Logical and statistical approaches to Explainable AI (XAI)

Valentyn Boreiko

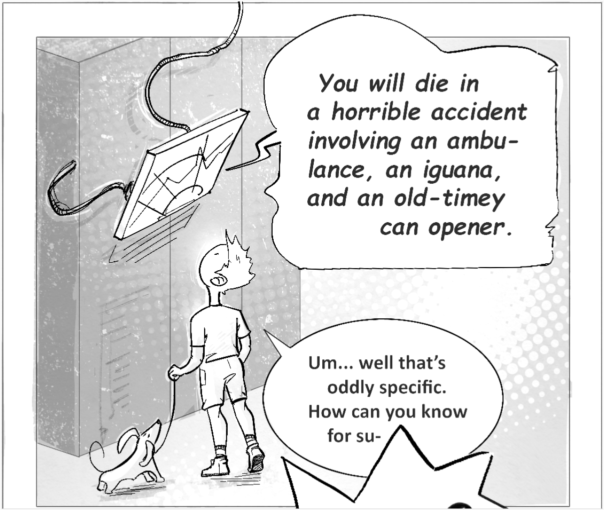
16.03.2020

Motivation

Transparency is paramount to ensuring that AI is not biased. The AI guidelines introduce a number of measures to ensure transparency in the AI industry. For instance, the data sets and processes that are used in building AI systems should be documented and traceable.

(European Parliament)

Motivation


You will die in a horrible accident involving an ambu-lance, an iguana, and an old-timey can opener.
Um... well that's oddly specific. How can you know for su-

Motivation

▸ https://miro.medium.com/max/2250/1*y04opolhQJawO0SxzBZ9MA.png

Overview

Overview

XAI is complex...

Overview

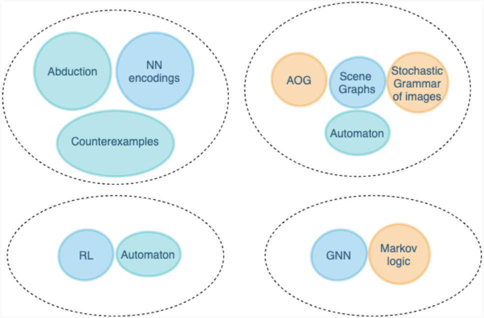
XAI is complex...

Overview

And intertwined...

Overview

And intertwined...



Overview – our focus

Overview

Computer vision (CV) & formal methods (FM)

Neural networks (NNs) in general & FM

Reinforcement learning (RL) & FM

Methods that try to generalize & FM

Computer vision (CV) & formal methods (FM)

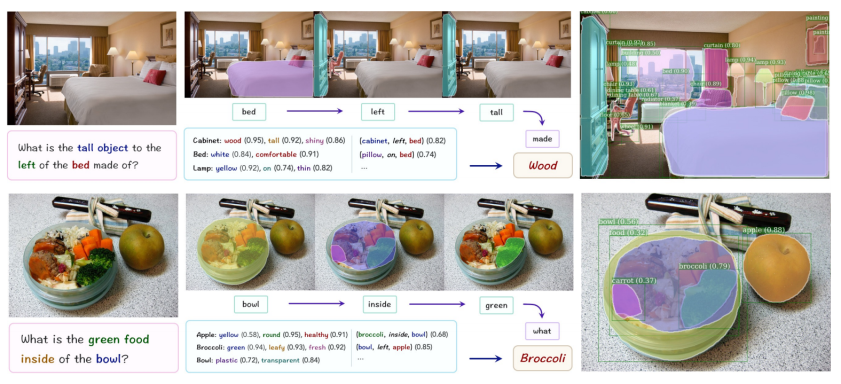
CV – Neural State Machine (NSM) – build the machine

Q: How to combine **automaton** and **neural networks-based approaches** usefully and apply them to CV

CV – Neural State Machine (NSM) – build the machine

Q: How to combine **automaton** and **neural networks-based approaches** usefully and apply them to CV
(task of visual question answering (**VQA**) in this paper)?

Q: How to combine **automaton** and **neural networks-based approaches** usefully and apply them to CV
(task of visual question answering (**VQA**) in this paper)?



D. Hudson et al. "Learning by Abstraction: The Neural State Machine". NeurIPS, 2019

CV – Neural State Machine (NSM)

A: Use NSM!

CV – Neural State Machine (NSM)

A: Use NSM!

- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$(C, S, E, \{r_i\}_{i=0}^{N}, p_0, \delta) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0, 1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

CV – Neural State Machine (NSM)

A: Use NSM!

- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$(\boxed{C}, \boxed{S}, \boxed{E}, \boxed{\{r_i\}_{i=0}^{N}}, \boxed{p_0}, \boxed{\delta}) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0,1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

CV – Neural State Machine (NSM)

A: Use NSM!

- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$(C, S, E, \{r_i\}_{i=0}^{N}, p_0, \delta) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0, 1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

CV – Neural State Machine (NSM)

A: Use NSM!

- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$\left(C, S, E, \{r_i\}_{i=0}^{N}, p_0, \delta \right) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0, 1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

CV – Neural State Machine (NSM)

A: Use NSM!

- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$(C, S, E, \{r_i\}_{i=0}^{N}, p_0, \delta) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0, 1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

A: Use NSM!

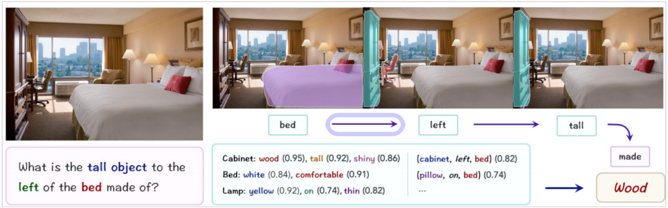
- model's *concept vocabulary* C,
- states via *object nodes* from the image,
- directed *relation edges* – transitions between the states,

$$(C, S, E, \{r_i\}_{i=0}^{N}, p_0, \delta) \tag{1}$$

- sequence of *instructions* in terms of C from the question,
- $p_0 : S \to [0, 1]$ is a (discrete, from now on) probability distribution of the initial state,
- $\delta_{S,E} : p_i \times r_i \to p_{i+1}$ is a *state transition function*.

CV – Neural State Machine (NSM) – use the machine

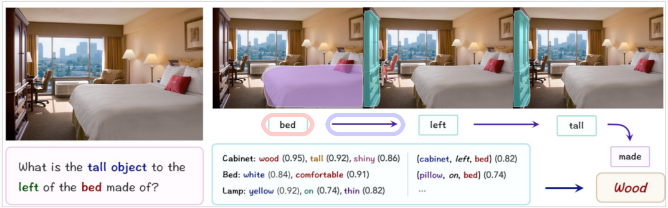
$$\delta_{S,E} \tag{2}$$



▸ D. Hudson et al. "Learning by Abstraction: The Neural State Machine". NeurIPS, 2019

CV – Neural State Machine (NSM) – use the machine

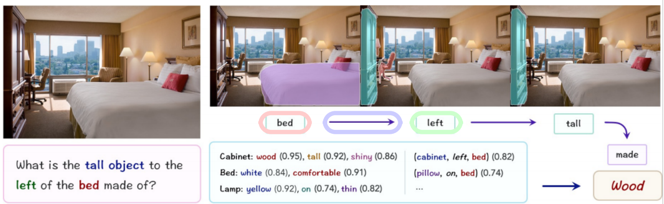
$$\delta_{S,E} : p_i \tag{2}$$



D. Hudson et al. "Learning by Abstraction: The Neural State Machine". NeurIPS, 2019

CV – Neural State Machine (NSM) – use the machine

$$\delta_{S,E} : p_i \times r_i \tag{2}$$



D. Hudson et al. "Learning by Abstraction: The Neural State Machine". NeurIPS, 2019

CV – Neural State Machine (NSM) – use the machine

$$\delta_{S,E} : p_i \times r_i \rightarrow p_{i+1} \tag{2}$$

▸ D. Hudson et al. "Learning by Abstraction: The Neural State Machine". NeurIPS, 2019

CV – NSM – summary

1. Embed scene graph and question parts using the same concept vocabulary C,
2. make an automaton out of the object nodes and transitions,
3. traverse automaton to answer the question – track the progress!

CV – NSM – summary

1. Embed scene graph and question parts using the same concept vocabulary C,
2. make an automaton out of the object nodes and transitions,
3. traverse automaton to answer the question – track the progress!

CV – NSM – summary

1. Embed scene graph and question parts using the same concept vocabulary C,
2. make an automaton out of the object nodes and transitions,
3. traverse automaton to answer the question – track the progress!

Neural networks (NNs) in general & FM

NNs in general

Q: Can FM help to make NNs in general explainable?

NNs in general

Q: Can FM help to make NNs in general explainable?

A: Yes!

