KILL: Knowledge Injection via Lambda Layer

Matteo Magnini* Giovanni Ciatto* Andrea Omicini*

*Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna
{matteo.magnini, giovanni.ciatto, andrea.omicini}@unibo.it

WOA 2022:
Workshop "From Objects to Agents"
September 1–2, 2022, Genova, Italy

Next in Line...

Symbolic Knowledge Injection I

Definition [Besold et al., 2017, Xie et al., 2019, Calegari et al., 2020]

Symbolic Knowledge Injection (SKI) can be defined as:

> *any* **algorithmic** *procedure affecting how sub-symbolic predictors draw their inferences in such a way that predictions are either* **computed** *as a function of, or made* **consistent** *with, some* **given** *symbolic knowledge.*

Symbolic Knowledge Injection II

Symbolic knowledge

A symbolic representation consists of: [van Gelder, 1990]

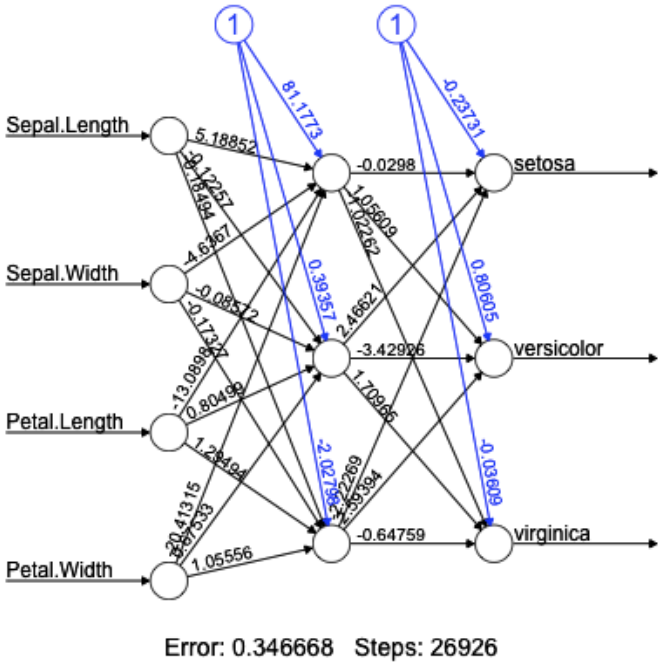
1. a set of symbols;
2. a set of grammatical rules governing the combining of symbols;
3. elementary symbols and any admissible combination of them can be assigned with meaning.
   - $\Rightarrow$ Symbolic knowledge is both human and machine interpretable,
   - first order logic (FOL) is an example of symbolic representation.

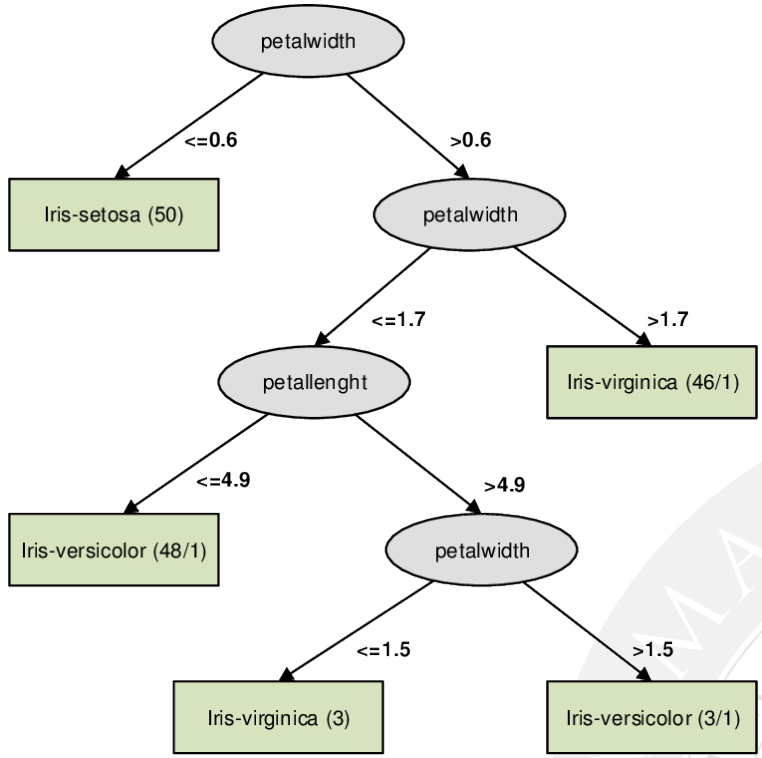
Symbolic Knowledge Injection III

Sub-symbolic predictors

- deep neural networks (DNN);
  - convolutional neural networks (CNN),
  - recurrent neural networks (RNN);
- kernel machines;
- basically everything that is sub-symbolic (models consisting of vectors, tensors, etc. of real numbers with no meaning for a human).

