A Linked Data Generation Method for Academic Conference Websites

Peng Wang[1], Mingqi Zhou[1], Xiang Zhang[1], and Fengbo Zhou[2]

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China
[2] Focus Technology Co., Ltd, Nanjing, China
{pwang,x.zhang}seu.edu.cn, zhoufengbo@made-in-china.com

Abstract. This paper proposes an automatic method for extracting information from academic conference Web pages, and organizes these information as ontologies, then matches these ontologies to the academic linked data. The main contributions include: (1) A page segmentation algorithm is proposed to divide conference Web pages into text blocks. (2) According to vision, key words and other text features, all text blocks are classified as 10 categories using bayes network model. The context information of text blocks are introduced to repair the initial classified results, which are improved to 96% precision and 98% recall. (3) An ontology is generated for each conference website, then all ontologies are matched as an academic linked data.

Keywords: Web information extraction, Ontology, Linked data.

1 Introduction

With the popularity of semantic Web technologies and the emergence of intelligent applications such as semantic search, more and more plain or semi-structured Web data is need to be reorganized as semantic data, which is the foundation of many intelligent applications. Linked data is such large scale semantic data. Recently, more and more linked data such as DBpedia [1], Freebase and Google knowledge graph is used in many fields including knowledge engineering, machine translation, social computing and information retrieval. Academic linked data is important for academic social network analysis and mining. However, current academic linked data is based on database like DBLP and mainly describe paper publication information. Therefore, academic activity knowledge are not included by current academic linked data. Academic conferences websites not only contain paper information, but also contain many academic activity information including research topic, conference time, location, participants, awards, and so on. Obtaining such information is not only useful for predicting research trends and analyzing academic social network, but also is the important supplement to current academic linked data. Since academic conferences Web pages are usually semi-structured and content are diversity, there is no effective way to automatically find, extract and organize these academic information to linked data.

To generate linked data from semi-structured pages, it usually needs three phases: Web information extraction, ontology generation and linked data construction. Web

M. Zhou et al. (Eds.): NLPCC 2012, CCIS 333, pp. 296–307, 2012.

information extraction is a classical problem [1-3], which aims at identifying interested information from unstructured or semi-structured data in Web pages, and translates it to into a semantic clearer structure such as XML and domain ontology. Although academic conference Web pages usually have strict layout and content description, there is no a fixed template for all conference Web pages to follow. An ontology formally represents knowledge as a set of concepts within a domain, and the relationships among those concepts [4]. However, generating ontology is a challenge [5-6]. Linked data is a way to employ ontology languages such as RDF to describe and share knowledge on the Web [7-9]. More and more linked data is generated in recent years, and it contains the knowledge about geographic information, life science, Wikipedia data, government information, images, and so on. Linked data is also the foundation of many intelligent applications such as semantic search and social network.

In summary, this paper has following contributors: (1) We propose a new page segmentation algorithm, which use DOM tree to compensate the information loss of classical vision-based segmentation algorithm VIPS; (2) We transform the conference Web information extraction problem into a classification problem, and classify text blocks as pre-defined categories according to vision, key words, text and content information; The initial classification results are improved by post-processing. Finally, academic information is extracted from the classified text blocks. (3) A global ontology is used to describe the background domain knowledge, and then the extracted academic information of each website is organized as local ontologies. Finally, academic linked data is generated by matching local ontologies. Our experimental results on the real world datasets show that the proposed method is highly effective and efficient for extracting academic information from conference Web pages and generating high quality academic linked data.

2 Page Segmentation

To extract the academic information, we first segment Web pages into blocks by VIPS[10], which is a popular vision-based page segmentation algorithm. VIPS can use Web page structures and some vision features, such as background color, text font, text size and distance between text blocks, to segment a Web page. These text blocks can be constructed as a vision tree, which assures that all leaf nodes only contain text information. VIPS can obtain good segmentation results for most Web pages, but we find it will lose important information when deal with some Web pages. It is caused by the reason that VIPS algorithm is only based on vision features of page elements, so it would ignore blocks whose display is inconsistent. Therefore, we introduce DOM-based analysis to improve VIPS segmentation results, especially finding missed text blocks.

