A Limitations

In this section, we discuss the limitations of our framework and outline some potential solutions.

Expressiveness. The theoretical capability of the cardinality constraint to represent any propositional logic formula does not necessarily imply the practical ability to learn any such formula in our framework; this remains a challenge. Fundamentally, logical constraint learning is an inductive method, and thus different learning methods would have different inductive biases. Cardinality constraint-based learning is more suitable for tasks where the logical constraints can be straightforwardly translated into the cardinality form. A typical example of such a task is Sudoku, where the target CNF formula consists of at least 8,829 clauses [Lynce and Ouaknine, 2006], while the total number of target cardinality constraints stands at a mere 324.

Technically, our logical constraint learning prefers equality constraints (e.g., $x + y = 2$), which actually induce logical conjunction (e.g., $x \wedge y = \mathrm{T}$) and may ignore potential logical disjunction which is represented by inequality constraints (e.g., $x \vee y = \mathrm{T}$ is expressed by $x + y \geq 1$). To overcome this issue, a practical trick is to introduce some auxiliary variables, which is commonly used in linear programming [Fang and Puthenpura, 1993]. Consider the disjunction $x \vee y = \mathrm{T}$; here, the auxiliary variables z_1, z_2 help form two equalities, namely, $x + y + z_1 = 2$ for $(x, y) = (\mathrm{T}, \mathrm{T})$ and $x + y + z_2 = 1$ for $(x, y) = (\mathrm{T}, \mathrm{F})$ or $(x, y) = (\mathrm{F}, \mathrm{T})$. One can refer to the Chain-XOR task (cf. Section G.1) for a concrete application of auxiliary variables.

Reasoning efficiency. The reasoning efficiency, particularly that of SMT solvers, during the inference phase can be a primary bottleneck in our framework. For instance, in the self-driving path planning task, when we scale the map size up to a 20×20 grid involving 800 Boolean variables (400 variables for grid obstacles and 400 for path designation), the Z3 MaxSAT solver would require more than two hours for some input.

To boost reasoning efficiency, there are several practical methods that could be applied. One straightforward method is to use an integer linear program (ILP) solver (e.g., Gurobi) as an alternative to the Z3 MaxSAT solver. In addition, some learning-based methods (e.g., Balunovic et al. [2018]) may enhance SMT solvers in our framework. Nonetheless, we do not expect merely using a more efficient solver can resolve the problem. The improve the scalability, a more promising way is to combine System 1 and System 2 also in the inference stage (e.g., Cornelio et al. [2023]). Generally speaking, in the inference stage, neural perception should first deliver a partial solution, which is then completed by the reasoning engine. Such a paradigm ensures fast reasoning via neural perception, drastically reducing the logical variables that require solving by the exact reasoning engine, thereby also improving its efficiency.

B Proofs of DC technique

<u>**Notations.**</u> We define $\boldsymbol{S} := (\boldsymbol{Q}^\top \boldsymbol{Q} + \tau \boldsymbol{I})$, $\boldsymbol{s} := (\boldsymbol{Q}^\top \boldsymbol{q}_1 + \tau \boldsymbol{q}_2)$, and denote the largest eigenvalues and largest diagonal element of $\boldsymbol{S}$ by $\sigma_{\max}$ and $\delta_{\max}$, respectively. Hence, the two problems can be equivalently rewritten as

$$(\mathrm{P}) \quad \min_{\boldsymbol{u} \in \{0,1\}^n} \boldsymbol{u}^\top \boldsymbol{S} \boldsymbol{u} - 2\boldsymbol{s}^\top \boldsymbol{u}, \qquad (\mathrm{P_t}) \quad \min_{\boldsymbol{u} \in [0,1]^n} \boldsymbol{u}^\top (\boldsymbol{S} - t\boldsymbol{I})\boldsymbol{u} - (2\boldsymbol{s} - t\boldsymbol{e})^\top \boldsymbol{u}.$$

B.1 Proof of Proposition 1

Proof. The results are primarily based on Bertsekas [2015, Proposition 1.3.4]: the minima of a strictly concave function cannot be in the relative interior of the feasible set.

We first show that if $t_0 \geq \sigma_{\max}$, then the two problems are equivalent [Le Thi and Ding Tao, 2001, Theorem 1]. Specifically, since $\boldsymbol{S} - t\boldsymbol{I}$ is negative definite, problem ($\mathrm{P_t}$) is strictly concave. Therefore, the minima should be in the vertex set of the feasible domain, which is consistent with problem (P).

We can further generalize this result to the case $t_0 \geq \delta_{\max}$[Hansen et al., 1993, Proposition 1]. In this case, considering the i-th component of $\boldsymbol{u}$, its second-order derivative in problem ($\mathrm{P_t}$) is $2(\boldsymbol{S}_{ii} - t)$. Similarly, the strict concavity of $\boldsymbol{u}_i$ ensures a binary solution, indicating the equivalence of problems (P) and ($\mathrm{P_t}$). $\square$

B.2 Proof of Proposition 2

Proof. The Karush–Kuhn–Tucker (KKT) conditions of the problem ($\mathrm{P_t}$) are as follows.

$$\begin{aligned}&[2\boldsymbol{S}\boldsymbol{u} - 2t\boldsymbol{u} - 2\boldsymbol{s} + t\boldsymbol{e}]_i - \boldsymbol{\alpha}_i + \boldsymbol{\beta}_i = \boldsymbol{0};\\ &\boldsymbol{u}_i \in [0,1]^n; \quad \boldsymbol{\alpha}_i \geq \boldsymbol{0}, \boldsymbol{\beta}_i \geq \boldsymbol{0};\\ &\boldsymbol{\alpha}_i \boldsymbol{u}_i = 0, \quad \boldsymbol{\beta}_i(\boldsymbol{u}_i - 1) = 0; \quad i = 1, \ldots, n.\end{aligned}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are multiplier vector. For $\boldsymbol{u} \in \{0,1\}^n$, the KKT condition is equivalent to

$$\boldsymbol{\alpha}_i = [2\boldsymbol{S}\boldsymbol{u} - 2t\boldsymbol{u} - 2\boldsymbol{s} + t\boldsymbol{e}]_i(1 - \boldsymbol{u}_i) \geq 0, \quad \boldsymbol{\beta}_i = [2\boldsymbol{S}\boldsymbol{u} - 2t\boldsymbol{u} - 2\boldsymbol{s} + t\boldsymbol{e}]_i \boldsymbol{u}_i \leq 0.$$

By using $(1 - 2\boldsymbol{u}_i) \in \{-1, 1\}$, we can further combine the above two inequalities, and obtain

$$2[\boldsymbol{S}\boldsymbol{u} - \boldsymbol{s}]_i(1 - 2\boldsymbol{u}_i) + t \geq 0, \quad i = 1, \ldots, n.$$

On the other hand, if $2[\boldsymbol{S}\boldsymbol{u} - \boldsymbol{s}]_i(1 - 2\boldsymbol{u}_i) + t \geq 0$ holds for each $i = 1, \ldots, n$, it is easy to check that $\boldsymbol{\alpha} \geq 0$ and $\boldsymbol{\beta} \geq 0$, which proves the first part of the proposition.

