International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Text Document Annotation and Retrieval Based on Content of the Document and Query Workload

Arunima P V[1], Ravinarayana B[2]

[1]PG Scholar, Department of Computer Science & Engineering, Mangalore Institute of Technology and Engineering, Mangalore, Karnataka (India)

[2]Associate Professor, Department of Computer Science & Engineering, Mangalore Institute of Technology and Engineering, Mangalore, Karnataka (India)

Abstract: ***Performing search and retrieval in large collection of textual data is a complex task. For effective searching of text documents, annotations are used. Annotations are the tags, keywords, attribute-value pairs, comments or summary that are attached to a document or a part of the document. Annotations can be considered as a structured representation of unstructured data. Since manually annotating each document in a large collection is not feasible, automatic annotation techniques are used. In this work, an automatic annotation generation technique based on the content of the document and query workload is introduced. Annotations are generated in the form of attribute-value pairs as they are more expressive than simple keyword annotations. The system generates both attributes and values for a document by analyzing the content of the document, annotations of the existing documents and query workload. These annotations are later used during the search and retrieval process for matching with the queries given by the user. As an enhancement to the system, a new ranking method for ranking the retrieved documents is also introduced.***

Keywords: Attribute-Value pair Annotation, Annotation Generation, Result Ranking, Query Workload

1. Introduction

We are familiar with many application domains where users collaborate and share their information such as news blogs, online shopping sites, disaster management networks, social networking groups, etc. When large amount of such information is shared, it constitute a huge collection of unstructured data. Search and retrieval process become difficult in such a collection. Hence, annotation of the textual data is necessary for effective searching in future. Annotations are the tags, keywords, attribute-value pairs, comments or summary that are attached to a document or a part of the document. Technically, annotations can be considered as semantic or content-based metadata. Annotations for a text document are generated either manually or automatically. With a lot of research being carried out in this area, the annotation generation systems have evolved from manual to semi-automatic and to fully automatic.

Most of the existing annotation generation systems generate annotations in the form of simple keywords or allow the annotator to choose from some predefined templates. Most of them allow users to annotate text documents only in an ad-hoc way. Annotation generation systems that use attribute-value pair annotation are generally more expressive. They contain more information when compared to simple keyword annotations. For example, 'country' = 'India' is an annotation in attribute-value pair form. Here country is the attribute and India is its corresponding value. When manually annotating documents with attribute-value pair annotation, users need to be more principled in their annotation efforts. They should know the underlying schema and which field types to use and when to use them. When there are too many fields to fill, the annotation task become very difficult which leads to most of the users skipping such annotation capabilities. Also, when such arbitrary annotations are used, their usefulness while searching is doubtful. While annotating, the users will not have much idea about which of them will be useful for future searches. Such difficulties often result in very basic or simple keyword annotations. Use of such basic annotations results in less effective searching in future. The proposed work aims at developing a data sharing platform which enables automatic annotation of text documents in the form of attribute-value pairs where both attributes and values are generated automatically and searching is performed effectively. CADS or Collaborative adaptive data sharing platform [1] is an existing piece of work which aims at generating attribute- value pair annotation. The work done in [1] solves the problem of generating attributes automatically for a text document. Then the user have to manually provide the corresponding values for the generated attributes. Such an annotation system solves the annotation generation problem partially. The user still has the burden of typing the values for each attribute that is generated. Hence, the proposed work, which automatically generates values along with attributes, further reduces the human effort in annotating. The system makes use of the content of the text document which is to be annotated, annotations of the existing documents and query workload in the process of generating annotations. Query workload means the queries that are being used by the users while searching for documents. The use of query workload as a parameter takes into account the user interest and hence it helps in generating annotations that will be more effective while searching.

2. Problem Statement and Proposed Solution

The proposed system generates both attributes and values automatically for each text document. The work done in the existing system CADS solves the problem of suggesting attributes for a document but values have to be entered manually. The primary goal of the proposed system is to reduce the human effort in annotating the documents.

1) Problem statement

Improving document annotation by suggesting both attributes and values for a document based on the content of the document, query workload and existing annotations, and also enhance the efficiency of document retrieval by introducing a ranking method.

The existing work solves only the problem of suggesting attributes for a new document that is uploaded to the database and the values have to enter manually by the person who uploads the document. The proposed work focuses on suggesting both attributes and values for each document that is uploaded to the database.