First, we need to obtain the basic vision semantic blocks by analyzing DOM tree of Web pages. A vision semantic block is a text block with independent meaning. A lot of blank nodes are removed from HTML tags. Then we traverse DOM tree to extract vision semantic blocks. A vision semantic block is between two newline tags such as <br/> and only contains style tags and texts.

Algorithm 1 shows the detail of generating the VIPS complete tree. Let SB be vision semantic block, LN be layout node of vision tree generated by VIPS, and DN be data node. This algorithm includes three steps: (1) It finds a SB by traverse LN to search matched layout nodes; (2) If it finds a matched layout node, then this node is also a vision semantic block; (3) If it does not find a matched layout node, then add this SB into the vision tree. This algorithm not only assures that there is no information loss, but also preserves the structure of vision tree. Fig. 3 shows the part of VIPS complete tree for bottom part of AAAI2010 main page. As vision tree shows, VIPS only extracts the text with bold font. It is not the result we expect. After the processing of Algorithm 1, new semantic blocks are added to the vision tree, which is the complete tree with correct text blocks.

Algorithm 1. Generating VIPS complete tree algorithm

```
Input: <LayoutNode,DataNode> LNDN[], VIPS result PN
Output: a VIPS complete tree T

begin
   for (PN.children[i] in PN.children[])
     if (LNDN[] has key PN.children[i])
       add LNDN[PN.children[i]] to T
else
       add new DataNode(PN.children[i]) to T
        LN_saved[].add(PN.children[i])
   end
   while (LN_saved[] is not empty) {
     currentLN = LN_saved[0]
     LN_saved[].remove(0)
     currentDN = LNDN[currentLN]
        for (currentLN.children[i] in currentLN.children[])
       if (LNDN[] has key currentLN.children[i])
            add LNDN[currentLN.children[i]] to T
       else
        add new DataNode(currentLN.children[i]) to T
       LN_saved[].push(currentLN.children[i])
    end
  end
end
```

Since some blocks such as navigation, copyright and advertisement do not contains the academic information. We regard these blocks as noise, which should be removed from VIPS complete tree. The noise removing process uses some vision features[11]. (1) **Position features** include block position in horizontal and vertical on page and ratio of block area to page area. (2) **Layout features** include alignment of blocks, whether neighbor blocks are overlapped or adjacent. (3) **Appearance features** include size font, image size, and font of link. (4) **Content features** include common words of blocks and special order of some words. According to these vision features, we can remove noise nodes from VIPS complete tree.

3 Text Blocks Classification

The academic information of a specific conference is distributed within a set of pages. For instance, the *Overview* page of the conference Web site usually contains the conference name, time, and location information, and a *Call for Papers* page usually contains topics of interest and submission information. In general, we are primarily concerned with five types of academic information on a conference website: (1) *Information about conference date*: conference begin and end date, submission deadline, notification date of accepted papers, and so on. (2) *Information about conference research topics*: call for papers(or workshops/research papers/ industrial papers), topics of interests, sessions, tracks and so on. (3) *Information about related people and institute*: organizers, program committee, authors, companies, universities, countries and so on. (4) *Information about location*: conference location, hotel, city and country. (5) *Information about papers*: title, authors of papers.

We divide all text blocks into 10 categories as Table 1 shows: (1)**DI**: It describes date information; (2)**PI**: It describes location information; (3)**AR**: It refers to top level information such as research area; (4)**TO**: It refers to research topics, and a AR block may have some corresponding TO blocks; (5)**PO**: It describes the role of people in conference such as Speaker and Chair. (6)**PE**: It refers to information of a person; (7)**PA**: It is the information about papers; (8)**CO**: It refers the blocks which is combined by the above 7 categories blocks; (9)**R**: It refers to the interested blocks but not belong to any categories; (10)**N**: It refers to the blocks not only belong to any categories but also not related to academic information. Fig. 4 shows each category and corresponding examples.

