SKI: Symbolic Knowledge Injection

state of the art and our current works

Matteo Magnini
matteo.magnini@unibo.it

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

24-05-2022

Definition

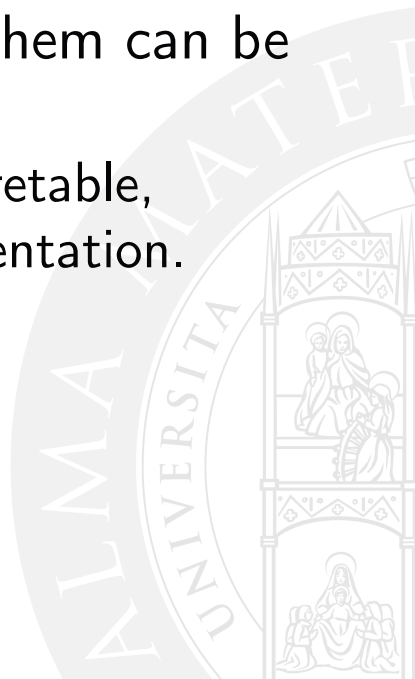
We define symbolic knowledge injection as:

> *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

A symbolic representation consists of:

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.



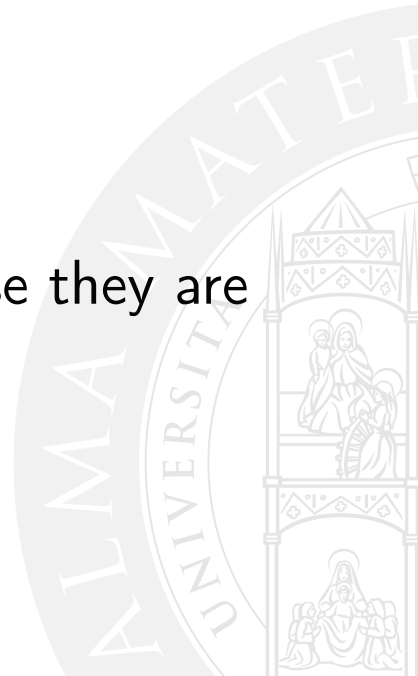
Sub-symbolic data

- ML methods, and sub-symbolic approaches in general, represent data as arrays of real numbers, and knowledge as functions over such data;
- despite numbers are technically symbols as well, we cannot consider arrays and their functions as symbolic knowledge representation (KR) means;
- sub-symbolic approaches frequently violate Items 2 and 3.

Sub-symbolic predictors

- deep neural networks (DNN);
  - convolutional neural networks (CNN),
  - recurrent neural networks (RNN);
- kernel machines;
- others.

The vast majority of predictors are NN most probably because they are easy to manipulate and they have top performances.



Why SKI?

There are several benefits:

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

Aim

Enrich (learning support)

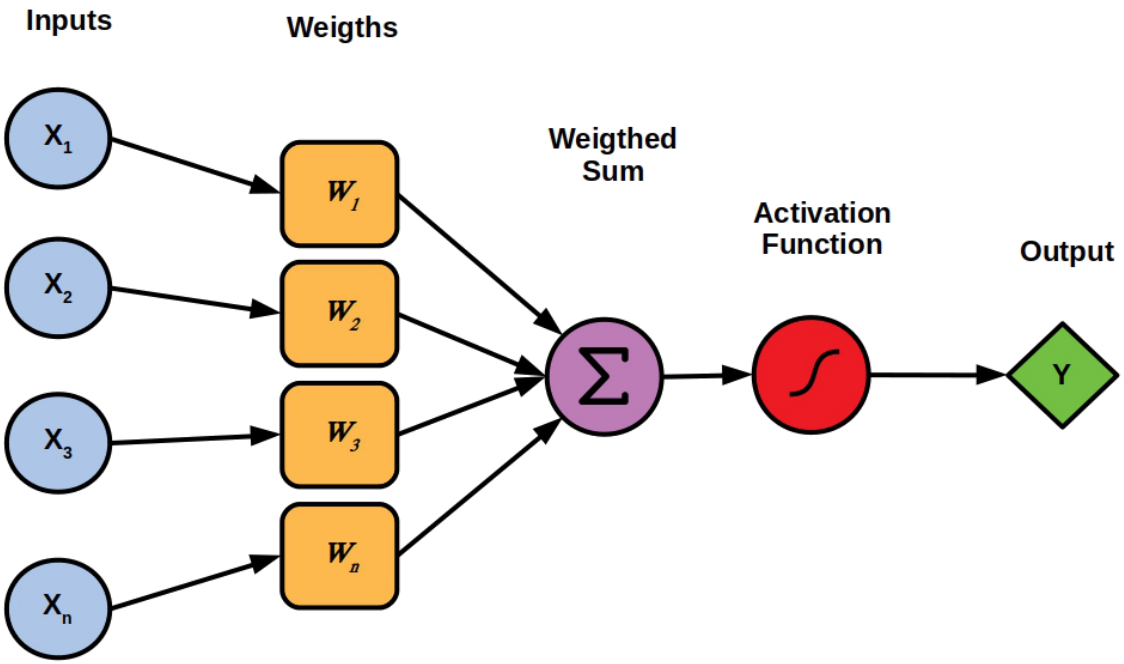
- reduce learning time;
- reduce the data size needed for training;
- improve predictor's accuracy.

