AST method for scoring string-to-text similiarity

Ekaterina Chernyak and Boris Mirkin

National Research University – Higher School of Economics,
School of Applied Mathematics and Information Science
echernyak@hse.ru, bmirkin@hse.ru
http://www.ami.hse.ru

Abstract. A suffix-tree based method for measuring similarity of a key phrase to an unstructured text is proposed. The measure involves less computation and it does not depend on the length of the text or the key phrase. This applies to:

1. finding interrelations between key phrases over a set of texts;
2. annotating a research article by topics from a taxonomy of the domain;
3. clustering relevant topics and mapping clusters on a domain taxonomy.

Key words: suffix tree, unstructured text analysis, string similarity measures

1 Introduction

Typically, string-to-text similarity measures are defined using the Vector Space Model (VSM) text representing model. Here, a richer text model, the suffix tree, is used to keep the sequential nature of sentences and make text analysis independent from the natural language and its grammar [9, 12, 3]. Conventional suffix-tree based similarity measures also suffer of drawbacks related to the intensity of computations and their sensitivity to the lengths of both texts and strings. We propose a measure that allows relaxing these limitations. Then we apply our similarity measure to two types of problems:

1. Analysis of interrelations between key phrases over a text collection;
2. Annotation of research articles by a corresponding domain taxonomy topics;
3. Analysis of teaching syllabuses and resident complaints by mapping them to taxonomies, clustering the taxonomy topics and lifting the clusters over the taxonomy tree.

These three problems may look close to traditional information retrieval task as stated in [10] and [11], where the main task is to find all relevant documents for the given query or even to rank them according to their relevance to the query. The key difference is that:

1. We fix the set of queries (e.g. key phrases or taxonomy). What is more, in problem of type 1 we investigate the structure of this set. In [10] no queries are looked at, expect the given one, which is the matter of document ranking problem.
2. We consider important all word occurred in the texts, not only those, which appear in the queries at it is done in [10] and [11].

Hence we treat the set of queries, Q, and the set of txt or documents, D, as a two fixed sets of words, further combined in so-called strings, of equivalent importance for the analysis.

Section 2 describes our method for an annotated suffix tree (AST) construction and the scoring function. Section 3 briefly explains the basic concept of the ST table used throughout in computations. Section 4 presents methods for solving a problem of type (1). Section 5 applies this to a problem of type (2). Two problems of type (3) are described in Section 6. The conclusion completes the text.

2 AST method

The suffix tree is a data structure used for storing of and searching for symbolic strings and their fragments [4]. In a sense, the suffix tree model is an alternative to the VSM, arguably, the most popular model for text representation [12]. When the suffix tree representation is used, the text is considered as a set of strings, where a string may be any semantically significant part of the text, like a word (like it is done in the Bag-of-Words model), a phrase or even a whole sentence. An annotated suffix tree (AST) is a suffix tree whose nodes (not edges!) are annotated by the frequencies of the strings fragments.

We split texts in short fragments, strings, to reduce the computation. Usually we take the strings to be fragments of three sequential words [2]. An annotated suffix tree (AST) for a string is a rooted tree, each node in which is labeled with one of the string symbols so that each path from the root to a leaf encodes one of the string suffixes. AST for a set of strings stores all the fragments of all the strings and their frequencies (see Figure 1). Using AST representation of texts, one is able to find the most frequent fragments in the text and their length.

To build an AST for a text, for its every string, its suffixes are added to the AST, starting from an empty set. To add a suffix to the AST, first check, whether there is already a path in the AST that encodes the whole suffix or its prefix. If such a path (a match) exists, we increase all the frequencies in the match and append new nodes with frequencies 1 to the last node in the match, if it doesnt cover the whole suffix. If there is no match, we create a new chain of nodes in the AST with frequencies 1.

