Samsung AI

Learning Where and When to Reason in Neuro-Symbolic Inference

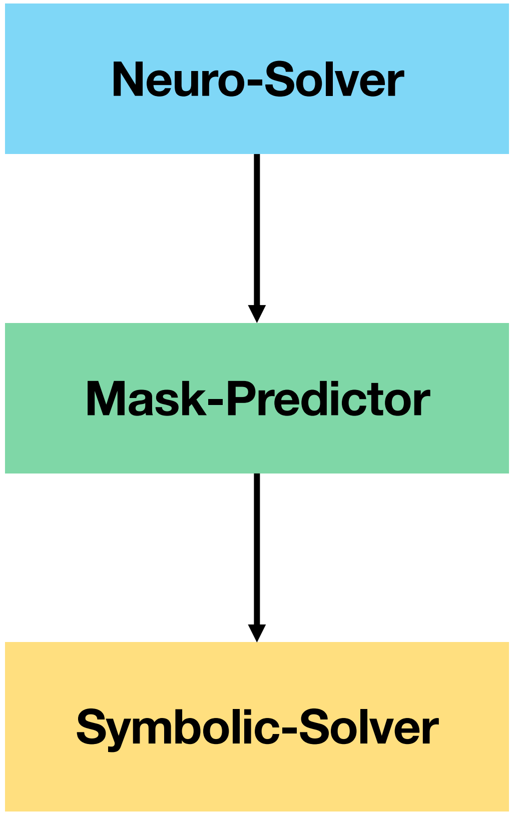
Cristina Cornelio, Jan Stühmer,
Shell Xu Hu, Timothy Hospedales

NeSy | 2023

(Presented at ICLR-23)

Intro/Motivation

- **SOTA:** “Soft”-constraints = enforced only at training time
    - (e.g., incorporation of constraints in the loss)
- **Goal:** Imposition of hard constrains at testing to ensure that the domain-specific knowledge is respected by the predictions
- **Idea:** Neuro-Symbolic integration method



- **3 components:** Neuro-Solver, Mask-Predictor and Symbolic-Solver
- Symbolic reasoning is not feasible in many scenarios
- Mask predictor: makes the reasoning more efficient, directing the reasoning focus

