Embedding Constraint Reasoning in Machine Learning to Build Generalist Systems

Yexiang Xue

Department of Computer Science
Purdue University

Intelligent Systems Integrate Learning and Reasoning

Perception

Learning

Knowledge

Reaction

Reasoning

Machine learning:

- **Bottom-up:** Learn predictive models from data
- **Challenging** in providing formal guarantees
- May **violate constraints** in rare and unseen situations

Automated reasoning:

- **Top-down:** Build models from problem description
- **Rigid models**: problem formulation must be agreed a-priori
- **Difficult to adapt** to data distributions

Data

Action

Intelligent Systems Integrate Learning and Reasoning

Perception

Learning

Knowledge

Reaction

Reasoning

Machine learning:

- guarantees
- May **violate constraints** in rare and unseen situations

Automated reasoning:

- odels iption lem
- **Difficult to adapt** to data distributions

Big challenge:
How to **integrate** learning with reasoning

Data

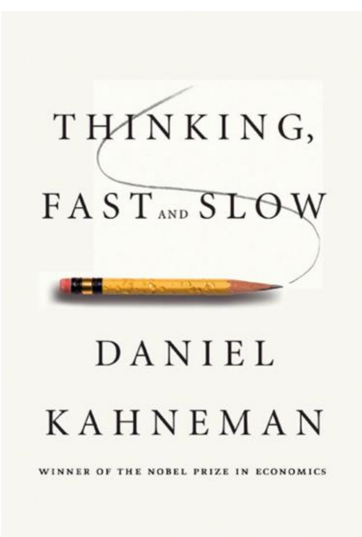
Action

Generalist Systems, Think Fast and Slow

Input Specifications:
- Add a blue microwave right of the oven
- Add a green toaster left of the oven and below the sink

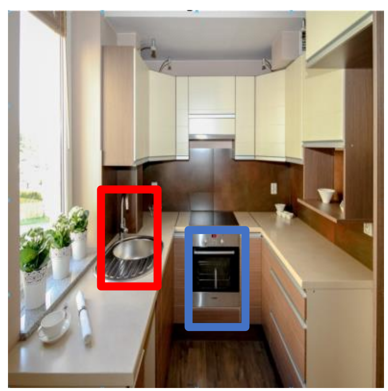
Reasoning *&* ***learning*** *are in charge of different cognitive systems.*

Need both for building a generalist AI.



Learning

System 1
perception
(fast thinking)



Reasoning
+ Learning

System 2
planning &
generation
(slow thinking)

Data-driven Operational Research

- Service Vehicle Dispatching
  - Recommend dispatching routes which (i) satisfy daily delivery requests; (ii) meet implicit preferences
  - Constraint reasoning along (e.g., solving TSP) cannot learn implicit preferences from data
  - Machine learning-based recommendations cannot meet daily requests

- <u>CO</u>nstraint <u>R</u>easoning <u>E</u>mbedded Learning (CORE) as a **combined ML + Reasoning** approach

- Valid routes: **1% (pure ML) → 100%**
- Satisfy drivers’ utilities well

Program Synthesis from Natural Language

If-then program synthesis

Input: SMS me to park in the garage if it snows tomorrow.

Output:

Trigger (If)	Action (then)..
Channel: Weather	Channel: SMS
Function: Snow tomorrow	Function: send message "park in the garage"

Text2SQL query synthesis

Input Table:

	Player	No.	Position	School
0	Antonio	21	Guard-Forward	Duke
1	Voshon	2	Guard	Minnesota
2	Marin	3	Guard-Forward	Butler CC

Input Query:

How many schools did player number 3 play at?

Output SQL Query:

SELECT COUNT "School" WHERE "No." = "3"

agg-op sel-col cond-col cond-op cond-val

- **Challenging machine learning** problem: learn a program from natural language; pure ML cannot satisfy syntactic or semantic rules.

- **Reasoning** can produce valid programs, but cannot understand natural language.

- **CORE machine learning + reasoning**

  if-then programs generation: validity 88% → **100%,** 2% ↑ in accuracy.
  Text2SQL: validity 83.7% → **100%,** 4.2% ↑ execution accuracy, 1.9% ↑ logic accuracy

Design Generation

Existing Kitchen Env:

Input Specifications:

- Add a blue microwave right of the oven
- Add a green toaster left of the oven and below the sink

(stated in propositional logic)

- Good designs need to meet industry standards and user needs, while capturing subtle aspects such as aesthetics and convenience.
- **Complete constraint reasoning approach**: satisfy design specifications, but cannot capture subtle aspects. In fact, cannot be encoded in objective functions.
- **Complete ML approach**: generate beautiful designs, but cannot meet specifications.

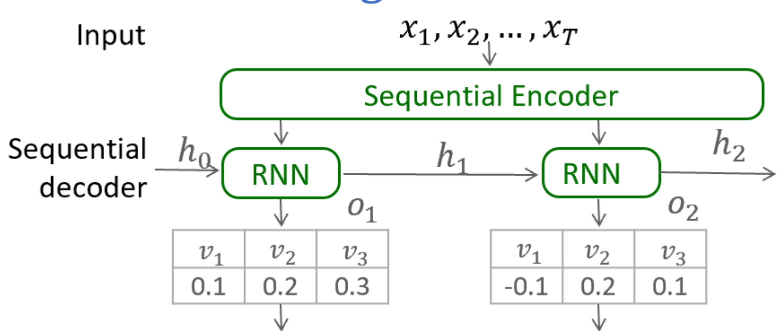
Baseline (Stable Diffusion)

