Closed Loop Neural-Symbolic Learning via Integrating Neural Perception, Grammar Parsing, and Symbolic Reasoning



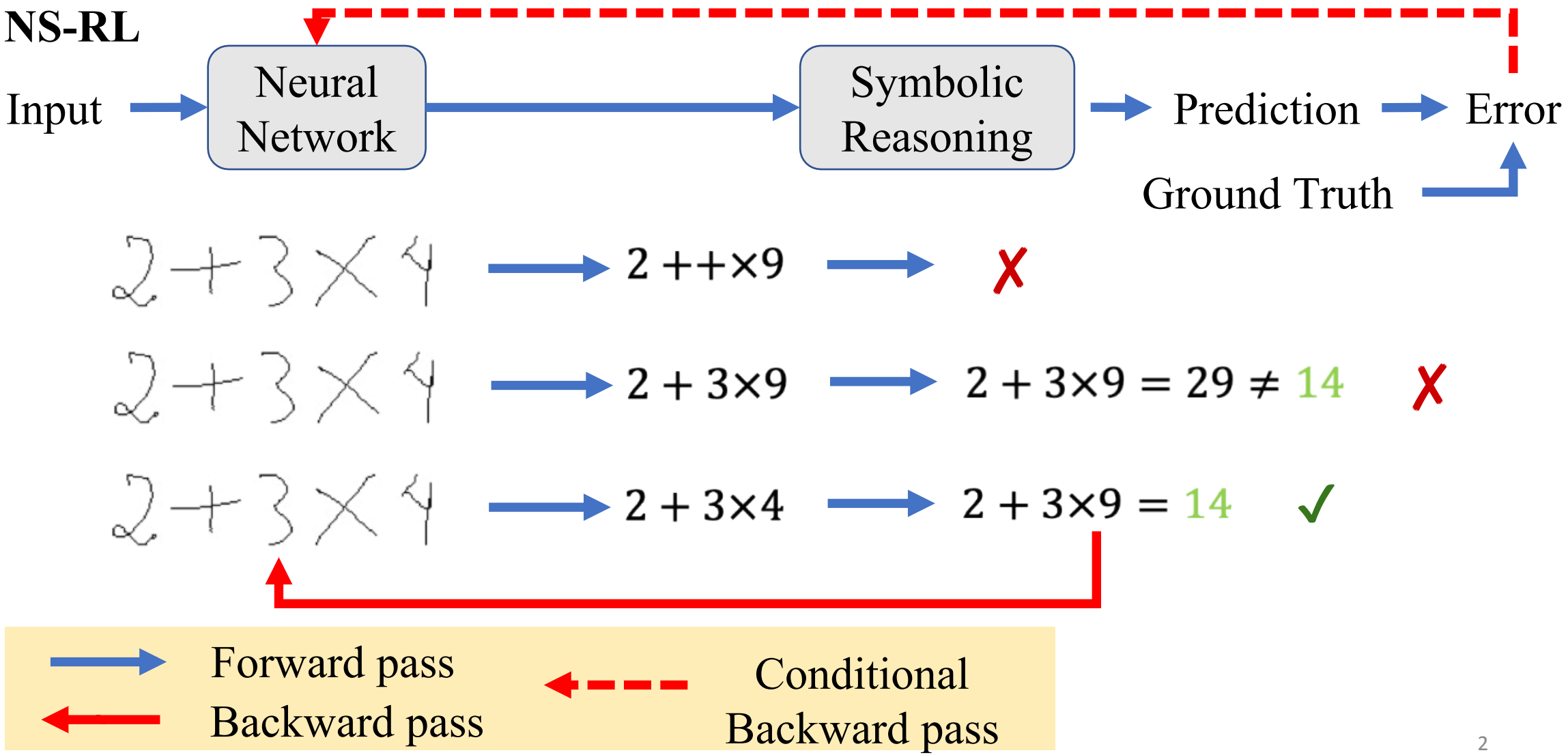



Qing Li[1] Siyuan Huang[1] Yining Hong[2] Yixin Chen[1] Ying Nian Wu[1] Song-Chun Zhu[1,2]

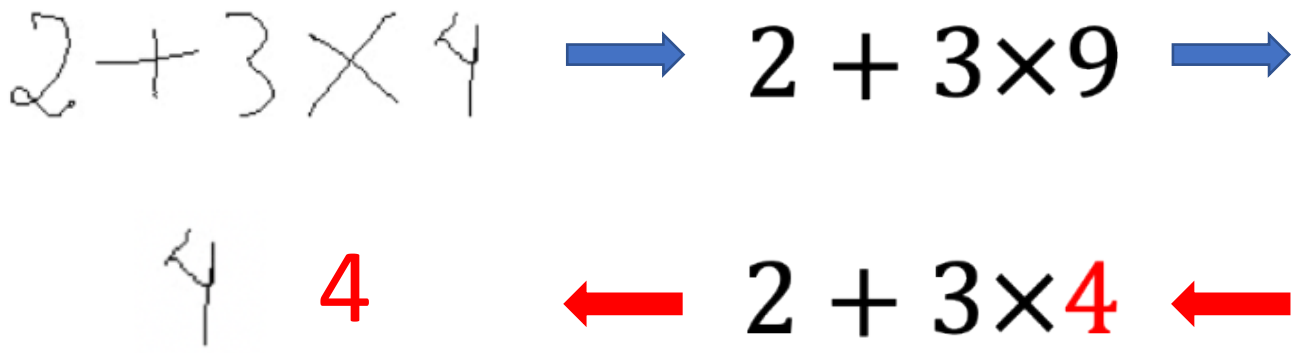
[1] Department of Statistics, UCLA

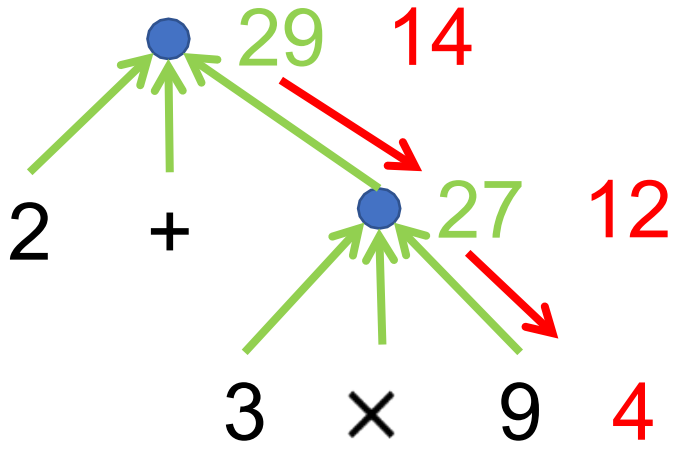
[2] Department of Computer Science, UCLA

Motivation


NS-RL
Input
Neural Network
Symbolic Reasoning
Prediction
Error
Ground Truth
2 ++×9
2 + 3×9
2 + 3×9 = 29 ≠ 14
2 + 3×4
2 + 3×9 = 14
Forward pass
Backward pass
Conditional Backward pass

How does human do this task?





