Mining Association Rules from Semantic web data using Spark framework

Chaitali Prabhu[1], Hemanth Kumar Reddy Mayaluru[2] and Manoj Prabhakar[3]

[1] Informatik III, Universität Bonn, Germany
s6chprab@uni-bonn.de
[2] Informatik III, Universität Bonn, Germany
s6hemaya@uni-bonn.de
[3] Informatik III, Universität Bonn, Germany
s6makann@uni-bonn.de

Abstract

This report is done as part of the Distributed Big Data Analytics Lab during Summer Semester'2018 at University of Bonn. The main aim of the project is, mining the association rules from the semantic data represented in RDF(s) and OWL, which is implemented in Scala using the Spark framework.

1 Problem Definition

With the standardization of ontology languages, the amount of ontologies and semantic annotations available on the Web is constantly growing. There are new challenges arising in data mining due to the complex and heterogeneous graph-structured data. The Data Mining models, represent relations in the data and are used for classifying new data or for describing hidden correlations in the data[3]. There are many traditional algorithms successfully applied on large homogeneous data sets composed of transactions to extract association rules. However, semantic data is different in it's nature from the other data sets. Here we are dealing with clinical data sets which is complex and in large volumes with different structure and semantics.

When we handle semantic data, identifying the interesting transactions from semi-structured data becomes a big challenge. So, to mine the transactions easily it is important to use the knowledge from ontologies. Ontologies describe the concepts in the domain, as well as the relationships holding between them.

In this project, to solve the above mentioned problem of generating transactions and mining association rules, we have implemented a method proposed in the research paper[1] which efficiently extract items and transactions that will suit the traditional association rule mining algorithms[2]. As we deal with clinical scenarios as stated in the research paper the data set will be of huge volumes. So we consider this a Big Data problem and use the Spark framework[6] for handling this scenario.

2 Approach

The basic approach used in solving the problem is inspired by the research paper[1]. It is divided into 3 stages: preprocessing, main algorithm and association rule mining algorithm.
We have the data file(rdf) and corresponding ontology file(owl) as input. Also, we take the query as an input from the user. We would then generate the instance transactions and composition triples from these inputs. This would be the preprocessing required for the inputs to the main algorithm. The main algorithm is to generate item transactions and is taken as given in the

paper. These item transactions are then passed to the association rule mining algorithm to obtain the association rules. The proposed architecture is shown in Figure[1].

2.1 Data Structures [6] [7]

RDD : to store the input rdf and owl files
Graph : to visualize the input triples as graph
Collections : to parallellize the algorithm
MLLib : using the API for association rule mining algorithm

2.2 Data set

As the data set used in the paper is unavailable, the sample data used in the paper to demonstrate the algorithm is used as the data set in our project. It is based on clinical reports. We have created the data set using Protege[8] and checked for consistency with ontology. Then we have modified the triples as a readable input format to convert into RDD. We have created multiple instances of the input file to check the scalability of the algorithm for different sized inputs.

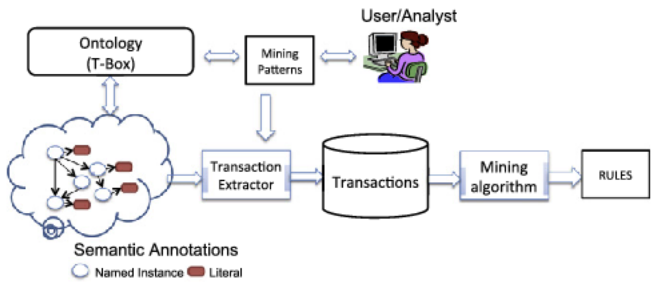


Figure 1: The Proposed Architecture

3 Implementation

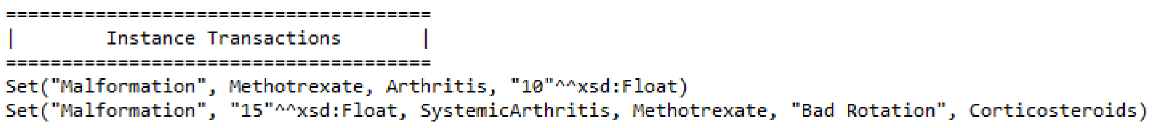
The implementation is inline with the approach mentioned above.