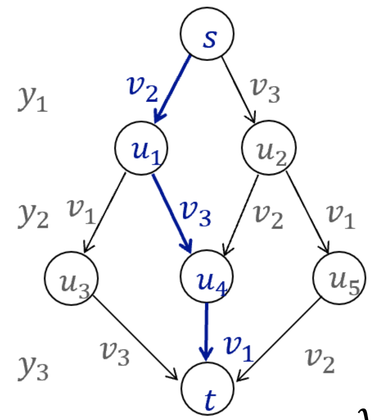
Ours (CORE)

Technical Approach

CORE framework

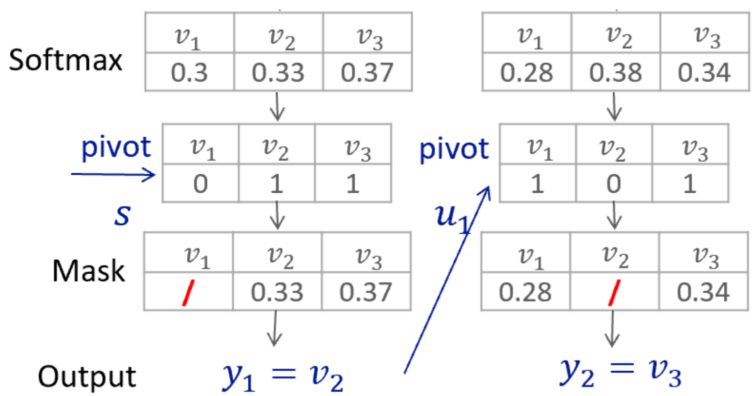
Machine Learning





Decision Diagram

y_1, y_2, y_3 all different; $y_1 \neq v_1$



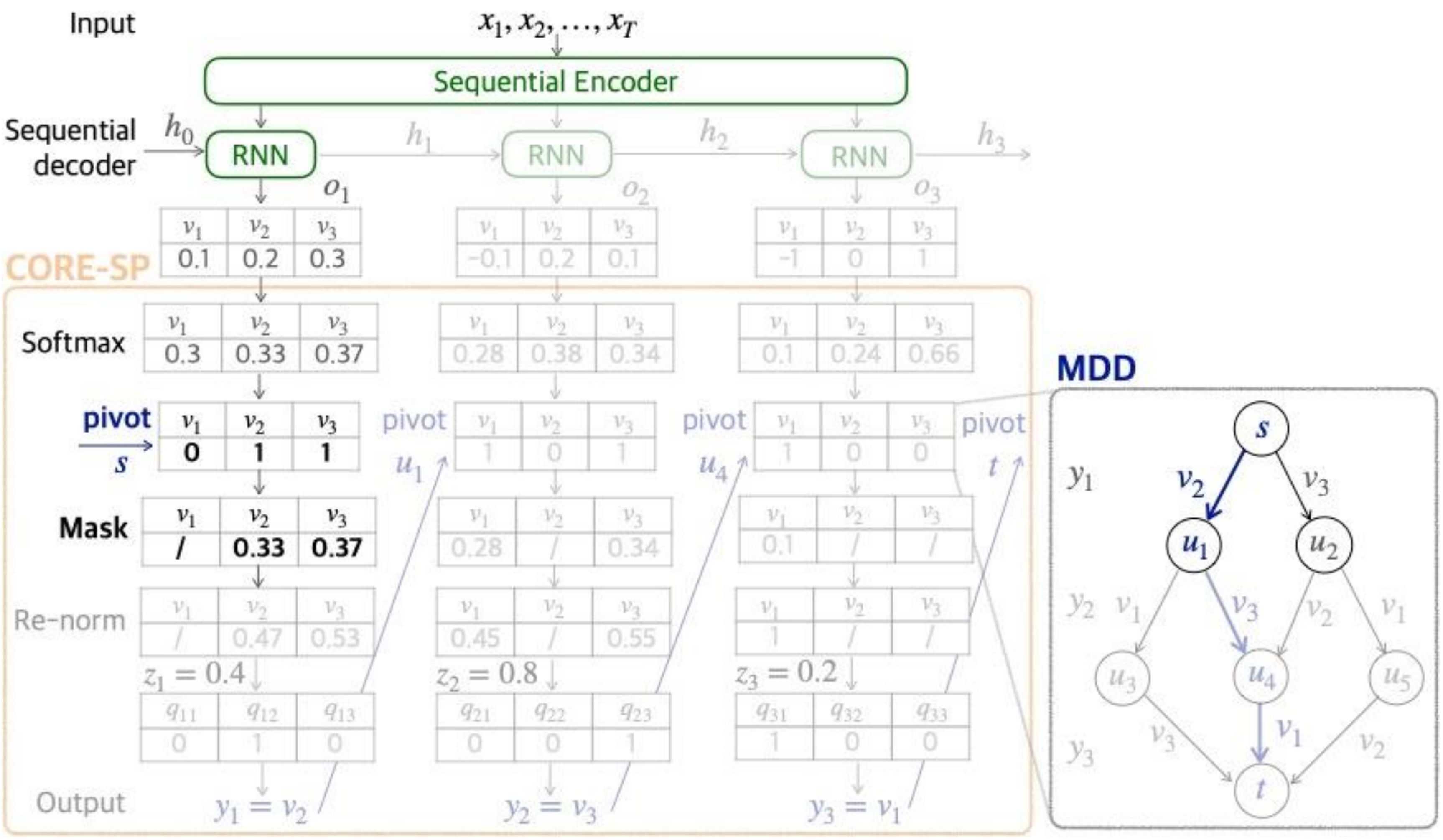
Constraint Reasoning

Compiled from Decision Diagrams; **embed** as a fully differentiable layer; filter out infeasible solutions.

(1)ML with safety, correctness, and/or fairness **assurances**.