2) Proposed solution

The proposed system consist of two types of actors namely author and user. Author is the person who uploads the documents in to the system. He will be presented with the generated annotations which he will submit along with the document. User is the person who searches for documents through queries. The relevant documents matching the given queries will be presented to the user. The proposed system consist of two phases namely insertion phase and query phase.

2.1 Insertion Phase

In the insertion phase, first the author registers and logins to the system with author details. Author chooses the file to be inserted. The file gets uploaded and the text content in the file is parsed to remove stop words. Annotations in the form of attribute-value pairs are generated automatically. The author analyses the annotations and submits it. The annotations get saved along with the file.

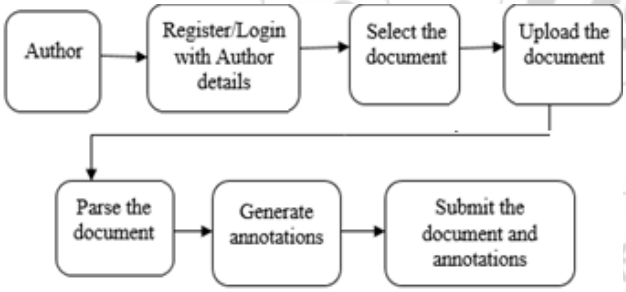

Figure 1: Insertion Phase

2.2 Query phase

In the query phase, the user registers and logins to the system with user details. The user will be presented with a query form. The user fills the query form with both attribute-value pairs or with either attributes or values. Then system searches for the appropriate documents. The resulting documents are ranked and final results are displayed.

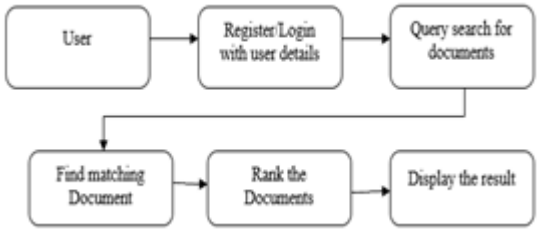

Figure 2: Query Phase

3. Module Description

The system implementation has been divided into four modules

- Document upload module
- Annotation generation module
- Query search module
- Result ranking module

3.1 Document Upload Module

In the document upload module, the author chooses the file to be uploaded in to the database using file chooser. The file is then parsed to remove stop words and special characters and resulting output is stored in a separate file which is then used for the annotation generation process.

3.2 Annotation Generation Module

In the annotation generation module, the annotations corresponding to the uploaded document are generated. Annotations are generated automatically based on the content of the document, query workload and annotations of the existing documents. A score is calculated for each attribute. Higher score implies the attribute is more relevant with respect to the content of the document and query workload. All the possible values corresponding to each attribute is also generated.

3.3 Query Search Module

In the query search module, the user will be presented with a query form. In order for retrieving the documents from the database, the user inputs the corresponding queries in the form of attribute-value pairs. Users can add more than one set of attribute-value pair combinations as queries to the system. The submitted queries will be added to the query workload which determines the querying value needed for annotation generation.

3.4 Result Ranking module

In result ranking module, the relevant documents that matches with queries submitted by the user are generated. Inorder for enabling only relevant documents to be retrieved, a scoring method is used for result ranking. The documents are ranked based on four parameters namely exact match, frequency of value, position of value and date of upload. A weightage is assigned to each of these parameters and a final score is calculated according to which the documents are sorted. Finally the documents are displayed in sorted order according to this score and the user can open the necessary document by clicking on it.

4. Implementation

4.1 Annotation Generation

In this section, solutions for the “Annotation Generation” problem is proposed. Two important factors needed to be considered while suggesting attributes for a document. First, the attributes must have high querying value with respect to

the query workload and second, the suggested attributes must be relevant to the content of the document. A probabilistic approach is used for combining these two factors. The two parameters necessary for suggesting attributes, the querying value and the content value are calculated as follows.

4.1.1 Querying value

The Querying value of an attribute indicates how relevant the attribute is with respect to the query workload. If an attribute occur frequently in the queries given by the user, then it shows that particular attribute is more relevant than others. Such an attribute should have high querying value. The querying value thus depends on the number of queries that contain that particular attribute as well as the total number of documents in the query workload. It is calculated by the following equation.

