Issue 7 volume 6 Nov.- Dec. 2017
ISSN 2249-9954

FACETS DISCOVERY FOR QUERIES USING QDMINER

PILLA BHARATHI [#1], **DR. B.PRAJNA**[#2]

[#1] M.Tech Scholar, Department of Computer Science and Systems Engineering,
Andhra University College of Engineering (A),
Visakhapatnam, AP, India.

[#2] Professor, Department of Computer Science and Systems Engineering,
Andhra University College of Engineering (A),
Visakhapatnam, AP, India.

ABSTRACT

With the amazing progress of both computer hardware and software, a vast amount of data is generated and collected daily. There is no doubt that data is meaningful only when one can extract the hidden information inside them. Query facets give knowledge to the users about a query without browsing hundreds of pages. Query facets are multiple groups of words or phrases that explain and summarize the content covered by a query. But finding query facets is a challenging problem. We assume that important aspects of a query is usually presented and repeated in the query's top retrieved documents. In this paper, we propose a solution called QDMiner, in order to mine the query facets. Using QDMiner, Query facets are mined automatically by extracting and grouping the frequent lists which are collected from the top search results. Our experimental results clearly tell that our proposed approach mines useful facets for the given query when compared with various primitive methods that are already available.

Key Words: Data Mining, QDMiner, Query Facet, Faceted Search

I. INTRODUCTION

We address the problem of finding query facets are multiple groups of words or phrases that explain and summarize the content covered by a query. A query facet is a set of items which describe and summarize one important aspect of a query. A query may have multiple facets that summarize the information covered by a query from different perspectives.

EXAMPLE

1. Facets for the query "watches" cover the knowledge about watches in five unique aspects, including brands, gender categories, supporting features, styles and colours.

2. The query "visit Beijing" has a query facet about popular resorts in Beijing (Tiananmen square, forbidden city, summer palace,....)

We propose aggregating frequent lists within the top search results to mine query facets and implement a system called QDMiner. More specifically, QDMiner extracts lists from the top search

Issue 7 volume 6 Nov.- Dec. 2017
ISSN 2249-9954

results, groups them into clusters[21], [23] based on the items they contain, then ranks the clusters and items based on how the lists and items appear in the top results. Two models, the Unique Website Model and the Context Similarity Model are proposed, to rank query facets. In the Unique Website Model, it is assumed that lists from the same website might contain duplicated information, whereas different websites are independent and each can contribute a separated vote for weighting facets.

We propose Context Similarity Model, in which the fine-grained similarity between each pair of lists is modelled. Query facets provide interesting and useful knowledge about a query and thus can be used to improve search experiences in many ways:

First, we can display query facets together with the original search results in an appropriate way. Thus, users can understand some important aspects of a query without browsing tens of pages.

Second, query facets may provide direct information or instant answers that users are seeking.

Third, query facets may also be used to improve the diversity of the ten blue links.

Query facets also contain structured knowledge covered by the query, and thus they can be used in other fields besides traditional web search, such as semantic search or entity search. The main goal of mining facets is different from query recommendation. The former is to summarize the knowledge and information contained in the query, whereas the latter is to find a list of related or expanded queries. However, query facets include semantically related phrases or terms that can be used as query reformulations or query suggestions sometimes. Different from transitional query suggestions, we can utilize query facets to generate structured query suggestions, i.e., multiple groups of semantically related query suggestions. This potentially provides richer information than traditional query suggestions and might help users find a better query more easily.

EXAMPLES
TABLE 1

QUERY:	**WATCHES**
FACETS	1. cartier, breitling, omega, citizen, tag heuer, bulova, casio, rolex, audemars, Piguet, Seiko, accutron, Movado,….
	2. men's, women's, kids, unisex
	3. analog, digital, chronograph, analog digital, quartz, mechanical, . . .
	4. Dress, casual, sport, fashion, luxury, bling, pocket, . .
	5. black, blue, white, green, red, brown, pink, orange, yellow, . . .

TABLE 2

QUERY:	**LOST**
FACETS	1. season 1, season 6, season 2, season 3, season 4, season 5
	2. matthew fox, naveen andrews, evangeline lilly, josh holloway, jorge garcia, Daniel dae kim, michael emerson
	3. jack, kate, locke, sawyer, claire, sayid, hurley, desmond, boone, charlie, ben, juliet, sun, jin, . . .
	4. what they died for, across the sea, what kate does, the candidate, the last recruit, everybody loves hugo, the end, . . .