Symbolic Knowledge Injection IV



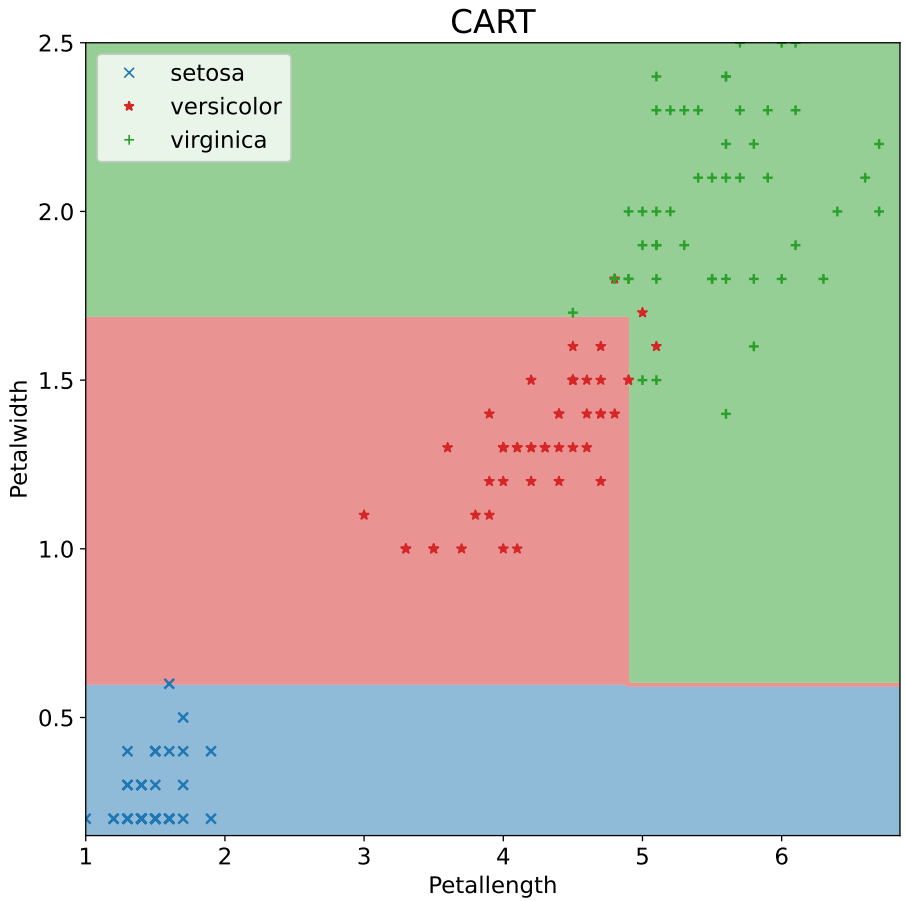


Symbolic Knowledge Injection V

Set of propositional logic rules built from the previous decision tree:

$$\begin{aligned}
&\mathit{iris}(\mathit{SepalLenght}, \mathit{SepalWidth}, \mathit{PetalLenght}, \mathit{PetalWidth}, \texttt{setosa}) :- \\
&\quad \mathit{PetalWidth} <= 0.6. \\
&\mathit{iris}(\mathit{SepalLenght}, \mathit{SepalWidth}, \mathit{PetalLenght}, \mathit{PetalWidth}, \texttt{versicolor}) :- \\
&\quad \mathit{PetalWidth} > 0.6, \mathit{PetalWidth} <= 1.7, \mathit{PetalLenght} <= 4.9. \\
&\mathit{iris}(\mathit{SepalLenght}, \mathit{SepalWidth}, \mathit{PetalLenght}, \mathit{PetalWidth}, \texttt{virginica}) :- \\
&\quad \mathit{PetalWidth} > 0.6, \mathit{PetalWidth} <= 1.5, \mathit{PetalLenght} > 4.9. \\
&\mathit{iris}(\mathit{SepalLenght}, \mathit{SepalWidth}, \mathit{PetalLenght}, \mathit{PetalWidth}, \texttt{versicolor}) :- \\
&\quad \mathit{PetalWidth} > 1.5, \mathit{PetalWidth} <= 1.7, \mathit{PetalLenght} > 4.9. \\
&\mathit{iris}(\mathit{SepalLenght}, \mathit{SepalWidth}, \mathit{PetalLenght}, \mathit{PetalWidth}, \texttt{virginica}) :- \\
&\quad \mathit{PetalWidth} > 1.7.
\end{aligned}$$

Symbolic Knowledge Injection VI

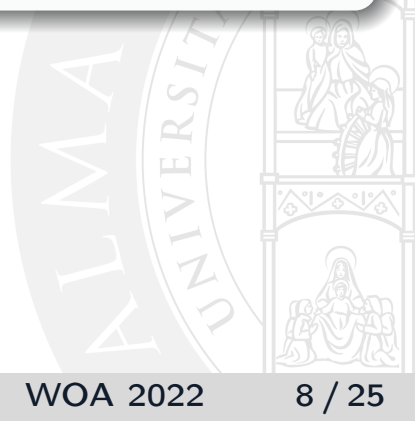


Symbolic Knowledge Injection VII

Why?

