(Towards) Deep Rule Learning

Johannes Fürnkranz
Johannes Kepler University, Linz
Institute for Applied Knowledge Processing
Computational Data Analytics Group

juffi@faw.jku.at

Joint Work with **Florian Beck**

The Sucess of Deep Learning

JOHANNES KEPLER UNIVERSITÄT LINZ

- **Hypothesis:**
  Most of the success of deep learning is due to the fact that it allows to learn **deep structures** in which auxiliary concepts develop which will facilitate the learning process

- **Problem:**
  No state-of-the-art rule learning algorithm is able to learn such structured, purely declarative rule bases

Example: Parity / XOR

- Consider the parity / XOR problem
  - $n + r$ binary attributes sampled with an equal distribution of 0/1
  - n relevant binary attributes (the first n w.l.o.g.)
  - r irrelevant binary attributes

- Target concept:
  - is there an even number of 1's in the relevant attributes?

Encoding Parity with a Flat Rule Set

Most rule learning algorithms learn flat theories

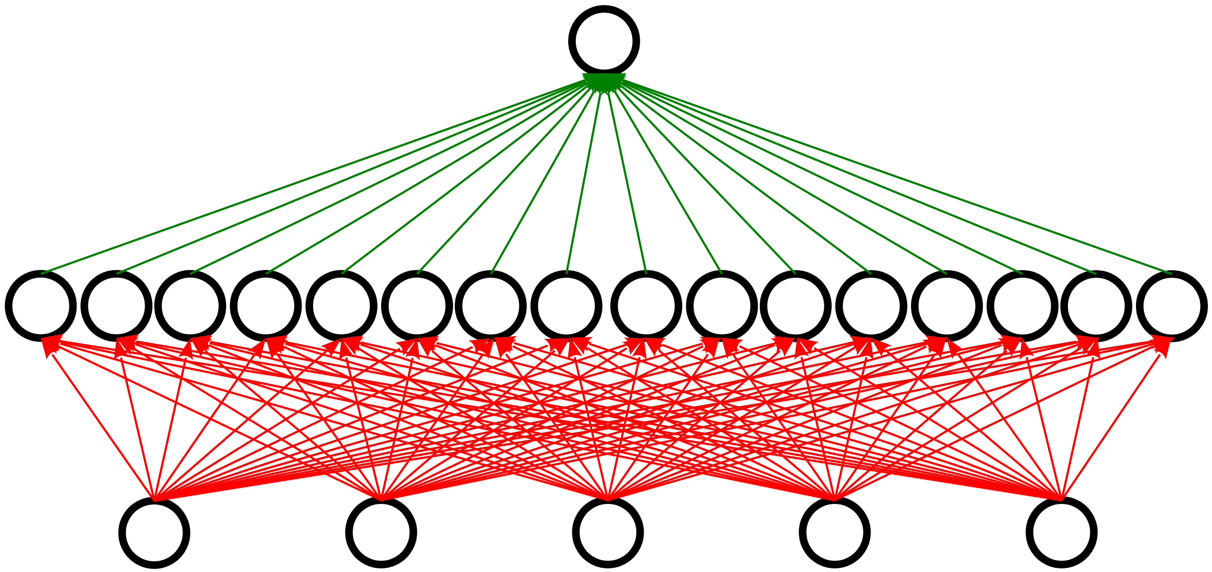
- n-bit parity needs 2^{n-1} flat rules
- each rule encoding one positive case in the truth table

```
parity :-     x1,     x2,     x3,     x4, not x5.
parity :-     x1,     x2, not x3, not x4, not x5.
parity :-     x1, not x2,     x3, not x4, not x5.
parity :-     x1, not x2, not x3,     x4, not x5.
parity :- not x1,     x2, not x3,     x4, not x5.
parity :- not x1,     x2,     x3, not x4, not x5.
parity :- not x1, not x2,     x3,     x4, not x5.
parity :- not x1, not x2, not x2, not x4, not x5.
parity :-     x1,     x2,     x3, not x4,     x5.
parity :-     x1,     x2, not x3,     x4,     x5.
parity :-     x1, not x2,     x3,     x4,     x5.
parity :- not x1,     x2,     x3,     x4,     x5.
parity :- not x1, not x2, not x3,     x4,     x5.
parity :- not x1, not x2,     x3, not x4,     x5.
parity :- not x1,     x2, not x3, not x4,     x5.
parity :-     x1, not x2, not x2, not x4,     x5.
```

DNF formula with 2^{n-1} literals, each having n variables

Network View of a Flat Rule Set

- Flat Rule Sets can be converted into a network using a single AND and a single OR layer (analogous to Sum-Product Networks)



- Each node in the hidden layer corresponds to one rule
  - typically it is a local pattern, covering part of the target

Encoding Parity with a Structured Rule Base

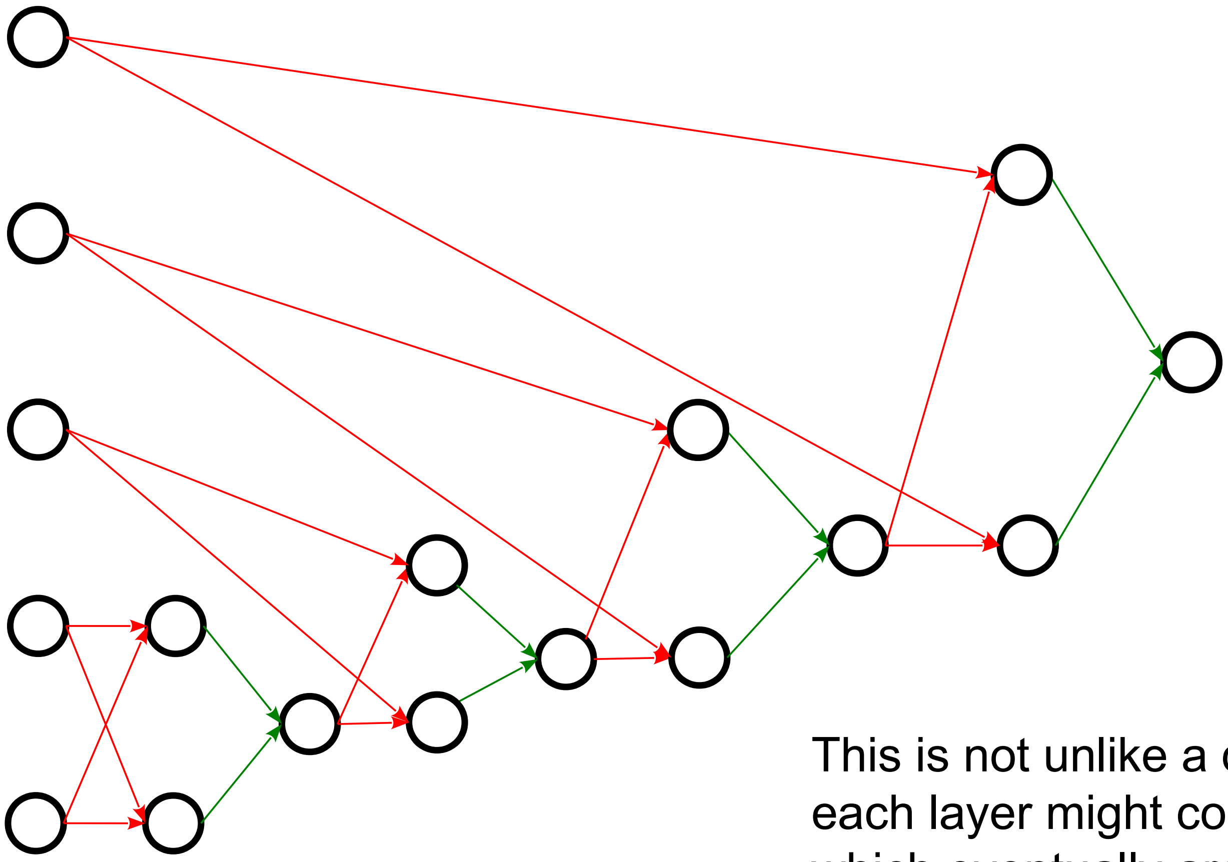
But structured concepts are often more interpretable

