Elements about Data Quality and Exploratory Knowledge Discovery with Formal Concept Analysis

Amedeo Napoli
LORIA (CNRS – INRIA Nancy Grand-Est – Université de Lorraine)
B.P. 239, 54506 Vandoeuvre les Nancy, France
Amedeo.Napoli@loria.fr

DaQuaTa International Workhsop
Lyon, December 12-13 2016
Summary of the presentation

1. Variations on KDD
2. Formal Concept Analysis and Pattern Structures
3. Pattern Structures applied to the Web of Data
4. Conclusion
Knowledge Discovery in Databases (KDD)

- **KDD** is applied to large volumes of complex data for discovering patterns which can be significant and reusable.

- KDD is based on three main steps: data preparation, data mining, and interpretation of the discovered units.

- KDD is iterative and interactive, i.e. it can be replayed and it is guided by an analyst.
Three Dimensions in KDD

- **The data dimension:**
  - KDD is **data oriented** by nature, and the data diversity raises problems related to **complexity**, **volume** and **distribution** of data...

- **The knowledge dimension:**
  - Data have a **context** and KDD is **knowledge oriented**, depending on **domain knowledge**, e.g. **constraints**, **preferences**...

- **The task dimension (problem-solving):**
  - KDD is intended to solve various **tasks** for human or software agents and may be guided by the task at hand.
0: (raw) data.

0 Celsius degrees: information.

When temperature is less than 0 Celsius degrees people should wear warm clothes: (actionable) knowledge.

In Winter time temperature at dawn in Nancy is around zero Celsius degrees (⇒ necessary condition).

Is it true that if temperature at dawn in Nancy is around zero Celsius degrees it is Winter time? (⇐ sufficient condition).

Necessary and sufficient conditions are the base of definitions.

Mining definitions gives an idea of the correctness and completeness of the data at hand.
Linked Data (LD) are part of the big data landscape and some challenges apply to LD: heterogeneity, diversity, complexity.

LD are part of the increasing amount of data which are contributed by citizens, posing problems of quality and provenance.

Thus the ability to find, integrate and reuse data from multiple heterogeneous sources becomes a major challenge.

Semantic technologies are meant to deal with such issues.

Linking and exploring data: exploratory search is considered to be more appropriate for navigating large infrastructure (e.g. following “same-as” links).

One key characteristic is to assigns URIs as global unique identifiers.
The data and knowledge dimensions combined

- **Functional dependencies (FDs)** are among the most common integrity constraints for DBs and they play an important role in the design and the analysis of relational DBs.
- However, in RDF data, relations are numerous and multi-valued, making correlations through FDs more complex.
- In this presentation, we introduce a framework based on “implications”, closely related to FDs, and well-adapted to the mining of LD.
The task (problem-solving) dimension

- Problem-solving can be both data-directed (synthesis) and goal-directed (analysis).
- In a data-directed approach problem-solving starts with the data first and ask research questions after mining the data for patterns (concepts, rules, relations...).
- Following this paradigm, smart data should come first and then smart applications: smart data will make future applications more reusable, more flexible and robust.
- Once data is “semantically lifted” it can be more easily integrated, reused and mined.
- Synthesis and analysis can be reconciled within a guided exploration.
Classification (partial ordering) is a polymorphic process and a good candidate for bridging discovery and representation of patterns.

- Discovery of classes for understanding data.
- Organization of classes into a partial order (subsumption).
- Representation and Reasoning: instantiation, class definition, information retrieval...

As a mathematical theory of cognitive theory of concepts and as a human centered approach, Formal Concept Analysis can play the role...
1 Variations on KDD

2 Formal Concept Analysis and Pattern Structures

3 Pattern Structures applied to the Web of Data

4 Conclusion
Classification based on Formal Concept Analysis:

A formal context \((G, M, I)\) is based on a set of objects \(G\), a set of attributes \(M\), and a binary relation \(I \subseteq G \times M\).

Two derivation operators are defined as follows, \(\forall A \subseteq G, B \subseteq M:\)

\[
A' = \{ m \in M | \forall g \in A, (g, m) \in I \} \\
B' = \{ g \in G | \forall m \in B, (g, m) \in I \}
\]

A formal concept \((A, B)\) verifies \(A' = B\) and \(A = B'\).

