ISSN 1870-4069

DW4SN: A Tool for Dynamic Data Warehouse Building from Social Network

Rania Yangui, Ahlem Nabli, Faiez Gargouri

Sfax University, Miracl Labortory, Tunisia

Abstract. Analyzing social networks data becomes increasingly important for many companies day-to-day operations. However, due to their scale, complexity and dynamics, these networks are difficult to be processed by means of traditional warehouse systems. In fact, the rapid growth of data with the frequent arrival of new needs requires that the system should be adaptable to changes. This work presents DW4SN (Data Warehouse for Social Network) tool for building data warehouse from social network, where clustering methods are used to define the DW schema and NoSQL systems are used to implement the warehouse. We validate our system on a real data set concerning crafts women social network. The main benefits were obtained in terms of scalability and dynamicity.

Keywords: Social networks, clustering, data mining.

1 Introduction

During the last decade the explosion of social media has lead to the generation of massive volumes of user-generated data that has given birth to a novel area of research, namely data warehousing for social network. One big challenge in this domain is that companies can effectively use published data. This puts emphasis on how classical data warehouse (DW) methodologies can be extended in order to deal with novel features of social network data, such as volume, dynamicity and heterogeneity. Developing a such warehouse is a complex and costly activity. It requires strategies, which should be specific to the social network characteristics and user's needs.

Recently, Online Social Networks started to be modeled with rich structured data that incorporate semantics using ontologies like Friend of a Friend (FOAF) to describe users, content and their relationships [1]. In the literature, few FOAF-based approaches for the design of DW systems from social network have been proposed with various degrees of automation [2] [3]. However, these approaches have shown some limitations. In fact, the rapid growth of data with the frequent arrival of new needs, which requires that the system should be adaptable to changes, is not considered. Moreover, these works are based on a relational approach which presents drawbacks concerning the DW scalability.

Thereby, in this paper we present a new prototype comosed of five modules for data warehouse building from social network. In particular, we detail three

main modules which enable the DW schema building via dynamic discovery of Multidimensional Concepts (MCs) using a semi-supervised hierarchical clustering and then implementing it under document-oriented NoSQL system. This system is attractive for developers due to its ability to handle large volumes of heterogeneous data. Thereafter, we validate our system on a real data set concerning crafts women SN.

The paper is organized in the following way. Section 2 describes the case study of the paper and some related works. Section 3 introduces the generic prototyping methodology. Section 4 describes a semi-automatic tool developed to support the proposed methodology. Section 5 evaluates the proposed system and discusses the results. Section 6 concludes the paper and gives some hints to future work.

2 Research Context and Motivations

In this section, we present an example from BWEC[1] (Business for women in emerging countries) project that aims to improve the social-economic situation of crafts women in emerging countries. These women are confronted with problems as lack of capital, unavailability of raw materials, lack of knowledge and skill of modern techniques and marketing problems.

The use of social networks could positively contribute to this population by getting to know each other and sharing their business experiences and problems. Indeed, SNs help handicraft women to share information, resources and equipments. By getting to know each other and sharing their business experiences and problems in public groups, women became aware of the need to reconcile efforts to strengthen their self-help initiatives and networks in the communities. Using data available on SNs let handicraft women increase knowledge they have about production process and commercialization strategies. Therefore, we need to analyze womens interests and changes in their relationships. In order to carry this out, it is necessary to accumulate womens profiles, interactions and the information about their activities in a data warehouse.

Nevertheless, as claimed by many works [9] [10], user generated data from social network are very difficult to store and analyze in terms of traditional data warehousing methodologies. Moreover, decision-makers of that project (crafts women) are unskilled DW users, and then they need DW prototypes to validate their analysis needs.

In the literature, many researchers have proposed approaches for DW building from SNs. Nevertheless, the majority of these works focused only on the intermediate logical design phase [6] [7] [8], while few others provided approaches including both conceptual and logical design levels [9] [10]. As the conceptual modeling is the core stone of a successful DW, special attention should be paid to it. In this context, the rapid growth of data with the frequent arrival of

[1] Towards a new Manner to use Affordable Technologies and Social Networks to Improve Business for Women in Emerging Countries http://projetat.cerist.dz/artisanat/

new needs requires that the system should be adaptable to changes. The data warehouse designer, with his/her limited knowledge of the domain can bring the some multidimensional concepts, but in other cases, such process needs to be automated. Clustering techniques can help in designing dynamic DW schema.