II. RELATED WORK

Mining query facets is related to several existing research topics. In this section, we briefly review them and discuss the difference from our approach**.**

FACETED SEARCH

Most existing faceted search and facets generation systems [1], [2], [3], [8], [9], are built on a specific domain (such as product search) or predefined facet categories. For example, Dakka and Ipeirotis [9] introduced an unsupervised technique for automatic extraction of facets that are useful for browsing text databases. Facet hierarchies are generated for a whole collection, instead of for a given query. Li et al. proposed Facetedpedia [8], a faceted retrieval system for information discovery and exploration in Wikipedia. Facetedpedia extracts and aggregates the rich semantic information from the specific knowledge database Wikipedia. In this paper, it explores to automatically find query dependent facets for open-domain queries based on a general web search engine. Facets of a query are automatically mined from the top web search results of the query without any additional domain knowledge required

DYNAMIC FACETED SEARCH IN STRUCTURED DATABASES

Minimum-effort driven navigational techniques are proposed for enterprise database systems based on the faceted search paradigm. These proposed techniques dynamically suggest facets for drilling down into the database such that the cost of navigation is minimized. At every step, the system asks the user a question or a set of questions on different facets and depending on the user response, dynamically fetches the next most promising set of facets, and the process repeats. Facets are selected based on their ability to rapidly drill down to the most promising tuples, as well as on the ability of the user to provide desired values for them. These facet selection algorithms also work in conjunction with any ranked retrieval model where a ranking function imposes a bias over the user preferences for the selected tuples.

AUTOMATIC EXTRACTION OF USEFUL FACET HIERARCHIES FROM TEXT DATABASES

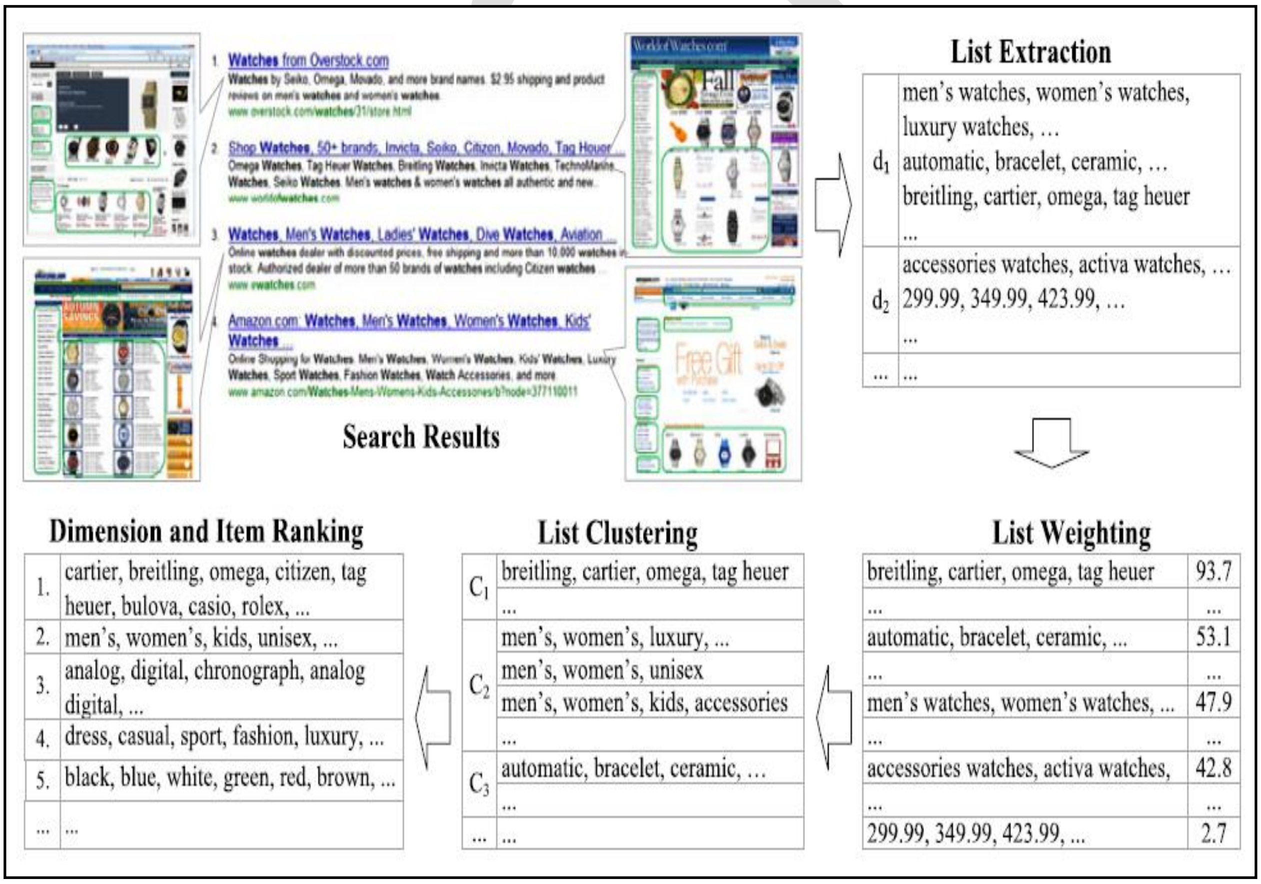
An unsupervised technique is presented for automatic extraction of facets useful for browsing text databases. In particular, through a pilot study, it is observed that facet terms rarely appear in text documents, showing that external resources are needed to identify useful facet terms. For this, important phrases in each document are identified at first. Then, each phrase is expanded with "context" phrases using external resources, such as WordNet and Wikipedia, causing facet terms to appear in the expanded database. Finally, the term distributions in the original database are compared with the expanded database to identify the terms that can be used to construct browsing facets.