- in parity we need only $O(n)$ rules with intermediate concepts

```
parity45    :-     x4,     x5.
parity45    :- not x4, not x5.

parity345   :-     x3, not parity45.
parity345   :- not x3,     parity45.

parity2345  :-     x2, not parity345.
parity2345  :- not x2,     parity345.

parity      :-     x1, not parity2345.
parity      :- not x1,     parity2345.
```

Network View of a Structured Rule Base

- This is encodes a deep network structure



This is not unlike a deep network:
each layer might contain more nodes,
which eventually are not needed

Why is it good to learn structured rule bases?

JOHANNES KEPLER UNIVERSITÄT LINZ

- **Expressivity?** It does not necessarily increase expressivity
  - any structured rule base can be converted into an equivalent DNF expression, i.e., a flat set of rules
  - but this is also true for NNs → universal approximation theorem (one layer is sufficient; Hornik et al. 1989)
  - in both cases the number of terms (size of hidden layers, conjuncts in the DNF) is unbounded
- **Learning Efficiency**
  - the hope is that deeper structures might be easier to learn
  - possibly contain fewer “parameters” that need to the found
- **Explicit encoding of the decision function**
  - Note that conventional rule learning algorithms rely on additional mechanisms for tie breaking if more than one (or no) rule fires

Some Research Questions

- **Representation:** How to represent deep rule sets to allow for efficient and effective reasoning and learning

- **Learning efficiency:** Are deep rule structures easier to learn than shallow DNF rule sets?

- **Restructuring:** Can we structure an existing (shallow) rule sets into a comprehensible deep rule sets?

- **Learning:** How can we learn deep rule sets?

Rule Sets

- are typically **not declarative**, require some sort of **tie breaking**
- two main approaches
  - **weighted rules** / probabilistic rules

$$\begin{array}{l}
\boldsymbol{r}_1(0.8) : a \wedge b \rightarrow x \\
\boldsymbol{r}_2(0.9) : b \wedge c \rightarrow y \\
\boldsymbol{r}_3(0.7) : c \wedge d \rightarrow x \\
\boldsymbol{d} : \qquad\qquad \rightarrow z
\end{array}$$

max: y (0.9)

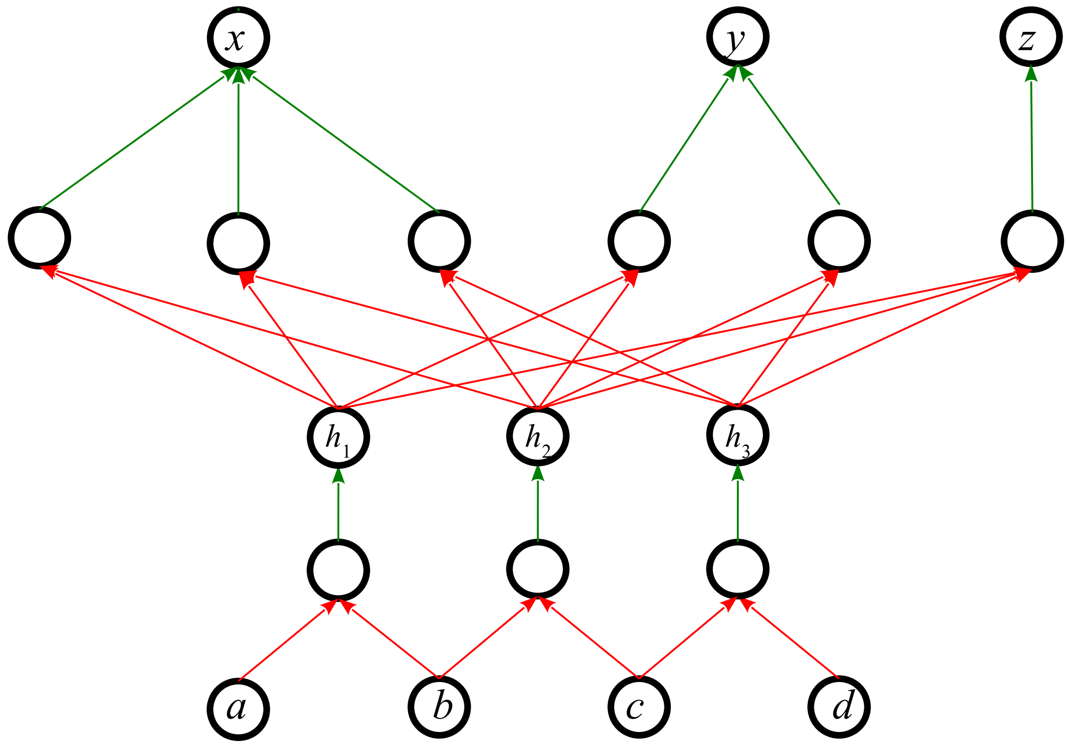
sum: x (0.7+0.8 > 0.9)