Moreover, as NoSQL data stores are becoming increasingly popular, some works have shown that it is possible to convert a multidimensional conceptual model into NoSQL storage [11,12,6,7,8]. These systems are attractive for developers due to their ability to handle large volumes of data, as well as data with a high degree of structural variety. However, these approaches have focused only on the rules for transforming the concepts of fact and dimension in a NoSQL structure. Furthermore, it appears that the majority of researchers use HBase for implementing a NoSQL DW. This is justified by the resemblance between the logic model of HBase and that of relational databases, particularly in terms of concepts of tables and rows.

Based on the above discussion, there is a strong need for a significant prototype that allows a semi-automatic building of dynamic DW from SN. Semi-Automating data warehouse conceptual and logical design can provide real value to DW development projects, and increase their chance of success while reducing cost, risk and manual effort.

3 Building Data Warehouse from Semantic Social Network: Generic Prototype

Given the sheer volume of social network data normally involved in the building of a data warehouse, a case can be made for semi-automated support in the construction of a dynamic warehouse. We build a prototype that can load a company's data warehouse and a FOAF ontology, design new multidimensional concepts in the data warehouse schema and implement the obtained warehouse under document-oriented NoSQL system. The prototype uses a clustering method for the dynamic discovery of MCs, and uses transformation rules for the mapping to NoSQL data base. The prototype is composed of five modules as depicted in Fig.1.

1. Data warehouse schema crawler: the system recovers and displays the multidimensional concepts from the company's data warehouse schema. This warehouse is created following a classical mixed approach [13].
2. FOAF Generator: The system loads the FOAF ontologies from livejournal social network and merges them.
3. Clustering Performer: the system computes the similarity measure and performs the hierarchical clustering.
4. Multidimensional Concepts Determiner: According to the result of clustering and guided by the designer, the system determines new MCs. Then the system updates the le that describes the DW schema by adding new MCs.
5. NoSQL mapper: The system applies a set of transformation rules at ETL (Extract Transform Load) lavel to implement the NoSQL DW.

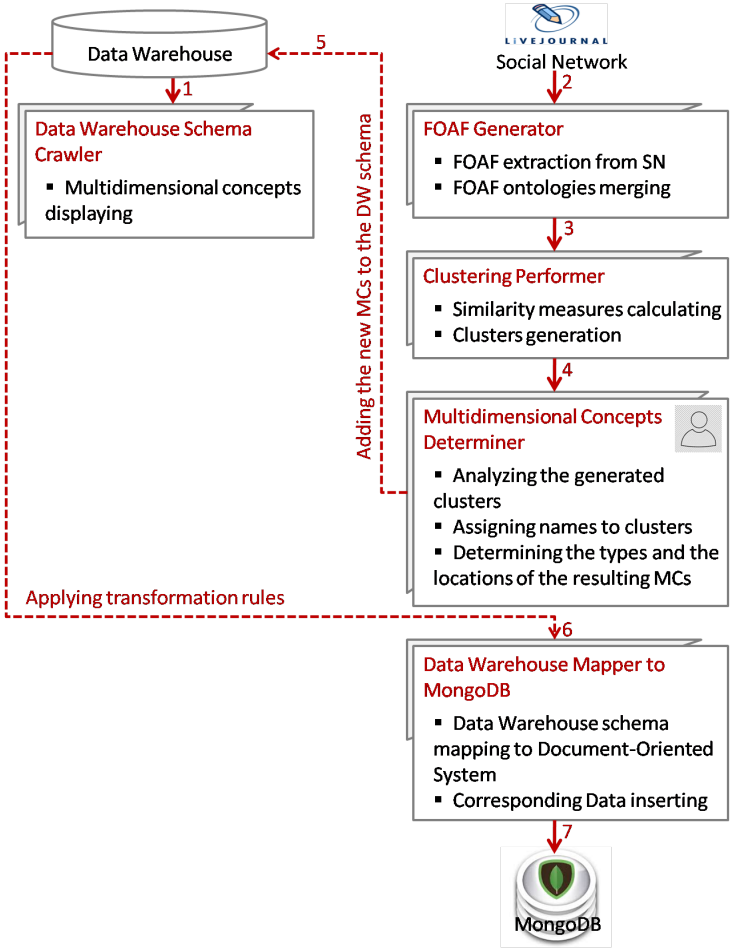

Fig. 1. Prototype woring.

Our prototype allows designer to easily define and validate their DW in an incremental way. In the next subsection, we will detail the three last modules.