There are several benefits:

- prevent the predictor to become a black-box!;
- reduce learning time;
- reduce the data size needed for training;
- improve predictor's accuracy;
- build a predictor that behave as a logic engine.



Symbolic Knowledge Injection VIII

Explainability can be achieved: [Gunning, 2016]

Post-hoc explanation

- applying an algorithm of symbolic knowledge extraction on a trained predictor;
- output $\rightarrow$ logic rules that describe the predictor's behaviour.

By design

- constraining the behaviour of predictors that are natively black-boxes with symbolic knowledge;
- structuring the predictor's architecture with symbolic knowledge;
- output $\rightarrow$ a predictor that does not violate the prior knowledge.

Symbolic Knowledge Injection IX

How?

There exist three major ways to perform knowledge injection on sub-symbolic predictors:

- constraining, a cost factor proportional to the violation of the knowledge is introduced during learning;
- structuring, the architecture of the predictor is built in such a way to mimic the knowledge;
- embedding, the symbolic knowledge is embedded into a tensor form and it is given in input as training data to the predictor.

Next in Line...

Algorithm I

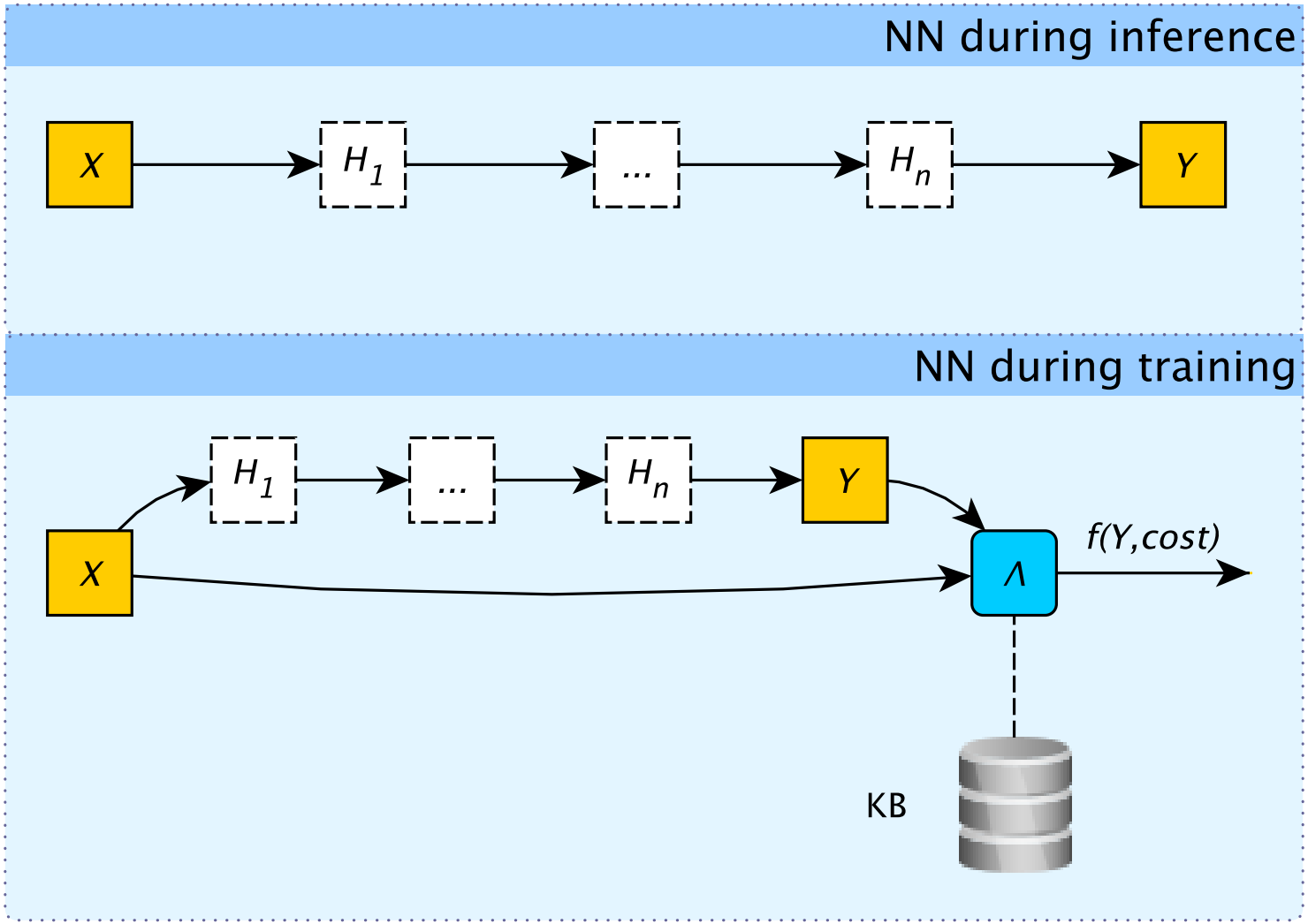
KILL: Knowledge Injection via Lambda Layer

A general SKI algorithm that does not impose constrains on the sub-symbolic predictor to enrich, except being a neural network.

