Abstracting Concepts from Text Documents by Using an Ontology

E.Chernyak[1], O.Chugunova[1], J.Askarova[1], S. Nascimento[2], B.Mirkin[1,3]

[1]Division of Applied Mathematics and Informatics, National Research University Higher School of Economics, Moscow, Russian Federation
[2]Department of Informatics, New University of Lisbon, Caparica, Portugal
[3]Department of Computer Science, Birkbeck University of London, London, UK

Abstract. A method for computationally visualizing and interpreting a text or corpus of texts in a taxonomy of the field is described. The method involves such stages as matching taxonomy topics and text(s) by using annotated suffix trees (ASTs), combining multiple information such as text abstracts, key-words and taxonomy cross-references, building clusters of taxonomy topics and their profiles, and lifting the profiles to higher ranks of the taxonomy hierarchy.

1 Introduction

The concept of ontology as a computational device for handling domain knowledge is one of the points of growing interest in machine intelligence. Initially main efforts of the researchers concentrated on building ontologies; currently the research interests are shifting towards the usage of ontologies (see, for example, [5], [6], [10], [4]). This paper is aimed at the latter perspective. Our ultimate goal is to devise a system that would allow the user to use a domain ontology for computational interpretation of a text or a set of texts from this field. The paper presents some initial stages of our work on the long path towards achieving the goal. These stages include the following: (a) selection of the conceptual hierarchy (taxonomy) as a formalization of the concept of ontology, (b) representation of both texts and taxonomy topics in a unified framework that facilitates and channels the sifting of the taxonomy topics through the texts to score matches between them in a comprehensive way, (c) developing quantitative profiles of the texts, (d) clustering them in a way that does not require much input from the user, and, in the very end, (e) lifting the profiles of clusters or individual texts to higher ranks of the hierarchy to visualize and interpret them.

When applying this approach to texts in Russian, additional issues emerge due to lack of adequate tools for both linguistic analysis and taxonomy development in Cyrillic alphabet.

The remainder describes the techniques that are being under development and, in part, is an adaptation of a method in [5]. Our preliminary attempts at applying the techniques to real data are described too. The conclusion states what has been already done and issues yet to be tackled.

This paper comprises research findings obtained in the framework of the research project "Development and adaptation of clustering methods to automate analysis of unstructured texts using domain ontologies" supported by The NRU Higher School of Economics 2011-2012 Academic Fund Program. The project was partly supported by the Program of Fundamental Studies of the NRU Higher School of Economics in 2011.

2 Method's description

2.1 Input information

There are two inputs to the method: (1) a domain ontology and (2) a domain related text collection.

We consider an ontology to be a rooted tree-like structure of topics in the domain, with the parental nodes corresponding to more general topics than the children. Besides the hierarchical relation between the topics, other relations might exist. There can be links between topics from different parts of the tree.

We work with two such sets: (1) the ACM-CCS ontology [1] and a collection of ACM journal abstracts; (2) the VINITI ontology of mathematics and informatics [3] and a collection of teaching syllabuses of mathematics and informatics in The National Research University Higher School of Economics Moscow (NRU HSE, in Russian).

The ACM-CCS ontology is a four-layer rooted tree in which three upper layers are coded as usual, whereas the fourth layer is not coded and can be considered as consisting of descriptions of the third layer topics. The tree has eleven major topics on the first layer, such as B. Hardware, C. Computer Systems Organization, etc. They are subdivided in 81 second-layer topics, which are further divided into third-level topics or so-called leaf topics. Almost all leaf topics are accomplished by topic descriptors that are sets of common phrases or terms corresponding to the topic. There are some cross-references between topics in different partitions. Here is a part of ACM–CCS ontology related to one of the eleven main subjects, D. Software:

D. Software
D.0 GENERAL
D.1 PROGRAMMING TECHNIQUES (E)
D.1.0 General
D.1.1 Applicative (Functional) Programming
D.1.2 Automatic Programming (I.2.2)
D.1.3 Concurrent Programming
Distributed programming
Parallel programming

Three coded layers above are presented by topics D., D.0, D.1.0, etc. The topic D.1.3 is supplied with the topic description involving two terms. The topics D.1 and D.1.2 have references to topics E and I.2.2, respectively.

The VINITI ontology of mathematics and informatics is the most extensive ontology of mathematics domain that is available in Russian [3]. It is an unbalanced rooted tree of mathematics and informatics topics supplied with many cross-references.

Usually, these ontologies are used to annotate documents or publications in large collections such as the ACM portal library or VINITI journals library. Here we concentrate on a different aspect of using ontologies – a procedure for abstracting concepts from text documents.

