Towards Explainability in Knowledge Enhanced Neural Networks

Data Science Master Thesis

Riccardo Mazzieri
September 21, 2021

Università degli Studi di Padova

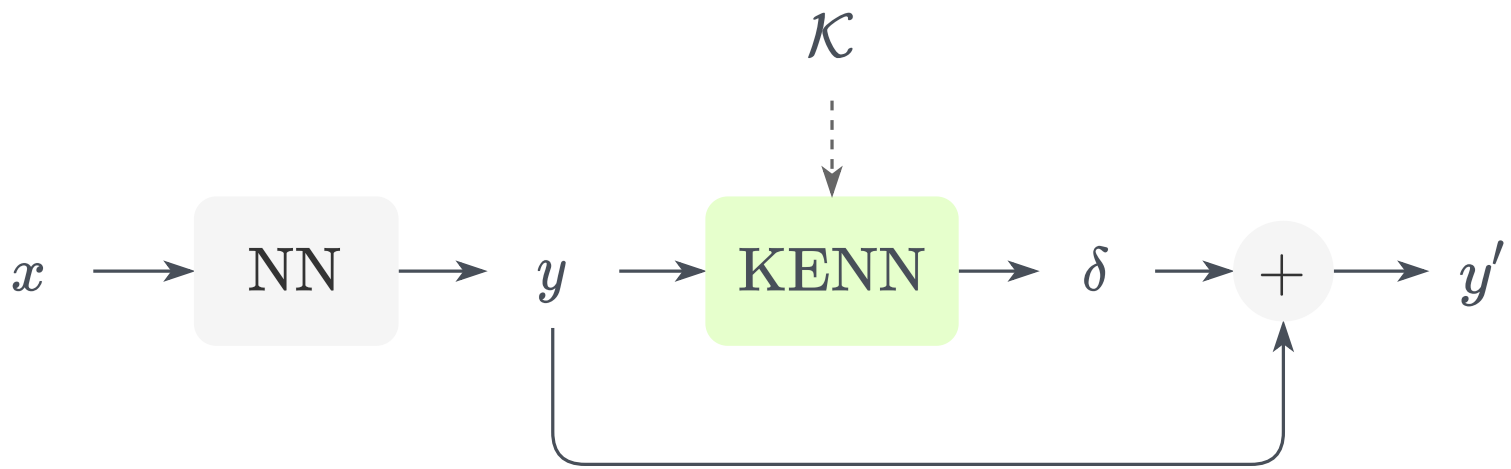
Outline

Introduction

Deep NNs have several flaws. For example:

- They are **data hungry**:
  - With few data, learning is not possible, even for simple logical reasoning tasks;
  - This motivates **Neural Symbolic Integration (NeSy)**.
- They are **black boxes**:
  - Decisions are not explainable, might lead to lack of trust in AI applications;
  - This motivates the research field of **Explainable AI (XAI)**.

KENN consists in a residual layer designed to improve the predictions of a base NN, by using logical prior knowledge, consisting in a set of FOL formulas $\mathcal{K}$.



Basic Terminology

Definition (The Language)

Our language will be a function-free first order language $\mathcal{L}$, defined by:

- A set of **constants**: $\mathcal{C} = \{a_1, \ldots, a_{|\mathcal{C}|}\}$;
- A set of **predicates**: $\mathcal{P} = \{P_1, \ldots, P_{|\mathcal{P}|}\}$;

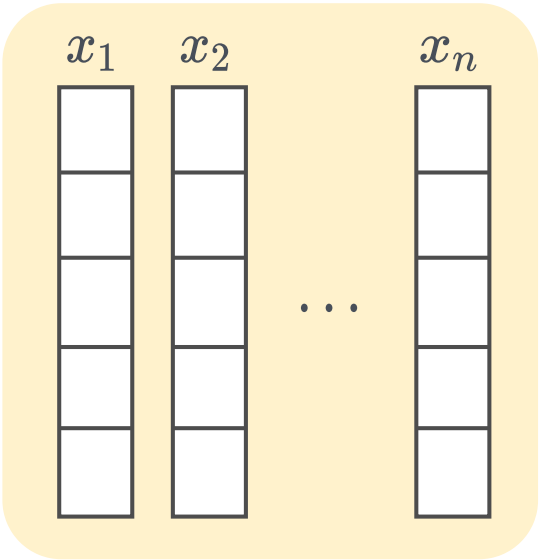
Definition (Clause)

A clause c is a formula expressed a disjunction of literals:

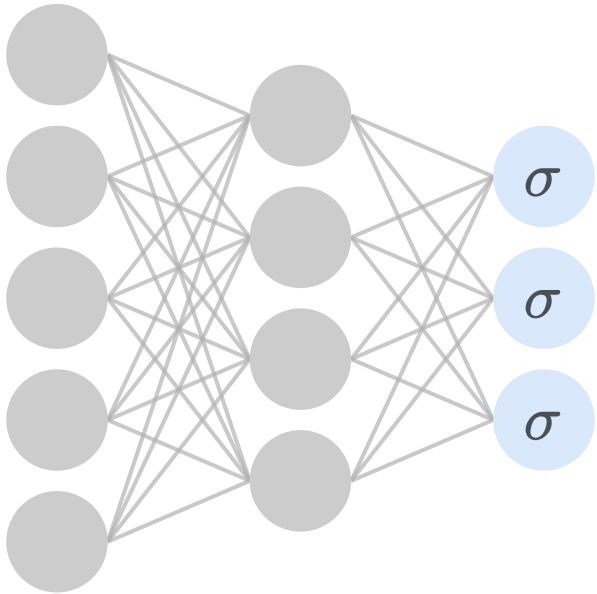
$$c := \bigvee_{i=1}^{k} l_i, \quad l_i \neq l_j \quad \forall i \neq j$$

Language Semantic

$$\mathcal{C} = \{a_1, a_2, \ldots, a_n\} \qquad \mathcal{P} = \{P_1, P_2, P_3\}$$