III. THE PROPOSED MINING QUERY FACTES USING QDMiner ALGORITHM

In this section we mainly discuss about the proposed mining query facets using QD miner Aproach, where the proposed approach is clearly explained in the figure 3 and also the detailed explanation about the system is explained in the next section i.e. in modules phase. Now let us discuss about this proposed method in detail:

From the below figure 1, we can clearly define the working flow of QDMiner, given a query q, we retrieve the top K results from a search engine and fetch all documents to form a set R as input. Then, query facets are mined by:

1. **List extraction** Lists and their context are extracted from each document in R. "men's watches, women's watches, luxury watches, . . ." is an example list extracted.
2. **List weighting** All extracted lists are weighted, and thus some unimportant or noisy lists, such as the price list "299.99, 349.99, 423.99, . . ." that occasionally occurs in a page, can be assigned by low weights.
3. **List clustering** Similar lists are grouped together to compose a facet. For example, different lists about watch gender types are grouped because they share the same items men's" and "women's".
4. **Facet and item ranking** Facets and their items are evaluated and ranked.
   For example, the facet on brands is ranked higher than the facet on colours based on how frequent the facets occur and how relevant the supporting documents are. Within the query facet on gender categories, "men's" and "women's" are ranked higher than "unisex" and "kids" based on how frequent the items appear, and their order in the original lists.



1. Watches from Overstock.com
Watches by Seiko, Omega, Movado, and more brand names. $2.95 shipping and product reviews on men's watches and women's watches.
www.overstock.com/watches/31/store.html

2. Shop Watches, 50+ brands, Invicta, Seiko, Citizen, Movado, Tag Heuer ...
Omega Watches, Tag Heuer Watches, Breitling Watches, Invicta Watches, TechnoMarine Watches, Seiko Watches. Men's watches & women's watches all authentic and new...
www.worldofwatches.com

3. Watches, Men's Watches, Ladies' Watches, Dive Watches, Aviation ...
Online watches dealer with discounted prices, free shipping and more than 10,000 watches in stock. Authorized dealer of more than 50 brands of watches including Citizen watches ...
www.ewatches.com

4. Amazon.com: Watches, Men's Watches, Women's Watches, Kids' Watches ...
Online Shopping for Watches, Men's Watches, Women's Watches, Kids' Watches, Luxury Watches, Sport Watches, Fashion Watches, Watch Accessories, and more
www.amazon.com/Watches-Mens-Womens-Kids-Accessories/b?node=377110011

Search Results

List Extraction

d_1	men's watches, women's watches, luxury watches, ... automatic, bracelet, ceramic, ... breitling, cartier, omega, tag heuer ...
d_2	accessories watches, activa watches, ... 299.99, 349.99, 423.99, ...
...	...