3.1 Clustering Performer

In the implementation of our prototype we have chosen to use SHICARO [14] (Semi-supervised Hierarchical Clustering based on Ranking features using Ontology) method that aims to cluster objects based on scheduled features in order to get the optimal results that meet the designer needs.

Since the expert knows the goal behind which the clustering is performed, SHICARO method performs clustering under the designer guidance. Thus, the designer is required to order the features from the highest to the lowest ones.

As input to this method, we have a set of FOAF instances and a set of scheduled features. At each iteration, a set of features $F = \{(f_1, r_1), \ldots, (f_n, r_n)\}$ that have the same rank r_i is applied to cluster objects. The number of iterations is equal to the maximum of the ranks. The N number of clusters to be generated is expressed in terms of the number of instances and of iterations:

$$N = Round\left(InstancesNumber/2^{Max(r_i)}\right).$$

Steps followed during implementation of SHICARO method are described by Algorithm 1.

Data: FOAF ontology, $F = \{(f_i, r_i)\}$
Result: $C = \{c_k\}$
initialization: Each instance is placed in its own cluster c_k ;
repeat
1. Find min r_i ;
2. Select the current future set F_c having min rang ;
3. Calculate p the number of clusters to generate at each iteration ;
4. Compute similarities based on the current features set F_c ;
5. Merge c_k to p clusters ;
6. Update C and F ;

until $F = \emptyset$;
Return C ;

Algorithm 1: SHICARO Algorithm.

This process is repeated for each generated cluster (step 1) based on the next current feature set until the number of features reaches 0. The extracted clusters become the input for the dynamic schema upgrading in the next subsection.

3.2 Multidimensional Concepts Determiner

This module is performed by the designer and consists in analyzing the generated clusters, determining the type of multidimensional concept, assigning names to clusters and specifying the location of insertion in the multidimensional schema. The multidimensional concepts determination module is described by Algorithm 2.

As described by Algorithm 2, fives cases are possible: adding new fact, adding new measure to an existing fact, adding new dimension, adding new week attribute or adding new parameter to an existing or a new hierarchy.

Data: A set of clusters $C = \{c_i\}$, Initial DW Schema $MS_{DW} = (F, D)$ where $F = \{f_1, \ldots, f_n\}$ and $D = \{d_1, \ldots, d_k\}$
Result: discovered MCs integrated in the MS_{DW}
forall the *cluster* c_i *in* C **do**
1. Display the cluster c_i to the user ;
2. The user input a name N_{c_i} to c_i ;
3. Identify a type T_{c_i} of c_i ;
4. **switch** *the value of* T_{c_i} **do**
   - **case** *"Fact"*
     - Assign N_{c_i} to the fact f ;
     - Add f to F ;
     - Display all dimensions D from MS_{DW} ;
     - Select the dimensions $d \in D$ related to f ;
     - Associate d to f in MS_{DW} ;
   - **case** *"Dimension"*
     - Assign N_{c_i} to the dimension d ;
     - Add d to D ;
     - Display all fact F from MS_{DW} ;
     - Choose the facts $f \in F$ to which d is related ;
     - Associate d to f in MS_{DW} ;
   - **case** *"Measure"*
     - Assign N_{c_i} to the measure m ;
     - Display all fact F from MS_{DW} ;
     - Choose the fact $f \in F$ to which m will be added ;
     - Add m to f ;
   - **case** *"Week Attribute"*
     - Assign N_{c_i} to the week attribute wa ;
     - Display the set of dimension D from MS_{DW} ;
     - Select the dimension d to which wa will be added ;
     - Display all parameters of d ;
     - Select the parameter p to which wa will be added ;
     - Add wa to p ;
   - **otherwise**
     - Assign N_{c_i} to the parameter p ;
     - Display the set of dimension D from MS_{DW} ;
     - Select the dimension d to which p will be added ;
     - Display all the hierarchies of d ;
     - **if** *p will be added to a new hierarchy h* **then**
       - Add h to d ;
       - Add p to h ;
     - **else**
       - Choose the hierarchy he to which p will be added ;
       - Choose the level l at which p will be added ;
       - Add p to he at level l ;
     - **end**
   - **end**
4. **endsw**

end

Algorithm 2: MCs Definition.