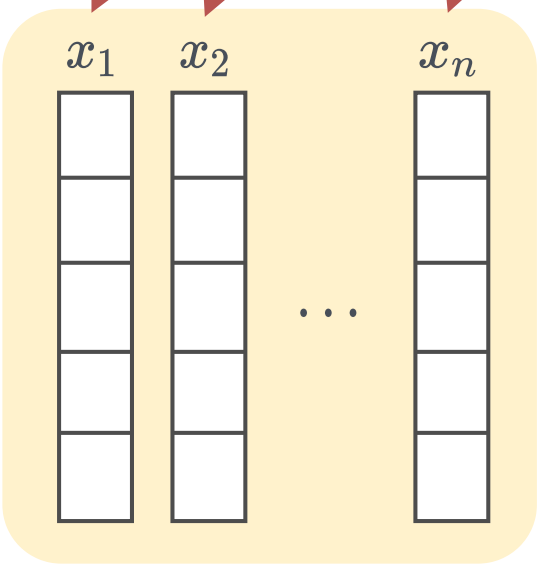


$$x_i \in \mathbb{R}^m$$



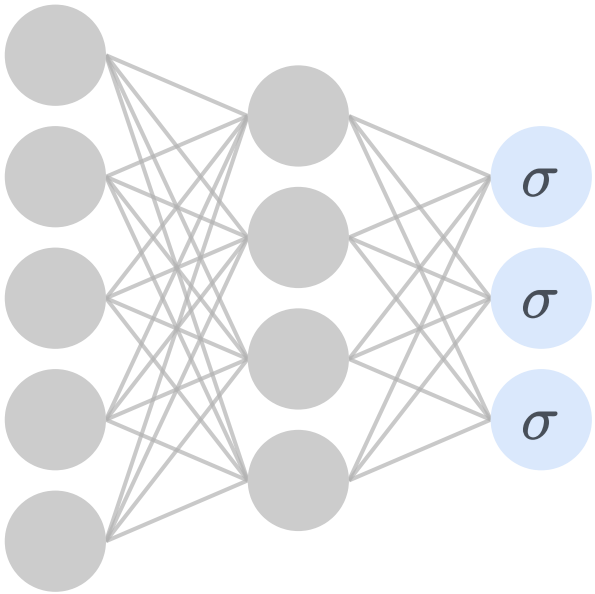
Language Semantic

$$\mathcal{C} = \{a_1, a_2, \ldots, a_n\}$$



$$x_i \in \mathbb{R}^m$$

$$\mathcal{P} = \{P_1, P_2, P_3\}$$



$$P_1(x) \in [0, 1]$$

$$P_2(x) \in [0, 1]$$

$$P_3(x) \in [0, 1]$$

Language Semantic

Note that truth values can be any real number in $[0, 1]$
$\Rightarrow$ We will follow the rules of **Fuzzy Logic**, an extension of Boolean Logic. Specifically, in Fuzzy logic:

Language Semantic

Note that truth values can be any real number in $[0, 1]$
$\Rightarrow$ We will follow the rules of **Fuzzy Logic**, an extension of Boolean Logic. Specifically, in Fuzzy logic:

- Truth value of a negated predicate is simply $\mathcal{I}(\neg A(x)) = 1 - \mathcal{I}(A(x))$

Language Semantic

Note that truth values can be any real number in $[0, 1]$
$\Rightarrow$ We will follow the rules of **Fuzzy Logic**, an extension of Boolean Logic. Specifically, in Fuzzy logic:

- Truth value of a negated predicate is simply $\mathcal{I}(\neg A(x)) = 1 - \mathcal{I}(A(x))$
- Truth value of a disjunction of literals is computed by means of a t-**conorm** function;

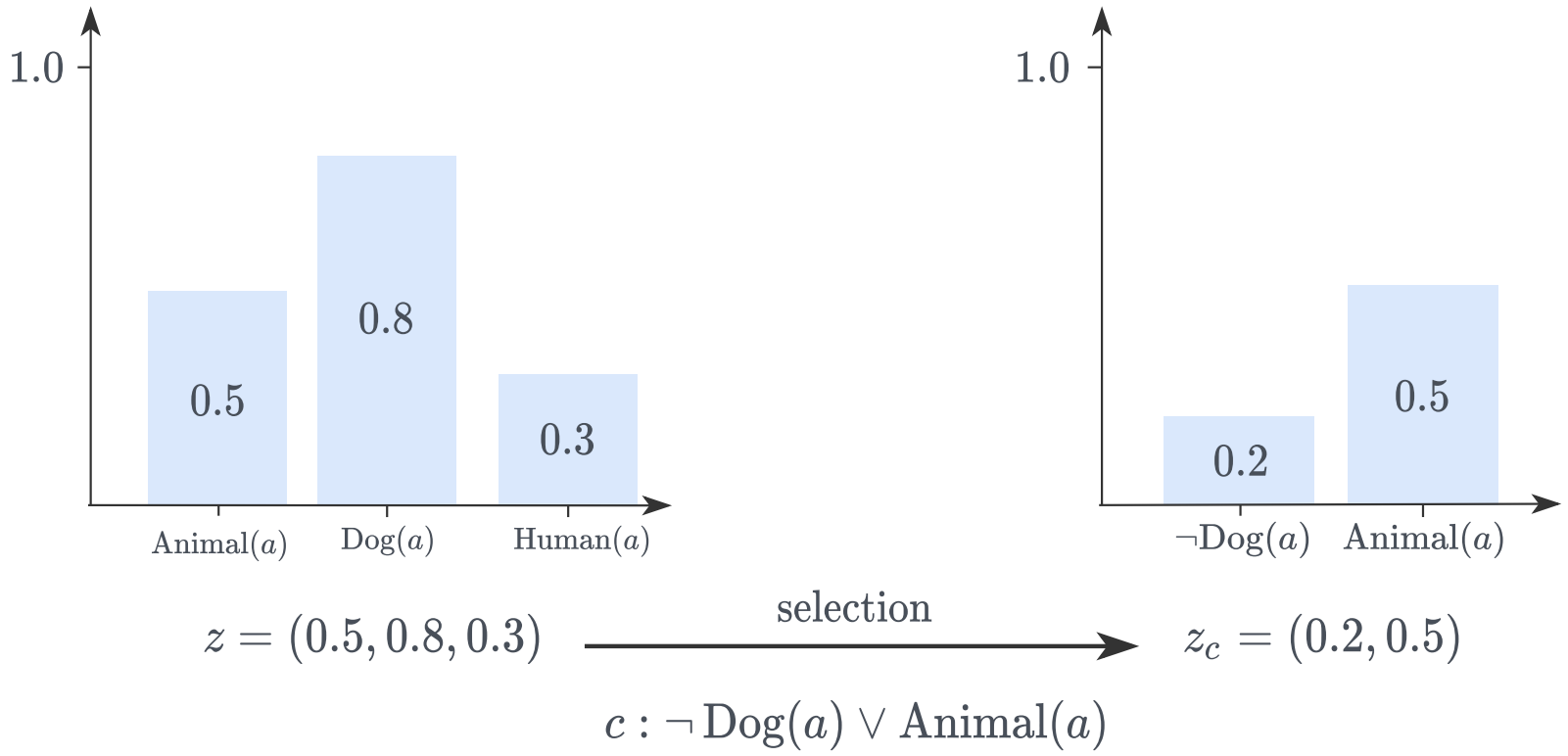
Language Semantic

Note that truth values can be any real number in $[0, 1]$
$\Rightarrow$ We will follow the rules of **Fuzzy Logic**, an extension of Boolean Logic. Specifically, in Fuzzy logic:

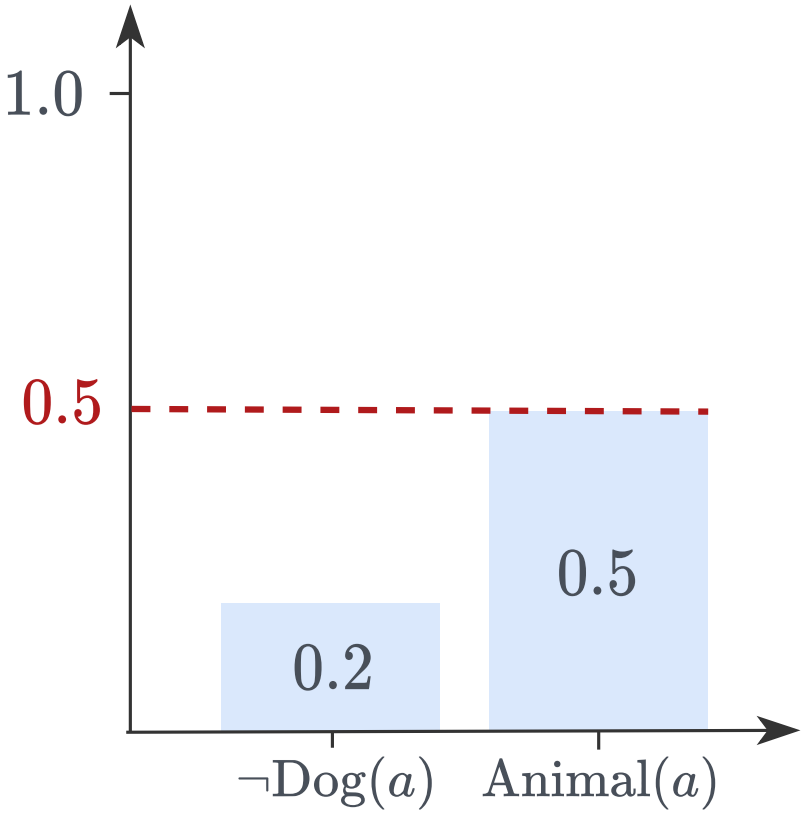
- Truth value of a negated predicate is simply $\mathcal{I}(\neg A(x)) = 1 - \mathcal{I}(A(x))$
- Truth value of a disjunction of literals is computed by means of a t-**conorm** function;
- KENN uses the Gödel t-conorm, which is defined as:

$$\perp_{\max}(t) = \max_{i=1,\ldots,m} t_i, \quad \forall t \in [0, 1]^m.$$

Example: truth value of a clause



Example: truth value of a clause



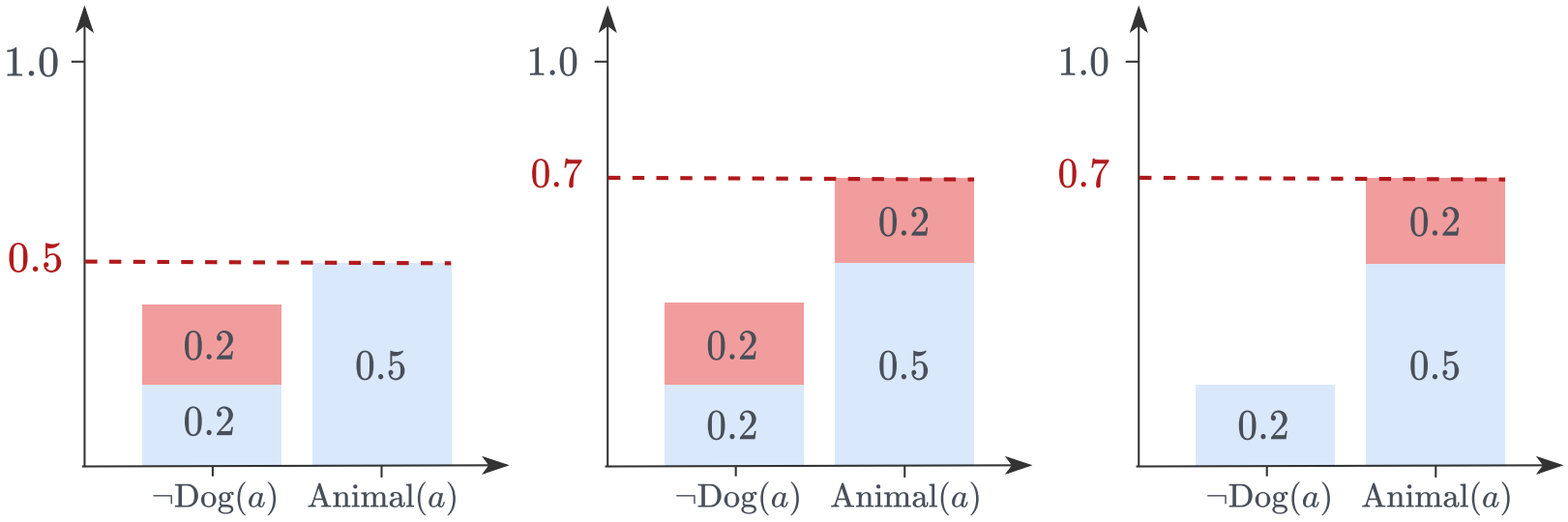
$$\perp_{\max}(0.2, 0.5) = \max(0.2, 0.5) = 0.5$$

