Attribution-Scores in Data Management and Explainable Machine Learning

Leopoldo Bertossi

Explanations in Databases

Receives	*R*.1	*R*.2
	s_2	s_1
	s_3	s_3
	s_4	s_3

Store	*S*.1
	s_2
	s_3
	s_4

- Query: Are there pairs of official stores in a receiving relationship?
- $\mathcal{Q}\colon \; \exists x \exists y (\mathit{Store}(x) \wedge \mathit{Receives}(x,y) \wedge \mathit{Store}(y))$

  The query is true in D: $\quad D \models \mathcal{Q}$
- What tuples "cause" the query to be true?
- How strong are they as causes?
- We expect tuples $\mathit{Receives}(s_3, s_3)$ and $\mathit{Receives}(s_4, s_3)$ to be "causes"
- Explanations for a query result ...

- Explanations for violation of semantic conditions, integrity constraints, etc.
- A DB system could provide *explanations*
- Explanations come in different forms
- Some of them are *causal explanations*
- Want to model, specify and compute causes
- Large part of our recent research is about the use of causality

  In different ways

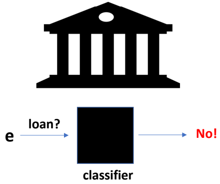
  In data management and machine learning

Explanations in Machine Learning

- Bank client $\mathbf{e} = \langle \text{john}, 18, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle$

  As an entity represented as a record of values for features Name, Age, Activity, Income, ...

- $\mathbf{e}$ requests a loan from a bank that uses a classifier



- The client asks *Why?*

- What kind of *explanation?*

  How?

  From what?

A Score-Based Approach: Responsibility

- Causality has been developed in AI for three decades or so
- In particular: Actual Causality
- Also the quantitative notion of Responsibility: a measure of causal contribution
- Both based on Counterfactual Interventions
- Hypothetical changes of values in a causal model to detect other changes

  "What would happen if we change ..."?

  By so doing identify actual causes
- Does the deletion of the DB tuple invalidates the query?
- Does a change of this feature value leads to label "Yes"?

- We have investigated actual causality and responsibility in data management and ML-based classification
- Semantics, computational mechanisms, intrinsic complexity, logic-based specifications, reasoning, etc.
- Also other *explanation scores*; a.k.a. "attribution scores"
- Assign numbers to, e.g., database tuples or features values to capture their causal, or, more generally, explanatory strength
- Some of them (in data management or ML)
  - Responsibility (in its original and generalized versions)
  - The Causal Effect score
  - The Shapley value (as Shap in ML)

This Tutorial

1. Causality in DBs
2. The DB repair connection
3. Responsibility
4. Causality under integrity constraints
5. Causal responsibility vs. causal effect
6. Shapley value in DBs
7. Responsibility of explanations for classification
8. Shapley value of explanations for classification

Companion papers: **[11]**, **[12]**

Causality in DBs

- Causal explanations for a query result: (Meliou et al., 2010)
    - Relational instance D and boolean conjunctive query (BCQ) $\mathcal{Q}$
    - A tuple $\tau \in D$ is a counterfactual cause for $\mathcal{Q}$ if $D \models \mathcal{Q}$ and $D \smallsetminus \{\tau\} \not\models \mathcal{Q}$
    - A tuple $\tau \in D$ is an actual cause for $\mathcal{Q}$ if there is a contingency set $\Gamma \subseteq D$, such that τ is a counterfactual cause for $\mathcal{Q}$ in $D \smallsetminus \Gamma$ (Halpern and Pearl, 2001)
- The responsibility of an actual cause τ for $\mathcal{Q}$:

  $\rho_D(\tau) \ := \ \frac{1}{|\Gamma| \ + \ 1}$, $|\Gamma| =$ size of smallest contingency set for τ

  (0 otherwise)
- High responsibility tuples provide more interesting explanations (Chockler and Halpern, 2004)

Example

- Database D with relations R and S below

  $\mathcal{Q}:\ \exists x \exists y (S(x) \land R(x,y) \land S(y))$ Here: $D \models \mathcal{Q}$

R	A	B
	a_4	a_3
	a_2	a_1
	a_3	a_3

S	A
	a_4
	a_2
	a_3

- Causes for $\mathcal{Q}$ to be true in D?
- $S(a_3)$ is counterfactual cause for $\mathcal{Q}$:

  If $S(a_3)$ is removed from D, $\mathcal{Q}$ is no longer an answer
- Its responsibility is $1 = \frac{1}{1+|\emptyset|}$
- $R(a_4, a_3)$ is actual cause with contingency set $\{R(a_3, a_3)\}$

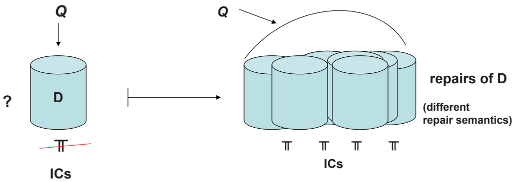
  If $R(a_3, a_3)$ is removed from D, $\mathcal{Q}$ is still true, but further removing $R(a_4, a_3)$ makes $\mathcal{Q}$ false
- Responsibility of $R(a_4, a_3)$ is $\frac{1}{2} = \frac{1}{1+1}$

  Its smallest contingency sets have size 1
- $R(a_3, a_3)$ and $S(a_4)$ are actual causes, with responsibility $\frac{1}{2}$

Computational Problems

- Among many of them:
  - Computing causes
  - Deciding if a tuple is a cause
  - Computing responsibilities
  - Computing most responsible causes (MRC)
  - Deciding if a tuple has responsibility above a threshold
- Rather complete complexity picture for CQs and UCQs
- Obtained mostly via connection between: **[2]**
  - causality and database repairs, and
  - causality and consistency-based diagnosis

Database Repairs



(Arenas et al., PODS 99)

Example: Denial constraints (DCs) (in particular, FDs)

$\neg\exists x \exists y (P(x) \wedge Q(x,y))$

$\neg\exists x \exists y (P(x) \wedge R(x,y))$

P	A
	a
	e

Q	A	B
	a	b

R	A	C
	a	c

- Subset-repairs (S-repairs): (maximal consistent subinstances)

  $D_1 = \{P(e), Q(a,b), R(a,c)\}$ $\qquad$ $D_2 = \{P(e), P(a)\}$

- Cardinality-repairs (C-repairs): (max-cardinality consistent subinstances)

  D_1

The Repair/Causality Connection

- BCQ: $\mathcal{Q}\colon\ \exists \bar{x}(P_1(\bar{x}_1) \wedge \cdots \wedge P_m(\bar{x}_m))$
- $\neg\mathcal{Q}$ becomes a DC

$$\kappa(\mathcal{Q})\colon\ \neg\exists \bar{x}(P_1(\bar{x}_1) \wedge \cdots \wedge P_m(\bar{x}_m))$$

- $\mathcal{Q}$ holds in D iff D inconsistent wrt. $\kappa(\mathcal{Q})$
- S-repairs associated to causes and minimal contingency sets

  Database tuple τ is actual cause with subset-minimal contingency set $\Gamma \iff D \smallsetminus (\Gamma \cup \{\tau\})$ is S-repair

  And its responsibility is $\frac{1}{1+|\Gamma|}$

- C-repairs associated to causes, minimum contingency sets, and maximum responsibilities

