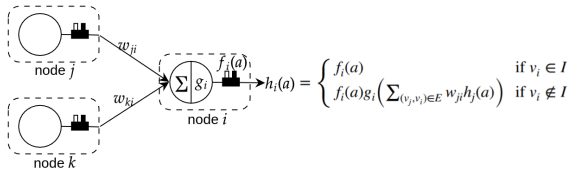
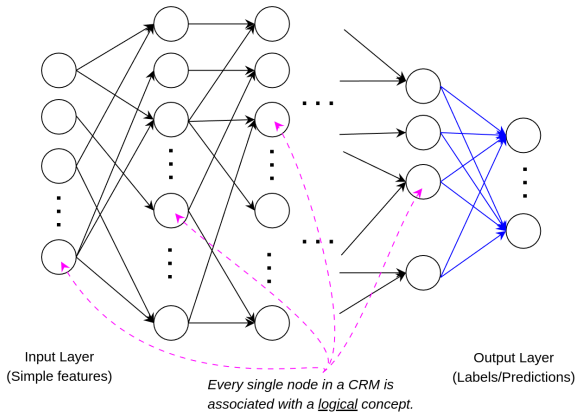


# Compositional Relational Machines

Tirtharaj Dash  
University of Cambridge, UK

Joint work with:  
Ashwin Srinivasan, A. Baskar, Devanshu Shah  
Department of CS & IS and APPCAIR  
BITS Pilani, Goa Campus, India

November 14, 2023



CRM nodes as Gated nodes

# Relational Features

A relational feature takes a clausal form:

$$C : \forall X (p(X) \leftarrow \exists \mathbf{Y} \text{ Body}(X, \mathbf{Y}))$$

or,

$$C : (p(X) \leftarrow \text{Body}(X, \mathbf{Y}))$$

Here,

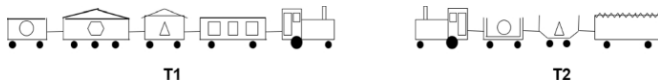
$p(X)$  : Head literal

$\text{Body}(X, \mathbf{Y})$  : Conjunction of body literals

Assumption:  $C$  is not self-recursive. We call  $C$  a “feature-clause”.

# Feature clauses

Let's look at the classic trains problem:



Some feature-clauses for trains:

$$C_1 : p(X) \leftarrow (has\_car(X, Y), short(Y))$$

$$C_2 : p(X) \leftarrow (has\_car(X, Y), short(Y), closed(Y))$$

$$C_3 : p(X) \leftarrow (has\_car(X, Y), has\_car(X, Z), short(Y), closed(Z))$$

The predicates *has\_car/2*, *short/1*, *closed/1*, etc. are defined as part of the background knowledge (*B*) about trains.

# Feature functions

A feature function is defined, for  $X = a$  as:

$$f_{C,B}(a) = \begin{cases} 1 & \text{if } B \cup (C\{X/a\}) \models p(a) \\ 0 & \text{otherwise} \end{cases}$$

Simply, for a feature-clause  $C_i$ , we refer to the corresponding feature-function as  $f_i(X)$ .

Example:

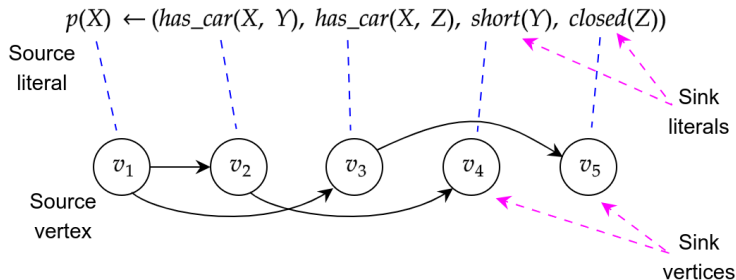


Some feature functions are:

$$f_1(t_1) = 1, f_2(t_1) = 1, f_2(t_2) = 0.$$

# Ordered Clause and Clause-dependency Graph

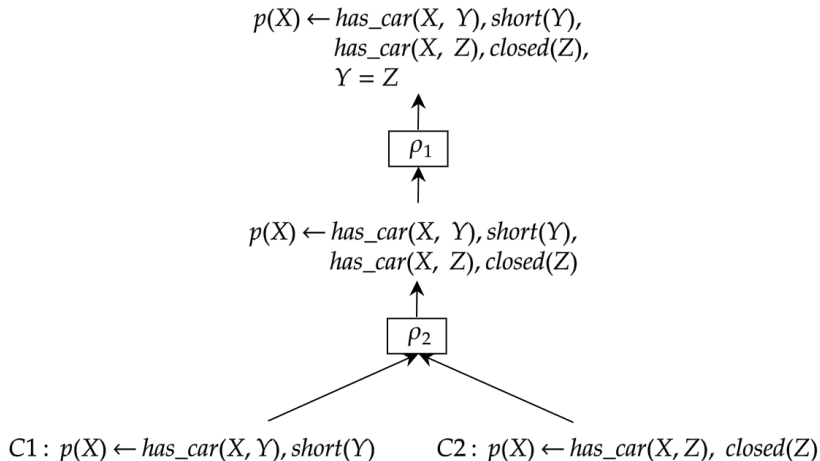
Ordered Clause: We impose an ordering of the literals in a clause. If  $C$  is a clause of the form  $\lambda_1 \leftarrow \lambda_2, \dots, \lambda_k$ , then the ordered clause is:  $\langle C \rangle = \langle \lambda_1, \lambda_2, \dots, \lambda_k \rangle$ .



Simple feature-clause: A feature-clause with a single sink literal [1].

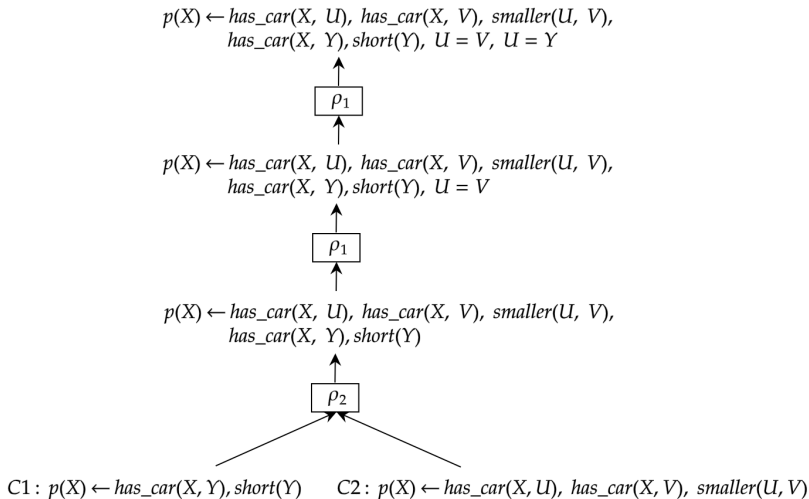
# $\rho$ -derivation of feature-clauses (Composition)

Example 1:



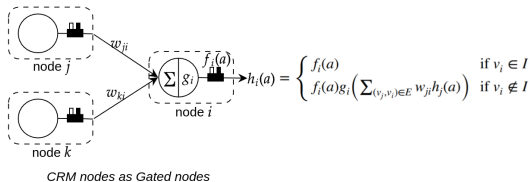
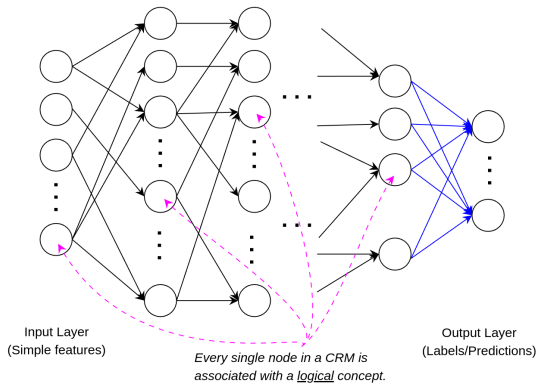
# $\rho$ -derivation of feature-clauses (Composition)

