KINS: Knowledge Injection via Network Structuring

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

37th Italian Conference on Computational Logic
(CILC 2022)
29^{th}, June – 1^{st} July, 2022, Bologna, IT

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:

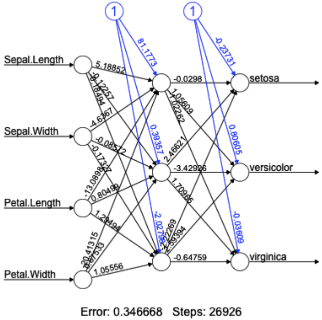
> *any algorithmic procedure affecting how sub-symbolic predictors draw their inferences in such a way that predictions are either computed as a function of, or made consistent with, some given symbolic knowledge.*

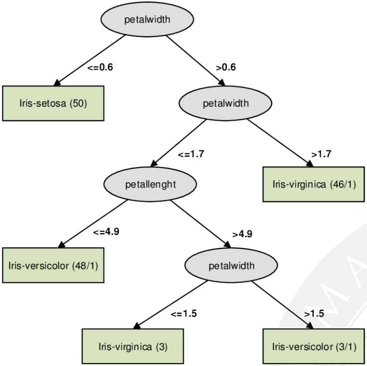
Symbolic Knowledge Injection II

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 III





Symbolic Knowledge Injection IV

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 V

Set of propositional logic rules for the classification task of the well known iris dataset:

$$big\_petal \land average\_sepal \to \texttt{virginica}.$$
$$big\_petal \land \neg average\_sepal \to \texttt{versicolor}.$$
$$big\_petal \to \texttt{setosa}.$$

$$average\_sepal \equiv (3 \leq SepalWidth < 5)$$
$$big\_petal \equiv (PetalLength > 3)$$

Symbolic Knowledge Injection VI

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 VII

Explainability can be achieved [Gunning, 2016]:

Post-hoc explanation

- applying an algorithm of symbolic knowledge extraction on a trained predictor;
- output → 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 → a predictor that does not violate the prior knowledge.

Symbolic Knowledge Injection VIII

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...

1. Premises
2. KINS algorithm
3. Future works

Algorithm I

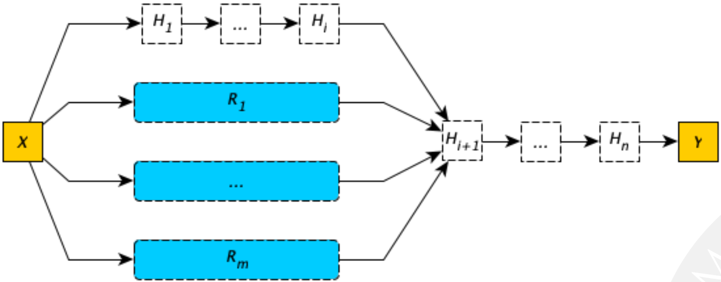
KINS: Knowledge Injection via Network Structuring

A general SKI algorithm that does not impose constrains on the sub-symbolic predictor to enrich.

- aim → enrich;
- predictor → neural network;
- how → structuring;
- logic → stratified Datalog with negation.

