DeepRED – Rule Extraction from Deep Neural Networks

Jan Ruben Zilke, *Eneldo Loza Mencía*, Frederik Janssen
Knowledge Engineering Group, TU Darmstadt
j.zilke@mail.de
{eneldo,janssen}@ke.tu-darmstadt.de

Outline

Comprehension and Extraction from Neural Networks

DeepRED: Rule extraction from Deep Neural Networks

Experimental Results

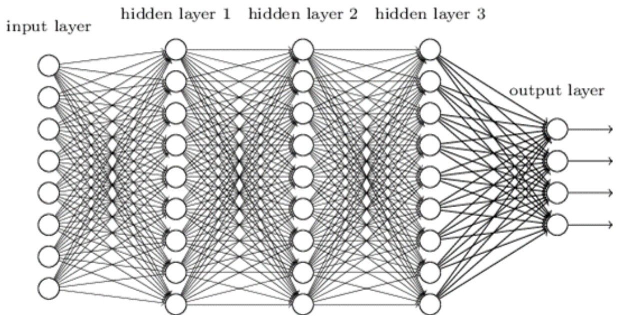
Conclusions

Comprehending Neural Networks

TECHNISCHE UNIVERSITÄT DARMSTADT

NNs are widely used for classification

- current hype about Deep Neural Networks (DNN)
- outperform previous state-of-the-art approaches in many domains
- DNNs might represent complex, abstract concepts in hidden nodes



Understanding how a NN comes to its decision is not trivial

- we only know the network's structure and its weights
- predictive model: usually NNs seen and used as a black box
- learned higher level concepts remain hidden
  - exception: visual domain

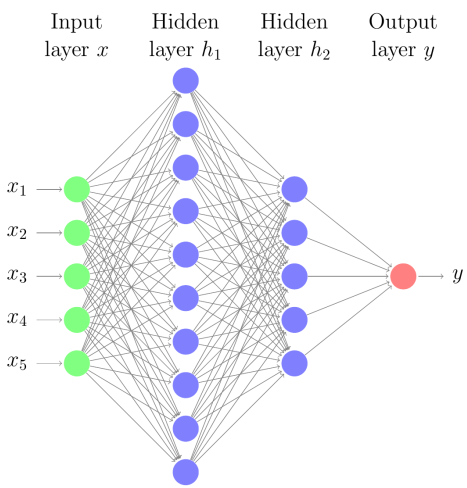
Comprehensible Decision Systems

TECHNISCHE UNIVERSITÄT DARMSTADT

Comprehensible description of a NN's behaviour sometimes essential

- safety critial domains, e.g. medicine, power stations, autonomous driving, financial markets

Solution: → **represent NN's behaviour as decision rules**

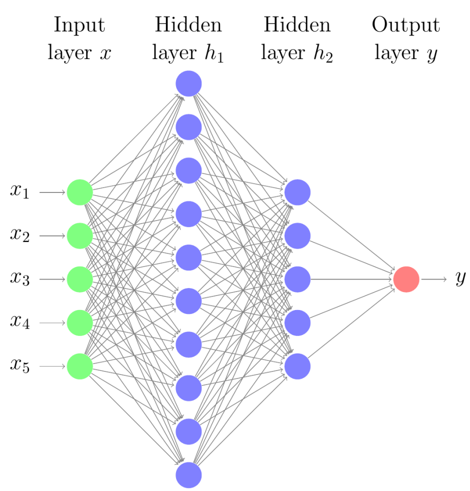



```
IF X1<0.5 AND X2>0.75 THEN OUT=1
IF X1>0.9 THEN OUT=1
IF X1>0.5 AND X1<0.9 AND X3>0.2 THEN OUT=1
IF X2>0.2 AND X3<0.5 AND X5<0.5 THEN OUT=1
IF X2>0.4 AND X3<0.7 THEN OUT=1
IF X2<0.2 THEN OUT=1
IF X4>0.8 THEN OUT=1
IF X3<0.7 AND X3>0.2 AND X4<0.3 THEN OUT=1
```

Comprehensible Decision Systems

Rules are considered to be *comprehensible* and *interpretable*

- symbolic rule model can be inspected
  - discover relations between inputs and target concept
  - experts can check critical rules, e.g.: IF … THEN *emergency braking*
- taken decisions can be explained by firing rules
  - firing rule reveals decisive attributes and the training examples from which the rule was learned



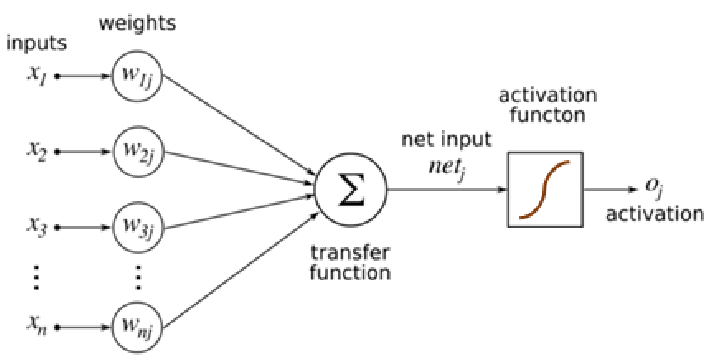

```
IF X1<0.5 AND X2>0.75 THEN OUT=1
IF X1>0.9 THEN OUT=1
IF X1>0.5 AND X1<0.9 AND X3>0.2 THEN OUT=1
IF X2>0.2 AND X3<0.5 AND X5<0.5 THEN OUT=1
IF X2>0.4 AND X3<0.7 THEN OUT=1
IF X2<0.2 THEN OUT=1
IF X4>0.8 THEN OUT=1
IF X3<0.7 AND X3>0.2 AND X4<0.3 THEN OUT=1
```

Extracting Rules from Neural Networks

TECHNISCHE UNIVERSITÄT DARMSTADT

Rule extraction strategies

- Decompositional (considering NN's structure)



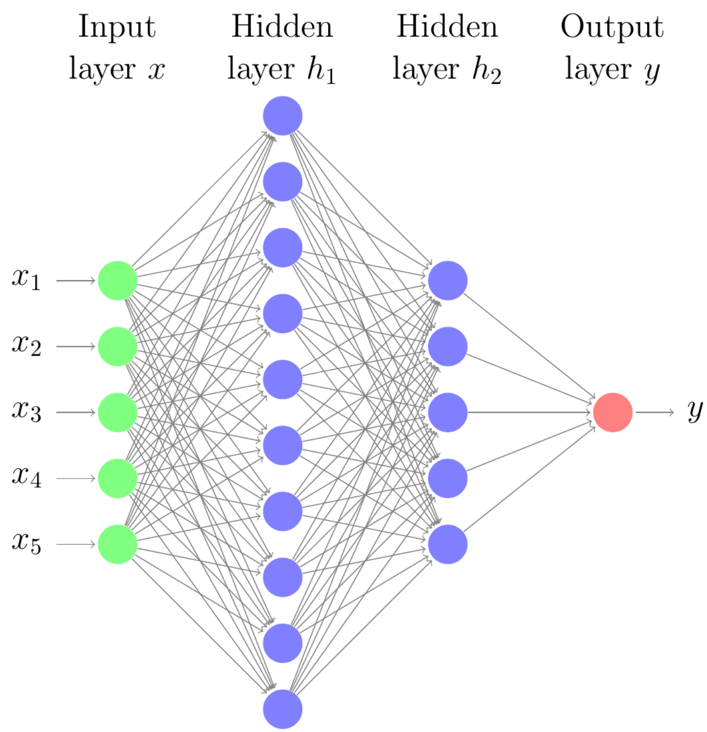
IF X1=hi OR X2=hi OR X3=hi THEN OUT=hi

Extracting Rules from Neural Networks

Rule extraction strategies

- Decompositional (considering NN's structure)
- Pedagogical (NN as black box)

