Neuro-Symbolic Constraint Programming for Structured Prediction

Paolo Dragone+, Stefano Teso, Andrea Passerini**

+ Twitter

* University of Trento, Italy

Neural Constrained Structured Prediction

Structure prediction under constraints

- Linear structured model

$$y = \underset{y' \in \mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \boldsymbol{\phi}(x, y') \rangle$$

$\boldsymbol{\phi}$ Joint input-output feature map

$\boldsymbol{w}$ Learned weight vector

$\mathcal{Y}$ Space of valid output configurations, defined w/ MILP constr.

- Feasible space and inference problem defined as a constraint program (MiniZinc, Gurobi).
- Structured-hinge loss, with or without L2 regularization
- Learning via stochastic sub-gradient method (SSG) and variants.

Neural constrained structured model

- Output of neural network(s) becomes input of structured model

$$\underset{y' \in \mathcal{Y}}{\operatorname{argmax}} \ \langle \boldsymbol{w}, \boldsymbol{\phi}(x, y') \rangle + \langle \boldsymbol{w}_\rho, \boldsymbol{\phi}_\rho(x, \hat{y}, y') \rangle + w_\delta \delta(y', \hat{y})$$

$\text{s.t. } \hat{y} = g(x; W)$ Neural network(s) output

$\langle \boldsymbol{w}_\rho, \boldsymbol{\phi}_\rho \rangle$ Refinement features - bias NN output towards true output

$w_\delta \delta(y', \hat{y})$ Output distance - bias output to be close to NN output

- Inference in two stages, NN first then structured model
- Learning end-to-end, with or without pre-training
- Backpropagation through network layers

