Supplement to Chapter 9: Understanding SATNet: Constraint Learning and Symbol Grounding

Supplment to Chapter 9 from Neuro Symbolic Reasoning and Learning - Current Advances and Future Directions

Learning the Relaxed Constraint Matrix

We now discuss the backpropagation process used to learn the relaxed constraint matrix. At a high level, this involves propagating the gradient from the outputs with respect to the loss function (denoted ℓ) by computing the gradient with respect to the relaxation of the outputs ($\frac{\partial \ell}{\partial V_O}$), then pushing the gradients through the semidefinite program routine, resulting in matrix U, which in turn allows for the computation of the gradient with respect to the relaxed inputs ($\frac{\partial \ell}{\partial V_I}$) and with respect to the relaxed constraint matrix ($\frac{\partial \ell}{\partial S}$) and with respect to the original inputs ($\frac{\partial \ell}{\partial \tilde{V}_I}$). First, we show the derivation of $\frac{\partial \ell}{\partial v_O}$ (v_o is a component of V_O) as follows.

$$\frac{\partial \ell}{\partial v_o} = \left(\frac{\partial \ell}{\partial \tilde{v}_o}\right) \frac{1}{\pi \sin(\pi \tilde{v}_o)} v_\top$$

Note that here o are only indices of set O and $\tilde{v}_o$ refers to the probabilistic value.

The next step in backward propagation is to push the gradients through the SDP optimization routine providing matrix U that, in turn, is used to compute $\frac{\partial \ell}{\partial V_I}$ and $\frac{\partial \ell}{\partial S}$. This is done by taking the total differential through the coordinate descent updates for each of the outputs where the coordinate descent with Algorithm .

Coordinate Descent for Backward Pass For each $o \in O$, $\frac{\partial \ell}{\partial v_o}$ U_O $U_O \leftarrow 0$ $\Psi \leftarrow (U_O) S_O^T$ *not converged* $o \in O$ $\mathsf{d}g_o \leftarrow \Psi s_o - \|s_o\|^2 u_o - \frac{\partial \ell}{\partial v_o} u_o \leftarrow -P_o \mathsf{d}g_o / \|g_o\|$ where $g_o = VS^T s_o - \|s_o\|^2 v_o$ and $P_o \equiv I_k - v_o v_o^T$ $\Psi \leftarrow \Psi + (u_o - u_o^{prev}) s_o^T$

With the calculation of the U matrix, we can then find $\frac{\partial \ell}{\partial V_I}$ and $\frac{\partial \ell}{\partial S}$ as follows (Expressions 0.1 and 0.2).

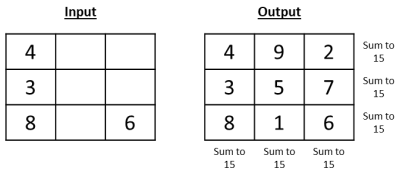

Fig. 0.1 Example input and output for Sudoku.

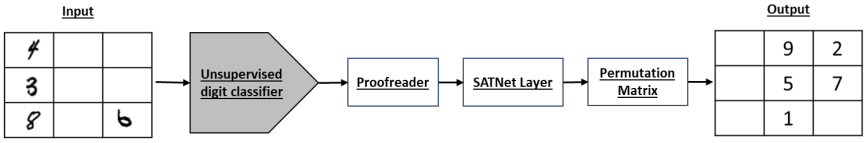

Fig. 0.2 Overview of SATNet integrated with an unsupervised perceptual model.

$$\frac{\partial \ell}{\partial V_I} = -\left(\sum_{o \in O} u_o s_o^T\right) S_I \tag{0.1}$$

$$\frac{\partial \ell}{\partial S} = -\left(\sum_{o \in O} u_o s_o^T\right)^T V - (SV^T)U \tag{0.2}$$

Finally, to compute the derivative with respect to the initial inputs (as opposed to the vector relaxations), we have the following.

$$\frac{\partial \ell}{\partial \tilde{v}_i} = -\left(\frac{\partial v_i}{\partial \tilde{v}_i}\right)^T \left(\sum_{o \in O} u_u s_o^T\right) s_i \tag{0.3}$$

$$\textit{Where} \quad \frac{\partial v_i}{\partial \tilde{v}_i} = \pi(\sin(\pi \tilde{v}_i) v^\top + \cos(\pi \tilde{v}_i)(I_k - v_\top v_\top^T) v_i^{rand}) \tag{0.4}$$

Note that additional direct dependencies between ℓ and $\tilde{v}_i$ should be added as a term to Expression 0.3.