The proof of the second part is a direct result of Beck and Teboulle [2000, Theorem 2.4]. To be specific, if $\boldsymbol{u}$ achieves a global minimum of (P), then $q(\boldsymbol{u}) \leq q(\boldsymbol{u'})$ for any $\boldsymbol{u}' \in \{0,1\}^n$. Hence, we only flip the i-th value of $\boldsymbol{u}$, i.e., considering $\boldsymbol{u}_i$ and $\boldsymbol{u}'_i = 1 - \boldsymbol{u}_i$, and it holds that

$$\begin{aligned}\boldsymbol{u}^\top \boldsymbol{S}\boldsymbol{u} - 2\boldsymbol{s}^\top \boldsymbol{u} &\leq (\boldsymbol{u}')^\top \boldsymbol{S}\boldsymbol{u}' - 2\boldsymbol{s}^\top \boldsymbol{u}'\\ &= (\boldsymbol{u}^\top \boldsymbol{S}\boldsymbol{u} - 2\boldsymbol{s}^\top \boldsymbol{u}) + 2[\boldsymbol{S}\boldsymbol{u} - \boldsymbol{s}]_i(1 - 2\boldsymbol{u}_i) + \boldsymbol{S}_{ii}.\end{aligned}$$

Rearranging the inequality, we obtain

$$2[\boldsymbol{S}\boldsymbol{u} - \boldsymbol{s}]_i(1 - 2\boldsymbol{u}_i) \geq -\boldsymbol{S}_{ii}, \quad i = 1, \ldots, n,$$

which completes the proof. □

C Proof of Theorem 1

Proof. **Notations.** We use $\|\cdot\|$ to denote the ℓ_2 norm for vectors and Frobenius norm for matrices. We define

$$\varphi(\boldsymbol{\phi}, \boldsymbol{\theta}, \mathbf{Z}, \mathbf{Y}) := \|\mathbf{Z}\boldsymbol{w}_u + \mathbf{Y}\boldsymbol{w}_v - \boldsymbol{b}\|^2 + \alpha\|(\mathbf{Z}, \mathbf{Y}) - (f_{\boldsymbol{\theta}}(\mathbf{X}), \mathbf{Y})\|^2 + \lambda\|\boldsymbol{w} - \boldsymbol{w}^0\|^2.$$

For the loss functions of logic programming and network training, we assume $\ell_1(\boldsymbol{\theta})$ and $\ell_2(\boldsymbol{\phi})$ to be $\mu_{\boldsymbol{\theta}}$ and $\mu_{\boldsymbol{\phi}}$ smooth, respectively. For ease of presentation, we define $\Delta^k = f_{\boldsymbol{\theta}^k}(\mathbf{X})\boldsymbol{w}_u^k + \mathbf{Y}\boldsymbol{w}_v^k - \boldsymbol{b}$, and let $c_{\max}$ be the upper bound of $\|\Delta^k\|$. Furthermore, by using the Woodbury identity formula, we can compute

$$\begin{aligned}(\mathbf{Z}^k; \mathbf{Y}^k) &= \arg\min_{(\mathbf{Z}, \mathbf{Y})} \|\mathbf{Z}\boldsymbol{w}_u^k + \mathbf{Y}\boldsymbol{w}_v^k - \boldsymbol{b}\|^2 + \alpha\|(\mathbf{Z}, \mathbf{Y}) - (f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y})\|^2 + \lambda\|\boldsymbol{w} - \boldsymbol{w}^0\|^2\\ &= (f_{\boldsymbol{\theta}^k}(\mathbf{X}); \mathbf{Y}) - \beta^k \Delta^k (\boldsymbol{w}^k)^\top, \quad \text{where} \quad \beta^k = \frac{1}{\alpha + \|\boldsymbol{w}^k\|^2}.\end{aligned}$$

Let $\rho^k := (\alpha\beta^k)$, we have

$$\begin{aligned}\varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y}) - \varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, \mathbf{Z}^k, \mathbf{Y}^k) &= (1 - ((\alpha\beta^k)^2 + (1 - \alpha\beta^k)^2)\|\Delta^k\|^2\\ &= 2\rho^k(1 - \rho^k)\|\Delta^k\|^2.\end{aligned}$$

Update of $\boldsymbol{\phi}$. We consider the single rule case (multiple rules can be directly decomposed), i.e., $\boldsymbol{\phi} = (\boldsymbol{w}, \boldsymbol{b})$ and $\boldsymbol{b} = (b; \ldots; b)$. The update of $\boldsymbol{\phi}$ is conducted on the loss function

$$\ell_2^k(\boldsymbol{w}, \boldsymbol{b}) = \varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y}) = \|f_{\boldsymbol{\theta}^k}(\mathbf{X})\boldsymbol{w}_u + \mathbf{Y}\boldsymbol{w}_v - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{w} - \boldsymbol{w}^0\|^2.$$

The smallest and the largest eigenvalues of $(f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y})^\top (f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y}) + \lambda \boldsymbol{I}$ are denoted by $\sigma_{\min}$ and $\sigma_{\max}$, respectively.