  τ is actual cause with min-cardinality contingency set Γ
  $\iff D \smallsetminus (\Gamma \cup \{\tau\})$ is C-repair And τ is MRAC

Exploiting the Connection

- Algorithmic and complexity results for repairs can be used
- Causality problem (CP): Computing/deciding actual causes can be done in polynomial time in data for CQs and UCQs (Meliou et al., 2010; **[2]**)
- Most computational problems related to repairs, in particular, C-repairs, are provably hard (data complexity) **[16]**
- Responsibility problem: Deciding if a tuple has responsibility above a certain threshold is NP-complete for UCQs **[2]**

  But *fixed-parameter tractable* (parameter inverse of threshold)
- Computing $\rho_D(\tau)$ is $FP^{NP(log(n))}$-complete for BCQs

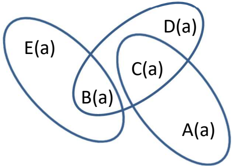
  The *functional* version of the responsibility problem
- Deciding if τ is a most responsible cause is $P^{NP(log(n))}$-complete for BCQs **[2]**

- Why repairs? Nothing special, but
  - Results for them were available
  - And obtained via more fundamental algorithmic/complexity results for graphs and hypergraphs (much more investigated)
  - Repairs can be formulated in (hyper)graph-theoretic terms **[16]**

Example: Inconsistent DB $D = \{A(a), B(a), C(a), D(a), E(a)\}$

$\Sigma = \{\neg\exists x(B(x) \wedge E(x)),\ \neg\exists x(B(x) \wedge C(x) \wedge D(x)),\ \neg\exists x(A(x) \wedge C(x))\}$

Conflict hypergraph (CHG): tuples are the nodes; hyperedges connect tuples that together violate a DC (bounded-size hyperedges)



S-repairs are maximal independent sets:

$D_1 = \{B(a), C(a)\},\ \ D_2 = \{C(a), D(a), E(a)\},\ \ D_3 = \{A(a), B(a), D(a)\}$

C-repairs: $D_2,\ D_3$ (correspond to minimum hitting sets for hyperedges)

- A tuple's responsibility is the size of a minimum vertex cover that contains it

Causality under Integrity Constraints

- ICs reflect some sort of (in)dependence among DB tuples (when satisfied)

  They should have an impact on causality

- Need a definition that involves them

  Counterfactual subinstances obtained by tuple deletions should satisfy them **[3]**

- Start assuming that $D \models \Sigma$ (the ICs)

- For $\tau \in D$ to be actual cause for $\mathcal{Q}(\bar{a})$, the contingency set Γ must satisfy:

$$D \smallsetminus \Gamma \models \Sigma \qquad\qquad D \smallsetminus \Gamma \models \mathcal{Q}(\bar{a})$$

$$D \smallsetminus (\Gamma \cup \{\tau\}) \models \Sigma \qquad\qquad D \smallsetminus (\Gamma \cup \{\tau\}) \not\models \mathcal{Q}(\bar{a})$$

- Responsibility $\rho^{D,\Sigma}_{\mathcal{Q}(\bar{a})}(\tau)$ defined as before

- Example:

Dep	DName	TStaff
t_1	Computing	John
t_2	Philosophy	Patrick
t_3	Math	Kevin

Course	CName	TStaff	DName
t_4	COM08	John	Computing
t_5	Math01	Kevin	Math
t_6	HIST02	Patrick	Philosophy
t_7	Math08	Eli	Math
t_8	COM01	John	Computing

(A) $\mathcal{Q}(x)\colon \exists y \exists z(Dep(y,x) \wedge Course(z,x,y))$ $\langle \text{John} \rangle \in \mathcal{Q}(D)$

(a) t_1 is counterfactual

(b) t_4 with single minimal contingency set $\Gamma_1 = \{t_8\}$

(c) t_8 with single minimal contingency set $\Gamma_2 = \{t_4\}$

- IC ψ: $\forall x \forall y\ (Dep(x,y) \rightarrow \exists u\ Course(u,y,x))$ (satisfied)

- t_4, t_8 not actual causes anymore: $D \smallsetminus \Gamma_1 \models \psi$, but $D \smallsetminus (\Gamma_1 \cup \{t_4\}) \not\models \psi$

- t_1 still is counterfactual cause

(B) $\mathcal{Q}_1(x)\colon\ \exists y\ Dep(y,x)$ $\langle \text{John} \rangle \in \mathcal{Q}_1(D)$

- Under IC: same causes as $\mathcal{Q}$: $\mathcal{Q} \equiv_\psi \mathcal{Q}_1$

(C) $\mathcal{Q}_2(x)$: $\exists y \exists z\, Course(z, x, y)$ $\langle \text{John} \rangle \in \mathcal{Q}_2(D)$

- W/O ψ: t_4 and t_8 only actual causes, with $\Gamma_1 = \{t_8\}$ and $\Gamma_2 = \{t_4\}$, resp.
- With IC: t_4 and t_8 still actual causes
- Contingency sets?
- We lose Γ_1 and Γ_2:

$$D \smallsetminus (\Gamma_1 \cup \{t_4\}) \not\models \psi, \quad D \smallsetminus (\Gamma_2 \cup \{t_8\}) \not\models \psi$$

- Smallest contingency set for t_4: $\Gamma_3 = \{t_8, t_1\}$

  Smallest contingency set for t_8: $\Gamma_4 = \{t_4, t_1\}$
- Responsibilities of t_4, t_8 decrease:

$$\rho^{D}_{\mathcal{Q}_2(\text{John})}(t_4) = \frac{1}{2}, \quad \text{but} \quad \rho^{D,\psi}_{\mathcal{Q}_2(\text{John})}(t_4) = \frac{1}{3}$$

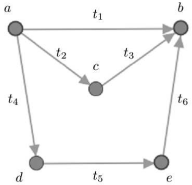
- t_1 is still not an actual cause, but affects the responsibility of actual causes

Additional Results

- Causality and ICs:
  - Causes preserved under logical equivalence of queries under ICs
  - Without ICs, deciding causality for CQs is tractable, but their presence may make complexity grow
  - There are a CQ $\mathcal{Q}$ and an inclusion dependency ψ, for which deciding causality is NP-complete (in data) **[3]**
- Causality beyond UCQs:
  - What about causality for Datalog queries?
  - For Datalog queries, cause computation can be NP-complete (vs. PTIME for UCQs)
  - Through a connection to Datalog abduction **[3]**

Causal Responsibility and Causal Effect

- Causal responsibility can be seen as an explanation score for database tuples in relation to query results
- It is not the only possible score
- Example: BQ Π is true if there is a path between a and b



$$\begin{aligned} yes &\leftarrow P(a,b) \\ P(x,y) &\leftarrow E(x,y) \\ P(x,y) &\leftarrow P(x,z), \\ & \quad\; E(z,y) \end{aligned}$$

E	X	Y
t_1	a	b
t_2	a	c
t_3	c	b
t_4	a	d
t_5	d	e
t_6	e	b

- $E \cup \Pi \models yes$ (query in Datalog, also union of CQs)
- All tuples are actual causes: every tuple in a path from a to b
- All tuples have the same responsibility: $\frac{1}{3}$
- Maybe counterintuitive: t_1 provides a direct path from a to b

- We proposed using an alternative to causal responsibility **[17]**
  A *causal effect* score
- With origin in causality for observational studies
- Retake question about how answer to query $\mathcal{Q}$ changes if τ is deleted/inserted from/into D
- Formulated as an *intervention* on a *structural causal model*
  What model?
- In this case provided by the the *lineage* of the query