To score similarity of a string to an AST, we match all its fragments with the AST and score each match as the sum of conditional probabilities of matching nodes divided by the length of the match. The conditional probability is the ratio of the node frequency to that of its parent. If no match is found, define the zero score. Then the average of all the scores is computed:

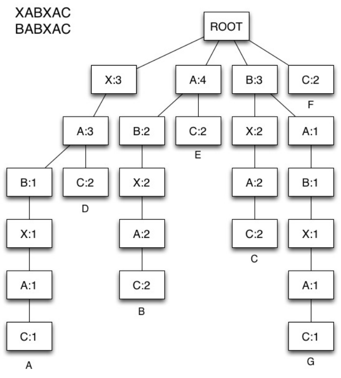

Fig. 1. AST for two strings XABXAX and BABXAC. Note, that the strings differ only in the first position and have common 5 suffixes.

$$scoreMatch(string, AST) = \frac{\sum_{suffix} score(suffix, AST)}{length(string)}$$
$$= \frac{\sum_{suffix} \frac{\sum_{u \in match} f_u / f_{parent of u}}{length(match)}}{length(string)} \quad (1)$$

The VSM-based models are based on finding word to word exact coincidence: a phrase, which is a set of words, may be considered relevant to a text, if a significant part of this set occur in the text. The AST measure avoids searching for exact occurrences. It takes all matching fragments into account, so that a word may match two or more times to the text. Hence, when estimating string-to-text similarity, we deal not with the space of words, but with the space of different matching fragments.

3 Building an ST table

To analyze the relationship between a set of strings and a collection of texts, we build a string-to-text similarity table ST by constructing an AST for each of the texts and estimating similarity of each of the strings to this AST. The rows of table ST correspond to the strings and columns, to the texts. Using an ST table allows us to treat strings as numerical attributes and exploit thus conventional data analysis techniques.

4 Analysis of interrelations on a set of strings over a related text collection

Consider a collection of web publications about current business processes in Russia and a set of key phrases that describe local events like "publishing financial reports" or "replacement of the finance management". To find relations between these events, an ST table key_phrase–to–web_ publication is built first. There can be three types of web publications:

1. those related to only one key phrase;
2. those related to two or more key phrases;
3. those related to no key phrases.

By specifying a threshold, we assign each of the key phrases A with a subset F(A) of related web publications.

Key phrase A implies B, if proportion of F(B) in F(A) is greater than 60%. Thus, one can draw a graph of the implications. For example, in the analysis of 960 web publications on business processes in2009 with about 40 key phrases, we discovered that only 12 of them are of type (2) (see Figure 2).

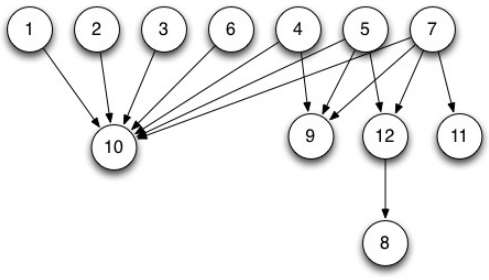

Fig. 2. Graph of the interrelation between the key phrases over a text collection. Codes: 1. Introduction of automated manufacturing; 2. Issuing news bulletins; 3. Change of the size of the shares belonging to the institutional investor; 4. Change of the extent of the ownership concentration; 5. Personnel training; 6. Vertical merger; 7. Brand selling/buying; 8. Entering international markets; 9. Change of the legal organizational form 10. More effective cost control 11. Making the finance reports publicly available 12. Change of the finance director.

5 Annotation of journal articles by topics from a taxonomy of the domain

Another application of the AST method is indexing scientific papers with topics of a taxonomy of the domain. Consider the Association of Computing Machinery (ACM) journals and the ACM developed taxonomy Computing Classification System (ACM-CCS). To represent the contents of their papers, authors of the ACM journals annotate the papers manually with topics from the ACM-CCS taxonomy. To automate this procedure using theAST method one has to:

1. Extract key elements of a paper, such as its heading, abstract, key words if given.

2. Build an AST for the extracted elements.
3. Estimate the similarity of every ACM-CCS topic to the text. The topic similarity values form what we refer to as the AST-profile of the publication.
4. Choose the ACM-CCS topics with the highest scores.

There are two examples of so-called ACM abstract profiles. We put on the left side of each profile the top 5 taxonomy topics, sorted according to taxonomy topic to abstract similarity measure. The manual annotation chosen by authors is on the right side. More precise ID stands for the topic ID in the ACM-CCS taxonomy, S is the similarity value, ACM-CCS topic is the topic itself, Rank is the place the manual annotation have achieved in the AST profile. Ranks can help us to estimate the quality of profiles: the higher ranks manual annotations get, the better the profile is. Hence the Profile A can be thought of as a rather good one and the Profile B as a poor one.