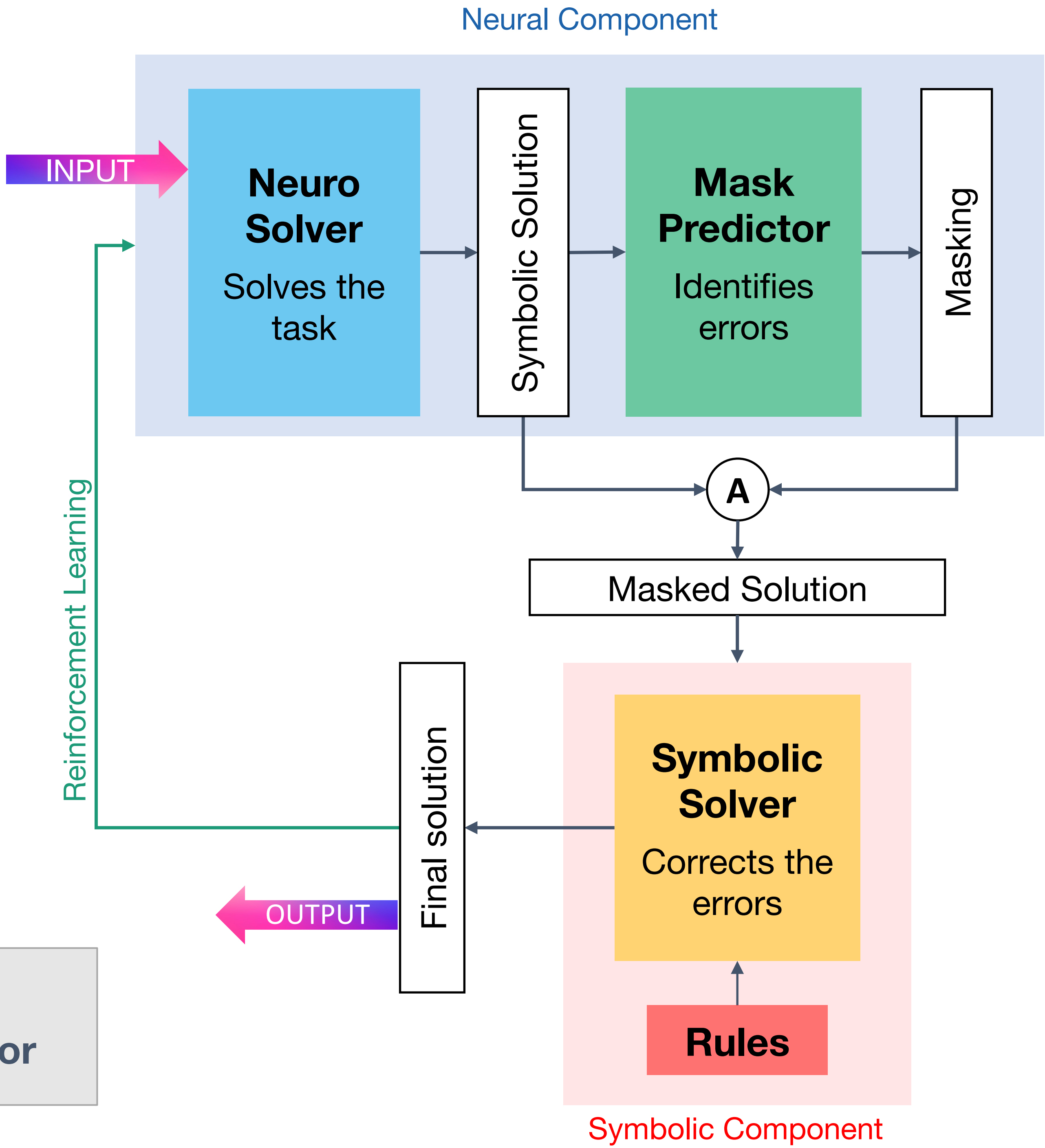
Architecture - NASR

(Neural Attention for Symbolic Reasoning)

Given: a task to solve and a set of rules $\mathcal{R}$.

1. **Neuro-Solver**: outputs an approximate solution
2. **Mask-Predictor:** identifies the components of the symbolic-solution that do not satisfy the rules R
3. **Adapter function:** Combines the symbolic-solution and the masking to form the masked solution (matching the Symbolic-Solver format)
4. **Symbolic-Solver:** uses the rules R to correct the masked components of the symbolic solution (any type of rules/constraint can be used)

Symbolic-Solver corrects the Neuro-Solver prediction errors identified by the Mask-Predictor



Learning paradigm

- $\mathcal{X}$ is the set of all possible inputs for the task under consideration
- $\mathcal{Y}$ is the set of all the possible complete solutions
- $\mathcal{Z} = [0,1]^k$, where k is the dimension of $y \in \mathcal{Y}$, and $0/1$ indicates a masked/non-masked element
- $\mathcal{Y}'$ is equal to $\mathcal{Y}$ with an additional token class 0, corresponding to a masked solution element

Final Hypothesis function
maps $\mathcal{X}$ to a probability distribution over $\mathcal{Y}$

$$\boxed{f_\theta(x)} = \boxed{sb}\left(\boxed{adapt}\left(\boxed{ns(x)},\ argmax\left(\boxed{mp}\left(\boxed{ns(x)}\right)\right)\right), \boxed{\mathcal{R}}\right)$$

Symbolic Solver
maps $\mathcal{Y}'$ to a probability distribution over $\mathcal{Y}$.

Rules $\mathcal{R}$: rules that we want to impose

Neuro-Solver
maps $\mathcal{X}$ to probability distribution over $\mathcal{Y}$

Adapter function combines a probability distribution over $\mathcal{Y}$ with an element in $\mathcal{Z}$
(e.g. element wise product)

Mask-Predictor
that takes in input a probability distribution over $\mathcal{Y}$ and produce as output a probability distribution over $\mathcal{Z}$

Learning paradigm

$$f_\theta(x) = sb\ \left(adapt\left(ns(x),\ \ argmax\left(mp\left(ns(x)\right)\right)\right), \mathcal{R} \right)$$

1- Supervised learning

- **Neuro-Solver** and the **Mask-Predictor** are first pre-trained individually (with supervision)
- They are then integrated together in the pipeline