The PPA method updates $\boldsymbol{w}$ by

$$\boldsymbol{w}^{k+1} = \arg\min_{\boldsymbol{w}} \ell_2^k(\boldsymbol{w}, \boldsymbol{b}) + \frac{1}{\gamma}\|\boldsymbol{w} - \boldsymbol{w}^k\|^2,$$

which can be reduced to

$$\boldsymbol{w}^{k+1} - \boldsymbol{w}^k = -\boldsymbol{M}^k \boldsymbol{\delta}^k, \quad \boldsymbol{\delta}^k = (f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y})^\top \Delta = \nabla_{\boldsymbol{w}} \ell_2^k(\boldsymbol{w}, \boldsymbol{b}),$$

where

$$\boldsymbol{M}^k = \big((f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y})^\top (f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y}) + \lambda \boldsymbol{I} + \frac{1}{\gamma}\boldsymbol{I}\big)^{-1}.$$

The $(2/\gamma)$-strongly convexity of the proximal term implies the Polyak-Łojasiewicz (PL) inequality, which derives that

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, \mathbf{Z}^k, \mathbf{Y}^k) = \varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y}) - 2\rho^k(1-\rho^k)\|\Delta\|^2 \\
&\quad = \ell_2^k(\boldsymbol{w}^k, \boldsymbol{b}) - 2\rho^k(1-\rho^k)\|\Delta^k\|^2 \geq \ell_2^k(\boldsymbol{w}^{k+1}, \boldsymbol{b}) - 2\rho^k(1-\rho^k)\|\Delta\|^2 + \frac{2}{\gamma}\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^k\|^2.
\end{aligned}$$

Plugging $\boldsymbol{w}^{k+1} - \boldsymbol{w}^k = -\boldsymbol{M}^k \boldsymbol{\delta}^k$ into the inequality, we have

$$\begin{aligned}
\varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, \mathbf{Z}^k, \mathbf{Y}^k) &\geq \ell_2^k(\boldsymbol{w}^{k+1}, \boldsymbol{b}) + \frac{2}{\gamma}(\boldsymbol{\delta}^k)^\top (\boldsymbol{M}^k)^2 \boldsymbol{\delta}^k - 2\rho^k(1-\rho^k)\|\Delta^k\|^2 \\
&\geq \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^k, \mathbf{Z}^{k+\frac{1}{2}}, \mathbf{Y}^{k+\frac{1}{2}}) + \frac{2}{\gamma}(\boldsymbol{\delta}^k)^\top (\boldsymbol{M}^k)^2 \boldsymbol{\delta}^k - 2\rho^k(1-\rho^k)\|\Delta^k\|^2,
\end{aligned}$$

where

$$\begin{aligned}
(\mathbf{Z}^{k+\frac{1}{2}}; \mathbf{Y}^{k+\frac{1}{2}}) &= \arg\min\nolimits_{(\bar{\mathbf{Z}}, \bar{\mathbf{Y}})} \|\bar{\mathbf{Z}} \boldsymbol{w}_u^{k+1} + \bar{\mathbf{Y}} \boldsymbol{w}_v^{k+1} - \boldsymbol{b}\|^2 + \alpha\|(\bar{\mathbf{Z}}, \bar{\mathbf{Y}}) - (f_{\boldsymbol{\theta}^k}(\mathbf{X}), \mathbf{Y})\|^2 \\
&= (f_{\boldsymbol{\theta}^k}(\mathbf{X}); \mathbf{Y}) - \beta^{k+\frac{1}{2}} \Delta^{k+\frac{1}{2}} (\boldsymbol{w}^{k+1})^\top, \quad \text{where} \quad \beta^{k+\frac{1}{2}} = \frac{1}{\alpha + \|\boldsymbol{w}^{k+1}\|^2}.
\end{aligned}$$

Note that $(\boldsymbol{M}^k)^2$ has the smallest eigenvalue $\gamma^2/(1+\gamma\sigma_{\max})^2$, and thus we have

$$\varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, \mathbf{Z}^k, \mathbf{Y}^k) \geq \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^k, \mathbf{Z}^{k+\frac{1}{2}}, \mathbf{Y}^{k+\frac{1}{2}}) + \frac{2\gamma}{(1+\gamma\sigma_{\max})^2}\|\nabla_{\boldsymbol{\phi}} \ell_2(\boldsymbol{w}, \boldsymbol{b})\|^2 - 2\rho^k(1-\rho^k)c_{\max}.$$

<u>**Update of $\boldsymbol{\theta}$**</u>. The update of $\boldsymbol{\theta}$ is conducted on the loss function

$$\ell_1^k(\boldsymbol{\theta}) = \|\mathbf{Z}^{k+\frac{1}{2}} - f_{\boldsymbol{\theta}}(\mathbf{X})\|^2.$$

By using $\mu_{\boldsymbol{\theta}}$-smooth of ℓ_1^k, we obtain that

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^k, \mathbf{Z}^{k+\frac{1}{2}}, \mathbf{Y}^{k+\frac{1}{2}}) - \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^{k+1}, \mathbf{Z}^{k+\frac{1}{2}}, \mathbf{Y}^{k+\frac{1}{2}}) = \ell_1^k(\boldsymbol{\theta}^{k+1}) - \ell_1^k(\boldsymbol{\theta}^k) \\
&\qquad \geq -\langle \nabla_{\boldsymbol{\theta}} \ell_1^k(\boldsymbol{\theta}^k), \boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k \rangle - \frac{\mu_\theta}{2}\|\boldsymbol{\theta}^{k+1} - \boldsymbol{\theta}^k\|^2 \geq \frac{1}{2}\eta\|\nabla_{\boldsymbol{\theta}} \ell_1^k(\boldsymbol{\theta}^k)\|^2.
\end{aligned}$$

Letting $\mathbf{Z}^{k+1} = \arg\min_{\mathbf{Z}} \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^{k+1}, \mathbf{Z})$, we conclude

$$\varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^k, \mathbf{Z}^{k+\frac{1}{2}}, \mathbf{Y}^{k+\frac{1}{2}}) \geq \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^{k+1}, \mathbf{Z}^{k+1}, \mathbf{Y}^{k+1}) + \frac{1}{2}\eta\|\nabla_{\boldsymbol{\theta}} \ell_1^k(\boldsymbol{\theta}^k)\|^2.$$

<u>**Convergent result.**</u> By combining the update of $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, we have

