Developing Multi-Query Form for F-score Result

[1]Nazia Kausar, [2]V.B.Gaikwad
[1]Post graduate student, [2]Associate Professor,
Department of Computer Engineering,
[1]Shree L.R. Tiwari College of Engineering, Thane, Maharashtra India
[2]Terna College of Engineering, Nerul , Maharashtra India

***Abstract* - Large and heterogeneous data are maintained by modern database and web database. Dynamic Query Form is a narrative database query form interface, which is able to generate query forms dynamically. The presence of DQF is to catch a user feedback and rank to query form components helping him/her in making decisions. At each iteration the system automatically generates list of rank of form components and the user then includes the desired form components into the query form. Query form is filled by user and submitted to view the query result at each iteration. So, a query form could be dynamically refined until the user is satisfied with the query results. We calculate the expected F-measure/score to measure the goodness of a query form. A Selective model is developed for estimating the goodness of a query form in DQF. Our implementation evaluation demonstrates the effectiveness and efficiency of the system. This is an Independent approach for efficiently generating dynamic search interfaces over databases. Where system provide the interaction among the user, experts and IR system. If user in the requirement of experts, you can also submit a query to the hundreds of thousands of experts in our System. This is the process of search on the basis of query expansion. Widely used query expansion methods are Global Analysis and Local Analysis for this purpose. Here presents a relative study of query expansion with dynamic document analysis and thesaurus based analysis with the HTML parser.**

***Index Terms* – Generation Multi-Query Form, User Interaction, Query Result**

I. INTRODUCTION

Most usable form of interface for users to querying the databases is Query Form. For this the predefined query was developed & accessed only by developers or Database Administrator in information retrieval system. So, we implement a new technique for this which is nor predefined and basically use to accept the user feedback and rank the form components effectively & efficiently. This new method is Dynamic Query Form to overcome the problems of existing customised query form system. We also implement the query expansion technique to minimize the information retrieval time.

This paper explained DQF, is a Multi-Query form interface in which users are able to generate query forms dynamically and to overcome problems in static query forms and customized query form. At each step system automatically creates a ranking list of form components by the using of Page-One-Rank algorithm & then user selects the form components of their interest into the query form. The query form generation is an iterative process. A query form generation process could be dynamically repeated until the user satisfies with the responded query results. It utilized the expected F-measure for measuring the goodness of a query form. This paper explained DQF, database query form interface in which users are able to generate query forms dynamically and to overcome problems in static query forms and customised query form. Here we are implementing a system which includes dynamic query form in the combination of relative study of the query expansion with HTML Parser in simple IR system. Query expansion approach with HTML parser, extracts text documents only but the search results consists of various other file formats.

II. PROPOSED SOLUTION

This is an approach for developing and accessing the dynamic query interfaces over databases. Where system plays the interaction among these entities: user, experts and IR system. Here the experts are available to developed the DQF and end users are able to access this DQF ,if end user in the requirement of experts, then user can also submit a query to the hundreds of thousands of experts in our System. Where Experts can make search on the basis of query expansion more frequently.

Design Consideration:

A. Generation of Dynamic Query Form:-

We use Multi-Query form interface in which users are able to generate query forms dynamically and to overcome problems in static query forms and customized query form

1. Dynamic Query Form (DQF) recommends a ranked list of query form components to the user.
2. The user has to select the desired form components into the current query form. The user fills out the current query form and submits a query.
3. DQF will execute the query and the results are shown.
4. The feedback about the query results is provided by user.