3.3 NoSQL Mapper

Given that a well-designed DW requires a well planned logical design, all updates and versions of a DW lead to a revision of the logical design. Generally, the mapping from the conceptual to the logical model is made according to three approaches: ROLAP (Relational-OLAP), MOLAP (Multidimensional-OLAP) and HOLAP (Hybrid-OLAP) [15]. However, these systems suffer from scaling-up to large data volume [7]. As an alternative, NoSQL systems begin to grow.

NoSQL is a dynamic and cloud friendly approach to dynamically process social network generated data. NoSQL Database, also known as Not Only SQL is an alternative to SQL database which does not require any kind of fixed schemas. NoSQL generally scales horizontally and avoids major join operations on the data.

In the literature, four types of NoSQL data stores exist [16]: Key-value Stores, Document Stores, Columnar Stores and Graph Stores. In the absence of a clear approach which allows the implementation of data warehouses under NoSQL model, we have compared in previous work [17], two NoSQL systems: columns-oriented and documents-oriented with two types of transformation: simple and hierarchical. The results showed that the documents-oriented NoSQL data warehouse with hierarchical transformation is more efficient in terms of interrogation. This is justified by the fact that data in the column-oriented systems are not available in the same place.

The hierarchical transformation to document-oriented system uses different collections for storing facts and dimensions, and uses the simple documents for representing measure and the composed attributes for representing dimension attributes while explaining hierarchies.

Rule 1 represents the transformation of a fact and its measures to the document-oriented model.

Rule 1: Each fact $F \in MS^{Fact}$ is transformed to a document DC (DC^N, DC^{Att}) where :

- **The name of the document is the name of the fact** $/DC^N \leftarrow F^N$**;**
- **Each measure** $M \in F$ **is transformed to a simple attribute** $SA \in DC^{att}/SA^N \leftarrow M$**;**
- **Each identifier of a related dimensions is transformed to a simple attribute** $SA \in DC^{att}/SA^N \leftarrow D^{id}$**;**

Moreover, the hierarchical transformation of a dimension and its attributes (Strong and Weak) to the document-oriented model is mentioned by Rule 2.

Rule 2: Each dimension $D(D^N, D^{Att}, D^{Hier})/D \in MS^{Dim}$ is transformed to a document $DC(DC^N, DC^{Att})$ where:

- **The name of the document D is equivalent to the name of the dimension $/DC^N \leftarrow D^N$;**
- **Each hierarchy H(H^N, H^P, P^{FAtt}) is transformed to a composed attribute $CA \in DC^{att}$ where :**
  - **The name of the composed attribute is the hierarchy name $/CA^N \leftarrow H^N$**
  - **The values of composed attributes are the simple attributes that represent the weak and the strong attributes.**

Based on the above defined rules (Rule 1 and Rule 2), we deduce the hierarchical transformation of a multidimensional schema (MS) as mentioned by Rule 3.

Rule 3: Each multidimensional schema MS (MS^N, MS^{Fact}, MS^{Dim}, $Func$) is transformed to a documents collection $DCC(DCC^N, DCC^{Val})$ where:

- **The name of the collection is the name of the MS $/DCC^N \leftarrow MS^N$;**
- **Each fact $F \in MS^{Fact}$ is transformed to a document $DC(DC^N, DC^{Att})$ (Rule 1);**
- **Each dimension $D(D^N; D^{Att}; D^{Hier}) \in MS^{Dim}$ is transformed to a document $DC(DC^N, DC^{Att})$ (Rule 2).**

4 DW4SN Tool

In this section, we present the propose system DW4SN(Data Warehouse for Social Network). This system is composed of two main operational phases (Fig.2).

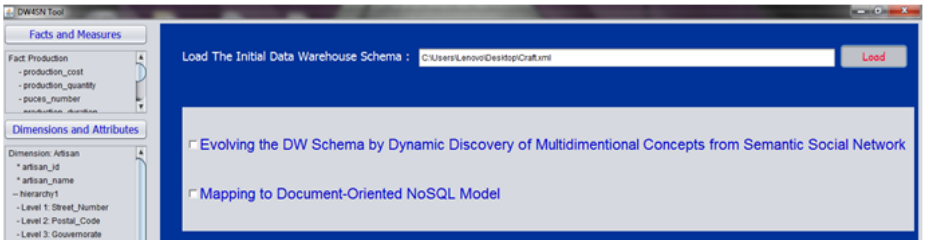

Fig. 2. DW4SN operational phases.

