A Solver-Free Learning Framework for ILP

Rishabh Ranjan, Yatin Nandwani, Mausam, Parag Singla

BTP2 Mid-Term Presentation – 18 February 2022

Recap

Motivation

Task

Perception → Reasoning

Model

Neural → Symbolic

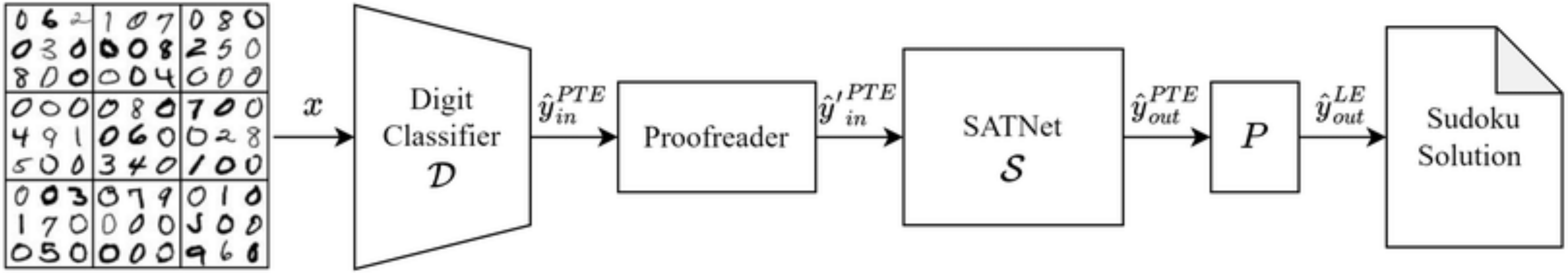


Figure from *Topan, Sever & Rolnick, David & Si, Xujie. (2021). Techniques for Symbol Grounding with SATNet.*

Integer Linear Programming

$x \rightarrow$ input

$A(x) \rightarrow$ constraint matrix

$b(x) \rightarrow$ bias vector

$c(x) \rightarrow$ cost vector

$y(A, b, c) \rightarrow$ solution vector / output

$n \rightarrow$ number of variables

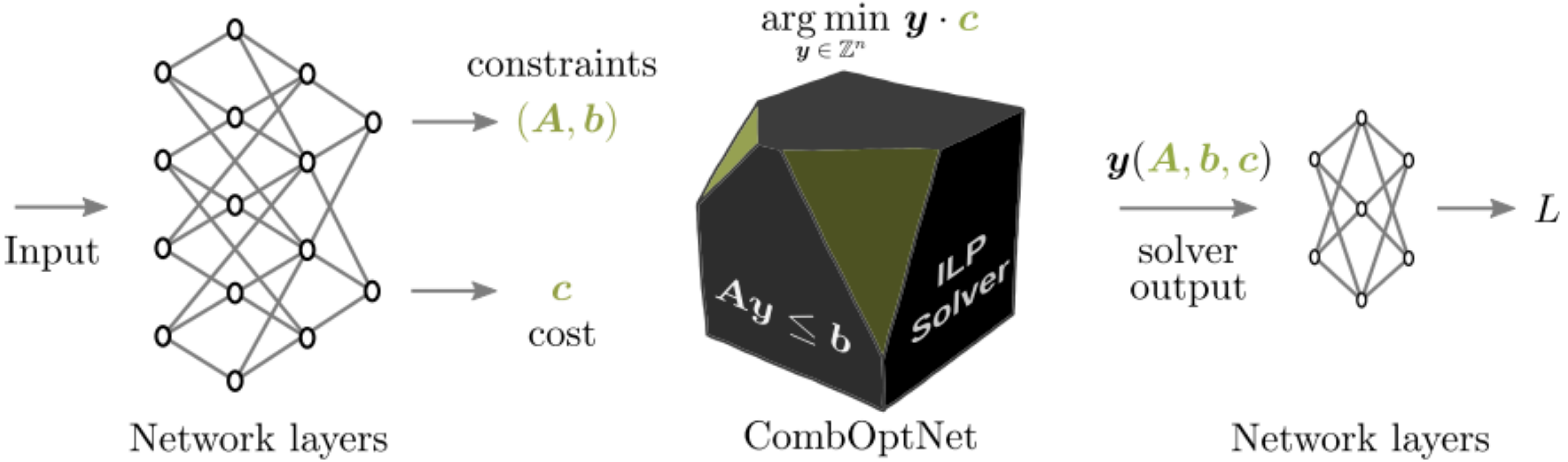
$\min c \cdot y$ such that $Ay + b \geq 0, y \in \mathbb{Z}^n$