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^k, \boldsymbol{\theta}^k, \mathbf{Z}^k, \mathbf{Y}^k) - \varphi(\boldsymbol{\phi}^{k+1}, \boldsymbol{\theta}^{k+1}, \mathbf{Z}^{k+1}, \mathbf{Y}^{k+1}) \\
&\qquad \geq \frac{1}{2}\eta\|\nabla_{\boldsymbol{\theta}} \ell_1^k(\boldsymbol{\theta}^k)\|^2 + \frac{2\gamma}{(1+\gamma\sigma_{\max})^2}\|\nabla_{\boldsymbol{\phi}} \ell_2(\boldsymbol{\phi}^k)\|^2 - 2\rho^k(1-\rho^k)c_{\max}.
\end{aligned}$$

Taking a telescopic sum over k, we obtain

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^0, \boldsymbol{\theta}^0, \mathbf{Z}^0, \mathbf{Y}^0) - \varphi(\boldsymbol{\phi}^K, \boldsymbol{\theta}^K, \mathbf{Z}^K, \mathbf{Y}^K) \\
&\qquad \geq \sum_{i=1}^{K} \frac{1}{2}\eta\|\nabla_{\boldsymbol{\theta}} \ell_1^k(\boldsymbol{\theta}^k)\|^2 + \frac{2\gamma}{(1+\gamma\sigma_{\max})^2}\|\nabla_{\boldsymbol{\phi}} \ell_2(\boldsymbol{\phi}^k)\|^2 - 2\rho^k(1-\rho^k)c_{\max}.
\end{aligned}$$

Since $\rho^k(1-\rho^k) \leq \kappa_\rho/(K+1)^2$, we have

$$\mathbb{E}[\|\nabla_{\boldsymbol{\theta}} \ell_2^k(\boldsymbol{\theta}^k)\|^2] \leq \frac{2}{(K+1)\eta}\big((\varphi(\boldsymbol{\phi}^0, \boldsymbol{\theta}^0, \mathbf{Z}^0, \mathbf{Y}^0) - \min\varphi) + 2\kappa c_{\max}\big),$$

and

$$\mathbb{E}[\|\nabla_{\boldsymbol{\phi}}\ell_2(\boldsymbol{\phi}^k)\|^2] \leq \frac{(1+\gamma\sigma_{\max})^2}{2(K+1)}\big((\varphi(\boldsymbol{\phi}^0,\boldsymbol{\theta}^0,\mathbf{Z}^0,\mathbf{Y}^0)-\min\varphi)+2\kappa c_{\max}\big).$$

<u>Stochastic version.</u> We first introduce an additional assumption: the gradient estimate is unbiased and has bounded variance [Bottou et al., 2018, Sec. 4], i.e.,

$$\mathbb{E}_\xi[\tilde{\nabla}_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)] = \nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k), \quad \mathbb{E}_\xi[\tilde{\nabla}_{\boldsymbol{\theta}}\ell_2^k(\boldsymbol{\phi}^k)] = \nabla_{\boldsymbol{\theta}}\ell_2^k(\boldsymbol{\phi}^k),$$

and

$$\mathbb{V}_\xi[\tilde{\nabla}_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)] \leq \zeta+\zeta_v\|\nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2, \quad \mathbb{V}_\xi[\tilde{\nabla}_{\boldsymbol{\phi}}\ell_1^k(\boldsymbol{\theta}^k)] \leq \zeta+\zeta_v\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2.$$

This assumption derives the following inequalities hold for $\zeta_g=\zeta_v+1$:

$$\mathbb{E}_\xi[\|\tilde{\nabla}_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2] \leq \zeta+\zeta_g\|\nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2, \quad \mathbb{E}_\xi[\|\tilde{\nabla}_{\boldsymbol{\theta}}\ell_2^k(\boldsymbol{\phi}^k)\|^2] \leq \zeta+\zeta_g\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2,$$

For the update of $\boldsymbol{\theta}$, we have

$$\varphi(\boldsymbol{\phi}^{k+1},\boldsymbol{\theta}^k,\mathbf{Z}^{k+\frac{1}{2}},\mathbf{Y}^{k+\frac{1}{2}})-\mathbb{E}_\xi[\varphi(\boldsymbol{\phi}^{k+1},\boldsymbol{\theta}^{k+1},\mathbf{Z}^{k+1},\mathbf{Y}^{k+1})] \geq \frac{\eta_k}{2}\|\nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2-\frac{\eta_k^2\mu_{\boldsymbol{\theta}}}{2}\zeta.$$

For the update of $\boldsymbol{\phi}$, using the $\mu_{\boldsymbol{\phi}}$-smooth, and taking the total expectation:

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^k,\boldsymbol{\theta}^k,\mathbf{Z}^k,\mathbf{Y}^k)-\mathbb{E}_\xi[\varphi(\boldsymbol{\phi}^{k+1},\boldsymbol{\theta}^k,\mathbf{Z}^{k+\frac{1}{2}},\mathbf{Y}^{k+\frac{1}{2}})]+2\rho^k(1-\rho^k)\|\Delta^k\|^2\\
&\qquad\geq (\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k))^\top\boldsymbol{M}^k(\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k))-\frac{\mu_{\boldsymbol{\phi}}}{2}\mathbb{E}_\xi[\|\tilde{\boldsymbol{M}}^k\tilde{\nabla}_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2]\\
&\qquad\geq \frac{1}{\epsilon^k+\sigma_{\max}}\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2-\frac{\mu_{\boldsymbol{\phi}}}{2(\epsilon^k+\sigma_{\min})^2}(\zeta+\zeta_g\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2),
\end{aligned}$$

where we define $\epsilon^k = 1/\gamma^k$ for simplicity. Now, let γ be sufficiently small (that is, satisfying $(\epsilon^k+\sigma_{\min})^2 \geq \mu_{\boldsymbol{\phi}}(\epsilon^k+\sigma_{\max})$), we obtain

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^k,\boldsymbol{\theta}^k,\mathbf{Z}^k,\mathbf{Y}^k)-\mathbb{E}_\xi[\varphi(\boldsymbol{\phi}^{k+1},\boldsymbol{\theta}^k,\mathbf{Z}^{k+\frac{1}{2}},\mathbf{Y}^{k+\frac{1}{2}})]+2\rho^k(1-\rho^k)\|\Delta^k\|^2\\
&\qquad\qquad\geq \frac{1}{2(\epsilon^k+\sigma_{\max})}\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2-\frac{\mu_{\boldsymbol{\phi}}}{2(\epsilon^k+\sigma_{\min})^2}\zeta.
\end{aligned}$$