  - **decision lists** $\mathcal{D} = (\boldsymbol{r}_2, \boldsymbol{r}_1, \boldsymbol{r}_3, \boldsymbol{d})$
    - sort the rules according to some criterion
      - e.g., order in which they are learned
      - e.g., order according to weight (effectively equivalent to using weighted max)
    - use the first rule that fires

Declarative Version of Weighted Rule Sets

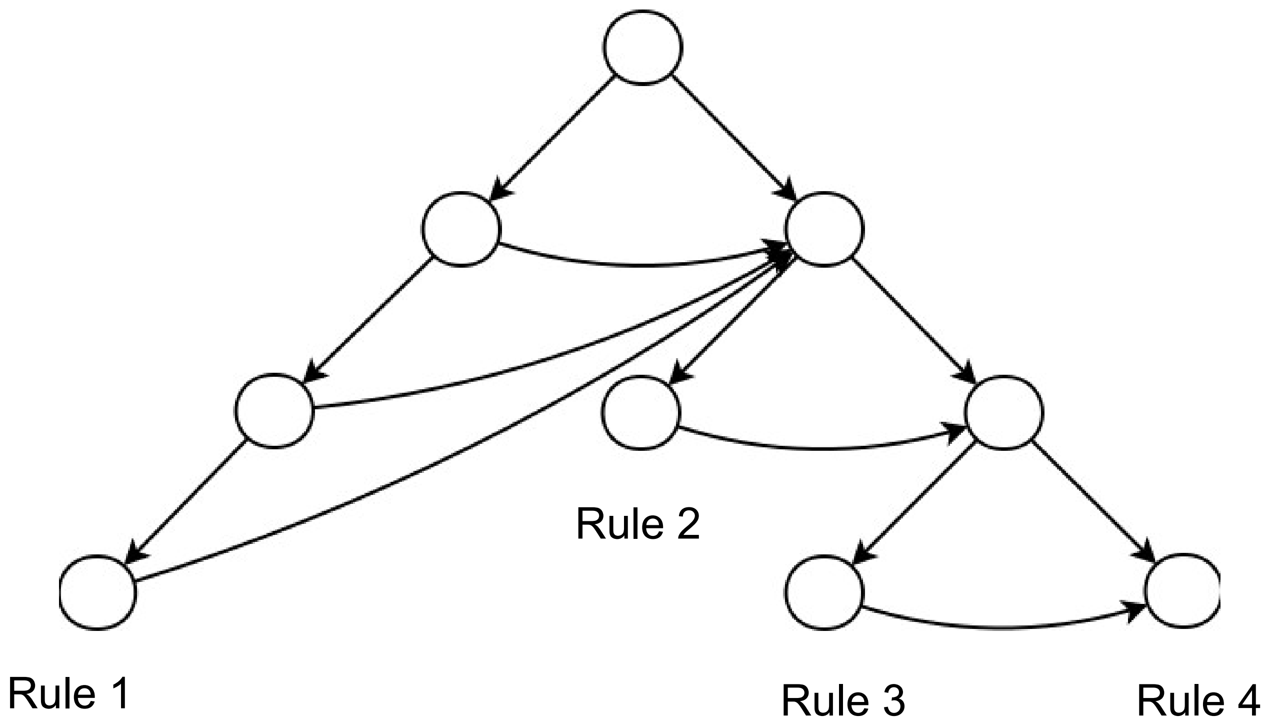
- Tie Breaking with Majority vote

$$
\begin{array}{r}
\hline
a \wedge b \rightarrow h_1 \\
b \wedge c \rightarrow h_2 \\
c \wedge d \rightarrow h_3 \\
h_1 \wedge h_3 \rightarrow x \\
h_1 \wedge \neg h_2 \rightarrow x \\
h_3 \wedge \neg h_2 \rightarrow x \\
h_2 \wedge \neg h_1 \rightarrow y \\
h_2 \wedge \neg h_3 \rightarrow y \\
\neg h_1 \wedge \neg h_2 \wedge \neg h_3 \rightarrow z \\
\hline
\end{array}
$$



Declarative Version of Decision List

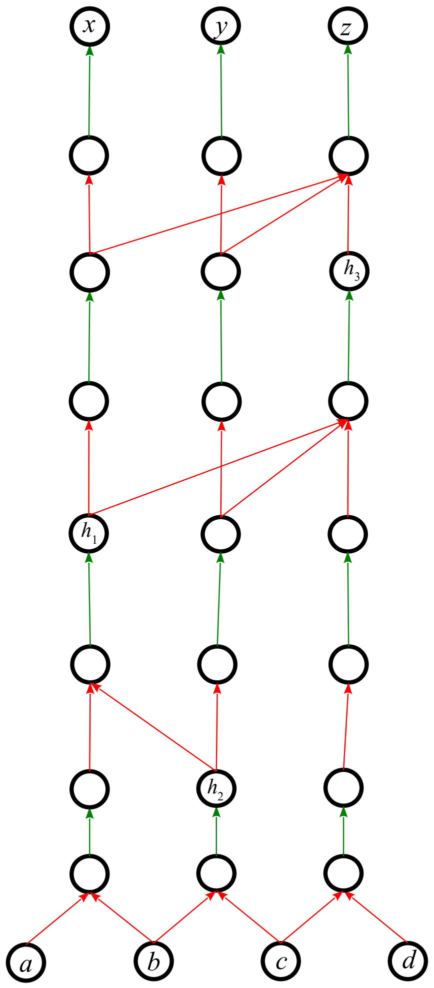
- A decision list is a decision graph, where not satisfied condition takes you to the start of the next rule
- Example of a decision list with 4 rules with 4, 2, 2, 1 conditions