Example: $D = \{R(a,b), R(a,c), R(c,b), S(b), S(c)\}$

BCQ $\mathcal{Q}:\ \exists x(R(x,y) \wedge S(y))$

- True in D, with lineage instantiated on D given by propositional formula:

$$\Phi_{\mathcal{Q}}(D) = (X_{R(a,b)} \wedge X_{S(b)}) \vee (X_{R(a,c)} \wedge X_{S(c)}) \vee (X_{R(c,b)} \wedge X_{S(b)}) \quad (*)$$

- X_τ: propositional variable that is true iff tuple $\tau \in D$
- Want to quantify contribution of a tuple to a query answer
- Assign uniform and independent probabilities to tuples in D

R^p	A	B	prob
	a	b	$\frac{1}{2}$
	a	c	$\frac{1}{2}$
	c	b	$\frac{1}{2}$

S^p	B	prob
	b	$\frac{1}{2}$
	c	$\frac{1}{2}$

Probabilistic database D^p

(tuples outside D get probability 0)

- X_τ's independent, identically distributed Bernouilli random variables

  $\mathcal{Q}$ is Bernouilli random variable

- This is because causal effect needs (assumes) a probability distribution
- What's the probability that $\mathcal{Q}$ takes a particular truth value when an intervention is performed on D?
- Interventions of the form $\quad do(X = x)$

  In the *structural equations* make X take value x
- For $\;y, x \in \{0, 1\}$: $\quad P(\mathcal{Q} = y \mid do(X_\tau = x))$?

  Corresponds to making X_τ false or true
- E.g. $\;do(X_{S(b)} = 0)$ leaves lineage (*) in the form:

$$\Phi_{\mathcal{Q}}(D)\frac{X_{S(b)}}{0} := (X_{R(a,c)} \wedge X_{S(c)}) \qquad (**)$$

- The *causal effect* of τ: (an expected difference)

$$\mathcal{CE}^{D,\mathcal{Q}}(\tau) \;:=\; \mathbb{E}(\mathcal{Q} \mid do(X_\tau = 1)) - \mathbb{E}(\mathcal{Q} \mid do(X_\tau = 0))$$

Example: (cont. page 21) $\mathcal{CE}^{D,\mathcal{Q}}(S(b)) = ?$

- For $do(X_{S(b)} = 0)$: (tuple deletion)

  Probability that instantiated lineage (**) is true (in D^p):

  $$P(\mathcal{Q} = 1 \mid do(X_{S(b)} = 0)) = P(X_{R(a,c)} = 1) \times P(X_{S(c)} = 1) = \tfrac{1}{4}$$

- For $do(X_{S(b)} = 1)$, instantiated lineage:

  $$\Phi_{\mathcal{Q}}(D)\frac{X_{S(b)}}{1} := X_{R(a,b)} \vee (X_{R(a,c)} \wedge X_{S(c)}) \vee X_{R(c,b)}$$

  Probability of it being true in D^p:

  $$P(\mathcal{Q} = 1 \mid do(X_{S(b)} = 1)) = P(X_{R(a,b)} \vee (X_{R(a,c)} \wedge X_{S(c)}) \vee X_{R(c,b)} = 1) = \cdots = \frac{13}{16}$$

- $\mathbb{E}(\mathcal{Q} \mid do(X_{S(b)} = 0)) = P(\mathcal{Q} = 1 \mid do(X_{S(b)} = 0)) = \frac{1}{4}$

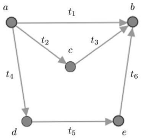
  $\mathbb{E}(\mathcal{Q} \mid do(X_{S(b)} = 1)) = \frac{13}{16}$

- $\mathcal{CE}^{D,\mathcal{Q}}(S(b)) = \frac{13}{16} - \frac{1}{4} = \frac{9}{16} > 0$

  An actual cause with this causal effect!

Example: (cont. page 19)

- Lineage of the query as a Boolean UCQs:



$$\Phi_{\mathcal{Q}}(D) = X_{t_1} \vee (X_{t_2} \wedge X_{t_3}) \vee (X_{t_4} \wedge X_{t_5} \wedge X_{t_6})$$

- $\mathcal{CE}^{D,\mathcal{Q}}(t_1) \ = \ 0.65625$
  $\mathcal{CE}^{D,\mathcal{Q}}(t_2) \ = \ \mathcal{CE}^{D,\mathcal{Q}}(t_3) \ = \ 0.21875$
  $\mathcal{CE}^{D,\mathcal{Q}}(t_4) \ = \ \mathcal{CE}^{D,\mathcal{Q}}(t_5) \ = \ \mathcal{CE}^{D,\mathcal{Q}}(t_6) = 0.09375$

- The causal effects are different for different tuples!

- More intuitive result than responsibility!

- It has been applied to aggregate queries **[17]**

- Causal Effect can be alternatively obtained via coalition game theory (coming)

Coalition Games and the Shapley Value

- Our initial motivation: How much does a database tuple contribute to the inconsistency of a DB?

  To the violation of ICs

- Similar ideas applicable to contribution to query result **[14, 15]**

- Usually *several tuples together* violate an IC or produce a query result

- Like players in a *coalition game* contributing, possibly differently, to a shared wealth-distribution function

- Apply standard measures used in game theory: the Shapley value of a player (as a measure of its contribution)

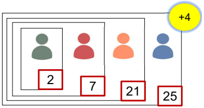
- Here database tuples become the players

- We need a game (function) ...

- Set of players D, and game function $\mathcal{G}: \ \mathcal{P}(D) \longrightarrow \mathbb{R}$
  ($\mathcal{P}(D)$ the power set of D)
- The Shapley value of player p among a set of players D:

$$Shapley(D, \mathcal{G}, p) := \sum_{S \subseteq D \setminus \{p\}} \frac{|S|!(|D|-|S|-1)!}{|D|!}(\mathcal{G}(S \cup \{p\}) - \mathcal{G}(S))$$

- $|S|!(|D|-|S|-1)!$ is number of permutations of D with all players in S coming first, then p, and then all the others
- Expected contribution of player p under all possible additions of p to a partial random sequence of players followed by a random sequence of the rest of the players



- Database tuples (and later feature values for an entity) will be players in a game

- The Shapley value is a established measure of contribution by players to a wealth function
- It emerges as the only measure enjoying certain properties
- For each application one defines an appropriate game function
- Shapley is difficult to compute

  Naive approach: exponentially many counterfactual combinations
- Actually, Shapley computation is #P-hard in general
- A complexity class of (possibly implicitly) computational counting problems
- Being #P-hard is evidence of difficulty: $\#SAT$ is #P-hard

  Counting satisfying assignments for a propositional formula

  At least as difficult as SAT

Shapley Values as Scores in DBs

- Database tuples can be seen as players in a coalition game
- Query $\mathcal{Q}: \ \exists x \exists y (\mathit{Store}(x) \land \mathit{Receives}(x, y) \land \mathit{Store}(y))$

  It takes values 0 or 1 in a database
- Game function becomes the Boolean value of the query

  The numerical value if $\mathcal{Q}$ is aggregate query
- Contribution of tuple τ to query result:

$$\mathit{Shapley}(D, \mathcal{Q}, \tau) := \sum_{S \subseteq D \setminus \{\tau\}} \frac{|S|!(|D|-|S|-1)!}{|D|!}(\mathcal{Q}(S \cup \{\tau\}) - \mathcal{Q}(S))$$