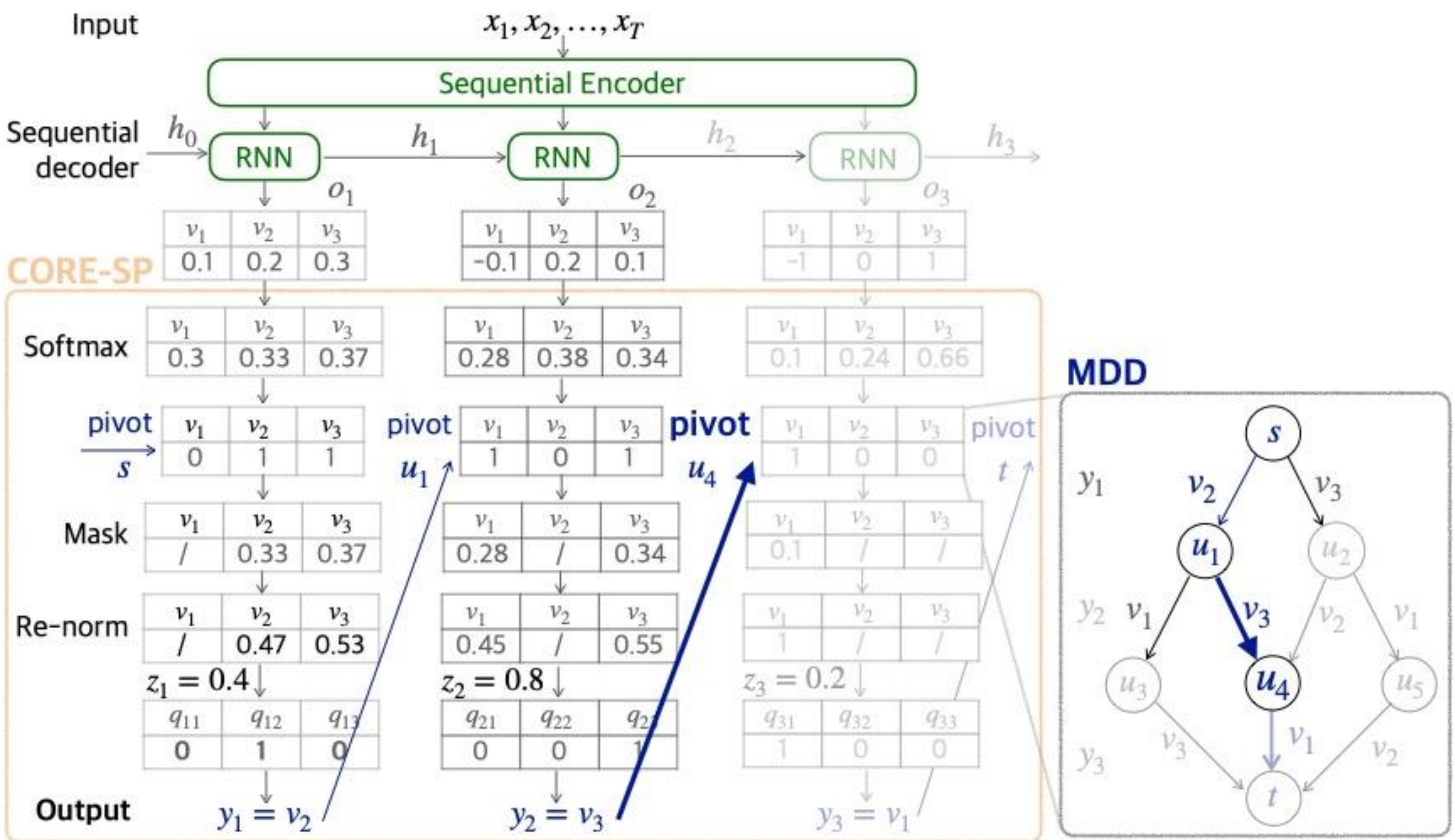
(2)Boost performance: learn faster, more accurate predictions.