The first one is the Dynamic Discovery of MCs during which DW4SN enriches the warehouse schema using hierarchical clustering with ranking features. The second phase of DW4SN is the Migration to Document-Oriented NoSQL system,

during which the system apply a set of rules that create the warehouse under MongoDB.

This work is directed by means of JAVA programming language. To access social network pages, we extracted the FOAF ontology for CraftsWomen group in livejournal. This group is created by the project community to gather crafts women from Tunisia and Algeria. For each instance of foaf: Person in the generated FOAF ontology, we extracted the corresponding personal FOAF. Then, we used the PROMPT plug-in integrated in Protg2000 Tool to merge the extracted FOAF files. We obtained an ontology with hundreds of instances.

To perform the clustering on FOAF instances, we used the Weka API integrated with java. At each iteration, we used the hierarchical clustering. The output of each iteration becomes the input of the next one. The number of clusters at each iteration is calculated in terms of the number of instances and the max of rank in the feature set. The user is required to load the initial DW, and then he is required to load the FOAF ontology and select the object on which he wants to perform the clustering. After that, the user is required to enter an ordered feature set that meets his need and then apply the clustering button. An interface that contains the clustering results is so displayed.

Fig.3 shows the Clustering configuration and results in our tool.

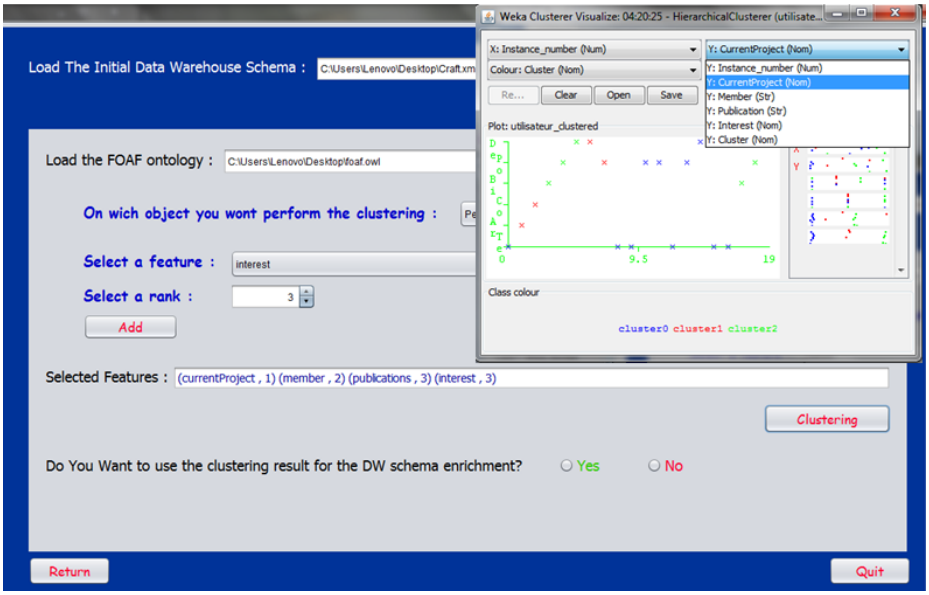

Fig. 3. Clustering configuration and results.

Thereafter, the system asks the user if he wants to use the clustering results for the DW schema enrichment. If the response of the user is positive, a new

interface is displayed. Then, the system asks the user to appoint the new MC and specify its type and its position in the DW schema. Many cases are possible:

- Adding a new fact. In this case the designer should select the related dimensions. In fact, due to the dynamicity of SN, the analysis axes may change and increase.
- Adding a new measure to an existing fact.
- Adding a new dimension. In this case, the user must select the fact to which the dimension will be related.
- Adding a new week attribute. In this case, the designer must select the concerned dimension.
- Adding a new parameter. In this case, the user should select the desired dimension and specify if the parameter will be added to an existing or a new hierarchy. If it is a new hierarchy, the system creates a hierarchy and inserts the parameter at the first level. Else (if it is an existing hierarchy), the designer should select the level at which he wants to add the parameter.

Fig.4 represents examples of adding a new MCs to the DW schema.