- All possible permutations of subinstances of D
- Average of differences between having τ or not
- We investigated algorithmic, complexity and approximation problems **[14, 15]**

- Players (tuples) can be split into endogenous and exogenous

  One wants to measure the contribution of endogenous tuples

  Exogenous are not subject to counterfactual interventions

  (they stay in all subinstances)

  They could be those in a particular table or particular source

- Consider BCQs without self-joins

  E.g. $\mathcal{Q}: \ \exists x \exists y(R(x) \wedge S(x,y) \wedge R(y))$ has self-join

- Dichotomy Theorem: For every fixed query $\mathcal{Q}$:

  (a) If $\mathcal{Q}$ hierarchical, then, for every D:

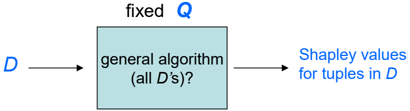
  $\{\mathit{Shapley}(D,\mathcal{Q},\tau) \mid \tau \in D\}$ can be computed in PTIME

  (b) If $\mathcal{Q}$ is non-hierarchical, the problem is $FP^{\#P}$-complete

  Among the hardest problems in the class of computational problems that run in PTIME calling an oracle from $\#P$, say $\#SAT$

- Bottom line: dichotomy easy vs. hard, and every query falls in one of the two cases

- The second case: $\mathcal{Q}$ fixed, non-hierarchical



Under usual complexity assumptions/conjectures, no PTIME algorithm (in the size of input D)

Every algorithm is bound to encounter hard input DBs D!

- $\mathcal{Q}$ is hierarchical if for every two existential variables x and y:

  $Atoms(x) \subseteq Atoms(y)$, or $Atoms(y) \subseteq Atoms(x)$, or

  $Atoms(x) \cap Atoms(y) = \emptyset$

- Example: $\mathcal{Q}: \ \exists x \exists y \exists z (R(x,y) \wedge S(x,z))$

  $Atoms(x) = \{R(x,y),\ S(x,z)\}$, $Atoms(y) = \{R(x,y)\}$,
  $Atoms(z) = \{S(x,z)\}$

  Hierarchical!

- Example: $\mathcal{Q}^{nh}: \ \exists x \exists y (R(x) \wedge S(x,y) \wedge T(y))$

  $Atoms(x) = \{R(x),\ S(x,y)\}$, $Atoms(y) = \{S(x,y), T(y)\}$

  Not hierarchical!

- Easily syntactically testable!

- Same criteria as for QA over prob DBs (Dalvi & Suciu; 2004)
  But new proof techniques were required
  However, there are newer unifying results
  (Deutch et al., Sigmod'22; c.f. also **[13]**)
- Dichotomy extends to summation over CQs
  (Shapley as an expectation, is linear)
- Hardness extends to aggregate non-hierarchical queries: max, min, avg
- What to do in hard cases?
- Approximation: For every fixed BCQ $\mathcal{Q}$, there is a multiplicative fully-polynomial randomized approximation scheme (FPRAS)
  An algorithm $A(\cdot, \epsilon, \delta)$ depending on given ϵ, δ, with:

$$P(\tau \in D \mid \frac{Sh(D, \mathcal{Q}, \tau)}{1+\epsilon} \leq A(\tau, \epsilon, \delta) \leq (1+\epsilon)Sh(D, \mathcal{Q}, \tau)\}) \geq 1-\delta$$

(also applies to summations)

- The Shapley value has been applied to measure contribution of tuples to inconsistency of a database

  For more on Shapley in data management, see **[13]**

- A related and popular score in game theory is the Banzhaf Power Index (order ignored)

  $Banzhaf(D, \mathcal{Q}, \tau) := \frac{1}{2^{|D|-1}} \cdot \sum_{S \subseteq (D \setminus \{\tau\})} (\mathcal{Q}(S \cup \{\tau\}) - \mathcal{Q}(S))$

  Banzhaf also difficult to compute; provably #P-hard in general

- Similar results obtained as for Shapley **[14]**

- Also proved: Causal-Effect score coincides with the Banzhaf Index!

Causality and XAI

- We have applied responsibility scores based on actual causality to explanations for outcomes from ML classification systems

- These methods can be applied without necessarily knowing "the internals" of the classifier

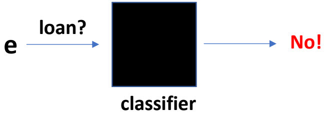
  The latter treated as a "black box" system, or being a black-box (e.g. a very complex NN)

  Only input/output relation is needed

- We have experimentally compared responsibility scores with other *local attribution scores* **[4]**

  - *Shap* (an incarnation in XAI of the Shapley value)
  - An *ad hoc* score for FICO data based on an "open-box" model (connected logistic regressions)

Resp **and Explanations** (gist and simple case)



$$\mathbf{e} \;=\; \langle \text{john}, 18, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle \qquad \text{No}$$

- Counterfactual versions:

$$\mathbf{e}' \;=\; \langle \text{john}, 25, \text{plumber}, 70\text{K}, \text{harlem}, \ldots \rangle \qquad \text{Yes}$$

$$\mathbf{e}'' = \langle \text{john}, 18, \text{plumber}, 80\text{K}, \text{brooklyn}, \ldots \rangle \qquad \text{Yes}$$

- For the gist:

1. Value for feature *Age* is counterfactual cause with explanatory responsibility $\mathit{Resp}(\mathbf{e}, \mathit{Age}) = 1$
2. Value for *Income* is actual cause with $\mathit{Resp}(\mathbf{e}, \mathit{Income}) = \frac{1}{2}$
   This one needs additional (contingent) changes ...

The *Resp* Score: Towards a General Definition

- For binary (two-valued) features the previous "definition" works fine (not in the previous example)
- Otherwise, there may be many values for a feature that do not change the label: original value not great explanation

  Similarly for features in a potential contingency set
- Better consider average labels obtained via counterfactual interventions

  Resp, our extended version of responsibility, will be expressed in terms of an expected value **[4, 9]**
- Below, $\mathcal{F}$ is the set of features, the classifier is binary, not necessarily the features

  For $F \in \mathcal{F}$, and entity $\mathbf{e}$, $F(\mathbf{e})$ is value for F in $\mathbf{e}$

  Label $L(\mathbf{e}) = 1$ is the one we want to explain

The Generalized *Resp* Score

- Assume $L(\mathbf{e}) = 1$, feature $F^\star$: want $\mathit{Resp}(\mathbf{e}, F^\star)$

  In the example, $F^\star = \text{Salary}$, $F^\star(\mathbf{e}) = 70\text{K}$, and $L(\mathbf{e}) = 1$

- With $F^\star(\mathbf{e})$ fixed, want to define "local" score for fixed contingent assignment $\Gamma := \bar{w}$ $\quad F^\star \notin \Gamma \subseteq \mathcal{F}$

  $\mathbf{e}^{\Gamma,\bar{w}} := \mathbf{e}[\Gamma := \bar{w}]$ (entity obtained changing feature values in $\mathbf{e}$ according to $\Gamma, \bar{w}$)