Related Work 1: CombOptNet

CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints

Anselm Paulus, Michal Rolínek, Vít Musil, Brandon Amos, and Georg Martius

ICML 2021



Constraint Satisfaction Framework

Cost is ZERO

- Geometry & ML perspective:

  contraint $\leftrightarrow$ hyperplane $\leftrightarrow$ linear classifier

  all constraints $\leftrightarrow$ polytope $\leftrightarrow$ *veto-ensemble* of linear classifiers

- Supervision dichotomy:

  Inference $\Longrightarrow$ combinatorial search via ILP solver

  Training $\Longrightarrow$ binary classification on +ve & -ve examples

- Example source:

  +ve example $\leftarrow$ unique, target solution

  -ve example(s) $\leftarrow$ exponential, sampling strategy?

Proposed Loss Function

- $d_j(A, b, y) = \dfrac{A_j \cdot y + b_j}{\|A_j\|}$ → signed distance from hyperplane
- $\ell(d_j, \pm)$ → some binary classification loss
- $\mathscr{L}(A, b, y_+) = \max_j \ell(d_j, +)$
- $\mathscr{L}(A, b, y_-) = \min_j \ell(d_j, -)$
- Smooth max/min with temperature τ
- Hinge Loss with fixed margin μ:

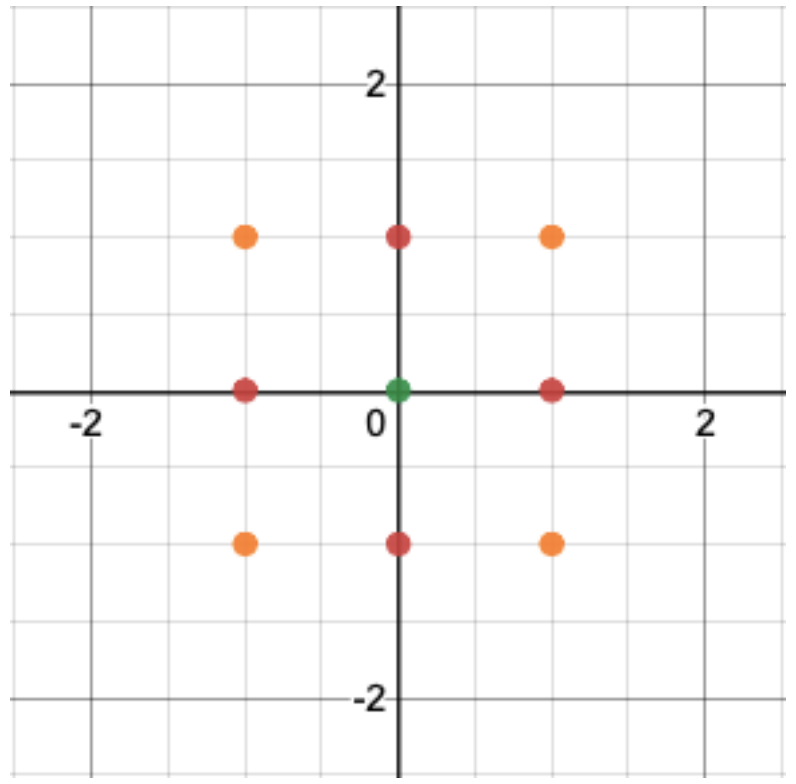
$$\ell(d_j, +) = \mathrm{ReLU}(\mu - d_j), \qquad \ell(d_j, -) = \mathrm{ReLU}(\mu + d_j)$$

- Cross-Entropy Loss:

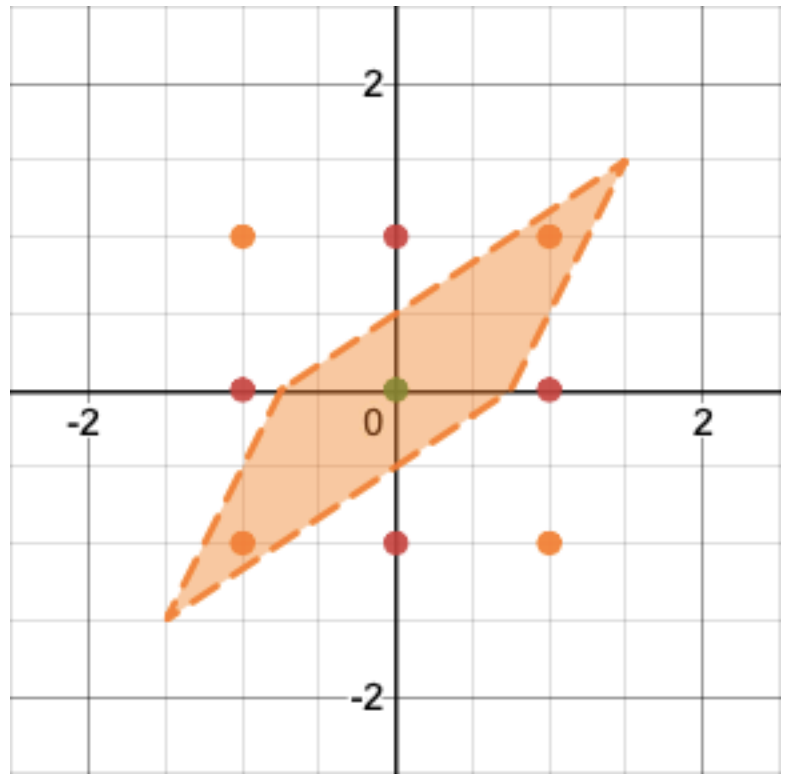
$$\ell(d_j, +) = -\log(\sigma(d_j)), \qquad \ell(d_j, -) = -\log(1 - \sigma(d_j))$$