Table 1. Categories of text blocks

	Category	Description
Date	DI(dateItem)	Dates about conference events
Location	PI(placeItem)	Location function and address
Research	AR(area)	Research area in high level
	TO(topic)	Research topic in each area
People	PO(position)	Positions of participants such as *Speaker*, *Chair* and *Co-Chair*
	PE(peopleItem)	People names and institutions
Paper	PA(paper)	Paper type, title and authors
Other	CO(collection)	Set of some above blocks, such as DI+PI means that it contains date and location. For example, a PI+DI in AAAI-11page: “AAAI is pleased to announce that the Twenty-Fifth Conference on Artificial Intelligence (AAAI-11) will be held in San Francisco, California at the Hyatt Regency San Francisco, from August 7–11, 2011.”
	R(related)	Not belong to above 8 categories, but contains useful information, such as workshop information.
	N(notRelated)	Not belong to above 8 categories and does not contain useful information

According to these categories, we can select some features to measure a given text blocks. We use vectors as Table 2 shows to describe each blocks. For a text block, we construct its features according to vision, key words and text content information. For example, given a text block: "Paper Submission Due: **Friday, May 6, 2011 (23:59 UTC - 11)**" and its HTML source code: *<li> Paper Submission Due: <b>Friday, May 6, 2011 (23:59 UTC - 11)</b></li>*, its feature can be constructed as:

(1) **Vision features**: isTitle=false, isHeader =false, startWithLi=true, left=(280-0)/950=0.3 (page width:950, left margin: 0, text left margin:280), with=640/950=0.7 (text width: 640);
(2) **Key word features**: nearestTitle=DI (its nearest and isTitle=true blocks is about date information), dateNum=2 (it contains 2 date words: Submission and Due), paperTypeNum=1 (it contains 1 key word about paper: Paper);
(3) **Text content features**: fontSize=0, fontWeight=0, textLength=58, textLink=0, wordNum=11, nameNum=5, wordToName=11/5=2.2.

There are many famous existing classification algorithms such as C4.5[8], K-Nearest Neighbors (kNN)[9] and Bayes Network[10]. C4.5 and Bayesian Network are the most widely used classifier models. After comparing the two classifier models, we choose bayesian network model to solve the text blocks classifier problem.

Table 2. Feature vectors of text blocks

Vector	Description	Value
isHeader	Whether the biggest font size	bool
isTitle	Whether the title font size	bool
nearestTitle	Type of the nearest title block	int
textLength	Length of text block	int
fontSizeToAverage	Average font size	int
fontWeightToAverage	Average font weight	int
startWithLi	Whether start with <li>	bool
dateTypeNum	Number of key words about date type, such as *deadline*	int
dateNum	Number of key words about date, such as *January*	int
placeTypeNum	Number of key words about location type, such as *Place*	int
placeNum	Number of key words about location, such as *Italy*	int
areaNum	Number of key words about research area	int
nameNum	Number of names	int
institutionNum	Number of institutions	int
positionNum	Number of positions	int
authorNum	Number of authors	int
abstractTypeNum	Number of key words about abstract	int
paperTypeNum	Number of key words about paper type	int
wordNum	Number of words of text blocks	int
wordToName	Ratio of number of words to number of names	double
linkTotext	Ratio of length of link to length of blocks	double
left	Ratio of left margin to page width	double
width	Ratio of block width to page width	double

The classified results can be improved by post-processing, which includes repairing wrong classified results and adding missed classified results. The text blocks with wrong classification can be determined by two sides:

(1) A block has special features but cannot be classified correctly. For example, given text block "*Camera Ready Papers Due: Thursday, August 11, 2011*", which have typical DI features, namely, *dateNum*=1, *dateTypeNum*=1, *isDateArea*=true

and *isPreviousDate* = true. However, this block is classified as Related. For this situation, it can be repaired by identify some typical combination of features. This method is suitable for text blocks with clear features, such as DI, TO, PO and PE.

(2) A text block is classified as a category but it does not contain corresponding features. For example, text blocks about submission instruction would be classified as CO. Although it contains a lot of words, it usually does not contain any people name, date or location. Therefore, we can check its feature values *wordToName*, *dateNum*, *dateTypeNum* and *placeNum* to determine whether it is a CO category. According to this way, we can repair some wrong classifications.

Some text blocks are classified as R category, and they have useful information, but their categories are not clear. These blocks should be checked further. Therefore, we use context feature to determine the blocks with R category. The context feature consists of 11 boolean values: *isDateArea*, *isPlaceArea*, *isTopicArea*, *isPeopleArea*, *isPaperArea*, *isPreviousDate*, *isPreviousPlace*, *isPreviousTopic*, *isPreviousPeople*, *isPreviousPaper* and *isVisuallySame*. We propose some rules to add text blocks without clear classification.