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

Algorithm II



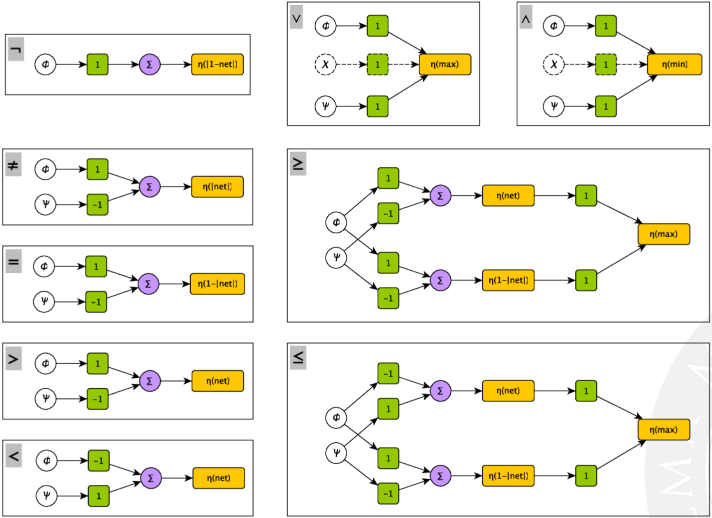
Algorithm III

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

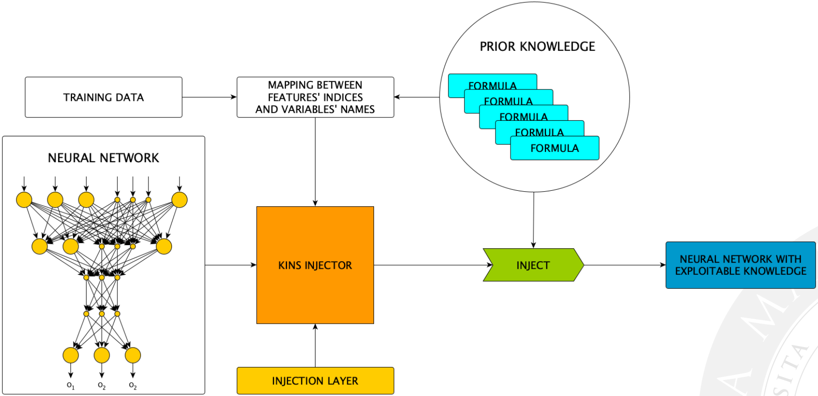
* encodes the value for the i^{th} output

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

Algorithm IV



Algorithm V


PRIOR KNOWLEDGE
FORMULA
FORMULA
FORMULA
FORMULA
FORMULA
TRAINING DATA
MAPPING BETWEEN FEATURES' INDICES AND VARIABLES' NAMES
NEURAL NETWORK
KINS INJECTOR
INJECT
NEURAL NETWORK WITH EXPLOITABLE KNOWLEDGE
INJECTION LAYER

Case study I

PSJGS: Primate Splice-Junction Gene Sequences dataset

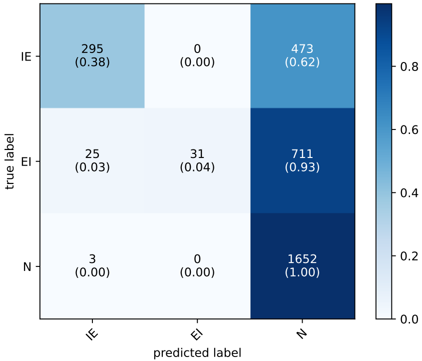
```
EI-stop ::- @-3 'TAA'.
EI-stop ::- @-3 'TAG'.
EI-stop ::- @-3 'TGA'.
EI-stop ::- @-4 'TAA'.
EI-stop ::- @-4 'TAG'.
EI-stop ::- @-4 'TGA'.
EI-stop ::- @-5 'TAA'.
EI-stop ::- @-5 'TAG'.
EI-stop ::- @-5 'TGA'.

IE-stop ::- @1 'TAA'.
IE-stop ::- @1 'TAG'.
IE-stop ::- @1 'TGA'.
IE-stop ::- @2 'TAA'.
IE-stop ::- @2 'TAG'.
IE-stop ::- @2 'TGA'.
IE-stop ::- @3 'TAA'.
IE-stop ::- @3 'TAG'.
IE-stop ::- @3 'TGA'.

pyramidine-rich :- 6 of (@-15 'YYYYYYYYYY').

EI :- @-3 'MAGGTRAGT', not(EI-stop).

IE :- pyramidine-rich, @-3 'YAGG', not(IE-stop).
```

```
Class, Id, DNA-sequence

EI,ATRINS-DONOR-521,CCAGCTGCAT...AGCCAGTCTG
EI,ATRINS-DONOR-905,AGACCCGCCG...GTGCCCCCGC
EI,BABAPOE-DONOR-30,GAGGTGAAGG...CACGGGGATG
...
IE,ATRINS-ACCEPTOR-701,TTCAGCGGCC...GCCCTGTGGA
IE,ATRINS-ACCEPTOR-1678,GGACCTGCTC...GGGGGCTCTA
IE,BABAPOE-ACCEPTOR-801,GCGGTTGATT...AAGATGAAGG
...
N,AGMKPNRSB-NEG-1,CAAAAGAACA...CAAGGCTACA
N,AGMORS12A-NEG-181,AGGGAGGTGT...GGGCATGGGG
N,AGMORS9A-NEG-481,TGGTCAATTC...TCTTGCTCTG
...

3190 Records
```