Increasing satisfaction of a single clause

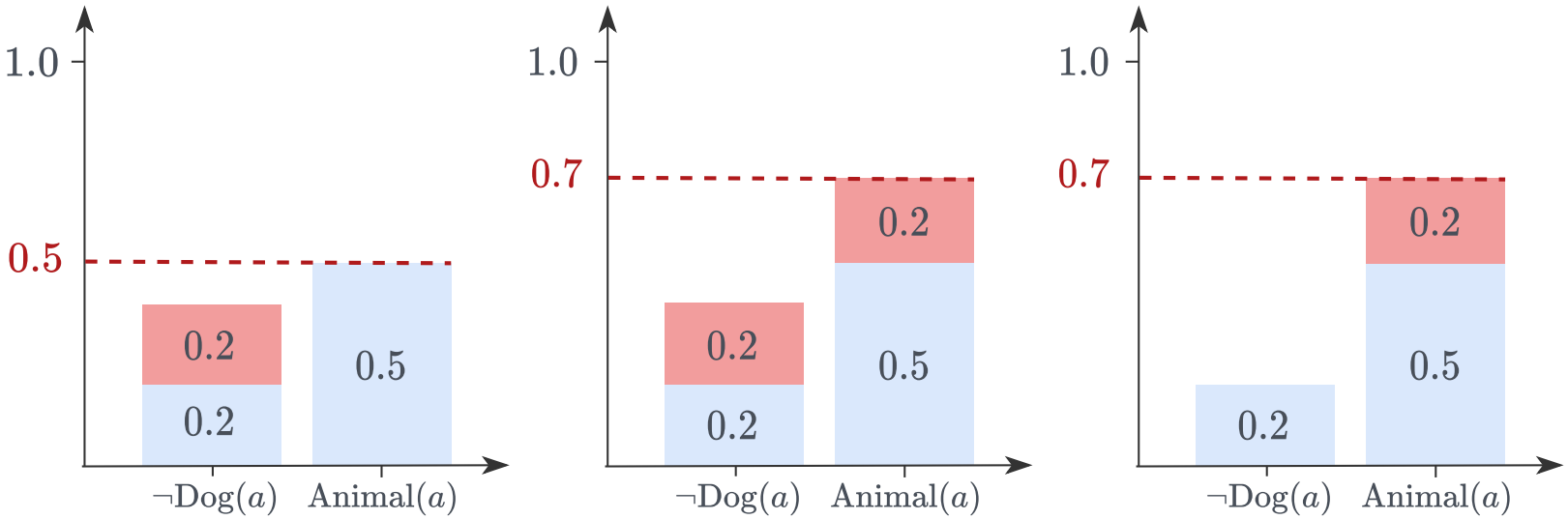
Given the vector of predictions of the NN y and given $c \in \mathcal{K}$, KENN computes a vector of changes δ^c, such that the final vector of predictions $y' = y + \delta^c$:

- Improves the truth value of c,
- Keeps the quantity $\|y' - y\|_2$ minimal.

Increasing satisfaction of a single clause



Increasing satisfaction of a single clause



$$\delta_s^{w_c}(z_c) = w_c \cdot \text{softmax}(z_c)$$

Boosting Preactivations

In practice, inside KENN the delta vectors are applied to the **preactivations** from the NN:

$$y' = \sigma(z + \delta^c).$$

Boosting Preactivations

In practice, inside KENN the delta vectors are applied to the **preactivations** from the NN:

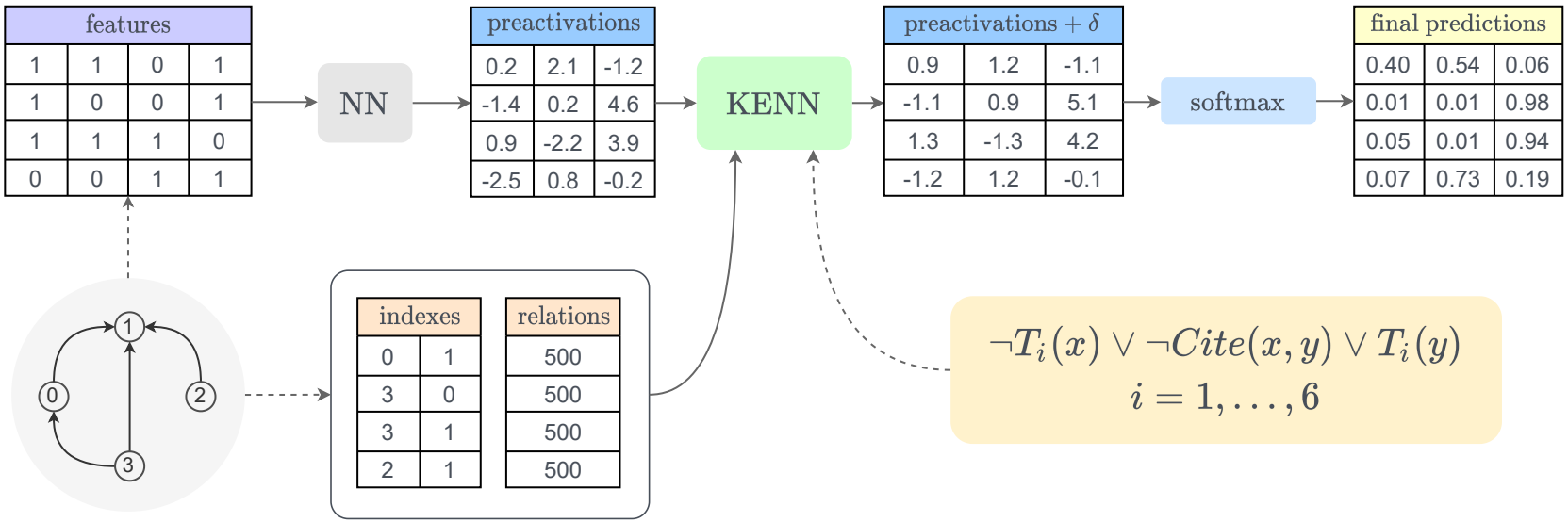
$$y' = \sigma(z + \delta^c).$$

Then, in order to increase the satisfaction of the entire knowledge, all the deltas are aggregated by being summed together. The final prediction will be:

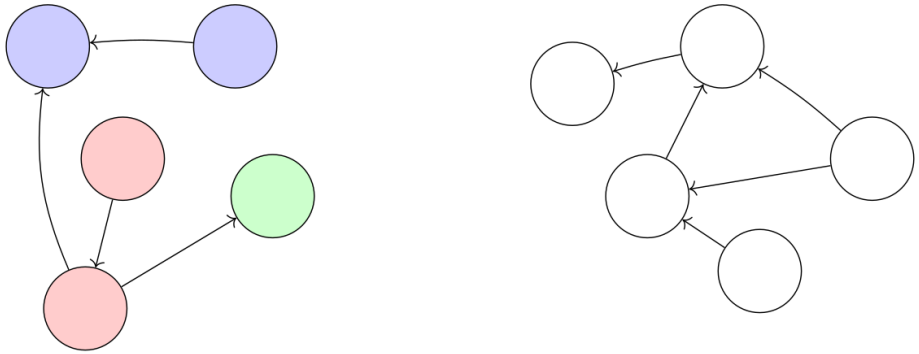
$$y' = \sigma\left(z + \sum_{c \in \mathcal{K}} \delta^c\right).$$