Rule 1: DateItem complete rule: A DI text block usually appears as three situations: (1) It appears as page title with bold and big font size; (2) It appears with other DI text blocks; (3) It appears separately. Since a DI block is very dependent to key words, for any situation, key words and other features should be considered.

For situation (1), we can use a simple rule to detect: *dateNum*>0 && *wordNum*<=5 && *isTitle*=ture.

For situation (2), the corresponding rule is : (*isPreviousDate* ||*isDateArea*) && *isVisuallySame* && *isDateArea* && *index-areaIndex*==1 && *isPreviousDate* && *isNextDate* && *isDateArea* && *isPreviousDate* && (*dateNum*>0 || *dateTypeNum*>0).

For situation (3), the rule is *startWithLi* && (*dateNum*>0 || *dateTypeNum*>0) && *wordNum*<= *DATE_ITEM_WORD_NUM*.

Rule 2: PlaceItem complete rule: PI classification usually has high precision. Some key words and text content features can determine whether a R block is a missed PI block. The rule is: *placeNum*>1 && *wordNum*<PLACE_WORD_NUM.

Rule 3: Area complete rule: A AR text block usually has big font size. Therefore, its complete rule is *isHeader*=true.

Rule 4: Topic complete rule: A TO text block has typical context features. It begins with a <li> tag, and if its neighbor block is TO, it is possible TO block too. The complete rule is: (*isTopicArea* || *isPreviousTopic*) && (*isVisuallySame*) && (*isTopicArea*) && (*index-areaIndex*=1) && *startWithLi*.

Rule 5: Position complete rule: A PO text block usually has big font size and few words, and it is also has key words about position. The complete rule is *isTitle* && *wordNum* < POSITION_WORD_NUM && *positionNum* > 0.

Rule 6: PeopleItem complete rule: A PE block should consider its context. Usually, it follows PO blocks. Some PE blocks begin with <li> tag, and some PE blocks contain key words about universities or companies. Most PE blocks have capital letters and short content. The complete rule is: (*isPeopleArea*||*isPreviousPeople*) && *isVisuallySame* && *isPeopleArea* && *index-areaIndex*=1 && *startWithLi* && *isPreviousPeople* && *institutionNum*>0 && *wordToName*< PEOPLE_WORD_TO_NAME && *wordNum* < PEOPLE_WORD_NUM.

Rule 7: Paper complete rule: A PA block usually begins with <li>, and its list is similar to TO blocks. The corresponding complete rule is: (*isPaperArea*||*isPreviousPaper*) && *isVisuallySame* && *nameNum*> TITLE_UPPER_WORD_NUM && *startWithLi* &&

nameNum>TITLE_UPPER_WORD_NUM && *isPaperArea* && *index-areaIndex*=1 && *nameNum*> TITLE_UPPER_WORD_NUM.

After the post-processing, initial classification results will be improved greatly. For the reason that classified text blocks describe a special information, these blocks are the academic information we want to extract. Namely, this paper solves an information extraction problem by transforming it into a text block classification problem.

4 Ontology Generation

To obtain the academic linked data, we need to organize the extracted academic information. Therefore, we first manually build a global ontology as background knowledge of academic domain, then automatically construct local ontologies for each conference website.

After investigating a lot of conference websites and some ontologies related to academic domain, we manually construct a global ontology for describing the knowledge of academic conference. The global ontology contains 97 concepts and 27 properties. Fig. 1 shows part of global ontology. Besides the hierarchy, concepts can be related by properties. For example, property *hasAuthor* can link two concepts *Paper* and *Author*, which are domain and range of *hasAuthor*. Global ontology has not instances, and it is stored as an ontology language RDF file.