Case study II

Class	Logic Formulation
EI	$class(\bar{X}, \mathtt{ei}) \leftarrow X_{-3} = \mathtt{m} \wedge X_{-2} = \mathtt{a} \wedge X_{-1} = \mathtt{g} \wedge X_{+1} = \mathtt{g} \wedge$ $X_{+2} = \mathtt{t} \wedge X_{+3} = \mathtt{a} = \mathtt{r} \wedge X_{+4} = \mathtt{a} \wedge$ $X_{+5} = \mathtt{g} \wedge X_{+6} = \mathtt{t} \wedge \neg(ei\_stop(\bar{X}))$ $ei\_stop(\bar{X}) \leftarrow X_{-3} = \mathtt{t} \wedge X_{-2} = \mathtt{a} \wedge X_{-1} = \mathtt{a}$ $ei\_stop(\bar{X}) \leftarrow X_{-3} = \mathtt{t} \wedge X_{-2} = \mathtt{a} \wedge X_{-1} = \mathtt{g}$ $ei\_stop(\bar{X}) \leftarrow X_{-3} = \mathtt{t} \wedge X_{-2} = \mathtt{g} \wedge X_{-1} = \mathtt{a}$ $ei\_stop(\bar{X}) \leftarrow X_{-4} = \mathtt{t} \wedge X_{-3} = \mathtt{a} \wedge X_{-2} = \mathtt{a}$ $ei\_stop(\bar{X}) \leftarrow X_{-4} = \mathtt{t} \wedge X_{-3} = \mathtt{a} \wedge X_{-2} = \mathtt{g}$ $ei\_stop(\bar{X}) \leftarrow X_{-4} = \mathtt{t} \wedge X_{-3} = \mathtt{g} \wedge X_{-2} = \mathtt{a}$ $ei\_stop(\bar{X}) \leftarrow X_{-5} = \mathtt{t} \wedge X_{-4} = \mathtt{a} \wedge X_{-3} = \mathtt{a}$ $ei\_stop(\bar{X}) \leftarrow X_{-5} = \mathtt{t} \wedge X_{-4} = \mathtt{a} \wedge X_{-3} = \mathtt{g}$ $ei\_stop(\bar{X}) \leftarrow X_{-5} = \mathtt{t} \wedge X_{-4} = \mathtt{g} \wedge X_{-3} = \mathtt{a}$
IE	$class(\bar{X}, \mathtt{ie}) \leftarrow pyramidine\_rich(\bar{X}) \wedge \neg(ie\_stop(\bar{X})) \wedge$ $X_{-3} = \mathtt{y} \wedge X_{-2} = \mathtt{a} \wedge X_{-1} = \mathtt{g} \wedge X_{+1} = \mathtt{g}$ $pyramidine\_rich(\bar{X}) \leftarrow 6 \leq (X_{-15} = \mathtt{y} + \ldots + X_{-6} = \mathtt{y})$ $ie\_stop(\bar{X}) \leftarrow X_{+2} = \mathtt{t} \wedge X_{+3} = \mathtt{a} \wedge X_{+4} = \mathtt{a}$ $ie\_stop(\bar{X}) \leftarrow X_{+2} = \mathtt{t} \wedge X_{+3} = \mathtt{a} \wedge X_{+4} = \mathtt{g}$ $ie\_stop(\bar{X}) \leftarrow X_{+2} = \mathtt{t} \wedge X_{+3} = \mathtt{g} \wedge X_{+4} = \mathtt{a}$ $ie\_stop(\bar{X}) \leftarrow X_{+3} = \mathtt{t} \wedge X_{+4} = \mathtt{a} \wedge X_{+5} = \mathtt{a}$ $ie\_stop(\bar{X}) \leftarrow X_{+3} = \mathtt{t} \wedge X_{+4} = \mathtt{a} \wedge X_{+5} = \mathtt{g}$ $ie\_stop(\bar{X}) \leftarrow X_{+3} = \mathtt{t} \wedge X_{+4} = \mathtt{g} \wedge X_{+5} = \mathtt{a}$ $ie\_stop(\bar{X}) \leftarrow X_{+4} = \mathtt{t} \wedge X_{+5} = \mathtt{a} \wedge X_{+6} = \mathtt{a}$ $ie\_stop(\bar{X}) \leftarrow X_{+4} = \mathtt{t} \wedge X_{+5} = \mathtt{a} \wedge X_{+6} = \mathtt{g}$ $ie\_stop(\bar{X}) \leftarrow X_{+4} = \mathtt{t} \wedge X_{+5} = \mathtt{g} \wedge X_{+6} = \mathtt{a}$

