Contemporary Engineering Sciences, Vol. 4, 2011, no. 7, 311 - 320

Level-Based Transformation of Topic-Maps into Similarities of Topics (LTTMST)

Mofleh Al-diabat

Al Albayt University
Department of Computer Science
moflehd@aabu.edu.jo

Salah Al-Gialain

The World Islamic Science & Education University
Department of Computer Information System
salah.gyalin@wise.edu.jo

Rayed Alsurdeh

Al Albayt University
Department of Computer Information System
Raid-alsrdi@aabu.edu.jo

Abstract

There are several information filtering and retrieval models like vector space model and fuzzy set model that able to find similarity between queries and terms within a collection but those models ignore the similarity between terms it self, this Lead to appear new algorithms that take terms dependency in account, one of those algorithms is "Transformation of topic-maps into similarities of topics – eTVSM" [4] which suggested that, the values of the vector entries depend on position of the topics within the ontology and it uses a binary value for vector entries (0 or 1).

However, the eTVSM algorithm. Have many shortcomings, such as ignoring a lot of relations between topics, during computing topic-vector ,and it assign binary values for topic-vectors , and to compute topic-vector it require from you to find all direct sub-topics-vectors of the node and this mean more calculations. This paper attempt to improve the eTVSM algorithm by gives more precise values for the vector entries and decrease calculations.

Keywords: Vector Space, VSM, eTVSM**,** GVSM, LTTMST, document similarity, web semantic

Introduction

A retrieval model specifies the representations used for documents and information needs (Query), and how they are compared [Turtle,

There are several models assume terms independent (orthogonal), like vector space model (VSM)[5] , Binary Independence Retrieval[6] and this may increase the performance and allows no regarding the computation of term angle but this assumption doesn't reflect the real situation, Because any natural language contain a lot of synonyms and words related strongly to each other .
The Semantic Web relies heavily on formal ontologies to structure data for comprehensive and transportable machine understanding. Thus, the production of ontologies factors largely in the Semantic Web's success[1] .

This led to appear new models that consider that terms not always independent and it may related to each other. Those models for examples generalize vector space model (GVSM) [7] ,and topic based vector space model(TVSM) [4] and some models use statistical co-occurrence for a pairs of term within a set of document and other models use numerical value which assigned to each pair of terms but How those values can be derived ? .

To solve this problem several algorithms and models are proposed that take ontology of terms in account like" ontology-based retrieval information system " [3] and eTVSM.

eTVSM assume that the entries for each vector depend on the position of topic within the ontology and it give 1's for the all super topics for the current topic and ignore the other topic (give it 0's) ,But all topics in the real world relate to each other by different degrees and each of them should assign different weights .

This paper will propose new algorithm to calculate similarity for any acyclic tree and it will
Give more precise values for the vectors entries and it will decrease from the previous problem.

LTTMST Algorithm :

Figure 1 explain topics map which contains set of topics all of them talk about the same subject and those topics are distributed on levels and this distribution depend on relations between those topics so topics in level two more related to topics in level one than level tree .

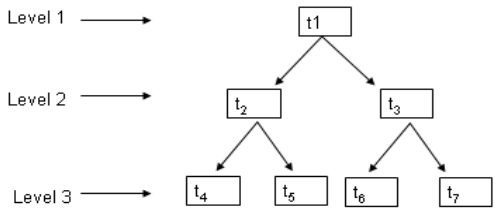

Figure 1: topic map tree

Definitions:

1. Let N is the number of Topics in the topic –map
2. we define $T = \{t_1, t_2, t_3, \ldots t_N\}$, Since T represent set of all possible topics in the topic-map .

So, for the previous Topic-map $T = \{t_1, t2, t3, t_4, t_5, t_6, t7\}$

3. For each $t_i \in T$:

- We define set $T_r(t_i)$,which contains all super and subordinator topics for topic t_i and topic t_i is included .

for example : $T_r(t_2) = \{t_1, t_2, t_4, t_5\}$

$T_r(t_1) = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7\}$

The super and sub topic relation allows us to construct more complex relations, and it used To obtain all super and sub topics of the target topic.