For a local ontology, its concepts and properties are contained in global ontology. We don't consider the new knowledge which is not described in global ontology. Therefore, for a extracted academic information, it is either a concept or an instance of local ontology. However, not all extracted information can be translated to concepts or instances directly. Therefore, the concepts and instances should be determined by the context of the academic information. For example, a paper information

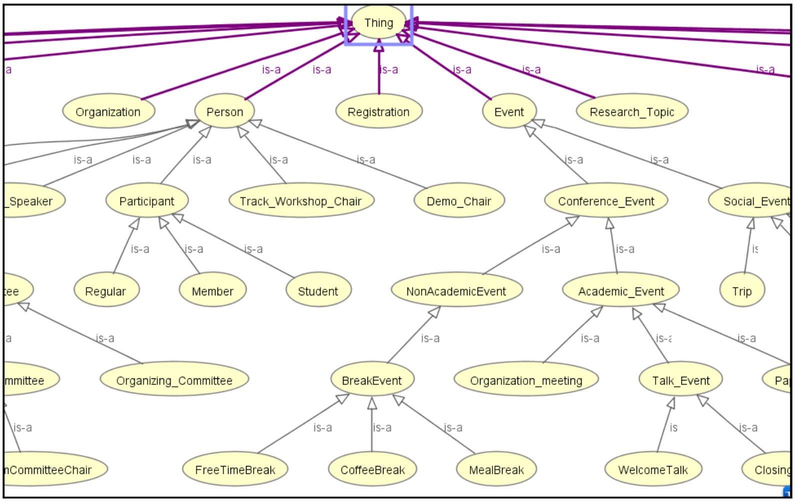

Fig. 1. Part of global ontology

usually appears in PA text blocks and contains title and authors, so it is an instance of concept *Paper*, all authors are instances of concept *Author*, titles will be the property values, and authors will be the values of *hasAuthor*. Through the above process, we can generate a local ontology for each conference website.

5 Linked Data Generation

In order to link all isolate local ontologies as the academic linked data, we need to match these ontologies. The linguistic-based method is a popular ontology matching techniques[12-13]. For the reason that text in ontology can describes some semantics, the linguistic-based matching method can discover matching results by calculating similarities between text documents For an academic conference local ontology, it contain regular and abundant text, therefore, the linguistic-based method is suitable. We use the ontology matching API provided by ontology matching system Lily [13] to discover matching results between local ontologies. Lily is an excellent matching system and can produce high quality matching results.

Our ontology matching strategy is calculating matches for each two ontologies, then associate all ontologies into the linked data by these matches. This strategy has the benefit of handling a number of generated local ontologies, but its disadvantage is consuming a lot of time for matching many ontologies. Fig. 2 shows part of linked data for three conferences. If two instances are matched, they can be combined in the linked data graph.

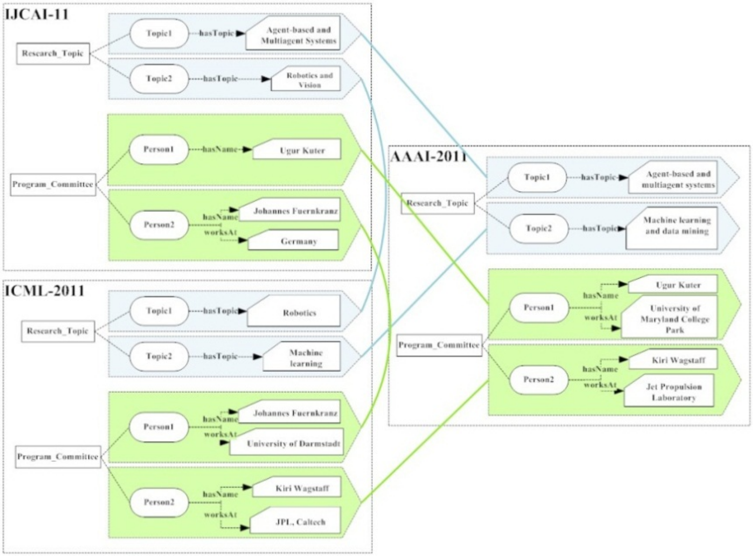

Fig. 2. Part of the linked data

6 Experiments

6.1 System Implementation and Dataset

The system is mainly implemented in Java, and the Web page segmentation module is implemented in C#. We also use Weka, an open-source machine learning library, to classify text blocks. Our experimental results are obtained on a PC with 2.40GHz CPU, 2GB RAM and Windows 7.

We collect 50 academic conference Web sites in computers science field, which has 283 different Web pages and 10028 labeled text blocks. In order to evaluate our approach, 10 students manually tag all the text blocks as reference results. The results are saved in CSV files.