Declarative Version of Decision List

- In our example

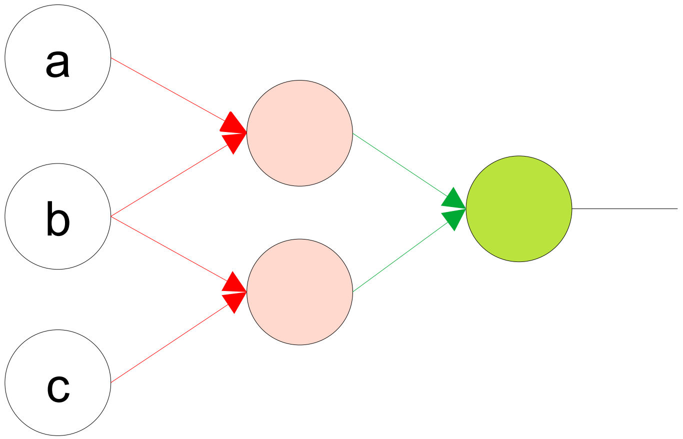
$$
\begin{array}{r}
\hline
b \wedge c \rightarrow h_2 \\
h_2 \rightarrow y \\
\neg h_2 \wedge a \wedge b \rightarrow h_1 \\
h_1 \rightarrow x \\
\neg h_1 \wedge \neg h_2 \wedge c \wedge d \rightarrow h_3 \\
h_3 \rightarrow x \\
\neg h_1 \wedge \neg h_2 \wedge \neg h_3 \rightarrow z \\
\hline
\end{array}
$$



NAND Representation

- Like any other Boolean function, AND/OR networks can be represented in a uniform way with single node types (NAND)
- Example:

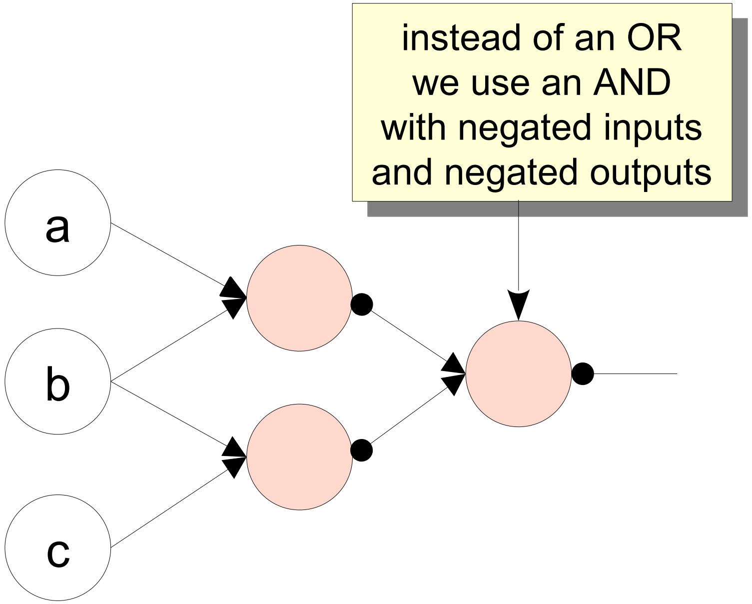
```
x :- a, b.
x :- b, c.

x = (a & b) | (b & c)

!x = !((a & b) | (b & c))
   = !(a & b) & !(b & c)

x = !(!(a&b) & !(b&c)) =
    (a NAND b) NAND (b NAND c)
```



NAND Representation

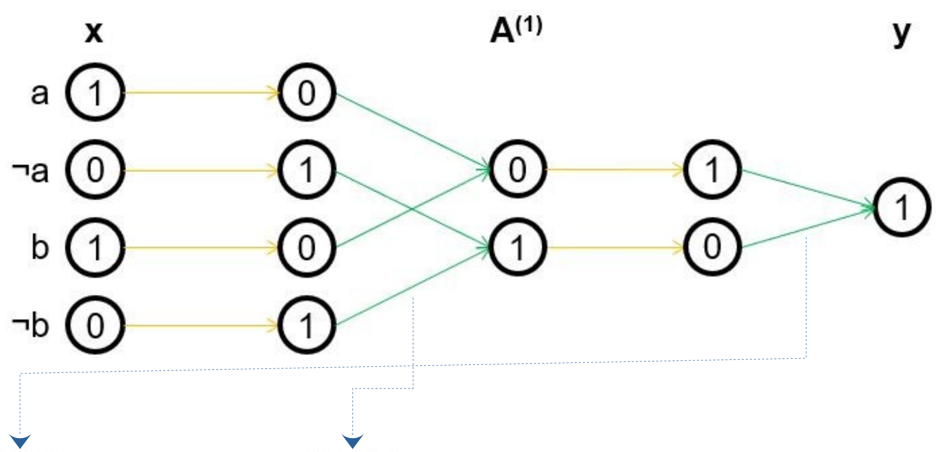
- Like any other Boolean function, AND/OR networks can be represented in a uniform way with single node types (NAND)
- Example:

```
x :- a, b.
x :- b, c.

x = (a & b) | (b & c)

!x = !((a & b) | (b & c))
   = !(a & b) & !(b & c)

x = !(!(a&b) & !(b&c)) =
    (a NAND b) NAND (b NAND c)
```



NAND Representation and Boolean Matrix Multiplication

- The NAND representation also allows for an effective representation as matrix multiplication



$$\overset{W^{(1)}}{1 \quad 1} \; @ \; \neg \left(\overset{W^{(0)}}{\begin{matrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{matrix}} \; @ \; \neg \overset{x}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}} \right) = \overset{y}{1}$$

(orange/¬: negation, green/@: binary matrix multiplication)

Some Research Questions

- **Representation:** How to represent deep rule sets to allow for efficient and effective reasoning and learning

- **Learning efficiency:** Are deep rule structures easier to learn than shallow DNF rule sets?

- **Restructuring:** Can we structure an existing (shallow) rule sets into a comprehensible deep rule sets?

- **Learning:** How can we learn deep rule sets?

Does a Deep Structure help?

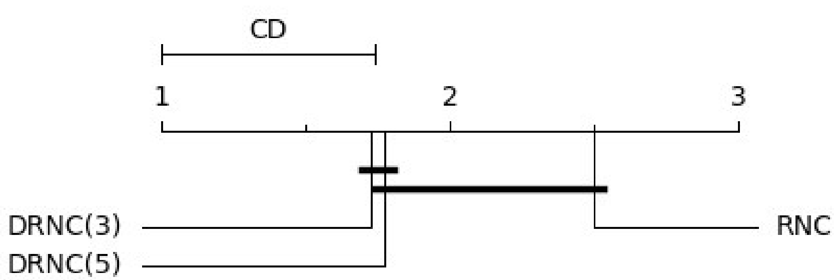
- To answer this empirically, we need to **compare** a powerful **shallow** rule learner with a powerful **deep** rule learner
  - But we do not have a powerful deep rule learner… (yet)
- Instead, we use a **simple optimization algorithm** to learn both, deep and shallow representations