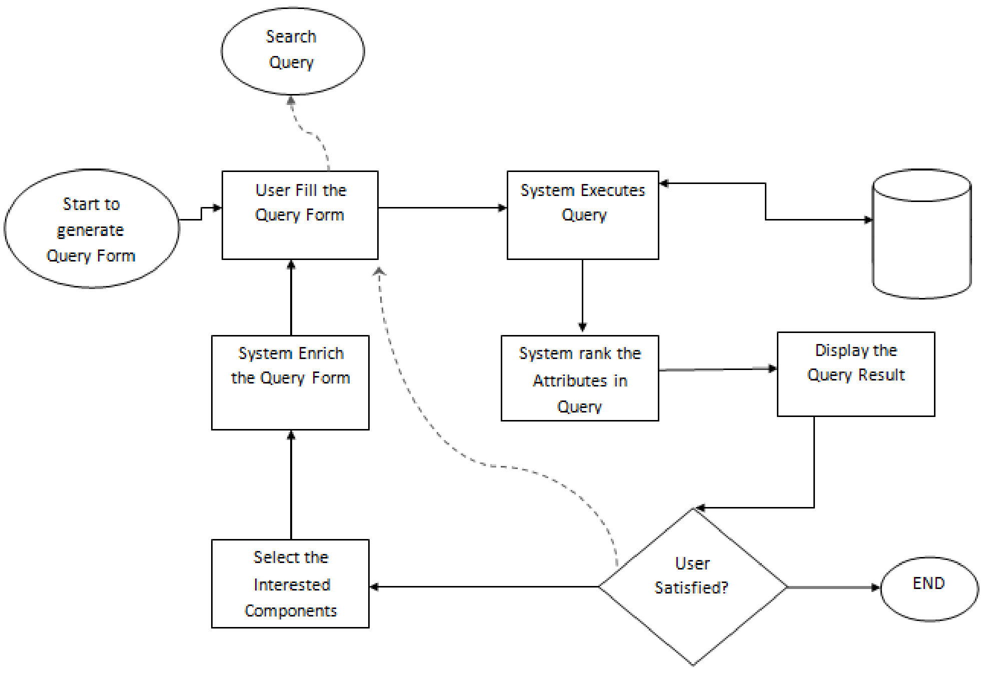

Search Query
Start to generate Query Form
User Fill the Query Form
System Executes Query
System Enrich the Query Form
System rank the Attributes in Query
Display the Query Result
Select the Interested Components
User Satisfied?
END

Figure 1: Flow Chart of QEDQFEDR

B. Access Dynamic Query Form:-

According to identified modules, task has been specified in their categories:

1. **Query Form Enrichment**

1.1 DQF system provides a facility of ranked list items of query composition to the end user.

1.2 User having the selection option among the components of current Query Form.

2. **Query Execution**

2.1 User provided with filling form facility and submission of their required query.

2.2 After submission, system will execute the query and get back to user with result.

2.3 If expected result and achieved results are same the no requirement of feedback session.

2.4 Otherwise user has to give the feedback by desired selected items of the first result.

III. METHODOLOGY USED

The system is identified with the following modules along with functional requirements. According to user dynamically form generation analysis and minimal time retrieval requirement it have been investigated and summarized into these classes:

1. Multi-Query Form

2. Query-By-Form interface

3. F-measure

4. Page-One-Rank:-

4.1 Ranking Algorithm

4.2 Query Expansion with HTML Parser

1. Multi -Query Form:

This section focuses on a technique to generate the query form called "Multi -Query". This is the method where we can use queries to generate the dynamic form and use queries to filling & accessing the forms. But the first step is to convert the Customised Query Form methodology in simplest way of Dynamic Query Form according to the end users (non-IT person).

Conversion of CQF into DQF:

1. First to collect all the queries as a set which is required to generate the form as Q={Q1,Q2,Q3.....Qn}, where n>0.
2. Specify the set of items I in the list of collected queries as I={I1,I2,I3....Im}, where m>0.
3. Identify that every item of I list belongs to any one query in the list of Q where only one item is identify with only one query item. As Qe$\leftarrow$Ie
4. Set or placed item in one panel of a relation of form. Whereas it may be contain more than one relation.

For the example, Suppose CQF keeps Queries related to generate the Employee and Department relations, here user has to type the queries but in DQF it is implemented as in dropdown button, user has to only select the name of table and table is created with user defined attributes.

2. Query-By-Form interface: Every query form resembles to an SQL query template.

A query form F is defined as a Tuple (AF , RF , σF , ⋈ (R)), this signifies a database query template like in below:

F = (SELECT A1,A2, ...,Ak FROM ⋈ (RF) WHERE σF)

Where,

- AF = {A1,A2, ...,Ak} are k attributes of projection, k > 0.
- RF = {R1,R2, ..., Rn } which is collection of n relations n > 0.
- Every attribute in AF will belong to one relation in RF.
- σF = conjunction of expressions for selections on relations in RF.
- ▹◃(RF) is a natural join function to create a conjunction of expressions for joining relations of the RF .