Putting the updates of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ together, we have

$$\begin{aligned}
&\varphi(\boldsymbol{\phi}^k,\boldsymbol{\theta}^k,\mathbf{Z}^k,\mathbf{Y}^k)-\mathbb{E}_\xi[\varphi(\boldsymbol{\phi}^{k+1},\boldsymbol{\theta}^{k+1},\mathbf{Z}^{k+1},\mathbf{Y}^{k+1})]+2\rho^k(1-\rho^k)\|\Delta^k\|^2\\
&\quad\geq \frac{1}{2}\eta_k\|\nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2-\frac{1}{2}\eta_k^2\mu_{\boldsymbol{\theta}}\zeta+\frac{1}{2(\epsilon^k+\sigma_{\max})}\|\nabla_{\boldsymbol{\phi}}\ell_2^k(\boldsymbol{\phi}^k)\|^2-\frac{\mu_{\boldsymbol{\phi}}}{2(\epsilon^k+\sigma_{\min})^2}\zeta.
\end{aligned}$$

Now, setting $\eta^k \leq \kappa_{\boldsymbol{\theta}}/\sqrt{K+1}$ and $\gamma^k \leq \kappa_{\boldsymbol{\phi}}/\sqrt{K+1}$, we can conclude

$$\mathbb{E}[\|\nabla_{\boldsymbol{\theta}}\ell_1^k(\boldsymbol{\theta}^k)\|^2] = \mathcal{O}(\frac{1}{\sqrt{K+1}}), \quad \mathbb{E}[\|\nabla_{\boldsymbol{\phi}}\ell_2(\boldsymbol{\phi}^k)\|^2] = \mathcal{O}(\frac{1}{\sqrt{K+1}}). \qquad \square$$

D Proof of Theorem 2

Proof. We consider the following problem,

$$(\mathrm{P}_\xi) \quad \min_{\boldsymbol{u}\in\{0,1\}^n} q_\xi(\boldsymbol{u}) := \boldsymbol{u}^\top(\boldsymbol{S}+\lambda\boldsymbol{I})\boldsymbol{u}-2(\boldsymbol{s}+\lambda\boldsymbol{\xi})^\top\boldsymbol{u}.$$

For given $t\geq 0$, the corresponding stationary points of (P_ξ) satisfy

$$2[\boldsymbol{S}\boldsymbol{u}-\boldsymbol{s}]_i(1-2\boldsymbol{u}_i)+2\lambda(\boldsymbol{u}_i-\boldsymbol{\xi}_i)(1-2\boldsymbol{u}_i)+t\geq 0, i=1,\ldots,n.$$

Note that

$$(\boldsymbol{u}_i-\boldsymbol{\xi}_i)(1-2\boldsymbol{u}_i) = \begin{cases} -\boldsymbol{\xi}_i & \text{if} \quad \boldsymbol{u}_i=0;\\ \boldsymbol{\xi}_i-1 & \text{if} \quad \boldsymbol{u}_i=1.\end{cases}$$

For given $\boldsymbol{u}\in\{0,1\}^n$, we denote $\varrho_i = 2[\boldsymbol{S}\boldsymbol{u}-\boldsymbol{s}]_i(1-2\boldsymbol{u}_i)$. Then, the probability that (P_ξ) has the stationary point $\boldsymbol{u}$ can be computed as

$$\Pr(\boldsymbol{u}) = \prod_{i=1}^n \Pr(\varrho_i+2\lambda(\boldsymbol{u}_i-\boldsymbol{\xi}_i)(1-2\boldsymbol{u}_i)+t\geq 0),$$

where

$$\Pr(2\lambda(\boldsymbol{u}_i - \boldsymbol{\xi}_i)(1 - 2\boldsymbol{u}_i) + \varrho_i + t \geq 0) = \min(\frac{1}{2\lambda}(t + \varrho_i), 1).$$

Hence, for given two different $\boldsymbol{u}_1^{(0)}$ and $\boldsymbol{u}_2^{(0)}$, the probability that the corresponding rules can converge to the same result $\boldsymbol{u}$ satisfying

$$\Pr(\boldsymbol{u}_1 = \boldsymbol{u}, \boldsymbol{u}_2 = \boldsymbol{u}) \leq \Pr(\boldsymbol{u})^2 = \prod_{i=1}^{n} \min(\frac{1}{2\lambda}(t + \varrho_i), 1)^2. \qquad \square$$

E Trust Region Method

Figure 3 illustrates the key concept of the trust region method. For simplicity, centre points $\boldsymbol{w}_1(\mathbf{0}), \ldots, \boldsymbol{w}_4(\mathbf{0})$ of the trust region are also set as the initial points of stochastic gradient descent. Stochastic gradient descent is implicitly biased to least norm solutions and finally converges to point $(0, 1)$ by enforcing the Boolean constraints. The trust region penalty encourages the stochastic gradient descent to converge to different optimal solutions in different trust regions.

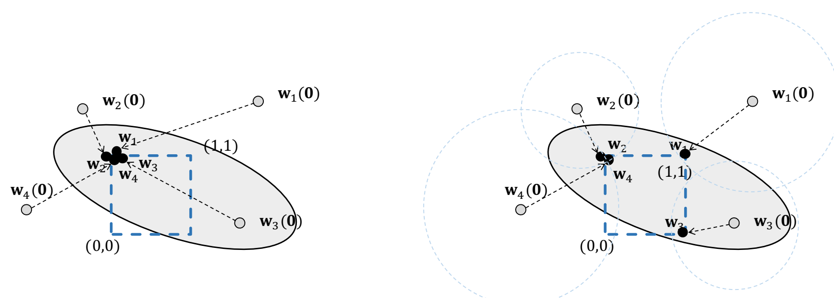


Figure 1: Avoid degeneracy by trust region method. In logical constraint learning, the imposition of the Boolean constraints and the implicit bias of the stochastic gradient descent cause $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_4$ to converge to the same result (left figure), while the trust region constraints guarantee that they can sufficiently indicate different rules (right figure).