1) Fix a network architecture
   - Shallow, single layer network RNC: [20]
   - Deep 3-layer network DRNC(3): [32, 8, 2]
   - Deep 5-layer network DRNC(5): [32, 16, 8, 4, 2]
2) Initialize Boolean weights probabilistically
3) Use stochastic local search to find best weight „flip“ on a mini-batch of data until convergence
4) Optimize finally on whole training set

Results on Artificial Datasets

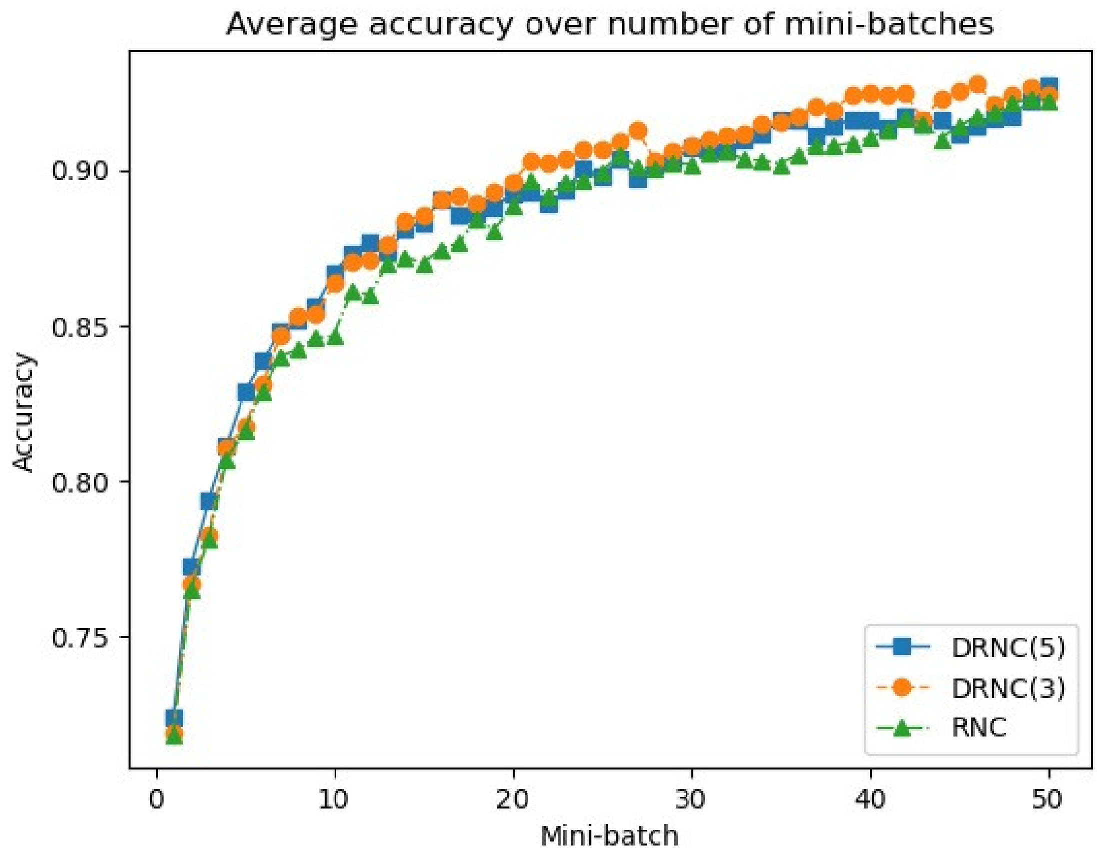
- 20 artificial datasets with 10 Boolean inputs, 1 Boolean output
  - generated from a randomly initialized (deep) Boolean network

seed %(+)	DRNC(5)	DRNC(3)	RNC	RIPPER	CART
Ø Accuracy	0.9467	0.9502	0.9386	0.9591	*0.9644*
Ø Rank	1.775	1.725	2.5		



- DRNC(3) [DRNC(5)] outperforms RNC on a significance level of more than 95% [90%]

Run-times (Artificial Datasets)



- DRNC(3) and DRNC(5) converge faster than RNC

Results on Real-World (UCI) Datasets

JOHANNES KEPLER
UNIVERSITÄT LINZ

dataset	%(+)	DRNC(5)	DRNC(3)	RNC	RIPPER	CART
car-evaluation	0.7002	0.8999	**0.9022**	0.8565	*0.9838*	0.9821
connect-4	0.6565	**0.7728**	0.7712	0.7597	0.7475	*0.8195*
kr-vs-kp	0.5222	0.9671	0.9643	**0.9725**	0.9837	*0.989*
monk-1	0.5000	***1***	0.9982	0.9910	0.9478	0.8939
monk-2	0.3428	0.7321	**0.7421**	0.7139	0.6872	*0.7869*
monk-3	0.5199	**0.9693**	0.9603	0.9567	0.9386	*0.9729*
mushroom	0.784	***1***	0.978	0.993	0.9992	*1*
tic-tac-toe	0.6534	0.8956	0.9196	**0.9541**	*1*	0.9217
vote	0.6138	***0.9655***	0.9288	0.9264	0.9011	0.9287
Ø Rank		1.556	2	2.444		

- DRNC(5) has the best performance on these real-world datasets, followed by DRNC(3)

Some Research Questions

JOHANNES KEPLER
UNIVERSITÄT LINZ

- **Representation:** How to represent deep rule sets to allow for efficient and effective reasoning and learning

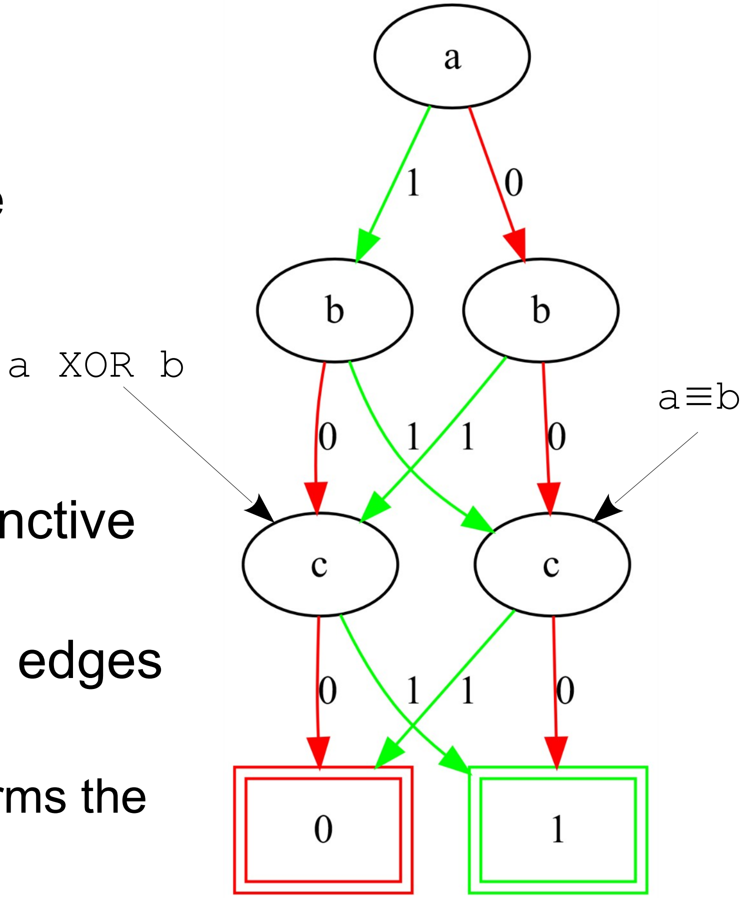
- **Learning efficiency:** Are deep rule structures easier to learn than shallow DNF rule sets?