Citeseer Experiments

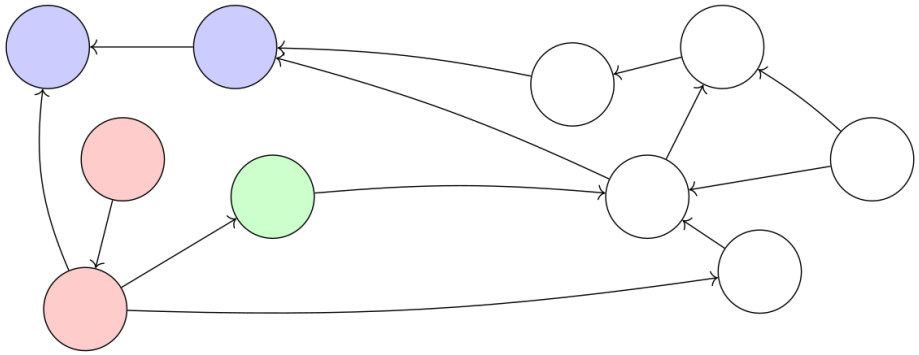
- We tested KENN on a Collective Classification task;
- The **Citeseer Dataset** was used: citation network with 4732 citations (edges) between 3312 papers (nodes);
- The task is to predict the topic of each paper.



