Towards Effective Deep Learning for Constraint Satisfaction Problems

Hong Xu(✉) Sven Koenig T. K. Satish Kumar
hongx@usc.edu skoenig@usc.edu tkskwork@gmail.com
University of Southern California, Los Angeles, California 90089, the United States of America

Presented at: the 15th International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)
Published in: Proceedings of the 24th International Conference on Principles and Practice of Constraint Programming (CP)

I. Abstract

In this paper, we apply deep learning to predict the satisfiabilities of CSPs. To the best of our knowledge, this is the first effective application of deep learning to CSPs that yields >99.99% prediction accuracy on random Boolean binary CSPs whose constraint tightnesses or constraint densities do not determine their satisfiabilities. We use a deep convolutional neural network on a matrix representation of CSPs. Since it is NP-hard to solve CSPs, labeled data required for training this neural network are in general costly to produce and are thus scarce. We address this issue using the asymptotic behavior of generalized Model A, a new random CSP generation model, along with domain adaptation and data augmentation techniques for CSPs. We demonstrate the effectiveness of our deep learning techniques using experiments on random Boolean binary CSPs. While these CSPs are known to be in P, we used them as an initial attempt.

II. Motivation and Challenges

- Deep learning has achieved great success in the past decade in fields such as computer vision and natural language processing. Unlike most other supervised learning algorithms, it does not require handcrafting features. Therefore, it is desirable to apply it to CSPs.
- However, deep learning usually requires huge amounts of training data to be effective.
- Therefore, it is challenging to apply deep learning on CSPs, since it is NP-hard to solve CSPs and thus it is costly to label them, such as computing their satisfiabilities or finding the most efficient algorithms to solve them.

III. Definition

- A CSP is defined as a tuple $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$, where
  - $\mathcal{X} = \{X_1, \ldots, X_n\}$ is a set of variables,
  - $\mathcal{D} = \{D_1, \ldots, D_n\}$ is a set of domains corresponding to their respective variables, and
  - $\mathcal{C} = \{C_1, \ldots, C_m\}$ is a set of constraints.
- Each constraint $C_i \in \mathcal{C}$ is a pair $\langle S(C_i), R_i \rangle$, where $S(C_i)$ is a subset of $\mathcal{X}$ and R_i is a $|S(C_i)|$-ary relation that specifies incompatible and compatible assignments of values to variables in $S(C_i)$.
- In a *table constraint* C_i, R_i is a set of tuples, each of which indicates the *compatibility* of an assignment of values to variables in $S(C_i)$. A tuple is compatible if it specifies a compatible assignment of values to variables, and is otherwise incompatible.

IV. (Deep) Convolutional Neural Network (cNN)

- cNNs were initially proposed for object recognition tasks [1].
  - Convolutional layers: perform a convolution operation.
  - Pooling layers: combine the outputs of several nodes in the previous layer into a single node in the current layer.
  - Fully connected layers: connect every node in the current layer to every node in the previous layer.
- Representing input CSPs as *CSP matrices*, where
  - each row / column represents an assignment of a value to a variable $X_i = x_i$ / $X_j = x_j$ and
  - the element in this row and column indicates the compatibility of the tuple $\{X_i = x_i, X_j = x_j\}$. (0 indicates compatibility, and 1 indicates incompatibility.)