- **Restructuring:** Can we structure an existing (shallow) rule sets into a comprehensive deep rule sets?

- **Learning:** How can we learn deep rule sets?

Binary Decision Diagrams

- Binary Decision Diagrams (BDDs) are a special case of decision graphs
- form a (binary) DAG instead of a tree
  - there might be different paths to the same decision node
  - these correspond to disjunctions
- Rule Extraction from a BDD
  - Every path from root to leaf forms a conjunctive rule with target class as head
  - Every interior node with multiple incoming edges forms a disjunctive intermediate concept
    - e.g. the left c-node in the BDD to the right forms the the concept `a XOR b`



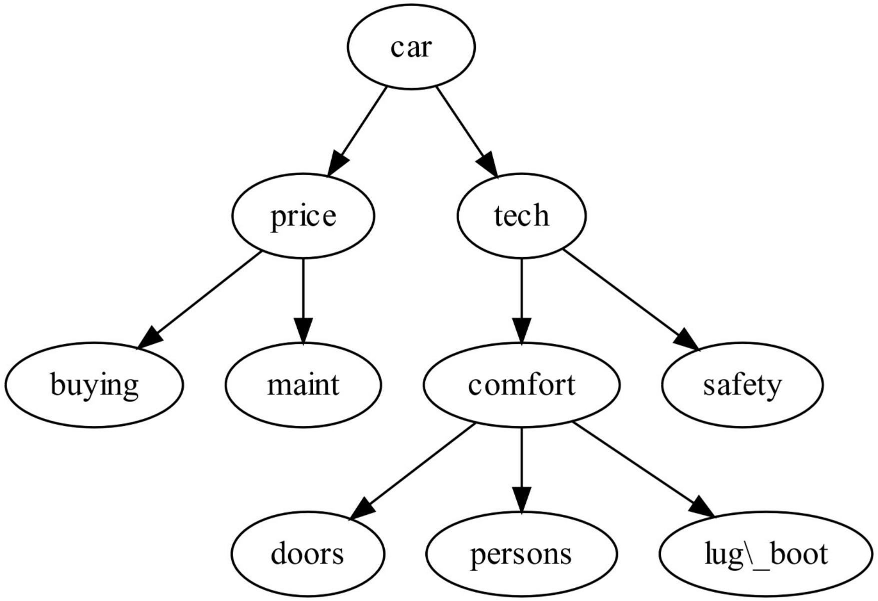
Structuring Rule Sets with BDDs

JOHANNES KEPLER
UNIVERSITÄT LINZ

- Inspired by multi-level logic optimization in electronic design automation

- Idea:
  - Take DNF description of positive class (or shallow rule set)
  - Convert DNF to BDD
  - Extract structured, deep rule set from BDD:
    - For each „join“ node with , define a rule for each incoming path, starting from the root or another „join“ node → disjunctive concepts
    - (optional) Detect overlapping conditions/paths → conjunctive concepts
    - (optional) Simplify rules with algebraic and boolean optimization

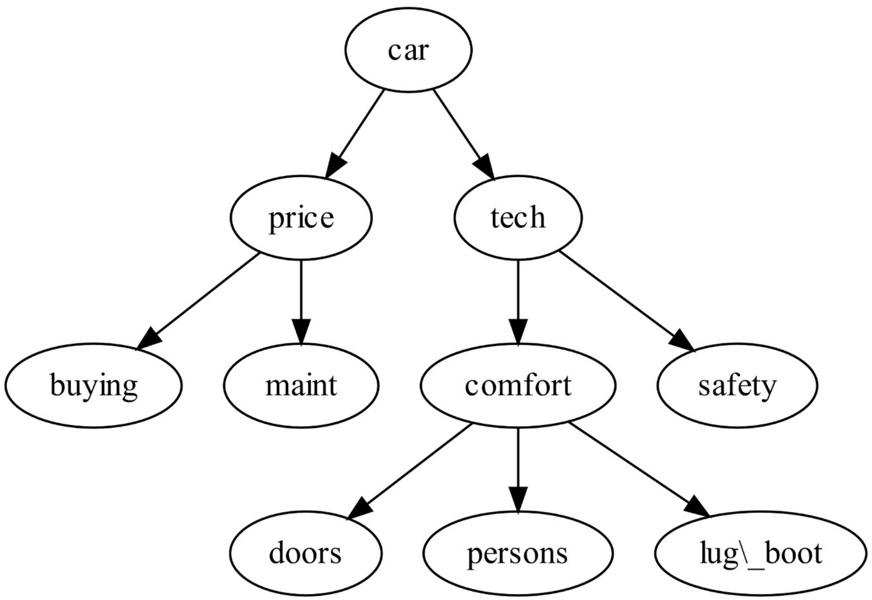
Example Domain: Car Sale



- I buy a car if the price is o.k. and the technical specs. are o.k.
- the price is o.k. if both the sales price and the maintenance costs are o.k.
- the technical specs. are o.k. if the car is comfortable and safe
- the car is comfortable if at least two of the following three are satisfied
  - has 4 doors
  - can seat 5 persons
  - has a luggage boot

Original dataset: Bohanec, Marko & Rajkovic, Vladislav. (1989). Knowledge-based explanation in multi-attribute decision making. Produktivnost.