  $\Gamma = \{\text{Location}\}$, and $\bar{w} := \langle \text{brooklin} \rangle$, a contingent (new) value for Location

  $$\mathbf{e}^{\{\text{Location}\},\langle\text{brooklin}\rangle} = \mathbf{e}[\text{Location} := \text{brooklin}]$$
  $$= \langle \text{john}, 25, \text{plumber}, 70\text{K}, \underline{\text{brooklin}}, 10\text{K}, \text{basic} \rangle$$

- Assume $L(\mathbf{e}^{\Gamma,\bar{w}}) = L(\mathbf{e}) = 1$

  Contingent changes do not switch label alone, but after a counterfactual change for $F^\star$

  Assume $L(\mathbf{e}[\text{Location} := \text{brooklin}]) = 1$

  Maybe $\mathbf{e}^{\Gamma',\bar{w}'}$, with $\Gamma' = \{\text{Activity}, \text{Education}\}$, $\bar{w}' = \langle \text{accountant}, \text{medium} \rangle$,

  $L(\mathbf{e}^{\Gamma',\bar{w}'}) = 1$

- For each $\mathbf{e}^{\Gamma,\bar{w}}$, consider all possible values v for $F^\star$

(fixed values for all other features)

For $\mathbf{e}[\text{Location} := \text{brooklin}]$, consider $\mathbf{e}_1' :=$
$\mathbf{e}[\text{Location} := \text{brooklin}; \underline{\text{Salary} := 60\text{K}}] = \mathbf{e}^{\text{Location},\langle\text{brooklin}\rangle}[\ \text{Salary} := 60\text{K}]),$
with $L(\mathbf{e}_1') = 1$

Or $\mathbf{e}_2' := \mathbf{e}[\text{Location} := \text{brooklin}; \underline{\text{Salary} := 80}]$, with $L(\mathbf{e}_2') = 0$

- Fixed contingency $(\Gamma, \bar{w})$ on $\mathbf{e}$ as above, define its *local* responsibility score

Difference between original label and the expected label due to further modifying value of $F^\star$ in all possible ways

$$\begin{aligned} \mathit{Resp}(\mathbf{e}, F^\star, \underline{\Gamma, \bar{w}}) \quad &:= \quad \frac{L(\mathbf{e}) - \mathbb{E}(\ L(\mathbf{e}') \mid F(\mathbf{e}') = \ F(\mathbf{e}^{\Gamma,\bar{w}}),\ \forall F \in (\mathcal{F} \smallsetminus \{F^\star\}\ )}{1 + |\Gamma|} \\ &= \quad \frac{1 - \mathbb{E}(\ L(\mathbf{e}^{\Gamma,\bar{w}}[F^\star := v]) \mid v \in \mathit{Dom}(F^\star)\ )}{1 + |\Gamma|} \qquad (*) \end{aligned}$$

Takes into account the size of contingency Γ

We assume a probability distribution over entity population, whose availability or choice is quite relevant **[4]**

- $F^\star(\mathbf{e})$ is *actual cause* for label 1 if, for some $(\Gamma, \bar{w})$, (*) is positive
- $F^\star(\mathbf{e})$ is a *counterfactual cause* if $\Gamma = \emptyset$ ($\bar{w}$ is empty) and (*) is positive
- Not necessarily all counterfactual causes (as original values in $\mathbf{e}$) have the same causal strength

  $F_i(\mathbf{e}), F_j(\mathbf{e})$ could be both counterfactual causes, but with different values for (*)

  E.g. if changes on the former switch label "fewer times" than for the latter
- Now the global score, with "best" contingencies $(\Gamma, \bar{w})$

  In particular with Γ of minimum size

$$Resp(\mathbf{e}, F^\star) \ := \max_{\Gamma,\bar{w}:\ |\Gamma| \text{ is min. } \& \ (^*) > 0} \ Resp(\mathbf{e}, F^\star, \Gamma, \bar{w})$$

$$Resp(\mathbf{e}, F^{\star}) \ := \max_{\Gamma,\bar{w}\colon |\Gamma| \text{ is min. } \& \ (^{*}) > 0} \ Resp(\mathbf{e}, F^{\star}, \Gamma, \bar{w})$$

- Computation:
  1. First find minimum-size contingency sets Γ's with associated updates $\bar{w}$ with (*) greater that 0
  2. Next, find the maximum value for (*) over those pairs $(\Gamma, \bar{w})$
  3. Starting with $\Gamma = \emptyset$, and iteratively increasing the cardinality of Γ find a $(\Gamma, \bar{w})$
  4. Stop increasing the cardinality, and just check if there is $(\Gamma', \bar{w}')$ with a greater value for (*) and same cardinality

Remarks on *Resp* (and other scores)

- We are usually interested in feature values with maximum scores

  Associated to minimum (cardinality) contingency sets
- Already with binary domains, *Resp* is intractable **[6]**
- *Resp* does not require the internals of a classifier
- It has been positively compared to other scores **[4]**

  Also shows optimizations of its computation
- Can we compute it faster when we have access to the internals?

  This kind of research was done for *Shap* (coming)

Shap Scores

- Based on the general Shapley value
- Set of players $\mathcal{F}$ contain features, relative to classified entity $\mathbf{e}$
- We need an appropriate $\mathbf{e}$-dependent game function that maps (sub)sets of players to real numbers
- For $S \subseteq \mathcal{F}$, and $\mathbf{e}_S$ the projection of $\mathbf{e}$ on S:

$$\mathcal{G}_{\mathbf{e}}(S) \ := \ \mathbb{E}(L(\mathbf{e}') \mid \mathbf{e}' \in \mathcal{E} \ \ \& \ \ \mathbf{e}'_S = \mathbf{e}_S)$$

- For a feature $F^\star \in \mathcal{F}$, compute: $\mathit{Shap}(\mathcal{F}, \mathcal{G}_{\mathbf{e}}, F^\star)$

$$\sum_{S \subseteq \mathcal{F} \setminus \{F^\star\}} \frac{|S|!(|\mathcal{F}|-|S|-1)!}{|\mathcal{F}|!} [\underbrace{\mathbb{E}(L(\mathbf{e}'|\mathbf{e}'_{S\cup\{F^\star\}} = \mathbf{e}_{S\cup\{F^\star\}})}_{\mathcal{G}_{\mathbf{e}}(S\cup\{F^\star\})} - \underbrace{\mathbb{E}(L(\mathbf{e}')|\mathbf{e}'_S = \mathbf{e}_S)}_{\mathcal{G}_{\mathbf{e}}(S)}]$$

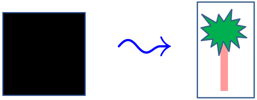
- *Shap* score has become popular (Lee & Lundberg, 2017)
- Assumes a probability distribution on entity population

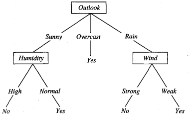
Shap Tractability?

- *Shap* may end up considering exponentially many combinations

  And multiple passes through the black-box classifier

- Can we do better with an open-box classifier?





  Exploiting its elements and internal structure?

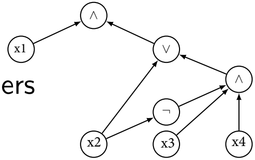
- What if we have a decision tree, or a random forest, or a Boolean circuit?

- Can we compute *Shap* in polynomial time?