Learning Paradigms



Inductive Learning



Transductive Learning

Results Inductive Paradigm

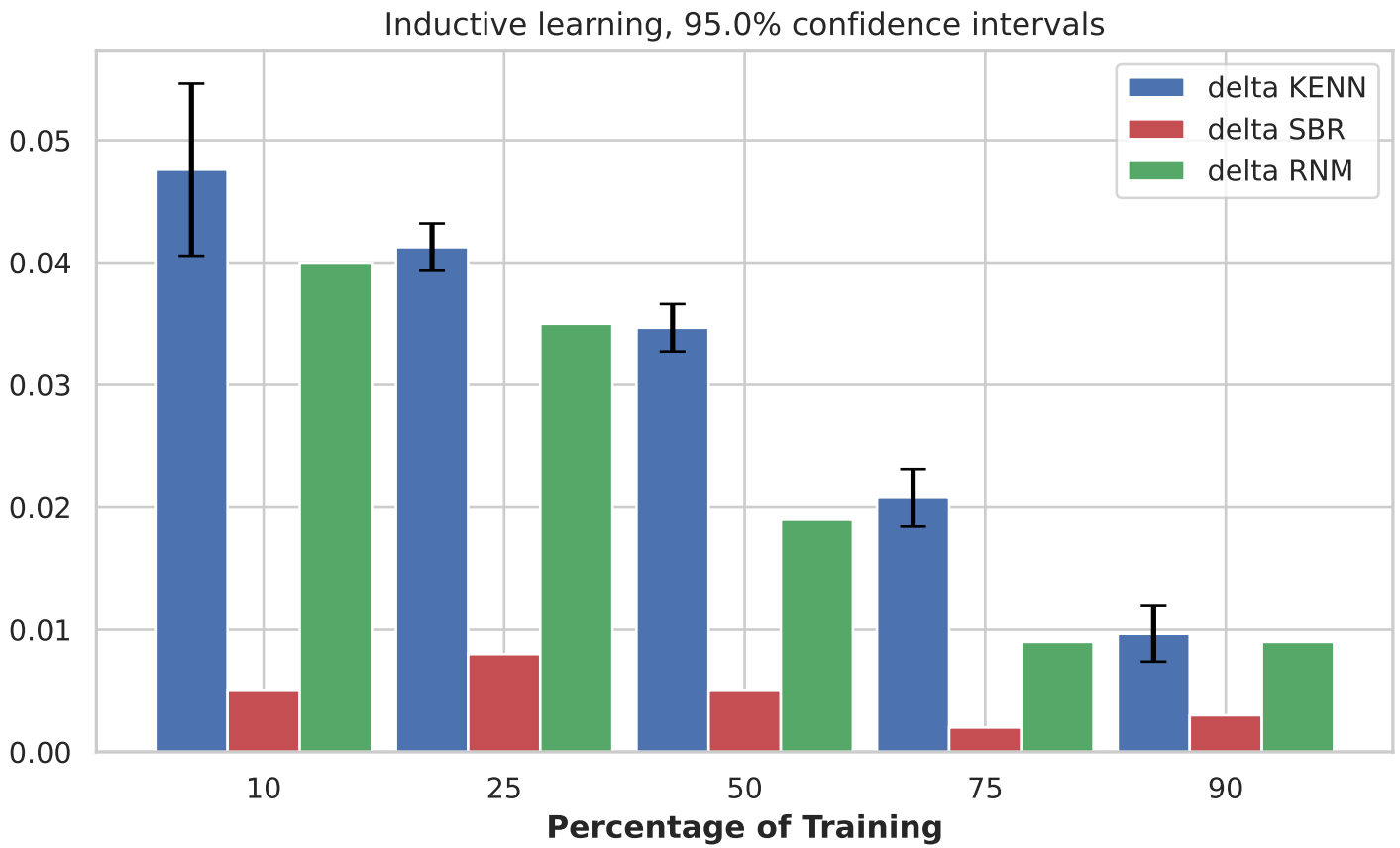


Figure: Relative improvements for the Inductive Paradigm

Results Transductive Paradigm

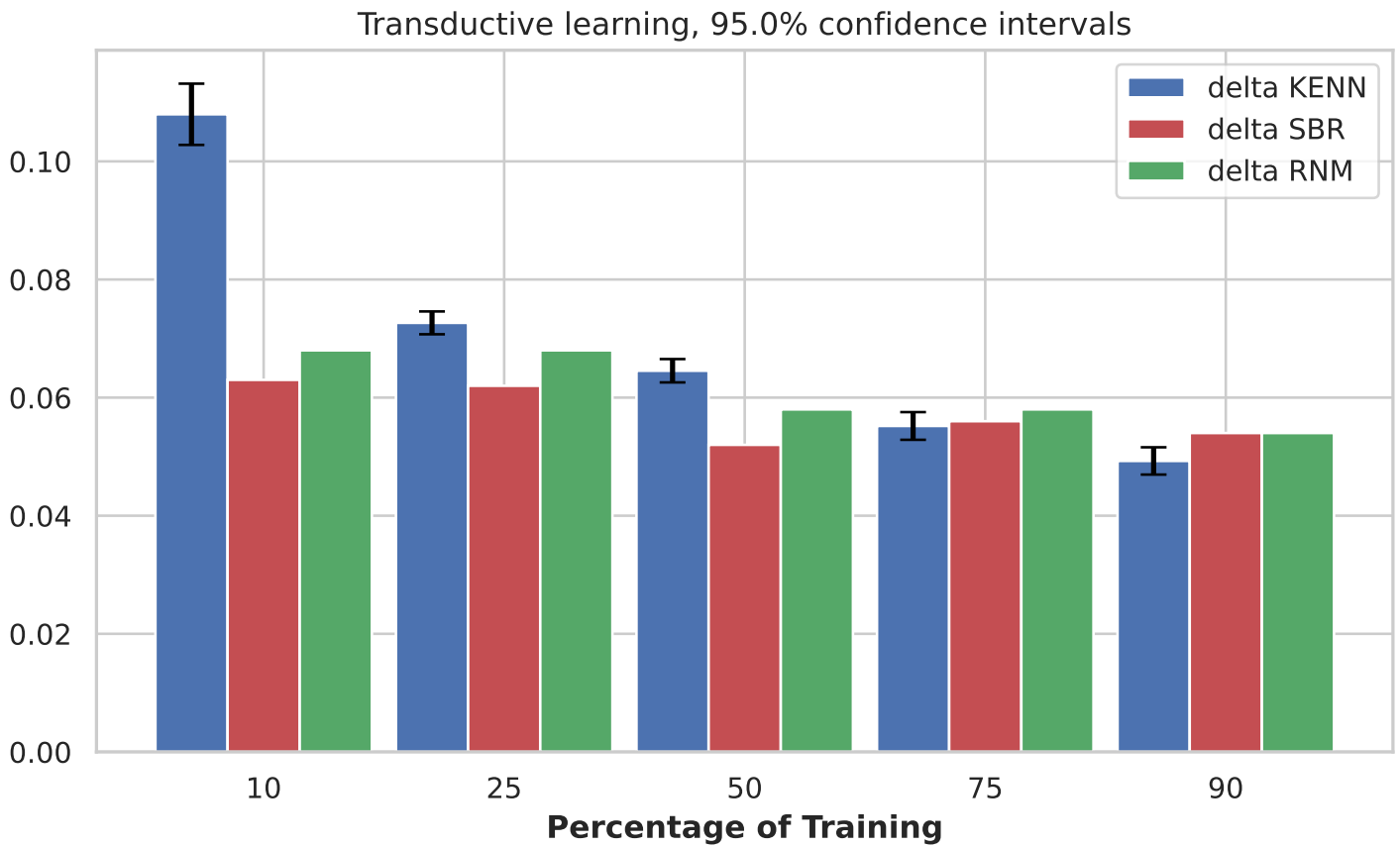


Figure: Relative improvements for the Transductive Paradigm

Explainability

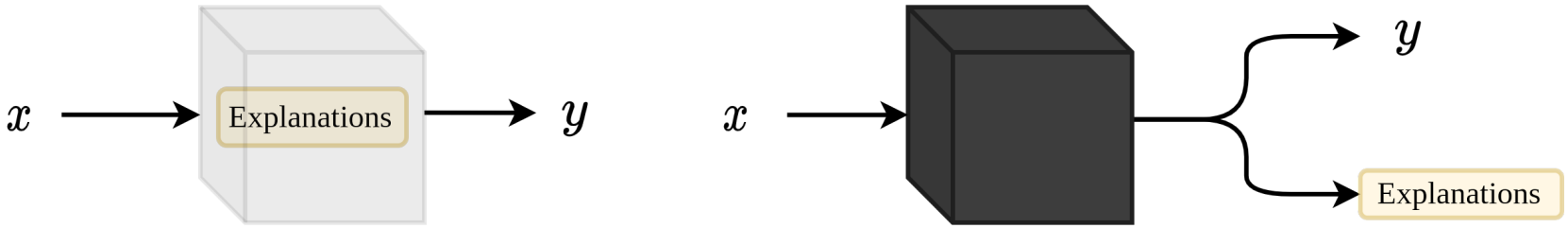
Definition (Explainability)

We define explainability in the context of supervised ML as the **generic process by which we extract any kind of explanation from a model**. This can be done by exploiting the natural properties of the model (in which case, such a model can be called explainable), or by devising techniques to extract explanations from any model.

Explainability

In XAI, two main paradigms for explainability are distinguished:

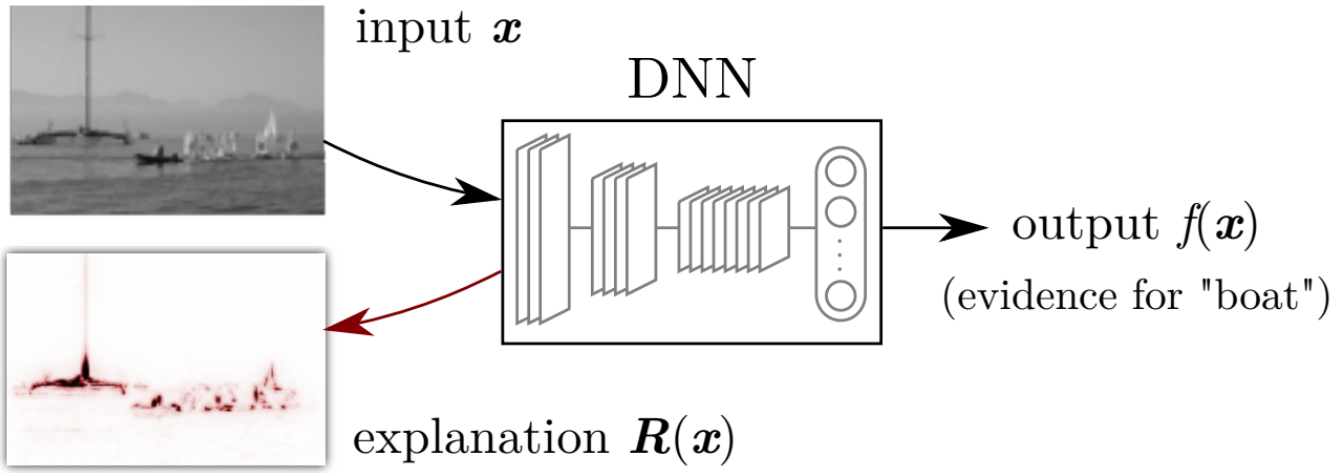
- **Transparency**
- **Post-hoc explainability**



Activation Maximization

$$x_i^* = \max_x \log p\left(\omega_i \mid x\right) - \lambda \|x\|^2$$

Local Explanations: saliency maps



$$R(x)_i = \left(\frac{\partial f}{\partial x_i}\right)^2$$

Explainability in KENN

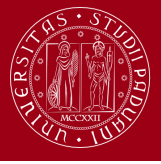
KENN can be considered a **partially transparent** model:

- A KENN layer will always be based on the prediction of a base NN, which will always be an inherently opaque model;
- On the contrary, everything happening inside the KENN layer is transparent;
- The explanations will only regard the knowledge enforcement stage.