Example 2:



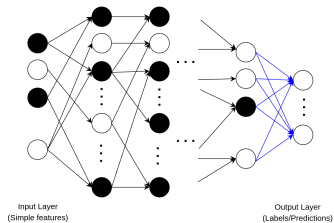
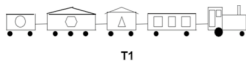


# CRM

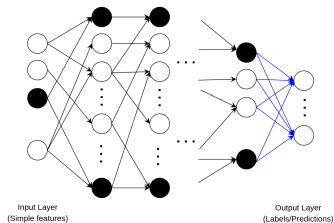
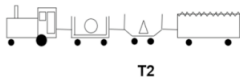


# CRM

## Relational instance 1:



## Relational instance 2:



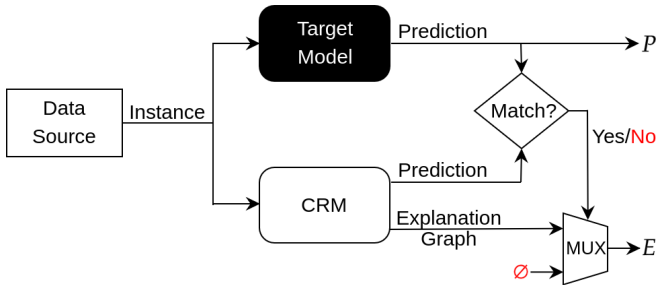
# Empirical Evaluation

## Experimental setup:

- ▶ Machine: 64GB RAM, 12 Intel Xeon CPUs
- ▶ Languages: Prolog (Aleph), PyTorch
- ▶ Feature-clauses:
  - ▶ No. of body literals: 2
  - ▶ Min. support: 10
  - ▶ Min. precision: 0.5
- ▶ Composition depth of CRMs: 3
- ▶ Activation function in CRM nodes: ReLU
- ▶ Optimiser: Adam (learning rate=0.001)
- ▶ Expl. Graph: Layer-wise Relevance Propagation (LRP)

# Empirical Evaluation

Two aspects: (a) Predictive fidelity, (b) Explanatory fidelity



Explanation: Constructed by back-tracing the top activations in each layer of the deep neural network.

# Empirical Evaluation

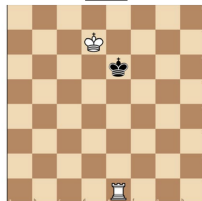
## (A) Synthetic datasets (Trains and Chess)

- ▶ Target theory (model) known

Trains



Chess

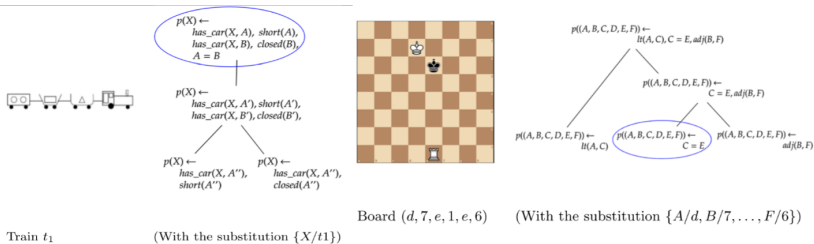


- ▶ Results

Dataset	Fidelity			
	CRM		Baseline	
	Pred.	Expl.	Pred.	Expl.
Trains	1.0	1.0	0.5	0.4
Chess	1.0	0.9	0.7	0.7

# Empirical Evaluation

- Some explanations generated by the CRM:



Target theory: Train  $X$  has a car  $Y$  and  $Y$  is short and closed.

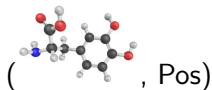
Target theory: White Rook and Black King are on the same file (column).

- CRM also generates “buggy” explanations for the chess problems: Most of these are close to or more-specific than the correct theory.