- aim $\rightarrow$ enrich;
- predictor $\rightarrow$ neural network;
- how $\rightarrow$ constraining;
- logic $\rightarrow$ stratified Datalog with negation.

Public implementation on PSyKI. [Magnini et al., 2022]

Algorithm II



Algorithm III

Formula	C. interpretation	Formula	C. interpretation
$[\![\neg\phi]\!]$	$\eta(1 - [\![\phi]\!])$	$[\![\phi \leq \psi]\!]$	$\eta([\![\phi]\!] - [\![\psi]\!])$
$[\![\phi \wedge \psi]\!]$	$\eta(max([\![\phi]\!], [\![\psi]\!]))$	$[\![class(\bar{X}, \mathrm{y}_i) \leftarrow \psi]\!]$	$[\![\psi]\!]^*$
$[\![\phi \vee \psi]\!]$	$\eta(min([\![\phi]\!], [\![\psi]\!]))$	$[\![\mathrm{expr}(\bar{X})]\!]$	$\mathrm{expr}([\![\bar{X}]\!])$
$[\![\phi = \psi]\!]$	$\eta(\lvert[\![\phi]\!] - [\![\psi]\!]\rvert)$	$[\![\mathtt{true}]\!]$	0
$[\![\phi \neq \psi]\!]$	$[\![\neg(\phi = \psi)]\!]$	$[\![\mathtt{false}]\!]$	1
$[\![\phi > \psi]\!]$	$\eta(0.5 - [\![\phi]\!] + [\![\psi]\!])$	$[\![X]\!]$	x
$[\![\phi \geq \psi]\!]$	$\eta([\![\psi]\!] - [\![\phi]\!])$	$[\![\mathtt{k}]\!]$	k
$[\![\phi < \psi]\!]$	$\eta(0.5 + [\![\phi]\!] - [\![\psi]\!])$	$[\![p(\bar{X})]\!]^{**}$	$[\![\psi_1 \vee \ldots \vee \psi_k]\!]$

* encodes the penalty for the i^{th} neuron

** assuming predicate p is defined by k clauses of the form:
$p(\bar{X}) \leftarrow \psi_1, \ \ldots, \ p(\bar{X}) \leftarrow \psi_k$