Accordingly, we consider two collections of texts.

First, we have taken an issue of ACM Journal on Emerging Technologies in Computing Systems (JETC), which is a free access journal [2, 8]. Each publication is represented by three items: 1) an abstract; 2) a set of keywords provided by authors; 3) a set of index terms that are ACM–CCS ontology topics, used on the journal's web site to manually index the article. We use both the abstract and keywords to represent the contents of an article.

Second, we have NRU HSE teaching syllabuses for the courses involving Mathematics and/or Informatics as they are taught in the School of Applied Mathematics and Informatics at NRU HSE. They can be easily downloaded from the NRU HSE web-site (`http://www.hse.ru`).

2.2 Method's composition

The method takes in a text, generates its profile, and then proceeds to further stages described below. The profile is a list of ontology topics generated for the input text. This is based on estimations of the degree of similarity between ontology topics and the text derived by using the so-called annotated suffix tree (AST) techniques [8].

This is the sequence of the method's steps:

1. texts and ontology preprocessing
2. presenting the texts as annotated suffix trees (ASTs)
3. evaluating similarity between the ontology topics and the texts according to texts' AST features
4. constructing the text profiles
   (a) computing the similarity matrix of the ontology topics according to the text corpus
   (b) computing the similarity matrix between the texts
5. finding and analyzing text clusters
6. finding clusters of ontology topics
7. mapping the clusters into higher layers of the ontology structure.

Steps 5 and 6 can be skipped so that the following applies to both individual texts and text corpora.

2.3 Texts preprocessing

Each text is split into sentences and each ontology topic usually consists of one sentence. We represent both the texts and ontology as sets of sentences that are taken as strings. To construct a simple machine representation, two stages need to be completed:

1. extracting meaningful parts from texts;
2. removing from them the unnecessary symbols such as html tags, punctuation marks, etc, and transforming them to the lower case.

While the latter step can be easily done automatically, the first one is conventionally manual. For example, NRU HSE teaching syllabuses include not only subjects but some administration issues as well. Another obstacle to automation of the process is the fact that the teaching syllabuses have no unified template but rather are formatted and stored in different styles and formats.

2.4 Annotated suffix tree representation of a text

An annotated suffix tree (AST) is a data structure used for computing and representing of all the text fragments frequencies. AST for a string is a rooted tree, where each node is labeled with one character and one number. Each path from the root to a leaf reads/encodes one of the string suffixes. Frequency of a node is the frequency of fragment occurrences in the string which is read/encoded by the corresponding path from the root to the node. AST for a collection of strings reads/ encodes every suffix of each string and their occurrence frequency in all the strings.

Examination of a set consisting of an ontology and a text corpus is done according to procedures described in [8]. It involves constructing an AST for each text and evaluating the relevance of each ontology topic to the text. The details can be found in [8]. Therefore, the ontology topics assigned with the highest estimations for a text are selected to form the text's profile either as a fuzzy set of the estimates or a crisp set of the selected topics.

In some cases, a text can be seen as a more complicated entity than just a set of strings. If, as it happens, keywords for a text are provided, one AST may be not enough to represent the keywords-text combination. The ontology, being a hierarchic structure with cross-references, should not be treated as a primitive set of strings too. Here we come to an advanced model of the query set.

2.5 Generating profiles: abstracts, keywords, cross-references

Consider a journal publication that is represented by its abstract together with keywords. On the one hand, keywords may be considered as part of the abstract. Hence after building an AST for the strings of the abstract, keywords can be added one by one to the tree. On the other hand, keywords can be treated as being apart from the abstract as a different constituent of the publication. In this case, one should build two ASTs. First is constructed for strings in the abstract, the second is built for the key-

words. Thus the process of ontology topics evaluation has to be repeated twice, using both of the created ASTs. These estimations are to be summed, possibly with different weights, to form the total ontology topics estimation.

To take into account the third constituent, the cross-references, let us first imagine an ontology as a graph structure. It is composed of two parts: 1) a tree structure that is the hierarchic relation between ontology topics; 2) random references between ontology topics at various layers that can be interpreted as edges of the ontology graph. Hence let us define the distance between two topics as the length of the path between them. If there is no such a path, the distance is set to zero. Now suppose that scores for all the ontology topics are computed. The score of the topic N is amended with the score of the topic N_1 related to N by using the distance between N and N_1. Denoting the distance between N and N_1 by $distance(N, N_1)$, the score of the topic N can be set as

$TotalTopicEval(N) := TopicEval + \alpha^{distance(N, N_1)} TopicEval(N_1)$, where α is a constant such that $0 \le \alpha \le 1$ and $TopicEval$ is the ontology topics scoring function.