Querying value of a particular attribute = (Number of queries with that particular attribute)/(Total number of queries in the query workload)

$$QVA = \frac{WA}{W} \quad (1)$$

Here QVA is Querying value of the attribute A. WA is the number of queries in the query workload that are attributed with the attribute A. W is the total number of queries in the query workload. To avoid getting zero value for the querying value when no query is attributed with the attribute A, we use Laplace smoothing by adding 1 in numerator and denominator [27]. Thus the equation becomes:

$$QVA = \frac{WA+1}{W+1} \quad (2)$$

4.1.2 Content value

The content value of an attribute indicates how relevant the attribute is with respect to the content of the text document uploaded by the author. When the attributes are generated only based on the query workload, and the uploaded document does not contain any value corresponding to the generated attribute, that attribute is useless. Hence we need to calculate the content value which shows the relevance of the attribute for the uploaded document. For a given attribute and for each word in the content of the document, we use five parameters for this purpose. They are:

- Total number of document in the document collection
- Number of documents annotated with that particular attribute
- Number of documents annotated with that particular attribute and the given word
- Number of documents not annotated with that particular attribute
- Number of documents not annotated with that particular attribute and the given word

The content value is calculated in two parts.

The first part corresponds to the documents which are annotated with that particular attribute.

$$CV1 = \frac{DAw}{(DA+D)} \quad (3)$$

To avoid zero probabilities, we use Laplace smoothing by adding 1 at numerator and denominator. Thus the equation becomes:

$$CV1 = \frac{DAw+1}{(DA+D+1)} \quad (4)$$

The second part corresponds to the documents which are not annotated with that particular attribute.

$$CV2 = \frac{DAw}{(DA+D)} \quad (5)$$

To avoid zero probabilities, here also we use Laplace smoothing by adding 1 at numerator and denominator. Thus the equation becomes:

$$CV2 = \frac{DAw+1}{(DA+D+1)} \quad (6)$$

Finally, the content value of that attribute with respect to a given word is calculated by dividing the first part by second.

$$CV = \frac{CV1}{CV2} \quad (7)$$

That is:

$$CV = \frac{DAw+1/(Da+D+1)}{DAw+1/(DA+D+1)} \quad (8)$$

4.1.3 Score Calculation

After calculating querying value and content value, they are combined together to obtain a score for the attribute. The attributes are ranked depending on this score. Top K attributes are generated in the descending order of their scores for each document. K can have any value which must be set initially before uploading the document. Otherwise, all the attributes in the query workload will be generated for each document. The score of each attribute is calculated as follows:

$$\text{Score (A)} = QV * CV \quad (9)$$

$$\text{Score (A)} = \frac{WA+1}{W+1} * \frac{DAw + \frac{1}{DA+D+1}}{DAw + \frac{1}{DA+D+1}} \quad (10)$$

The content value of each attribute depends on the content of the document uploaded and hence it calculated at run time. On the other hand, the querying value depends only on the query workload and is independent of the document that is uploaded. Hence there is no need to calculate querying value of each attribute during runtime. Instead, we keep a precomputed list of attributes along with their querying values which are updated periodically based on the queries generated in the query workload. This reduces the runtime of the system in generating annotations when a document is uploaded.

4.1.4 Value Generation

New attributes will be added to the attribute list periodically. Along with it, a list of the most possible values for each attribute is kept. The list is populated based on the query workload and previous annotations. When a new document is uploaded, all the new values it have are mapped with the corresponding attribute. Same is the case with new queries submitted to the system. Multiple occurrences of a particular value are rejected, that is, the list consist of only a single occurrence of a particular value. The values in the list are compared with the content of the document to suggest the appropriate value for each attribute generated. The values are suggested for each attribute based on the order in which they appear in the document. Multiple occurrences of values for a particular attribute is generated by using a dropdown

list. Thus the author can choose the most desired value if a document contain multiple values for a particular attribute. After all the annotations that the system identified are generated the author can add the annotation and is provided with option to edit the generated annotations or to add new annotations. Finally, the user submits the annotations along with the document. All the relevant attributes along with possible values are generated. The user is provided option to change the value a paticular attribute with a drop down list or can type new values in the space provided.

4.2 Result Ranking

The documents are ranked based on four parameters.