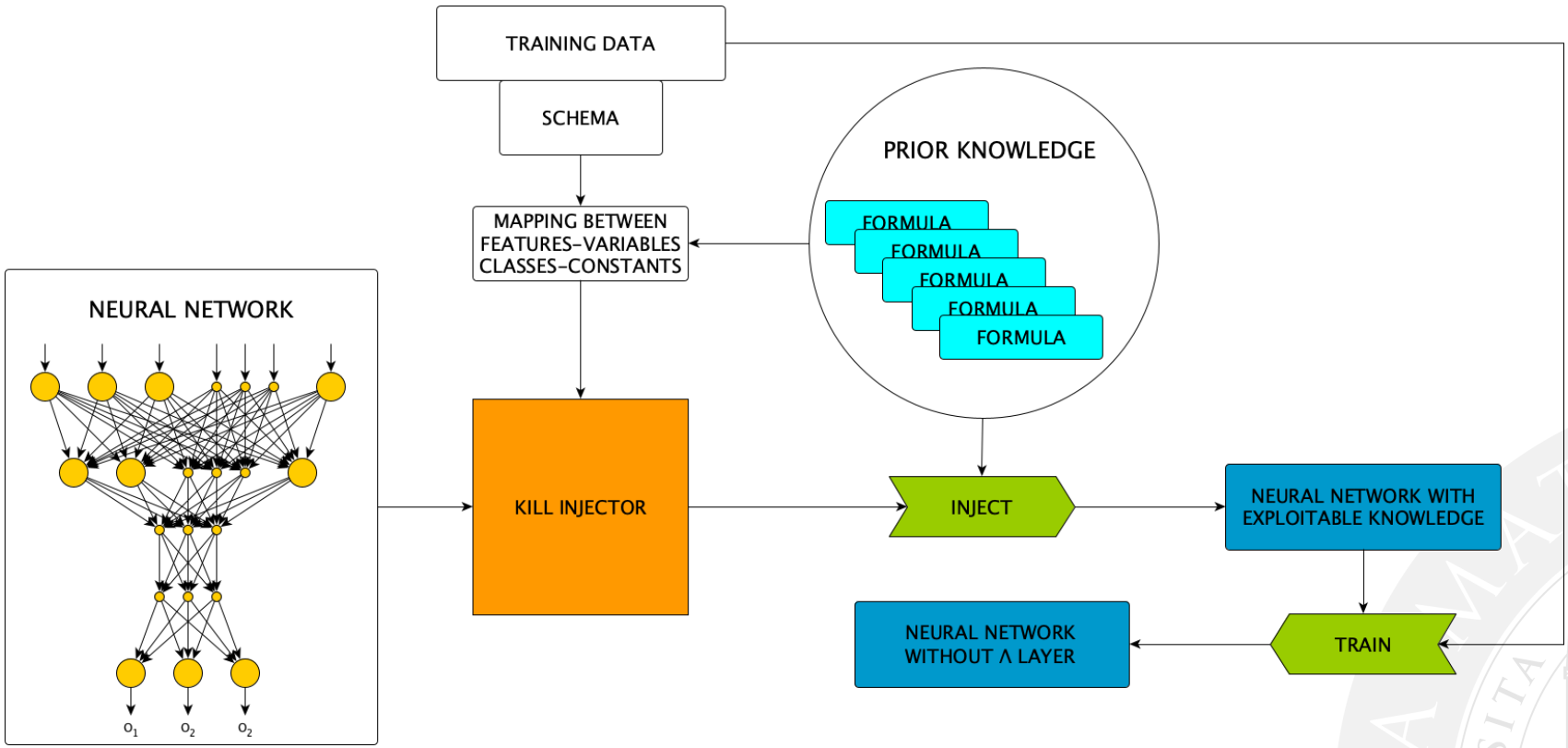
Algorithm IV

Cost function

Whenever the neural network wrongly predicts a class and violates the prior knowledge a cost proportional to the violation is added. In this way the output of the network differs more from the expected one and this affects the back propagation step.

$$Y' = f(Y, cost)$$
$$f = Y \; x \; (\mathbf{1} + cost)$$
$$cost(X, Y) = \eta(p(X) - (\mathbf{1} - Y)) \quad (\mathbf{1} - Y \text{ because 0 means true})$$

Algorithm V


TRAINING DATA
SCHEMA
PRIOR KNOWLEDGE
FORMULA
FORMULA
FORMULA
FORMULA
FORMULA
MAPPING BETWEEN
FEATURES–VARIABLES
CLASSES–CONSTANTS
NEURAL NETWORK
KILL INJECTOR
INJECT
NEURAL NETWORK WITH
EXPLOITABLE KNOWLEDGE
NEURAL NETWORK
WITHOUT Λ LAYER
TRAIN

Next in Line...

Case study I

PHDS: Poker Hand Data Set

Each record represents one poker hand. 5 cards identified by 2 values: suit and rank. Classes: 10. Training set: 25.010. Test set: 1.000.000.

id	S1	R1	S2	R2	S3	R3	S4	R4	S5	R5	class
1	1	10	1	11	1	13	1	12	1	1	9
2	2	11	2	13	2	10	2	12	2	1	9
3	3	12	3	11	3	13	3	10	3	1	9
4	4	10	4	11	4	1	4	13	4	12	9
5	4	1	4	13	4	12	4	11	4	10	9
6	1	2	1	4	1	5	1	3	1	6	8
7	1	9	1	12	1	10	1	11	1	13	8
8	2	1	2	2	2	3	2	4	2	5	8
9	3	5	3	6	3	9	3	7	3	8	8
10	4	1	4	4	4	2	4	3	4	5	8
11	1	1	2	1	3	9	1	5	2	3	1
12	2	6	2	1	4	13	2	4	4	9	0
13	1	10	4	6	1	2	1	1	3	8	0
14	2	13	2	1	4	4	1	5	2	11	0
15	3	8	4	12	3	9	4	2	3	2	1

Case study II

Some injected rules.

Class	Logic Formulation
Pair	$class(R_1, \ldots, S_5, \texttt{pair}) \leftarrow pair(R_1, \ldots, S_5)$
	$pair(R_1, \ldots, S_5) \leftarrow R_1 = R_2$
	$pair(R_1, \ldots, S_5) \leftarrow R_1 = R_3$
	$pair(R_1, \ldots, S_5) \leftarrow R_1 = R_4$
	$pair(R_1, \ldots, S_5) \leftarrow R_1 = R_5$
	$pair(R_1, \ldots, S_5) \leftarrow R_2 = R_3$
	$pair(R_1, \ldots, S_5) \leftarrow R_2 = R_4$
	$pair(R_1, \ldots, S_5) \leftarrow R_2 = R_5$
	$pair(R_1, \ldots, S_5) \leftarrow R_3 = R_4$
	$pair(R_1, \ldots, S_5) \leftarrow R_3 = R_5$
	$pair(R_1, \ldots, S_5) \leftarrow R_4 = R_5$

Case study III

Two Pairs	$class(R_1, \ldots, S_5, \texttt{two}) \leftarrow two(R_1, \ldots, S_5)$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_3 = R_4$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_2 = R_4$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_4 \wedge R_2 = R_3$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_3 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_3 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_5 \wedge R_2 = R_3$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_4 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_4 \wedge R_2 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_5 \wedge R_2 = R_4$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_4 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_4 \wedge R_3 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_1 = R_5 \wedge R_3 = R_4$
	$two(R_1, \ldots, S_5) \leftarrow R_2 = R_3 \wedge R_4 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_2 = R_4 \wedge R_3 = R_5$
	$two(R_1, \ldots, S_5) \leftarrow R_2 = R_5 \wedge R_3 = R_4$