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…



o
1
0
1
0
1
1
1
…

Extracting Rules from Neural Networks

Rule extraction strategies

- Decompositional (considering NN's structure)
- Pedagogical (NN as black box)

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

```
IF X1<0.5 AND X2>0.75 THEN OUT=1
IF X1>0.9 THEN OUT=1
IF X1>0.5 AND X1<0.9 AND X3>0.2 THEN OUT=1
IF X2>0.2 AND X3<0.5 AND X5<0.5 THEN OUT=1
IF X2>0.4 AND X3<0.7 THEN OUT=1
IF X2<0.2 THEN OUT=1
IF X4>0.8 THEN OUT=1
IF X3<0.7 AND X3>0.2 AND X4<0.3 THEN OUT=1
```

o
1
0
1
0
1
1
1
…

Extracting Rules from Neural Networks

TECHNISCHE UNIVERSITÄT DARMSTADT

Rule extraction strategies

- Decompositional (considering NN's structure)
- Pedagogical (NN as black box)
- Eclectic (mixture of both)

Models

- previous research in the 90s focussed on extracting rules from flat NNs
- types of extracted rules (DNFs, decision tree, fuzzy rules, ...)

DeepRED: Extraction of Rules from Deep Neural Networks

Goals

- make hidden features accessible (in contrast to pedagogical)
- exploit deep structure to improve efficacy of rule extraction and induction process

Based on CRED

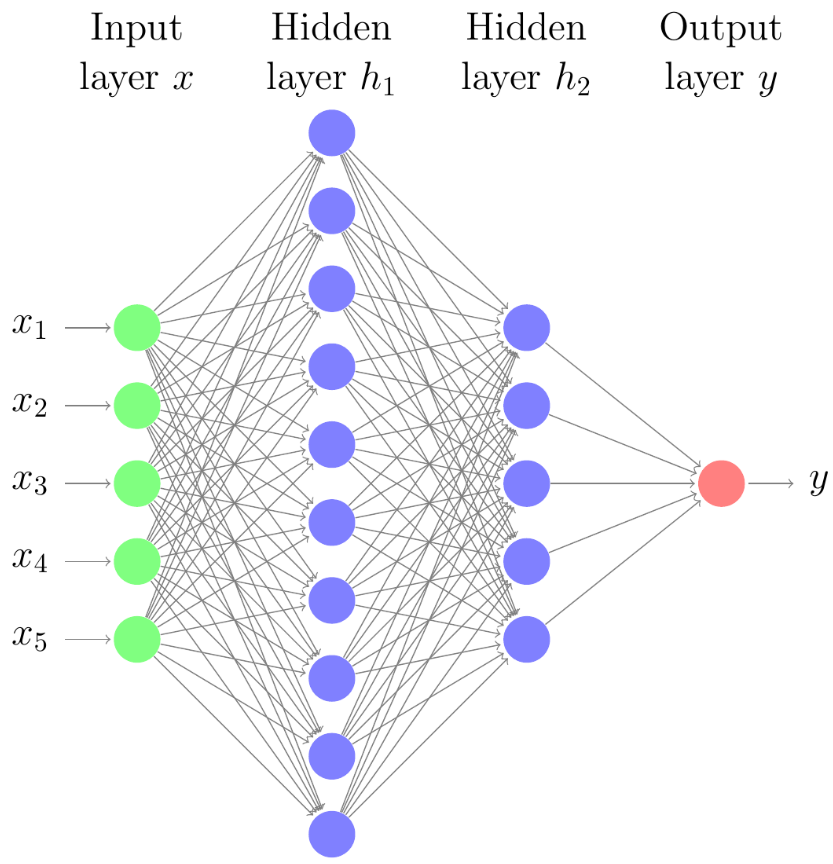
- *Continuous/discrete Rule Extractor via Decision tree induction* (CRED) [Sato and Tsukimoto, 2001]
- only supports NNs with one hidden layer
- uses C4.5 to induce rules

DeepRED extends CRED to arbitrary number of layers

- roughly speaking: apply CRED layer by layer
- decomposible w.r.t. neurons, pedagogical w.r.t. neurons' behaviour

Pedagogical Baseline

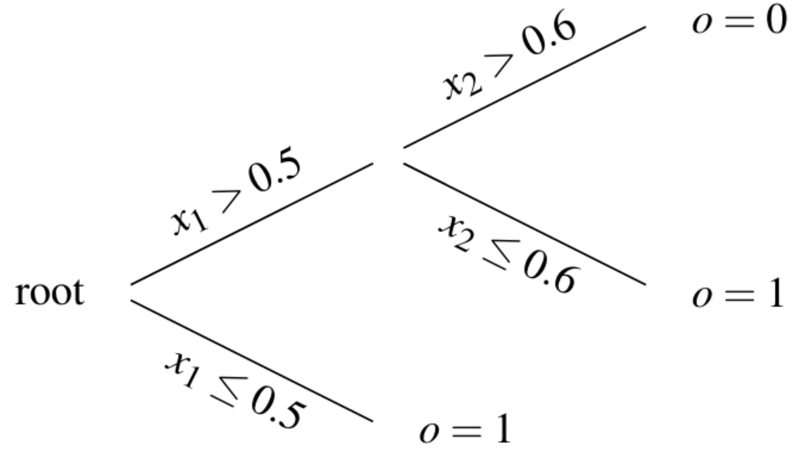
x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…



o
1
0
1
0
1
1
1
…

Pedagogical Baseline

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…



o
1
0
1
0
1
1
1
…

DeepRED
Step 1: track activations at every layer

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$
0.865	0.079	…	0.818
0.050	0.675	…	0.613
0.767	0.485	…	0.020
0.388	0.160	…	0.491
0.555	0.767	…	0.606
0.312	0.231	…	0.376
0.770	0.211	…	0.805
…	…	…	…

$h_{2,1}$	$h_{2,2}$	…	$h_{2,5}$
0.034	0.635	…	0.928
0.089	0.049	…	0.435
0.057	0.369	…	0.233
0.346	0.462	…	0.181
0.834	0.945	…	0.354
0.443	0.644	…	0.892
0.778	0.691	…	0.708
…	…	…	…

o
1
0
1
0
1
1
1
…

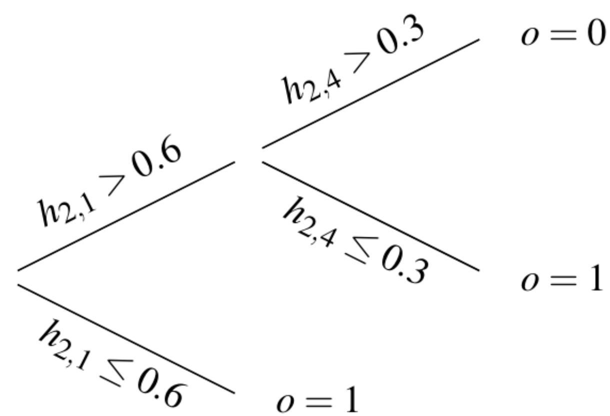
Step 2: Find a decision tree that describes an output node using activation values of the previous hidden layer h_i