F Experiment Details

Computing configuration. We implemented our approach via the PyTorch DL framework. The experiments were conducted on a GPU server with two Intel Xeon Gold 5118 CPU@2.30GHz, 400GB RAM, and 9 GeForce RTX 2080 Ti GPUs. The server ran Ubuntu 16.04 with GNU/Linux kernel 4.4.0.

Hyperparameter tuning. Some hyperparameters are introduced in our framework. In Table 3 we summarize the (hyper-)parameters, together with their corresponding initialization or update strategies. Most of these hyperparameters are quite stable and thus only need to be fixed to a constant or set by standard strategies. We only discuss the setting of $\boldsymbol{b}_{\min}$, $\boldsymbol{b}_{\max}$ and $\boldsymbol{b}$. We recommend $\boldsymbol{b}$ to be tuned manually rather than set by PPA update, and one can gradually increase $\boldsymbol{b}$ from 1 to $n-1$ (n is the number of involved logical variables), and collect all logical constraints as candidate constraints. For $\boldsymbol{b}_{\min}$ and $\boldsymbol{b}_{\max}$, due to the prediction error, it is unreasonable to set $\boldsymbol{b}_{\min}$ and $\boldsymbol{b}_{\max}$ that ensure all examples to satisfy the logical constraint. An alternative method is to set a threshold (e.g. $k\%$) on the training (or validation) set, and the constraint is only required to be satisfied by at least $k\%$ examples.

G Additional Experiment Results

G.1 Chained XOR

The chained XOR, also known as the parity function, is a basic logical function, yet it has proven challenging for neural networks to learn it explicitly [Shalev-Shwartz et al., 2017, Wang et al., 2019] To be specific, given a sequence of length L, the parity function outputs 1 if there are an odd number

Table 1: The list of (hyper-)parameters and their initialization or update strategies.

Param.	Description	Setting
$\boldsymbol{\theta}$	Neural network parameters	Updated by stochastic gradient descent
$\boldsymbol{W}$	Matrix of logical constraints	Updated by stochastic PPA
$\boldsymbol{b}$	Bias term of logical constraints	Pre-set or Updated by stochastic PPA
$\boldsymbol{b}_{\min}/\boldsymbol{b}_{\max}$	Lower/Upper bound of logical constraints	Estimated by training set
α	Trade-off weight in symbol grounding	Fixed to $\alpha = 1.0$
λ	Weight of trust region penalty	Fixed to $\lambda = 0.1$
t_1/t_2	Weight of DC penalty	Increased per epoch
η	Learning rate of network training	Adam schedule
γ	Step size of constraint learning	Adaptively set ($\gamma = 0.001$ by default)

of 1's in the sequence, and 0 otherwise. The goal of the Chained XOR task is to learn this parity function with fixed L. Note that this task does not involve any perception task.

We compare our method with SATNet and L1R32H4. In this task, SATNet uses an implicit but strong background knowledge that the task can be decomposed into L single XOR tasks. Neither L1R32H4 nor our method uses such knowledge. For L1R32H4, we adapt the embedding layer to this task and fix any other configuration. Regarding our method, we introduce $L-1$ auxiliary variables.[1]

It is worth noting that these auxiliary variables essentially serve as a form of symbol grounding. Elaborately, the learned logical constraints by our method can be formulated as follows,

$$\boldsymbol{w}_1\mathbf{x}_1 + \cdots + \boldsymbol{w}_L\mathbf{x}_L + \boldsymbol{w}_{L+1}\mathbf{z}_1 + \cdots + \boldsymbol{w}_{2L-1}\mathbf{z}_{L-1} = b,$$

where $\boldsymbol{w}_i \in \mathcal{B}, i = 1, \ldots, 2L-1$, $\mathbf{x}_i \in \mathcal{B}, i = 1, \ldots, L$ and $\mathbf{z}_i \in \mathcal{B}, i = 1, \ldots, L$. The auxiliary variables $\mathbf{z}_i, i = 1, \ldots, L$ have different truth assignments for different examples, indicting *how* the logical constraint is satisfied by the given input. Now, combining the symbol grounding of auxiliary variables, we revise the optimization problem (1) of our framework as

$$\begin{aligned}
&\min_{(\boldsymbol{W},\boldsymbol{b})} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}[\|\boldsymbol{W}(\mathbf{x};\bar{\mathbf{z}};\mathbf{y}) - \boldsymbol{b}\|^2] + \lambda\|\boldsymbol{W} - \boldsymbol{W}^{(0)}\|^2,\\
&\quad \text{s.t.} \quad \bar{\mathbf{z}} = \arg\min_{\mathbf{z}\in\mathcal{Z}} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}[\|\boldsymbol{W}(\mathbf{x};\mathbf{z};\mathbf{y}) - \boldsymbol{b}\|^2], \quad \boldsymbol{W} \in \mathcal{B}^{m\times(u+v)}, \quad \boldsymbol{b} \in \mathcal{N}_+^m.
\end{aligned}$$

The symbol grounding is solely guided by logical constraints, as neural perception is not involved.

The experimental results are plotted in Figure 4. The results show that L1R32H4 is unable to learn such a simple reasoning pattern, while SATNet often fails to converge even with sufficient iterations, leading to unstable results. Our method consistently delivers full accuracy across all settings, thereby demonstrating superior performance and enhanced scalability in comparison to existing state-of-the-art methods. To further exemplify the efficacy of our method, we formulate the learned constraints in the task of $L = 20$. Eliminating redundant constraints and replacing the auxiliary variables with logical disjunctions, the final learned constraint can be expressed as

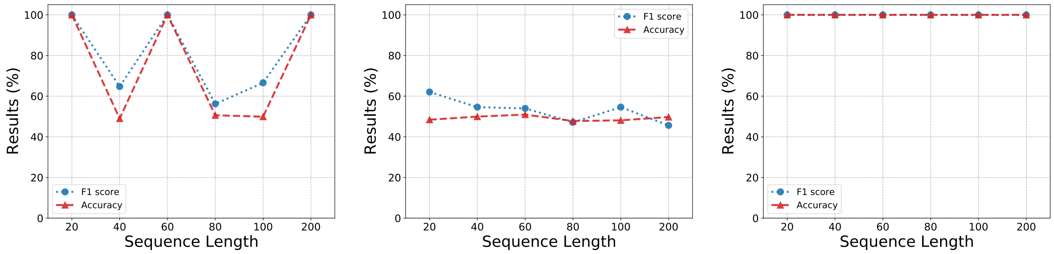
$$(\mathbf{x}_1 + \cdots + \mathbf{x}_{20} + y = 0) \vee (\mathbf{x}_1 + \cdots + \mathbf{x}_{20} + y = 2) \vee \cdots \vee (\mathbf{x}_1 + \cdots + \mathbf{x}_{20} + y = 20),$$

which shows that our method concludes with complete and precise logical constraints.