NNs in general

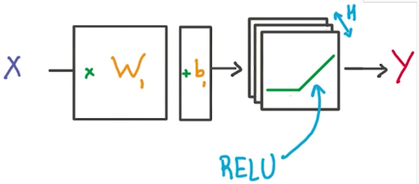
Q: Can FM help to make NNs in general explainable?
A: Yes!

We focus on 2 methods:

- NN encoding,
- differentiable inductive logic programming (∂ILP).

NNs in general – NN encoding

Mixed-Integer Linear Programming (MILP) encoding.



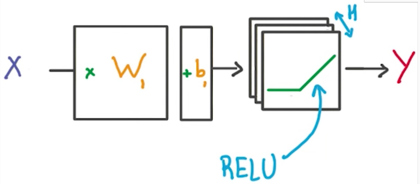
- ReLU block from NN,
- encode and solve via MILP or SMT solver or use Abduction,
- do *not* use for training of NNs, but for their explanations.

---

https://images.app.goo.gl/8TS2xkkthSTysjAJ6

NNs in general – NN encoding

Mixed-Integer Linear Programming (MILP) encoding.

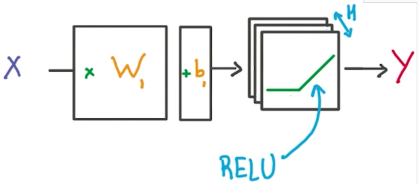


- ReLU block from NN,
- encode and solve via MILP or SMT solver or use Abduction,
- do *not* use for training of NNs, but for their explanations.

▸ https://images.app.goo.gl/8TS2xkkthSTysjAJ6

NNs in general – NN encoding

Mixed-Integer Linear Programming (**MILP**) encoding.

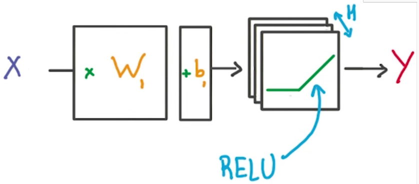


- ReLU block from NN,
- encode and solve via MILP or SMT solver or use Abduction,
- do *not* use for training of NNs, but for their explanations.

▸ https://images.app.goo.gl/8TS2xkkthSTysjAJ6

NNs in general – NN encoding

Mixed-Integer Linear Programming (MILP) encoding.



- ReLU block from NN,
- encode and solve via MILP or SMT solver or use Abduction,
- do *not* use for training of NNs, but for their explanations.

▸ https://images.app.goo.gl/8TS2xkkthSTysjAJ6

NNs in general – ∂ILP

Q: How to combine noisy data with the explainable program generation?

NNs in general – ∂ILP

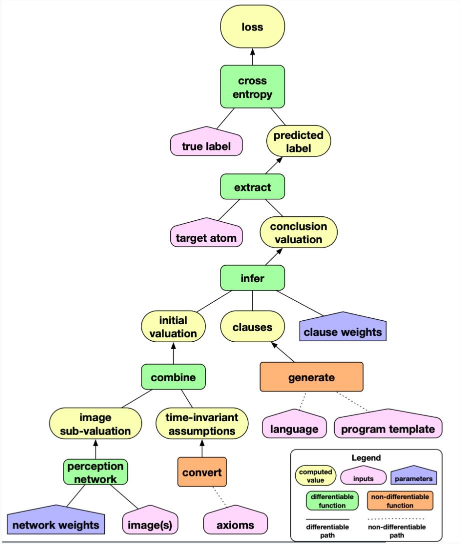
Q: How to combine noisy data with the explainable program generation?
A: Convert ILP to the binary classification!

NNs in general – ∂ILP

Q: How to combine noisy data with the explainable program generation?
A: Convert ILP to the binary classification!

Concrete example: Recognize even digits from MNIST pictures

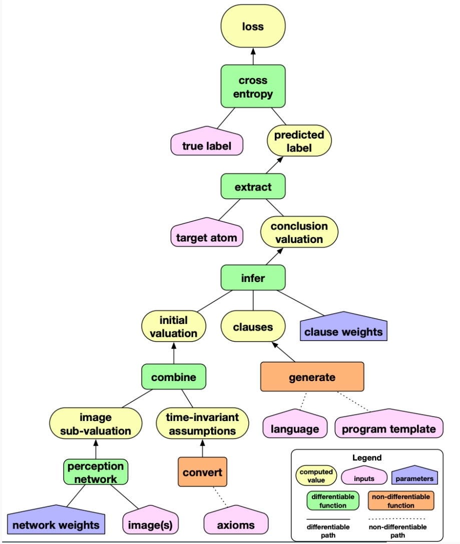
NNs in general – ∂ILP – an example



- R. Evans et al. "Learning Explanatory Rules from Noisy Data". JAIR, 2018.