Tractability for BC-Classifiers: Big Picture

- We investigated this problem in detail **[1]**
- Tractable and intractable cases, with algorithms for the former

  Investigated good approximation algorithms
- Choosing the right abstraction (model) is crucial
- We considered Boolean-Circuit Classifiers (BCCs), i.e. propositional formulas with (binary) output gate



- It was known already that *Shap* is intractable for "Monotone 2CNF"-classifiers under the product distribution [4]
- So, it had to be a broad and interesting class of BCs

Shap for Boolean-Circuit Classifiers

- Features $F_i \in \mathcal{F},\ i = 1, \ldots, n, \quad Dom(F_i) = \{0, 1\}$,
  $\mathbf{e} \in \mathcal{E} := \{0, 1\}^n, \quad L(\mathbf{e}) \in \{0, 1\}$
- There is also a probability distribution P on $\mathcal{E}$
- For BC-classifier L: $\quad Shap(\mathcal{F}, G_{\mathbf{e}}, F^\star) \ =$

  $\sum_{S \subseteq \mathcal{F} \setminus \{F^\star\}} \frac{|S|!(|\mathcal{F}|-|S|-1)!}{|\mathcal{F}|!} [\mathbb{E}(L(\mathbf{e}'|\mathbf{e}'_{S \cup \{F^\star\}} = \mathbf{e}_{S \cup \{F^\star\}}) - \mathbb{E}(L(\mathbf{e}')|\mathbf{e}'_S = \mathbf{e}_S)]$

  Depends on $\mathbf{e}$ and L
- $SAT(L) := \{\mathbf{e}' \mid L(\mathbf{e}') = 1\} \qquad \#SAT(L) := |SAT(L)|$

  Counting the number of inputs that get label 1
- We established that *Shap* is at least as hard as model counting for the BC:

  Proposition: For the uniform distribution P^u, and $\mathbf{e} \in \mathcal{E}$

  $$\#SAT(L) \ = \ 2^{|\mathcal{F}|} \times (\ L(\mathbf{e}) - \textstyle\sum_{i=1}^{n} Shap(\mathcal{F}, G_{\mathbf{e}}, F_i)\ )$$

- Then: $\#SAT \leq^{Turing}_{PTIME} Shap$

  When $\#SAT(L)$ is hard for a Boolean classifier L, $Shap$ is also hard

- Negative Corollary: Computing $Shap$ is $\#P$-hard for

  - Linear perceptron classifier
    By reduction from $\#Knapsack$ (with weights in binary)

  - Boolean classifiers defined by Monotone 2DNF or Monotone 2CNF (Provan & Ball, 1983)

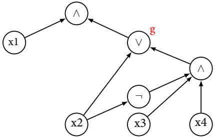
- Can we do better for other classes of binary classifiers?

  Other classes of Boolean-circuit classifiers?

Deterministic and Decomposable BCs

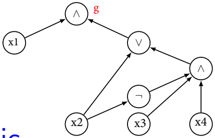
- A Boolean circuit over set of variables X is a DAG $\mathcal{C}$ with:
  - Each node without incoming edges (input) is labeled with either a variable $x \in X$ or a constant in $\{0,1\}$
  - Each other node is labeled with a gate in $\{\neg, \wedge, \vee\}$
  - There is a single sink node, O, called the output
- $\mathbf{e}\colon X \to \{0,1\}$ (equivalently $\mathbf{e} \in \{0,1\}^{|X|}$) is accepted by $\mathcal{C}$, written $\mathcal{C}(\mathbf{e}) = 1$, iff O takes value 1
- For a gate g of $\mathcal{C}$, $\mathcal{C}(g)$ is the induced subgraph containing gates on a path in $\mathcal{C}$ to g

  $Var(g)$ is the set of variables of $\mathcal{C}(g)$
  $Var(g) = \{x2, x3, x4\}$



- $\mathcal{C}$ is deterministic if every $\vee$-gate g with input gates g_1, g_2: $\mathcal{C}(g_1)(\mathbf{e}) \neq \mathcal{C}(g_2)(\mathbf{e})$, for every $\mathbf{e}$

- $\mathcal{C}$ is decomposable if every $\wedge$-gate g with input gates g_1, g_2: $\quad Var(g_1) \cap Var(g_2) = \emptyset$



- We concentrated on the class of deterministic and decomposable Boolean circuits (dDBCs)
- *Shap* computation in polynomial time not initially precluded
- A class of BCCs that includes -via efficient (knowledge) compilation- many interesting ones, syntactic and not ...
    - Decision trees (and random forests)
    - Ordered binary decision diagrams (OBDDs)
    - Sentential decision diagrams (SDDs)
    - Deterministic-decomposable negation normal-form (dDNNFs)

Shap for dDBCs

- Proposition: For dDBCs $\mathcal{C}$, $\#SAT(\mathcal{C})$ can be computed in polynomial time ($\not\Longrightarrow$ the same for *Shap*)

  Idea: Bottom-up procedure that inductively computes $\#SAT(\mathcal{C}(g))$, for each gate g of $\mathcal{C}$

- To show that *Shap* can be computed efficiently for dDBCs, we need a detailed analysis

- We assume the uniform distribution for the moment

- A related problem: "satisfiable circle of an entity"

  $$SAT(\mathcal{C}, \mathbf{e}, \ell) := SAT(\mathcal{C}) \cap \{\ e' \mid \underbrace{\|\mathbf{e} - \mathbf{e}'\|_1 = \ell}_{\ell \text{ value discrepancies}}\ \}$$

  $$\#SAT(\mathcal{C}, \mathbf{e}, \ell) := |SAT(\mathcal{C}, \mathbf{e}, \ell)|$$

- Proposition: If computing $\#SAT(\mathcal{C}, \mathbf{e}, \ell)$ is tractable, so is $Shap(X, \mathcal{G}_{\mathbf{e}}, x)$

- Main Lemma: $\#SAT(\mathcal{C}, \mathbf{e}, \ell)$ can be solved in polynomial time for dDBCs $\mathcal{C}$, entities $\mathbf{e}$, and $1 \leq \ell \leq |X|$

  Idea: Inductively compute $\#SAT(\mathcal{C}(g), \mathbf{e}_{Var(g)}, \ell)$ for each gate $g \in \mathcal{C}$ and integer $\ell \leq |Var(g)|$

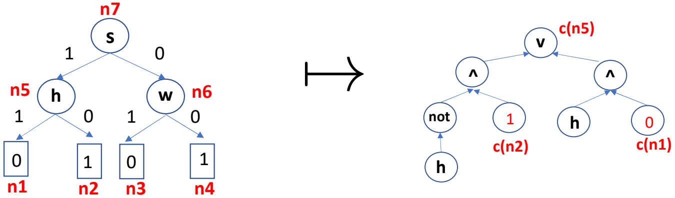
  - Input gate: immediate
  - $\neg$-gate:
    $\#SAT(\mathcal{C}(\neg g), \mathbf{e}_{Var(g)}, \ell) \ = \ \binom{Var(g)}{\ell} - \#SAT(\mathcal{C}(g), \mathbf{e}_{Var(g)}, \ell)$
  - $\vee$-gate: (uses determinism)
    $\#SAT(\mathcal{C}(g_1 \vee g_2), \mathbf{e}_{Var(g_1) \cup Var(g_2)}, \ell) \ =$
    $\#SAT(\mathcal{C}(g_1), \mathbf{e}_{Var(g_1)}, \ell) + \#SAT(\mathcal{C}(g_2), \mathbf{e}_{Var(g_2)}, \ell)$
  - $\wedge$-gate: (uses decomposition)
    $\#SAT(\mathcal{C}(g_1 \wedge g_2), \mathbf{e}_{Var(g_1) \cup Var(g_2)}, \ell) \ =$
    $\sum_{j+k=\ell} \#SAT(\mathcal{C}(g_1), \mathbf{e}_{Var(g_1)}, j) \times \#SAT(\mathcal{C}(g_2), \mathbf{e}_{Var(g_2)}, k)$