W can say that RF is determined by AF & σF . RF is union group of relations which keeps at least 1 attribute of AF or σF. So in the determination of components of query form F are in actual determined by AF & σF .So, in the user interface, according this only AF & σF are visible to the user. After this we can focus on projection & Selection components of a query form.

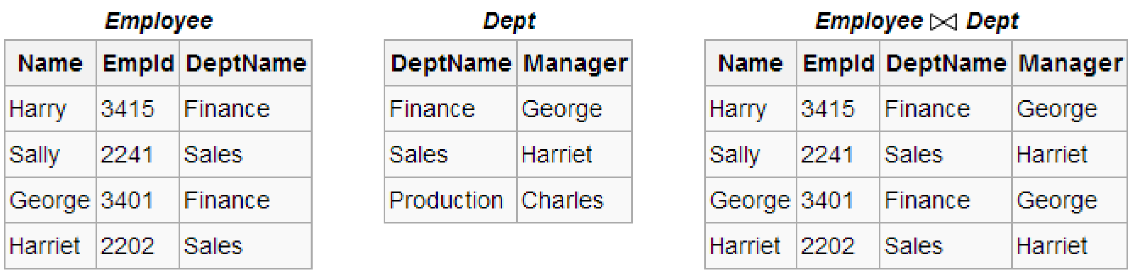
In the case of “Aggregation” & “Order by” in Structured Query Language provided limited choices for users. In which for Aggregation have MIN,MAX,AVG, and so on & for Order by can display result in increasing order Or decreasing order. In our dynamic query form we can easily expand the functionality by include those options by implementing them as dropdown boxes in user interface of query form.

For the example identified two tables name as “Employee” and “Dept”. Written the query to fetch all the attributes from both of the table where employee details and their respective departments are displayed:

SQL Query: Select Distinct * from ▹◃(Employee,Dept)where Employee.DeptName=Dept.DeptName

This▹◃ symbol for natural join which allows the combination of relations that are associated by a foreign key. Where primary key of one table works as a foreign key in another table. In the given example a Primary key in table Employee holds from Employee.DeptName to Dept.DeptName as a foreign key in table Dept & so the natural join of Employee & Dept will display all employees with their departments.

Output of above query:



Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

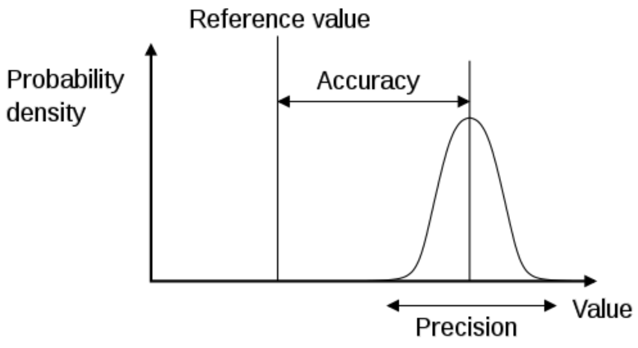
DeptName	Manager
Finance	George
Sales	Harriet
Production	Charles

Employee ⋈ Dept

Name	EmpId	DeptName	Manager
Harry	3415	Finance	George
Sally	2241	Sales	Harriet
George	3401	Finance	George
Harriet	2202	Sales	Harriet

3. F-measure/score:

F-score or F-measure is also known as F1 score is basically used to measure the accuracy. Here we use this method to test the accuracy of our result .It includes both the precision p & the recall r of the test to compute the score where p is the number of correct positive results which is divided by number of all positive results & r is the number of correct positive results which is divided by the number of positive results that should have been returned. So after getting p & r , calculate F1 score can be identified as a weighted average of the precision & recall where an F1 score has best value at 1 & worst score at 0. Figure 2



It can be understand by the using of cross multiplication of selected and non selected data items as:

	CORRECT	NOT CORRECT
SELECTED	TRUE POSSITIVE	FALSE POSSITIVE
NOT SELECTED	FALSE NEGATIVE	TRUE NEGATIVE

There may be four ways of being right or wrong:

1. **TN or True Negative:** case was negative & predicted negative
2. **TP or True Positive:** case was positive & predicted positive
3. **FN or False Negative:** case was positive but predicted negative
4. **FP or False Positive:** case was negative but predicted positive

Precision- For the information retrieval **precision** is the fraction of retrieved documents that are relevant to the searched keyword**:**

Precision= TP/ (TP+FP)

*Recall- **In the** information retrieval **system recall** is the fraction of the documents that ar**e relevant to the query that has been** successfully retrieved:*

Recall= TP/(TP+FN)

Accuracy:
In the area of science, engineering, industry, & statistics accuracy of measurement system is the degree of closeness of measurements of a quantity to that quantity's actual (true) value.