Solver-Free Negative Sampling

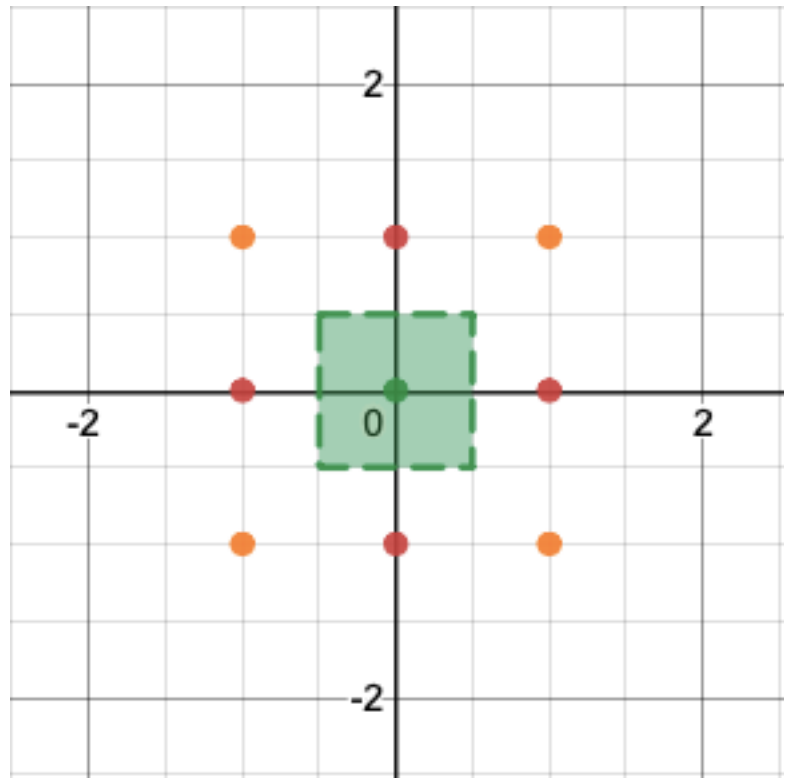
Nearest Neighbours



+ve & -ve examples

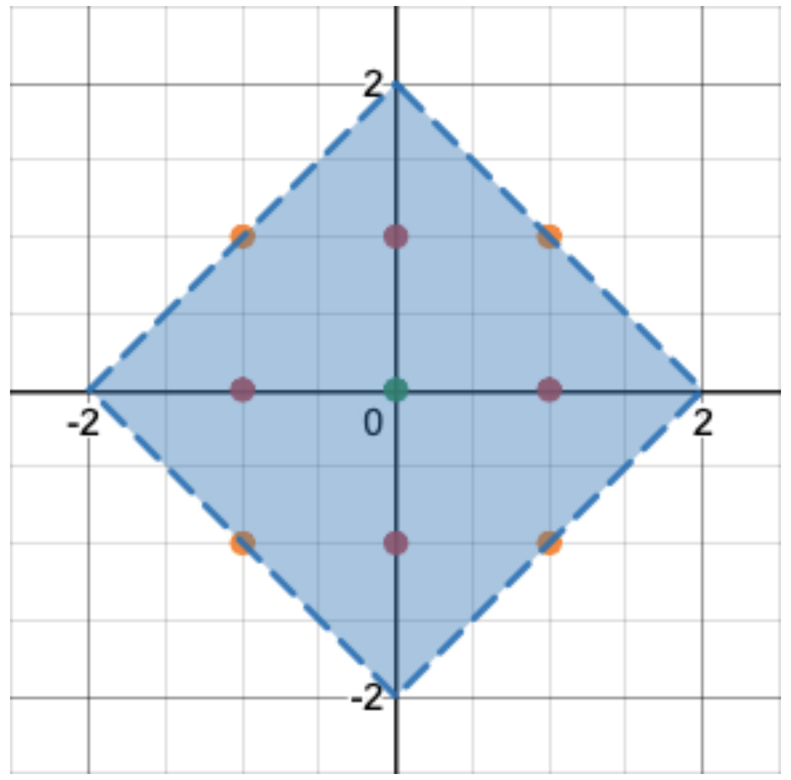


a polytope excluding nearest neighbours

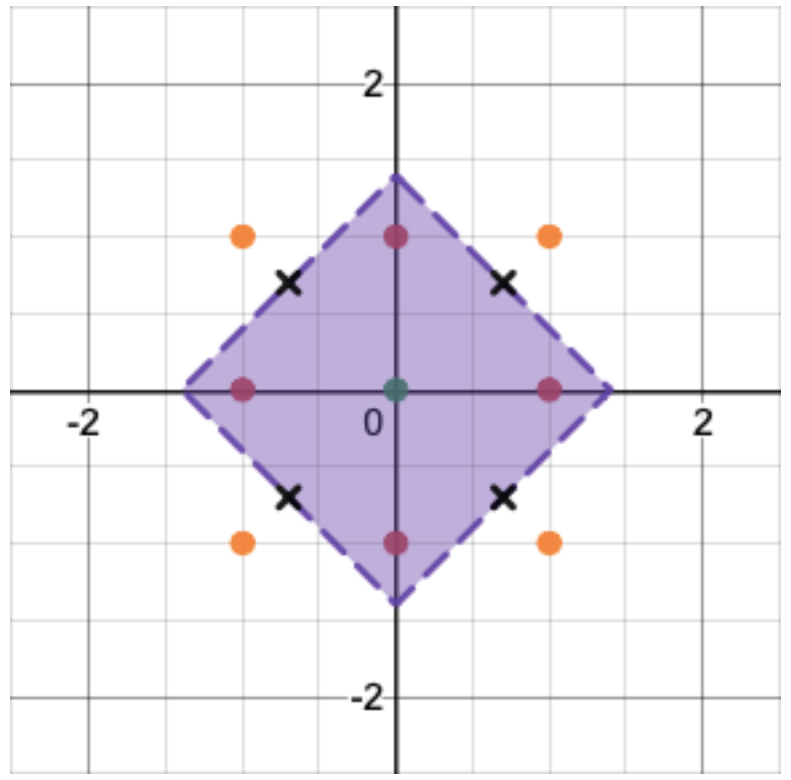


a possible learnt polytope

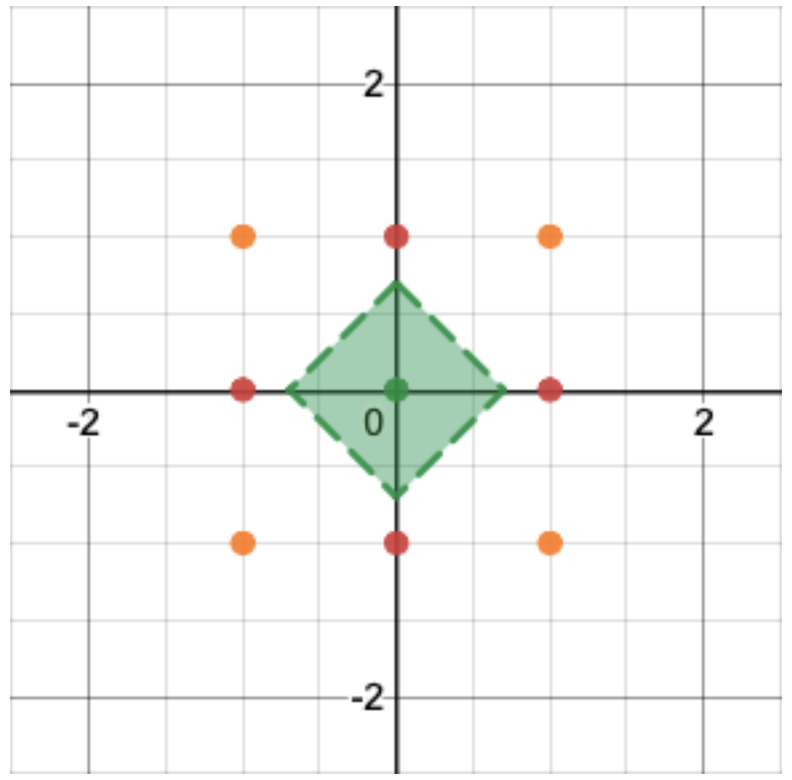
Project and Round