- Theorem: *Shap* can be computed in polynomial time for dDBCs under the uniform distribution
- It can be extended to any product distribution on $\{0,1\}^{|X|}$

- Corollary: Via polynomial time transformations, under the uniform and product distributions, *Shap* can be computed in polynomial time for
  - Decision trees (and random forests)
  - Ordered binary decision diagrams (OBDDs)
  - Sentential decision diagrams (SDDs)
  - Deterministic-decomposable negation normal-form (dDNNFs)
- An optimized efficient algorithm for *Shap* computation can be applied to any of these **[1]**

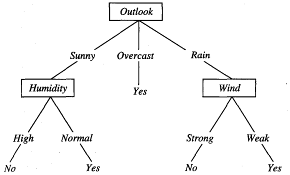
Shap for Decision Trees and ...

- Compiling binary decision trees into dDBCs
- An inductive construction starting from the bottom of the DT
- Leaves of DT become constant binary gates in dDBC
- By induction one can prove the resulting circuit is dDBC
- Final dDBC is the compilation $c(r)$ of root node r of DT

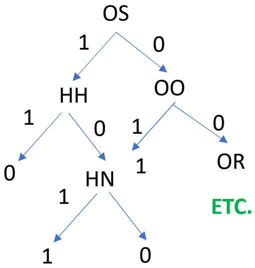


- Final equivalent dDBC: $c(n7)$
- Computable in linear time

- Beyond binary features?



- "Binarize" features
- *OutlookSunny* (OS)
  OutlookOvercast, *OutlookRain*, etc.
  become propositional features



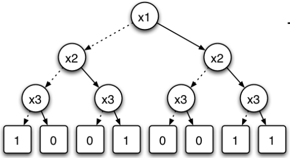
Certain entities become impossible (probability 0)

$$\mathbf{e} = \langle \underbrace{0,1,1}_{\text{for OS, OO, OR}} ,\ldots\rangle \quad \times$$

$$\mathbf{e} = \langle \underbrace{0,1,0}_{\text{for OS, OO, OR}} ,\ldots\rangle \quad \textit{ok}$$

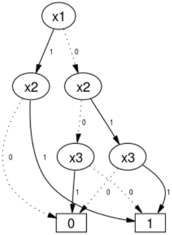
- Our polynomial time algorithm for *Shap* can be applied to *Ordered Binary Decision Diagrams* (OBDDs)
- Relevant for several reasons in *Knowledge Compilation*
- In particular, to represent "opaque" classifiers as OBDDs, e.g. binary neural networks [Shi, Shih, Darwiche, Choi; KR20]
- Opening the ground for efficiently applying *Shap* to them

$f(x_1, x_2, x_3) = (\neg x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (x_1, \wedge x_2) \vee (x_2 \wedge x_3)$



Binary Decision Tree

x1	x2	x3	*f*
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

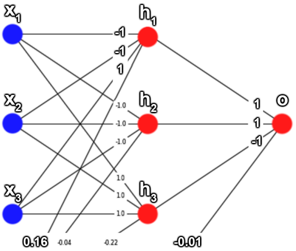


OBDD

(same variable order along full paths)

Shap on Neural Networks

- Binary Neural Networks (BNNs) are commonly considered black-box models
- Naively computing *Shap* on a BNN is bound to be complex
- Better try to compile the BNN into an open-box BC where *Shap* can be computed efficiently
- We have experimented with *Shap* computation with a black-box BNN and with its compilation into a dDBC **[10]**
- Even if the compilation is not entirely of polynomial time, it may be worth performing this one-time computation
- Particularly if the target dDBC will be used multiple times, as is the case for explanations
- We illustrate the approach by means of an example



$$\phi_g(\bar{i}) = sp(\bar{w}_g \bullet \bar{i} + b_g) := \begin{cases} 1 & \text{if } \bar{w}_g \bullet \bar{i} + b_g \geq 0, \\ -1 & \text{otherwise,} \end{cases}$$

- The BNN is described by means of a propositional formula, which is further transformed and optimized into CNF

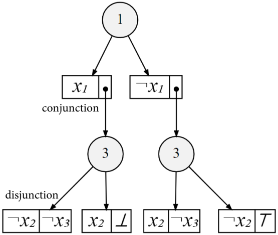
$$\begin{aligned} o \longleftrightarrow\ & (-[(x_3 \wedge (x_2 \vee x_1)) \vee (x_2 \wedge x_1)] \wedge \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \vee \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)])) \vee \\ & ([(-x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)] \wedge \\ & [(x_3 \wedge (-x_2 \vee -x_1)) \vee (-x_2 \wedge -x_1)]). \end{aligned}$$

In CNF:

$$o \longleftrightarrow (-x_1 \vee -x_2) \wedge (-x_1 \vee -x_3) \wedge (-x_2 \vee -x_3)$$

- The CNF is transformed into an SDD

  It succinctly represents the CNF

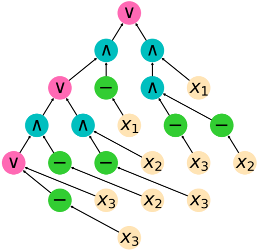


- The expensive compilation step

  But upper-bounded by an exponential only in the tree-width of the CNF

  A measure of how close to a tree is the undirected graph associated to the CNF

  An edge between variables if together in a clause



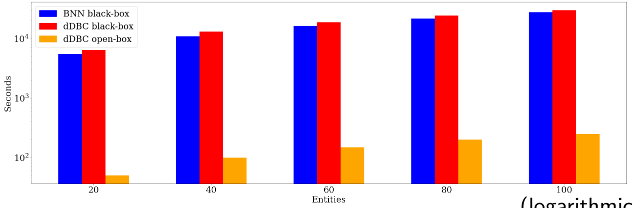
- Finally, the SDD is easily transformed into a dDBC

- On it *Shap* is computed, possibly multiple times

- With considerable efficiency gain

- In our experiments, we used a BNN with 14 gates
- It was compiled into a dDBC with 18,670 nodes
- A one-time computation that fully replaces the BNN
- We compared *Shap* computation time for black-box BNN, open-box dDBC, and black-box dDBC

  Total time for computing all Shap scores with increasing number of classification inputs



(logarithmic scale)

- The uniform distribution was used

Look Ahead

- The above results on *Shap* computation hold under the uniform and product distributions

  The latter imposes independence among features

- Other distributions have been considered for *Shap* and other scores

  The empirical and product-empirical distributions **[4]**

  They naturally arise when no more information available about the distribution

- Imposing domain semantics (domain knowledge) is relevant to explore

- Can we modify *Shap* definition and computation accordingly?

  Or the distribution? **[7]**

- Do we still have an efficient algorithm?
- In the case of databases, do complexity results change under integrity constraints (ICs)?