Formal concepts are partially ordered w.r.t. inclusion of extents (or dually of intents) for forming a concept lattice:

\((A_1, B_1) \leq (A_2, B_2)\) iff \(A_1 \subseteq A_2\)
The contexts of planets

The initial context of planets:

<table>
<thead>
<tr>
<th>Planet</th>
<th>Size</th>
<th>Distance to Sun</th>
<th>Moon(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jupiter</td>
<td>large</td>
<td>far</td>
<td>yes</td>
</tr>
<tr>
<td>Mars</td>
<td>small</td>
<td>near</td>
<td>yes</td>
</tr>
<tr>
<td>Mercury</td>
<td>small</td>
<td>near</td>
<td>no</td>
</tr>
<tr>
<td>Neptune</td>
<td>medium</td>
<td>far</td>
<td>yes</td>
</tr>
<tr>
<td>Pluto</td>
<td>small</td>
<td>far</td>
<td>yes</td>
</tr>
<tr>
<td>Saturn</td>
<td>large</td>
<td>far</td>
<td>yes</td>
</tr>
<tr>
<td>Earth</td>
<td>small</td>
<td>near</td>
<td>yes</td>
</tr>
<tr>
<td>Uranus</td>
<td>medium</td>
<td>far</td>
<td>yes</td>
</tr>
<tr>
<td>Venus</td>
<td>small</td>
<td>near</td>
<td>no</td>
</tr>
</tbody>
</table>

The context of planets after plain scaling:

<table>
<thead>
<tr>
<th>Planets</th>
<th>Size</th>
<th>Distance to Sun</th>
<th>Moon(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jupiter</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Mars</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Mercury</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Neptune</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Pluto</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Saturn</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Earth</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Uranus</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Venus</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
The concept lattice of planets
Using the concept lattice

- Visualization
- Navigation and Information Retrieval
- Interpretation of concepts and rules
Intersection considered as a similarity operator:

- $\cap$ behaves like a similarity operator:

$$\{m_1, m_2\} \cap \{m_1, m_3\} = \{m_1\}$$

- $\cap$ induces a partial order $\subseteq$ as follows:

$$S_1 \cap S_2 = S_1 \iff S_1 \subseteq S_2$$

$$\{m_1\} \cap \{m_1, m_2\} = \{m_1\} \iff \{m_1\} \subseteq \{m_1, m_2\}$$

- $\cap$ has the properties of a meet $\sqcap$ in a semi-lattice, i.e. a commutative, associative and idempotent operation:

$$c \cap d = c \iff c \sqsubseteq d$$

\[\begin{array}{|c|c|c|}
\hline
 & m_1 & m_2 & m_3 \\
\hline
g_1 & \times & & \times \\
g_2 & \times & \times & \\
g_3 & \times & \times & \times \\
g_4 & \times & \times & \times \\
g_5 & \times & \times & \times \\
\hline
\end{array}\]
A formal context \((G, M, I)\) is based on a set of objects \(G\), a set of attributes \(M\), and a binary relation \(I \subseteq G \times M\).

Two derivation operators are defined as follows, \(\forall A \subseteq G, B \subseteq M:\)

\[
A' = \{ m \in M | \forall g \in A, (g, m) \in I \}
\]

\[
B' = \{ g \in G | \forall m \in B, (g, m) \in I \}
\]

A formal concept \((A, B)\) verifies \(A' = B\) and \(A = B'\).

Formal concepts are partially ordered w.r.t. inclusion of extents (or dually of intents):

\[(A_1, B_1) \leq (A_2, B_2) \text{ iff } A_1 \subseteq A_2\]

A pattern structure \((G, (\mathcal{D}, \sqcap), \delta)\) is based on a set of objects \(G\), a meet semi-lattice of object descriptions \((\mathcal{D}, \sqcap)\), and a mapping \(\delta : G \rightarrow \mathcal{D}\) which associates a description to each object.

Two derivation operators are defined as follows, \(\forall A \subseteq G, d \in (\mathcal{D}, \sqcap):\)

\[
A'^{\square} = \sqcap_{g \in A} \delta(g)
\]

\[
d'^{\square} = \{ g \in G | d \sqsubseteq \delta(g) \}
\]

A formal concept \((A, d)\) verifies \(A'^{\square} = d\) and \(A = d'^{\square}\).