Acc= (TP+TN)/(TP+FP+FN+TN)

or

$$\text{accuracy} = \frac{\text{number of true positives} + \text{number of true negatives}}{\text{number of true positives} + \text{false positives} + \text{false negatives} + \text{true negatives}}$$

Calculating precision & recall is in practically very easy. Suppose there are 100 positive cases if we have total 10,000 cases. We want to predict which ones are positive & pick 200 to have a better chance of catching many of the 100 positive cases. We record the IDs of our predictions & when we get the actual results then sum up how many times we were right or wrong. Now let’s see how many of the 10,000 cases fall in every bucket:

	Predicted Negative	**Predicted Positive**
Negative Cases	TN: 9,760	FP: 140
Positive Cases	FN: 40	TP: 60

1. What percent of the positive cases did catch?
Answer: Recall= TP/ (TP+FN) = 60/(60+40)= 60/100=6/10
The "recall" was 60 out of 100 = 60%

2. What percent of positive predictions were correct?
Answer: Precision= TP/ (TP+FP) = 60/ (60+140) =60/200 = 3/10
The "precision" was 60 out of 200 = 30%

3. What percent of predictions were correct?
Answer: Acc= (TP+TN)/(TP+FP+FN+TN)= (60+9760)/(60+40+140+9760)
The "accuracy" was (9,760+60) out of 10,000 = 98.2%

Implementation Plan

With the help of collecting design consideration explained in the given section we can find a brief idea on the implementation part of our project. We use .Net Framework 4.0 will be used owing to the fact that it provides an easier way to create application that present, capture, manipulate & implement time solution. In .Net, C# will be used to reduce the code, making it simple as well as reduce the overall size of the application. The database which will be used is in SQL. Web forms will be used instead of components based code for dynamic accessing. We will implement the application as dynamic nature product, which will share the information between users over large number of network in real time.

IV. Conclusion

In this way we propose a dynamic query form generation approach which helps users dynamically generate query forms. The main idea behind it is to use a simplified model to rank form components based on user preferences. We capture user preference by the help of both historical queries & run-time feedback such as click through event of mouse.We includes the ranking procedure which improves the accuracy of information retrieval process. We also try to develop the model which satisfy the user completely in minimal retrieval time and get the good quality result with the help of user feedback session. With the inclusion of DDA and HTML parser system become more usable and efficient by display the result in different format with the expansion.

V. References

[1] http://htmlparser.sourceforge.net/
[2] http://www.ics.forth.gr/_publications/mdscan.eurosec11.pdf
[3] http://dl.acm.org/citation.cfm?id=1972555
[4] David M W (2007/2011). "Evaluation From Precision, Recall & F-Factor to ROC, Informedness, & Markedness & Correlation". Journal of Machine Learning Technologies 2 (1): 37–63.
[5] P. Peereman, R. (2004). "The exploitation of distributional information in syllable processing". J. Neurolinguistics 17: 97–119. doi:10.1016/s0911-6044(03)00059-9.
[6] David M. W. (2012). "Conference of the European Chapter of the Association for Computational Linguistics (EACL2012) Joint ROBUS-UNSUP Workshop". |chapter= ignored (help)

[7] Fawcelt, Tom (2006). "An Introduction to ROC Analysis". Pattern Recognition Letters 27 (8): 861 – 874.doi:10.1016/j.patrec.2005.10.010.
[8] Van Rijsbergen, C. J. (1979). *Information Retrieval* (2nd ed.). Butterworth.
[9] web2py.com/books/default/chapter/29/07/forms-and-validators
[10] http://www.tutorialspoint.com/jsp/jstl_xml_parse_tag.htm