- Exact match
- Frequency of value
- Position of value
- Date of upload

A weightage is assigned to each of these parameters and a final score is obtained according to which the documents are sorted. Since the exact match of the queries with the annotation of the document is more important, it is given 80 percentage weightage. Frequency of value implies the number of times the value in the query is contained in the document. When multiple queries are given, the sum of occurrences of all the values is calculated. It is then normalized with respect to total number of words in the file. The weightage of this parameter is 10 percent. The first two parameters are taken as positive values. The third parameter given the index of the value in the query. If multiple queries are given, the value with the least index is taken. This index is calculated by counting the number of words that occur in the document before this particular value. This parameter has a weightage of 1 percent and since this value has to be minimum for higher relevance, it is taken as negative. Finally the difference between date of upload and current date is calculated and this parameter is given 9 percent weightage and is taken as negative.

$$\text{Score} = 0.8 * e + 0.1 * f - 0.01 * i - 0.09 * d \quad (11)$$

Here e is the number of attribute-value pairs in the query workload that exactly matches which the given queries. f is the number of the times the values in the given query occur in the text content normalized with respect to the total number of words in the document. i is the least index among the values in the annotation with respect to the text content and d is the difference between the current data and the date of upload of the document. After the score is calculated with these parameters, the corresponding documents are retrieved and displayed in sorted order according to this score. The user can open the corresponding document by clicking on it.

5. Performance Evaluation

Performance of the system is evaluated in terms of three parameters.

- Precision
- Recall
- F-Measure

Precision indicates the fraction of retrieved instances that are relevant and recall indicates the fraction of relevant instances that are retrieved. Both precision and recall are based on an understanding and measure of relevance. F-measure gives the harmonic mean of precision and recall. Inorder for calculating the precision and recall of the system, the True positive (TP), False Positive (FP) and True negative (TN) values are evaluated and recorded for 50 samples of documents in both cases when only attributes are suggested (as in existing system CADS) and when both attributes and values are suggested (as in proposed system).

Amazon product reviews about various electronic products like camera, mobile phones, tablets, etc. are used for training the system. The system is populated with 500 queries generated based on the popularity of the terms in Google trends. Around 35 different attributes are used in queries. The most frequently used attributes in query workload and their number of occurrences are:

- product(81)
- brand (66)
- model (42)
- primarycam (27)
- modelname (26)
- sensorresolution (26)
- display (24)
- ram (21)
- colour (19)

The values obtained while uploading each product description document are recorded and the summary of the evaluation is shown in table 1.1.

Table I: Performance Evaluation Summary

Document	Attribute only case			Proposed System		
	TP	FP	TN	TP	FP	TN
1-10	46	44	154	86	49	114
11-20	71	29	129	128	38	71
21-30	77	23	123	137	29	63
31-40	84	16	116	160	18	40
41-50	86	14	114	160	16	40

Based on these values, the precision, recall and f-measure values are calculated using the below formulas

Precision=True Positives / (True Positives+False Positives) = TP/(TP+FP) (12)

Recall=True Positives / (True Positives + False Negatives) = TP/(TP+FN) (13)

F-measure = 2*precision*recall/(precision + recall) (14)

The resulting values are shown in Table 2.

Table II: Precision, Recall and F-Measure

Document	Attribute only case			ProposedSystem		
	Precision	Recall	F-Measure	Precision	Recall	F-Measure
1-10	51.11	23	31.72	63.7	42	50.62
11-20	71	35.5	47.33	77.1	63	69.84
21-30	77	38.5	52.33	82.53	68.5	74.86
31-40	84	42	56	89.88	80	84.65
42-50	86	43	57.33	90.9	80	85.1

Figure 1.3 shows the precision graph between the attribute only suggestion case (existing system) and attribute-value suggestion case (proposed system). Figure 1.4 and 1.5 shows the recall graph and f-measure graph respectively between the same.