1. Always generate a valid formula
2. Back-trace the error in the reasoning tree
3. Find the error source and propose a fix
4. Update the perception

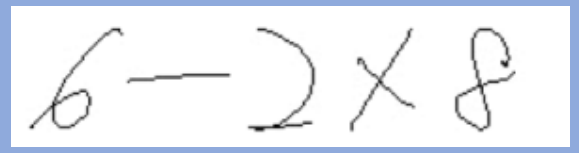
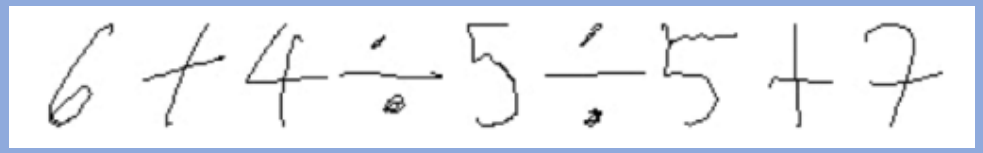
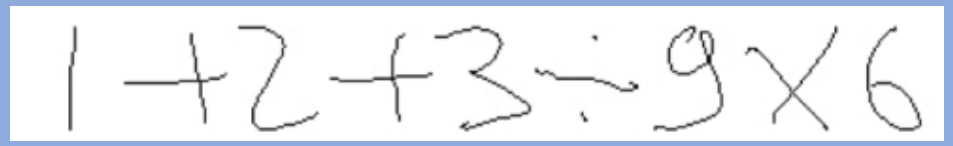
Prior knowledge

Abductive reasoning

Contributions

- **Grammar** to bridge neural network and symbolic reasoning
  - NGS: Neural perception + Grammar parsing + Symbolic reasoning
- **Back-search**
  - Mimic human's ability to learn from failures via abductive reasoning
- A new benchmark **HWF** for neural-symbolic learning
  - Hand-written Formula Recognition with Weak Supervision

Hand-written Formula Recognition (HWF)

Input	Latent	Output
	$6-2\times8$	-10
	$6+4\div5\div5+7$	13.16
	$1+2+3\div9\times6$	5

Weakly-supervised!

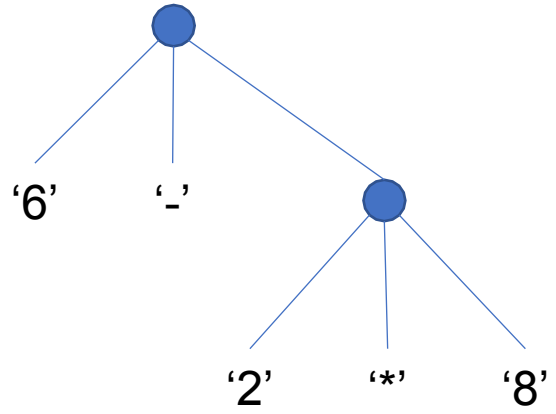
NGS

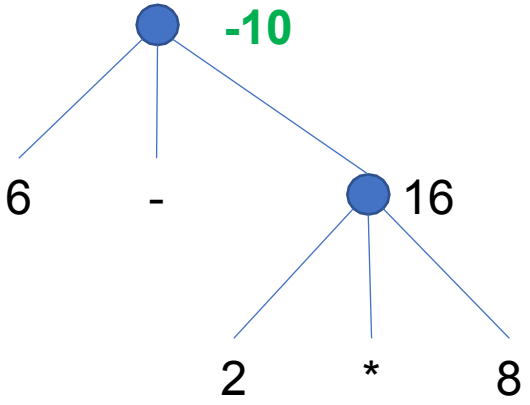
Neural Network

Grammar Parsing

Symbolic Reasoning

	‘0’	‘1’	…	‘9’	‘+’	‘-’	‘*’	‘/’
								
								
…								





Forward Pass (Inference)

- x : the input

- z : the formal representation for x, "$6 - 2 \times 8$"
  - Latent, unobserved
- y : the desired output, e.g., -10

Forward Pass (Inference)

Neural Perception

$$p_\theta(z|x) = softmax(\phi_\theta(z,x)),$$
$$= \frac{\exp(\phi_\theta(z,x))}{\sum_{z'}\exp(\phi_\theta(z',x))}$$

Grammar Parsing

$$\hat{z} = \arg\max_{z \in L(G)} p_\theta(z|x)$$

Symbolic Reasoning

$$\hat{y} = f(\hat{z};\Delta)$$

Backward Pass (Learning)

- Assumptions
  - Grammar and Symbolic reasoning are perfectly designed by hand, based on our domain knowledge.
  - Only the parameters of neural network need to be learned.
- Gradient descent cannot be applied directly
  - Grammar parsing and symbolic reasoning are non-differentiable.

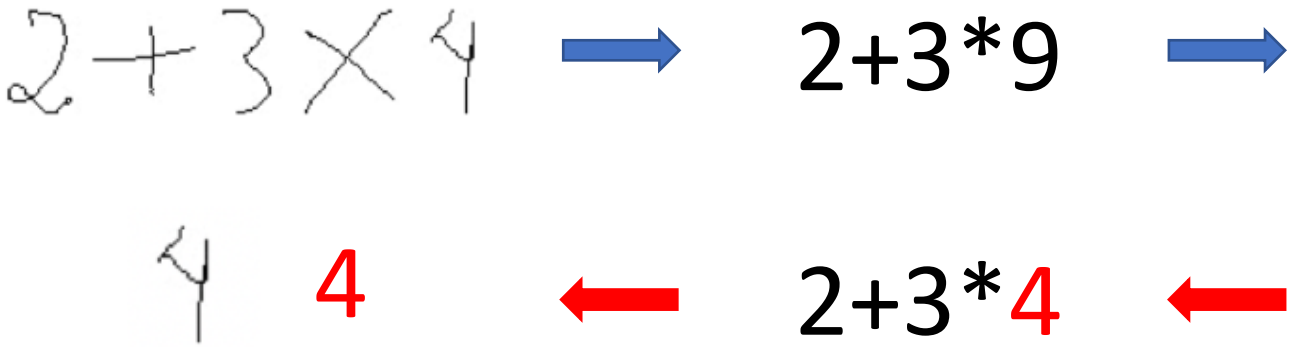
1-step Back-search (1-BS)

