Explaining Classifiers in Ontology-Based Data Access

Federico Croce and Maurizio Lenzerini

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

SAPIENZA
Università di Roma

Explainable Logic-Based Knowledge Representation (XLoKR 2020)

Workshop at KR2020 (17th International Conference on Principles of Knowledge Representation and Reasoning)

September 13th, 2020
Introduction
Problem statement

• Machine Learning (ML) has many elegant and efficient solutions to very difficult problems: Machine Translation, Vision, Autonomous Driving, and more

• An empiric rule shows that the more a ML algorithm is accurate, the less we understand its “magic”

• Deep learning is an extreme example of a high accuracy, black-box model
ML interpretability (empiric)

Interpretability

Accuracy

- Linear Regression
- Decision Trees
- K-Nearest Neighbors
- Random Forests
- Support Vector Machines
- Deep Neural Networks
Why should we care?

• Caring only about performances is not the right choice in many fields: finance, justice, healthcare, privacy

• One famous example is COMPAS algorithm [1], used across the US to predict future criminals, and proved to be biased against black people

Why should we care? (cont.)

AARON HOLMES SEP 11, 2020

• A sheriff launched an algorithm to predict who might commit a crime. Dozens of people said they were harassed by deputies for no reason [2].

• But according to a six-month investigation published this week by the Tampa Bay Times, the high-tech tool deployed by the Pasco Sheriff’s Office didn’t lead to a reduction in violent crimes. Instead, 21 families singled out by the algorithm said they were routinely harassed by deputies, even when there was no evidence of a specific crime.

Possible Solutions

State-of-the-art: LIME, SHAP, Scoped Rules, Counterfactual and Adversarial Examples, Feature Visualization

Possible Solutions (cont.)

- **Our solution** is a form of *reverse engineering* of an Ontology-Based Data Management (OBDM) system: finding a query over the ontology that semantically describes the tagged individuals in the dataset.
Preliminaries
Ontology-Based Data Management

It is a **three-layered architecture:**
- The **ontology** is a declarative and explicit representation of the domain of interest
- The **data layer** is constituted by the existing dataset
- The **mapping layer** is a set of declarative assertions specifying how the sources in the data layer relate to the ontology
The notion of *certain answers*

- Let $\mathcal{O}$ be an ontology, $\mathcal{S}$ a dataset, and $\mathcal{M}$ a set of mappings, we call $\mathcal{J} = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ an OBDM specification.

- Let $q_{\mathcal{O}}$ be a query over $\mathcal{O}$, we define the *certain answers* of $q_{\mathcal{O}}$ w.r.t. $\mathcal{J}$ and a database $D$, denoted by $\text{cert}^D_{q_{\mathcal{O}}, \mathcal{J}}$ as the set of tuples $\tilde{t}$ of $\mathcal{S}$-constants, such that

$$\tilde{t} \in q_{\mathcal{O}}^B \text{ for every possible interpretation } B \text{ that satisfies } \mathcal{J} \text{ for an } \mathcal{S}\text{-database } D \text{ (called a model of } \mathcal{J} \text{ w.r.t. } D)$$
The Classifier

Given a dataset $D$, we consider a binary classifier:

$$\lambda : \text{dom}(D)^n \rightarrow \{+1, -1\}$$

Also, we will denote the set of tuples that have been classified positively (resp. negatively) as:

$$\lambda^+ = \{\vec{t} \in \text{dom}(D)^n \mid \lambda(\vec{t}) = +1\}$$

(resp. $\lambda^- = \{\vec{t} \in \text{dom}(D)^n \mid \lambda(\vec{t}) = -1\}$)
The Framework
The Notion of Border

• For each tuple \( \vec{t} \in D \) and natural number \( r \), we define \( B_{\vec{t}, r} (D) \) as the **Border** of radius \( r \) for \( t \) in \( D \), representing all the atoms in \( D \) that are *reachable* from \( \vec{t} \) in at most \( r \) joins.