→ **NASR** without RL

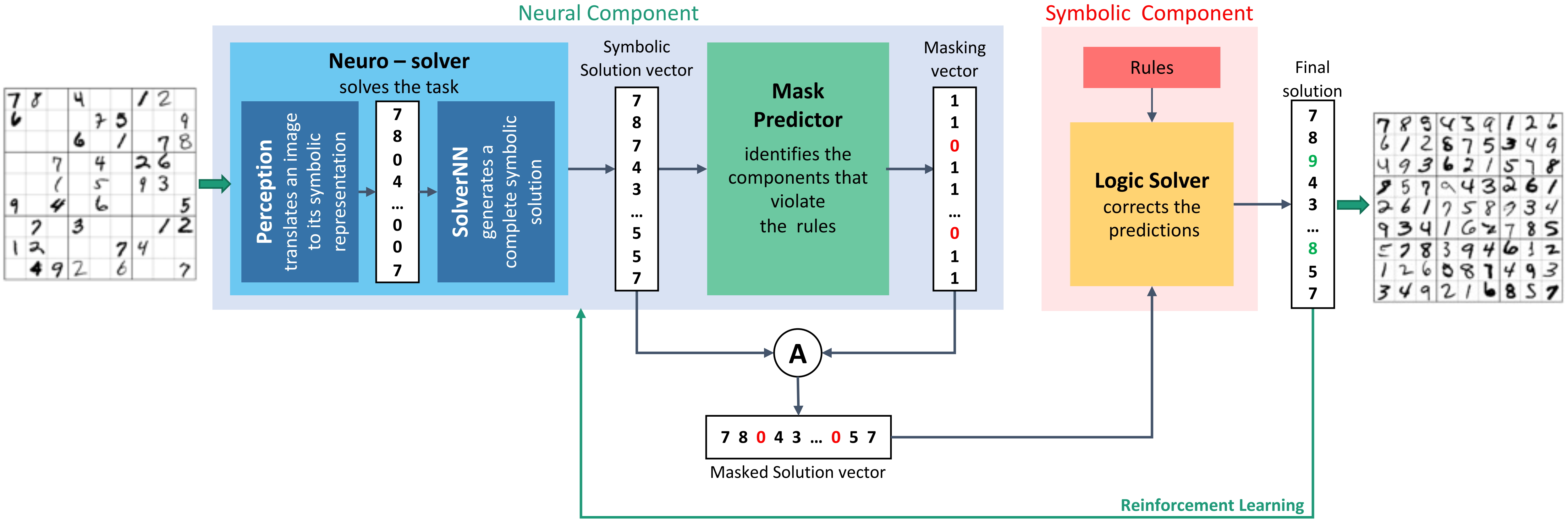
2 - Reinforcement learning

- **NASR** is then refined using **reinforcement learning**

$$\mathcal{L}(x;\theta) = -\ \ r\ /\ \log P_\theta(m|ns(x))$$

NASR with RL ←

Experiments – Visual Sudoku



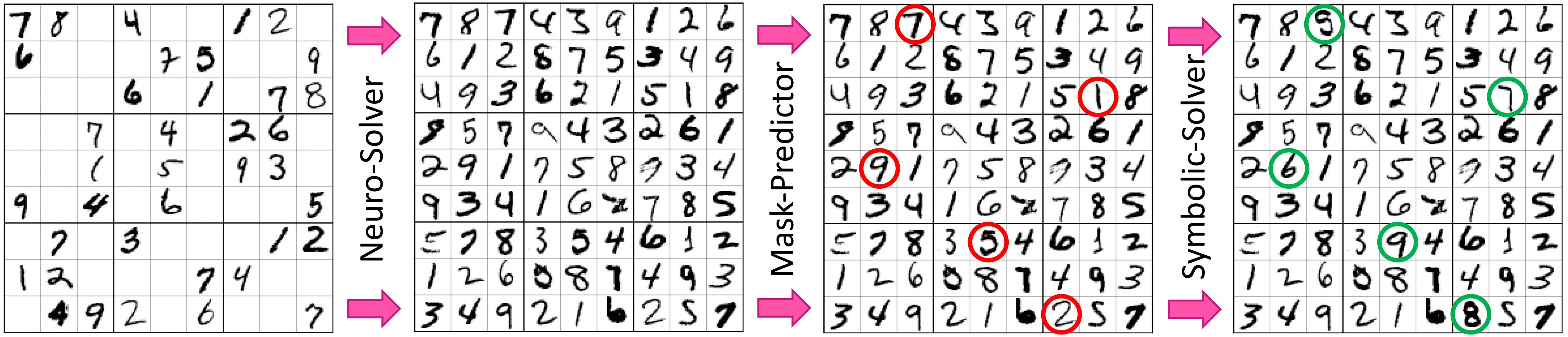
- **Perception:** simple (CNN) for single MNIST digit classification
- **SolverNN** & **Mask-Predictor:** Transformer (4 sequential self-attention blocks)
- **Adapter function:** Pointwise product
- **Symbolic-Solver:** PySwip (Python interface to SWI-Prolog) & a brute force backtrack-based algorithm

Experiments – Visual Sudoku

Dataset → challenge

- *big_kaggle* → scaling
- *minimal* → minimal number of hints
- *multiple_sol* → multiple solutions
- *satnet_data*