- Make initial valuation – from the images and axioms,
- use the language from the predicates $\{zero/1, succesor/2, image/1, target/0, pred1/1, pred2/2\}$ and constants,
- use program template – generate all possible predicates,
- assign weights to the predicates,
- backpropagate to predict the right labels $\in \{0, 1\}$.

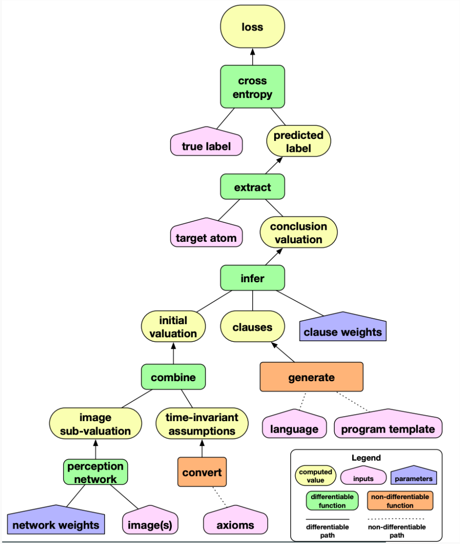
NNs in general – ∂ILP – an example



- R. Evans et al. "Learning Explanatory Rules from Noisy Data". JAIR, 2018.

- Make initial valuation – from the images and axioms,
- use the language from the predicates $\{zero/1, succesor/2, image/1, target/0, pred1/1, pred2/2\}$ and constants,
- use program template – generate all possible predicates,
- assign weights to the predicates,
- backpropagate to predict the right labels $\in \{0, 1\}$.

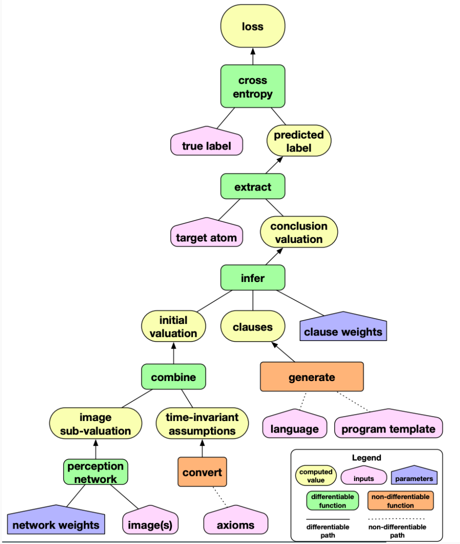
NNs in general – ∂ILP – an example



- Make initial valuation – from the images and axioms,
- use the language from the predicates $\{zero/1, succesor/2, image/1, target/0, pred1/1, pred2/2\}$ and constants,
- use program template – generate all possible predicates,
- assign weights to the predicates,
- backpropagate to predict the right labels $\in \{0, 1\}$.

R. Evans et al. "Learning Explanatory Rules from Noisy Data". JAIR, 2018.

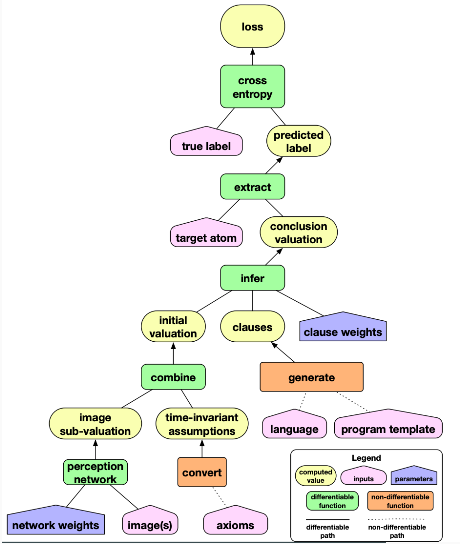
NNs in general – ∂ILP – an example



- R. Evans et al. "Learning Explanatory Rules from Noisy Data". JAIR, 2018.

- Make initial valuation – from the images and axioms,
- use the language from the predicates $\{zero/1, succesor/2, image/1, target/0, pred1/1, pred2/2\}$ and constants,
- use program template – generate all possible predicates,
- assign weights to the predicates,
- backpropagate to predict the right labels $\in \{0, 1\}$.

NNs in general – ∂ILP – an example



▸ R. Evans et al. "Learning Explanatory Rules from Noisy Data". JAIR, 2018.