Example Domain: Car Sale

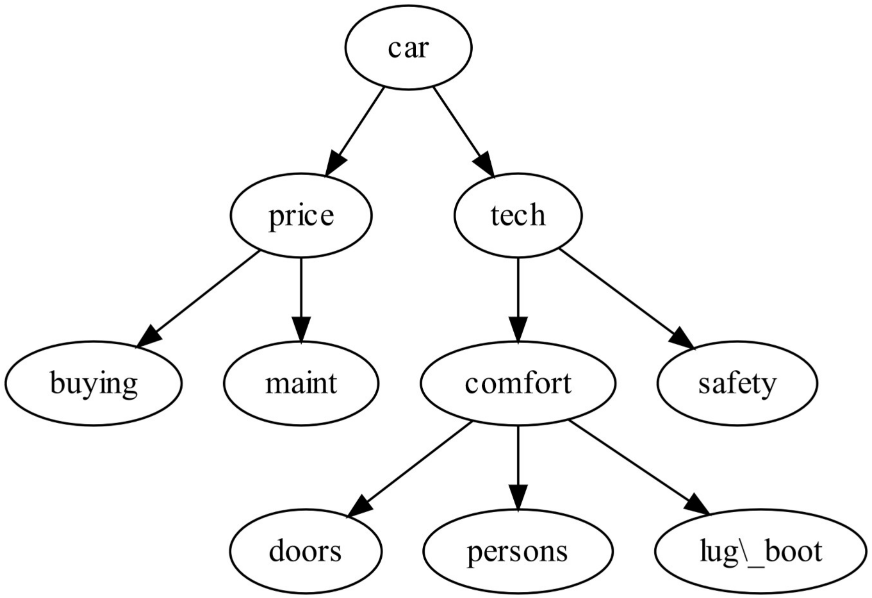


Deep rule set

```
car :-        price, tech.

price :-      buying.
price :-      maint.

tech :-       comfort.
tech :-       safety.

comfort :-    doors, persons.
comfort :-    doors, lug_boot.
comfort :-    persons, lug_boot.
```

Original dataset: Bohanec, Marko & Rajkovic, Vladislav. (1989). Knowledge-based explanation in multi-attribute decision making. Produktivnost.

Example Domain: Car Sale



Shallow rule set

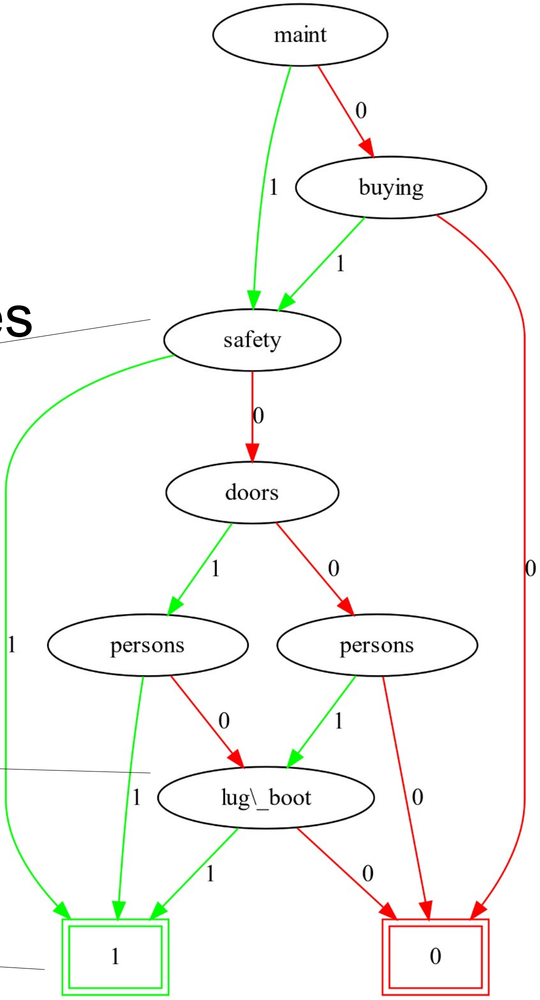
```
car :- buying, doors, persons.
car :- buying, doors, lug_boot.
car :- buying, persons,lug_boot.
car :- buying, safety.
car :- maint, doors, persons.
car :- maint, doors, lug_boot.
car :- maint, persons, lug_boot.
car :- maint, safety.
```

Original dataset: Bohanec, Marko & Rajkovic, Vladislav. (1989). Knowledge-based explanation in multi-attribute decision making. Produktivnost.

Example Domain: Car Sales

- Start with the flat rule set
- construct a BDD from this rule set
- Extract intermediate concepts from „join“ nodes

```
c1 :-  maint.
c1 :-  \+ maint, buying.

c2 :- c1, \+ safety, doors, \+ persons.
c2 :- c1, \+ safety, \+ doors, persons.

car :- c1, safety.
car :- c1, \+ safety, doors, persons.
car :- c2, lug_boot.
```



Graph: BDD Interface University of Utah, http://formal.cs.utah.edu:8080/pbl/BDD.php