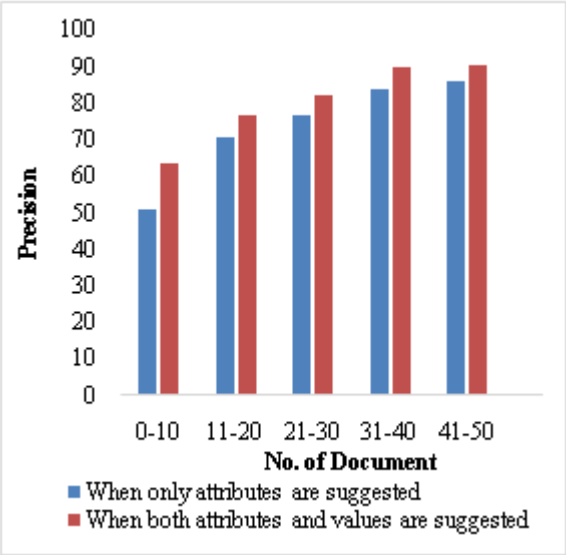

Figure 1.3: Precision graph

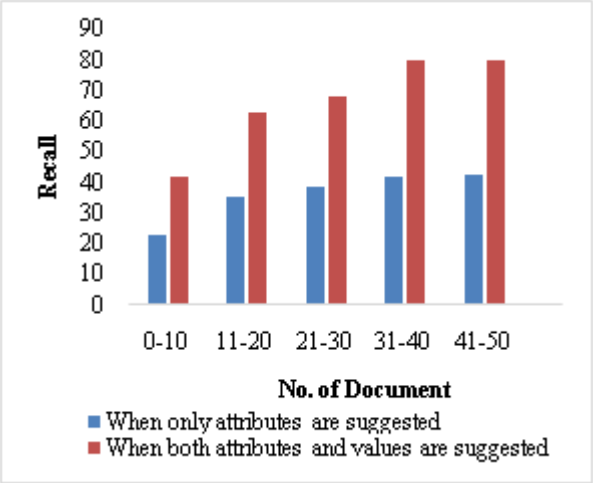

Figure 1.4: Recall graph

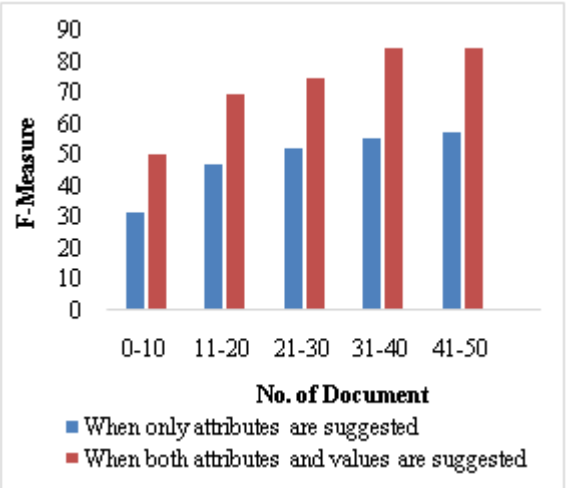

Figure 1.5: f-measure graph

6. Conclusion

Annotation of the document is necessary for effective searching in a data sharing platform. Adaptive techniques to suggest annotations for a document, while trying to satisfy the user querying needs is proposed. The main goal of this system is to reduce the human effort in manually annotating each document. When the existing system solved only the problem of suggesting attributes where users had to manually enter the values, the proposed work solves the problem of suggesting values along with attributes automatically and thus maximising the reduction in human effort. The system combines query workload and previous annotations along with the content of the document in order for generating attributes and corresponding values.

7. Acknowledgment

I take this opportunity to thank Prof. Ravinarayana B, Prof. P V Bhat & Prof. DrNagesh H R, for their valuable guidance and for providing all the necessary support to accomplish this research. I would like to extend my gratitude towards my beloved Principal G L Eschar Prasad for his great support

References

[1] VagelisHristidis, Eduardo J.Ruiz and Panagiotis G. Ipeirotis, ”Facilitating document annotation using content and querying value”, Knowledge and Data Engineering, IEEE Transactions, 2014.

[2] Berners-Lee, Tim, James Hendler, and Ora Lassila, ”The semantic web.”,Scientific american 284.5 (2001): 28-37.

[3] Wu, Xindong, ”Data mining with big data.”, Knowledge and Data Engineering,IEEE Transactions on 26.1 (2014): 97-107.

[4] Cimiano, Philipp, Siegfried Handschuh, and Steen Staab, ”Towards theself-annotating web.”, Proceedings of the 13th international conference on World WideWeb. ACM, 2004.

[5] Breitman, Karen Koogan, and J. C. Sampaio do Prado Leite, ”Ontology asa requirements engineering product.”, Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International. IEEE, 2003.

[6] Cimiano, Philipp, GnterLadwig, and Steen Staab, ”Gimme the context:context-driven automatic semantic annotation with C-PANKOW.” Proceedings of the 14th international conference on World Wide Web. ACM, 2005.