- Make initial valuation – from the images and axioms,
- use the language from the predicates $\{zero/1, succesor/2, image/1, target/0, pred1/1, pred2/2\}$ and constants,
- use program template – generate all possible predicates,
- assign weights to the predicates,
- backpropagate to predict the right labels $\in \{0, 1\}$.

Solution:

$$target() \leftarrow image(X), pred1(X)$$
$$pred1(X) \leftarrow zero(X)$$
$$pred1(X) \leftarrow succesor(Y, X), pred2(Y)$$
$$pred2(Y) \leftarrow succesor(Y, X), pred1(Y)$$

Solution:

$$target() \leftarrow image(X), pred1(X)$$
$$pred1(X) \leftarrow zero(X)$$
$$pred1(X) \leftarrow succesor(Y, X), pred2(Y)$$
$$pred2(Y) \leftarrow succesor(Y, X), pred1(Y)$$

NNs in general – summary

1. (MILP) encoding of NN helps to explain but not to train,
2. use MILP, SMT solver or Abduction with encoding,
3. generate explainable programs from noisy data with ∂ILP.

NNs in general – summary

1. (MILP) encoding of NN helps to explain but not to train,
2. use MILP, SMT solver or Abduction with encoding,
3. generate explainable programs from noisy data with ∂ILP.

NNs in general – summary

1. (MILP) encoding of NN helps to explain but not to train,
2. use MILP, SMT solver or Abduction with encoding,
3. generate explainable programs from noisy data with ∂ILP.

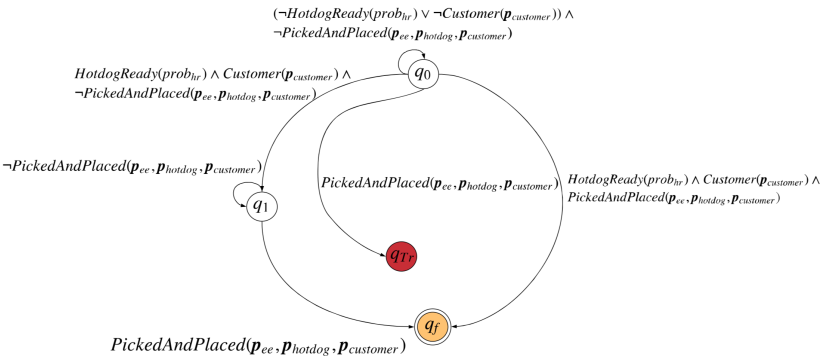
Reinforcement learning (RL) & FM

RL – finite-state predicate automaton (FSPA)

Q: How to combine **automaton**, truncated linear temporal logic (**TLTL**), and **neural networks-based approaches** usefully and apply them to RL?

RL – finite-state predicate automaton (FSPA)

Q: How to combine **automaton**, truncated linear temporal logic (**TLTL**), and **neural networks-based approaches** usefully and apply them to RL?



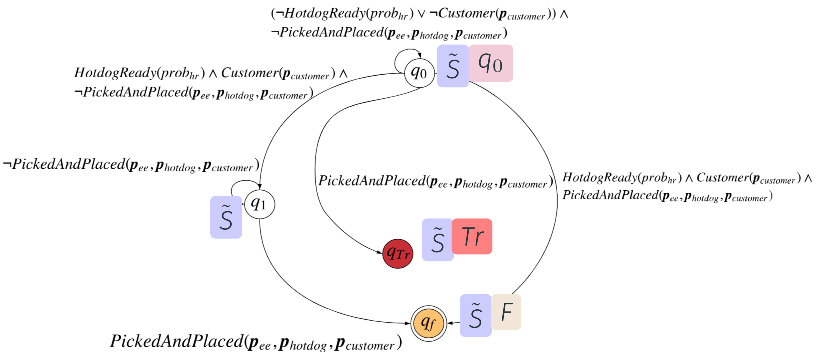
X. Li et al. "A formal methods approach to interpretable reinforcement learning for robotic planning". Science Robotics, 2019.

A: Use an FSPA-augmented MDP!

$$ \tag{3} $$

A: Use an FSPA-augmented MDP!