Experiments Artificial Datasets

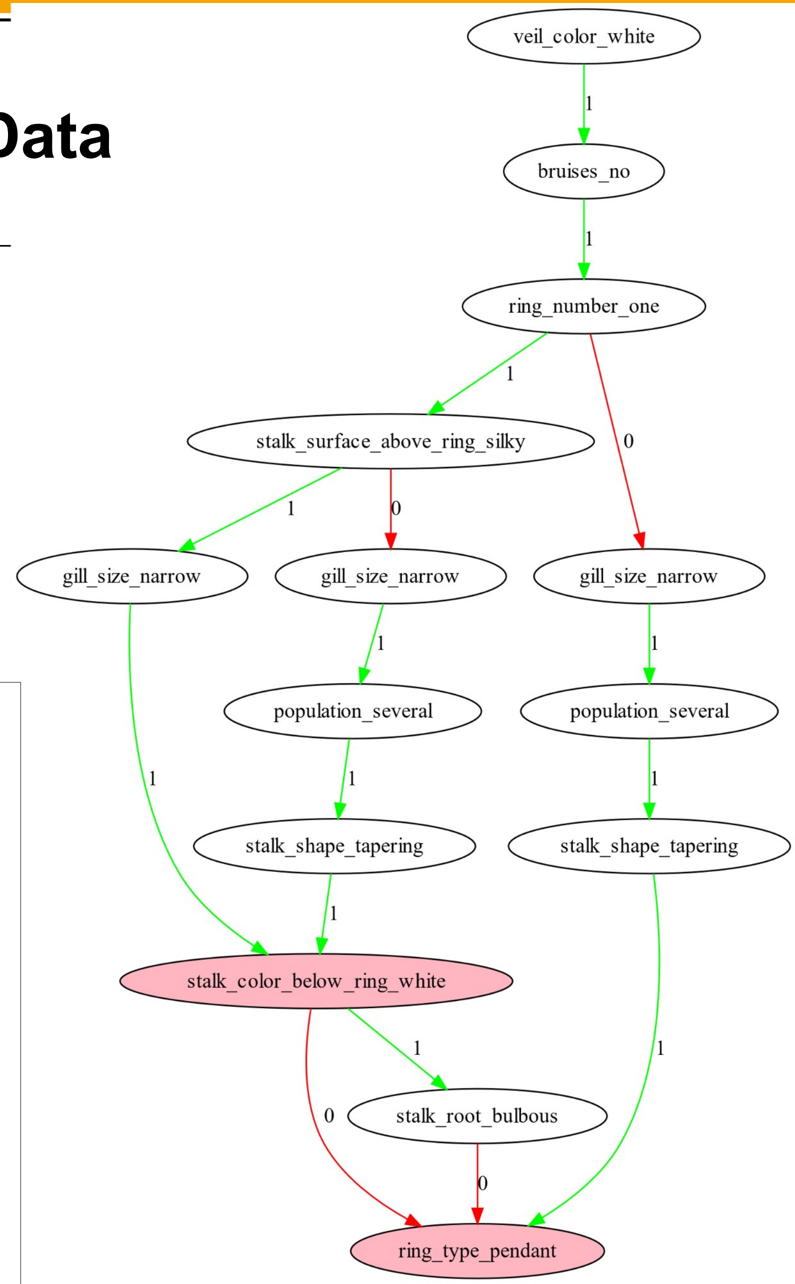
JOHANNES KEPLER
UNIVERSITÄT LINZ

- 10 Boolean inputs, 1 Boolean output
- Number of rules R, concepts C, and vertices V for the ground truth and our learner "LORD".
- $|X|$ denote values for DNF, and $|X'|$ denote values for BDD

	Ground truth						LORD					
seed	$\lvert R\rvert$	$\lvert R'\rvert$	$\lvert C\rvert$	$\lvert C'\rvert$	$\lvert V\rvert$	$\lvert V'\rvert$	$\lvert R\rvert$	$\lvert R'\rvert$	$\lvert C\rvert$	$\lvert C'\rvert$	$\lvert V\rvert$	$\lvert V'\rvert$
5	8	20	0	7	29	18	13	20	0	6	52	19
16	17	27	0	4	49	25	18	26	0	4	51	26
19	4	8	0	1	9	10	5	6	0	1	11	7
24	15	46	0	12	59	45	18	39	0	11	72	36
36	21	62	0	16	85	53	25	61	0	12	99	62
44	16	28	0	5	48	28	25	31	0	5	75	28
70	18	41	0	7	78	46	26	56	0	13	113	51
81	20	43	0	11	84	39	28	34	0	10	121	33
82	4	5	0	1	8	6	4	8	0	2	8	6
85	3	3	0	0	8	6	3	3	0	0	8	6
89	12	48	0	11	47	41	19	41	0	11	72	36
107	17	47	0	10	70	41	32	32	0	7	132	33
112	18	37	0	8	67	33	33	41	0	10	126	36
118	14	32	0	8	51	27	16	32	0	6	59	30
∅	13.95	29.45	0	6.60	51.15	28.40	17.65	29.45	0	6.55	65.70	27.60

Generally we get
- more intermediate subconcepts (obviously...)
- fewer nodes (more compact concepts)
- more rules (for defining the subconcepts)

Experiments on Mushroom Data



- Mushroom dataset from UCI repository
- Graph only shows part of generated BDD with two „join“ nodes

Rule learner	$\lvert R \rvert$	$\lvert R' \rvert$	$\lvert C \rvert$	$\lvert C' \rvert$	$\lvert V \rvert$	$\lvert V' \rvert$
q_{Lap} learner	7	57	0	12	35	46
h_{Lap} learner	11	13	0	1	13	13
LORD	8	12	0	2	10	10

- Many new concepts
- but restructured rule sets are neither more compact nor easier to interpret

Some Research Questions

- **Representation:** How to represent deep rule sets to allow for efficient and effective reasoning and learning

- **Learning efficiency:** Are deep rule structures easier to learn than shallow DNF rule sets?

- **Restructuring:** Can we structure an existing (shallow) rule sets into a comprehensible deep rule sets?

- **Learning:** How can we learn deep rule sets?

Some Research Questions

Exercise, left to the reader...

- **Learning:** How can we learn deep rule sets?

Conclusions

- There is some evidence that deep rule sets could facilitate rule learning
  - but no conclusive answer yet
- There is some evidence that structuring rule sets could yield more compact concept descriptions
  - again, needs confirmation on challenging real-world tasks

→ Deep Rule Learning is a promising topic for further research

- Challenges:
  - Efficient learning algorithms for training intermediate concepts
  - Learning bias for compact structured rule sets
  - Are structured rule sets more interpretable than unstructured rule sets?

References

JOHANNES KEPLER UNIVERSITÄT LINZ

- Beck F., Fürnkranz J.: An Empirical Investigation into Deep and Shallow Rule Learning. *Frontiers in Artificial Intelligence* 4, 2021. doi:10.3389/frai.2021.689398

- Beck F., Fürnkranz J.: Beyond DNF: First Steps towards Deep Rule Learning, in *Proceedings of the 21st Conference Information Technologies -- Applications and Theory (ITAT)*, series CEUR Workshop Proceedings, vol. 2962, CEUR-WS.org, pp. 61--68, 2021.
- Beck F., Fürnkranz J.; Huynh, V.Q.P.: Structuring Rule Sets Using Binary Decision Diagrams, in *Proceedings of the 5th International Joint Conference on Rules and Reasoning (RuleML+RR)*, Springer-Verlag, 2021.
- Beck F., Fürnkranz J.: An Investigation into Mini-Batch Rule Learning, in *Proceedings of the 2nd Workshop on Deep Continuous-Discrete Machine Learning (DeCoDeML)*, 2020.
- Fürnkranz J., Hüllermeier E., Loza Mencía E., Rapp M.: Learning Structured Declarative Rule Sets – A Challenge for Deep Discrete Learning, in *Proceedings of the 2nd Workshop on Deep Continuous-Discrete Machine Learning (DeCoDeML)*, 2020.