Manifold (symbolic knowledge manipulation)

- logic inference;
- information retrieval;
- knowledge base completion/fusion.

Predictors

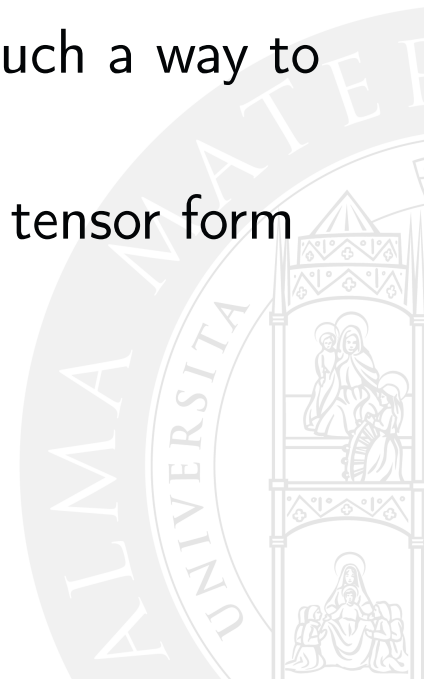
- theoretically, one can inject prior knowledge into any sub-symbolic predictor;
- in practice, NN are almost the sole predictors treated in literature;
- however, lot of different NN architecture are considered.



How

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

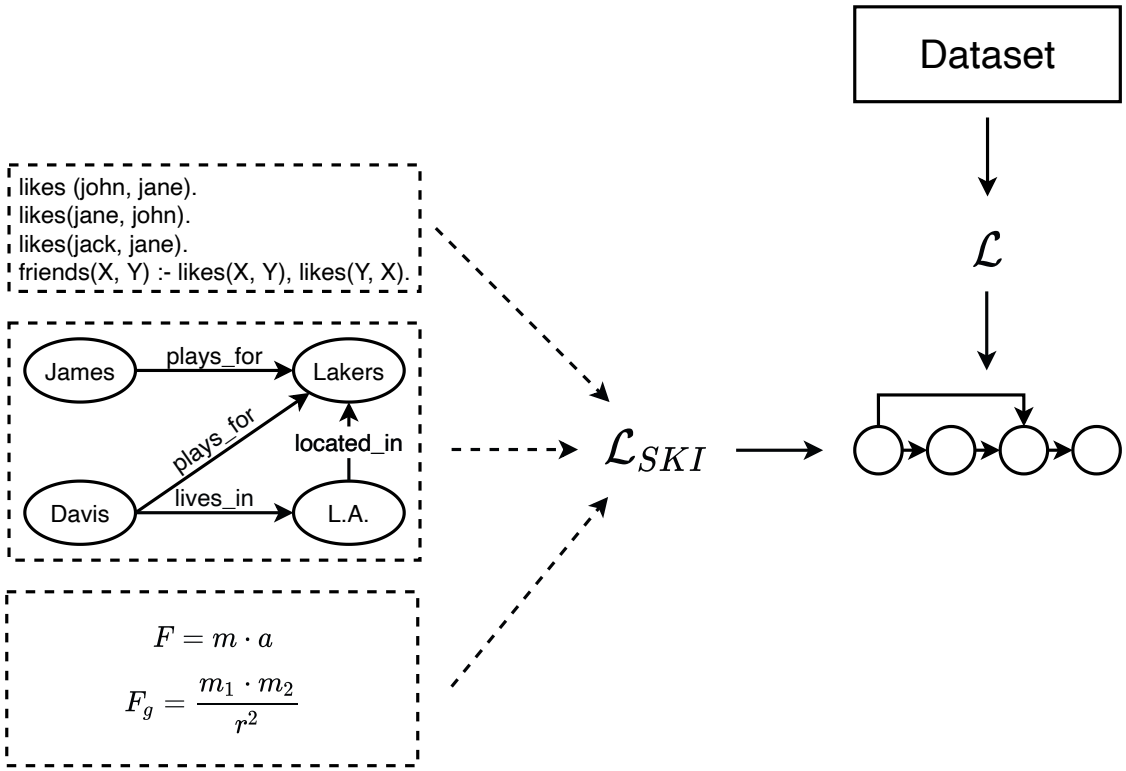
- constraining, a cost factor proportional to the violation of the knowledge is introduced during learning;
- structuring, the architecture of the predictor is built in such a way to mimic the knowledge;
- embedding, the symbolic knowledge is embedded into a tensor form and it is given in input as training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