CORE Filters Invalid Actions on MDD



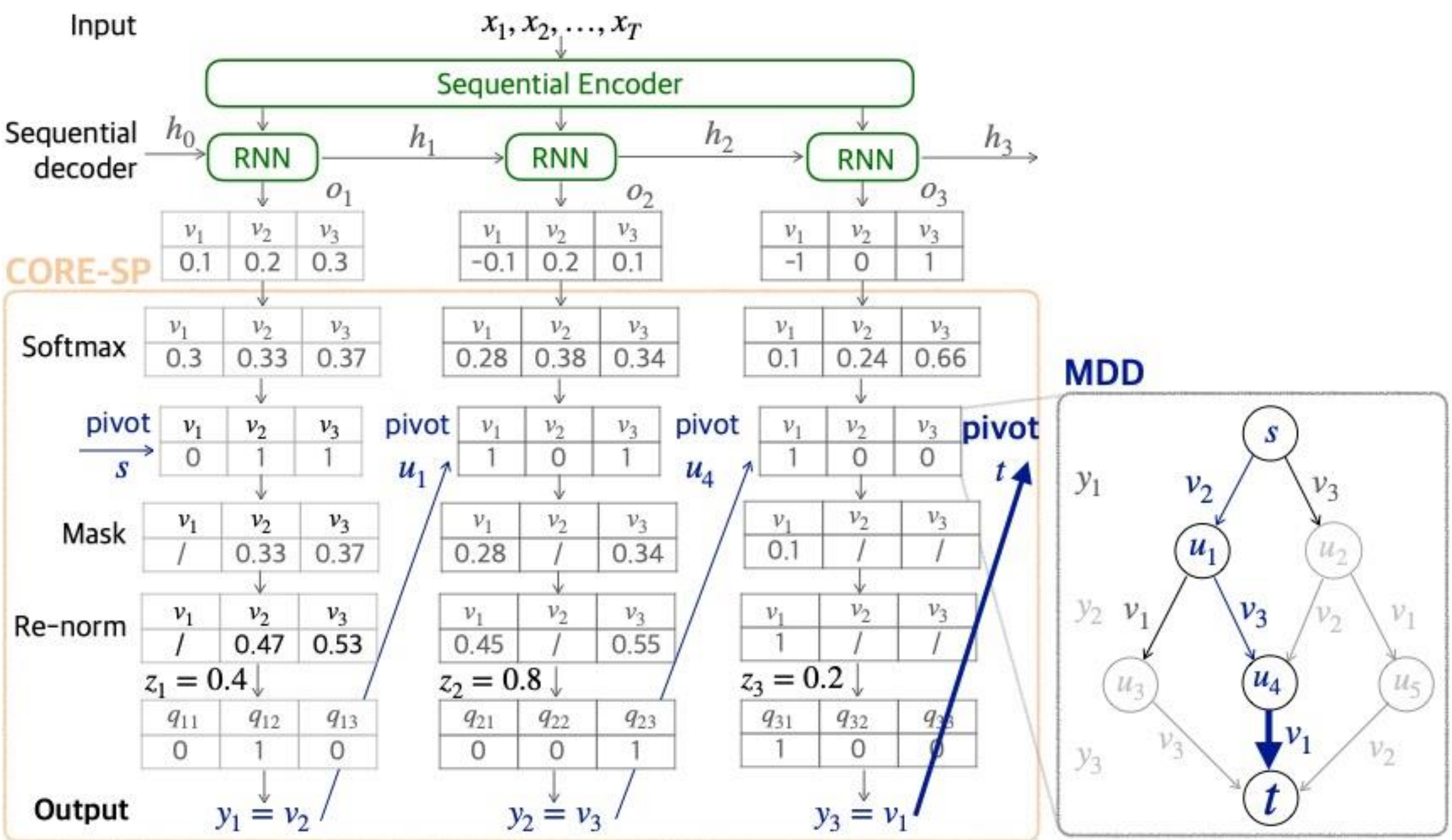
The MDD enforces Constraints C over output from the Seq2seq model.

