Dealing with Causal Explanations in Practical Knowledge Representation

Pedro Cabalar[1], Jorge Fandinno[2] and Brais Muñiz[1]

[1]University of Corunna, SPAIN
[2]University of Nebraska at Omaha, NE, USA

ACLAI'22
November 5th, 2022
Cercedilla, Spain

Explainable AI

- Edward A. Lee "Limits of Machines, Limits of Humans"
  CAIML 2022 Symposium, May 24th, 2022, Vienna

  ☞ We will soon see NNs trained to generate explanations for other NNs classification results

- Groucho Marx:

  Those are my principles, and if you don't like them...
  well, I have others

  i.e. we me get equally "convincing" explanations for p and for $\neg p$

- We humans do it all the time:
  "I feel I should reject this project application,
  afterwards, I articulate some justification for the review"

- Explanations should be logically verifiable
  and should allow balance: why p? versus why not p?

Why vs Why-not: Paintball firing squad

- Denise's team have blue paint balls; Sheldon's team uses red balls
- Accidentally, Sheldon shoots his teammate Leonard, whose chest becomes red
- Denise commands two of her (blue) team riflemen, who were pointing Leonard, to shoot
- The blue riflemen disobey and decide not to shoot
- why is Leonard's chest painted in red?
  why is not Leonard's chest painted in blue?
  why is not Denise's chest painted in blue?
  why is not Leonard's chest painted in green?

Causality versus correlation

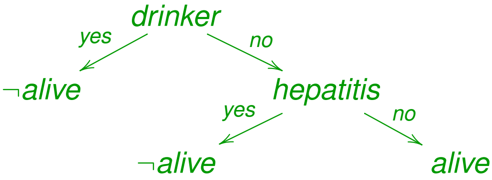
- An example domain: CHUAC hospital, A Coruña
  - $\sim$ 260 liver transplantation cases
  - $\sim$ 60 variables (features) from donor, patient and transplantation
  - Current criterion for waiting list: patient's criticality
    We want to extend it with transplantation utility
  - Explainability is crucial!
- Starting point: train a Decision Tree (DT) to predict patient's survival in next 5 years
- Hypothesis: explaining a DT prediction is obvious. Just follow the tree path
- Reality: we got a quite good DT for prediction, but the doctor found explanations counterintuitive!

Causality versus correlation

- Example: we predict $\neg$*alive* using the condition
  $\neg \mathit{drinker} \wedge \mathit{hepatitis}$ $\underbrace{\neg \mathit{drinker}}_{\mathit{good}} \wedge \underbrace{\mathit{hepatitis}}_{\mathit{bad}}$

  Reading: $\neg$*alive* **because** he doesn't drink ?!?

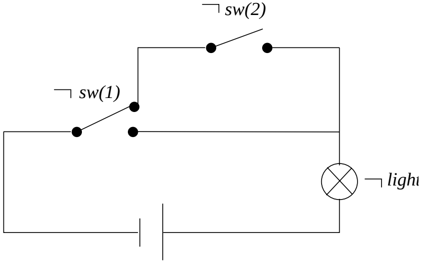
- What is actually happening:



- Causal reading:
  among non-drinkers, prediction $\neg$*alive* **because** *hepatitis*
  Moreover *hepatitis* alone suffices for the explanation $\rightarrow$ explanation redundancy in DTs [Darwiche & Marquis 2022]

Causality versus correlation

- Learning methods alone cannot distinguish correlation from causality. Symbolic learning (DTs, ILP) does not guarantee (causal) explainability!
- How do we get the causal information?
  - For explanation purposes, we can just use the Doctor's expertise
  - For discovery of causal connections, we need (at least) interventions
    ☞ Judea Pearl, "The book of the Why"
- Sometimes interventions are not enough



Causality versus correlation

- Two systems showing the same behaviour may have different causal explanations
  $p \wedge q$ versus $p \wedge (p \rightarrow q)$
- When we want to explain a KR specification, we sometimes have already provided causal information in it!
- For instance, ASP rules can be seen as causal rules

```
light(on) :- sw(1,closed).
```

