences, 183(1), pp. 16-34, 2012.