G.2 Nonograms

Nonograms is a logic puzzle with simple rules but challenging solutions. Given a grid of squares, the task of nonograms is to plot a binary image, i.e., filling each grid in black or marking it by X. The required numbers of black squares on that row (resp. column) are given beside (resp. above) each row (resp. column) of the grid. Figure 5 gives a simple example.

[1]Note that the number of auxiliary variables should not exceed the number of logical variables. If so, the logical constraints trivially converge to any result.



(a) SATNet (b) L1R32H4 (c) Ours

Figure 2: Results (%) of chained XOR task, including accuracy and F_1 score (of class 0). The sequence length ranges from 20 to 200, showing that our method stably outperforms competitors.

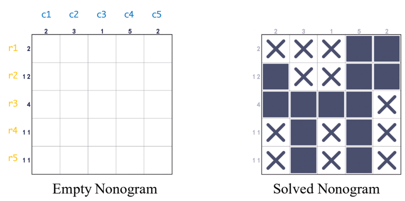


Figure 3: An example of nonograms.

Data Size	L1R32H4	Ours
1000	14.4	**100.0**
5000	62.0	**100.0**
9000	81.2	**100.0**

Table 2: Accuracy (%) of the nonograms task.

In contrast to the supervised setting used in Yang et al. [2023], we evaluate our method on a weakly supervised learning setting. Elaborately, instead of the fully solved board, only partial solutions (i.e., only one row or one column) are observed. Note that this supervision is enough to solve the nonograms, because the only logical rule to be learned is that the different black squares (in each row or column) should not be connected.

For our method, we do not introduce a neural network in this task, and only aim to learn the logical constraints. We carry out the experiments on 7×7 nonograms, with training data sizes ranging from 1,000 to 9,000. The results are given in Table 4, showing the efficacy of our logical constraint learning. Compared to the L1R32H4 method, whose effectiveness highly depends on the training data size, our method works well even with extremely limited data.

G.3 Visual SudoKu Solving

In the visual SudoKu task, it is worth noting that the computation of $\mathbf{z}$ cannot be conducted by batch processing. This is because the index of $\mathbf{y}$ varies for each data point. For instance, in different SudoKu games, the cells to be filled are different, and thus the symbol $\mathbf{z}$ has to be computed in a point-wise way. To solve this issue, we introduce an auxiliary $\bar{\mathbf{y}}$ to approximate the output symbol $\mathbf{y}$:

$$(\bar{\mathbf{z}}, \bar{\mathbf{y}}) = \arg\min_{\bar{\mathbf{z}} \in \mathcal{Z}, \bar{\mathbf{y}} \in \mathcal{Y}} \|\boldsymbol{W}(\bar{\mathbf{z}}; \bar{\mathbf{y}}) - \boldsymbol{b}\|^2 + \alpha \|(\bar{\mathbf{z}}; \bar{\mathbf{y}}) - (f_{\boldsymbol{\theta}}(\mathbf{x}); \mathbf{y})\|^2.$$

On the SATNet dataset, we use the recurrent transformer as the perception model [Yang et al., 2023], because we observe that the recurrent transformer can significantly improve the perception accuracy, and even outperforms the state-of-the-art of MNIST digit recognition model. However, we find that its performance degrades on the more difficult dataset RRN, and thus we still use a standard convolutional neural network model as the perception model for this dataset.

We include detailed results of board and cell accuracy in Table 5. It can be observed that our method is consistently superior to the existing methods, and significantly outperforms the current state-of-the-art method L1R32H4 on the RRN dataset (total board accuracy improvement exceeds 20%). Also note that the solving accuracy of our method always performs the best, illustrating the efficacy of our logical constraint learning.

Next, we exchange the evaluation dataset, namely, using the RRN dataset to evaluate the model trained on the SATNet dataset, and vice versa. The results are presented in Table 6. The accurate logical constraints and exact logical reasoning engine guarantee the best performance of our method on transfer tasks. Specifically, the performance of L1R32H4 drops significantly when transfer the SATNet trained model to RNN dataset our method beats the alternative methods on both transfer tasks.

Table 3: Detailed cell and board accuracy (%) of **original** visual Sudoku task.

Method	SATNet dataset			RRN dataset		
	Perception board acc.	Solving board acc.	Total board acc.	Perception board acc.	Solving board acc.	Total board acc.
RRN	0.0	0.0	0.0	0.0	0.0	0.0
SATNet	0.0	0.0	0.0	0.0	0.0	0.0
SATNet*	72.7	75.9	67.3	75.7	0.1	0.1
L1R32H4	94.1	91.0	90.5	87.7	65.8	65.7
NTR	87.4	0.0	0.0	91.4	3.9	3.9
NDC	79.9	0.0	0.0	88.0	0.0	0.01
Ours	**95.5**	**95.9**	**95.5**	**93.1**	**94.4**	**93.1**
	Perception cell acc.	Solving cell acc.	Total cell acc.	Perception cell acc.	Solving cell acc.	Total cell acc.
RRN	0.0	0.0	0.0	0.0	0.0	0.0
SATNet	0.0	0.0	0.0	0.0	0.0	0.0
SATNet*	99.1	98.6	98.8	75.7	59.7	72.0
L1R32H4	99.8	99.1	99.4	99.3	89.5	92.6
NTR	99.7	60.1	77.8	99.7	38.5	57.3
NDC	99.4	10.8	50.4	99.5	10.9	38.7
Ours	**99.9**	**99.6**	**99.7**	**99.7**	**98.3**	**98.7**

G.4 Self-driving Path Planning

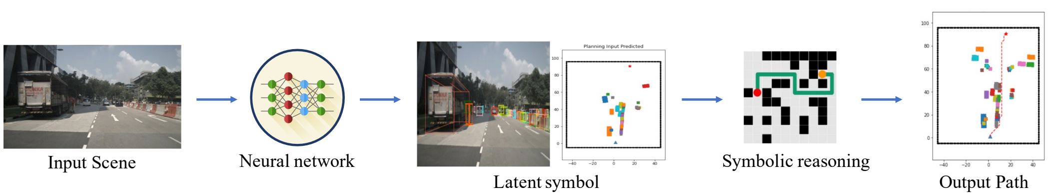


Figure 4: A neuro-symbolic system in self-driving tasks. The neural perception detects the obstacles from the image collected by the camera; the symbolic reasoning plans the driving path based on the obstacle map. The neuro-symbolic learning task is to build these two modules in an end-to-end way.