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$
0.865	0.079	…	0.818
0.050	0.675	…	0.613
0.767	0.485	…	0.020
0.388	0.160	…	0.491
0.555	0.767	…	0.606
0.312	0.231	…	0.376
0.770	0.211	…	0.805
…	…	…	…

$h_{2,1}$	$h_{2,2}$	…	$h_{2,5}$
0.034	0.635	…	0.928
0.089	0.049	…	0.435
0.057	0.369	…	0.233
0.346	0.462	…	0.181
0.834	0.945	…	0.354
0.443	0.644	…	0.892
0.778	0.691	…	0.708
…	…	…	…

o
1
0
1
0
1
1
1
…



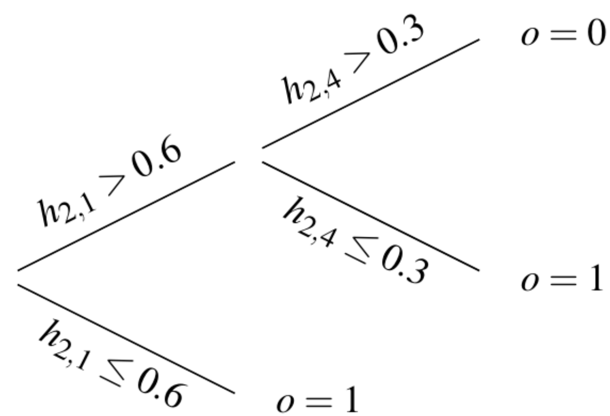
Step 3: Advance to next layer h_{i-1} Describe activations in current layer h_i w.r.t. activations in previous layer h_{i-1}

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$
0.865	0.079	…	0.818
0.050	0.675	…	0.613
0.767	0.485	…	0.020
0.388	0.160	…	0.491
0.555	0.767	…	0.606
0.312	0.231	…	0.376
0.770	0.211	…	0.805
…	…	…	…

$h_{2,1}$	$h_{2,2}$	…	$h_{2,5}$
0.034	0.635	…	0.928
0.089	0.049	…	0.435
0.057	0.369	…	0.233
0.346	0.462	…	0.181
0.834	0.945	…	0.354
0.443	0.644	…	0.892
0.778	0.691	…	0.708
…	…	…	…

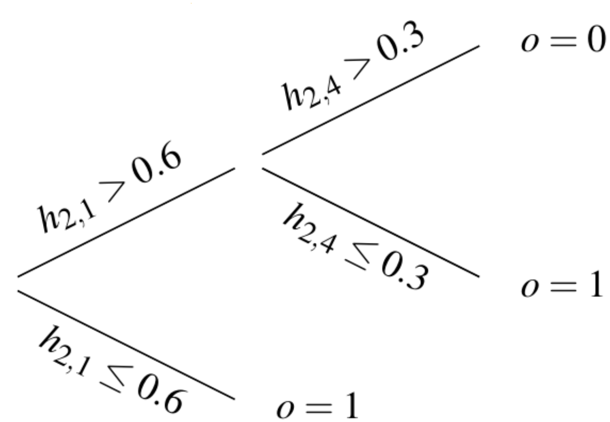
o
1
0
1
0
1
1
1
…



Step 3.1: Replace target activations h_i by split points on h_i using in prediction model $h_i \rightarrow h_{i+1}$

TECHNISCHE UNIVERSITÄT DARMSTADT

x_1	x_2	x_3	x_4	x_5	$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$	$h_{2,1}$	$h_{2,2}$	…	$h_{2,5}$	o
0.5	1	0.200	0.648	0.875	0.865	0.079	…	0.818	0.034	0.635	…	0.928	1
0.5	1	0.197	0.889	0.487	0.050	0.675	…	0.613	0.089	0.049	…	0.435	0
0.5	0.25	0.972	0.754	0.711	0.767	0.485	…	0.020	0.057	0.369	…	0.233	1
0	0.75	0.884	0.580	0.213	0.388	0.160	…	0.491	0.346	0.462	…	0.181	0
0.5	0	0.860	0.795	0.475	0.555	0.767	…	0.606	0.834	0.945	…	0.354	1
1	0.75	0.505	0.905	0.692	0.312	0.231	…	0.376	0.443	0.644	…	0.892	1
1	0.75	0.731	0.084	0.409	0.770	0.211	…	0.805	0.778	0.691	…	0.708	1
…	…	…	…	…	…	…	…	…	…	…	…	…	…



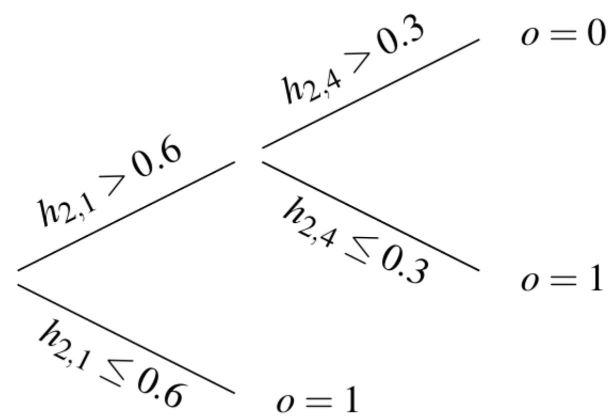
Step 3.1: Replace target activations h_i by split points on h_i using in prediction model $h_i \rightarrow h_{i+1}$

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
...	...	...	...	...

$h_{1,1}$	$h_{1,2}$	...	$h_{1,10}$
0.865	0.079	...	0.818
0.050	0.675	...	0.613
0.767	0.485	...	0.020
0.388	0.160	...	0.491
0.555	0.767	...	0.606
0.312	0.231	...	0.376
0.770	0.211	...	0.805
...	...	...	...

$h_{2,1}>0.3$	$h_{2,1}>0.6$	...	$h_{2,4}>0.3$
0	0	...	1
0	0	...	1
0	0	...	0
1	0	...	0
1	1	...	1
1	0	...	1
1	1	...	1
...	...	...	...

o
1
0
1
0
1
1
1
...