- We define set $T_n(t_i) = T \setminus T_r(t_i)$, for example : $T_n(t_2) = \{t_3, t_6, t_7\}$

$T_n(t_1) = \{\}$

- *We find* $D_i = \{ L(i,1) , L(i,2) , L(i,3) , \ldots, L(i,n)\}$

Since, L(i,d): distance between topic (*i*) and topic (*d*) (distance in levels)

- *Distance between topics is positive number and it measure the degree of similarity between topics so, the distance between topic and it self is Zero and when it increase the similarity between these topic will decrease .*

- *For each topic Ti T a topic vector Ti={t1,t2,t3, t4, t5, t6,t7} the values within vector depend on the position of the Ti in the topic-map*

$$T_{i,d} = \begin{cases} 0 & \text{if } T_d \in T_n(t_i) \\ (N - L(i,d)) / N & \text{For other topic,} \end{cases}$$

From the previous figure 1 we get the following values for each topic vector

To find entries for (T1) ,you must compute :

- *Tr (t1) ,which contains all super and subordinator topics for topic T1*

So Tr (t1) will include {*t1,t2,t3,* t4,t5,t6, t7}

- *Tn (t1) ,which contains topics that not super or subordinator topics for topic T1*
  So Tn (t1) will include nothing (empty set)

Then we find the distance between T1 and the elements in Tr (t1) set :

Tr (t1) ={0 , 1 , 1 , 2 , 2, 2 , 2} ,Since
0 represents distance between t1 and t1.
1 represents distance between t1 and t2 and so.

Then you can find entries for vector by the previous relation (Ti,d)
Since levels number is (3)

T1 = |((3-0)/3, (3-1)/3,(3-1)/3, (3-2)/3, (3-2)/3, (3-2)/3, (3-2)/3)|

T1 = |(1, 2/3,2/3, 1/3, 1/3, 1/3, 1/3)|

Then we divide each element in T1 by squawroot for the sum of squares for all elements in T1 $\sqrt{\sum}$ (12 + 2/3 2+…+1/3 2) =1.5275

=(0.6547,0.4364,0.4364,0.2182,0.2182,0.2182,0.2182) .
Then we find vector length for T1 by squawroot for the sum of squares for all elements
$\sum$(0.65472 + 0.43642+…+,0.2182 2) =0.9999
| T1 |=1

Figure 2: similarity matrix of the sample map

T2 = |(4/3, 1,0, 2/3, 2/3, 0, 0)|
T2 = (0.6963, 0.5222, 0, 0.3482, 0.3482, 0, 0).
| T2 |=1

T3 = |(4/3, 0,1, 0, , 0, 2/3, 2/3)|
T3 = (0.6963, 0, 0.5222,0,0, 0.3482, 0.3482).
| T3 |=1

T4 = |(5/3, 4/3,0, 1, , 0, 0, 0)|

T4 =(0.7071,0.5657,0,0.4243,0,0,0) .
| T4 |=1

T5 = |(5/3, 4/3,0, 0, , 1, 0, 0)|
T5 =(0.7071,0.5657,0,0,0.4243,0,0) .
| T5 |=1

T6 = |(5/3,0, 4/3,0, 0, , 0, 1, 0)|
T6 =(0.7071,0,0.5657,0,0,0.4243,0) .
| T6 |=1

T7 = |(5/3,0, 4/3,0, 0, , 0, 0, 1)|
T7 =(0.7071,0,0.5657,0,0,0 , 0.4243) .
| T7 |=1

Finally we find similarity between each pair of topics by product of corresponding topic vectors .Because it equal to the cosine of the angle β between topic vectors.

$$\begin{aligned} sim(\tau_i, \tau_j) &= \vec{\tau}_i \vec{\tau}_j \\ &= \sum_{k=1}^{t} \tau_{i,k} \tau_{j,k} \\ &= \cos \beta_{ij} \end{aligned} \qquad [5]$$

Figure 2: represent the similarity among topics in the previous tree since similarity between any two topics is the intersection between column (first topic) and row (second topic) for example similarity between t1 and t5 is (0.5832)
And it equal the similarity between t5 and t1

	t1	t 2	t 3	t 4	t 5	t 6	t 7
t1	1	0.7619	0.7619	0.5832	0.5832	0.5832	0.5832
t2	0.7619	1	0.1905	0.8165	0.8165	0.1166	0.1166
t 3	0.7619	0.1905	1	0.1166	0.1166	0.8165	0.8165
t 4	0.5832	0.8165	0.1166	1	0.3571	0.0714	0.0714
t 5	0.5832	0.8165	0.1166	0.3571	1	0.0714	0.0714
t 6	0.5832	0.1166	0.8165	0.0714	0.0714	1	0.3571
t 7	0.5832	0.1166	0.8165	0.0714	0.0714	0.3571	1

- Similarity between any topic and the addition of all its direct subtopics is one (similarity =1) .