Case study IV

Three of a Kind	$class(R_1, \ldots, S_5, \texttt{three}) \leftarrow three(R_1, \ldots, S_5)$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_3$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_4$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_5$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_1 = R_4$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_1 = R_5$ $three(R_1, \ldots, S_5) \leftarrow R_1 = R_4 \wedge R_1 = R_5$ $three(R_1, \ldots, S_5) \leftarrow R_2 = R_3 \wedge R_2 = R_4$ $three(R_1, \ldots, S_5) \leftarrow R_2 = R_3 \wedge R_2 = R_5$ $three(R_1, \ldots, S_5) \leftarrow R_2 = R_4 \wedge R_2 = R_5$ $three(R_1, \ldots, S_5) \leftarrow R_3 = R_4 \wedge R_3 = R_5$
Flush	$class(R_1, \ldots, S_5, \texttt{flush}) \leftarrow flush(R_1, \ldots, S_5)$ $flush(R_1, \ldots, S_5) \leftarrow S_1 = S_2 \wedge S_1 = S_3 \wedge S_1 = S_4 \wedge S_1 = S_5$
Four of a Kind	$class(R_1, \ldots, S_5, \texttt{four}) \leftarrow four(R_1, \ldots, S_5)$ $four(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_3 \wedge R_1 = R_4$ $four(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_3 \wedge R_1 = R_5$ $four(R_1, \ldots, S_5) \leftarrow R_1 = R_2 \wedge R_1 = R_4 \wedge R_1 = R_5$ $four(R_1, \ldots, S_5) \leftarrow R_1 = R_3 \wedge R_1 = R_4 \wedge R_1 = R_5$ $four(R_1, \ldots, S_5) \leftarrow R_2 = R_3 \wedge R_2 = R_4 \wedge R_2 = R_5$

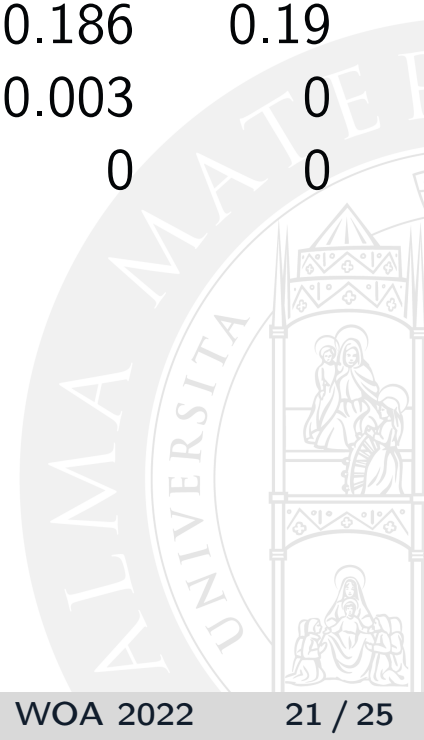
Case study V

Setup

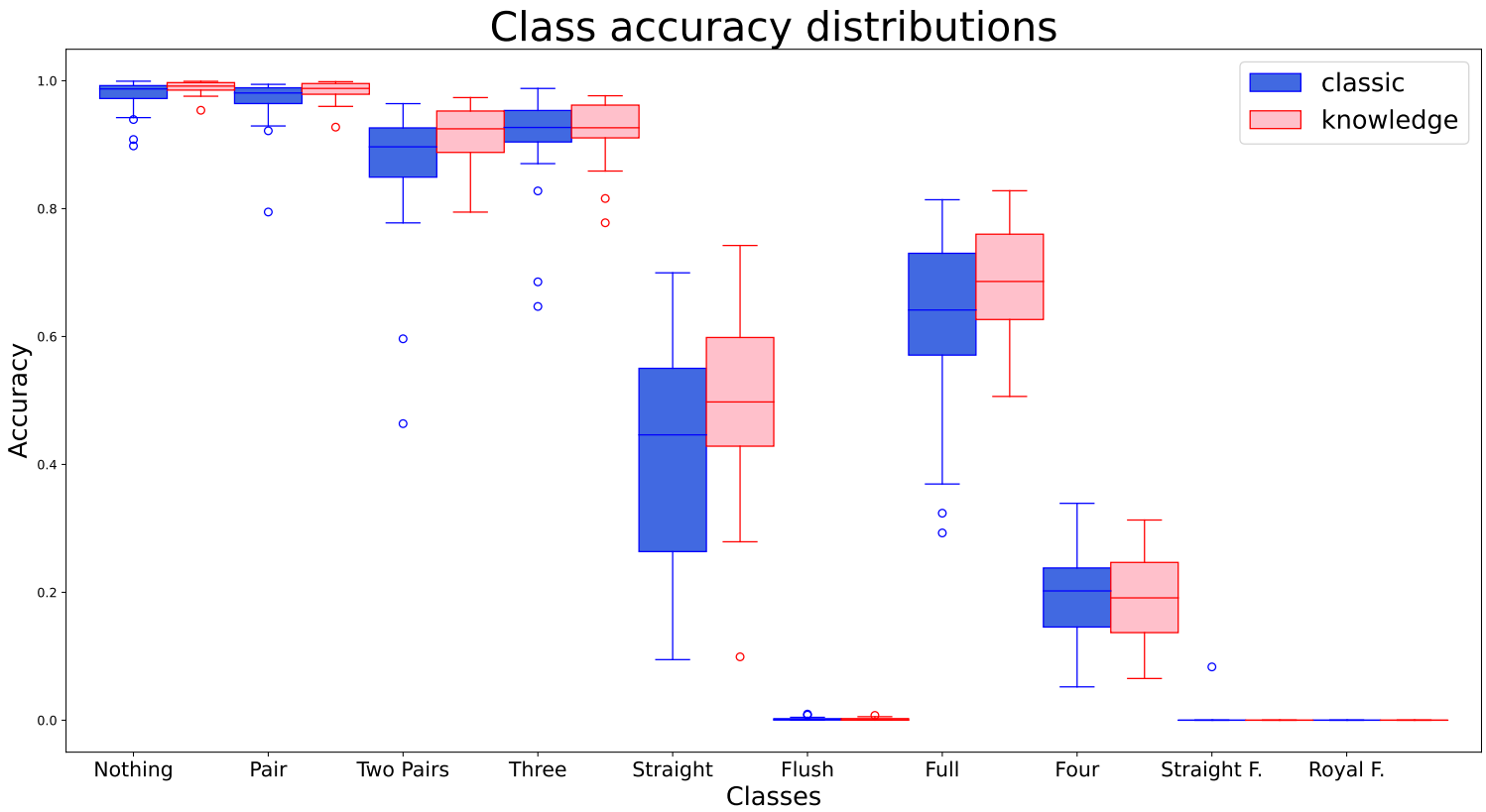
- neural network: 3-layers fully connected (128, 128, 10 neurons per layer respectively) with rectified linear unit (ReLU) as activation function, except for the last layer (softmax);
- knowledge: see previous slides;
- categorical cross-entropy as loss function
- training: Adams as optimiser for 100 epochs (with early stop conditions);
- experiment repeated 30 times to have a statistic significant population.