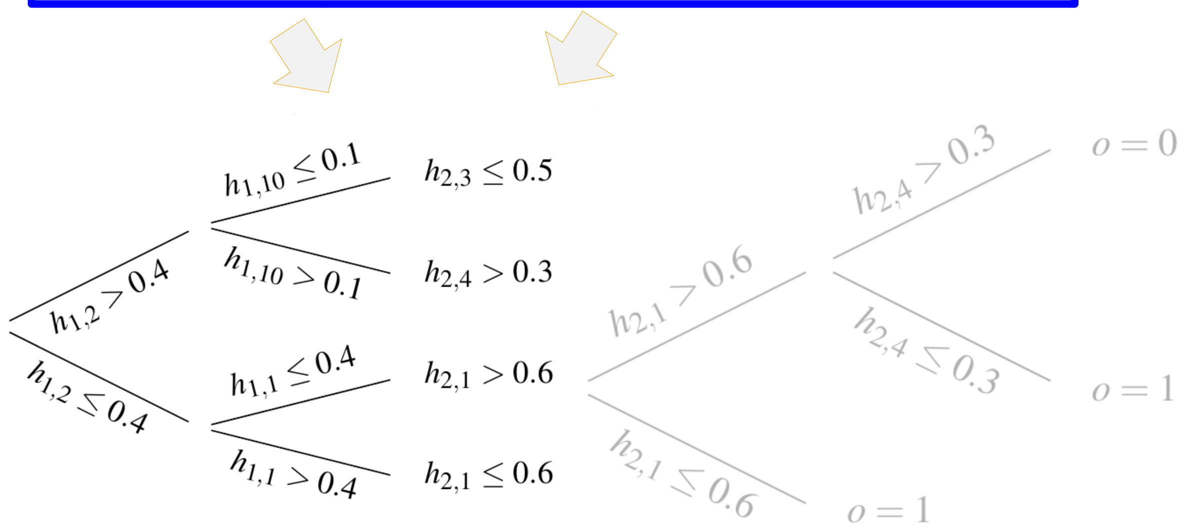
Step 3.1: Induce model $h_{i-1} \rightarrow h_i$

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
...	...	...	...	...

$h_{1,1}$	$h_{1,2}$	...	$h_{1,10}$
0.865	0.079	...	0.818
0.050	0.675	...	0.613
0.767	0.485	...	0.020
0.388	0.160	...	0.491
0.555	0.767	...	0.606
0.312	0.231	...	0.376
0.770	0.211	...	0.805
...	...	...	...

$h_{2,1}>0.3$	$h_{2,1}>0.6$	...	$h_{2,4}>0.3$
0	0	...	1
0	0	...	1
0	0	...	0
1	0	...	0
1	1	...	1
1	0	...	1
1	1	...	1
...	...	...	...

o
1
0
1
0
1
1
1
...



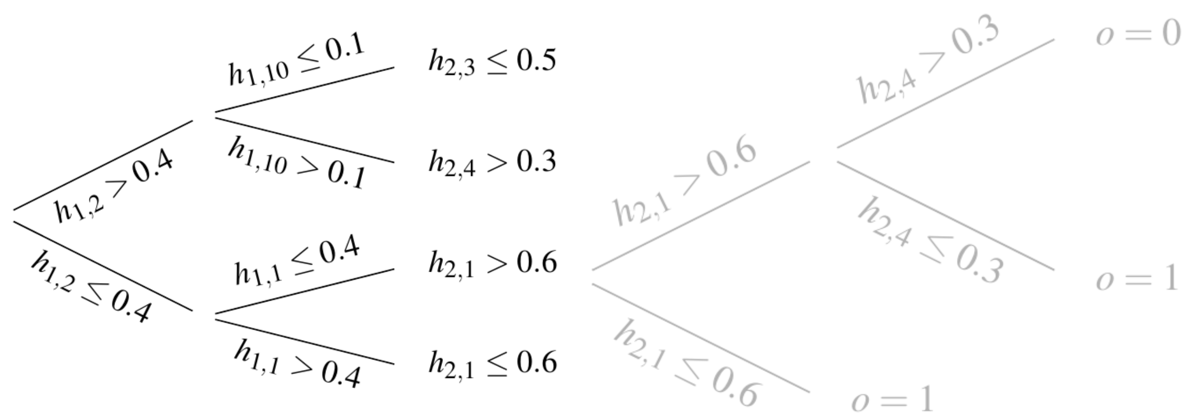
Step 3.2: Repeat step 3 for all hidden layers until $h_{i+1} = x$

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$
0.865	0.079	…	0.818
0.050	0.675	…	0.613
0.767	0.485	…	0.020
0.388	0.160	…	0.491
0.555	0.767	…	0.606
0.312	0.231	…	0.376
0.770	0.211	…	0.805
…	…	…	…

$h_{2,1}>0.3$	$h_{2,1}>0.6$	…	$h_{2,4}>0.3$
0	0	…	1
0	0	…	1
0	0	…	0
1	0	…	0
1	1	…	1
1	0	…	1
1	1	…	1
…	…	…	…

o
1
0
1
0
1
1
1
…



Step 3.2: Repeat step 3 for all hidden layers until $h_{i+1} = x$

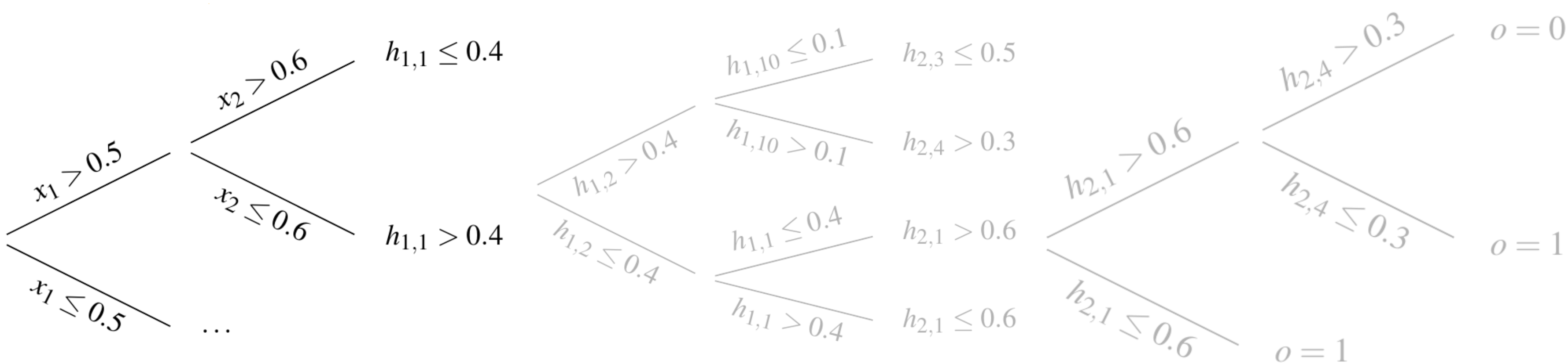
TECHNISCHE UNIVERSITÄT DARMSTADT

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}>0.4$	$h_{1,2}>0.4$	…	$h_{1,10}>0.1$
1	0	…	1
0	1	…	1
1	1	…	0
0	0	…	1
1	1	…	1
0	0	…	1
1	0	…	1
…	…	…	…

$h_{2,1}>0.3$	$h_{2,1}>0.6$	…	$h_{2,4}>0.3$
0	0	…	1
0	0	…	1
0	0	…	0
1	0	…	0
1	1	…	1
1	0	…	1
1	1	…	1
…	…	…	…