List Weighting

breitling, cartier, omega, tag heuer	93.7
...	...
automatic, bracelet, ceramic, ...	53.1
...	...
men's watches, women's watches, ...	47.9
...	...
accessories watches, activa watches,	42.8
...	...
299.99, 349.99, 423.99, ...	2.7

List Clustering

C_1	breitling, cartier, omega, tag heuer
	...
C_2	men's, women's, luxury, ...
	men's, women's, unisex
	men's, women's, kids, accessories
	...
C_3	automatic, bracelet, ceramic, ...
	...
...	...

Dimension and Item Ranking

1.	cartier, breitling, omega, citizen, tag heuer, bulova, casio, rolex, ...
2.	men's, women's, kids, unisex, ...
3.	analog, digital, chronograph, analog digital, ...
4.	dress, casual, sport, fashion, luxury, ...
5.	black, blue, white, green, red, brown, ...
...	...

FIGURE. 1. THE ARCHITECTURE OF PROPOSED QDMiner ALGORITHM

IV. . IMPLEMENTATION AND ITS METHODOLOGY

Implementation is the stage where the theoretical design is converted into programmatically manner. In this stage we will divide the application into a number of modules and then coded for deployment. We have implemented the proposed concept on Java programming language. The front end of the application takes JSP, HTML, CSS & Java Beans and as a Back-End Data base we took My SQL data base along with a Real time Products Data Set from a local web site which run under tomcat. The application is divided mainly into following 4 modules. They are as follows:

1. List Extraction Module
2. List Weighting Module
3. List Clustering Module
4. Facet and Item Ranking Module

Now let us discuss about each and every module in detail as follows:

1. LIST EXTRACTION MODULE

Lists are extracted from the data for a given query. Whenever, a query is given, all the lists which are related to the query and which are in the top search results are extracted. These lists are then processed to remove useless data. We find that quality of query facets is affected by the quality and the quantity of search results. Using more results can generate better facets at the beginning, whereas the improvement of using more results ranked lower than 50 becomes subtle.

2. LIST WEIGHTING MODULE

Some of the extracted lists are not informative or even useless. Some of them are extraction errors. These types of lists are useless for finding facets. So these lists must be ignored, and rely more on better lists to generate good facets. A good list contains items that are informative to the query. Therefore, aggregating and sorting of all lists of a query by their weights is implemented in this project. Important lists are commonly supported and they repeat in the top search results, whereas unimportant lists just infrequently appear in results. This makes it possible to distinguish good lists from bad ones, and to further rank facets in terms of importance.

Suppose, If “watches” is the search query and if the extracted lists contain items such as ratings, feedback, price etc., these items in the list are not so important and have low weights and are ignored later.

EXAMPLE

QUERY: WATCHES

LISTS

1) cartier, breitling, omega, citizen, tag heuer, bulova, casio, rolex, audemars piguet, seiko, accutron, movado, . . .
2) men’s, women’s, kids, unisex
3) analog, digital, chronograph, analog digital, quartz, mechanical, . . .

4)	dress, casual, sport, fashion, luxury, bling, pocket, . . .
5)	black, blue, white, green, red, brown, pink, orange, yellow, . . .
6)	good one, nice design, nice color, low price, budget piece
7)	rating 4.5, rating 2, 3, 1 , 3.5,…..
8)	good one, citizen, kids, analogue white, rating 4,…….
9)	2999.99, 3000.66, 900.7, 2600.56,……

Here, in the above example, the first 5 lists are good lists. Because, these lists gives details such as watch brands, gender, types of watches, style of watches and colors.
But, list 6 contains details like watches feedback. This data may not be that important to be considered as a list in the process of mining facets.
List 7 contains data such as ratings that is numerical data. Even this data may not be that important to be considered as a list for mining facets.
List 8 contains mixed data. This list is an extraction error. This list is also useless.
List 9 contains data such as prices. This type of numbers is also useless
So, from the above lists, lists 1-5 have higher weight and are found useful. These are considered as good lists.
And the lists 6-9 have lower weights and are found useless. These lists are considered as bad list and are ignored.

3. LIST CLUSTERING MODULE

Individual weighted lists are not used as query facets because:

(1) An individual list may inevitably include noise.
(2) An individual list usually contains a small number of items of a facet and thus it is far from complete;
(3) Many lists contain duplicated information. They are not exactly same, but share overlapped items.

To conquer the above issues, Similar lists are grouped together to compose facets.
Two lists can be grouped together if they share enough items.
We define distance d_l (l1, l2) between two lists l1 and l2 as
d_l (l1, l2)= 1-((l1∩l2)/min{│l1│, │l2│}).
Here l1∩l2 is the number of shared items within l1 and l2
This means that two groups of lists can only be merged together when every two lists of them are similar enough.