We randomly select 10 sites which contain 62 pages as training dataset for constructing bayes network model. Other 40 conference Websites are used as test dataset. We use Precision, Recall and F1-Measure as criteria to measure the system performance.

6.2 Experimental Results and Analysis

First experiment is verifying the complete tree. Table 3 shows the vision tree results generated by VIPS and the complete tree results generated by our new algorithm on 15 conference websites. We can see that the complete trees have more leaf nodes than vision trees. It means our algorithm can find more text blocks than VIPS.

The experimental results of removing noise blocks are given in Table 4. We can observe some facts: (1) There are many noise blocks in the complete tree. In some websites, almost half of all blocks are noise blocks. (2) Our removing noise method can

Table 3. Experimental results of VIPS complete tree

Websites	Vision tree by VIPS		VIPS complete tree	
	Nodes	Leaf nodes	Nodes	Leaf nodes
AAAI-10	106	82	109	85
CIKM-11	109	80	119	90
ICDE-10	94	69	101	76
ICDM-10	131	98	143	110
ICSE-11	145	99	153	107
INFOCOM-10	143	93	143	93
SIGIR-11	133	86	133	86
SIGMOD-10	86	59	86	59
SOSP-11	147	101	147	101
VLDB-10	117	81	123	87
AAAI-11	153	108	157	112
ACL-11	178	137	193	152
AIDE-11	170	121	174	125
ASAP-10	50	31	59	40
ASPLOS-11	60	39	76	55

Table 4. Experimental results of removing noise from VIPS complete tree

Websites	Initial complete tree		Remove noise complete tree		Removed node /Before removed	
	Nodes	Leaf nodes	Nodes	Leaf nodes	Nodes	Leaf nodes
AAAI-10	109	85	45	30	0.59	0.65
CIKM-11	119	90	64	32	0.46	0.64
ICDE-10	101	76	46	22	0.54	0.71
ICDM-10	143	110	105	70	0.27	0.36
ICSE-11	153	107	80	37	0.48	0.65
INFOCOM-10	143	93	103	69	0.28	0.26
SIGIR-11	133	86	75	42	0.44	0.51
SIGMOD-10	86	59	49	27	0.43	0.54
SOSP-11	147	101	125	85	0.15	0.16
VLDB-10	123	87	42	17	0.66	0.80
AAAI-11	157	112	102	68	0.35	0.39
ACL-11	193	152	158	110	0.18	0.28
AIDE-11	174	125	108	71	0.38	0.43
ASAP-10	59	40	46	17	0.22	0.58
ASPLOS-11	76	55	39	17	0.49	0.69
Avg.					**0.39**	**0.51**

remove average 39% noise nodes and 51% noise leaf nodes. Therefore, it will reduce the number of nodes should be processed in extraction and improve the efficiency.

The second experiment is the comparison between initial classification results and the results after post-processing. The results are obtained on 20 randomly websites. Table 5 evaluates all the classification results. We have two conclusions: (1) The initial classification results only have average 0.75 precision, 0.67 recall and 0.68 F1-measure. After post-processing, the classification results are improved to average 0.96

Table 5. Experimental results of text blocks classification

Category	Initial classification			Post-processing		
	Precision	Recall	F1	Precision	Recall	F1
DI	0.95	0.75	0.84	0.96	0.99	0.98
AR	0.84	0.91	0.87	0.92	0.98	0.95
TO	0.90	0.41	0.57	0.99	0.99	0.99
PO	0.80	0.69	0.74	0.99	0.98	0.99
PE	0.85	0.60	0.71	0.99	0.99	0.99
PI	0.79	0.72	0.75	0.97	0.97	0.97
CO	0.35	0.80	0.49	0.91	0.95	0.93
PA	0.50	0.50	0.50	1.00	1.00	1.00
Avg.	0.75	0.67	0.68	**0.96**	**0.98**	**0.97**

precision, 0.98 recall and 0.97 F1-measure. Therefore, the post-processing key roles in academic information extraction. (2) Some text blocks like DI, PO, PE and TO, which have clear vision and text content features, have better classification results. The average F1-measure on these blocks is 0.99.