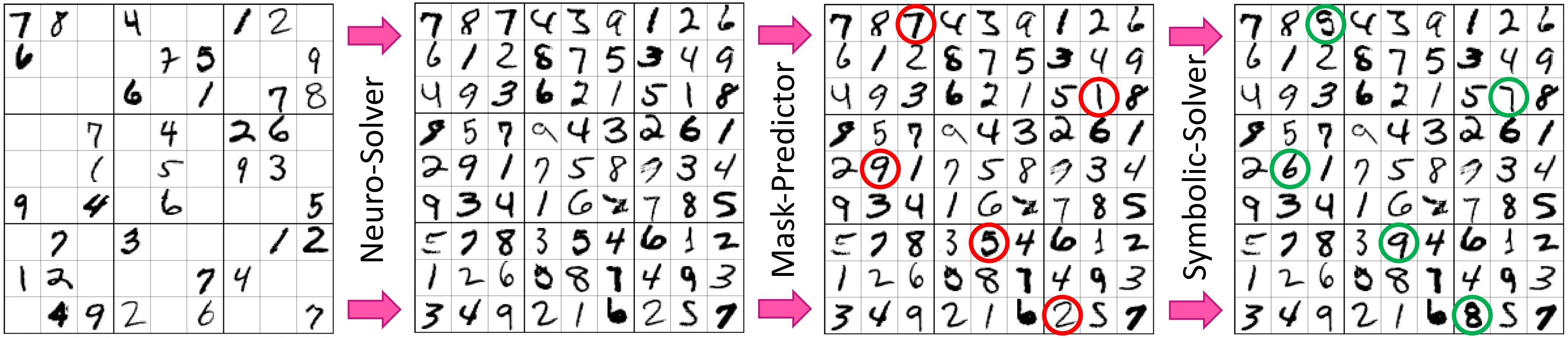
dataset	# hints avg.	# hints [min , max]	size
big_kaggle	33.82	[29, 37]	100’000
minimal	17	[17, 17]	50’000
multiple_sol	34.75	[34, 35]	10’000
satnet_data	36.22	[31, 42]	10’000



Experiments – Visual Sudoku

Comparison between:

- **NASR** (our)
  - with RL or without RL (only pretrained)
- **Symbolic baseline**
  - images → symbolic vector → symbolic solver
- **NeurASP** (Yang et al. IJCAI 2020)
- **SatNet** (Wang et al. ICML 2019)
- **SatNet + NASR**
  - SatNet as Neuro-Solver in NASR



SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver, P.W. Wang, P.L. Donti, B. Wilder, Z. Kolter, ICML 2019
NeurASP: Embracing Neural Networks into Answer Set Programming, Z. Yang , A. Ishay, J. Lee, IJCAI 2020

Results – Visual Sudoku

Results summary:

- We significantly **outperform the baseline** in most of the cases (and never perform worst)
  - We are more **robust to noise** compared to the symbolic baseline.
- We **improve** the performance of an **existing method, by integrating it** in our pipeline;

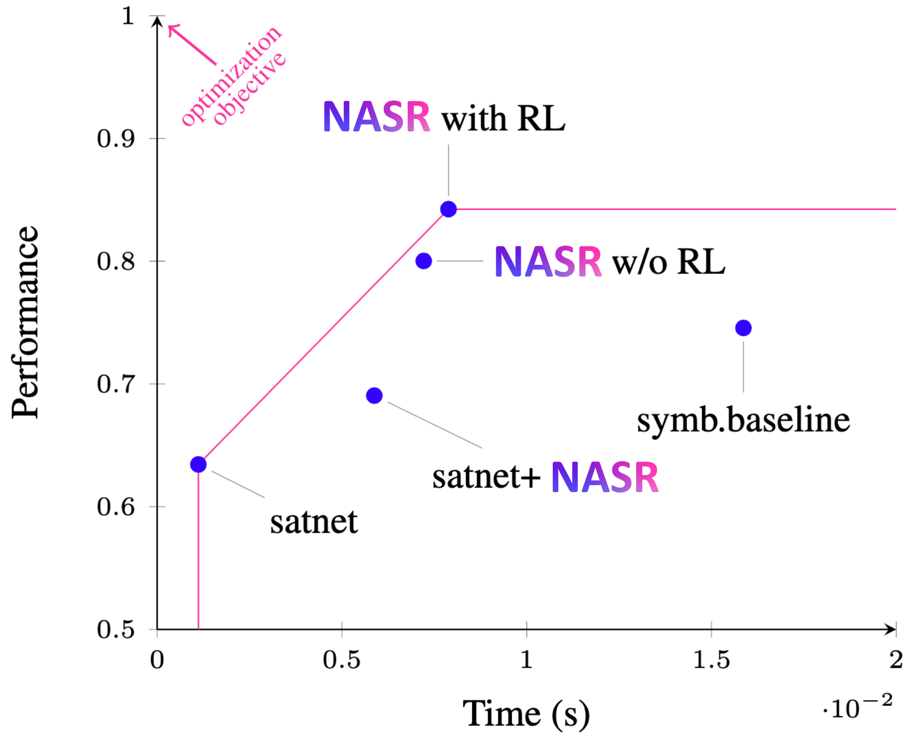
	big kaggle	minimal 17	multiple sol	satnet data
Symbolic baseline	74.56	**87.70**	63.50	63.20
NeurASP	timeout	**89.00***	timeout	timeout
SatNet	63.44	0.00	0.00	60.10
SatNet + NASR (our)	69.05	0.02	24.20	**81.40**
NASR (our)	**84.24**	**87.00**	**73.00**	**82.20**