Pattern concepts are partially ordered w.r.t. inclusion of extents (or dually inclusion of intents):

\[(A_1, d_1) \leq (A_2, d_2) \text{ iff } A_1 \subseteq A_2\]
### Interval Pattern Structure \((G, (D, \sqcap), \delta)\)

| \(|\) | \(m_1\) | \(m_2\) | \(m_3\) |
|-----|-------|-------|-------|
| \(g_1\) | 5     | 7     | 6     |
| \(g_2\) | 6     | 8     | 4     |
| \(g_3\) | 4     | 8     | 5     |
| \(g_4\) | 4     | 9     | 8     |
| \(g_5\) | 5     | 8     | 5     |

\[
\{g_1, g_2\} \sqsubseteq = \sqcap_{g \in \{g_1, g_2\}} \delta(g) \\
= \langle 5, 7, 6 \rangle \sqcap \langle 6, 8, 4 \rangle \\
= \langle [5, 6], [7, 8], [4, 6] \rangle
\]

\[
\langle [5, 6], [7, 8], [4, 6] \rangle \sqsubseteq = \{g \in G | \langle [5, 6], [7, 8], [4, 6] \rangle \sqsubseteq \delta(g) \} \\
= \{g_1, g_2, g_5\}
\]

\((\{g_1, g_2, g_5\}, \langle [5, 6], [7, 8], [4, 6] \rangle)\) is a pattern concept
Interval Pattern Structure \((G, (D, \sqcap), \delta)\)

<table>
<thead>
<tr>
<th></th>
<th>Spring</th>
<th>Summer</th>
<th>Autumn</th>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paris</td>
<td>[-1.28]</td>
<td>[6.34]</td>
<td>[-2.24]</td>
<td>[-7.19]</td>
</tr>
<tr>
<td>Nice</td>
<td>[6.32]</td>
<td>[16.32]</td>
<td>[6.26]</td>
<td>[3.23]</td>
</tr>
<tr>
<td>Lyon</td>
<td>[1.35]</td>
<td>[10.34]</td>
<td>[-2.27]</td>
<td>[-5.19]</td>
</tr>
</tbody>
</table>

-1, 35 [6, 34] [-2, 27] [-7, 23]

Paris, Nice, Lyon

Paris, Nice

Paris, Lyon

Nice, Lyon

Paris

Nice

Lyon
1 Variations on KDD

2 Formal Concept Analysis and Pattern Structures

3 Pattern Structures applied to the Web of Data

4 Conclusion
DBpedia is the largest reservoir of LD in the world currently containing more than 5 million triples (all the information stored in DBpedia is obtained by parsing Wikipedia).

DBpedia content is obtained from semi-structured sources of information in Wikipedia, namely infoboxes and categories.

However, categorical information in DBpedia cannot be used as it should be expected, i.e. as a “definition of a class of documents”.

The DBpedia web site provides some query examples using the SPARQL query language (Online access).

Two generic examples are explained hereafter.

A first query has the description: **People who were born in Berlin before 1900.**

This query translates into a graph-based search of entities of the type “Person”, which have the property “birthPlace” pointing to the entity representing the “City of Berlin” and another property named “birthDate” with a value less than 1900.

```sparql
SELECT ?x
WHERE {
  ?x rdf:type dbo:Person .
  ?x dbp:birthDate dbp:Berlin .
  ?x dbp:birthPlace ?d .
  FILTER (?d <= 1900)
}
```
A second query does not work at all in the same way and asks for objects annotated as “French Films” (category).

While we could expect a graph-based search of objects of the type “Film” with a property called “hasCountry” pointing to the entity representing “France”, we have a much rougher approach. The actual SPARQL query asks for objects (of any type) annotated as “French films”.

```
SELECT ?x
WHERE {
  ?x dcterms:subject category:FrenchFilm .
}
```
For a software agent, an information is better expressed through a **definition**.

Currently, DBpedia mixes two paradigms of data access (at least…) in an effort to benefit from the structured nature of the category organization but there is no means for ensuring correctness and completeness in the data.