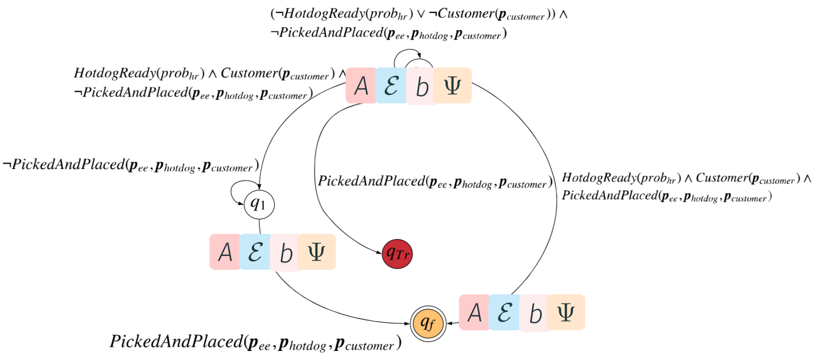
$$(\tilde{S}, q_0, F, Tr) \tag{3}$$



X. Li et al. “A formal methods approach to interpretable reinforcement learning for robotic planning”. Science Robotics, 2019.

A: Use an FSPA-augmented MDP!

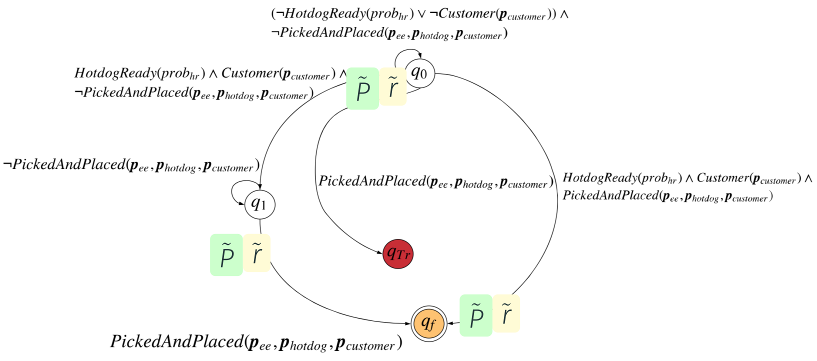
$$(\tilde{S}, A, \mathcal{E}, \Psi, q_0, b, F, Tr) \quad (3)$$



X. Li et al. “A formal methods approach to interpretable reinforcement learning for robotic planning”. Science Robotics, 2019.

A: Use an FSPA-augmented MDP!

$$(\tilde{S}, A, \tilde{P}, \tilde{r}, \mathcal{E}, \Psi, q_0, b, F, Tr) \tag{3}$$



X. Li et al. "A formal methods approach to interpretable reinforcement learning for robotic planning". Science Robotics, 2019.

RL – FSPA – problem statement

Input:

- robotic system with states S and actions A,
- TLTL task ϕ_{task} over S,
- safety requirements,
- knowledge base K.

Output:

- trajectory $s_{0:T}$ that satisfies ϕ_{task} and K.

RL – FSPA – problem statement

Input:

- robotic system with states S and actions A,
- TLTL task ϕ_{task} over S,
- safety requirements,
- knowledge base K.

Output:

- trajectory $s_{0:T}$ that satisfies ϕ_{task} and K.

RL – FSPA – problem statement

Input:

- robotic system with states S and actions A,
- TLTL task ϕ_{task} over S,
- safety requirements,
- knowledge base K.

Output:

- trajectory $s_{0:T}$ that satisfies ϕ_{task} and K.

RL – FSPA – problem statement

Input:

- robotic system with states S and actions A,
- TLTL task ϕ_{task} over S,
- safety requirements,
- knowledge base K.

Output:

- trajectory $s_{0:T}$ that satisfies ϕ_{task} and K.

RL – FSPA – problem statement

Input:

- robotic system with states S and actions A,
- TLTL task ϕ_{task} over S,
- safety requirements,
- knowledge base K.

Output:

- trajectory $s_{0:T}$ that satisfies ϕ_{task} and K.

RL – FSPA – an example

Q: How to serve a hotdog?

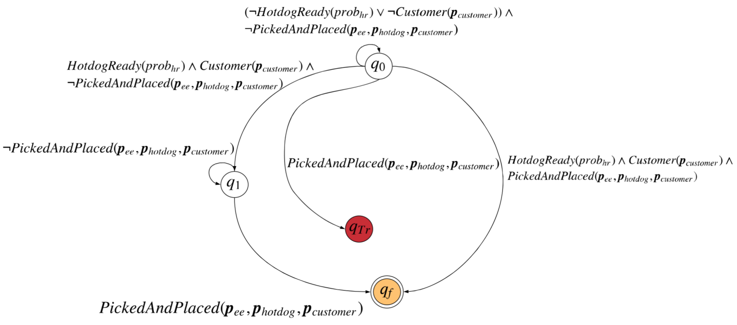
Q: How to serve a hotdog?