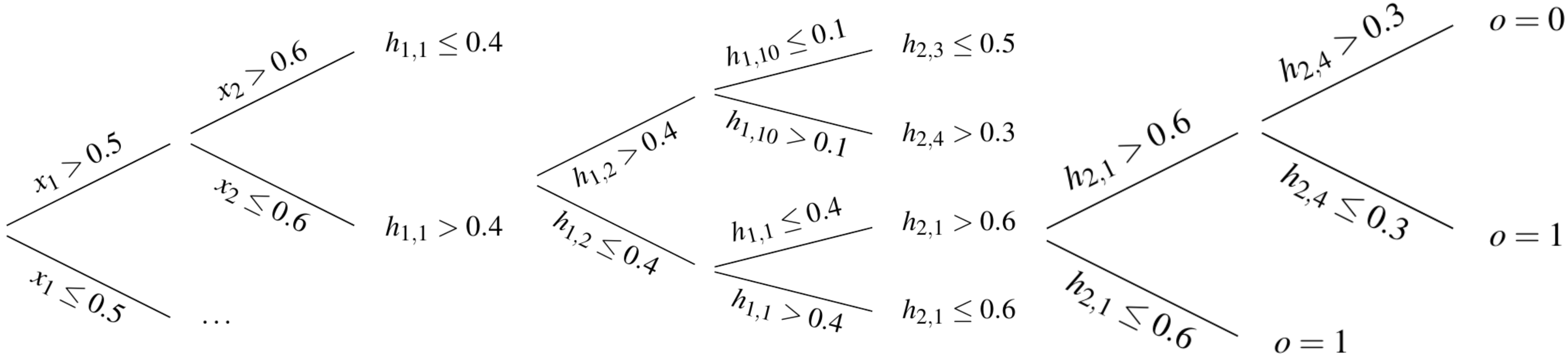
o
1
0
1
0
1
1
1
…



Step 4:
Extract rules and clean up

Represent output as a function of inputs

- Extract rule sets $R(h_{i-1} \rightarrow h_i)$ from decision trees
- Advance layerwise
  - put $R(h_{i-1} \rightarrow h_i)$ into $R(h_i \rightarrow h_o)$ to get $R(h_{i-1} \rightarrow h_o)$
  - delete unsatisfiable and redundant terms



Step 4:
Extract rules and clean up

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Represent output as a function of inputs

- Extract rule sets $R(h_{i-1} \rightarrow h_i)$ from decision trees
- Advance layerwise
  - put $R(h_{i-1} \rightarrow h_i)$ into $R(h_i \rightarrow h_o)$ to get $R(h_{i-1} \rightarrow h_o)$
  - delete unsatisfiable and redundant terms

```
IF x1>0.5  AND x2>0.6
    THEN h11<=0.4
IF x1>0.5  AND x2<=0.6
    THEN h11>0.4
IF x1<=0.5  …
...
```

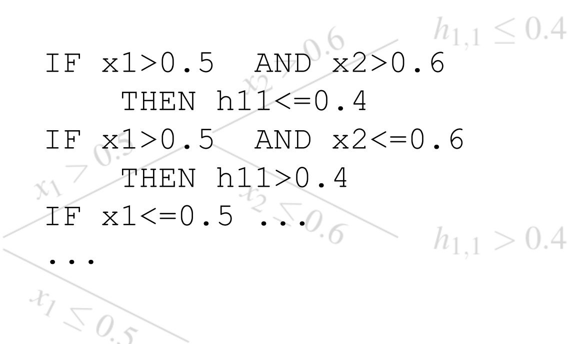
```
IF h12>0.4 AND h110<=0.1
   THEN h23<=0.5
IF h12>0.4 AND h110>0.1
   THEN h24>0.3
IF h12<=0.4 AND h11<=0.4
   THEN h21>0.6
IF h12<=0.4 AND h11 >0.1
   THEN h21<=0.6
```

```
IF h21>0.6 AND h24>0.3
    THEN o=0
IF h21>0.6 AND h24<=0.3
    THEN o=1
IF h21<=0.6
    THEN o=1
```

Step 4: Extract rules and clean up

Represent output as a function of inputs

- Extract rule sets $R(h_{i-1} \rightarrow h_i)$ from decision trees
- Advance layerwise
  - put $R(h_{i-1} \rightarrow h_i)$ into $R(h_i \rightarrow h_o)$ to get $R(h_{i-1} \rightarrow h_o)$
  - delete unsatisfiable and redundant terms




```
IF x1>0.5  AND x2>0.6
    THEN h11<=0.4
IF x1>0.5  AND x2<=0.6
    THEN h11>0.4
IF x1<=0.5 ...
...
```

```
IF (h12<=0.4 AND h11>0.1) AND (h12>0.4 AND h110>0.1)
THEN o=0
...
```

Step 4: Extract rules and clean up

Represent output as a function of inputs

- Extract rule sets $R(h_{i-1} \rightarrow h_i)$ from decision trees
- Advance layerwise
  - put $R(h_{i-1} \rightarrow h_i)$ into $R(h_i \rightarrow h_o)$ to get $R(h_{i-1} \rightarrow h_o)$
  - delete unsatisfiable and redundant terms

```
IF x1<0.5 AND x2>0.75 THEN o=1
IF x1>0.9 THEN o=1
IF x1>0.5 AND x1<0.9 AND x3>0.2 THEN o=1
IF x2>0.2 AND x3<0.5 AND x5<0.5 THEN o=1
IF x2>0.4 AND x3<0.7 THEN o=1
IF x2<0.2 THEN o=1
IF x4>0.8 THEN o=1
IF x3<0.7 AND x3>0.2 AND x4<0.3 THEN o=1
```

Step 4:
Extract rules and clean up

Represent output as a function of inputs

- Extract rule sets $R(h_{i-1} \rightarrow h_i)$ from decision trees
- Advance layerwise
  - put $R(h_{i-1} \rightarrow h_i)$ into $R(h_i \rightarrow h_o)$ to get $R(h_{i-1} \rightarrow h_o)$
  - delete unsatisfiable and redundant terms
- DeepRED can generally represent any neuron as a function of the outputs of any preceding layer

Optional RxREN pruning

  - prunes insignificant inputs by testing NN performance while ignoring the given input

Experimental setup

Datasets and DNNs used

	#attributes	#training ex.	#test ex.	NN structure	acc(training)	acc(test)
MNIST	784	12056	2195	784-10-5-2	99.6%	98.8%
letter	16	1239	438	16-40-30-26	96.9%	97.3%
artif-I	5	20000	10000	5-10-5-2	99.5%	99.4%
artif-II	5	3348	1652	5-10-5-2	99.4%	99.0%
XOR	8	150	106	8-8-4-4-2-2-2	100%	100%

Evaluation measures

- fidelity on test set: accuracy on mimicking NN's behaviour
- number of terms: tries to assess comprehensibility of found rule set

Algorithm setup

- 36 combinations of varying C4.5 parameters, pruning parameters and train set sizes

Can DeepRED make use of complex concepts hidden in NNs?