an interim learnt polytope



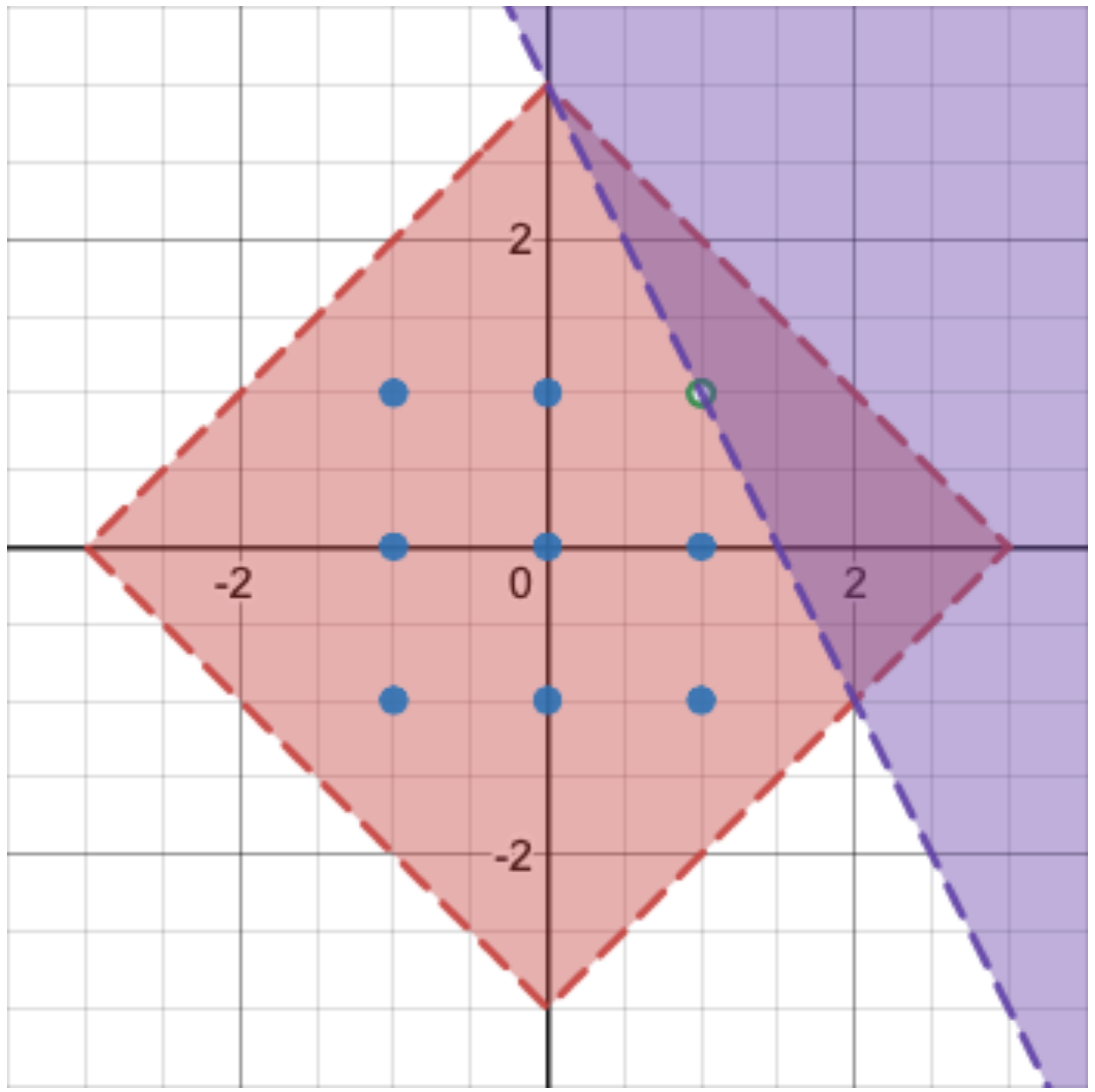
non-integral projected negatives



a possible learnt polytope

Extending to Constraint Optimization

Capture cost via new constraint: $c \cdot y \leq c \cdot y_+$



Experimental Setup

Task: Symbolic Sudoku

- **Sudoku sizes:** 4x4, 6x6
- **Sudoku modelling:** given digits as constraints, given digits as cost weights
- **Binary classification loss:** Cross-Entropy, Hinge
- **Negative sampling:** Nearest Neighbours, Projection, ILP solution, LP solution
- **Baselines:** CombOptNet, Rectifier Networks
- Test accuracy is either 100% or 0%

New Work

Visual Sudoku

Input: grid of digit images

Output: grid of integers

Model:

$A, b \rightarrow$ input-independent, learnable

$c = \mathrm{CONCAT}(c_{0,0}, \ldots, c_{H,W})$

$c_{i,j} = \mathrm{CNN}(x_{i,j})$

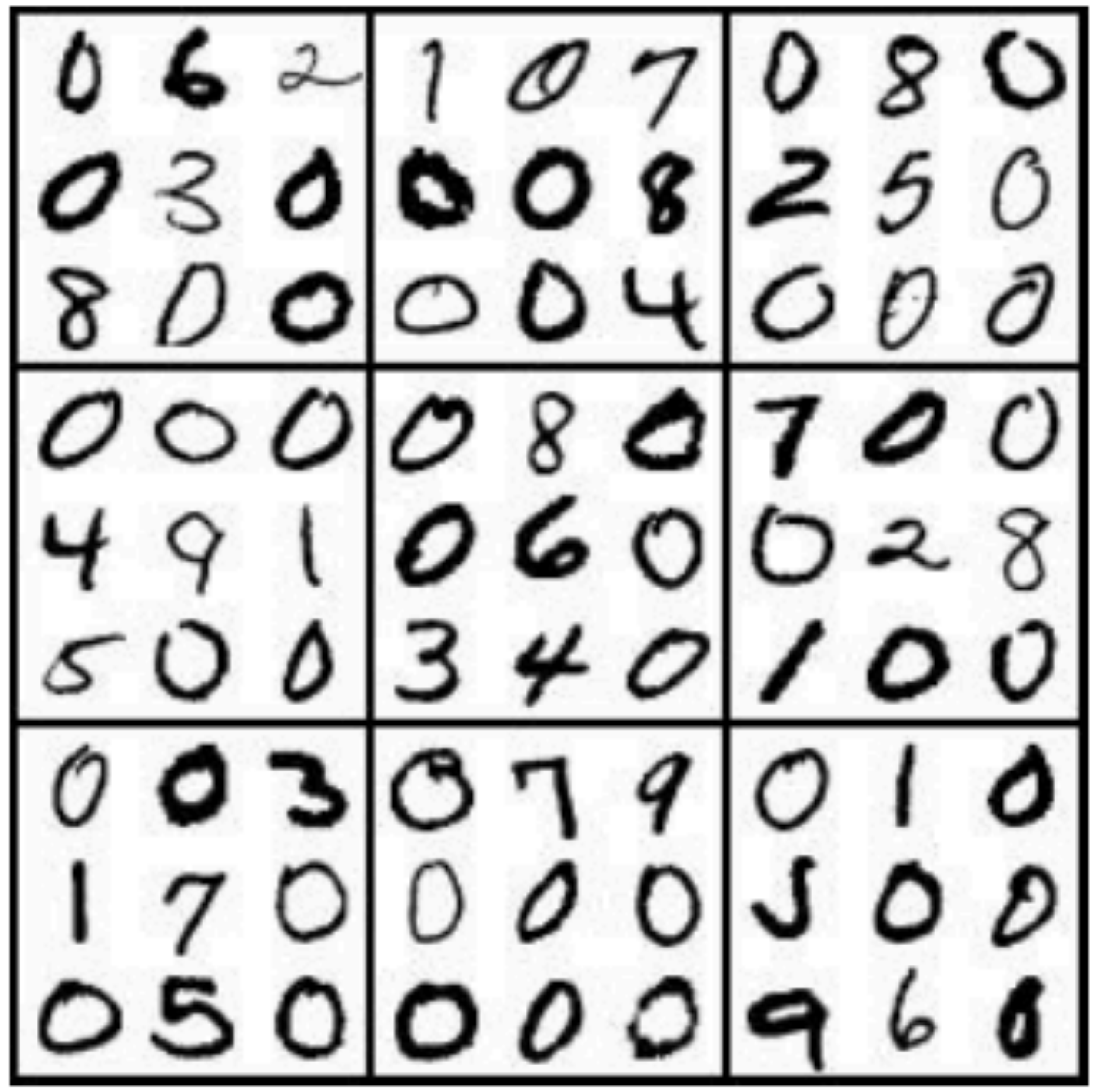


Figure 3. An example visual Sudoku image input, i.e. an image of a Sudoku board constructed with MNIST digits. Cells filled with the numbers 1-9 are fixed, and zeros represent unknowns.

Figure from *Wang, Po-Wei, et al. "Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver."*

A New Negative Sampler

- Shared constraints exclude invalid sudokus
- Cost constraint excludes input-incompatible but valid sudokus

- **Batch Negative Sampler:**

  return other examples from the batch

Results

- 100% *classifier* and *board* **accuracy** on 4x4 and 6x6 sudoku
- **Training time** (representative numbers for 4x4 sudoku):
  - Our approach:
    - symbolic: ~3 mins, ~5 epochs
    - visual: ~15 mins, ~15 epochs
    - A, b given: ~1 min, ~1 epoch
  - CombOptNet:
    - symbolic: ~11 hrs, ~5 epochs
    - A, b given: ~5 mins, ~1 epoch

Recurring Issue - 1

- Training accuracy increases, then decreases
- Loss decreases, but accuracy also decreases
- **Instances:**
  - 6x6 visual → reaches 100% but decreases
  - 6x6 symbolic → reaches 100% but decreases
  - synthetic dataset (8 vars, 8 constrs):
    - CombOptNet: 77% accuracy, then decreases, but loss also increases
    - Our approach: 17% accuracy at peak, then decreases

Recurring Issue - 2

- +ve error decreases but -ve error increases, OR,
- -ve error decreases but +ve error increases
- difficult to optimize

- **Instances:**
  - 9x9 sudoku
  - synthetic dataset

Reasons

- $d_j(A, b, y) = \dfrac{A_j \cdot y + b_j}{\|A_j\|}$ → signed distance from hyperplane
- $\boxed{\ell(d_j, \pm)}$ → some binary classification loss
- $\mathscr{L}(A, b, y_+) = \max_j \ell(d_j, +)$
- $\mathscr{L}(A, b, y_-) = \min_j \ell(d_j, -)$
- Smooth max/min with temperature τ
- Hinge Loss with fixed margin μ:
$$\ell(d_j, +) = \mathrm{ReLU}(\mu - d_j), \qquad \ell(d_j, -) = \mathrm{ReLU}(\mu + d_j)$$
- Cross-Entropy Loss:
$$\ell(d_j, +) = -\log(\sigma(d_j)), \qquad \ell(d_j, -) = -\log(1 - \sigma(d_j))$$

New Formulation

- View polytope as one classifier, not an ensemble
- Point-wise approximation as linear classifier
- $j = \arg\min d_j(y_\pm)$
- Hard min over learnable constraints
- Remove negatives excluded by non-learnable constraints
- $\mathscr{L}(A, b, y_+) = \mathrm{ReLU}(-A_j \cdot y_+ - b_j)$
- $\mathscr{L}(A, b, y_-) = \|A_j\|^2 + C \cdot \mathrm{ReLU}(1 + A_j \cdot y_- + b_j)$

Preliminary Results

- Synthetic dataset:
  - 95% accuracy, ~30 mins
  - accuracy increases monotonically
  - easier to optimize

- Sudoku: needs debugging :(

Next Steps

- Similar formulation for cross-entropy
- Get it to work on sudoku
- Scale to 9x9 sudokus
- Other tasks and datasets

Thank You!