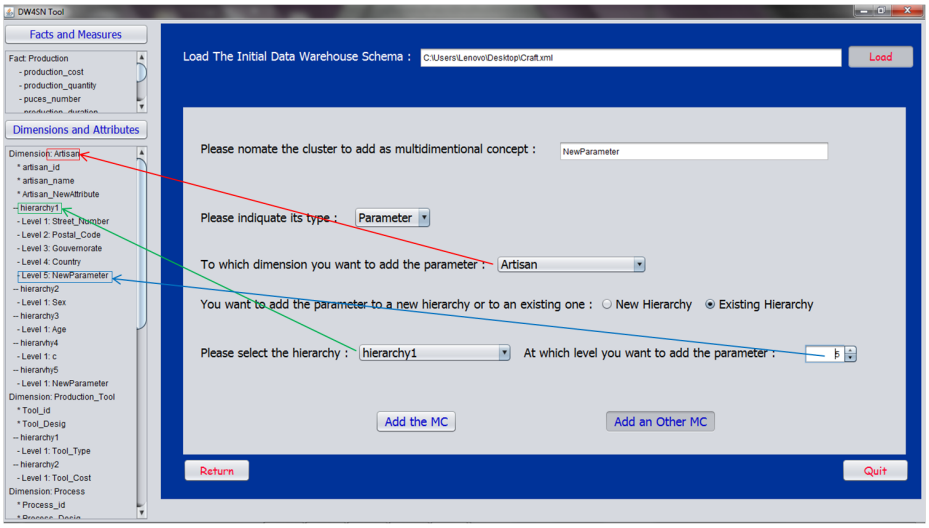

Fig. 4. Adding new MCs.

The screenshot shows an example of adding a new parameter to an existing hierarchy (hierarchy 1) at the level 5 in the "Artisan" dimension.

To implement the DW under MongoDB, we used the java routines integrated with the data integration tool Talend for Big Data. In fact, the schema-less nature of the document-oriented data base means that we can store documents in any shape but the notion of schema itself doesnt disappear from the model.

Therefore, to create a NoSQL DW, we are required to write a program that relies on some form of implicit schema (Fig.5).

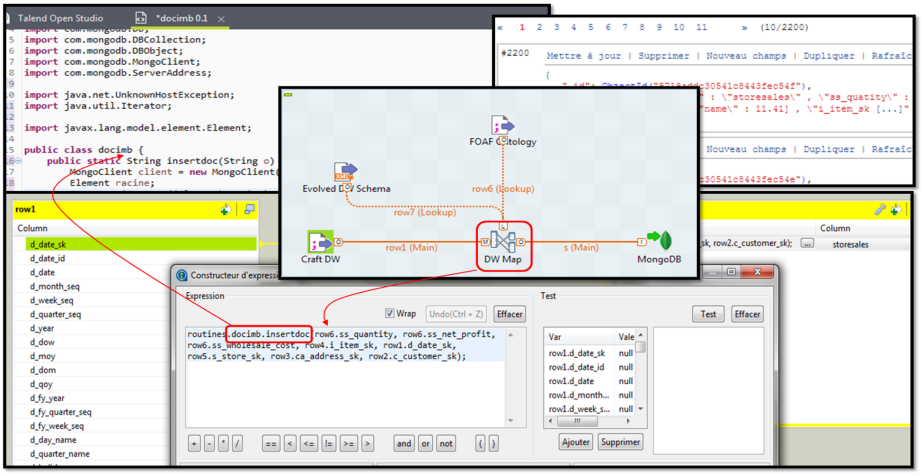

Fig. 5. Example of ETL routines.

Fig.5 shows four main interfaces: the created Job, an excerpt of the implemented java routine, the expression editor, and an excerpt of the resulting MongoDB data base.

A Job is a graphical design, of one or more components connected together such as tFileInputDelimited (Craft DW, FOAF Ontology, Evolved DW Schema), tMap (DWMap) and FileOutputDelimited (MongoDB). Otherwise, a routine is a complex Java code, generally used to optimize data processing and improve Job capacities. In this work, the routine is used for implementing the transformation rules. The implemented routine is then called and edited in the expression editor. This editor provides visual interface to write any function or transformation in a handy dedicated view.

5 Discussion

In this paper, we proposed a novel tool for building data warehouse from social network. The focus of this work is to overcome the existing limitation identified in the literature and to accomplish the ever growing requirement of modern analytical systems. In this part, we evaluate the proposed system and discuss the results. For that, we applied it in a real case study: BWEC project. In this project our task is to build a data warehouse from craft women social network.

One of the important aspects of the proposed prototype is the dynamic management of data warehouse schema. To get the optimal results that meet the

designer needs, we proposed, an algorithm for semi-supervised hierarchical clustering based on scheduling features using FOAF ontology. Then, we proposed an algorithm for multidimensional concepts determination based on the generated clusters. For the sake of discussion, we compared the grouping among the simple hierarchical clustering (HC) and SHICARO method for 11 month in 2016. We compared the F-score measures per month.