TECHNISCHE UNIVERSITÄT DARMSTADT

artif-I

- artificial dataset randomly drawn
- output defined by rule set which cannot easily be realized by decision trees
  - contains pairwise comparisons between inputs

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

```
IF x1 = x2 THEN out=1
IF x1 > x2 AND x3 > 0.4 THEN out=1
IF x3 > x4 AND x4 > x5 AND x2 > 0 THEN out=1
ELSE out=0
```

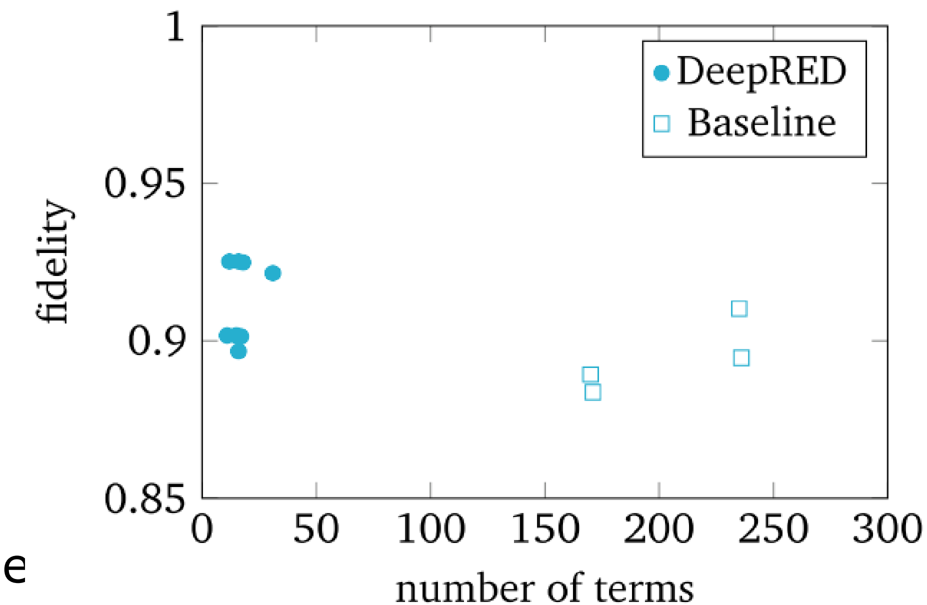
o
1
0
1
0
1
1
1
…

Can DeepRED make use of complex concepts hidden in NNs?

TECHNISCHE UNIVERSITÄT DARMSTADT

artif-I

- artificial dataset randomly drawn
- output defined by rule set which cannot easily be realized by decision trees
  - contains pairwise comparisons between inputs



Results

- DeepRED outperforms pedagogical baseline
  - especially in comprehensibility dimension
- hidden concepts lead to compactne

Can DeepRED make use of complex concepts hidden in NNs?

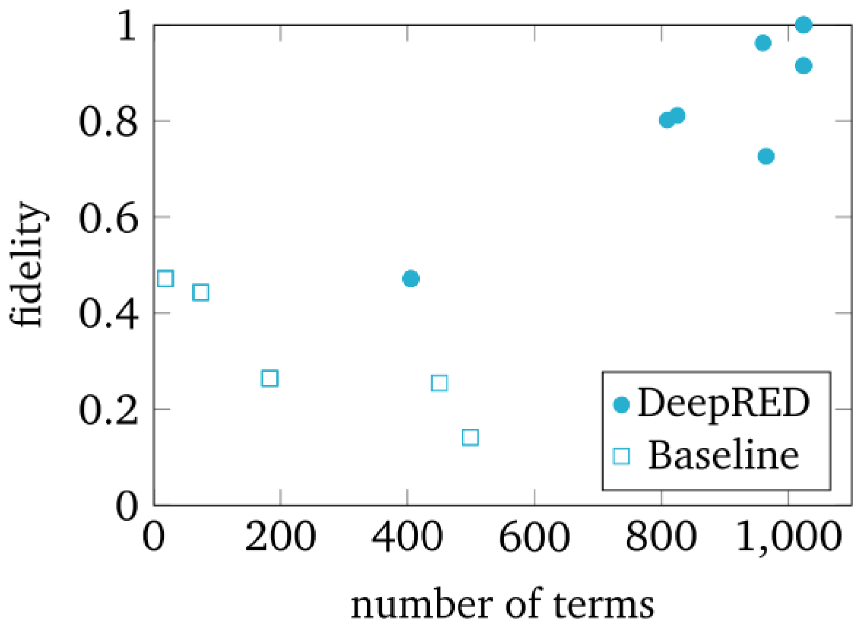
TECHNISCHE UNIVERSITÄT DARMSTADT

XOR

- parity function: $x \in \{0,1\}^8 \rightarrow XOR(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8\}$
- 2^8 examples split into 150 training and 106 test examples
- top-down approaches (e.g. C4.5) usually need all examples to learn consistent model

Results

- as expected, baseline fails
- DeepRED is able to extract rules that classify all or almost all test examples correctly



Can DeepRED make use of complex concepts hidden in NNs?

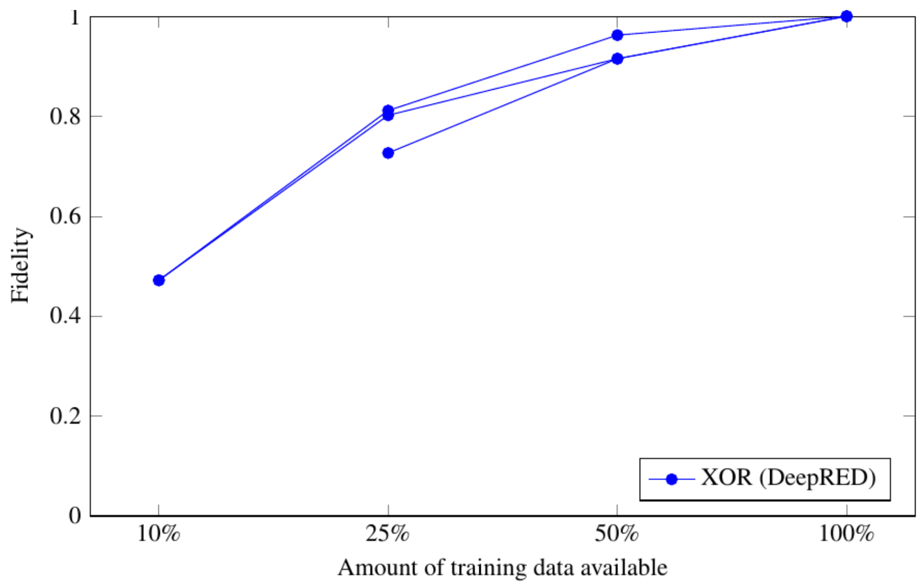
TECHNISCHE UNIVERSITÄT DARMSTADT

XOR

- parity function: $x \in \{0,1\}^8 \rightarrow XOR(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8\}$
- 2^8 examples split into 150 training and 106 test examples
- top-down approaches (e.g. C4.5) usually need all examples to learn consistent model

Results

- even with only 75 training examples DeepRED extracts meaningful rules (>90% fidelity)
- DeepRED effectively captures inherent concepts otherwise non accessible



More insights

TECHNISCHE UNIVERSITÄT DARMSTADT

Limitations

- artif-II
  - *can* easily be realized by decision tree
  - baseline finds more comprehensible rules with very good fidelity

Pruning

- removal of up to 10% inputs possible without substantial decrease in fidelity
- but reduction in number of conditions of several magnitudes

Training set size

- DeepRED quite stable w.r.t. reduction of training set

Conclusions

TECHNISCHE UNIVERSITÄT DARMSTADT

DeepRED

- to our knowledge, first attempt on extracting rules form deep neural networks
  - important step towards making NN's decisions transparent
- outperforms pedagogical baselines for most of the analyzed cases
- DeepRED benefits from deep architecture of NNs when addressing data with complex concepts

Questions?