[7] Maynard, Diana. ”Multi-source and multilingual information extraction.”,Expert Update 6, No. 3 (2003): 11-16.

[8] Cunningham H., Maynard D., Bontcheva K., Tablan V., ”GATE: A Frameworkand Graphical Development Environment for Robust NLP Tools and Applications.” In Proc. of the ACL’02.

[9] Giannopoulos, Giorgos, Nikos Bikakis, Theodore Dalamagas, and TimosSellis, ”GoNTogle: a tool for semantic annotation and search.”, The Semantic Web: Research and Applications. Springer Berlin Heidelberg, 2010. 376- 380.

[10] Bikakis, Nikos, Giorgos Giannopoulos, Theodore Dalamagas, andTimosSellis. ”Integrating keywords and semantics on document annotation and search.”, In On the Move to Meaningful Internet Systems, OTM 2010, pp. 921-938. Springer Berlin Heidelberg, 2010.
[11] Popov, Borislav, AtanasKiryakov, Angel Kirilov, DimitarManov,DamyanOgnyano, and Miroslav Goranov. ”KIM: semantic annotation platform.”, In The SemanticWeb-ISWC 2003, pp. 834-849. Springer Berlin Heidelberg, 2003.
[12] Handschuh, Siegfried, Steen Staab, and Fabio Ciravegna. ”S-CREAM:semi-automatic creation of metadata.” In Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, pp. 358- 372. Springer Berlin Heidelberg, 2002
[13] Dill, Stephen, NadavEiron and David Gibson, ”SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation.” In Proceedings of the 12th international conference on World Wide Web, pp.178-186. ACM, 2003.
[14] Banko, Michele, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. ”Open information extraction for the web.” In IJCAI, vol. 7, pp. 2670-2676. 2007.
[15] Kogut, Paul A., andWilliam S. Holmes III. ”AeroDAML: Applying Information Extraction to Generate DAML Annotations fromWeb Pages.” Semannot@ K-CAP, 2001.
[16] Laclavik, Michal, et al. ”Ontology based text annotation-OnTeA.” Frontiers in Artificial Intelligence and Applications 154 (2007): 311.39
[17] Pudota, Nirmala, et al. ”Automatic keyphrase extraction and ontology mining for contentbased tag recommendation.” International Journal of Intelligent Systems 25.12, 2010
[18] Khattak, Asad Masood, N. Ahmad,Jibran Mustafa, ”Context-Aware Search in Dynamic Repositories of Digital Documents.”, Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on. IEEE, 2013.
[19] Lu, Yiyao, et al. ”Annotating search results from Web databases.” Knowledge and Data Engineering, IEEE Transactions on 25.3 (2013): 514-527.
[20] Harith, A., et al. ”Automatic ontology-based knowledge extraction and tailored biography generation from the web.” IEEE Intelligent Systems 18.1 (2003): 14-21.
[21] Iwata, Tomoharu, Takeshi Yamada, and Naonori Ueda. ”Modeling Noisy Annotated Data with Application to Social Annotation.” Knowledge and Data Engineering, IEEE Transactions on 25.7 (2013): 1601-1613.
[22] Brut, Mihaela M., Florence Sedes, and Stefan Daniel Dumitrescu. ”A semantic-oriented approach for organizing and developing annotation for elearning.” Learning Technologies, IEEE Transactions on 4.3 (2011): 239- 248.
[23] Handschuh, Siegfried, Raphael Volz, and SteenStaab. ”Annotation for the deep Web.” IEEE Intelligent Systems 18.5 (2003): 42-48.
[24] Heflin, Je, and James Hendler, ”A portrait of the SemanticWeb in action.”, Intelligent Systems, IEEE 16.2 (2001): 54-59.
[25] Vargas-Vera, Maria, ”MnM: Ontology driven semi-automatic and automatic support for semantic markup.”, Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web. Springer Berlin Heidelberg, 2002. 379-391.
[26] VagelisHristidis and Eduardo J. Ruiz, ”Cads: A collaborative adaptive data sharing platform.”, VLDB Workshop on Personalized Access, Profile Management, and Context Awareness in Databases (PersDB 2009).
[27] C. D. Manning, P. Raghavan, and H. Schutze, ”Introduction to InformationRetrieval.”, 1st ed. Cambridge University Press, July 2008.