Bojanczyk M. et al. Two variable logic on data trees and XML reasoning Journal of the ACM, 2009, Vol. 56(3), pp. 2-48					
AST found profile			ACM-CCS index terms (manual annotation)		
ID	S	ACM-CCS topic	ID	Rank	ACM-CCS topic
I.6.2	0.4969	Simulation Languages	F.4.3	3	Formal Languages
I.1.3	0.4415	Languages and Systems	H.2.3	4	Languages
F.4.3	0.3796	Formal Languages	H.2.1	13	Logical Design
H.2.3	0.3757	Languages	F.4.1	28	Mathematical Logic
D.4.5	0.2738	Reliability	I.7.2	53	Document Preparation

Table 1. Profile A

Grohe M., et al. Lower bounds for processing data with few random accesses to external memory Journal of the ACM, 2009, Vol. 56(3), pp. 1-58					
AST found profile			ACM-CCS index terms (manual annotation)		
ID	S	ACM-CCS topic	ID	Rank	ACM-CCS topic
J.1	0.5991	Administrative Data Processing	F.1.3	161	Complexity Measures and Classes
I.2.7	0.4757	Natural Language Processing	H.2.4	166	Systems
H.2.5	0.4704	Heterogeneous Databases	F.1.1	220	Models of Computation
H.2.8	0.3419	Database Applications			
C.5.1	0.3146	Large and Medium Computers			

Table 2. Profile B

Unfortunately, our methods outputs AST profiles are not always similar to the authors sets. This may happen because:

A. The method evaluates common words, such as "theorem", "method" or problem" too high. This issue can be addressed by using a stop list of common words.
B. The method works when the formulations of topics use similar letters. It doesn't cope with synonyms. A solution to this issue would be in taking in account a set of synonyms and near synonyms for each of the taxonomy topics.
C. The authors sometimes go too far in their annotations by assuming implications of their methods which are not much considered in the text.

6 Clustering relevant topics and mapping clusters on a domain taxonomy

Suppose we have a collection of texts and a taxonomy, which belong to the same topic domain. We treat taxonomy as a set of topics, each presented by only one string, organized in a rooted tree. The higher the topic is, the more general it is. Hence we can employ the AST method to construct the ST table. In such a table rows, i.e. strings, stand for leaf taxonomy topics, and columns for the texts. Note, that we restrict ourselves only with leaf topics, because it is essential for the further analysis. However all the topics might be used in the way it is described in the previous section. According to the AST table we may find groups of similar topic which match with the text in almost the same fashion by means of some cluster analysis methods. First of all, the clusters may be of their own interest, because they consists of topics, that appear in texts similarly although they dont necessarily belong to one branch of taxonomy. Secondly, using the lifting method, we can map these clusters into the taxonomy. The lifting method outputs a few taxonomy topics of higher levels which cover the cluster of leaf topics in the best possible way. This allows to interpret the whole collection of texts in terms of several taxonomy topics of higher levels, that is a way of data aggregation over hierarchically organized taxonomy. Let us enumerate the main steps of the cluster-lift method:

1. constructing the leaf_taxonomy_topic text ST table
2. finding clusters of leaf taxonomy topics
3. mapping the clusters into higher levels of the taxonomy structure.

To cluster the ST table we may first find leaf_taxonomy_topic leaf_taxonomy_topic similarity matrix by taking dot products of rows of the ST and apply then Additive Fuzzy Spectral Clustering (FADDIS) method [], that uses the Spectral Clustering approach to the Additive Fuzzy Clustering Model to find clusters of leaf taxonomy topics. The other possible way to cluster leaf taxonomy topics is to use iK-Means [5] method to extract clusters one-by one from the ST table. The final step is to lift the clusters in the taxonomy. The lifting algorithm [7] proceeds according to the assumption that if all or almost all elements of a cluster could be covered by a topic on a higher levels than the whole cluster lifts to that very topic. If the assumption does not hold than lifting is impossible. More details on all the methods used are provided in [6]. Below two applications of the cluster-lift method are presented.