Accordingly, we propose a formalism for bridging the gap between the syntactic nature of categorical annotations with their semantic correspondent in the form of a concept definition.
We achieve this by mining patterns derived from entities annotated by a given category, e.g. all entities annotated as “French films” are of “type Film” and of “French nationality”.

We describe how these category-pattern equivalences can be described as “definitions” according to implication among attributes which can be mined using Formal Concept Analysis and pattern structures (for dealing with heterogeneous complex data).
Building definitions from implications and association rules

- A concept lattice can be built from the data and then used for discovering implication (i.e. association rules whose confidence is 100%) which provide a basis for “subject definition” in terms of necessary and sufficient conditions.
- If $X \implies Y$ and $Y \implies X$, then $X \equiv Y$, and $X \equiv Y$ is called a definition.
- If $X \implies Y$ and $Y \rightarrow X$ has high confidence, we may be in the presence of “incomplete data”.
- In some cases, data can be completed in order to obtain a definition.
From RDF Statements to a Formal Context

RDF triples

<Person1,dc:subject,dbpc:Computer_Scientists>
<Person1,dc:subject,dbpc:Turing_Award_Laureates>
<Person1,dbp:field,dbp:Computer_Sciences>
<Person1,rdf:type,dbo:Scientists>
...

<table>
<thead>
<tr>
<th>Predicates</th>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>URI</td>
</tr>
<tr>
<td>A</td>
<td>dc:subject</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>dbp:award</td>
</tr>
<tr>
<td>C</td>
<td>rdf:type</td>
</tr>
<tr>
<td>D</td>
<td>dbp:field</td>
</tr>
<tr>
<td>E</td>
<td>dbp:birthPlace</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure: The formal context is built from RDF triples after scaling from DBpedia. Each cross (×) corresponds to a triple <subject,predicate,object>
<table>
<thead>
<tr>
<th>Rule</th>
<th>Confidence</th>
<th>Support</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>d \implies c</td>
<td>100%</td>
<td>5</td>
<td>Every scientist has won a Turing Award.</td>
</tr>
<tr>
<td>c \implies d</td>
<td>100%</td>
<td>5</td>
<td>Every person who has won a Turing Award is a scientist.</td>
</tr>
<tr>
<td>e \implies c,d</td>
<td>100%</td>
<td>2</td>
<td>All the people having the field computer science is a Turing award winner scientist.</td>
</tr>
<tr>
<td>c,d \implies a,b</td>
<td>100%</td>
<td>5</td>
<td>All the Scientists winning “Turing Award” are categorized as “Turing Award Laureates” and “Computer Scientists”</td>
</tr>
<tr>
<td>a,b \rightarrow c,d</td>
<td>71%</td>
<td>7</td>
<td>71% of the persons categorized as “Turing Award Laureates” and “Computer Scientists” are “Scientists” who have won “Turing Award”.</td>
</tr>
</tbody>
</table>

- \( c, d \Rightarrow a, b \) but \( \text{conf} (\{a, b\} \rightarrow \{c, d\}) = 0.71 \)
- Actually, \( \text{Person}6 \) and \( \text{Person}7 \) can be considered as having an incomplete description, and a definition may exist provided that data are completed:
  \( a, b \equiv c, d \) i.e., \( a, b \Rightarrow c, d \) and \( c, d \Rightarrow a, b \)
1 Variations on KDD

2 Formal Concept Analysis and Pattern Structures

3 Pattern Structures applied to the Web of Data

4 Conclusion
Some applications of pattern structures

- **Text mining with tree-based pattern structures.**

- **Mining sequential data for analyzing patient trajectories**

- **Information Retrieval and Recommendation.**

- **Discovery of Functional Dependencies.**

- **Biclustering and Triadic Analysis.**

- **Best Patterns w.r.t. Interest Measures.**
Conclusions

- Towards a combining of **symbolic and numeric** knowledge discovery methods.
- Implementing **Latent Semantic Indexing** within FCA for feature selection.
- **Biclustering and recommendation**.
- **Mining relations** between symptoms and diseases – Orphamine and the Hybride project.
- **Graphical models**, feature selection and classification.