We use a modified QT (Quality Threshold) clustering algorithm to group similar lists. QT is a clustering algorithm that groups data into high quality clusters. Compared to other clustering algorithms, QT ensures quality by finding large clusters whose diameters do not exceed a user-defined diameter threshold. This method prevents dissimilar data from being forced under the same cluster and ensures good quality of clusters. In QT, the number of clusters is not required to be specified [15]-[18]. The QT algorithm assumes that all data is equally important, and the cluster that has the most number of points is selected in each iteration. In our problem, lists are not equally important. Better lists should be grouped first. We modify the original QT algorithm to first group highly weighted lists. The algorithm, which we refer to as WQT (Quality Threshold with Weighted data points), is described as follows.

STEP 1: Choose a maximum diameter Diamax and a minimum weight W_{min} for clusters.

STEP 2: Build a candidate cluster for the most important point by iteratively including the point that is closest to the group, until the diameter of the cluster surpasses the threshold Diamax. Here the most important point is the list which has the highest weight.

STEP 3: Save the candidate cluster if the total weight of its points w_c is not smaller than W_{min}, and remove all points in the cluster from further consideration.

STEP 4: Recurse with the reduced set of points.

Recall that the main difference between WQT and QT is that WQT tries to get more neighbors for important points, and generated clusters are biased towards important points.

Suppose we have six lists:
l1= (cartier, breitling, omega, citizen),
l2 = (breitling, omega, citizen, tag heuer),
l3= (breitling, omega, citizen, movie, music, book),
l4= (movie, music, book),
l5= (music, book, radio), and
l6= (movie, book, radio).

Their corresponding weights satisfy:

Sl1 >Sl2 > Sl3 > Sl4 > Sl5 >Sl6.

QT ignores their weights and generate a cluster (l3; l4; l5; l6) in the first iteration. Whereas WQT will generate a cluster (l1; l2; l3) for list l1. We prefer the second result, especially when Sl1 is much larger than Sl3. In addition, WQT is more efficient than QT, as it just builds one candidate cluster while QT builds a candidate cluster for each remaining point. Weight of a cluster is computed based on the rating of the lists. After the clustering process, similar lists will be grouped into a candidate query facet.

4. FACET AND ITEM RANKING MODULE

After the candidate query facets are generated, evaluate the importance of facets and items, and rank them based on their importance. Based on our motivation that a good facet should frequently appear in the top results, a facet c is more important if:

- The lists in c are extracted from more unique content of search results; (and)
- The lists in c are more important, i.e., they have higher weights.

We emphasize "**unique** "content here, because sometimes there are duplicated content and in a facet, the importance of an item depends on how many lists contain the item and its ranks in the lists. A better item is usually ranked higher by its creator than a worse item in the original list.

V. CONCLUSION

In this paper, we study the problem of finding query facets. We proposed a systematic solution, which we refer to as QDMiner, to automatically mine query facets by aggregating frequent lists from top search results. We create Human annotated data sets and apply existing metrics to mine quality query facets. Further, analyze the problem of duplicated lists, and find that facets can be improved by modelling fine-grained similarities between lists within a facet by comparing their similarities.

VI. REFERENCES

[1] O. Ben-Yitzhak, N. Golbandi, N. Har'El, R. Lempel, A. Neumann, S. Ofek-Koifman, D. Sheinwald, E. Shekita, B. Sznajder, and S. Yogev, "Beyond basic faceted search," in Proc. Int. Conf. Web Search Data Mining, 2008, pp. 33–44.

[2] M. Diao, S. Mukherjea, N. Rajput, and K. Srivastava,, "Faceted search and browsing of audio content on spoken web," in Proc. 19th ACM Int. Conf. Inf. Knowl. Manage., 2010, pp. 1029–1038.
[3] D. Dash, J. Rao, N. Megiddo, A. Ailamaki, and G. Lohman, "Dynamic faceted search for discovery-driven analysis," in ACM Int. Conf. Inf. Knowl. Manage., pp. 3–12, 2008.

[4] W. Kong and J. Allan, "Extending faceted search to the general web," in Proc.ACMInt. Conf. Inf. Knowl. Manage., 2014, pp. 839–848.

[5] T. Cheng, X. Yan, and K. C.-C. Chang, "Supporting entity search: A large-scale prototype search engine," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2007, pp. 1144–1146.