We selected 5 websites: AAAI-11, ASPLOS-11, NIPS-11, ICPR-11 and PKDD-11, then analyze the generated ontologies. Table 6 shows the statistics of some kinds of generated ontology information. We can see that these ontologies have average 0.97 precision, 0.99 recall and 0.98 F1-measure. Therefore, our method can generate high quality academic ontologies for conference websites.

Finally, we match the 3109 generated ontologies, then integrate them as a linked data.

Table 6. Statistics of generated academic ontologies for conferences

websites		AR	DI	PI	PE	TO	CO
AAAI-11	C	7	58	2	51	81	0
	R	0	0	1	1	0	0
	M	0	0	0	3	0	0
ASPLOS-11	C	2	23	7	15	11	2
	R	1	0	0	0	0	0
	M	0	1	0	0	0	0
ICPR-11	C	1	18	0	0	24	0
	R	0	0	0	0	0	0
	M	0	1	0	0	0	0
NIPS-11	C	1	16	0	58	10	0
	R	0	0	0	0	0	0
	M	0	0	0	4	0	0
PKDD-11	C	1	26	1	26	0	0
	R	0	0	0	0	0	0
	M	0	1	0	2	0	0
Total	C	12	141	10	150	126	2
	R	1	0	1	1	0	0
	M	0	3	0	9	0	0
	Precision	0.92	1.00	0.91	0.99	1.00	1.00
	Recall	1.00	0.98	1.00	0.94	1.00	1.00
	F-measure	0.96	0.99	0.95	0.97	1.00	1.00

7 Conclusions

This paper addresses the problem of extracting academic information from conference Web pages, then organizing academic information as ontologies and finally generating academic linked data by matching these ontologies. An new approach to extract academic information is proposed. A global ontology is used to describe the background domain knowledge, and then the extracted academic information of each website is organized as local ontologies. Finally, academic linked data is generated by matching local ontologies

Acknowledgments. This work is supported by the NSF of China (61003156 and 61003055) and the Natural Science Foundation of Jiangsu Province (BK2009136 and BK2011335).

References

1. Chang, C.-H., Kayed, M., Girgis, M.R., Shaalan, K.: A Survey of Web Information Extraction Systems. IEEE Transactions on Knowledge and Data Engineering 18(10), 1411–1428 (2006)
2. Hammer, J., Mchugh, J., Garcia-molina, H.: Semistructured data: the TSIMMIS experience. In: Proceedings of the 1st East-European Symposium on Advances in Databases and Information Systems (ADBIS 1997), St. Petersburg, Rusia (1997)
3. Arocena, G.O., Mendelzon, A.O.: WebOQL: Restructuring documents, databases, and Webs. In: Proceedings of the 14th IEEE International Conference on Data Engineering (ICDE 1998), Orlando, Florida, USA (1998)
4. Gruber, T.R.: Towards Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human Computer Studies (43), 907–928 (1995)
5. Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. IEEE Intelligent Systems 16(2), 72–79 (2001)
6. Suryanto, H., Compton, P.: Discovery of Ontologies from Knowledge Bases. In: Proceedings of the First International Conference on Knowledge Capture (2001)
7. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data-The Story So Far. International Journal Semantic Web and Information System 5(3), 1–22 (2009)
8. Bizer, C.: The Emerging Web of Linked Data. IEEE Intelligent Systems 24(5), 87–92 (2009)
9. Bizera, C., Lehmannb, J., Kobilarova, G., et al.: DBpedia - A crystallization point for the Web of Data. Journal of Web Semantics 7, 154–165 (2009)
10. Cai, D., Yu, S., Wen, J.-R., Ma, W.-Y.: VIPS: a Vision-based Page Segmentation Algorithm. Microsoft Technical Report (2003)
11. Liu, W., Meng, X., Meng, W.: ViDE: A Vision-Based Approach for Deep Web Data Extraction. IEEE Transactions on Knowledge and Data Engineering 22(3), 447–460 (2010)
12. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. In: Spaccapietra, S. (ed.) Journal on Data Semantics IV. LNCS, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)
13. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review 18(1), 1–31 (2003)
14. Wang, P., Xu, B.: Lily: ontology alignment results for OAEI 2009. In: The 4th International Workshop on Ontology Matching (OM 2009), Washington DC, USA (October 25, 2009)