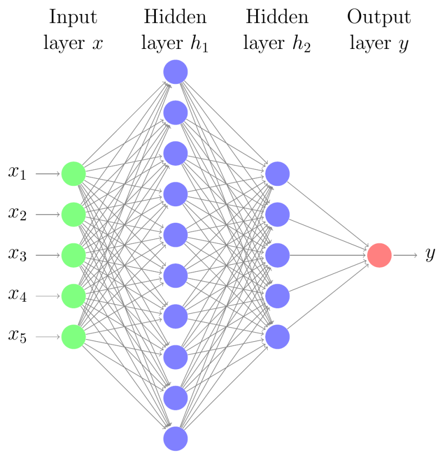
TECHNISCHE UNIVERSITÄT DARMSTADT

x_1	x_2	x_3	x_4	x_5
0.5	1	0.200	0.648	0.875
0.5	1	0.197	0.889	0.487
0.5	0.25	0.972	0.754	0.711
0	0.75	0.884	0.580	0.213
0.5	0	0.860	0.795	0.475
1	0.75	0.505	0.905	0.692
1	0.75	0.731	0.084	0.409
…	…	…	…	…

$h_{1,1}$	$h_{1,2}$	…	$h_{1,10}$
0.865	0.079	…	0.818
0.050	0.675	…	0.613
0.767	0.485	…	0.020
0.388	0.160	…	0.491
0.555	0.767	…	0.606
0.312	0.231	…	0.376
0.770	0.211	…	0.805
…	…	…	…

$h_{2,1}$	$h_{2,2}$	…	$h_{2,5}$
0.034	0.635	…	0.928
0.089	0.049	…	0.435
0.057	0.369	…	0.233
0.346	0.462	…	0.181
0.834	0.945	…	0.354
0.443	0.644	…	0.892
0.778	0.691	…	0.708
…	…	…	…

o
1
0
1
0
1
1
1
…




```
IF x1 = x2 THEN out=1
IF x1 > x2 AND x3 > 0.4 THEN out=1
IF x3 > x4 AND x4 > x5 AND x2 > 0 THEN out=1
IF x4=look OR x4=see THEN out=1
ELSE out=0
```

Sources

- CRED algorithm: Sato, M. and Tsukimoto, H. (2001). Rule extraction from neural networks via decision tree induction. In Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on, volume 3, pages 1870–1875. IEEE.
- RxREN algorithm: Augasta, M. G. and Kathirvalavakumar, T. (2012a). Reverse engineering the neural networks for rule extraction in classification problems. Neural processing letters, 35(2):131–150.

Evaluation with overall 180 + 60 experiments

- Rule extraction algorithms
  - DeepRED
  - DeepRED with RxREN pruning
  - C4.5 as baseline (pedagogical)
- Different parameter settings
  - Amount of training data available: for all algorithms
  - Stopping criteria for C4.5 (class dominance, database size): for all algorithms
  - Pruning threshold: for DeepRED with/without RxREN pruning

Training set
10%
25%
50%
100%

C4.5 parameters
92% / $\leq$ 2%
95% / $\leq$ 1%
99% / $\leq$ 0%

RxREN pruning
No pruning
5%
10%

Generous hardware constraints to extract high-quality rules

- Measures to rate the extracted rules
  - Fidelity
  - Number of terms

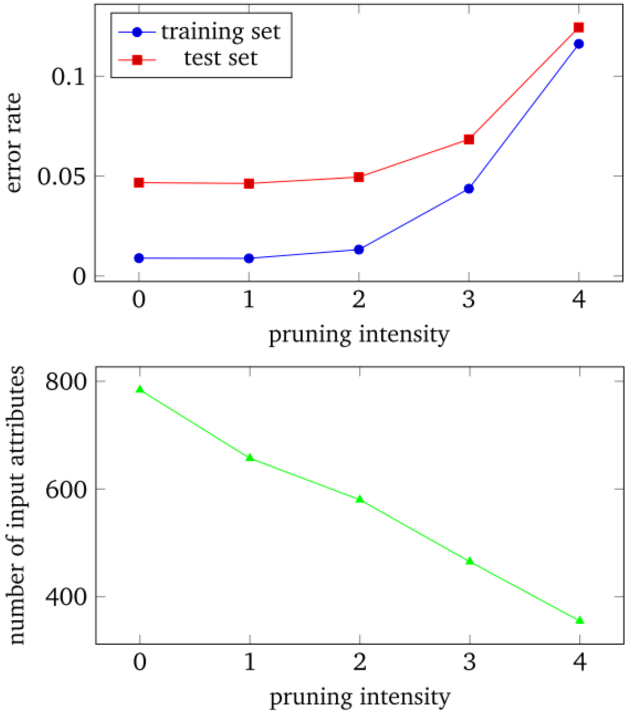
- Hardware settings
  - Lichtenberg High Performance Computer
  - Maximum memory consumption: 10,000MB
  - Maximum computation time: 24 hours

Evaluation setting aims at analysing if expectations are met

- Expectations
  - DeepRED is able to extract rules from DNNs
    (proof of concept)
  - DeepRED outperforms baseline on rather complex problems
    (complex = difficult to describe by decision trees)
  - RxREN pruning leads to more comprehensible rules
    (if not all inputs are relevant)
  - DeepRED extracts more accurate rules if more data is available
    (however, less dependant on data set size than baseline)

Additional input pruning can help extracting comprehensible rules

- DeepRED intrinsically implements hidden neuron pruning
  - A neuron is ignored if it isn't present in the decision trees of next deeper layer
- RxREN input pruning
  - Prunes insignificant inputs by testing NN performance while ignoring the given input
  - Can improve the basis for DeepRED to extract more comprehensible rules
  - E.g. ignoring 204 of 784 inputs decreases the NN's performance only by 0.3 pp



DeepRED can successfully extract rules from DNNs

- For every dataset at least one parameter setting leads to extracted rules
- This also holds true for every RxREN pruning setting (except for MNIST)
- But: High abortion rates due to too many intermediate rules

	artif-I	artif-II	letter	MNIST	XOR
Executed	36	36	36	21	12
Successful	11	23	26	4	12
Aborted (memory)	24	13	10	7	0
Aborted (time)	1	0	0	10	0

- (Automated) parameter tun

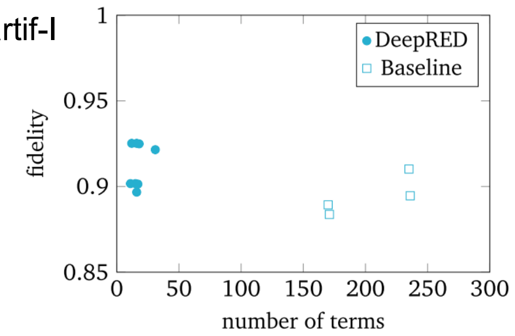
DeepRED can extract comprehensible rules for rather complex problems

- DeepRED outperforms baseline on artif-I
  (artif-I cannot easily be realized by decision tree)
  - Especially in comprehensibility dimension