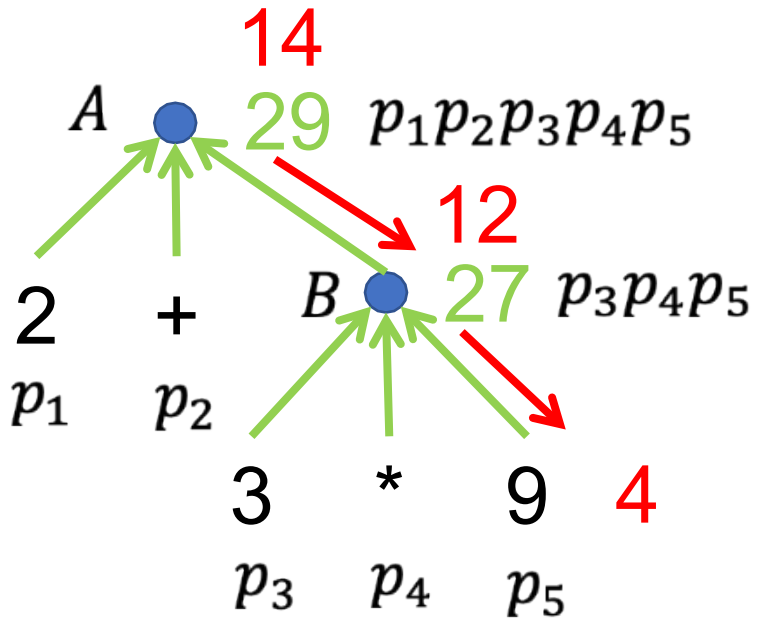
- Top-down search is guided by the bottom-up perception probability
- Dynamic Programming + Priority Queue

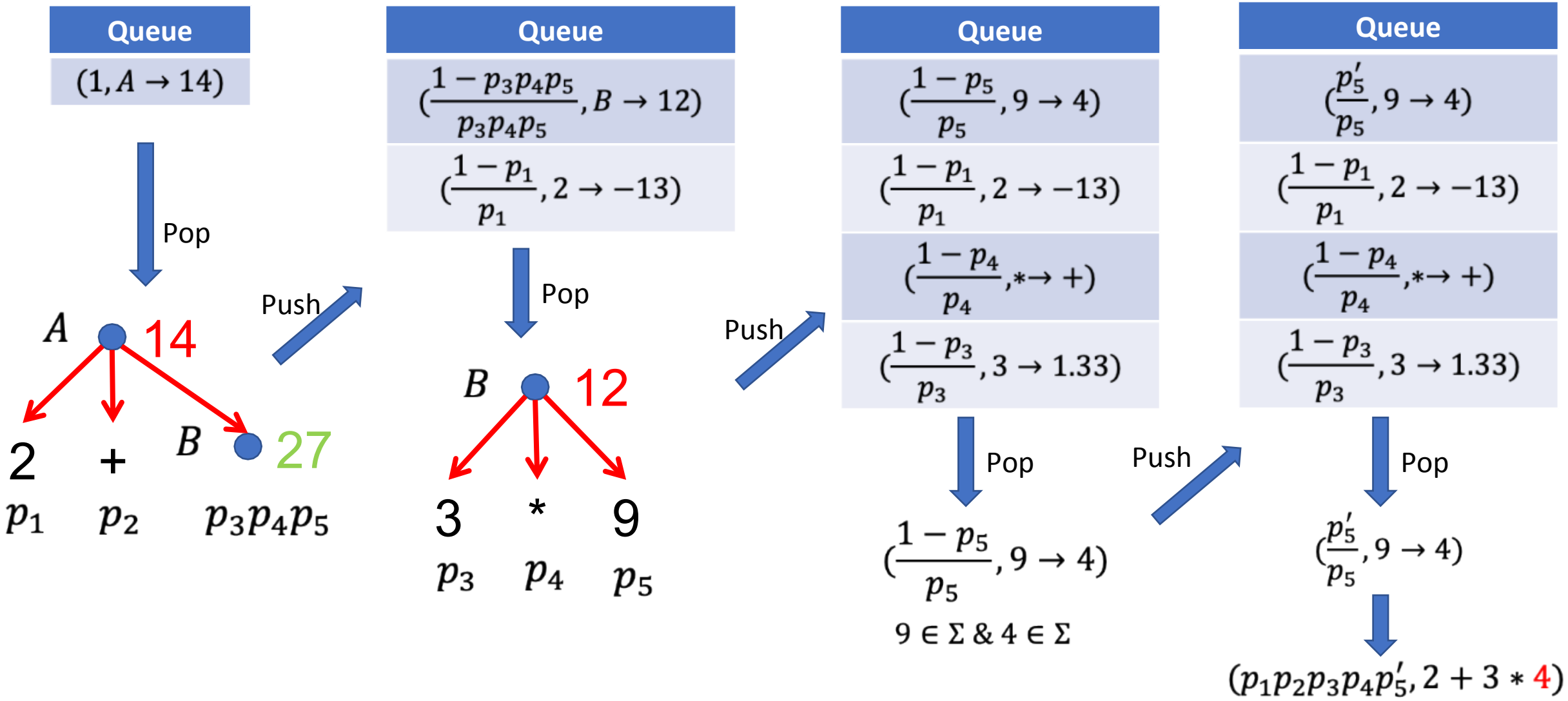
Algorithm 1 1-step back-search (1-BS)

Input: $\hat{z}, S, y$
$q = PriorityQueue()$
$q.push(S, y, 1)$
while $A, \alpha_A, p = q.pop()$ **do**
  if $A \in \Sigma$ **then**
    $z^* = \hat{z}(A \rightarrow \alpha_A)$
    return z^*
  for $B \in child(A)$ **do**
    $\alpha_B = solve(B, A, \alpha_A | \Delta, G)$
    $q.push(B, \alpha_B, p(B \rightarrow \alpha_B))$
return $\emptyset$