Fig.6 shows the F-Score results per month for HC and SHICARO algorithms.

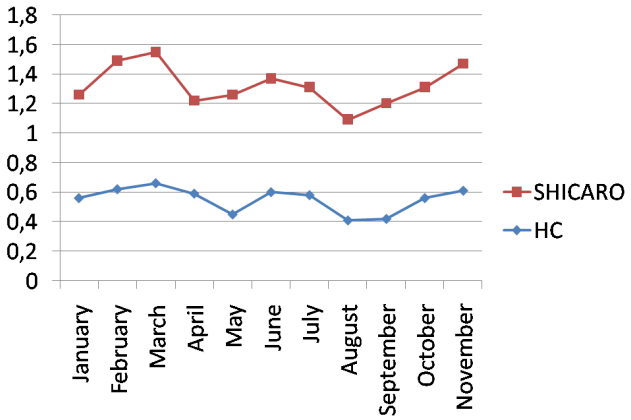

Fig. 6. F-Score results per month for HC and SHICARO algorithms.

It has been observed that SHICARO algorithm is the best (the average of F-Score is using SHICARO is while 11 for HC) . The proposed system provides a high degree of automation that enables designers to quickly and easily designing and building conceptual DW schema and to reduce the deployment costs.

Moreover, as NoSQL data stores are becoming increasingly popular in application development, we proposed transformation rules to map conceptual schema to document-oriented model. Document-oriented DB is attractive for developers due to their ability to handle large volumes of data with a high degree of structural variety. Typically, NoSQL data stores are accessed programmatically since they are schema-less. Thus, the implementation of the transformation rules is performed at the ETL level while integrating data.

In order to evaluate the implemented NoSQL DWs, we choose to use Read Request Latency (RRL) metric. This metric has the purpose to evaluate the systems ability to respond quickly to user requests. We built a sample data warehouse which was stored in both MySQL and MongoDB in order to be compared for disk space usage and Read Request Latency (Fig. 7).

Regarding requests, we choose to use four queries classified into two categories. The first category consists on increasing the number of dimensions and attributes to test the performance of the decision system with the presence of

joins in the users queries. As for the second category, it is more complex and consists on using some operators.

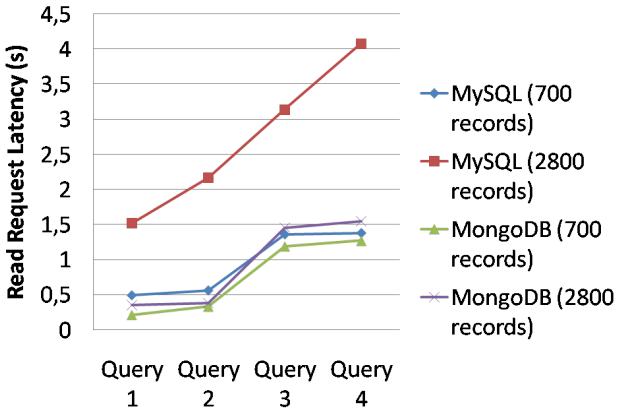

Fig. 7. Read Request Latency Values.

We found that the DW implemented under MySQL consumed less storage space. This is justified by the fact that MongoDB allows every record to have a completely different structure than every other. It stores the schema of every record with the record itself. However, DW implemented under SQL is lack of scaling and inefficient of handling big data.

We deducted that, the NoSQL data warehouse can respond to increasing queries without performance degradation (0.21 for Q1 and 1.27 for Q4). Moreover, the NoSQL data warehouse can rapidly adapt to growth in data volume and query intensity without degrading performance (0.21 for 700 records while 0.35 for 2800 records using the same query). The NoSQL data warehouse can quickly adapt to changes in data structure and content without requiring any schema redesign, additional data migration, or new data storage structures.

6 Conclusion

In this work, we have presented a prototyping methodology and the associated tool, to build a data warehouse from social network. To do that, we proposed a method to build DW schema via content-based discovery of facts, dimensions, hierarchies and measures using hierarchical clustering. This method uses a Semi-supervised Hierarchical Clustering based on ranking features using ontology.

The second contribution of this work is to propose rules for transforming a multidimensional conceptual schema into NoSQL system. As NoSQL databases are schema-less, the creation of a schema occurs while inserting data at ETL

level. In this level, we implemented the defined transformation rules as routines that reflect the implicit schema.