% of completely correct sudoku boards

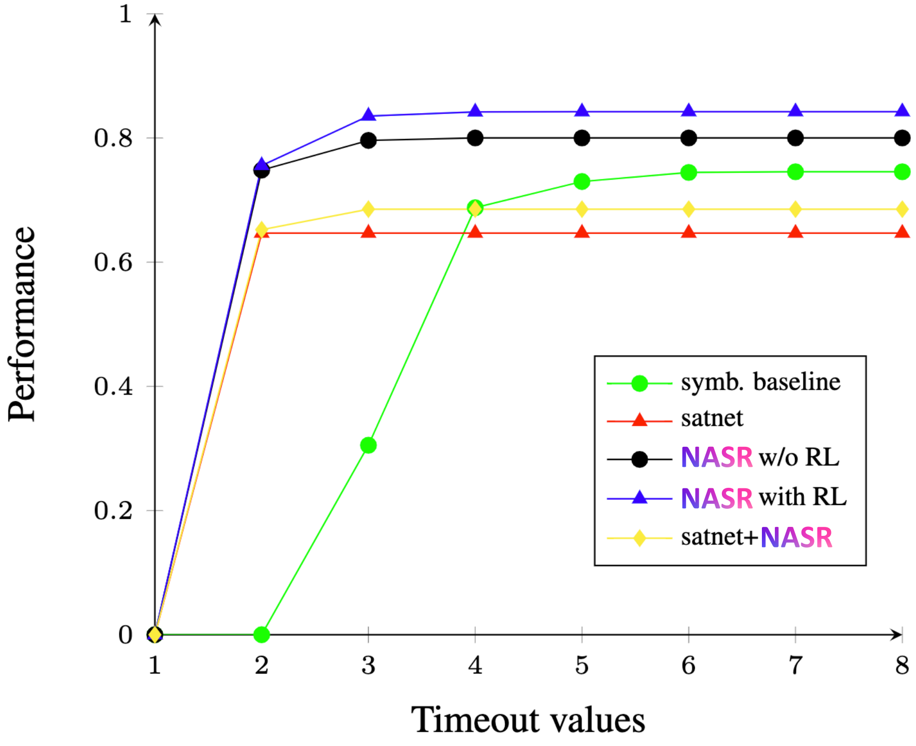
Results – Visual Sudoku: Efficiency

We are **more efficient** in terms of trade-off between:

- performance (percentage of completely correct boards)
- computational time



Pareto front performance vs. computational time



Performance limiting the computational time

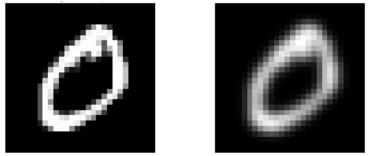
Efficiency results for *big_kaggle* dataset

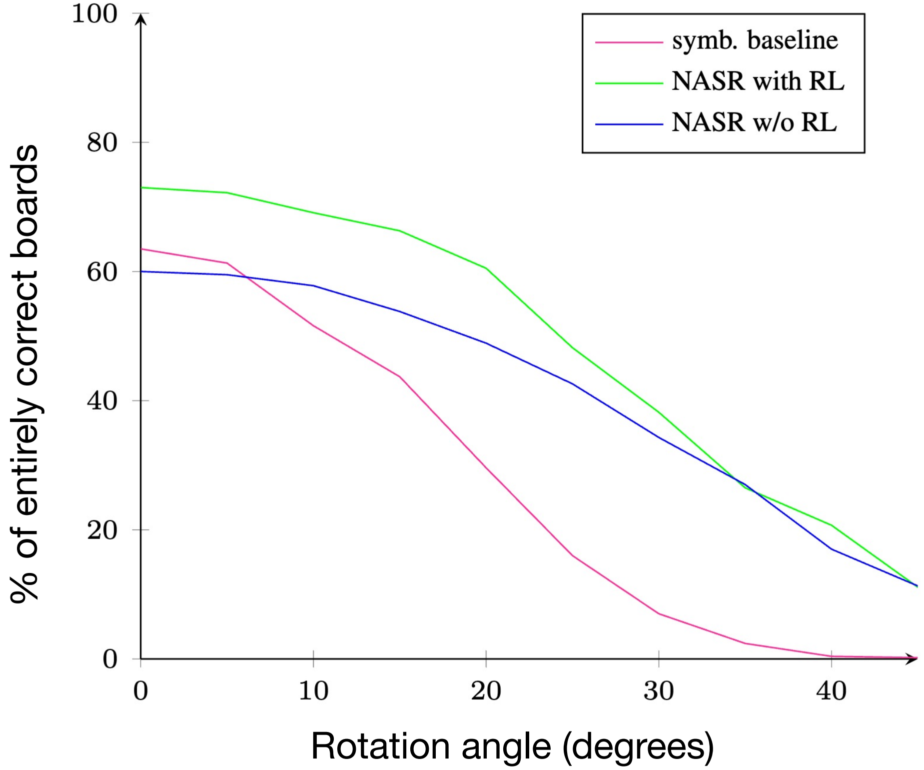
Results – Visual Sudoku: Robustness to noise