$$
\begin{gathered}
\mathit{HotdogServed}(s) = \mathcal{F}\mathit{PickedAndPlaced}(p_{endPose}, p_{hotdog}, p_{customer}) \wedge \\
(\neg \mathit{PickedAndPlaced}(p_{endPose}, p_{hotdog}, p_{customer}) \mathcal{U} \\
(\mathit{HotdogReady}(prob_{hr}) \wedge \mathit{Customer}(p_{customer})))
\end{gathered}
$$

RL – FSPA – an example

Q: How to serve a hotdog?

$$HotdogServed(s) = \mathcal{F} PickedAndPlaced(p_{endPose}, p_{hotdog}, p_{customer}) \wedge$$
$$(\neg PickedAndPlaced(p_{endPose}, p_{hotdog}, p_{customer}) \mathcal{U}$$
$$(HotdogReady(prob_{hr}) \wedge Customer(p_{customer})))$$



X. Li et al. "A formal methods approach to interpretable reinforcement learning for robotic planning". Science Robotics, 2019.

RL – FSPA – summary

1. Write the task as a temporal predicate – TLTL,
2. use TLTL and knowledge base to build FSPA,
3. adapt reward and transition functions of MDP to consider FSPA.

RL – FSPA – summary

1. Write the task as a temporal predicate – TLTL,
2. use TLTL and knowledge base to build FSPA,
3. adapt reward and transition functions of MDP to consider FSPA.

RL – FSPA – summary

1. Write the task as a temporal predicate – TLTL,
2. use TLTL and knowledge base to build FSPA,
3. adapt reward and transition functions of MDP to consider FSPA.

Methods that try to generalize & FM

Methods that try to generalize

Q: Can FM generalize across ML methods in explainability?

Methods that try to generalize

Q: Can FM generalize across ML methods in explainability?
A: Yes!

Methods that try to generalize

Q: Can FM generalize across ML methods in explainability?
A: Yes!

We focus on 2 methods:

Methods that try to generalize

Q: Can FM generalize across ML methods in explainability?
A: Yes!

We focus on 2 methods:

- Abduction and counterexamples,
- Markov logic networks (MLN)

Methods that try to generalize – Abduction

Compute an explanation with guarantee.

$$
\begin{array}{l}
\textbf{foreach } l \in C \textbf{ do} \\
\quad \textbf{if } \mathcal{M}, \mathcal{C} \backslash \{l\} \models (\mathcal{F} \rightarrow \mathcal{E}) \textbf{ then} \\
\quad \quad C \leftarrow C \backslash \{l\} \\
\quad \textbf{end} \\
\textbf{end}
\end{array}
$$

- Encode ML model with prediction $\mathcal{E}$ into formula $\mathcal{F}$,
- take initial data C for the prediction $\mathcal{E}$,
- get the minimal subset C_m s.t. $\mathcal{F} \wedge C_m \models \mathcal{E}$.

▸ A. Ignatiev et al. "Abduction-Based Explanations for Machine Learning Models". AAAI, 2019.

Methods that try to generalize – Abduction

Compute an explanation with guarantee.

foreach $l \in C$ **do**
  if $\mathcal{M}, \mathcal{C} \backslash \{l\} \models (\mathcal{F} \rightarrow \mathcal{E})$ **then**
    $C \leftarrow C \backslash \{l\}$
  end
end

- Encode ML model with prediction $\mathcal{E}$ into formula $\mathcal{F}$,
- take initial data C for the prediction $\mathcal{E}$,
- get the minimal subset C_m s.t. $\mathcal{F} \wedge C_m \models \mathcal{E}$.

▸ A. Ignatiev et al. “Abduction-Based Explanations for Machine Learning Models”. AAAI, 2019.

Methods that try to generalize – Abduction

Compute an explanation with guarantee.

foreach $l \in C$ **do**
  if $\mathcal{M}, \mathcal{C} \backslash \{l\} \models (\mathcal{F} \rightarrow \mathcal{E})$ **then**
    $C \leftarrow C \backslash \{l\}$
  end
end

- Encode ML model with prediction $\mathcal{E}$ into formula $\mathcal{F}$,
- take initial data C for the prediction $\mathcal{E}$,
- get the minimal subset C_m s.t. $\mathcal{F} \wedge C_m \models \mathcal{E}$.

▸ A. Ignatiev et al. "Abduction-Based Explanations for Machine Learning Models". AAAI, 2019.