A running example for 1-BS






Queue
$(1, A \rightarrow 14)$
Pop
A
14
2
+
B
27
p_1
p_2
$p_3p_4p_5$
Push
Queue
$(\frac{1-p_3p_4p_5}{p_3p_4p_5}, B \rightarrow 12)$
$(\frac{1-p_1}{p_1}, 2 \rightarrow -13)$
Pop
B
12
3
*
9
p_3
p_4
p_5
Push
Queue
$(\frac{1-p_5}{p_5}, 9 \rightarrow 4)$
$(\frac{1-p_1}{p_1}, 2 \rightarrow -13)$
$(\frac{1-p_4}{p_4}, * \rightarrow +)$
$(\frac{1-p_3}{p_3}, 3 \rightarrow 1.33)$
Pop
$(\frac{1-p_5}{p_5}, 9 \rightarrow 4)$
$9 \in \Sigma \ \& \ 4 \in \Sigma$
Push
Queue
$(\frac{p_5'}{p_5}, 9 \rightarrow 4)$
$(\frac{1-p_1}{p_1}, 2 \rightarrow -13)$
$(\frac{1-p_4}{p_4}, * \rightarrow +)$
$(\frac{1-p_3}{p_3}, 3 \rightarrow 1.33)$
Pop
$(\frac{p_5'}{p_5}, 9 \rightarrow 4)$
$(p_1p_2p_3p_4p_5', 2 + 3 * 4)$

Why can BS be better than RL?

- Learning as Maximum Marginal Likelihood
- REINFORCE as Rejection Sampling
- m-BS as MCMC sampling
  - Metropolis-Hastings sampler

Learning as Maximum Marginal Likelihood

Marginal likelihood

$$p(y|x) = \sum_z p(y, z|x) = \sum_z p(y|z)p_\theta(z|x)$$

$$\begin{aligned}
\nabla_\theta L(x, y) &= \nabla_\theta \log p(y|x) \\
&= \frac{1}{p(y|x)} \nabla_\theta p(y|x) \\
&= \sum_z \frac{p(y|z)p_\theta(z|x)}{\sum_{z'} p(y|z')p_\theta(z'|x)} \nabla_\theta \log p_\theta(z|x) \\
&= \mathbb{E}_{z \sim p(z|x,y)}[\nabla_\theta \log p_\theta(z|x)] \\
&\approx \nabla_\theta \log p_\theta(\hat{z}|x), \hat{z} \sim p(z|x, y)
\end{aligned}$$

Monte Carlo sampling

Posterior distribution

$$p(z|x,y) = \frac{p(y|z)p_\theta(z|x)}{\sum_{z'} p(y|z')p_\theta(z'|x)}$$

$$= \begin{cases} 0, & \text{for} \quad z \notin Q \\ \frac{p_\theta(z|x)}{\sum_{z' \in Q} p_\theta(z'|x)}, & \text{for } z \in Q \end{cases}$$

$$Q = \{z : p(y|z) = 1\} = \{z : f(z;\Delta) = y\}$$

REINFORCE as Rejection Sampling

- Target distribution: $p(z|x,y)$
- Proposal distribution: $p_\theta(z|x)$
- Rejection sampling:
  1. Sample $\hat{z}$ from $p_\theta(z|x)$
  2. If $f(\hat{z};\Delta) \neq y$ $\dashrightarrow$ $p(\hat{z}|x,y) = 0$

     reject $\hat{z}$

     else $\dashrightarrow$ $\frac{p(\hat{z}|x,y)}{M p_\theta(\hat{z}|x)} = 1$, where $M = \frac{1}{\sum_{z' \in Q} p_\theta(z'|x)}$

     accept $\hat{z}$

m-BS as MCMC Sampling

Algorithm 2 m-step back-search (m-BS)

Hyperparameters: T, λ
Input: $\hat{z}, y$
$z^{(0)} = \hat{z}$
for $t \leftarrow 0$ to $T-1$ **do**
  $z^* = 1\text{-BS}(z^t, y)$
  draw $u \sim \mathcal{U}(0,1)$
  if $u \leq \lambda$ and $z^* \neq \emptyset$ **then**
    $z^{t+1} = z^*$
  else
    $z^{t+1} = \text{RANDOMWALK}(z^t)$
return z^T

function RANDOMWALK(z^t)
  sample $z^* \sim g(\cdot|z^t)$
  compute acceptance ratio $a = min(1, \frac{p_\theta(z^*|x)}{p_\theta(z^t|x)})$
  draw $u \sim \mathcal{U}(0,1)$
  $z^{t+1} = \begin{cases} z^*, & \text{if } u \leq a \\ z^t, & \text{otherwise.} \end{cases}$

- m-BS is a Metropolis-Hastings sampler for $p(z|x,y)$

$$\hat{z} \sim p_\theta(z|x) \rightsquigarrow z^* \in Q$$

[Proof in Sec. 3.2.3]