Constraining

- Knowledge cost factor is introduced in the loss function;
- for NN the cost affects backpropagation during training.

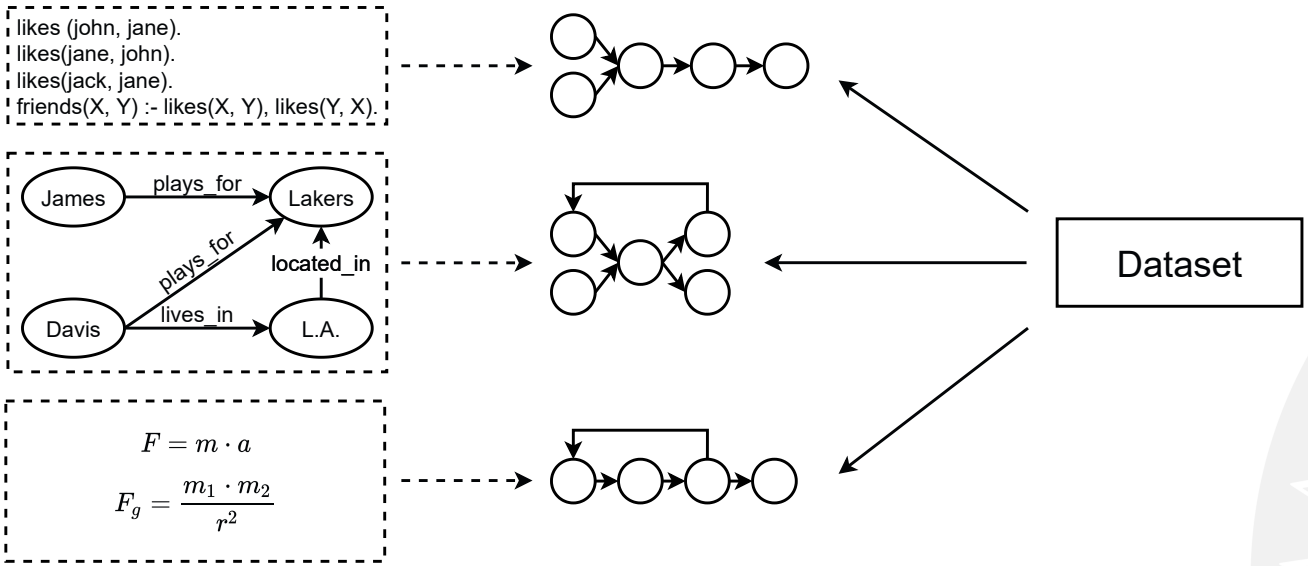
  $\Rightarrow$ Predictor does not violate the prior knowledge (to a certain extent)



Structuring

- Inner architecture is shaped to be able to "mimic" the knowledge;
- for NN this means *ad-hoc* layers.