Example: similarity between topic 2 and [topic4+topic5] ($sim(t_4+t_5,t_2)$)

t_4+t_5 =| 10/ 3,8/3,0,1,1,0,0|
=(0.7413,0.593,0,0.2224,0.2224,0,0)

$| t_4+t_5 |=1$

$sim (t_4+t_5,t_2) =1$

- similarity between super topic and any direct sub topic is equivalent for all sub topics :

Example: similarity between topic 1 and topic2 is equivalent to : similarity between topic 1 and topic 3

Comparison with other topic-based similarity algorithm:

The comparison between different Information Retrieval models is difficult issue because it depends on performance end evaluation of these models, Evaluation of any Information Retrieval model is a highly heuristic task, primary because of human activities involved in the process.

In eTVSM algorithm, to find a vector for any topic you must find all vectors for the sub topics and after that you can calculate the vector for this topic but in the LTTMST algorithm you can directly find the topic-vector for any topic regardless its location in the tree.

In LTTMST it easy to expand topics tree and it will require from you less calculation to rebuilding topics-vectors.

In LTTMST and eTVSM algorithms similarity decrease sequentially from top to bottom where distance between these topics is increase and each of them give more weight for sub tree to which topic is belong.

In LTTMST and eTVSM algorithms similarity between any topic and its direct subtopics is equivalent for all sub topics.
In the eTVSM algorithm the vectors of the topic-leaves are building firstly and other vectors depend on these vectors So, to find vector for topic that situate at level one you must calculate vectors for all its sub vectors and that require cumulative calculations where as in LTTMST algorithm you can find vector entries for topic one for example without compute its sub vectors. Figure 3 demonstrates the differences in both algorithms performance. It can be concluded from the figure the decrease in execution time for finding the similarity matrix when applying LTTMST for topics over 500.

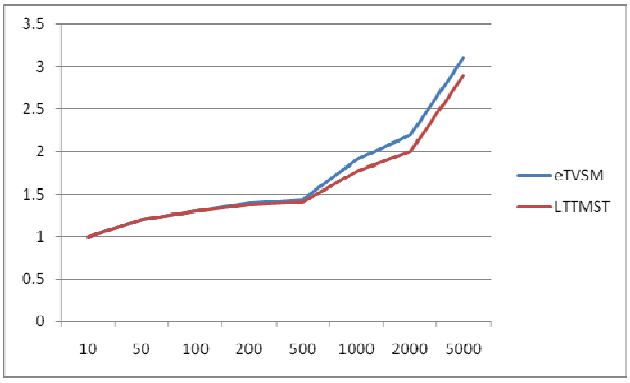

Figure 3 : performance comparison between eTVSM and LTTMST

Conclusion:

This paper proposes a new algorithm to transform topic-maps into Similarities of Topics this approach has the following advantages:

- You can compute the vector entries for any topic without need to find Its subtopics vector entries previously.
- It works with different types of trees.
- Doesn't need sophisticated computation and it is easy to implement .

It is preferable to implement with collections that contains synonyms and many words related to particular subject .

References

[1] A., Maedche, Ontology Teaching for the Semantic, Ontology Learning for the Semantic Web. IEEE Intelligent Systems, 16(2001): pp.72-79.

[2] Becker, J., and Kuropka, D. Topic-based vector space model. In Proceedings of the 6th International
Conference on Business Information Systems (2003),
pp. 7–12.

[3] D., Vallet, M., Fernández, P., Castells; an Ontology-Based Information Retrieval Model. (2005).

[4] D., Kuropka, A proposal for transformation of topic maps into similarities of topics, Web: http://kuropka.net/files/Topic-ims.pdf (2006)

[5] G. Salton, M. Lesk: "computer evaluation of indexing and text processing Journal of the Association for Computing Machinery, 15:1:8--36, 1968

[6] S. E, Robertson, K. S, Jones, Relevance weighting of search terms. Journal of the American Society for Information Sciences 27, 3 (1976), 129–146.

[7] S. Wong, W. Ziarko, V. Raghavan, P . Wong, On modeling of information retrieval concepts in vector spaces". ACM Transactions on Database Systems Volume 12, issue 2, (Jun 1987), 299–321.

Received: July, 2011