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

August 28, 2018

University of Southern California
the 24th International Conference on Principles and Practice of Constraint Programming (CP 2018)
Lille, France

Executive Summary

- The Constraint Satisfaction Problem (CSP) is a fundamental problem in constraint programming.
- Traditionally, the CSP has been solved using search and constraint propagation.
- For the first time, we attack this problem using a convolutional Neural Network (cNN) with preliminary high effectiveness on subclasses of CSPs that are known to be in P.

Overview

In this talk:

- We intend to use convolutional neural networks (cNNs) to predict the satisfiability of the CSP.
- We review the concepts of the CSP and cNNs.
- We present how a CSP instance can be input of a cNN.
- We develop Generalized Model A-based Method (GMAM) to efficiently generate massive training data with low mislabeling rates, and present how they can be applied to general CSP instances.
- As a proof of concept, we experimentally evaluated our approaches on binary Boolean CSP instances (which are known to be in P).
- We discuss potential limitations of our approaches.

Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

Constraint Satisfaction Problem (CSP)

- N variables $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$.
- Each variable X_i has a discrete-valued domain $\mathcal{D}(X_i)$.
- M constraints $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$.
- Each constraint C_i is a list of tuples in which each specifies the compatibility of an assignment a of values to a subset $S(C_i)$ of the variables.
- Find an assignment a of values to these variables so as to satisfy all constraints in $\mathcal{C}$.
    - Decision version: Does there exist such an assignment a?
- Known to be NP-complete.

Example

- $\mathcal{X} = \{X_1, X_2, X_3\}, \mathcal{C} = \{C_1, C_2\}, \mathcal{D}(X_1) = \mathcal{D}(X_2) = \mathcal{D}(X_3) = \{0, 1\}$
- C_1 disallows $\{X_1 = 0, X_2 = 0\}$ and $\{X_1 = 1, X_2 = 1\}$.
- C_2 disallows $\{X_2 = 0, X_3 = 0\}$ and $\{X_2 = 1, X_3 = 1\}$.
- There exists a solution, and $\{X_1 = 0, X_2 = 1, X_3 = 0\}$ is one solution.

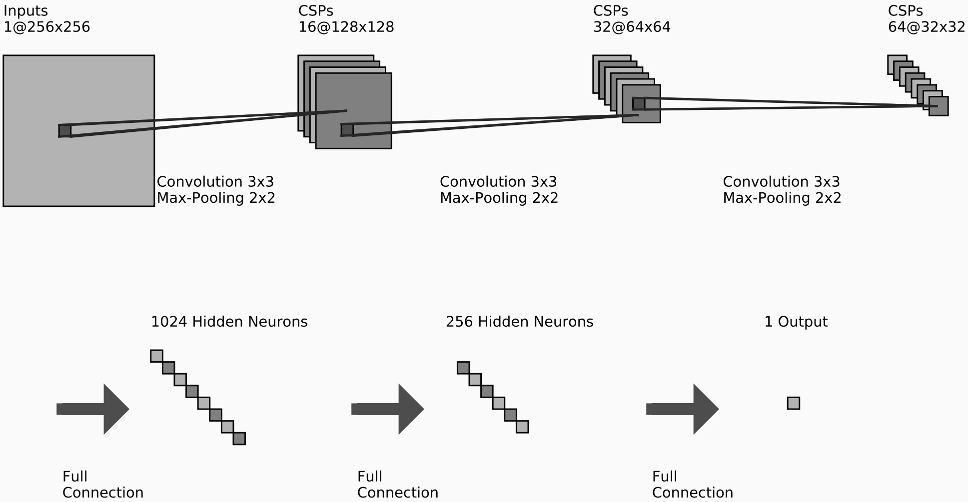
Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

The Convolutional Neural Network (cNN)

- is a class of deep NN architectures.
- was initially proposed for an object recognition problem and has recently achieved great success.
- is a multi-layer feedforward NN that takes a multi-dimensional (usually 2-D or 3-D) matrix as input.
- has three types of layers:
    - A *convolutional layer* performs a convolution operation.
    - A *pooling layer* combines the outputs of several nodes in the previous layer into a single node in the current layer.
    - A *fully connected layer* connects every node in the current layer to every node in the previous layer.