2.6 Similarity between ontology topics according to the profiles

The scoring of ontology topics over a text corpus results in a topic-to-text matrix A, where a_{ij} is the total score of topic i in text j. The columns in the matrix A are referred to as profiles of the texts. This matrix can be transformed into a similarity matrix of the ontology topics by computing dot products of rows of matrix A. This allows us to use similarity clustering methods, including spectral clustering as discussed in [11] and [9].

For the sake of simplicity, the procedures of clustering and cluster lifting further described are based on the similarity matrix (and therefore the clusters) involving only leaf topics. Therefore, all texts profiles are to be cleaned of upper layer topics before forming the similarity topic-to-topic matrix.

2.7 Spectral clustering

Additive Fuzzy Spectral Clustering method (FADDIS) [5] combines the Additive Fuzzy Clustering Model and the Spectral Clustering approach. The Spectral Clustering approach relies on the eigenstructure of the similarity matrix. Additive Fuzzy Clustering Method finds one cluster at a time by subtracting the similarities taken into account by preceding clusters from the initial similarity matrix [5]. Therefore FADDIS method sequentially finds the cluster membership vector and its intensity using the maximum eigenvalue and corresponding eigenvector of the residual similarity matrix. A special attention should be given to the data pre-processing stage: FADDIS involves the pseudo-inverse Laplace transformation of the initial similarity

matrix. It was shown by experiments that such a transformation may make more clear the structure of clusters to be extracted.

2.8 Query set lifting over ontology

After a fuzzy, or crisp, topic set is extracted as a cluster or single text profile, this set is considered as an abstraction query to the ontology: a few topics of the higher rank are to be found so that the query set is covered, to an extent, by these high-rank nodes, or "head subjects" representing the query set in as general way as possible. We refer to such a result, and the process, as "lifting" the query set over the hierarchically organized ontology.

The lifting algorithm [5,6] proceeds according to the assumption that if all or almost all elements of a set are covered by a high-layer topic, then the set has been lifted to that very topic.

To conform to this hypothesis we introduce a penalty function to be minimized by lifting the query topic set to the ontology root. It is defined as the weighted sum of different types of nodes that do not fit. The "odd" nodes are determined during the lifting procedure. At the level of leaves we have leaves that either belong to the query set or not. A topic that generalizes most of the topics in a cluster is algorithmically interpreted at the head subject for the query set. Those nodes that are covered by a head subject but do not belong to the set are referred to as gaps. Those nodes that are not covered by a head subject but do belong to the query set are referred to as offshoots. The problem is to minimize the total number of head subjects, gaps and offshoots.

We denote the number of head subjects by H, the number of offshoots, by O, and the number of gaps, by G. Then we recurrently minimize the penalty function $P = h * H + off * O + g * G$ at each step of the lifting process; here h, off and g are the corresponding penalty weights.

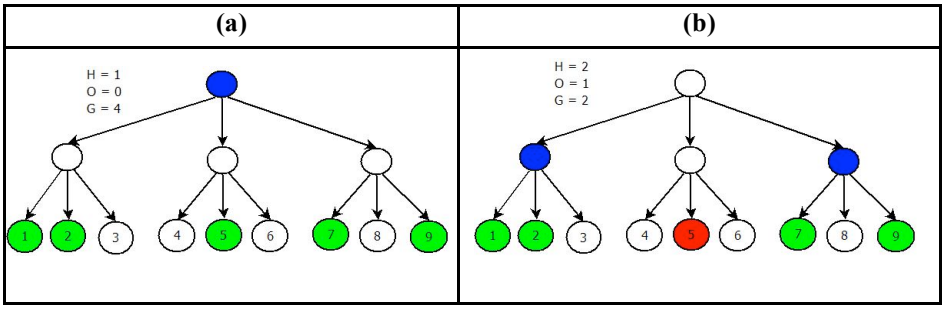

Fig. 1. An illustrative example of mapping a query set to ontology at different penalty weights (see explanation in the text).