$$g(z_1|z_2) = Poisson(d(z_1, z_2); \beta)$$

where $d(z_1, z_2)$ denotes the edit distance between z_1, z_2

Experiments

- Hand-written Formula Recognition
- Neural-symbolic VQA

Hand-written Formula Recognition

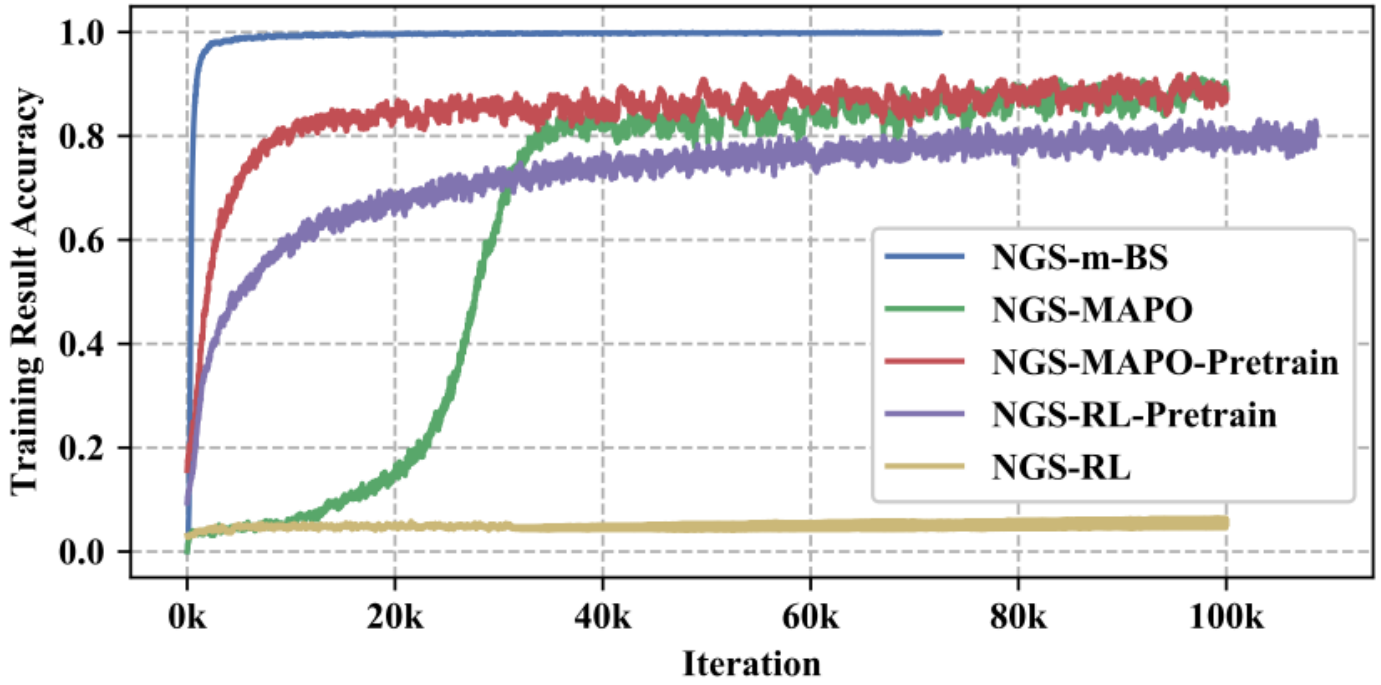
- Dataset
  - Built from the CROHME challenge
  - 10k expressions for training, 2k expressions for testing
- Evaluation
  - Symbol accuracy, Result Accuracy
- Models
  - NGS-RL, NGS-RL-Pretrained
  - NGS-MAPO*, NGS-MAPO-Pretrained
  - NGS-BS

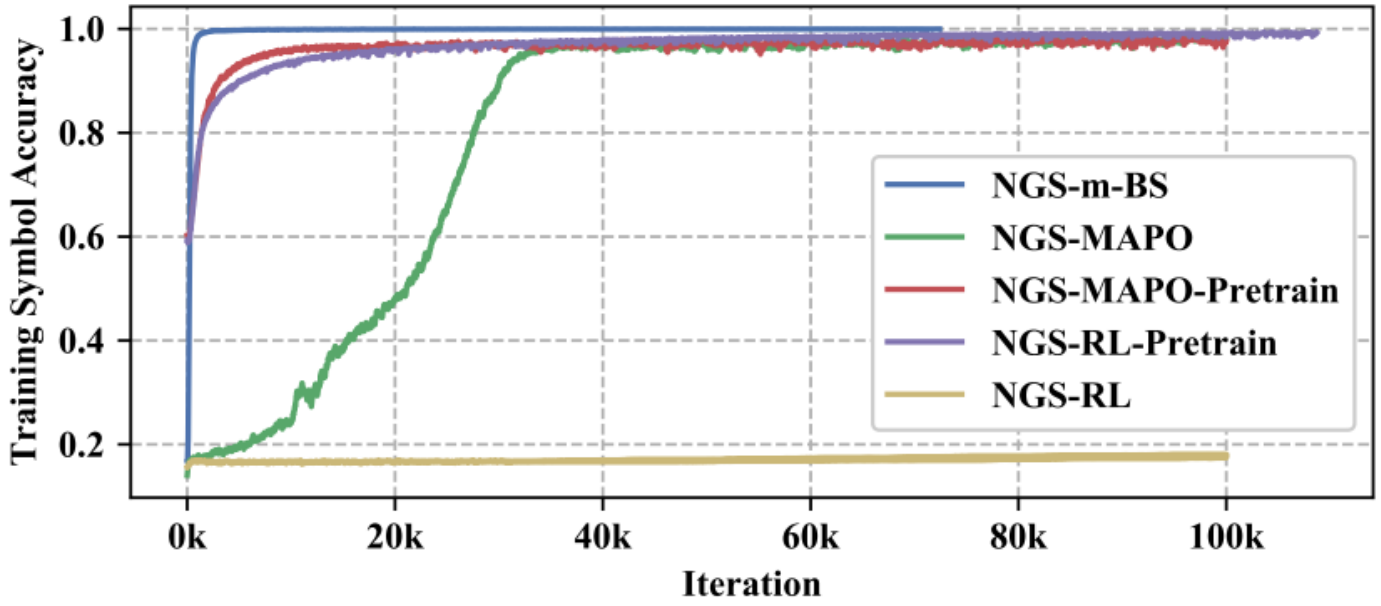
*Pretrain NN on a set of fully-supervised data
*Memory-Augmented Policy Optimization [1]

[1] Liang, Chen, et al. "Memory augmented policy optimization for program synthesis and semantic parsing." *Advances in Neural Information Processing Systems*. 2018.