Architecture



CSP-cNN. L2 regularization coefficient 0.01 (output layer 0.1).

A Binary CSP Instance as a Matrix

- A symmetric square matrix
  - Each row and column represents a variable $X_i \in \mathcal{X}$ and an assignment $x_i \in \mathcal{D}(X_i)$ of value to it (i.e., $X_i = x_i$)
  - An entry is 0 if its corresponding assignments of values are compatible. Otherwise, it is 1.
- Example: $\{X_i = 0, X_j = 1\}$ and $\{X_i = 1, X_j = 0\}$ are incompatible.

	$X_i = 0$	$X_i = 1$	$X_j = 0$	$X_j = 1$
$X_i = 0$	0	1	0	1
$X_i = 1$	1	0	1	0
$X_j = 0$	0	1	0	1
$X_j = 1$	1	0	1	0

Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

Lack of Training Data

- Deep cNNs need huge amounts of data to be effective.
- The CSP is NP-hard, which makes it hard to generate labeled training data.
- Need to generate huge amounts of training data with
    - efficient labeling and
    - substantial information.

Generalized Model A

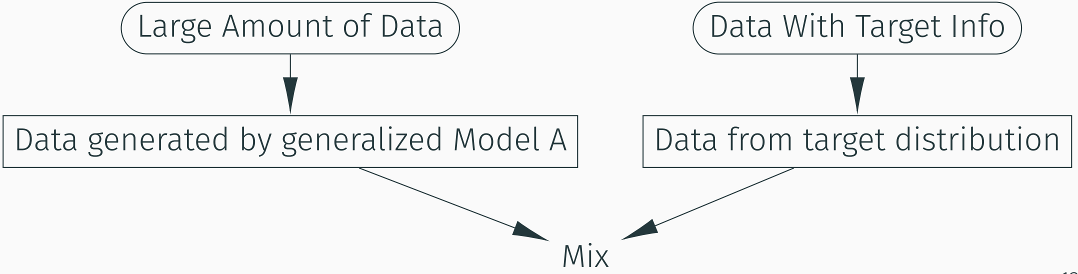
- Generalized Model A is a random CSP generation model.
    - Randomly add a constraint between each pair of variables X_i and X_j with probability $p > 0$.
    - Add an incompatible tuple for each assignment $\{X_i = x_i, X_j = x_j\}$ with probability $q_{ij} > 0$.
- Property: As the number of variables tends to infinity, it generates only unsatisfiable CSP instances (extension of results for Model A (Smith et al. 1996)).
- Quick labeling: A CSP instance generated by generalized Model A is likely to be unsatisfiable, and we can inject solutions in CSP instances generated by generalized Model A to generate satisfiable CSP instances.

Generating Training Data

- Randomly select p and q_{ij} and use generalized Model A to generate CSP instances.
- Inject a solution: For half of these instances, randomly generate an assignment of values to all variables and remove all tuples that are incompatible with it.
- We now have training data, in which half are satisfiable and half are not.
- Mislabeling rate: Satisfiable CSP instances are 100% correctly labeled. We proved that unsatisfiable CSP instances have mislabeling rate no greater than $\prod_{X_i \in \mathcal{X}} |\mathcal{D}(X_i)| \prod_{X_i, X_j \in \mathcal{X}} (1 - pq_{ij})$.
- This mislabeling rate can be as small as 2.14×10^{-13} if $p, q_{ij} > 0.12$.
- No obvious parameter indicating their satisfiabilities.

To Predict on CSP Instances not from Generalized Model A...

- Training data from target data source are usually scarce due to CSP's NP-hardness.
- Need domain adaptation: Mixing training data from target data source and generalized Model A.



To Creating More Instances...

- Augmenting CSP instances from target data source without changing their satisfiabilities (label-preserved transformation):
  - Exchanging rows and columns representing different variables.
  - Exchanging rows and columns representing different values of the same variable.
- Example: Exchange the red and blue rows and columns.