Case study VI

Metric	Classic	KILL	Metric	Classic	KILL
Accuracy	0.962	0.978	**Acc. Straight**	0.415	0.509
Macro-F1	0.512	0.538	**Acc. Flush**	0.002	0.002
Weighted-F1	0.96	0.977	**Acc. Full**	0.628	0.69
Acc. Nothing	0.977	0.989	**Acc. Four**	0.186	0.19
Acc. Pair	0.968	0.985	**Acc. Straight F.**	0.003	0
Acc. Two Pairs	0.867	0.914	**Acc. Royal F.**	0	0
Acc. Three	0.913	0.922			



Case study VII

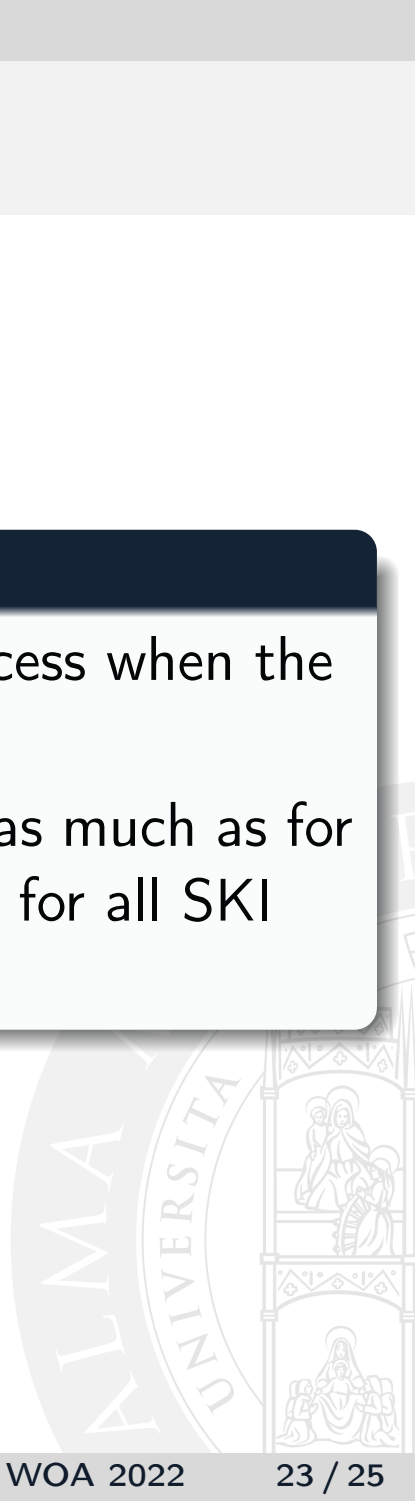


Next in Line...