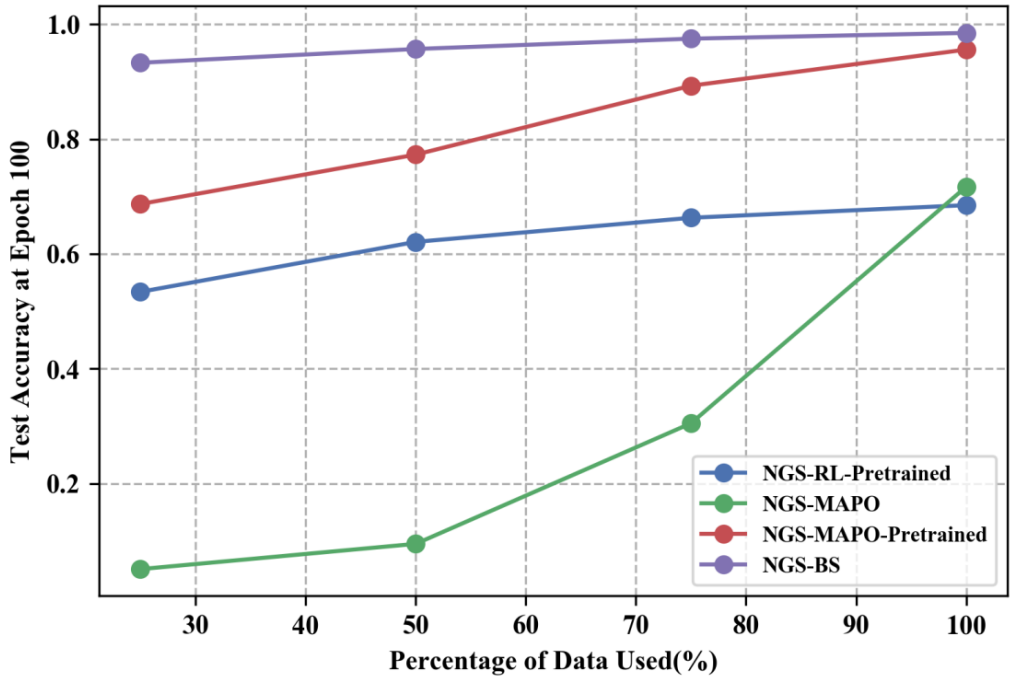
Learning curves

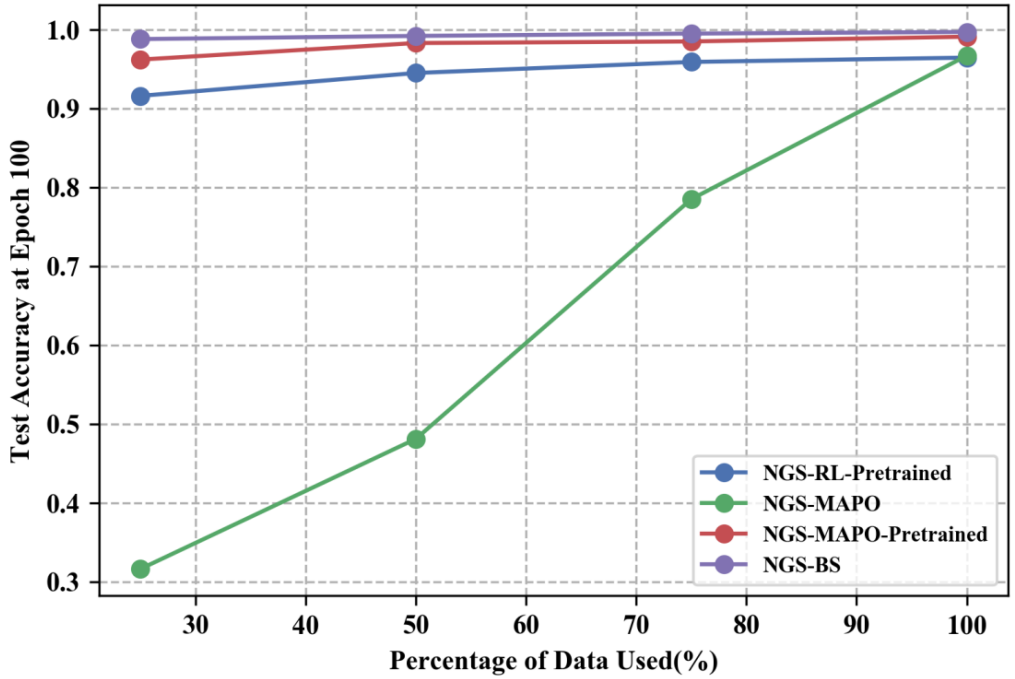
1. NGS-RL fails without pretraining
2. NGS-MAPO works without pretraining but takes a long time to start improving (cold start).
3. Both NGS-RL and NGS-MAPO have noisy learning curves.
4. NGS-BS doesn't suffer from the cold start.
5. NGS-BS converges much faster and the learning curve is smooth.
6. NGS-BS achieves nearly perfect accuracy.