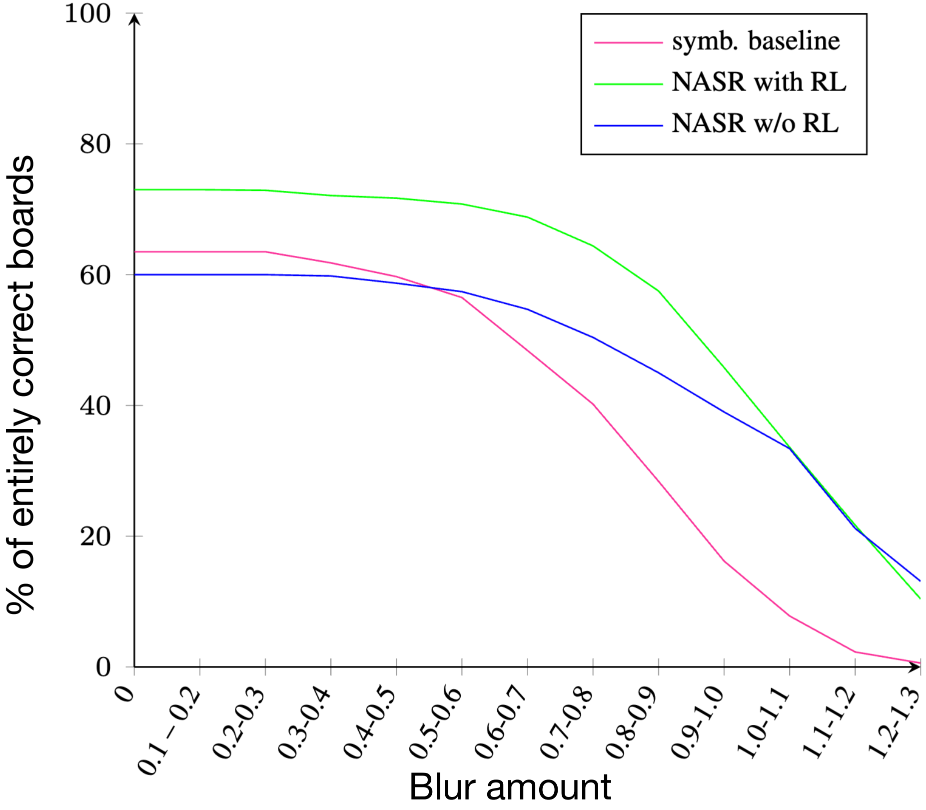
- Results shows that we are **always better (or equal) to the baseline.**
- However, our method is much more **robust to noise** compared to the baseline:

Rotation of digits:

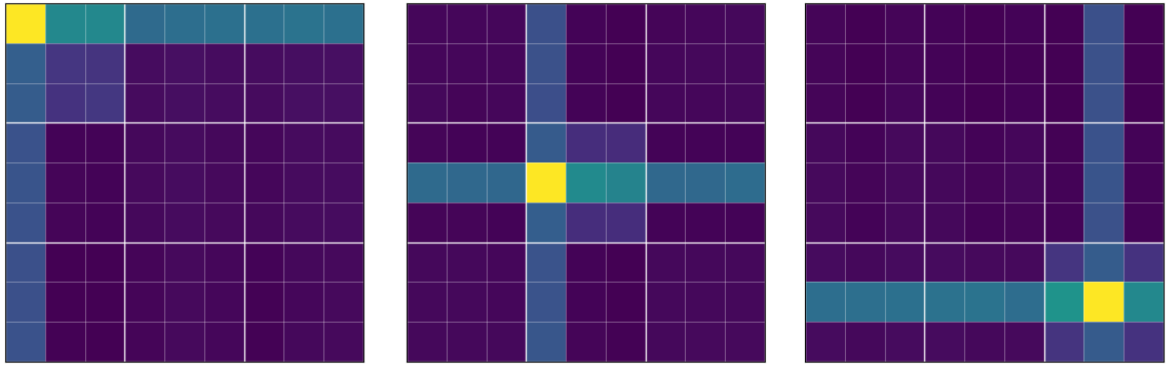
Adding gaussian blur:





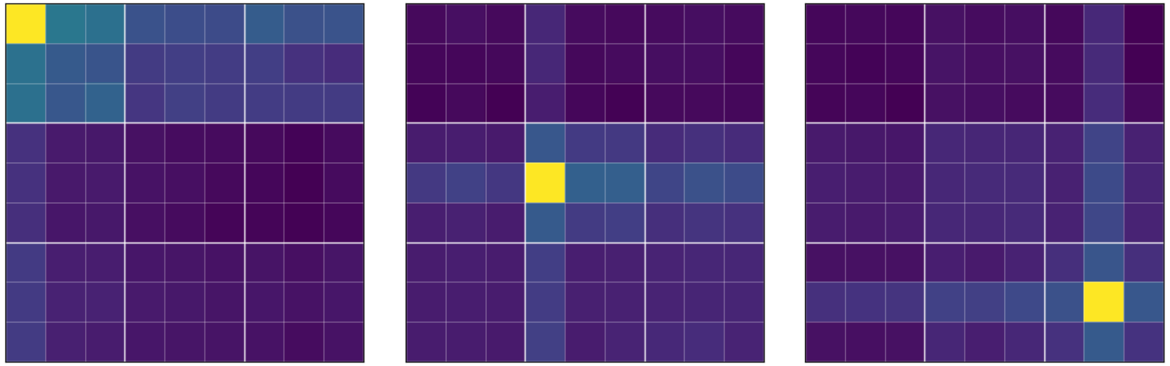


Results – Visual Sudoku: Attention maps



Cell(1,1) *Cell(5,4)* *Cell(8,8)*

Neuro-Solver



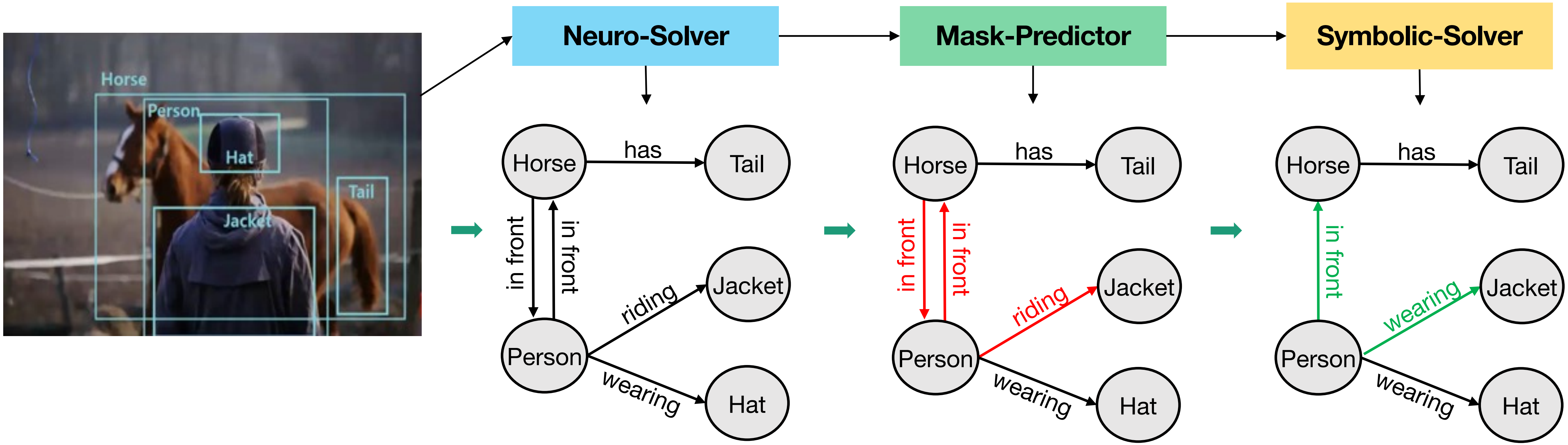
Cell(1,1) *Cell(5,4)* *Cell(8,8)*

Mask-Predictor

Looking at the attention in the Transformer:

- **When considering a cell:**
  - Average of all the attention layers for the Neuro-Solver and for the Mask-Predictor
  - Noticeable focus on the **row**, the **column** and the **3×3 block** (corresponding to the 3 Sudoku rules)
- **It is learning the correct sudoku rules**

Results – Scene Graph Prediction



Dataset:

- **GQA dataset**
  - Balanced version of Visual Genome

Constraints/Rules:

- Domain/Range of relations
  (e.g., *domain(wear)=person*)