# Empirical Evaluation

(B) Real datasets (drug design: NCI GI50;  $n = 10$ )

- ▶ Target theory is NOT known. BotGNNs are used as the target models [*Dash et al., MLJ, 2022*].

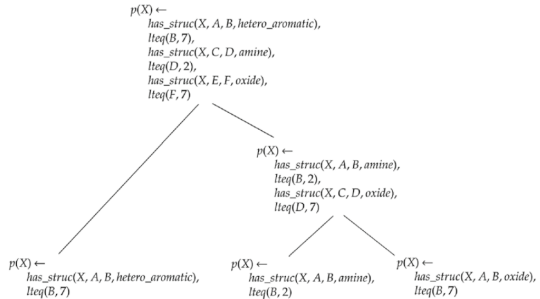
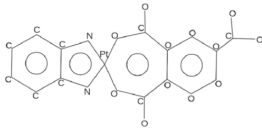


- ▶ Results: Predictive fidelity

Dataset	CRM	Baseline
786_0	0.77	0.53
A498	0.79	0.59
A549_ATCC	0.85	0.63
ACHN	0.73	0.58
BT_549	0.78	0.51
CAKI_1	0.81	0.69
CCRF_CEM	0.82	0.68
COLO_205	0.77	0.53
DLD_1	0.90	1.00
DMS_114	0.89	0.91
Avg.	0.81 (0.05)	0.66 (0.17)

# Empirical Evaluation

- An explanation generated by CRM:





# Empirical Evaluation

Some additional predictive comparisons:

Dataset	Predictive accuracy				
	CRM	GNN	DRM (500)	CILP++	Baseline
786_0	0.66 (0.01)	0.69 (0.01)	0.69 (0.01)	0.67 (0.01)	0.55 (0.01)
A498	0.67 (0.01)	0.72 (0.01)	0.70 (0.01)	0.66 (0.01)	0.52 (0.01)
A549_ATCC	0.64 (0.01)	0.67 (0.01)	0.70 (0.01)	0.60 (0.01)	0.51 (0.01)
ACHN	0.64 (0.01)	0.70 (0.01)	0.70 (0.01)	0.64 (0.01)	0.51 (0.01)
BT_549	0.66 (0.01)	0.68 (0.01)	0.70 (0.01)	0.65 (0.01)	0.53 (0.01)
CAKI_1	0.63 (0.01)	0.68 (0.01)	0.66 (0.01)	0.64 (0.01)	0.54 (0.01)
CCRF_CEM	0.65 (0.01)	0.71 (0.01)	0.71 (0.01)	0.68 (0.01)	0.63 (0.01)
COLO_205	0.60 (0.01)	0.69 (0.01)	0.67 (0.01)	0.66 (0.01)	0.56 (0.01)
DLD_1	0.69 (0.02)	0.69 (0.02)	0.70 (0.02)	0.72 (0.02)	0.69 (0.02)
DMS_114	0.68 (0.02)	0.74 (0.02)	0.75 (0.02)	0.75 (0.02)	0.76 (0.02)

GNN: Dash et al., MLJ, 2022.

DRM: Dash et al., ICANN, 2019.

CILP++: Franca et al., MLJ, 2014.

# Concluding Remarks

We make 3 contributions in this work:

- ▶ Conceptual: Provide conceptual basis of relational features, development of composition operators, and prove their completeness.
- ▶ Implementation: Use the concepts to construct a “explainable” deep neural network, called CRM in which each neuron has an associated logical concept.
- ▶ Application: Demonstrate predictive and explanatory potential of CRMs.

We are currently working on:


- ▶ CRMs as high-quality standalone predictors;
- ▶ Fast learning of large-CRMs for real-world problems.

# Thank you!

[Home](#) > [Machine Learning](#) > Article

## Composition of relational features with an application to explaining black-box predictors

Published: 26 October 2023 | (2023)



**Machine Learning**

Full Paper: [Machine Learning, 2023](#)

Code: <https://github.com/tirtharajdash/CRM>

E-mail: [td522 \[at\] cam.ac.uk](mailto:td522@cam.ac.uk)