	$X_i = 0$	$X_i = 1$	$X_j = 0$	$X_j = 1$
$X_i = 0$	0	1	0	1
$X_i = 1$	1	0	0	1
$X_j = 0$	0	0	0	1
$X_j = 1$	1	1	1	0

Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

On CSP Instances Generated by Generalized Model A

- 220,000 binary Boolean CSP instances by Generalized Model A.
- They are in P; we evaluated on them as a proof of concept.
    - p and q_{ij} are randomly selected in the range $[0.12, 0.99]$ (mislabeling rate $\leq 2.14 \times 10^{-13}$).
    - Half are labeled satisfiable and half are labeled unsatisfiable.
    - Training data: $200,000$ CSP instances
    - Validation and Test data: $10,000$ and $10,000$ CSP instances
- Training hyperparameters:
    - He-initialization
    - Stochastic gradient descent (SGD)
    - Mini-batch size 128
    - Learning rates: 0.01 in the first 5 and 0.001 in the last 54 epoches
    - Loss function: Binary cross entropy

On CSP Instances Generated by Generalized Model A

- Compared with three other NNs and a naive method
  - NN-1 and NN-2: Plain NNs with 1 and 2 hidden layers.
  - NN-image: An NN that can be applied to CSPs (Loreggia et al. 2016).
  - M: A naive method using the number of incompatible tuples.
  - Trained NN-1 and NN-2/NN-image using SGD for 120/60 epoches with learning rates 0.01 in the first 60/5 epoches and 0.001 in the last 60/55 epoches.
- Results:

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

- Although preliminary, to the best of our knowledge, this is the very first known effective deep learning application on the CSP with no obvious parameters indicating their satisfiabilities.

On a Different Set of Instances: Generated by Modified Model E

- Modified Model E: Generating very different CSP instances from those using generalized Model A.
- Divide all variables into two partitions and randomly add a binary constraint between every pair of variables with probability 0.99.
- For each constraint, randomly mark exactly two tuples as incompatible.
- Generate 1200 binary Boolean CSP instances and compute their satisfiabilities using Choco (Prud'homme et al. 2017).
- Once again, these instances are in P, but we evaluated on them as a proof of concept.

On a Different Set of Instances: Generated by Modified Model E

- 3-fold cross validation: 800 training data points and 400 test data points
- Mixed: Augment each training data for 124 times and mix them with CSP instances generated by generalized Model A (300,000 data points for training).
- Baselines:
  - MMEM: Train on these training data after augmenting them for 374 times (to generate 300,000 data points).
  - GMAM: Train on CSP instances generated using generalized Model A only.
- Results:

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

Varying Percentage of MMEM Generated Data when Training

- We varied the percentage of data generated by modified Model E (i.e., augmented data) in the training dataset.
- Results

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

- There exists an optimal mixture percentage.
- This mixture percentage is another hyperparameter to tune.

Agenda

- The Constraint Satisfaction Problem (CSP)
- Convolutional Neural Networks (cNNs) for the CSP
- Generating Massive Training Data
- Experimental Evaluation
- Discussions and Conclusions

Discussion on the Limitations

- So far, we have only experimented on small easy random CSPs that were generated in two very specific ways.
- We still need to
  - understand the generality of our approach, e.g., on larger, hard, and real-world CSPs,
  - analyze what our CSP-cNN learns,
  - evaluate how robust our approach is with respect to the training data and hyperparameters, and
  - understand exactly how our approach should be used, for example, how the effectiveness of our CSP-cNN depends on the amount of available training data and the amount of data augmentation used to increase them.

Conclusion and Future Work

- We developed a machine learning algorithm for predicting satisfiabilities for CSP instances using a deep cNN.
- As a proof of concept, we demonstrate its effectiveness on binary Boolean CSP instances generated using generalized Model A and modified Model E.
- For the first time, we have an effective deep learning approach for the CSP, although we evaluated them on CSPs in P.
- This opens up many future directions:
  - Would it work well on hard CSP instances?
  - Using this satisfiability prediction to guide search algorithms for solving the CSP: Choose the most effective variable to instantiate next.
  - Apply transfer learning techniques to predict other interesting properties of CSP instances, such as the best algorithm to solve them.

References I

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.

Charles Prud’homme, Jean-Guillaume Fages, and Xavier Lorca. *Choco Documentation*. TASC - LS2N CNRS UMR 6241, COSLING S.A.S. 2017. URL: `http://www.choco-solver.org`.

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`.