[6] K. Balog, E. Meij, and M. de Rijke, "Entity search: Building bridges between two worlds," in Proc. 3rd Int. Semantic Search Workshop, 2010, pp. 9:1–9:5.

[7] M. Bron, K. Balog, and M. de Rijke, "Ranking related entities: Components and analyses," in Proc. ACM Int. Conf. Inf. Knowl. Manage., 2010, pp. 1079–1088.

[8] C. Li, N. Yan, S. B. Roy, L. Lisham, and G. Das, "Facetedpedia: Dynamic generation of query-dependent faceted interfaces for wikipedia," in Proc. 19th Int. Conf. World Wide Web, 2010, pp. 651–660.

[9] W. Dakka and P. G. Ipeirotis, "Automatic extraction of useful facet hierarchies from text databases," in Proc. IEEE 24th Int. Conf. Data Eng., 2008, pp. 466–475.

[10] A. Herdagdelen, M. Ciaramita, D. Mahler, M. Holmqvist, K. Hall, S. Riezler, and E. Alfonseca, "Generalized syntactic and semantic models of query reformulation," in Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inf. retrieval, 2010, pp. 283–290.

[11] M. Mitra, A. Singhal, and C. Buckley, "Improving automatic query expansion," in Proc. 21st Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 1998, pp. 206–214.

[12] P. Anick, "Using terminological feedback for web search refinement:

A log-based study," in Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2003, pp. 88–95.

[13] S. Riezler, Y. Liu, and A. Vasserman, "Translating queries into snippets for improved query expansion," in Proc. 22nd Int. Conf. Comput. Ling., 2008, pp. 737–744.

[14] X. Xue and W. B. Croft, "Modeling reformulation using query distributions," ACM Trans. Inf. Syst., vol. 31, no. 2, pp. 6:1–6:34, May 2013.

[15] L. Bing, W. Lam, T.-L. Wong, and S. Jameel, "Web query reformulation via joint modeling of latent topic dependency and term context," ACM Trans. Inf. Syst., vol. 33, no. 2, pp. 6:1–6:38, eb. 2015.

[16] J. Huang and E. N. Efthimiadis, "Analyzing and evaluating query reformulation strategies in web search logs," in Proc. 18th ACM Conf. Inf. Knowl. Manage., 2009, pp. 77–86.

[17] R. Baeza-Yates, C. Hurtado, and M. Mendoza, "Query recommendation using query logs in search engines," in Proc. Int. Conf. Current Trends Database Technol., 2004, pp. 588–596.

[18] Z. Zhang and O. Nasraoui, "Mining search engine query logs for query recommendation," in Proc. 15th Int. Conf. World Wide Web, 2006, pp. 1039–1040.

[19] L. Li, L. Zhong, Z. Yang, and M. Kitsuregawa, "Qubic: An adaptive approach to query-based recommendation," J. Intell. Inf. Syst., vol. 40, no. 3, pp. 555–587, Jun. 2013.

[20] I. Szpektor, A. Gionis, and Y. Maarek, "Improving recommendation for long-tail queries via templates," in Proc. 20th Int. Conf.

[21] Document Clustering technique based on Noun Hypernyms S Prajna bodapati International Journal of Electronics & Communication Technology IJECT 2 (sp-1).

[22] Model based Clustering S B.Prajna National Conference on upcoming trends in IT.

[23] Clustering document trees based on similarity measure SM Prajna Bodapati, A Manasa Sudha Asian Journal of Computer Science and Information Technology.

[24] A Divide and Conquer strategy for Document Clustering using Reformed K-Means DBP V.R. Mounika International Journal of Scientific & Engineering Research 7 (10), 1053-1057.

[25] Document Clustering Technique based on Noun Hypernyms SM Bodapati Prajna International Conference on advances Computer Science, Communication & Bio.

VII. ABOUT THE AUTHORS

PILLA BHARATHI is currently pursuing her 2 Years M. Tech in Department of Computer Science and Systems Engineering (Specialization in Information Technology) at Andhra University College of Engineering, Visakhapatnam, Andhra Pradesh, India. Her area of interests includes Data Mining.

DR. B.PRAJNA is currently working as a Professor in Department of Computer Science and Systems Engineering at Andhra University College of Engineering, Visakhapatnam, Andhra Pradesh, India. She has more than 15 years of teaching experience. Her research interest includes Data Mining.