6.1 Teaching syllabuses and the taxonomy of mathematics and informatics

The input is twofold. First, we take the most extensive taxonomy of mathematics and informatics domain in Russian [8], that is called the VINITI taxonomy. It is an unbalanced and rather messy rooted tree of mathematics and informatics topics, provided with a lot of cross-references. Second, we downloaded from the web page (www.hse.ru) of our university a collection of teaching syllabuses. These syllabuses correspond to all courses related to Mathematics and/or Informatics as they are taught in the School of Applied Mathematics and Informatics of NRU HSE. Here we used the The study of the VINITI taxonomy and the collection of teaching syllabuses shows several shortcomings, both of the syllabuses and the taxonomy: almost every cluster we get after applying the method to the data, contained topics from Topology branch of the taxonomy. It means that one or another notion form topology is studied during almost all mathematical courses. But there is no such a subject in the curriculum.

As the VINITI taxonomy has not been updated since the early 1980s, it was expected that it may have issues in covering more modern topics in mathematics and informatics. With the help of teaching syllabuses we establish several nests of topics that should be possibly added to the ontology. For example, the topic Lattices is by now a leaf in the taxonomy. According to our results, it should be a parent node with three offsprings: Modular lattices, Distributive lattices and Semimodular lattices.

The ontology has been found of rather imbalanced in the coverage. The Differential Equations and Mathematical Analysis branches are significantly more saturated than the other branches and comprise almost the half of the taxonomy. Yet less classical branches as Game Theory or Programming Theory are way too small and not comprehensive at all. They build up a very small part of the taxonomy, especially in comparison to the giant branches Differential Equations and Mathematical Analysis. We thought that the main teaching syllabuses should be named after the first-level or second-level taxonomy topics. On the contrary, we found that such syllabuses as Discrete Mathematics have not been set among the high-layer taxonomy topics in the VINITI taxonomy.

In this study we used the FADDIS model to cluster the leaf taxonomy clusters. Unfortunately, because of cluster elements being from disconnected branches of taxonomy, the lifting procedure almost failed. We only achieved one layer up lift in a few situations, such as Continuous distributions and Discrete distributions being lifted to their common parent Probability distributions.

6.2 Resident complaints and the taxonomy of community facilities

During the last years special systems for submitting any kind of complaints from residents are introduced in some cities in Russia. One of those systems is exploited in Nizhniy Novgorod for the residents complaining on the problems with community facilities. Our colleagues [???] from the Nizhniy Novgorod NRU HSE campus developed a taxonomy, that describes almost all constituents of community facilities in order to automate the analysis of the flow of complaints.

First we noticed that there many significant concepts missing from this taxonomy. For example, there were topics Elementary school and Middle school, while the Kindergarten topic was missing. We manually extracted some frequent nouns or collocations from the given collection of complaints and updated the taxonomy with these extracted topics.

Second we built the leaf_taxonomy_topic resident complaint ST table by means of the AST method and then applied the cluster-lift method. We used iK-means method [5] to extract clusters of taxonomy topics from the ST table. We cleaned the clusters from extra large or small clusters. The rest of clusters were parsimoniously lifted. For example, cluster 1 in the Figure 3 below consisted of 4 taxonomy topics: 1.2.1. Hot water problems, 1.2.2 Cold water problems, 1.2.3 Water meter problems, 1.11.2 Public water pump. On the first iteration of the lifting method the cluster was mapped to 1.2 Water Supply and to 1.11 Urban landscaping and public amenities. These two were then mapped on the second iteration to the first level topic 1. Housing services.