Local explanations from a single clause

Esempio...

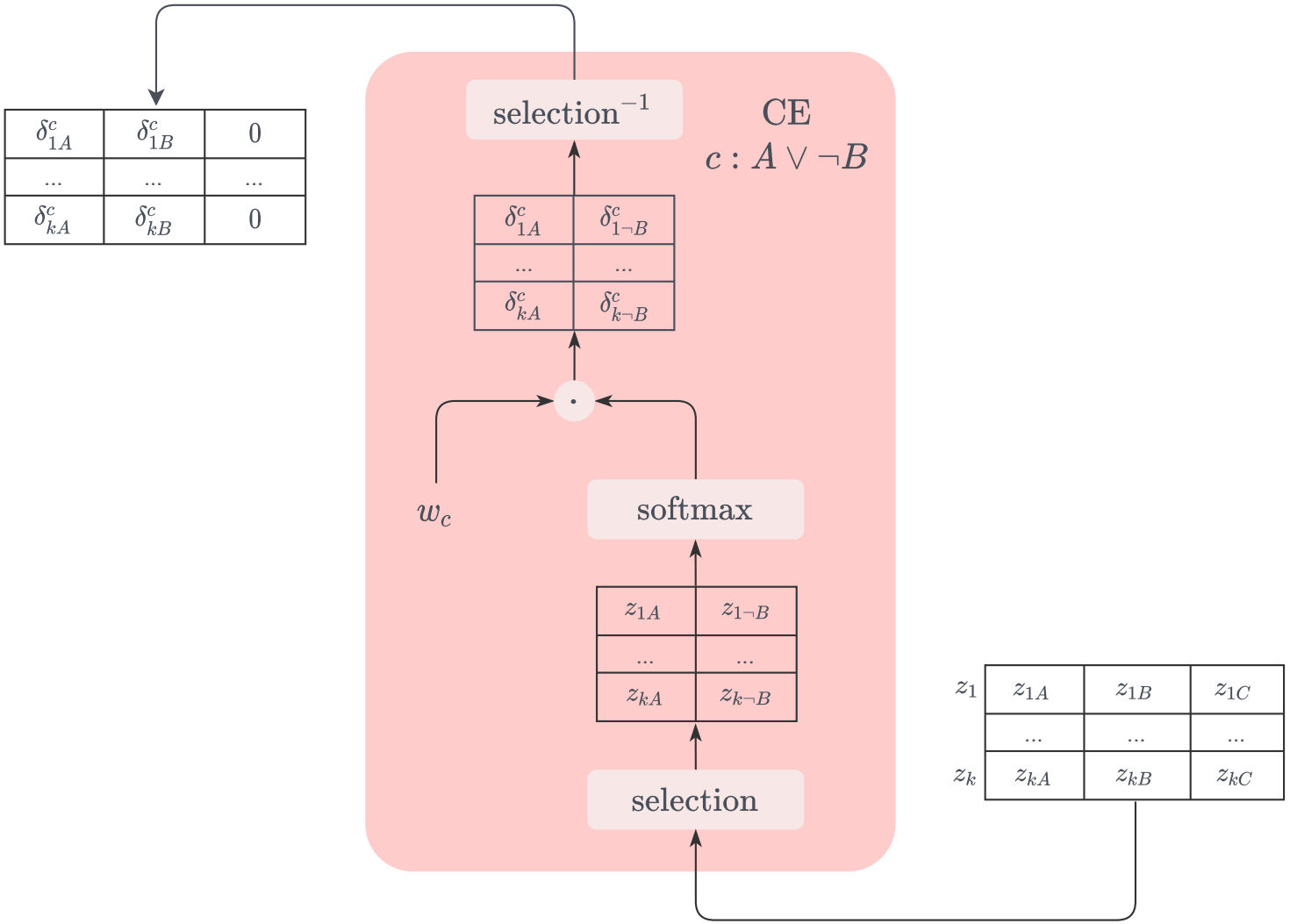
Assessing impact of more clauses



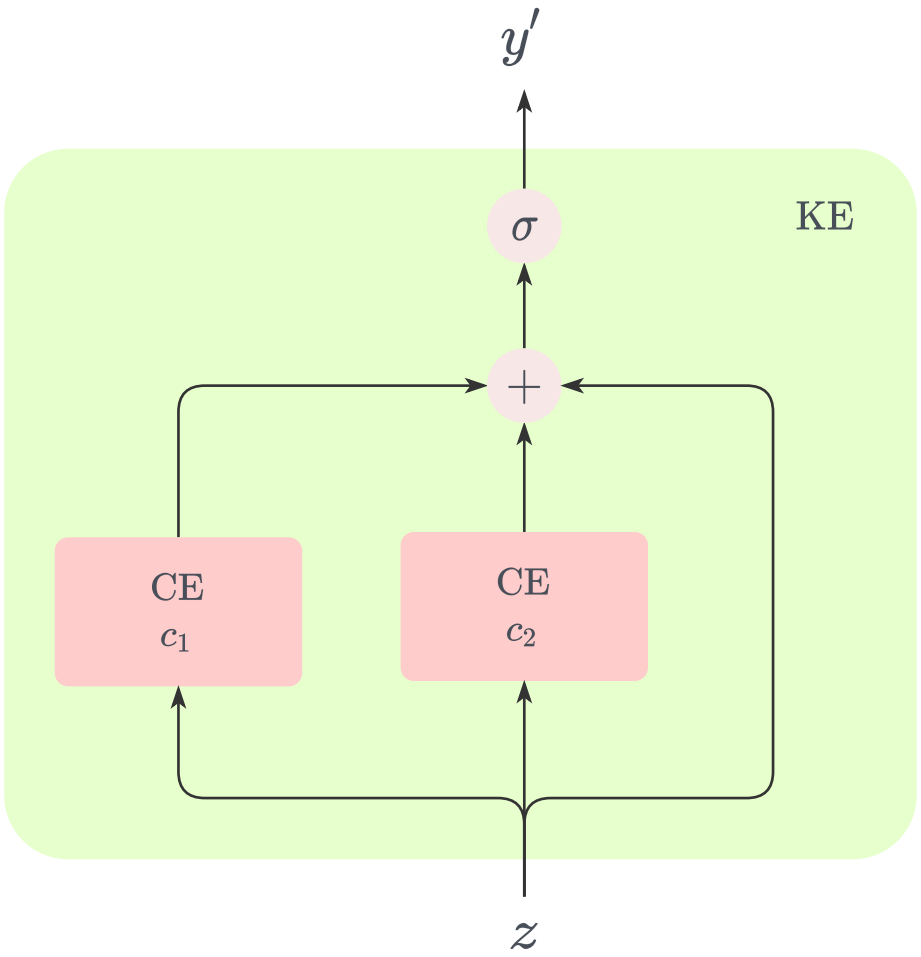
Custom score.