$$\nabla L = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial W_k}$$

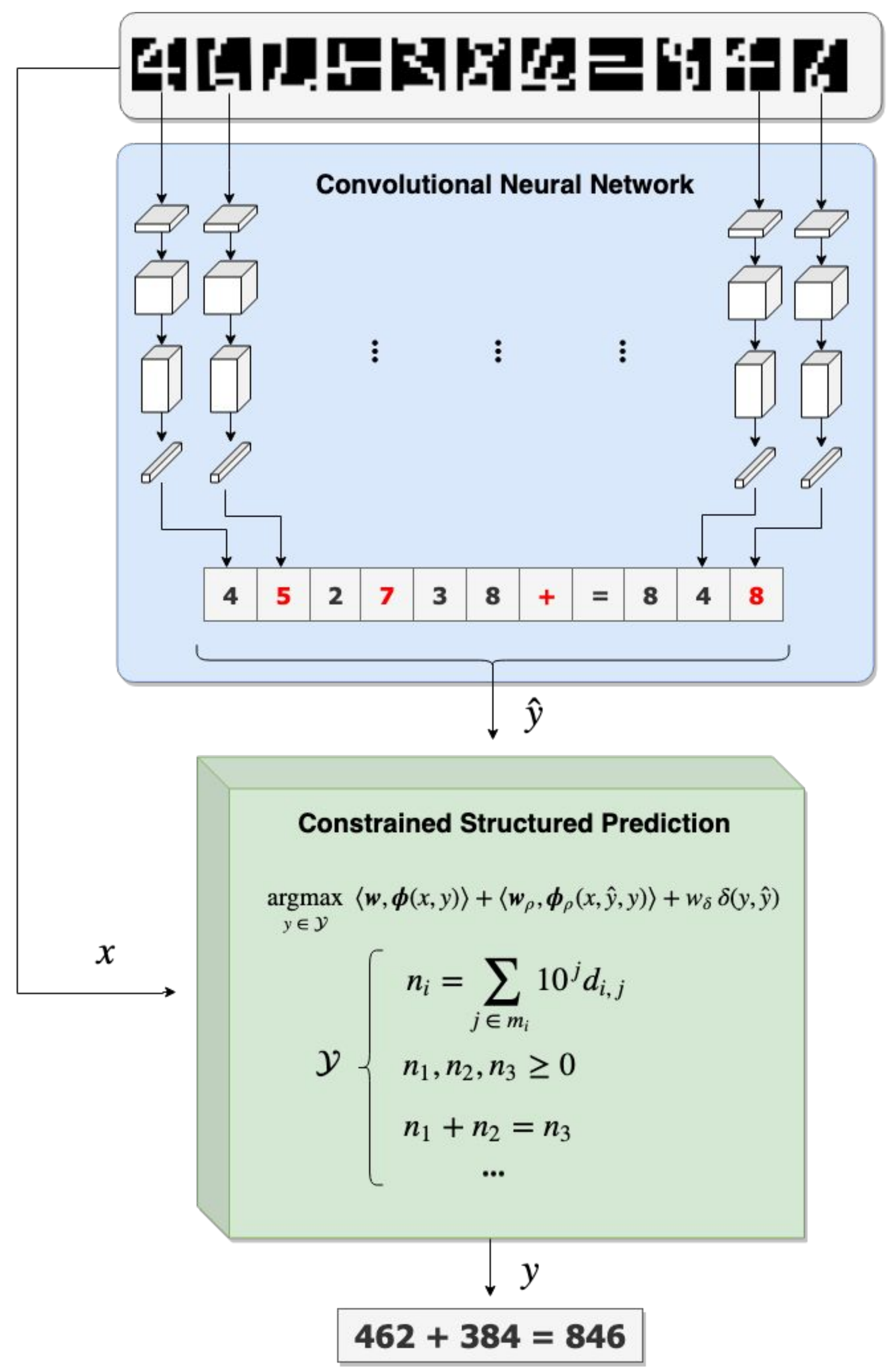
paper

https://arxiv.org/abs/2103.17232

Neural Constrained Sequence Prediction

Example application: valid equation recognition

- Data: Sequence of images representing valid equations
- Valid equations: a + b = c; assume is alway true
- Images processed by convolutional NN
- Single images output combined by structured model into a final output
- Output of structured model always satisfied validity constraints