Tasks: Predicate classification

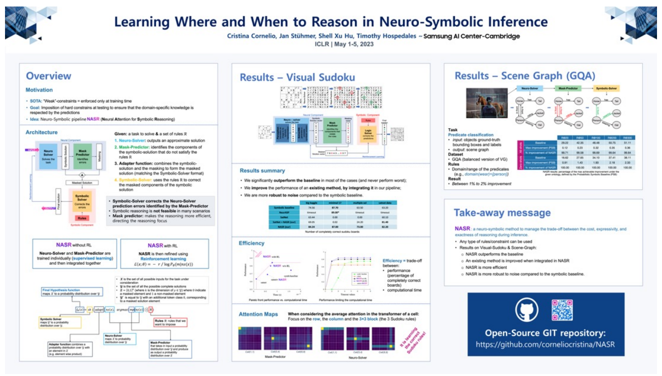
- Input**:** ground truth bounding boxes for the objects and objects labels
- Output: Scene Graph

Results – Scene Graph Prediction

		R@20	R@50	R@100	R@200	R@300
All-shots	Baseline	29.22	42.35	48.48	50.75	51.11
	Max-improvement (PSB)	0.12	0.23	0.32	0.35	0.36
	% improvement of NASR	99.71	99.58	99.69	99.64	99.64
Zero-shots	Baseline	16.62	27.65	34.10	37.41	38.11
	Max-improvement (PSB)	0.91	1.43	1.93	2.18	2.33
	% improvement of NASR	100.00	100.00	100.00	100.00	100.00

NASR results: percentage of the max achievable improvement under the given ontology, defined by the Probabilistic Symbolic Baseline (PSB)

Learning Where and When to Reason in Neuro-Symbolic Inference



Check out our poster after the coffee break!
Looking for collaborations ☺!

NASR

https://github.com/corneliocristina/NASR

Questions?

Experiments – Visual Sudoku

RL scenario:

- **Input state =** solution board provided by the Neuro-Solver (Perception+SolverNN).
- **Actions space =** set of all possible complete masking boards configurations
- **Action *m* =** simultaneous execution of 81 independent sub-actions board + deterministic application of the Symbolic-Solver to the masked solution board
  - **Sub-action m_i =** decision of masking or not a single cell *i* in the solution board
- **Final state =** solution board provided as output by the Symbolic-Solver
- **Rewards** (positive & normalized) **=** sum of two types of rewards with different order of magnitude
  - the main reward, $r_e \in \{0, 10\}$, when the entire board is correct and a
  - marginal reward $r_c \in [0, 1]$ for each correct cell *i*

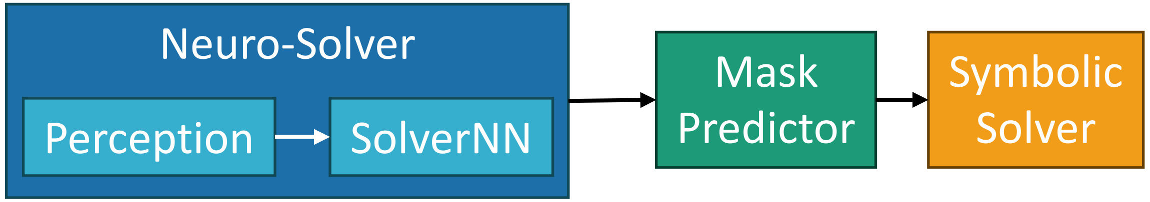
$$r = r_e + r_c = 10 \cdot \delta_{b',b} + \frac{1}{81} \sum_{i=0}^{81} \delta_{b'_i, b_i}$$

b' output board
b ground truth board

- **Loss function**, for each batch B**:**

$$\mathcal{L}(B;\theta) = -\sum_{x \in B} r \log P_\theta(\tilde{m}|ns(x)) = -\sum_{x \in B} \left(r \sum_{i=0}^{81} \log P_\theta(\tilde{m}_i|ns(x))\right)$$

Results – Visual sudoku:
Analysis of the Components Roles



	% Perception errors corrected by NASR		% SolverNN errors identified by Mask-Predictor	
	by SolverNN	by Mask-Predictor	hint cells	solution cells
big kaggle	85.90	14.10	53.17	77.42
minimal 17	95.90	4.10	8.21	32.24
multiple sol	87.21	12.79	42.09	24.73
satnet data	85.43	14.57	44.76	71.16

Experiment 1:
- hints cells that have been incorrectly predicted by the Perception but have been corrected by NASR, % that is corrected directly by the SolverNN or the Mask-Predictor
- **Result:** Perception errors are usually corrected by the SolverNN

Experiment 2:
- Mask-Predictor distinguishes between the errors in the hint cells and the errors in the solutions cells.
- **Result:** the Mask-Predictor does not systematically prefer either of the two

Architecture – Scene graph