Discussion and future works I

Discussion

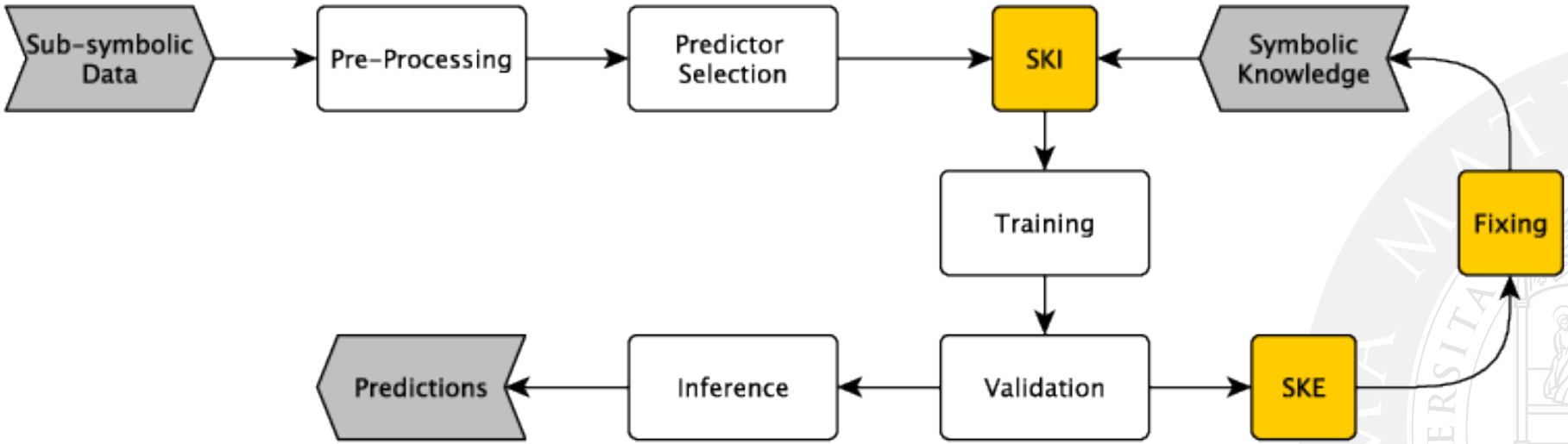
- the case study shows that KILL can be applied with success when the injected knowledge is correct;
- predictions for less represented classes are not improved as much as for more frequent ones (this is very likely a common feature for all SKI algorithms based on constraining).



Discussion and future works II

Future works

- explore scenarios where the injected knowledge is not perfectly correct;
- test the performances of KILL inside the train-extract-fix-inject workflow.



KILL: Knowledge Injection via Lambda Layer

Matteo Magnini* Giovanni Ciatto* Andrea Omicini*

*Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna
{matteo.magnini, giovanni.ciatto, andrea.omicini}@unibo.it

WOA 2022:
Workshop "From Objects to Agents"
September 1–2, 2022, Genova, Italy

References I

[Besold et al., 2017] Besold, T. R., d'Avila Garcez, A. S., Bader, S., Bowman, H., Domingos, P. M., Hitzler, P., Kühnberger, K., Lamb, L. C., Lowd, D., Lima, P. M. V., de Penning, L., Pinkas, G., Poon, H., and Zaverucha, G. (2017).
Neural-symbolic learning and reasoning: A survey and interpretation.
CoRR, abs/1711.03902
http://arxiv.org/abs/1711.03902.

[Calegari et al., 2020] Calegari, R., Ciatto, G., and Omicini, A. (2020).
On the integration of symbolic and sub-symbolic techniques for XAI: A survey.
Intelligenza Artificiale, 14(1):7–32
DOI:10.3233/IA-190036.

[Gunning, 2016] Gunning, D. (2016).
Explainable artificial intelligence (XAI).
Funding Program DARPA-BAA-16-53, Defense Advanced Research Projects Agency (DARPA)
http://www.darpa.mil/program/explainable-artificial-intelligence.

References II

[Magnini et al., 2022] Magnini, M., Ciatto, G., and Omicini, A. (in press, 2022).
On the design of psyki: A platform for symbolic knowledge injection into sub-symbolic predictors.
In Calvaresi, D., Najjar, A., Winikoff, M., and Främling, K., editors, *Explainable and Transparent AI and Multi-Agent Systems - Fourth International Workshop, EXTRAAMAS 2022, Virtual Event, May 9-10, 2022, Revised Selected Papers*, Lecture Notes in Computer Science. Springer
https://apice.unibo.it/xwiki/bin/view/Publications/PsykiExtraamas2022.

[van Gelder, 1990] van Gelder, T. (1990).
Why distributed representation is inherently non-symbolic.
In Dorffner, G., editor, *Konnektionismus in Artificial Intelligence und Kognitionsforschung. Proceedings 6. Österreichische Artificial Intelligence-Tagung (KONNAI), Salzburg, Österreich, 18. bis 21. September 1990*, volume 252 of *Informatik-Fachberichte*, pages 58–66. Springer
DOI:10.1007/978-3-642-76070-9_6.

References III

[Xie et al., 2019] Xie, Y., Xu, Z., Meel, K. S., Kankanhalli, M. S., and Soh, H. (2019).
Embedding symbolic knowledge into deep networks.
In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4235–4245
https://proceedings.neurips.cc/paper/2019/hash/7b66b4fd401a271a1c7224027ce111bc-Abstract.html.