*Example:* Let a database be \( D = \{ R(a, b), S(a, c), Z(c, d), W(d, e), W(e, h), R(f, g) \} \) and let \( t = \langle a \rangle \). By denoting with \( W_{t,n} (D) \) the atoms in \( D \) that are reachable from \( t \) in at most \( n \) joins, we have that:

- \( W_{t,0} (D) = \{ R(a, b), S(a, c) \} \)
- \( W_{t,1} (D) = \{ Z(c, d) \} \)
- \( W_{t,2} (D) = \{ W(d, e) \} \)

Therefore, the border of radius 2 of \( t \) in \( D \) is:

\[
B_{t,2} (D) = \{ R(a, b), S(a, c), Z(c, d), W(d, e) \}
\]
The \( J \)-match

- A query \( q_\mathcal{O} \) \( J \)-matches a Border \( \mathcal{B}_{\mathbf{t},r}(D) \) of radius \( r \) of a tuple \( \mathbf{t} \) in a source database \( D \), if \( \mathbf{t} \) is in the certain answers of \( q_\mathcal{O} \) w.r.t to \( J \) and \( D \), i.e. if

\[
\mathbf{t} \in \text{cert}_{q_\mathcal{O},J}^{\mathcal{B}_{\mathbf{t},r}(D)}
\]
The goal of the framework

• The goal of our framework, is to find a semantic description of \( \lambda \) that is as close as possible to a set of user-defined criteria.

• Each criterion has a function associated to it, that returns a quantitative measure of how much a given query meets the criteria.

• The user also defines an expression to compute, for a given query, a unique value out of all the measures returned by the functions of each criterion.
The criteria, the functions and the expression

• $\delta_1 = \text{“Maximize the number of tuples } t \in \lambda^+ \text{ such that } q_0 \text{ } J\text{-matches } B_{t,r}(D)\text{”}$
• $\delta_2 = \text{“Minimize the number of tuples } t \in \lambda^- \text{ such that } q_0 \text{ } J\text{-matches } B_{t,r}(D)\text{”}$
• $\delta_3 = \text{“Minimize the number of disjuncts of the query } q_0\text{”}$

• $f_{\delta_1}(q_0) = \frac{|\{t \in \lambda^+ \text{ s.t. } q_0 \text{ } J\text{-matches } B_{t,r}(D)\}|}{|\lambda^+|}$
• $f_{\delta_2}(q_0) = 1 - \frac{|\{t \in \lambda^- \text{ s.t. } q_0 \text{ } J\text{-matches } B_{t,r}(D)\}|}{|\lambda^-|}$
• $f_{\delta_3}(q_0) = \frac{1}{|CQs \text{ in } q_0|}$

• $Z_F(q_0) = \frac{\alpha f_{\delta_1}(q_0) + \beta f_{\delta_2}(q_0) + \gamma f_{\delta_3}(q_0)}{\alpha + \beta + \gamma}$ (we call this the $Z$ score of $q_0$ under $F$)

where $\alpha, \beta, \gamma$ represents the importance of criterion $\delta_1, \delta_2, \delta_3$ respectively
The Algorithm
Consider the following database $D$

<table>
<thead>
<tr>
<th>STUD</th>
<th>$\lambda$</th>
<th>ENR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A10</td>
<td>+1</td>
<td>A10 Math</td>
</tr>
<tr>
<td>B80</td>
<td>+1</td>
<td>TV</td>
</tr>
<tr>
<td>C12</td>
<td>+1</td>
<td>Sap</td>
</tr>
<tr>
<td>D50</td>
<td>+1</td>
<td>Rome</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C12 Science</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Norm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D50 Science</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E25 Arts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pol</td>
</tr>
<tr>
<td>E25</td>
<td>-1</td>
<td>Pol</td>
</tr>
</tbody>
</table>
Let the ontology be:

$$O = \{ \text{MathStudent} \sqsubseteq \text{ScientificStudent}, \text{ScienceStudent} \sqsubseteq \text{ScientificStudent} \}$$