Answer Set Programming as a causal formalism

Answer Set Programming (ASP) = successful paradigm for practical KR

- Declarative Problem Solving: only specification (no real "programming")
- Rich formalism for KR: defaults, choice rules, aggregates, theory atoms, ...
- Easy to change or combine reasoning modes: deduction vs abduction, simulation, planning, diagnosis, ...
- Default negation: *not p* = no evidence for *p* = there is no cause for *p*
- *not p* is a default; *p* breaks the default and has a cause

Answer Set Programming

ASP program = encoding of some problem
1 answer set = 1 solution to the problem

```
Answer: 1
o(toggle(s1),2) o(toggle(s1),5) h(light,off,0) h(protect,on,0)
h(relayline,off,0) h(s1,open,0) h(s2,open,0) h(s2,open,1)
h(s1,open,1) h(relayline,off,1) h(protect,on,1) h(light,off,1)
h(protect,on,2) h(s1,closed,2) h(s2,closed,2)
Answer: 2
o(toggle(s1),2) o(toggle(s1),5) h(light,off,0) h(protect,on,0)
h(relayline,off,0) ...
```

(a) Which is the goal behind explaining?
(b) What do we query?
(c) How does an explanation look like?

ASP explanations

(a) Which is the goal behind explaining?

- Debugging: fix something that went unexpected.
  Most literature on ASP explanations
- Causality: deal with relevant cause-effect relations
  Causes seen as breaks of defaults
- Problem solving: assist the user to find alternative solutions

(b) What do we query?

- Why p holds (or does not hold) in answer set M?
- Why is (not) M an answer set?
- Why don't we have answer sets?

Examples: why is the light eventually on?
why does cell $2, 3$ in the sudoku must contain a 9?

(c) How does an explanation look like?

- A tree or a graph relating atoms and positive/negative dependences
- Facts involved in the query through constraints:

Ex. These input cells and/or constraints are involved in having a 9

ASP explanations

Our approach

- Causal explanations
- Queries: we focus on why *p*?
  why-not *p*? not covered yet
- Explanations: (directed acyclic) graphs
  The induced tree for *p* is a Modus Ponens proof (using the positive part of the program)
- We do not generate explanations for *not p*
  defaults do not have a cause
- Remember we can use strong (constructive) negation $-p$
  (see single/married example)

1. Justified models
2. System `xclingo`
3. Conclusions

Labelled logic program

A labelled logic program is a set of labelled rules of the form:

$$\ell : H \leftarrow B \wedge N \tag{1}$$

If r is a rule of the form (1):

- $Lb(r) \stackrel{\text{def}}{=} \ell$
- $H(r) \stackrel{\text{def}}{=} H$
- $Body(r) \stackrel{\text{def}}{=} B \wedge N$
- $B^+(r) \stackrel{\text{def}}{=} B$
- $B^-(r) \stackrel{\text{def}}{=} N$

Each rule is uniquely identified by its label. Λ_P is the set of labels of program P.

Explanation: formal definition

Let P be a labelled program and $I \models P$ a classical model of P (labels ignored).
An explanation G of I under P is a labelled directed graph $G = \langle I, E, \lambda \rangle$

- whose vertices are the atoms in I
- the edges in $E \subseteq I \times I$ connect pairs of atoms
- the function $\lambda : I \to \Lambda_P$ assigns a label to each atom in I (vertex)

An explanation $G = \langle I, E, \lambda \rangle$, must satisfy:

1. G is acyclic
2. It contains no repeated labels: $\lambda(p) \neq \lambda(q)$ for every pair $p, q \in I$
3. for every $p \in I$, the rule r such that $Lb(r) = \lambda(p)$ satisfies:
   $I \models Body(r)$ and $B^+(r) = \{q \mid (q, p) \in E\} =$ incoming nodes for p.