Addressing Symbol Grounding in SATNet.

The research gap identified by [1] led to further investigation into SATNet's performance. There was interest to see if a combined perceptual-reasoning framework could be trained. A promising approach emerged in [2]. The idea is to use unsupervised learning, self-grounded training, and an additional proofreading layer to train the model without any supervision. An overview of this framework is shown in 0.2.

Unsupervised Digit Classifier. In [2] the authors use an unsupervised method on the input data to cluster it into classes. The number of classes (9 in the case of Sudoku) is assumed to be known ahead of time. In [2] the authors use InfoGAN to perform the clustering. To add the clustering knowledge to the neural architecture, they use it to train LeNet based on a 1-hot encoding - but note there is still done without access to ground truth information of the digits (LeNet is trained using the results of the clustering algorithm).

Self-Grounded Training. The output of LeNet has used an input for the remaining layers. However, the meaning of the digits is still not yet known. The output of LeNet is an encoding of the digits (the pre-trained encoding) denoted $\hat{y}_{in}^{PTE}$. This is related to the actual label encoding, $\hat{y}_{in}^{LE}$ by a permutation matrix P, giving us the following.

$$\hat{y}_{in}^{PTE}\mathsf{P} = \hat{y}_{in}^{LE} \tag{0.5}$$

Note that SATNet can be trained correctly on *either* $\hat{y}_{in}^{PTE}$ or $\hat{y}_{in}^{LE}$ (this was empirically verified in [2]). This is because SATNet is permutation-invariant. We note that the output set in the training, y^{LE} is symbolic (even when masked). Hence, we can assume the following relationship.

$$\hat{y}_{out}^{PTE}\mathsf{P} \approx y^{LE} \tag{0.6}$$

This leads to the following loss function.

$$\mathcal{L}(\hat{y}_{out}^{PTE}, y^{LE}) = 1 - \mathsf{mean}_i(\max_j(e^{-BCE(y^{LE}(j),\hat{y}_{out}^{PTE}(i))})) \tag{0.7}$$

Where BCE is the binary cross-entropy loss between two vectors. By minimizing this loss function, we can recover an approximate permutation matrix.

$$\hat{\mathsf{P}}_{ij} = e^{-BCE(y^{LE}(j),\hat{y}_{out}^{PTE}(i))} \tag{0.8}$$

In practice, the max function is replaced with an approximation (in [2] the authors use a 2-norm, but mention that results were not sensitive to this choice).

Training proceeds as follows. The LeNet layers (already trained from the results of the unsupervised step) are now frozen and the training of the SATNet layers proceeds using the loss function of Expression 0.7. This continues until $\hat{P}$ has converged. Now, the loss function is replaced with standard cross entropy, using $\hat{P}$ (now frozen) to return the proper label encodings. The LeNet layers are also unfrozen at this time. We also note an additional "proofreading" linear layer is also added to improve performance. This layer is added prior to the SATNet layers. It constitutes a final step of training where the rest of the architecture is frozen and the proofreading layer is trained.

Results. While the method proposed by [2] does not provide an ideal end-to-end neural architecture (e.g., train a single architecture with a single run of gradient descent), it does allow for training with a totally untrained perceptual layer where

the only information known was the number of classes. It achieves an accuracy of 64.8%, which is comparable to SATNet's performance with indirect supervision.

References

1. Chang, O., Flokas, L., Lipson, H., Spranger, M.: Assessing satnet's ability to solve the symbol grounding problem. In: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 1428–1439. Curran Associates, Inc. (2020). URL https://proceedings.neurips.cc/paper/2020/file/0ff8033cf9437c213ee13937b1c4c455-Paper.pdf
2. Topan, S., Rolnick, D., Si, X.: Techniques for symbol grounding with satnet. Advances in Neural Information Processing Systems **34**, 20733–20744 (2021)
3. Wang, P., Donti, P.L., Wilder, B., Kolter, J.Z.: Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In: K. Chaudhuri, R. Salakhutdinov (eds.) Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, *Proceedings of Machine Learning Research*, vol. 97, pp. 6545–6554. PMLR (2019). URL http://proceedings.mlr.press/v97/wang19e.html