And the mappings:

$$M = \{ \text{ENR}(x, \text{Math}, z) \leadsto \text{MathStudent}(x), \text{ENR}(x, \text{Science}, z) \leadsto \text{ScienceStudent}(x), \text{ENR}(x, y, z) \leadsto \text{enrolledIn}(x, z), \text{LOC}(x, y) \leadsto \text{locatedIn}(x, y) \}$$
Example (3/7)

The corresponding borders of radius 1, for each tuple are:

\[ \mathcal{B}_{A10,1}(D) = \{\text{STUD}(A10), \text{ENR}(A10, \text{Math, TV}), \text{LOC}(\text{TV, Rome})\} \]
\[ \mathcal{B}_{B80,1}(D) = \{\text{STUD}(B80), \text{ENR}(B80, \text{Math, Sap}), \text{LOC}(\text{Sap, Rome})\} \]
\[ \mathcal{B}_{C12,1}(D) = \{\text{STUD}(C12), \text{ENR}(C12, \text{Science, Norm})\} \]
\[ \mathcal{B}_{D50,1}(D) = \{\text{STUD}(D50), \text{ENR}(D50, \text{Science, TV}), \text{LOC}(\text{TV, Rome})\} \]
\[ \mathcal{B}_{E25,1}(D) = \{\text{STUD}(E25), \text{ENR}(E25, \text{Arts, Pol}), \text{LOC}(\text{Pol, Milan})\} \]
Example (4/7)

Consider each border associated to the tuples in $\lambda^+$ as a CQ, and compute the complete s-to-o rewriting of each query, as described in [3]. In a nutshell, this means to apply all the mappings to the queries.

$q_1(A10) \leftarrow \text{MathStudent}(A10) \land \text{enrolledIn}(A10, \text{TV}) \land \text{locatedIn}(\text{TV, Rome})$

$q_2(B80) \leftarrow \text{MathStudent}(B80) \land \text{enrolledIn}(B80, \text{Sap}) \land \text{locatedIn}(\text{Sap, Rome})$

$q_3(C12) \leftarrow \text{ScienceStudent}(C12) \land \text{enrolledIn}(C12, \text{Norm})$

$q_4(D50) \leftarrow \text{ScienceStudent}(D50) \land \text{enrolledIn}(D50, \text{TV}) \land \text{locatedIn}(\text{TV, Rome})$

Example (5/7)

• To reduce the number of queries generated, we introduce the notion of **query patterns**

• We say that two CQs have the same *pattern*, if they are conjunctions of the same set of atoms

• Our intuition is that similar tuples of the database will be described by similar properties, and will form similar *query patterns* when processed by the previous steps of the algorithm

• For each *pattern*, we only keep the constants that are shared by all the queries of the pattern. All the other constants will be substituted by new variables.
Example (6/7)

- The query patterns of the example are:

\[
q_5(x) \leftarrow \text{MathStudent}(x) \land \text{enrolledIn}(x, y) \land \text{locatedIn}(y, \text{Rome})
q_6(C12) \leftarrow \text{ScienceStudent}(C12) \land \text{enrolledIn}(C12, \text{Norm})
q_7(D50) \leftarrow \text{ScienceStudent}(D50) \land \text{enrolledIn}(D50, \text{TV}) \land \text{locatedIn}(\text{TV}, \text{Rome})
\]
Example (7/7)

• Let $k$ be the highest number of atoms appearing in a query pattern. We enumerate and compute the $Z$ score of all the possible UCQs such that:
  
i. Each CQ only uses atoms that either belong to a query pattern, or are implied by one of such atoms and the ontology
  
ii. Each CQ has at most $k$ atoms

One can verify that the query $q(x) \leftarrow \text{ScientificStudent}(x)$ achieves the highest $Z$ score of 1.0, and is therefore the best explanation of the classifier $\lambda$. 
Conclusions

• Our framework uses the Ontology-Based Data Management paradigm to provide an explanation to the behavior of a classifier

• The short-term goal is to explore possible optimizations of the algorithm drafted in this presentation

• The future work includes an evaluation of the framework to real world scenarios, as well as comparison with other similar works