Case study III



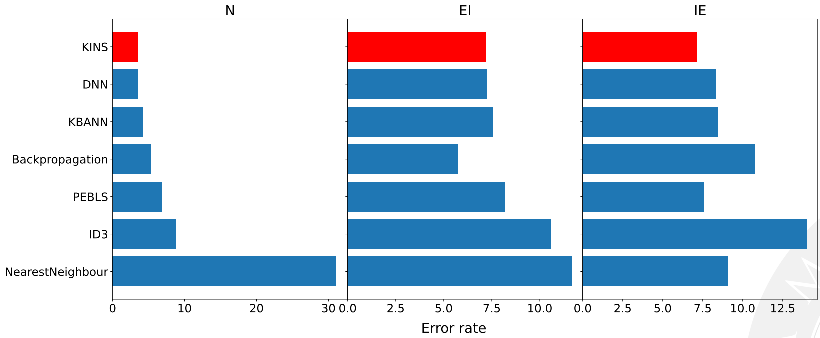
Case study IV

```
set_seed(self.seed)
# Loading dataset and apply one-hot encoding for each feature
# This means that for feature i_th we have 4 new features,
# one for each base: i_th_a, i_th_c, i_th_g, i_th_t.
data = get_splice_junction_data('data')
y = data_to_int(data.iloc[:, -1:], CLASS_MAPPING)
x = get_binary_data(data.iloc[:, :-1], AGGREGATE_FEATURE_MAPPING)
y.columns = [x.shape[1]]
data = x.join(y)
# Loading rules and conversion in Datalog form
rules = get_splice_junction_rules('kb')
rules = get_splice_junction_datalog_rules(rules)
rules = get_binary_datalog_rules(rules)
rules = [get_formula_from_string(rule) for rule in rules]
# Creation of the base model
model = create_fully_connected_nn_with_dropout()
injector = NetworkComposer(model, get_splice_junction_extended_feature_mapping()) # aka KINS!
result = k_fold_cross_validation(data, injector, rules, seed=self.seed)
result.to_csv(self.file + '.csv', sep=';')
```

Case study V

- neural network: 3-layers fully connected (64, 32, 3 neurons per layer respectively) with a 20% of dropout;
- mapping between features and variables: a map where keys are variables' names (e.g., $X_{-30}a, X_{-30}c, X_{-30}g, X_{-30}t, X_{-29}a, \ldots, X_{+30}t$) and features' indices (e.g. $0, 1, \ldots, 239$);
- injection layer: layer 0;
- knowledge: see slide 15;
- training: Adams as optimiser for 100 epochs (with early stop conditions);

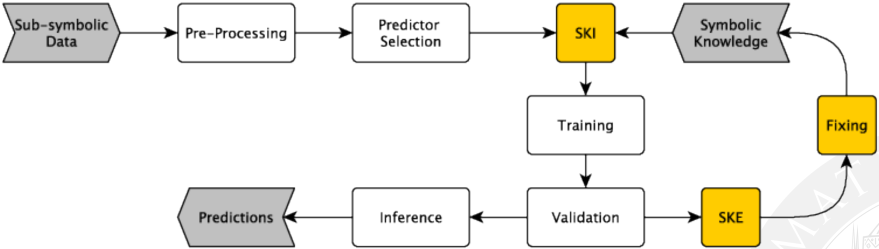
Case study VI



Next in Line...

1. Premises
2. KINS algorithm
3. Future works

TEFI: train-extract-fix-inject



KINS: Knowledge Injection via Network Structuring

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

37th Italian Conference on Computational Logic
(CILC 2022)
29^{th}, June – 1^{st} July, 2022, Bologna, IT

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.