Consider the following example of a three-layer ontology and a query set consisting of leaves 1, 2, 5, 7 and 9 (Fig. 1). On Fig. 1(a) there is only one head subject that covers leaves 1, 2, 5, 7 and 9 as well as leaves 3, 4, 6, 8 which are gaps. On Fig. 1(b)

there are 2 head subjects, one offshoot (leaf 5) and 2 gaps: leaves 3 and 8. The opti-

Table 1. Profile A. The spin-wave nanoscale reconfigurable mesh and the labeling problem

AST–profile				ACM index terms	
TE	ID	Ontology topic	#	ID	Ontology topic
133.072	C.1.4	Parallel Architectures	1	C.1.4	Parallel Architectures
128.647	C.1.1	Single Data Stream Architectures			
121.059	C.1.2	Multiple Data Stream Architectures (Multiprocessors)			
107.253	D.2.11	Software Architectures	3	C.1.2	Multiple Data Stream Architectures (Multiprocessors)
105.72	C.1.3	Other Architecture Styles			
...	..	...			

mal lifting is determined now by the minimal value of penalty in both cases that depends on the relation between the gap and offshoot penalties.

3 Examples of experimental studies

3.1 ACM Journal abstracts

As mentioned above, we downloaded and examined a number of journal publications. Each of them is represented by an abstract, several keywords and manually indexed ACM-CCS topics.

This last item gives us a tool to evaluate the machine-constructed profiles, based on AST-evaluation of ACM–CCS ontology topics, so that we are able to find how well the estimated topics match those manually selected. Here is an example of profiles for two journal publications [2, 8], one good and the other poor, in Tables 1 and 2.

Each of the tables consists of two parts. The left part presents our machine generated annotated suffix tree profile (AST–profile), the right one stands for the index terms, which were used by publications' authors to annotate the publication. In the tables: TE is the total score of the ontology topic, ID is the index of the ontology topic. '#" denotes the place of the ontology topic in the descending sorted order of the profile. We expect that index terms are to be high scored according to their abstracts by the AST–procedure and to be placed on the top of AST–profile.

The profile A was constructed for the publication that can be previewed on the following web page `http://portal.acm.org/citation.cfm?id=1265951`. The profile B was generated for the publication on `http://portal.acm.org/citation.cfm?id=1265956`. Only five best scoring ontology topics are present here. However, the AST-profile consists of the all leaves of the ACM CCS ontology.

Table 2. Profile B. A self-organizing defect tolerant SIMD architecture			
AST–profile			ACM index terms

TE	ID	Ontology topic	#	ID	Ontology topic
127.503	C.4	PERFORMANCE OF SYSTEMS	40	C.1.2	Multiple Data Stream Architectures (Multiprocessors)
102.03	B.8.1	Reliability, Testing, and Fault-Tolerance			
79.475	B.4.5	Reliability, Testing, and Fault-Tolerance	108	B.4.3	Interconnections (Subsystems)
76.611	B.8.2	Performance Analysis and Design Aids			
			135	B.6.1	Design Styles
72.382	B.3.4	Reliability, Testing, and Fault-Tolerance			
...	...	...			

There are two ontology topics for the publication A. One can see them among the top five ontology topics, on the first and on the third place correspondingly. The publication B is annotated with three ontology topics that are placed on 40th, 108th and 135th places of the AST-profile. While the profile A should be regarded as more or less satisfactory, the profile B is totally inadequate. This difference comes from the difference in the abstracts. The AST–procedure takes into account only matches between the ontology topic's substrings and the abstract's substrings. If no long substrings match, the whole ontology topic will be scored rather low (a long enough subsequence is of 5-7 symbols). In the case of publication B, there are hardly any matches between the ontology topics and the abstract that are longer than 3-4 symbols. In contrast, the abstract of the publication A includes whole words of some ontology topics, such as 'parallel' and 'architecture'. What is more, it is the involvement of the common word 'architecture' that causes so many unrelated ontology topics to be high scored too.

The AST–method is able to detect all the fuzzy matches between an ontology topic and a text. From this point of view the topics "Single Data Stream Architectures" and "Multiple Data Stream Architectures (Multiprocessors)" are identical if only substring "Data Stream Architecture" occurs in the text under examination. The small difference in their total scores may be caused simply by the presence of shorter substrings like 'gle' or even 'e'. Here is the main shortcoming of the AST–method. It is not possible to catch an ontology topic in a text if it is formulated by using other words than in the ontology.

3.2 Syllabuses for HSE courses in Applied Mathematics and Informatics: Preliminary Results

The study of the VINITI Mathematics ontology and the collection of teaching syllabuses showed several shortcomings, both of the ontology and the syllabuses. After

applying the AST–procedure, we derived the topic-to-topic similarity matrix and extracted crisp clusters by means of the FADDIS method mentioned above. Here are a couple of observations. First: Almost each of the crisp clusters contained some topics from the Topology partition. It means that one or two topologic concepts are studied in almost every mathematical course. This should imply that a course in Topology should be included in the curriculum, which is not the case so far. Second: As the VININTY Mathematics ontology has not been updated since 1980's, it was expected that it may have issues in covering more modern topics in mathematics and informatics. Our analysis suggests several nests that should be possibly added to the ontology. For example, the topic 'Lattices' is a leaf in the current ontology. According to our results, it should be a parental node with three children: 'Modular lattices', 'Distributive lattices' and 'Semimodular lattices'. Third, the VININTY Mathematics ontology has been found as being rather imbalanced in the coverage. The profiles of the 'Differential Equations' and 'Calculus' courses according to the ontology are covering all details. This is no wonder because these two constitute almost half of the ontology. Yet branches for less classical subjects such as 'Game Theory' or 'Optimization' are small and not informative.