CORE Filters Invalid Actions on MDD



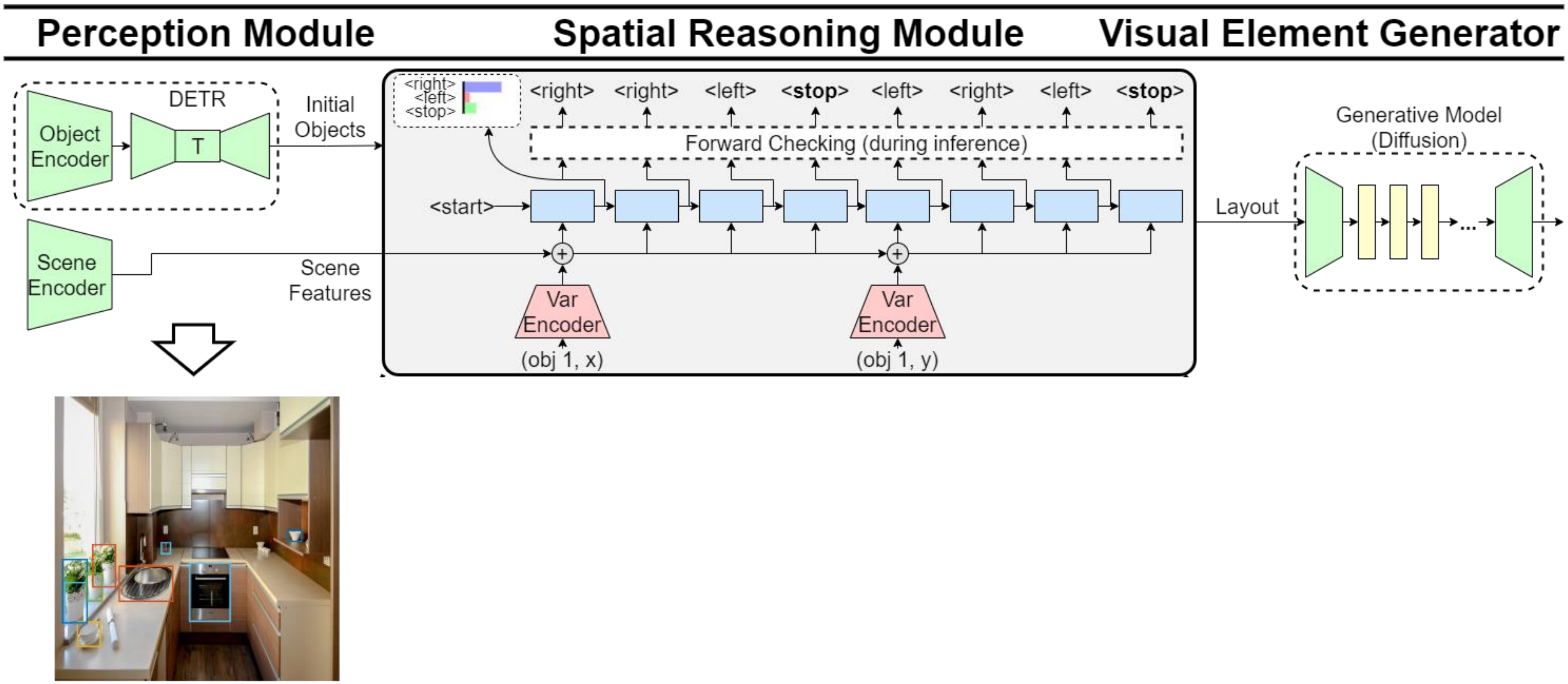
Pivot change from u_1 to u_4

CORE Filters Invalid Actions on MDD

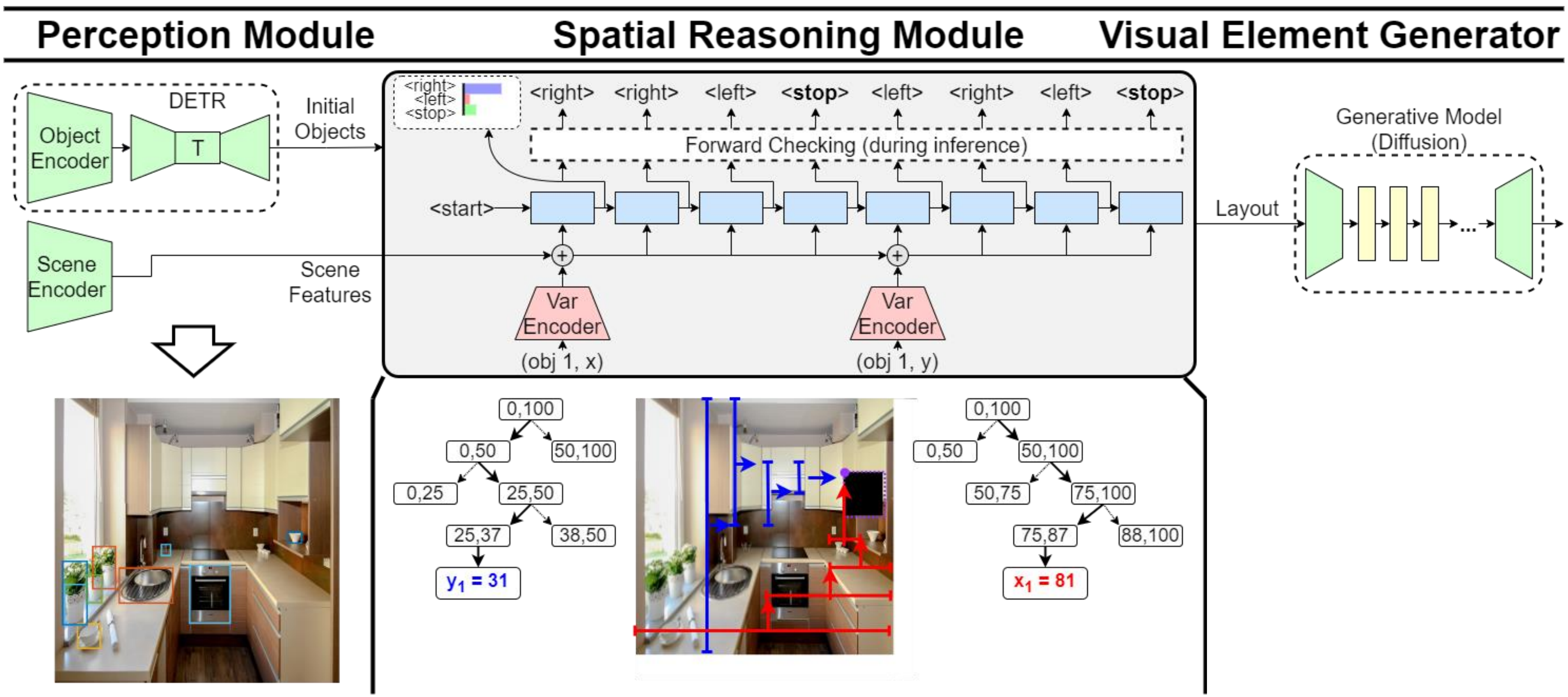


Pivot change from u_4 to t

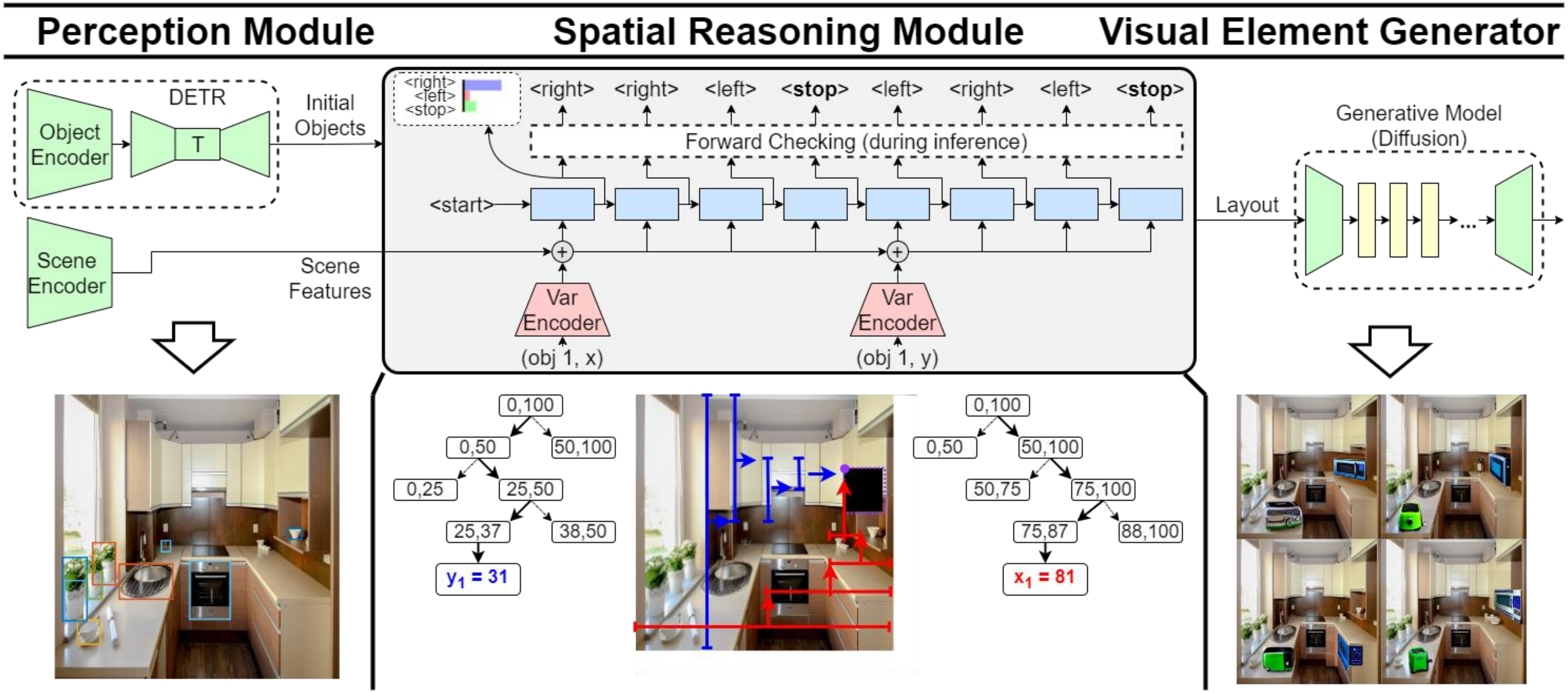
CORE Applied to Design Generation



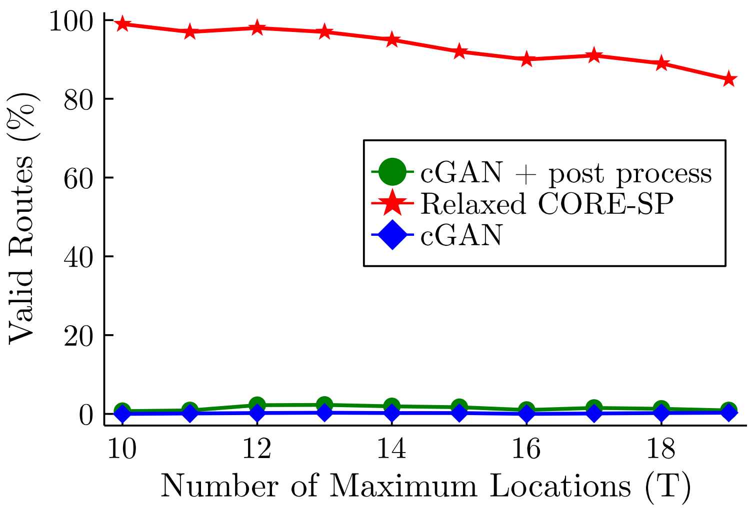
CORE Applied to Design Generation

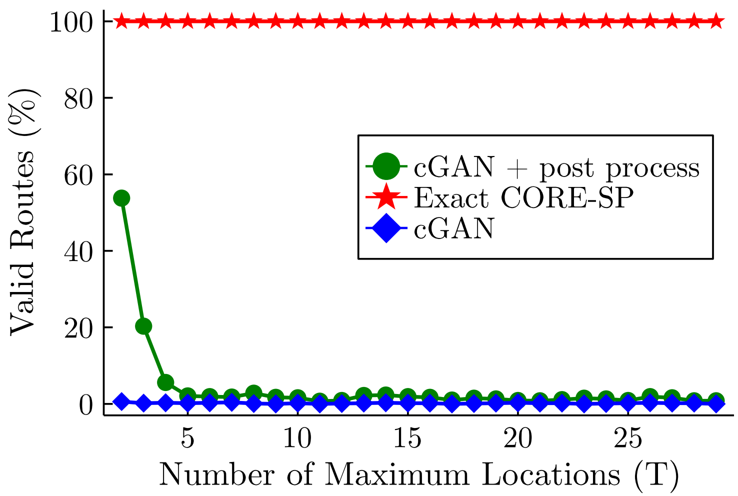