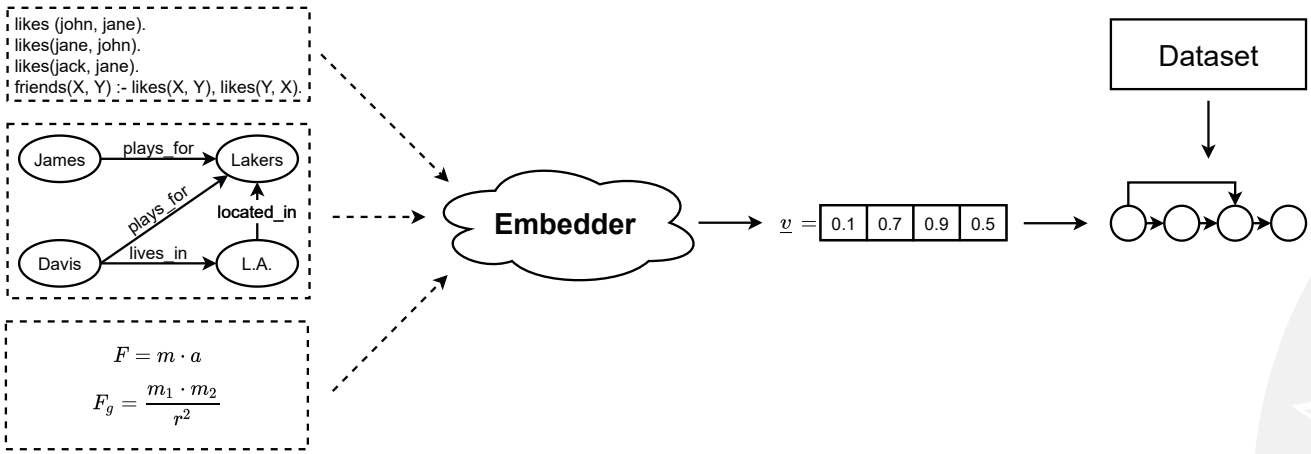
  $\Rightarrow$ Predictor directly exploits knowledge when needed.



Embedding

- Symbolic knowledge is embedded into a tensor form;
- this is used as predictor's input data (alone or with a "standard" dataset).

  $\Rightarrow$ Predictor's aim is manifold in most cases.



Logics

- first order logic (FOL);
- knowledge graph (KG);
- propositional logic (PL).

FOL

- FOL is extremely flexible and expressive;
- you can use recursion and define recursive structures;
- maybe too "powerful" for canonic NN.
  - $\Rightarrow$ Most NN are natively DAG (directed acyclic graph)
    - this allow backpropagation as training algorithm but ...
    - how can you support recursion?

FOL

- FOL is extremely flexible and expressive;
- you can use recursion and define recursive structures;
- maybe too "powerful" for canonic NN.
  - $\Rightarrow$ Most NN are natively DAG (directed acyclic graph)
    - this allow backpropagation as training algorithm but ...
    - how can you support recursion?

You can't! Unless you use some tricks.

KG

- Only constants, variables and n-ary predicates with $n < 3$;
- collections of triplets $\langle \mathtt{a}\ f\ \mathtt{b} \rangle$ or $f(\mathtt{a}, \mathtt{b})$
- essentially directed graph:
  - nodes $\Rightarrow$ individuals,
  - vertices $\Rightarrow$ properties connecting individuals;
- may instantiate an ontology, i.e., a formal description of classes characterising a given domain.

PL

- No quantifiers, terms, and non-atomic predicates;
- expressions involving one or many 0-ary predicates (propositions) possibly interconnected by ordinary logic connectives;
- low expressiveness, but easy to work with.

First works

Notable works

SKI workflow

SKE & SKI

Other scientific fields

SKI: Symbolic Knowledge Injection

state of the art and our current works

Matteo Magnini
matteo.magnini@unibo.it

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

24-05-2022

References