Data efficiency

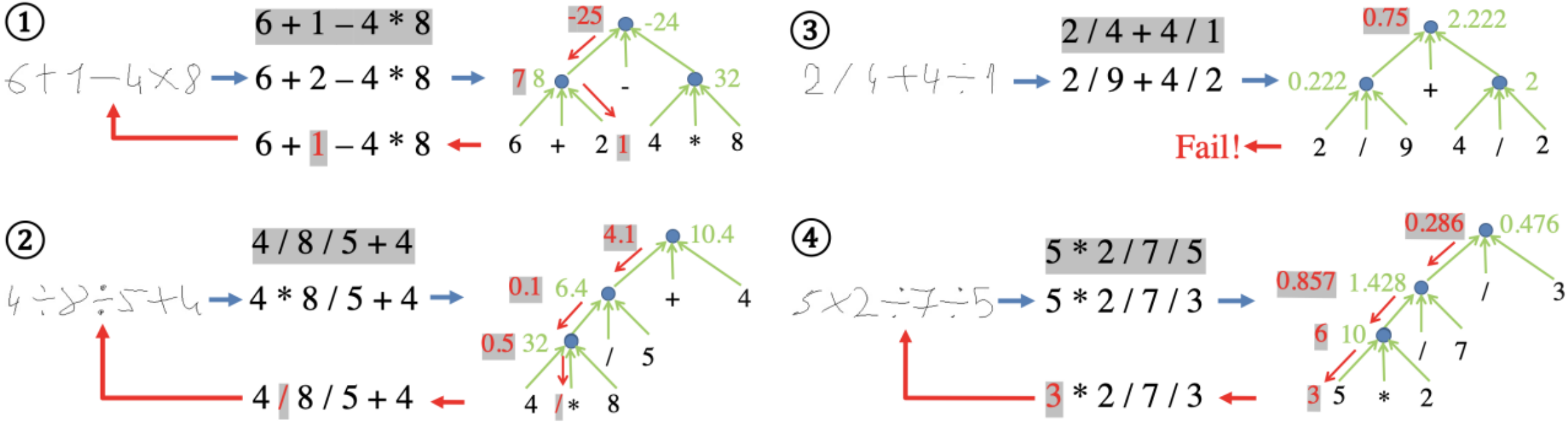




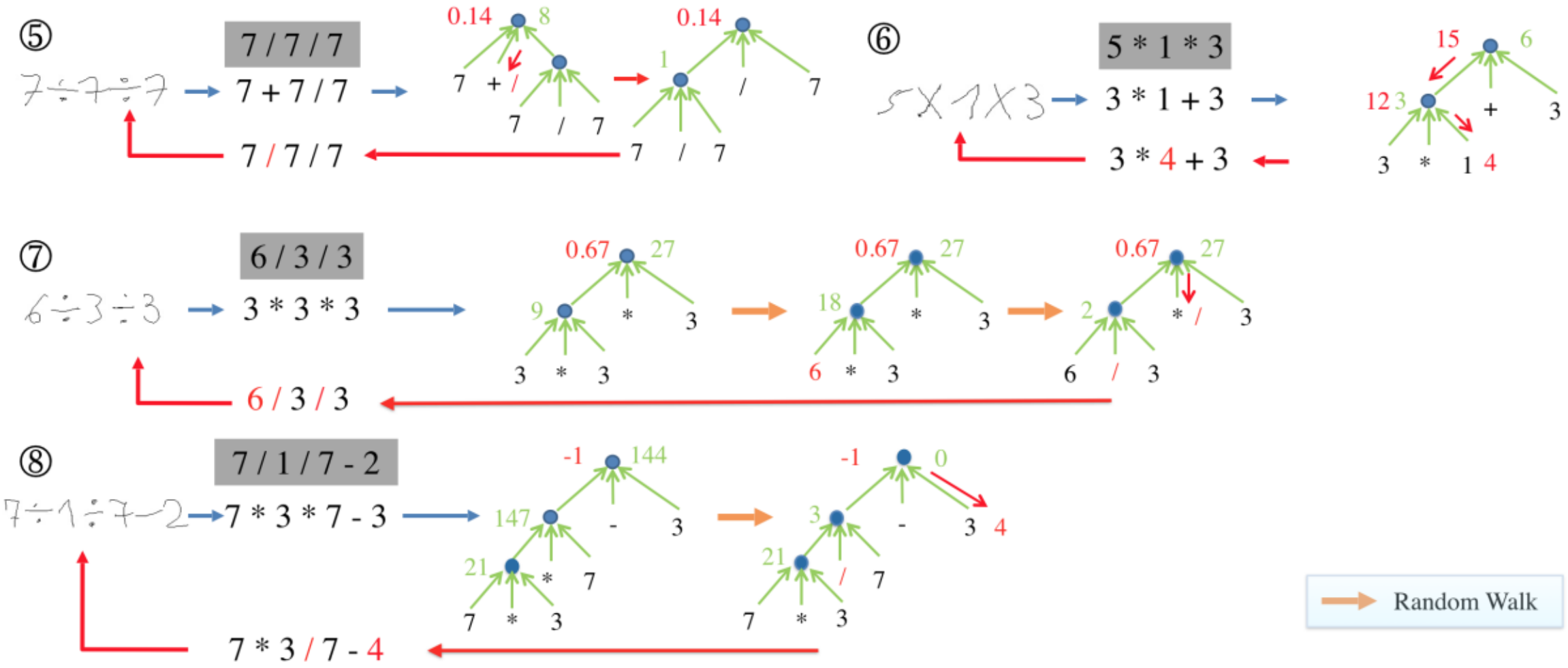
	25%	50 %	75 %	100%
NGS-RL	0.035	0.036	0.034	0.034
NGS-MAPO	0.051	0.095	0.305	0.717
NGS-RL-Pretrained	0.534	0.621	0.663	0.685
NGS-MAPO-Pretrained	0.687	0.773	0.893	0.956
NGS-BS	**0.933**	**0.957**	**0.975**	**0.985**

25%	50 %	75 %	100%
0.170	0.170	0.170	0.170
0.316	0.481	0.785	0.967
0.916	0.945	0.959	0.964
0.962	0.983	0.985	0.991
0.988	**0.992**	**0.995**	**0.997**