3.1 Major Functions

- Stage 1 : **Preprocessing**

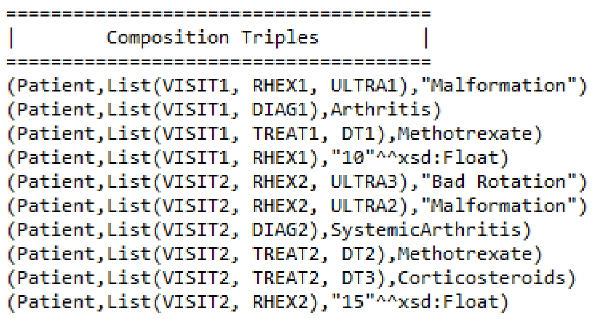
  The input rdf file, which is read as triples (or quadruples) serves as the instance store for the algorithm. The input rdf file is visualized as a map. This file is further processed to retrieve the subjects and objects and is stored into an RDD for vertices. The predicates are stored into an RDD for edges. Using these RDDs, we create a graph using *Spark.Graph*[6].

  The next major function is to create instance transactions and composition triples. This function takes the query and graph created from the rdf file as inputs and generates the instance transactions and composition triples. In this function, we extract the target concept (here: Patient) and the features (here: Disease,Drug,damageIndex) given in the input query. Every path from the target concept to each of the features in a composition triple, with the intermediate path taken as composition path. For a given target concept, based on the context(here: Report or Visit), we create instance transactions by taking all features common to a context.



```
======================================
|        Instance Transactions       |
======================================
Set("Malformation", Methotrexate, Arthritis, "10"^^xsd:Float)
Set("Malformation", "15"^^xsd:Float, SystemicArthritis, Methotrexate, "Bad Rotation", Corticosteroids)
```

Figure 2: Instance Transactions

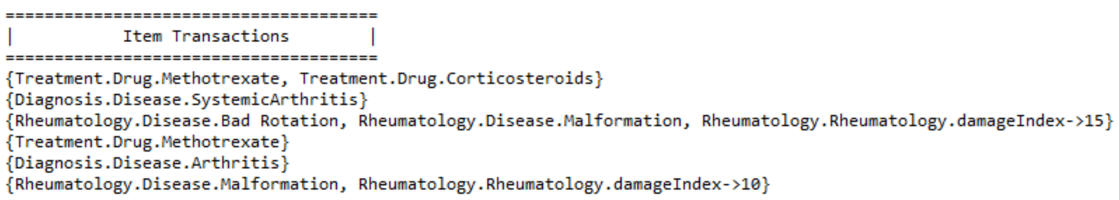


```
======================================
|        Composition Triples         |
======================================
(Patient,List(VISIT1, RHEX1, ULTRA1),"Malformation")
(Patient,List(VISIT1, DIAG1),Arthritis)
(Patient,List(VISIT1, TREAT1, DT1),Methotrexate)
(Patient,List(VISIT1, RHEX1),"10"^^xsd:Float)
(Patient,List(VISIT2, RHEX2, ULTRA3),"Bad Rotation")
(Patient,List(VISIT2, RHEX2, ULTRA2),"Malformation")
(Patient,List(VISIT2, DIAG2),SystemicArthritis)
(Patient,List(VISIT2, TREAT2, DT2),Methotrexate)
(Patient,List(VISIT2, TREAT2, DT3),Corticosteroids)
(Patient,List(VISIT2, RHEX2),"15"^^xsd:Float)
```

Figure 3: Composition Triples

- Stage 2 : **Main Algorithm**

  The main algorithm takes the composition triples from the preprocessing stage and generates item transactions required for mining rules. The inputs for the algorithm include the composition triples and context from the query. The algorithm checks those instances that have a data type property in the feature set with the corresponding values asserted in the Instance Store. A transaction is composed of a set of items co-occurring within instances belonging to the context. The items are generated from instances belonging to features and the user specified context. In the algorithm, we parse through the triples based on the context(Visit or Report). The most specific concept is taken from the feature maps of the function updateFeatures(). The function itemTransactions() parses through the composition triples and generates the item transactions for a particular context. These are given as input for the association rule mining algorithm.