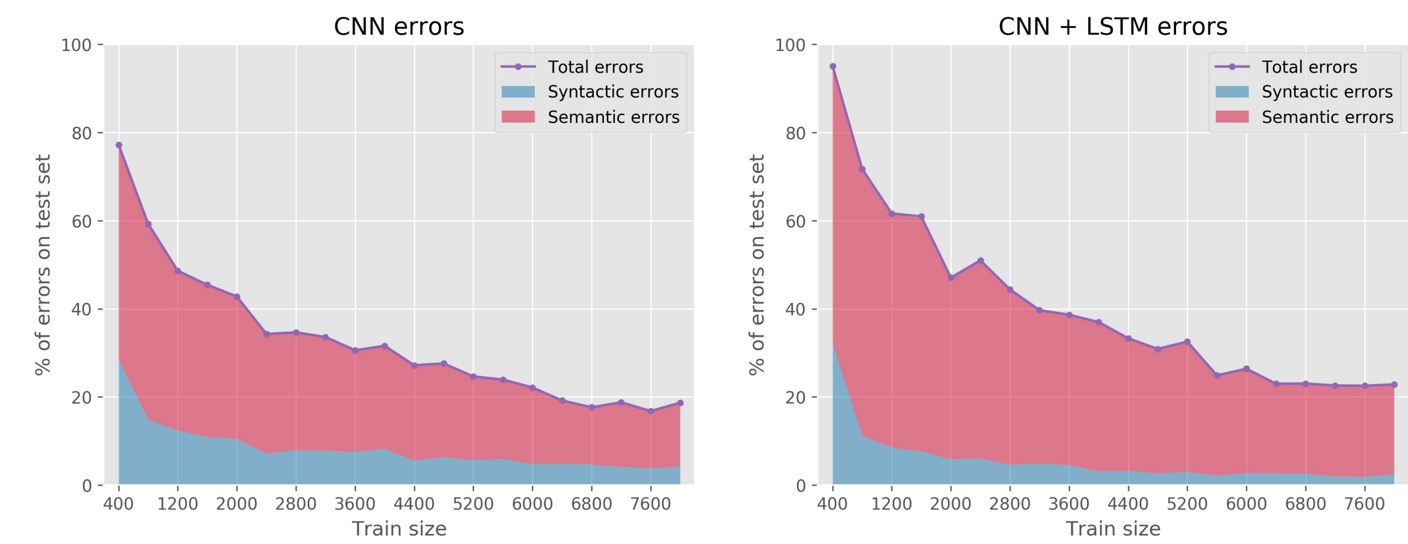
Structured model features

- Prediction features:
  - Emission - bias output symbol based on pixel values
  - Transition - bias output symbol based on previous symbol
- Refinement features:
  - "Correct" NN outputs, bias towards correct symbol
- Distance:
  - Hamming distance between NN sequence and output

Nester - Experiments

Types of prediction error

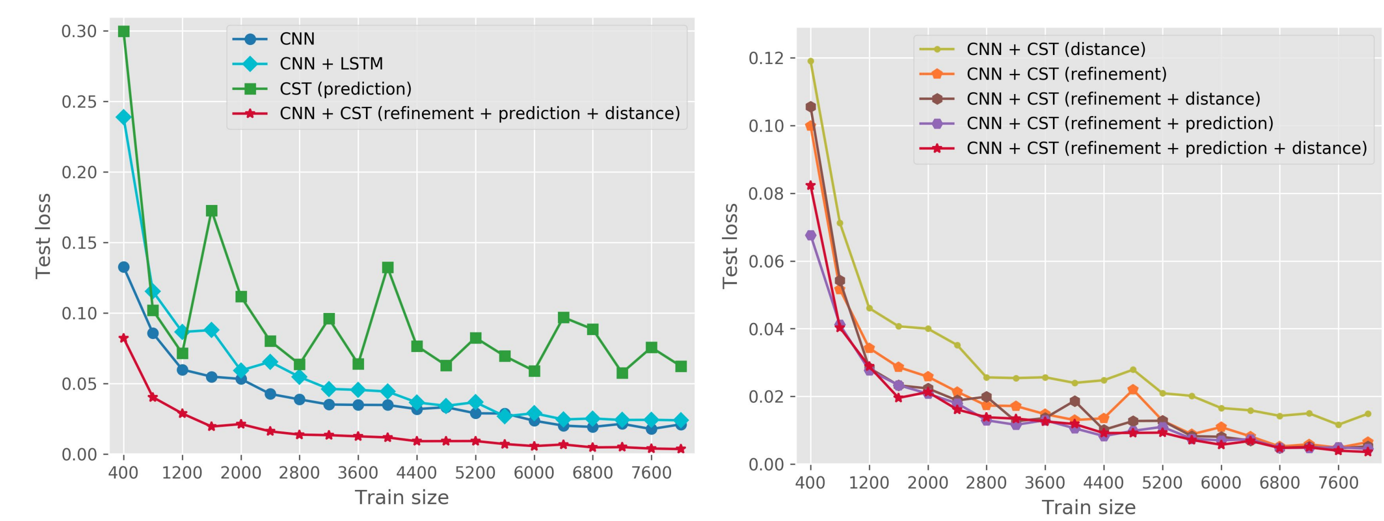
- Syntactic - the sequence is not formatted as "a + b = c"
- Semantic - the equation is not valid, i.e. "a + b ≠ c"



- CNN makes many semantic errors
- Adding LSTM on top of CNN for sequence prediction reduces syntactic errors quickly but not semantic errors
- Semantic errors are generally harder to fix

Comparisons

- CST - constrained structured model
- CNN - convolutional NN
- CNN + CST - combined model (Nester)



- Combined model performs better than both CNN and CST by themselves
- CNN + CST learns faster in low data regimes
- Gap between Nester and CNN - LSTM diminishes with more data
- Using all prediction, refinement and distance features produces the best model