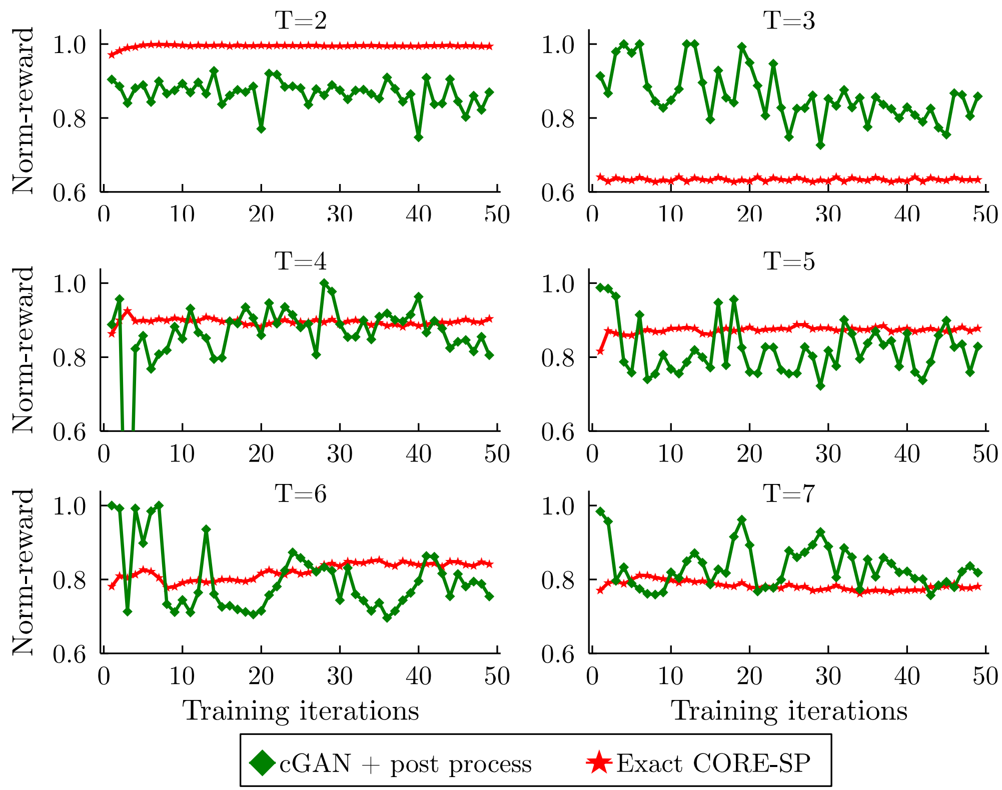
CORE Applied to Design Generation

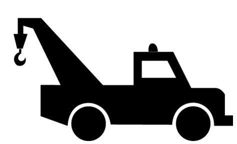


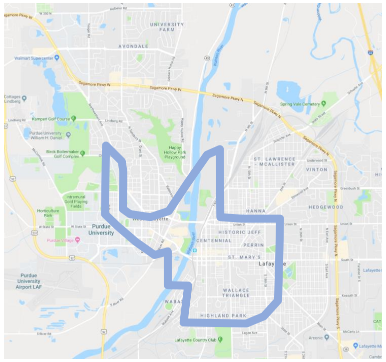
Service Vehicle Dispatching





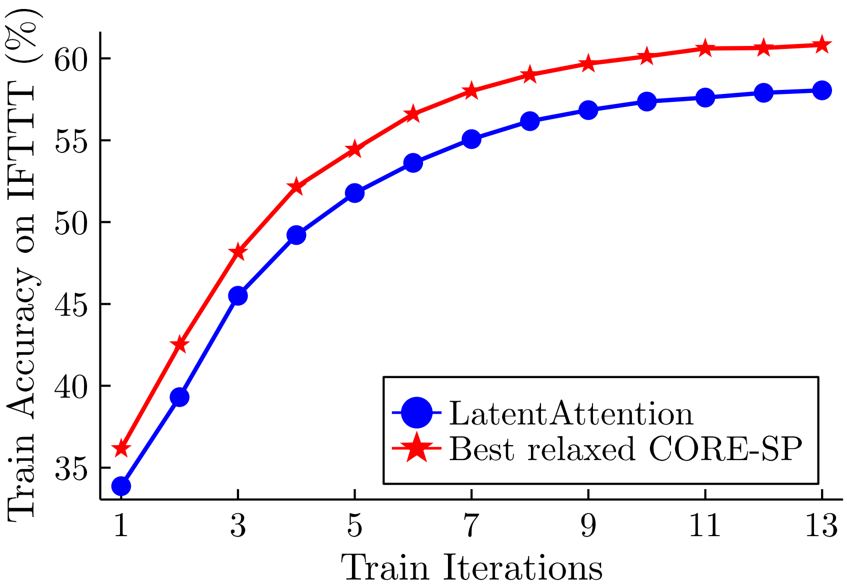


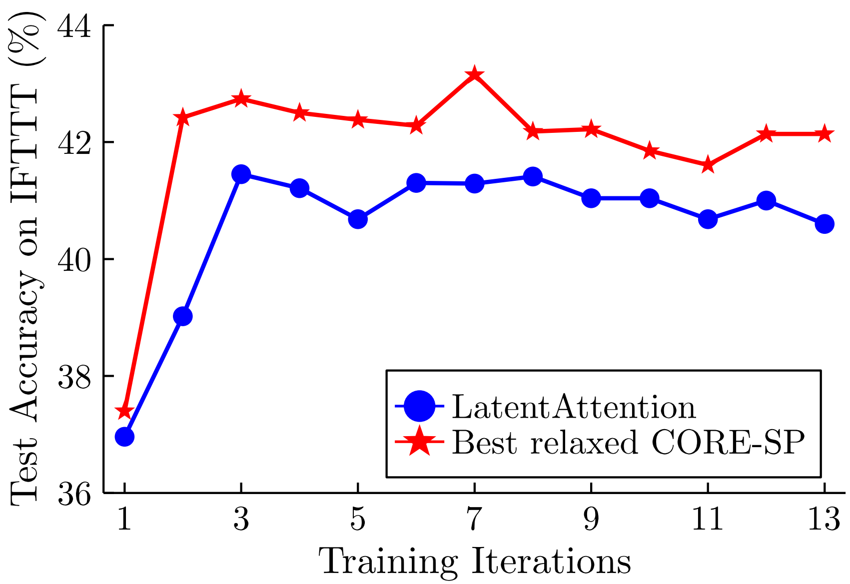




If-then Program Synthesis

	IFTTT			Zapier		
Methods	Width	Accuracy	Valid (%)	Width	Accuracy	Valid (%)
LatentAttention	N/A	42.17%	87.51%	N/A	31.74	88.00%
+best relaxed CORE	80	**44.12%**	**99.19%**	1200	**34.28**	**99.53%**
+ exact CORE	111	**43.07%**	**100%**	1353	**32.83**	**100%**





Program Synthesis from Natural Language

Text2SQL Prediction:

Input Table:

	Player	No.	Position	School
0	Antonio	21	Guard-Forward	Duke
1	Voshon	2	Guard	Minnesota
2	Marin	3	Guard-Forward	Butler CC

Input Query:

How many schools did player number 3 play at?