Justified models

- A classical model $I \models P$ is justified if it has at least one explanation
- Let $SM(P)$=stable models of P, and $JM(P)$ =justified models of P.

Theorem

If P is non-disjunctive, $JM(P) = SM(P)$.

- If P has disjunction, then $SM(P) \subseteq JM(P) \subseteq SupportedModels(P)$
- We may get an exponential number of explanations, even for Horn-programs!

Examples (1)

Consider the program

$$
\begin{array}{rccl}
\ell_1 : & p & & \\
\ell_2 : & q & \leftarrow & p \\
\ell_3 : & r & \leftarrow & p, q
\end{array}
$$

One answer set: $\{p, q, r\}$.One explanation:

$$
\begin{tikzcd}
 & \ell_1 : p \arrow[dl] \arrow[dd] \\
\ell_2 : q \arrow[dr] & \\
 & \ell_3 : r
\end{tikzcd}
$$

Examples (2)

Consider the program

$$\begin{array}{lrcl} \ell_1 : & p \vee q & & \\ \ell_2 : & q & \leftarrow & p \\ \ell_3 : & p & \leftarrow & q \end{array}$$

One answer set: $\{p, q\}$.Two explanations:

$$\begin{array}{c} \ell_1 : \ p \\ \downarrow \\ \ell_2 : \ q \end{array} \qquad\qquad \begin{array}{c} \ell_1 : \ q \\ \downarrow \\ \ell_3 : \ p \end{array}$$

Chain of firing squads

Consider the program, for $i = 0 \ldots, n-1$ and some $n \geq 1$.

$$\begin{array}{rrcl} o_0: & order_0 & & \\ a_i: & firea_i & \leftarrow & order_i \\ b_i: & fireb_i & \leftarrow & order_i \\ oa_{i+1}: & order_{i+1} & \leftarrow & firea_i \\ ob_{i+1}: & order_{i+1} & \leftarrow & fireb_i \end{array}$$

Horn-program: least model = all atoms true.
But we get 2^n possible explanations obtained from the regular expression:

$$o_0; (a_0; oa_1 + b_0; ob_1); (a_1; oa_2 + b_1; ob_2); \ldots$$

Some justified models are not stable

Take the program

$$\begin{array}{rccl} \ell_1: & a & \leftarrow & \neg b \\ \ell_2: & b & \leftarrow & \neg a \\ \ell_3: & d & \leftarrow & a \wedge \neg c \\ \ell_4: & d & \leftarrow & \neg b \end{array}$$

$SM(P) = JM(P)$= two models:

- Model $\{b\}$ has one explanation with one node $\ell_2 : b$
- Model $\{a, d\}$ has two explanations

$$\begin{array}{ccc} \ell_1 : a & \qquad\qquad & \ell_1 : a \\ \downarrow & & \\ \ell_3 : d & & \ell_4 : d \end{array}$$

Some justified models are not stable

Take the program

$$\ell_1 : a \vee b$$
$$\ell_2 : a \vee c$$

- Classical model $\{a, b, c\}$ is not justified: not enough labels!
- Model $\{a, c\}$ is justified by $\{(\ell_1 : a), (\ell_2 : c)\}$
- Model $\{a, b\}$ is justified by $\{(\ell_1 : b), (\ell_2 : a)\}$
- Model $\{b, c\}$ is justified by $\{(\ell_1 : b), (\ell_2 : c)\}$
- Model $\{a\}$ has two explanations: $\{(\ell_1 : a)\}$ and $\{(\ell_2 : a)\}$

Only $\{a\}$ and $\{b, c\}$ are stable due to minimality

Hiding some labels

- Variation: some rules may be unlabelled
- We can assume the have some hidden auxiliary label
- Explanation graphs: we remove nodes with auxiliary labels and rearrange the graph afterwards

xclingo: eXplainable clingo

- Generates explanations from ASP programs
  - Provides text-based, human readable explanations.
  - User-defined text labels (*annotations*) or automatic ones
  - The user chooses the detail level. Adapt to reader's context
  - Annotations do not affect the original code. They are ASP comments (start with %).
  - Constraints can also be explained.