One more observation is that the main teaching subjects have no matches among the VININTY Mathematics ontology higher ranks. Such is, for instance, 'Discrete Mathematics'.

4 Conclusion

An idea and some initial stages of a different method for abstracting concepts from text documents are presented. It is based on using ontologies as representation of knowledge. We try to simulate the process of abstraction of texts in three coherent steps. First, we match ontology topics to the texts and construct texts profiles by employing text mining techniques. Next step is performed for a corpus of documents: considering leaf topics as the base for abstraction, we find clusters of topics. Finally, we lift cluster query sets to higher layers of the hierarchy to find and visualize head subjects, along with their gaps and offshoots. The head subjects represent the abstraction sought by the method. The method is being developed as an adaptation of the method from [5]. However, a number of novel procedures have been developed in this work as well. Such are using sentence-by-sentence AST modeling, combining different aspects of the texts (such as key-words and cross-referencing) into the scoring system and the like.

The computation experiments lead us to a number of issues that are to be subjects for further developments. The lack of an adequate taxonomy of Mathematics and Informatics in Russian is among them. The AST technology suffers from the effects of repetitive terms such as "architecture", "method", "system" and the like, that act as noise to falsely raise the similarity scores. On the other hand, the scores are dropping down when the texts use slightly different terms for the taxonomy topics. This latter aspect could be treated by using neighbors of the taxonomy topics found in texts retrieved by search engines when queried with the topics. We expect that the neighbors

would allow not only better scoring in the cases of different terminology, but also would be useful in filling in the gaps generated by lifting the head subjects. The other directions for development would be extension of the concept of ontology from the hierarchy to (semi) lattice structures and finding adequate formalisms for dealing with situations at which there are several ontologies related to the texts.

5 List of references

1. ACM Computing Classification System (1998), http://www.acm.org/about/class/1998 (Cited 9 September 2008)
2. Eshaghian-Wilner M. M., Khitun A., Navab S., Wang K. L.: "The spin-wave nanoscale reconfigurable mesh and the labeling problem". ACM Journal on Emergering Technologies in Computing Systems (JETC) 3(2) (2007)
3. I. Yu. Nikol'skaya, V. M. Yefremenkova: "Mathematics in VINITI RAS: From Abstract Journal to Databases". Scientific and Technical Information Processing 35(3) 128-138 (2008)
4. J. Mercadé, A. Espinosa, J-E. Adsuara, R. Adrados, J. Segura and T. Maes: "Orymold: ontology based gene expression data integration and analysis tool applied to rice", BMC Bioinformatics, 10:158 (2009) doi:10.1186/1471-2105-10-158.
5. Mirkin B., Nascimento S., Fenner T., Pereira L. M.: Fuzzy Thematic Clusters Mapped to Higher Ranks in a Taxonomy. International Journal of Software and Informatics 4(3), 257—275 (2010)
6. Mirkin B., Nascimento S., Pereira L.M.: Cluster-lift method for mapping research activities over a concept tree. Recent Advances in Machine Learning II, 245-247 (2010)
7. Pampapathi R., Mirkin B., Levene M.: A suffix tree approach to anti-spam email filtering. Machine Learning 65(1), 309-338 (2006)
8. Patwardhan J., Dwyer C., Lebeck A. R.: "A self-organizing defect tolerant SIMD architecture". ACM Journal on Emergering Technologies in Computing Systems (JETC) 3(2) (2007)
9. Sato M., Sato Y., Jain L.C.: Fuzzy Clustering Models and Applications. Physics-Verlag (1997). ISBN:3790810266
10. V. Karkaletsis, P. Fragkou, G. Petasis and E. Iosif: Ontology Based Information Extraction from Text, Lecture Notes in Computer Science, V. 6050, Knowledge-Driven Multimedia Information Extraction and Ontology Evolution, 89-109 (2011)
11. Von Luxburg, U.: A tutorial in Spectral Clustering. Statistics and Computing, 17 (4), 395-416 (2006)