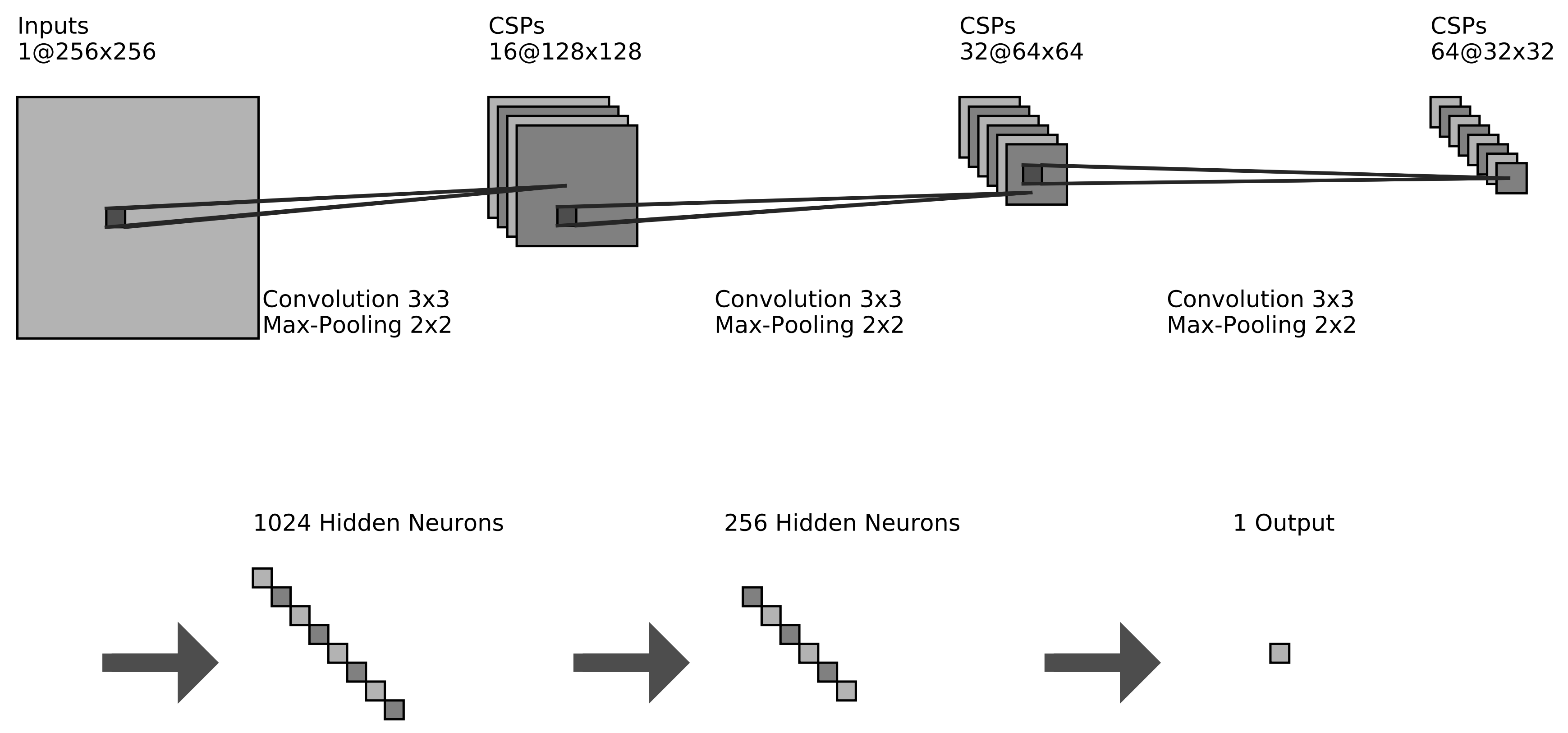

Figure 1: The architecture of our cNN, referred to as CSP-cNN.

V. Efficient Training Data Generation

- We created generalized Model A, a random binary CSP generation model generalized from Model A [3].
  - For every two variables X_i and X_j, add a constraint between them with probability p.
  - For every constraint between X_i and X_j, mark each tuple as incompatible with probability q_{ij}.
- We label all CSPs generated by generalized Model A `UNSATISFIABLE`.
- To generate a CSP labeled `SATISFIABLE`,
  - generate a random solution a,
  - generate a CSP using generalized Model A, and
  - in each constraint, mark a tuple incompatible if it conflicts with a.
- The probability to (mis)label a satisfiable CSP `UNSATISFIABLE` is no greater than $\prod_{X_i \in \mathcal{X}} |D(X_i)| \prod_{X_i, X_j \in \mathcal{X}} (1 - pq_{ij})$.
- We refer to this training data generation method as *generalized Model A-based method* (GMAM).

VI. Domain Adaptation and Data Augmentation

- *Domain adaptation*:
  - A deep NN is usually ineffective on target dataset if it is trained on a dataset from a different distribution.
  - Mix the target dataset with a GMAM generated dataset.
- *Data Augmentation*: Create more training data by exchanging rows and columns in CSP matrices.

VII. Experimental Results

- On Boolean binary CSPs with 128 variables
- GMAM generated dataset: 300,000 training data points, 10,000 validation data points, and 10,000 test data points.

	CSP-cNN	NN-image [2]	NN-1	NN-2	M
Acc (%)	>99.99	50.01	98.11	98.66	64.79

- *Modified Model E-based method* (MMEM) generated dataset: 1200 data points in total, 3-fold cross validation.
  - bipartite structure
  - Data augmentation on MMEM data and mixing with GMAM generated data

Data Trained	Mixed	MMEM	GMAM
Accuracy (%)	100.00/100.00/100.00	50.00/50.00/50.00	50.00

- Varying percentage of MMEM generated data in the training dataset

Percentage of MMEM (%)	0.00	33.33	36.00	40.00	46.66	53.33	66.67	70.67	78.67	100.00
Average Accuracy (%)	50.00	100.00	100.00	83.33	66.67	83.33	66.67	66.67	50.00	50.00

VIII. References

[1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1.4 (1989), pp. 541–551. DOI: `10.1162/neco.1989.1.4.541`.

[2] Andrea Loreggia, Yuri Malitsky, Horst Samulowitz, and Vijay Saraswat. "Deep Learning for Algorithm Portfolios". In: *the AAAI Conference on Artificial Intelligence*. 2016, pp. 1280–1286.

[3] Barbara M. Smith and Martin E. Dyer. "Locating the phase transition in binary constraint satisfaction problems". In: *Artificial Intelligence* 81.1 (1996), pp. 155–181. DOI: `10.1016/0004-3702(95)00052-6`.

The research at the University of Southern California was supported by the National Science Foundation under grant numbers 1724392, 1409987, and 1319966.