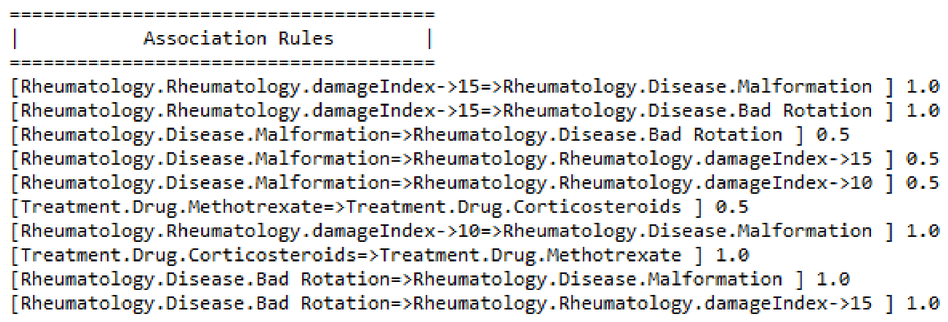


```
======================================
|          Item Transactions         |
======================================
{Treatment.Drug.Methotrexate, Treatment.Drug.Corticosteroids}
{Diagnosis.Disease.SystemicArthritis}
{Rheumatology.Disease.Bad Rotation, Rheumatology.Disease.Malformation, Rheumatology.Rheumatology.damageIndex->15}
{Treatment.Drug.Methotrexate}
{Diagnosis.Disease.Arthritis}
{Rheumatology.Disease.Malformation, Rheumatology.Rheumatology.damageIndex->10}
```

Figure 4: Item Transactions

- Stage 3 : **Association rule mining Algorithm**

  The item transactions generated from the previous stage is passed to the association_rules() function to generate frequent items and then these are given as input to the association rule mining algorithm which will generate the rules based on the minimum confidence provided. The Spark API, AssociationRules()[6] is used for generating the rules.



```
======================================
|          Association Rules         |
======================================
[Rheumatology.Rheumatology.damageIndex->15=>Rheumatology.Disease.Malformation ] 1.0
[Rheumatology.Rheumatology.damageIndex->15=>Rheumatology.Disease.Bad Rotation ] 1.0
[Rheumatology.Disease.Malformation=>Rheumatology.Disease.Bad Rotation ] 0.5
[Rheumatology.Disease.Malformation=>Rheumatology.Rheumatology.damageIndex->15 ] 0.5
[Rheumatology.Disease.Malformation=>Rheumatology.Rheumatology.damageIndex->10 ] 0.5
[Treatment.Drug.Methotrexate=>Treatment.Drug.Corticosteroids ] 0.5
[Rheumatology.Rheumatology.damageIndex->10=>Rheumatology.Disease.Malformation ] 1.0
[Treatment.Drug.Corticosteroids=>Treatment.Drug.Methotrexate ] 1.0
[Rheumatology.Disease.Bad Rotation=>Rheumatology.Disease.Malformation ] 1.0
[Rheumatology.Disease.Bad Rotation=>Rheumatology.Rheumatology.damageIndex->15 ] 1.0
```

Figure 5: Association Rules

4 Evaluation

All the experiments are performed on a single system with configuration of Intel Core i3 (8th gen) processor and 8 GB RAM. We have created a clinical data set based on the sample data given in the paper. The focus here is not on the association rules algorithm. The evaluation is based on the generated item transactions for a given query. Experiments were carried out on different queries, i.e. by using different contexts and features to generate item transactions.
For example,
Query Pattern : {Patient, Report,Disease,Drug,damageIndex}
Outputs : Shown in the listed figures -
Instance Transactions[2], Composition Triples[3], Item Transactions[4], Association Rules[5]