  That is, the implicit counterfactuals must respect the ICs
- For causal responsibility there is a change under ICs **[3]**

Self-References I

[1] M. Arenas, P. Barcelo, L. Bertossi, M. Monet. “The Tractability of SHAP-scores over Deterministic and Decomposable Boolean Circuits”. *Journal of Machine Learning Research*, 2023, 24(63):1-58. (extended version of AAAI 2021 paper).

[2] L. Bertossi, L. and B. Salimi. “From Causes for Database Queries to Repairs and Model-Based Diagnosis and Back”. *Theory of Computing Systems*, 2017, 61(1):191-232.

[3] L. Bertossi and B. Salimi. “Causes for Query Answers from Databases: Datalog Abduction, View-Updates, and Integrity Constraints”. *International Journal of Approximate Reasoning*, 2017, 90:226-252.

[4] L. Bertossi, J. Li, M. Schleich, D. Suciu and Z. Vagena. “Causality-based Explanation of Classification Outcomes”. Proc. 4th International Workshop on "Data Management for End-to-End Machine Learning" (DEEM) at ACM SIGMOD/PODS, 2020, pp. 6.1-6.10.

[5] L. Bertossi. “Specifying and Computing Causes for Query Answers in Databases via Database Repairs and Repair Programs”. *Knowledge and Information Systems*, 2021, 63(1):199-231.

[6] L. Bertossi. “Declarative Approaches to Counterfactual Explanations for Classification”. *Theory and Practice of Logic Programming*, 2021, 23 (3): 559–593, 2023. arXiv 2011.07423.

[7] L. Bertossi. “Score-Based Explanations in Data Management and Machine Learning: An Answer-Set Programming Approach to Counterfactual Analysis”. In *Reasoning Web. Declarative Artificial Intelligence*. Reasoning Web 2021. Springer LNCS 13100, 2022, pp. 145-184.

[8] L. Bertossi and G. Reyes. “Answer-Set Programs for Reasoning about Counterfactual Interventions and Responsibility Scores for Classification”. In Proc. 1st International Joint Conference on Learning and Reasoning (IJCLR'21), Springer LNAI 13191, 2022, pp. 41-56. Extended version: arXiv 2107.10159.

Self-References II

[9] L. Bertossi. "Attribution-Scores and Causal Counterfactuals as Explanations in Artificial Intelligence". In 'Reasoning Web: Causality, Explanations and Declarative Knowledge'. Springer LNCS 13759, 2023. arXiv 2303.02829.

[10] L. Bertossi and J. E. Leon. "Efficient Computation of Shap Explanation Scores for Neural Network Classifiers via Knowledge Compilation". Proc. JELIA, 2023. To appear. arXiv 2303.06516.

[11] L. Bertossi. "From Database Repairs to Causality in Databases and Beyond". To appear in special issue of Springer TLDKS dedicated to 'Bases des Donnees Avances' (BDA'22). arXiv 2306.09374

[12] L. Bertossi. "Attribution-Scores in Data Management and Explainable Machine Learning". To appear Proc. ADBIS'23. arXiv 2308.00184.

[13] L. Bertossi, B. Kimelfeld, E. Livshits and M. Monet. "The Shapley Value in Database Management". ACM Sigmod Record, 2023, 52(2):6-17.

[14] E. Livshits, L. Bertossi, B. Kimelfeld and M. Sebag. "The Shapley Value of Tuples in Query Answering". *Logical Methods in Computer Science*, 2022, 17(3):22.1-22.33.

[15] E. Livshits, L. Bertossi, B. Kimelfeld and M. Sebag. "Query Games in Databases". *ACM Sigmod Record*, 2021, 50(1):78-85.

[16] A. Lopatenko and L. Bertossi. "Complexity of Consistent Query Answering in Databases under Cardinality-Based and Incremental Repair Semantics". Proc. ICDT'07, 2007, Springer LNCS 4353, pp. 179-193. Extended version: arXiv 1605.07159.

Self-References III

[17] B. Salimi, L. Bertossi, D. Suciu and G. Van den Broeck. "Quantifying Causal Effects on Query Answering in Databases". Proc. 8th USENIX Workshop on the Theory and Practice of Provenance (TaPP), 2016. arXiv 1603.02705.

EXTRA SLIDES

The Shapley Value

Consider a set $\mathbf{X} = \{X_1, \ldots, X_n\}$ of *n agents* or *variables* or *features*, and a *utility function* $g : 2^{\mathbf{X}} \to \mathbb{R}$, and define the Shapley and Banzhaf values as:

$$\phi_{\mathbf{X},g}(X_j) = \frac{1}{n!} \sum_{\pi \in \Pi_n} \left(g(\pi^{<X_j} \cup \{X_j\}) - g(\pi^{<X_j}) \right) \tag{1}$$

$$\beta_{\mathbf{X},g}(X_j) = \frac{1}{2^n} \sum_{S \subseteq \mathbf{X}} \left(g(S \cup \{X_j\}) - g(S) \right) \tag{2}$$

where Π_n is the set of permutations of $\mathbf{X}$, and $\pi^{<X_j}$ is the set of agents that come before X_j in the permutation π

More generally, given $n+1$ numbers $a_0, \dots, a_n \in \mathbb{R}$, we define the *generalized value* as

$$\gamma_{\mathbf{X},g}(X_j) = \sum_{S \subseteq \mathbf{X}} a_{|S|} \left(g(S \cup \{X_j\}) - g(S) \right) \tag{3}$$

The Shapley and Banzhaf values are the special cases $a_k := k!(n-k-1)!$ and $a_k := 1$ respectively

The Shapley value is the unique value satisfying:

Efficiency $\sum_{j=1,m} \phi_{\mathbf{X},g}(X_j) = g(\mathbf{X})$

Symmetry $\phi_{\mathbf{X},g}(X_i) = \phi_{\mathbf{X},g}(X_j)$ whenever X_i, X_j are interchangeable, meaning that $g(S \cup \{X_i\}) = g(S \cup \{X_j\})$ for all sets S not containing X_i, X_j

Dummy $\phi_{\mathbf{X},g}(X_i) = 0$ if i does not contribute to the utility function, i.e. $g(S \cup \{X_i\}) = g(S)$ forall S

Additivity $\phi_{\mathbf{X},g_1+g_2} = \phi_{\mathbf{X},g_1} + \phi_{\mathbf{X},g_2}$

The Banzhaf value satisfies all properties above except for efficiency

The utility function $g : 2^{\mathbf{X}} \rightarrow \mathbb{R}$ is exponentially large

Various applications restrict the way the function is specified

- We can compute the expectation of the label: $\phi_0(L) = \mathbb{E}(L)$, which will give a value in $[0,1]$, say 0.4
- Now fix $\mathbf{e}^\star$, for which we have the label $L(\mathbf{e}^\star)$, e.g. 1

  We want to account for the difference between this label and $\phi_0(L)$: $L(\mathbf{e}^\star) - \phi_0(L) = 1 - 0.4 = 0.6$
- The question is which feature value contributes the most to the difference $L(\mathbf{e}^\star) - \phi_0(L)$
  In our experiments we usually concentrate on entities $\mathbf{e}^\star$ with $L(\mathbf{e}^\star) = 1$, that means "rejection", which has to be explained
- The difference is expressed as a sum of individual contributions, $\phi_i(L, \mathbf{e}^\star)$, from the different features $F_i \in \mathcal{F}$:

$$\sum_i \phi_i(L, \mathbf{e}^\star) = L(\mathbf{e}^\star) - \phi_0(L)$$