Thank you for your attention

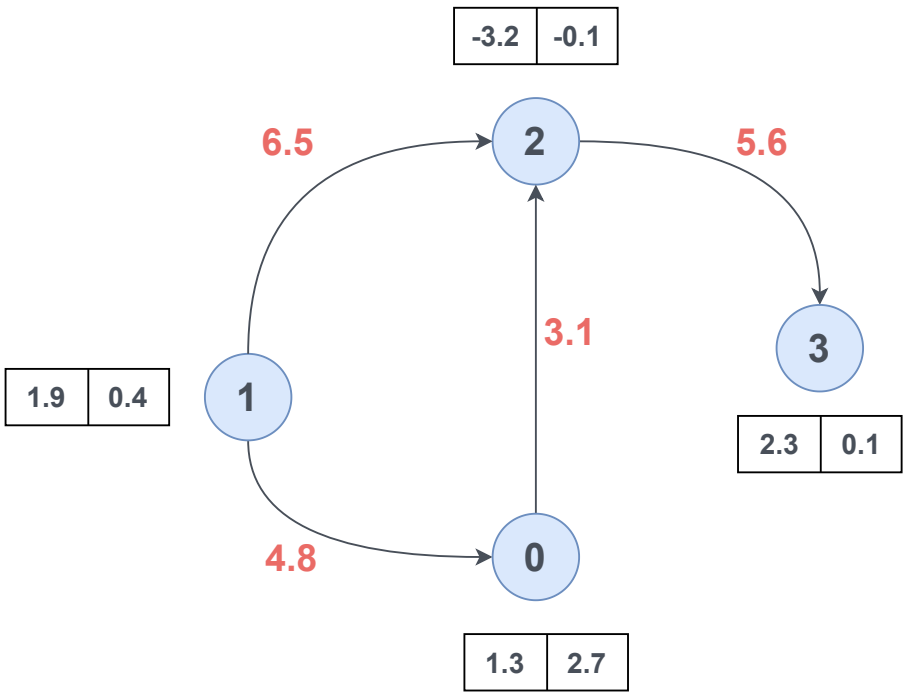
Appendix: Clause Enhancer



Appendix: Knowledge Enhancer



Appendix: KENN for relational data



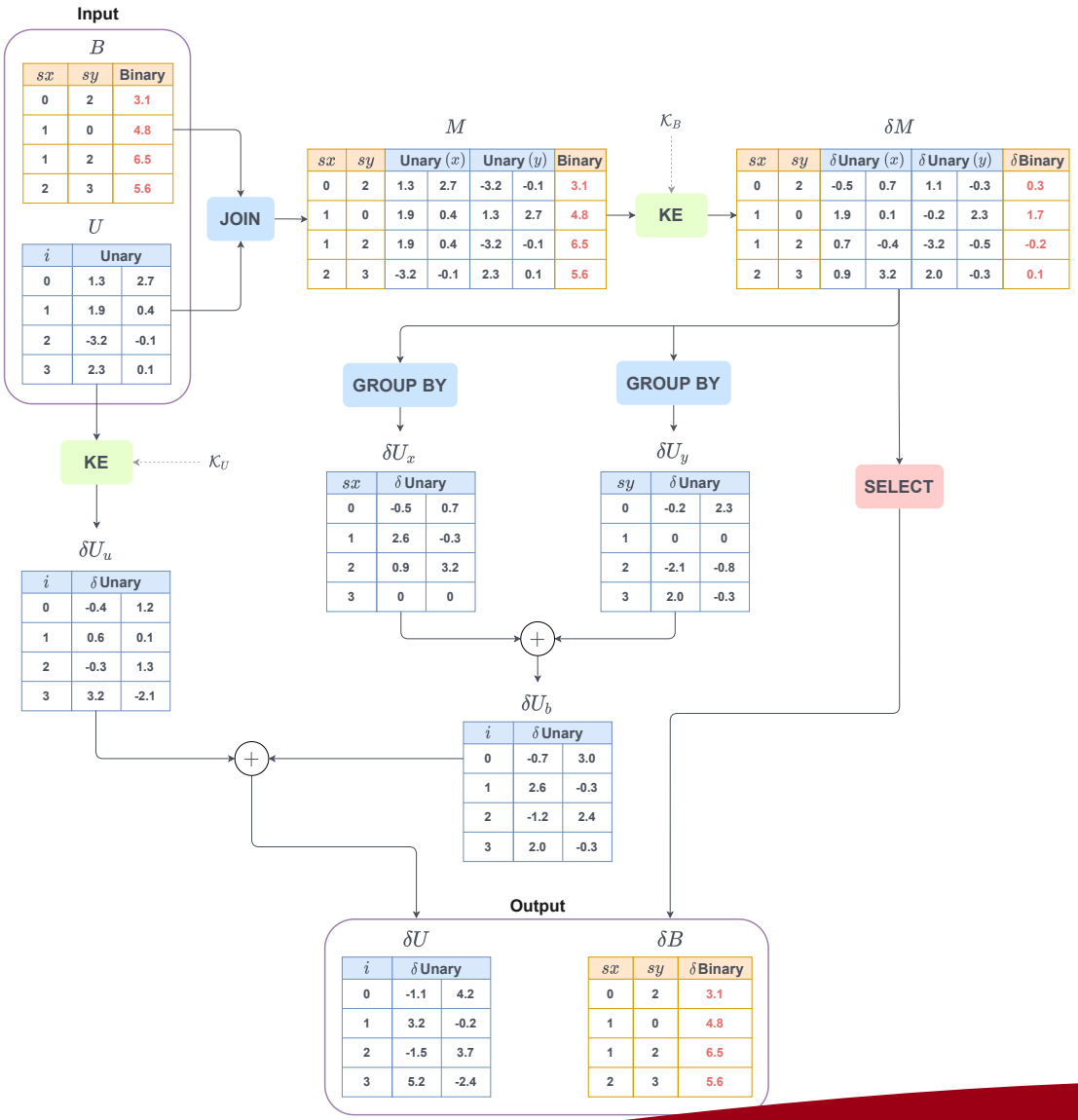
U

i	Unary	
0	1.3	2.7
1	1.9	0.4
2	-3.2	-0.1
3	2.3	0.1

B

sx	sy	Binary
0	2	3.1
1	0	4.8
1	2	6.5
2	3	5.6

Appendix: KENN for relational data



Appendix: Clause Weights learning