The overall proposed methodology meets the majority of the posed challenges. In fact, NoSQL allows an effective modeling of the massive data volumes generated by the SN. Moreover, the dynamic discovery of MC using semi-supervised hierarchical clustering handles the rapid growth of the mass of information. As perspectives, we aim implementing OLAP operators (D-Roll up, D-Slice, D-Dice) with the phases of the invisible join technique for a document-oriented data warehouse.

References

1. Chelmis, C., Wu H., Sorathia, V., Prasanna V.K.: Semantic Social Network Analysis for the Enterprise. In: Computing and Informatics, pp. 479–502 (2014)
2. Sohn, J.S., In-Jeong, C.: Dynamic FOAF management method for social networks in the social web environment. The journal of supercomputing, pp. 1–17 (2013)
3. Wojciech, K.: Facebook crawler as software agent for business intelligence system. Studia informatica, pp. 89–110 (2014)
4. U. Rehman, N., Mansmann, S. Scholl, M. H.: Discovering dynamic classification hierarchies in OLAP dimensions. In: Proceedings of the 20th international conference on Foundations of Intelligent Systems, pp. 425-434 (2012)
5. Gallinucci, E., Golfarelli, M. Rizzi, S.: Meta-Stars: Dynamic, Schemaless, and Semantically-Rich Topic Hierarchies in Social BI. In: Proceeding of the 18th International Conference on Extending Database Technology, pp. 529–532 (2015)
6. Dehdouh, K., Boussaid, O., Bentayeb, F.: Using the column oriented NoSQL model for implementing big data warehouses. In: Proceedings of the 21st International Conference on Parallel and Distributed Processing Techniques and Applications, pp. 469–475 (2015)
7. Chevalier, M., El Malki, M., Kopliku, A., Teste, O., Tournier, R.: Implementing Multidimensional Data Warehouses into NoSQL. In: Proceedings of the 17th International Conference on Enterprise Information Systems, pp. 172–183 (2015)
8. Santos, M.Y., Martinho, B. Costa, C.: Modelling and implementing big data warehouses for decision support. Journal of Management Analytics, pp. 1–19 (2017)
9. U. Rehman, N., Mansmann, S. Scholl, M. H.: Discovering dynamic classification hierarchies in OLAP dimensions. In: Proceedings of the 20th international conference on Foundations of Intelligent Systems, pp. 425-434 (2012)
10. Gallinucci, E., Golfarelli, M. Rizzi, S.: Meta-Stars: Dynamic, Schemaless, and Semantically-Rich Topic Hierarchies in Social BI. In: Proceeding of the 18th International Conference on Extending Database Technology, pp. 529–532 (2015)
11. Bringay, S., Laurent, A., Poncelet, P., Roche, M., Teisseire, M.: Towards an On-Line Analysis of Tweets Processing. In: Proceedings of the 22nd International Conference on Database and Expert Systems Applications, pp. 154–161 (2011)
12. Dede, E., Govindaraju, M., Gunter, D., Canon, R., Ramakrishnan L.: Performance evaluation of a MongoDB and hadoop platform for scientific data analysis. In: Proceedings of the 4th ACM workshop on Scientific cloud computing, pp. 13–20 (2013)
13. Ahlem, N., Senda B., Rania Y., Faiyez G.: Two-ETL Phases for Data Warehouse Creation: Design and Implementation. In: Proceedings of the 19th East European

Conference in Advances in Databases and Information Systems, Poitiers, France, pp. 138–150 (2015)
14. Yangui, R., Nabli, A., Gargouri, F.: Semi-supervised Hierarchical Clustering bAsed on Ranking features using Ontology. In: The 2d International Conference on Management and Technology in Knowledge, Service, Tourism and Hospitality, pp.225-231 (2014)
15. Chaudhuri, S., Dayal, U., Ganti, V.: Database technology for decision support systems. IEEE Computer Society, pp. 48–55 (2002)
16. Sharma, V.: SQL and NoSQL Databases. International Journal of Advanced Research in Computer Science and Software Engineering, pp. 20–27 (2012)
17. Yangui, R., Nabli, A., Gargouri, F.:Automatic Transformation of Data Warehouse Schema to NoSQL Data Base: Comparative Study. In: Knowledge-Based and Intelligent Information and Engineering Systems, Proceedings of the 20th International Conference, pp. 255-264 (2016)