Methods that try to generalize – Abduction

Compute an explanation with guarantee.

foreach $l \in C$ **do**
  if $\mathcal{M}, \mathcal{C}\backslash\{l\} \models (\mathcal{F} \rightarrow \mathcal{E})$ **then**
    $C \leftarrow C\backslash\{l\}$
  end
end

- Encode ML model with prediction $\mathcal{E}$ into formula $\mathcal{F}$,
- take initial data C for the prediction $\mathcal{E}$,
- get the minimal subset C_m s.t. $\mathcal{F} \wedge C_m \models \mathcal{E}$.

▸ A. Ignatiev et al. "Abduction-Based Explanations for Machine Learning Models". AAAI, 2019.

Methods that try to generalize – Counterexamples

Same principle,

Methods that try to generalize – Counterexamples

Same principle,

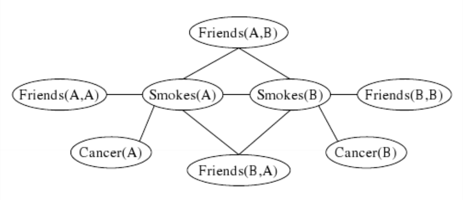
generate counterexamples and explanations simultaneously.

Methods that try to generalize – MLN

Relational reasoning is a central component of generally intelligent behavior, but has proven difficult for neural networks to learn.

("A simple neural network module for relational reasoning". NeurIPS, 20

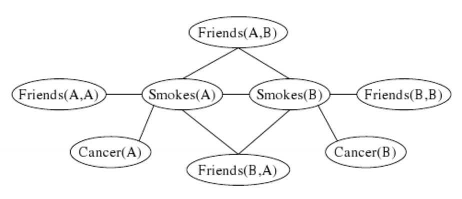
Methods that try to generalize – MLN and graph neural networks (GNN)



- Take Markov network,
- over first-order knowledge base,
- learn the weights via GNN.

▸ https://images.app.goo.gl/KCkz3jFqPVePAp298

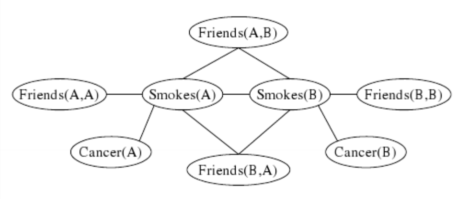
Methods that try to generalize – MLN and graph neural networks (GNN)



- Take Markov network,
- over first-order knowledge base,
- learn the weights via GNN.

▸ https://images.app.goo.gl/KCkz3jFqPVePAp298

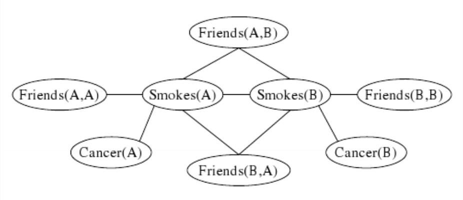
Methods that try to generalize – MLN and graph neural networks (GNN)



- Take Markov network,
- over first-order knowledge base,
- learn the weights via GNN.

▸ https://images.app.goo.gl/KCkz3jFqPVePAp298

Methods that try to generalize – MLN and graph neural networks (GNN)



- Take Markov network,
- over first-order knowledge base,
- learn the weights via GNN.

▶ https://images.app.goo.gl/KCkz3jFqPVePAp298

Methods that try to generalize – summary

1. Use Abduction to get explanations with guarantees,
2. use counterexample-based method – get counterexamples and explanations,
3. for relational learning – learn weights of MLN with GNN.

Methods that try to generalize – summary

1. Use Abduction to get explanations with guarantees,
2. use counterexample-based method – get counterexamples and explanations,
3. for relational learning – learn weights of MLN with GNN.

Methods that try to generalize – summary

1. Use Abduction to get explanations with guarantees,
2. use counterexample-based method – get counterexamples and explanations,
3. for relational learning – learn weights of MLN with GNN.

FM and ML in XAI – take-home message

1. For specialized domains – domain-specific methods (NSM in CV, FSPA in RL),
2. for NNs in general – use encodings or ∂ILP,
3. there are ways to generalize across models – Abduction & Counterexamples.

FM and ML in XAI – take-home message

1. For specialized domains – domain-specific methods (NSM in CV, FSPA in RL),
2. for NNs in general – use encodings or ∂ILP,
3. there are ways to generalize across models – Abduction & Counterexamples.

FM and ML in XAI – take-home message

1. For specialized domains – domain-specific methods (NSM in CV, FSPA in RL),
2. for NNs in general – use encodings or ∂ILP,
3. there are ways to generalize across models – Abduction & Counterexamples.