The goal of the self-driving path planning task is to train the neural network for object detection and to learn the logical constraints for path planning in an end-to-end way. As shown in Figure 6, we construct two maps and each contains 10×10 grids (binary variables). The neural perception detects the obstacles from the image $\mathbf{x}$ and locates it in the first map, which is essentially the symbol $\mathbf{z}$. Next, the logical reasoning computes the final path from the symbol $\mathbf{z}$ and tags it on the second map as the output $\mathbf{y}$.

As a detailed reference, we select some results of path planning generated by different methods and plot them in Figure 7. We find that some correct properties are learned by our method. For example, given the point $\mathbf{y}_{34}$ in the path, we have the following connectivity:

$$(\mathbf{y}_{34} = s) + (\mathbf{y}_{34} = e) + \mathrm{Adj}(\mathbf{y}_{34}) = 2,$$

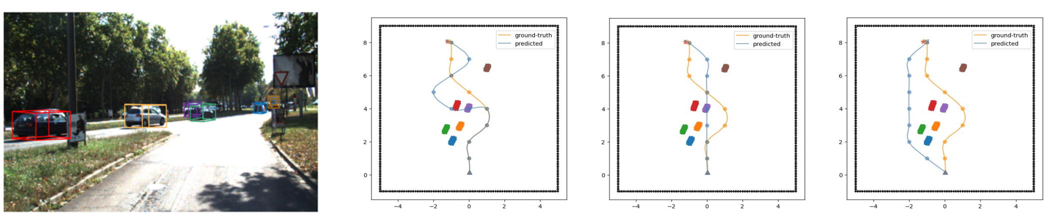
which means that the path point $\mathbf{y}_{34}$ should be connected by its adjacent points. In addition, some distinct constraints are also learned, for example,

$$\mathbf{y}_{32} + \mathbf{z}_{32} + \mathbf{z}_{11} + \mathbf{z}_{01} = 1.$$

Table 4: Detailed cell and board accuracy (%) of **transfer** visual Sudoku task.

Method	SATNet $\rightarrow$ RRN			RRN $\rightarrow$ SATNet		
	Perception board acc.	Solving board acc.	Total board acc.	Perception board acc.	Solving board acc.	Total board acc.
RRN	0.0	0.0	0.0	0.0	0.0	0.0
SATNet	0.0	0.0	0.0	0.0	0.0	0.0
SATNet*	80.8	1.4	1.4	0.0	0.0	0.0
L1R32H4	84.8	21.3	21.3	94.9	95.0	94.5
NTR	90.2	0.0	0.0	86.9	0.0	0.0
NDC	86.1	0.0	0.0	82.4	0.0	0.0
Ours	**93.9**	**95.2**	**93.9**	**95.2**	**95.3**	**95.2**
	Perception cell acc.	Solving cell acc.	Total cell acc.	Perception cell acc.	Solving cell acc.	Total cell acc.
RRN	0.0	0.0	0.0	0.0	0.0	0.0
SATNet	0.0	0.0	0.0	0.0	0.0	0.0
SATNet*	99.1	66.2	76.5	65.8	53.8	59.2
L1R32H4	99.3	89.5	92.6	99.7	99.6	99.7
NTR	99.6	37.1	56.3	99.6	62.4	79.0
NDC	99.4	11.0	38.7	99.5	11.3	50.7
Ours	**99.8**	**98.4**	**98.8**	**99.8**	**99.7**	99.7

In this constraint, $\mathbf{z}_{11}$ and $\mathbf{z}_{01}$ are two noise points, and they always take the value of 0. Therefore, it actually ensures that if $\mathbf{z}_{32}$ is an obstacle, then $\mathbf{y}_{32}$ should not be selected as a path point. However, it is still unknown whether our neuro-symbolic framework derives all the results as expected, because some of the learned constraints are too complex to be understood.



(a) Input scene (with labels) (b) ResNet method (c) RT method (d) Ours

Figure 5: Some results of neuro-symbolic learning methods in self-driving path planning task.

References

Inês Lynce and Joël Ouaknine. Sudoku as a sat problem. In *AI&M*, 2006.

Shu-Cherng Fang and Sarat Puthenpura. *Linear optimization and extensions: theory and algorithms*. Prentice-Hall, Inc., 1993.

Mislav Balunovic, Pavol Bielik, and Martin Vechev. Learning to solve smt formulas. *Advances in Neural Information Processing Systems*, 31, 2018.

Cristina Cornelio, Jan Stuehmer, Shell Xu Hu, and Timothy Hospedales. Learning where and when to reason in neuro-symbolic inference. In *The Eleventh International Conference on Learning Representations*, 2023.

Dimitri Bertsekas. *Convex optimization algorithms*. Athena Scientific, 2015.

Hoai An Le Thi and Pham Ding Tao. A continuous approch for globally solving linearly constrained quadratic. *Optimization*, 50(1-2):93–120, 2001.

Pierre Hansen, Brigitte Jaumard, MichèLe Ruiz, and Junjie Xiong. Global minimization of indefinite quadratic functions subject to box constraints. *Naval Research Logistics (NRL)*, 40(3):373–392, 1993.

Amir Beck and Marc Teboulle. Global optimality conditions for quadratic optimization problems with binary constraints. *SIAM journal on optimization*, 11(1):179–188, 2000.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *International Conference on Machine Learning*, pages 3067–3075. PMLR, 2017.

Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pages 6545–6554. PMLR, 2019.

Zhun Yang, Adam Ishay, and Joohyung Lee. Learning to solve constraint satisfaction problems with recurrent transformer. In *The Eleventh International Conference on Learning Representations*, 2023.