Output SQL Query:

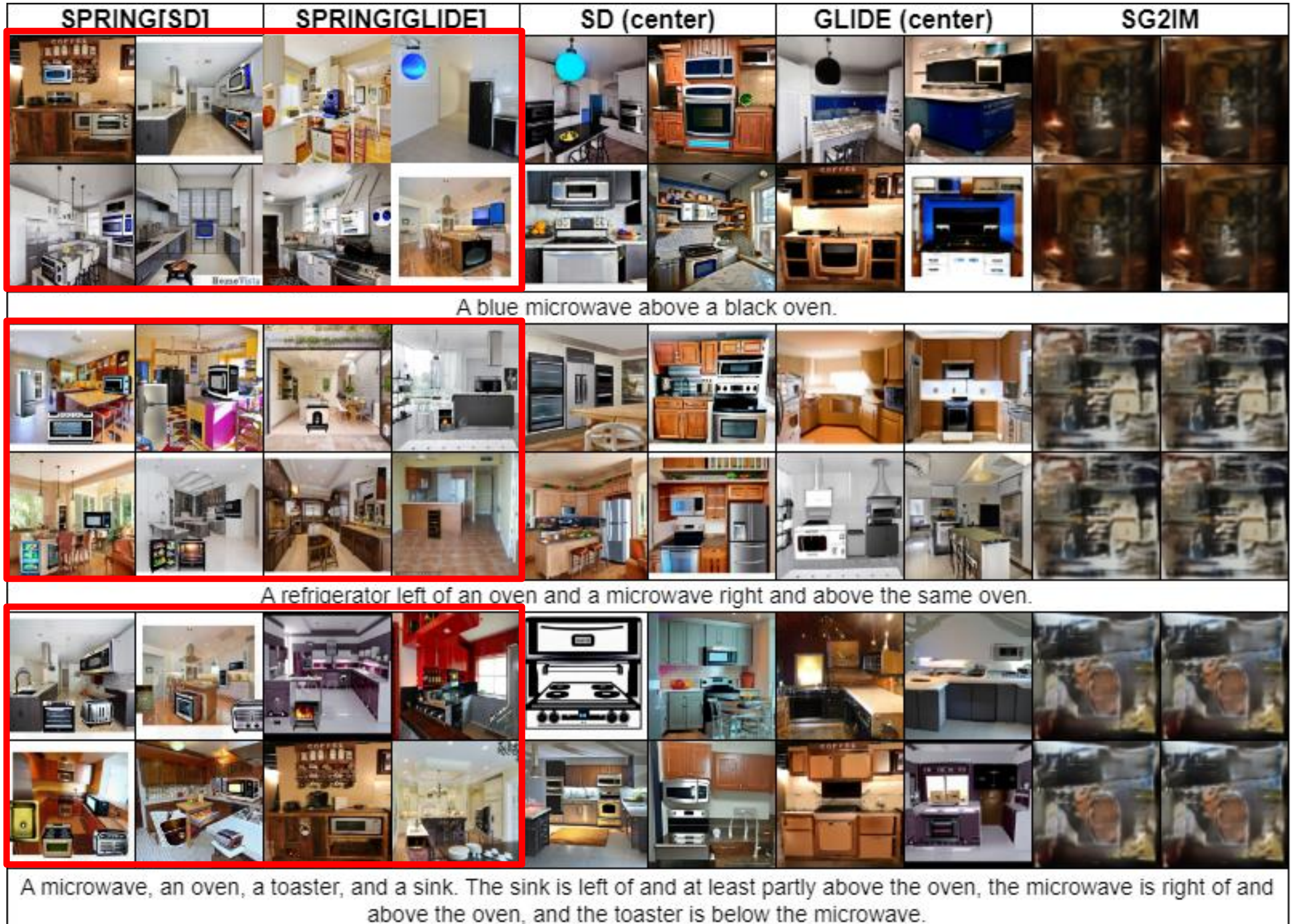
SELECT COUNT “School” WHERE “No.” = “3”

agg-op sel-col cond-col cond-op cond-val

Accuracy per component	Full test set SQLNova	Full test set CORE-SP	Moderate test set SQLNova	Moderate test set CORE-SP	Hard test set SQLNova	Hard test set CORE-SP
`sel-col`	96.3%	96.3%	96.4%	**97.0%**	96.6%	**97.7%**
`agg-op`	**89.8%**	89.7%	75.7%	**77.8%**	75.4%	**75.8%**
#`WHERE`	**98.1%**	97.9%	98.5%	**98.6%**	**98.9%**	98.5%
`cond-col`	93.6%	93.6%	**94.0%**	93.8%	93.6%	**93.7%**
`cond-op`	96.7%	**96.9%**	89.8%	**91.6%**	84.8%	**87.9%**
`where-val-idx`	94.5%	**94.8%**	89.4%	**92.3%**	86.7%	**87.5%**
`where-val`	94.7%	**94.9%**	89.3%	**92.2%**	86.4$	**87.1%**

Overall Accuracy	Full test set SQLNova	Full test set CORE-SP	Moderate test set SQLNova	Moderate test set CORE-SP	Hard test set SQLNova	Hard test set CORE-SP
Logical Accuracy	79.3%	**79.9%**	61.6%	**65.8%**	58.3%	**62.5%**
Execution Accuracy	85.5%	**86.1%**	75.4%	**79.1%**	76.1%	**78.0%**
Valid SQL	99.3%	**100.0%**	94.3%	**100%**	83.7%	**100%**

CORE for Design Generation



Conclusion

- Presented three application domains which need tight integration of machine learning with automated reasoning
  - Data-driven Operational Research
  - Program Synthesis from Natural Language
  - AI-driven Design Generation

- Demonstrated CORE as a hybrid learning + reasoning approach to
  - Generation structures satisfying constraints
  - Boost learning performance (higher accuracies, learning faster generalizing better)

- Future directions
  - Explore richer set of constraints; e.g., constraints stated in natural language
  - Complex constraint satisfaction problems beyond reach of exact decision diagrams.