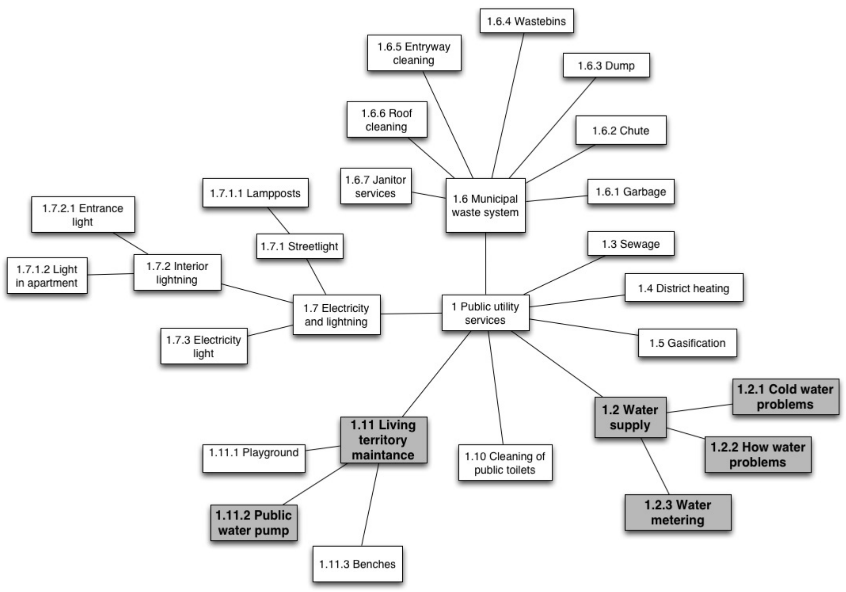

Fig. 3. Lifting cluster to higher levels

7 Conclusion

The AST method used for estimating string-to-text similarity has several advantages over the VSM-based methods. It doesn't require any complicated preprocessing procedure like stemming or POS-tagging and is then independent from grammar. By using the string concept, we can explore long enough text fragments, so that short semantical links aren't lost. The experimental computations lead us to the number of issues that are to be subject of the further developments:

1. the AST method deals only with matching strings. To make it more efficient we should take synonyms or near synonyms into account.
2. we have done so far some manual attempts to improve taxonomies so that they would become more balanced and up to date. It is, perhaps, possible to automate the process of improving or refining taxonomies using either given collections of texts or some external sources.
3. by now we have assumed that the set of strings, that is further used for matching with the texts and building the ST table, is given by some experts or is taken from some official source. We may extract these strings from the texts by using some well-known methods of long key phrases extraction.

References

1. ACM Computing Classification System, http://www.acm.org/about/class/ (1998)
2. Chernyak E., Chugunova O., Askarova J., Nascimento S., Mirkin B.: Abstracting Concepts from Text Documents by Using an Ontology. 1st International Workshop on Concept Discovery in Unstructured Data, pp.20 – 30. University Higher School of Economics, Moscow (2011)
3. Grossi, R., Vitter, J.: Compressed Suffix Arrays and Suffix Trees with Applications to Text Indexing and String Matching. SIAM Journal on Computing, Vol. 35, No. 2, pp. 378-407 (2005)
4. Gusfield D.: Algorithms on Strings, Trees, and Sequences, Cambridge University Press (1997)
5. Mirkin B.: Clustering for Data Mining: A Data Recovery Approach. Boca Raton Fl., Chapman and Hall/CRC (2005)
6. Mirkin B., Fenner T., Nascimento S., Pereira L.M.: A Hybrid Cluster-Lift Method for the Analysis of Research Activities. Lecture Notes in Computer Science, Vol. 6076, No. 1, pp. 152-161 (2010)
7. 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. Vol 4, No. 3, pp. 257-275 (2010)
8. Nikolskaya I. Yu, Yefremenkova V. M.: Mathematics in VINITI RAS: From Abstract Journal to Databases. Scientific and Technical Information Processing. Vol. 35, No. 3, pp. 128-138 (2008) (in Russian)
9. Pampapathi R., Mirkin B., Levene M.: A suffix tree approach to anti-spam email filtering. Machine Learning, Vol. 65, No. 1, pp.309–338 (2006)
10. Robertson S., Zaragoza H.: The Probabilistic Relevance Framework: BM25 and Beyond. Journal Foundations and Trends in Information Retrieval, Vol. 3, No. 4, pp. 333-369 (2009)

11. Salton G. , Wong A., Yang C. S.: A vector space model for automatic indexing. Communications of the ACM. Vol .18, No. 11, pp .613-620 (1975)
12. Zamir O, Etzioni. O.: Web document clustering: A feasibility demonstration. In: Proceedings of SIGIR98, pp.46 – 54, University of Washington, Seattle (1998)