Examples

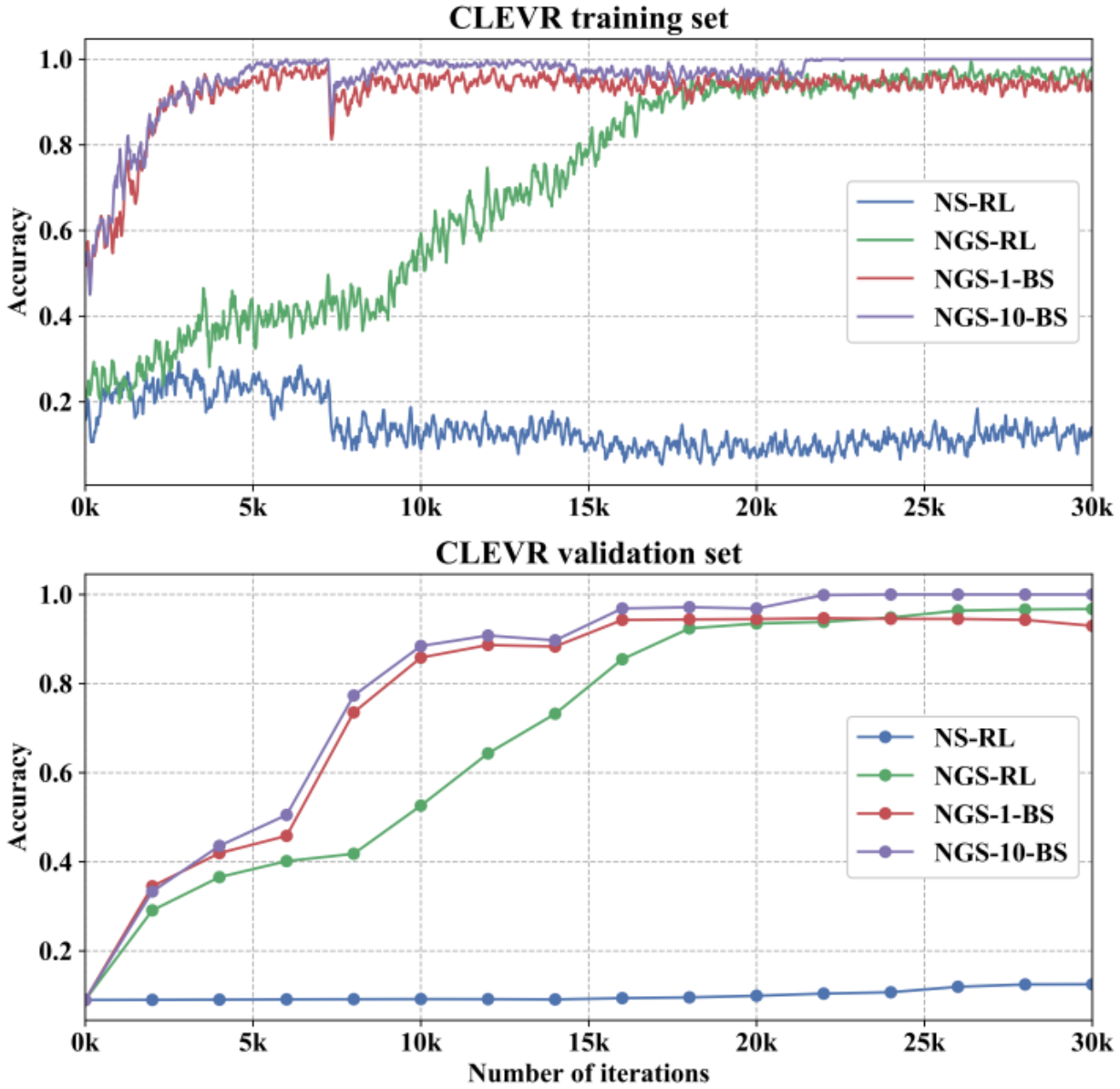

①
6+1−4×8
6 + 1 – 4 * 8
6 + 2 – 4 * 8
6 + 1 – 4 * 8
7 8
-25
-24
32
6 + 2 1 4 * 8
②
4÷8÷5+4
4 / 8 / 5 + 4
4 * 8 / 5 + 4
4 / 8 / 5 + 4
0.1 6.4
4.1
10.4
+ 4
0.5 32
/ 5
4 /* 8
③
2/4+4/1
2 / 4 + 4 / 1
2 / 9 + 4 / 2
Fail!
0.75
2.222
0.222
+
2
2 / 9 4 / 2
④
5×2÷7÷5
5 * 2 / 7 / 5
5 * 2 / 7 / 3
3 * 2 / 7 / 3
0.286
0.476
0.857
1.428
/ 3
6 10
/ 7
3 5 * 2

Examples


⑤
7÷7÷7
7 / 7 / 7
7 + 7 / 7
7 / 7 / 7
0.14
8
7 + /
7 / 7
0.14
1
/
7
7 / 7
⑥
5X1X3
5 * 1 * 3
3 * 1 + 3
3 * 4 + 3
15
6
12 3
+
3
3 * 1 4
⑦
6÷3÷3
6 / 3 / 3
3 * 3 * 3
6 / 3 / 3
0.67
27
9
*
3
3 * 3
18
6 * 3
2
* /
6 / 3
⑧
7÷1÷7−2
7 / 1 / 7 - 2
7 * 3 * 7 - 3
7 * 3 / 7 - 4
-1
144
147
-
3
21
*
7
7 * 3
-1
0
3
-
3 4
/
Random Walk

Neural Symbolic VQA

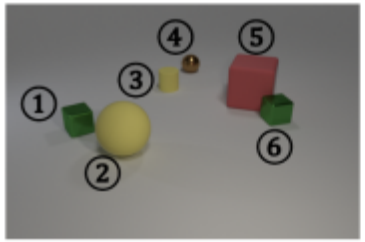
- NS-VQA on CLEVR [1]
- Replace the Seq2Seq question parser with Pointer Network

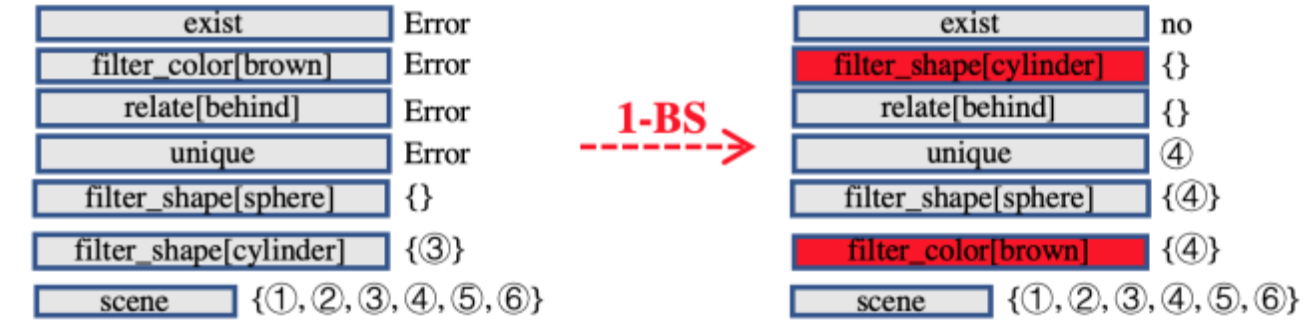


[1] Yi, Kexin, et al. "Neural-symbolic VQA: Disentangling reasoning from vision and language understanding." NeurIPS 2018.

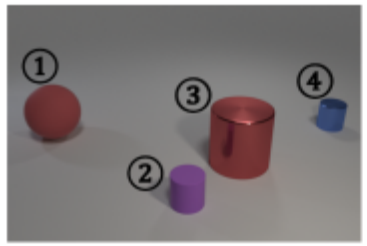
Examples

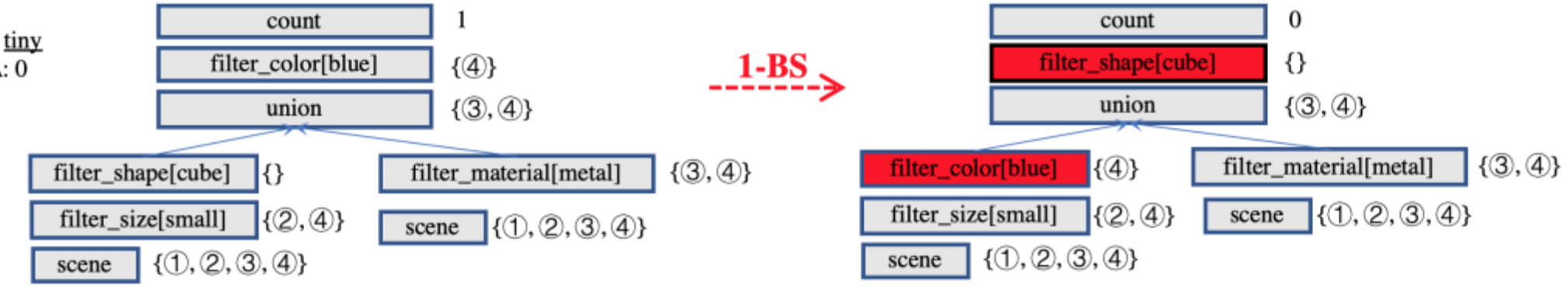
Q: Are there any cylinders behind the brown ball? **A:** no





Q: How many cubes are either tiny blue objects or metal things? **A:** 0





Conclusions & Future works

- RL is inefficient for weakly-supervised neural-symbolic learning.
- Back-Search boosts neural-symbolic learning.
- m-BS is a Metropolis-Hastings sampler for the posterior distribution.
- Back-search might be applied to a variety of neural-symbolic tasks, such as semantic parsing, math word problem.
- How to incorporate grammar learning and logic induction is still an open problem.

Thank you!

Project: https://liqing-ustc.github.io/NGS/

Code: https://github.com/liqing-ustc/NGS