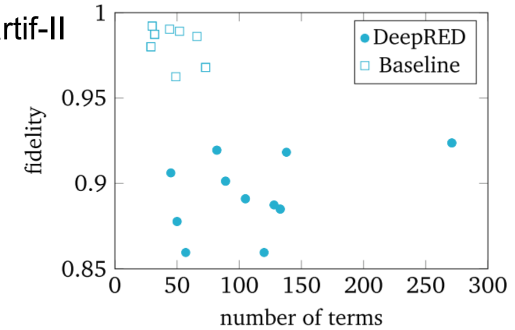
- Baseline finds more comprehensible rules for artif-II with very good fidelity rates
  (artif-II can easily be realized by decision tree)

- DeepRED is able to extract rules that classify all XOR test examples correctly

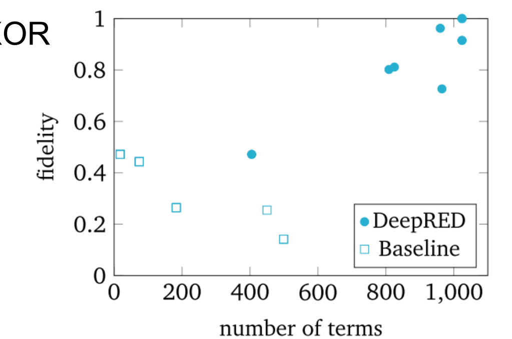
artif-I



artif-II

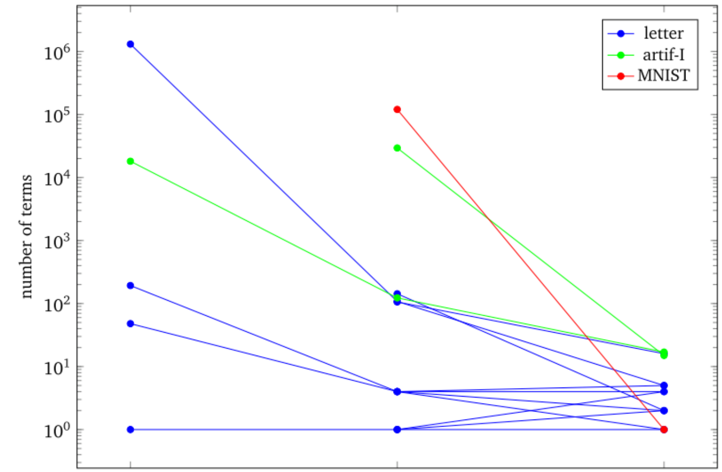


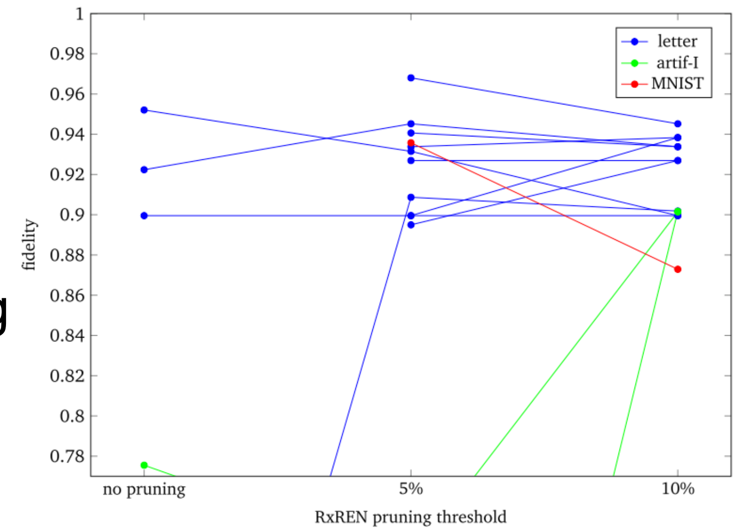
XOR



RxREN pruning helps DeepRED to extract more comprehensible rules

- Pruning never leads to worse comprehensibility
  - Often, pruning enables rule extraction
  - Larger pruning thresholds can negatively affect comprehensibility
- For artif-I, 10% pruning threshold leads to best rule set (fidelity and #terms)
- Overall, a more elaborate setup of pruning threshold could lead to optimized results

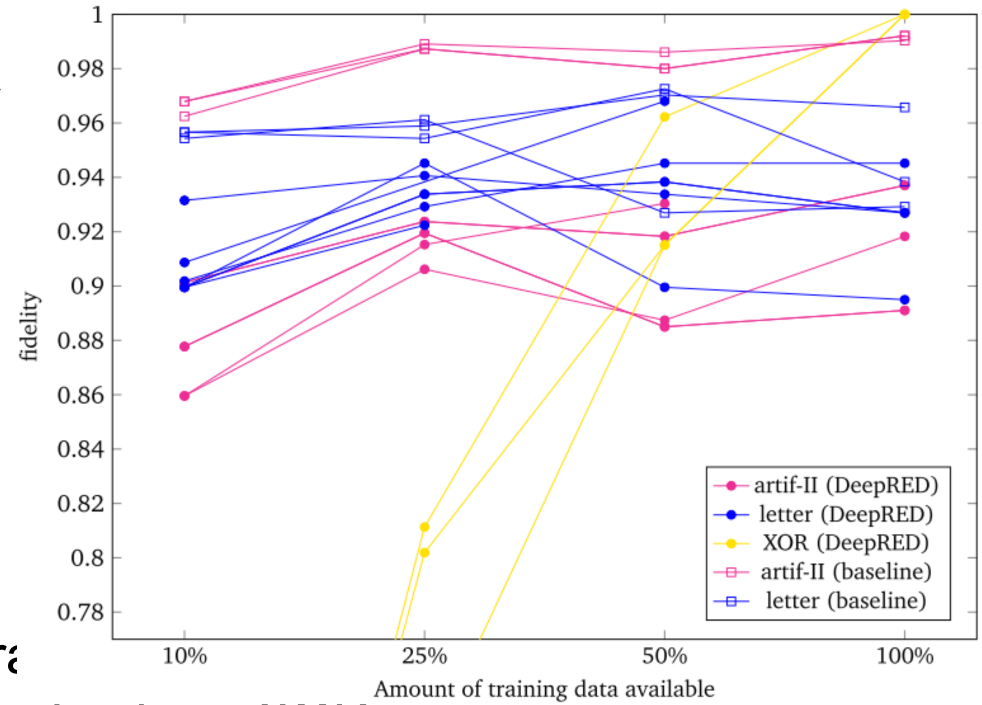




For most tasks the fidelity of the extracted rules is independent from training size

TECHNISCHE UNIVERSITÄT DARMSTADT

- Having 25% of the training data is better than having 10%
  - But more data doesn't necessarily help DeepRED
  - Reasons for decrease in some cases currently unknown
  - Baseline profits from more data



- DeepRED benefits from more tra
  structure to extract high-quality rules from XOR
  - Pedagogical baseline cannot extract sensible rules

Research and evaluation led to several ideas for future work

- Evaluation has shown challenges and opportunities
  - Good parameter settings are important to receive good results
  - Automated mechanism to fine-tune C4.5 and pruning parameters would be helpful
  - More elaborate approaches to select necessary examples from the training set could improve results
- A replacement or extension of C4.5 in DeepRED could be valuable
- Extending other rule extraction algorithms to DNNs still is necessary to learn from these results