$clingo prog.lp

```
Solving...
Answer: 1
o(toggle(s1),2) o(toggle(s1),5) h(light,off,0) h(protect,on,0) h(relayline,off,0) h(s1,open,0) h(s2,open,0) h(s2,ope
n,1) h(s1,open,1) h(relayline,off,1) h(protect,on,1) h(light,off,1) h(protect,on,2) h(s1,closed,2) h(s2,closed,2) h(
light,on,2) h(relayline,on,2) h(relayline,on,3) h(light,on,3) h(s2,closed,3) h(s1,closed,3) h(protect,on,3) h(protec
t,on,4) h(s1,closed,4) h(s2,closed,4) h(light,on,4) h(relayline,on,4) h(light,on,5) h(s2,closed,5) h(s1,open,5) h(re
layline,off,5) h(protect,on,5) h(protect,on,6) h(relayline,off,6) h(s1,open,6) h(s2,closed,6) h(light,on,6) h(light,
on,7) h(s2,closed,7) h(s1,open,7) h(relayline,off,7) h(protect,on,7) h(protect,on,8) h(relayline,off,8) h(s1,open,8)
 h(s2,closed,8) h(light,on,8)
SATISFIABLE
```

`xclingo`: eXplainable clingo

- Generates explanations from ASP programs
  - Provides text-based, human readable explanations.
  - User-defined text labels (*annotations*) or automatic ones
  - User chooses the detail level, enabling both debugging and causal explanation.
  - Annotations do not affect the original semantics. They are ASP comments (start with %).
  - Explains fired constraints.

`$xclingo prog.lp`

```
Answer 1
  *
  |__The relay stopped working at 5
  |  |__s1 was open at 5
  |  |  |__s1 was closed at 2
  |  |  |  |__s1 was initially open
  |  |  |  |__The agent toggled s1 at 2
  |  |  |__The agent toggled s1 at 5
```

`xclingo`: eXplainable clingo

- Application 1: liver transplantation decision support system (with CHUAC hospital)
- `xclingo 1.0` = algorithm that generates a single output with all the explanations for 1 answer set M of Π
- Application 2: bAbI challenge

```
1 Daniel went to the bedroom.
2 Daniel picked up the apple there.
3 Mary grabbed the milk there.
4 Mary left the milk.
5 John journeyed to the office.
6 Daniel put down the apple there.
7 Where is the apple?          bedroom          6 1
```

System `tExplain` (Univ. of Nebraska at Omaha)

- Text translation (`Text2LP`) generates a high-level language (Sparc) later translated to `clingo`
- Explanations obtained with `xclingo`. Very demanding:
  - Final program is generated, not written by human
  - Too many explanations: normally, one suffices

`xclingo`: eXplainable clingo

- `xclingo 2.0` = builds an ASP program Π' to explain the answer M of Π. Each answer set of Π' is an explanation (graph) for M
  Advantages: Π' graphs can be queried, Π' behaviour can be extended (e.g. minimisation)
- Labelled constraints are treated as weak constraints whose violation is minimised
  When violated, the label is printed (and the proofs for their positive body atoms)

Conclusions

Lesson learnt: the specific encoding must be explanation-oriented

- Some rules are not causal:

```
person(X) :- employee(X).
person(X) :- owns(X,_).
```

- Some (auxiliary) predicates should not trigger causes
- Three versions of inertia: relevant causes are
  - Only last one (toilet light)
  - Only first one (broken glass)
  - All (adding money)
- Future work: minimisation, why-not, problem solving explanations, ...

Dealing with Causal Explanations in Practical Knowledge Representation

Pedro Cabalar[1], Jorge Fandinno[2] and Brais Muñiz[1]

[1]University of Corunna, SPAIN
[2]University of Nebraska at Omaha, NE, USA

ACLAI'22
November 5th, 2022
Cercedilla, Spain