5 Installation and Usage Instructions

- Import the code from git hub into Scala IDE for Eclipse as a Maven project.
- Copy the *"hadoop-common-2.2.0-bin-master"* folder into a local directory and update the path in the following command present in the main.scala file.
  System.setProperty("hadoop.home.dir", "/Update Location here/");
- Pass the path to the input rdf file as argument.
  –input src/main/resources/InputFile.nt
- Perform the following steps,
  Right click on the project – Maven – Update Project. Then, *Run as – Scala Application*

6 Project Timelines

Week	**Task**	**Person Involved**
Week 1 (May 15 to May 22)	Paper Reading and Understanding	Manoj, Chaitali, Hemanth
Week 2 (May 22 to May 29)	Presentation 1 preparation	Manoj, Chaitali, Hemanth
Week 3 (May 29 to June 5)	Data set search	Manoj, Chaitali, Hemanth
Week 4 (June 5 to June 12)	Data set search	Manoj, Chaitali, Hemanth
Week 5 (June 12 to June 19)	Project planning and task division	Manoj, Chaitali, Hemanth
Week 6 (June 19 to June 26)	Data set Creation	Manoj, Chaitali, Hemanth
Week 7 (June 26 to July 03)	Preprocessing	Chaitali
Week 8,9,10 (July 03 to July 24)	Preprocessing (with CI)	Chaitali
Week 8,9,10 (July 03 to July 24)	Rule Generation, Preprocessing (with CI)	Manoj
Week 8,9,10 (July 03 to July 24)	Main Algorithm (with CI)	Hemanth
Week 11 (July 24 to July 31)	Combining the modules together	Manoj, Chaitali, Hemanth
Week 12 (August 07 to August 14)	Report and Presentation	Manoj, Chaitali, Hemanth
Week 13 (August 14 to August 21)	Report and Presentation	Manoj, Chaitali, Hemanth

[CI : Code Integration]

Table 1: Project Timeline

7 Future Work

There is future scope in various aspects. One major task would be to generalize the query pattern by using ontology axioms and also to automatically discover interesting contexts and their corresponding association rules. Another important task would be to check the consistency of the ontology with the input for different data sets. As an extension to this, extraction of the Most Specific Concepts (MSC) also needs to be dynamic. A new direction would be to combine clustering and association mining algorithms to summarize document collection(Frequent Itemset based Hierarchical Clustering - FIHC) as mentioned in the research paper[1].

References

[1] Victoria Nebot, Rafael Berlanga. *Mining Association Rules from Semantic Web Data*. Elsevier, 2011.

[2] R. Agrawal, T. Imielinski, A.N. Swami, *Mining association rules between sets of items in large databases*, SIGMOD Conference, ACM Press, 1993. pp. 207216.

[3] Bellandi, Andrea Furletti, Barbara Grossi, Valerio Romei, Andrea, (2008),*Ontological support for Association Rule Mining*, Proceedings of the IASTED International Conference on Artificial Intelligence and Applications, AIA 2008. 110-115.

[4] Ashraf Sadat Heydari Yazdi Mohsen Kahani, *A Novel Model for Mining Association Rules from Semantic Web Data* , IEEE, 2014

[5] Daniel Fleischhacker Johanna Volker Heiner Stuckenschmidt, *Mining RDF Data for Property Axioms*, In: Meersman, R., et al. (eds.) OTM 2012, Part II. LNCS, vol. 7566, pp. 718735. Springer, Heidelberg (2012)

[6] *Apache Spark,* `https://spark.apache.org`

[7] *SANSA-Stack - Scalable Semantic Analytics Stack,* `http://sansa-stack.net/`

[8] *Protege,* `https://protege.stanford.edu/`

[9] `https://alvinalexander.com/`