Deep Learning With Constraints

Yatin Nandwani
Work done in collaboration with Abhishek Pathak
Under the guidance of Prof. Mausam and Prof. Parag Singla
Learning with Constraints: *Motivation*

➔ Modern day AI == Deep Learning (DL) [Learn from Data]
Learning with Constraints: *Motivation*

➔ Modern day AI == Deep Learning (DL) [Learn from Data]
➔ Can we inject symbolic knowledge in Deep Learning? E.g. Person => Noun [Learn from Data Knowledge](credit: Vivek S Kumar)
Learning with Constraints: *Motivation*

→ Modern day AI == Deep Learning (DL) [*Learn from Data*]

→ Can we inject symbolic knowledge in Deep Learning? E.g.

  Person => Noun [*Learn from Data Knowledge*] (*credit: Vivek S Kumar*)

→ **Constraints:** One of the ways of representing symbolic knowledge.  

\[ 1\{y_{PER.} = 1\} \implies 1\{y_{Noun.} = 1\} \]
Learning with Constraints: *Motivation*

➔ Modern day AI == Deep Learning (DL) [*Learn from Data*]

➔ Can we inject symbolic knowledge in Deep Learning? E.g.

\[ \text{Person} \Rightarrow \text{Noun} \ [\text{Learn from Data Knowledge}] \]

(credit: Vivek S Kumar)

➔ **Constraints**: One of the ways of representing symbolic knowledge.

\[ \mathbb{1}\{y_{\text{PER.}} = 1\} \implies \mathbb{1}\{y_{\text{Noun.}} = 1\} \]

➔ Limited work in training DL models with (soft) constraints
Learning with Constraints: Motivation

→ Modern day AI == Deep Learning (DL) [Learn from Data]
→ Can we inject symbolic knowledge in Deep Learning? E.g.
  Person => Noun [Learn from Data Knowledge]

(credit: Vivek S Kumar)

→ Constraints: One of the ways of representing symbolic knowledge.
  \( \mathbb{1}\{y_{PER.} = 1\} \implies \mathbb{1}\{y_{Noun.} = 1\} \)

→ Limited work in training DL models with (soft) constraints
→ What if constraints are hard?
Neural + Constraints

- Augmenting deep neural models (DNN) with Domain Knowledge (DK)

- Domain Knowledge expressed in the form of *Constraints* (C)

  ➢ Learning with (hard) constraints: Learn DNN weights s.t. output satisfies constraints C
Related Work
Related Work

**Inference**
- Gradient based inference (Lee et al. [’19])
- Neural+CRF as post processing (Chen et al [’18])

**Training**
- Semantic loss (Xu et al. [’18])
- Semi-supervised SRL (Mehta et al. [’18])
- Posterior Regularization + Distillation (Hu et al. [’16])

**Constraints**
- **Soft**
  - CCM (Roth & Yih [2005], Chang et al. [2013])
  - Dual Decomposition (Rush & Collins [2012])
- **Hard**

**Our Work**
Learning with Constraints: *Running Example*

- **Task:** Fine Grained Entity Typing
Learning with Constraints: *Running Example*

**Input:**

**Sample Mention:** "Barack Obama is the President of the United States"

**Output:**

*president, leader, politician...*
Learning with Constraints: *Running Example*

**Input:**

Bag of Mentions

**Sample Mention:**

“Barack Obama is the President of the United States”

**Output:**

president, leader, politician...

```
Mention 1
Mention 2
Mention N

Neural Network

president ✓
leader ✓
politician ✓
sportsman ✗
```
Learning with Constraints: Running Example

- **Constraints:** Hierarchy on Output label space
Learning with Constraints: Running Example

- **Constraints:** Hierarchy on Output label space
Learning with Constraints:

Running Example

- Person
- Lawyer
- Artist
- Musician
- Actor
- Doctor

Constraints:

Hierarchy on Output label space

Source:

https://github.com/iesl/TypeNet
https://github.com/MurtyShikhar/Hierarchical-Typing
Learning with Constraints: *Representation of Constraints*

→ Using Soft Logic

\[ \mathbb{1} \{y_{ARTIST} = 1\} \implies \mathbb{1} \{y_{PERSON} = 1\} \]
Learning with Constraints: *Representation of Constraints*

→ Using Soft Logic

\[ 1 \{y_{ARTIST} = 1\} \implies 1 \{y_{PERSON} = 1\} \]

\[ (\neg 1 \{y_{ARTIST} = 1\}) \lor (1 \{y_{PERSON} = 1\}) \]
Learning with Constraints: *Representation of Constraints*

→ Using Soft Logic

\[ 1 \{ y_{\text{ARTIST}} = 1 \} \implies 1 \{ y_{\text{PERSON}} = 1 \} \]

\[ (\neg 1 \{ y_{\text{ARTIST}} = 1 \}) \lor (1 \{ y_{\text{PERSON}} = 1 \}) \]

\[ (1 - p(y_{\text{ARTIST}})) + p(y_{\text{PERSON}}) \]
<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>T-norm: Choice 1</th>
<th>T-norm: Choice 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(v)</td>
<td>(p(v = 1))</td>
<td></td>
</tr>
<tr>
<td>(\neg v)</td>
<td>(1 - p(v = 1))</td>
<td></td>
</tr>
<tr>
<td>(v_1 \lor v_2)</td>
<td>(\min(p(v_1 = 1) + p(v_2 = 1), 1))</td>
<td>(\max(p(v_1 = 1), p(v_2 = 1)))</td>
</tr>
<tr>
<td>(v_1 \land v_2)</td>
<td>(\max(p(v_1 = 1) + p(v_2 = 1) - 1, 0))</td>
<td>(\min(p(v_1 = 1), p(v_2 = 1)))</td>
</tr>
</tbody>
</table>

\[
\mathbb{1} \{y_{ARTIST} = 1\} \implies \mathbb{1} \{y_{PERSON} = 1\}
\]

\[
\neg \mathbb{1} \{y_{ARTIST} = 1\} \lor \mathbb{1} \{y_{PERSON} = 1\}
\]

\[
(1 - p(y_{ARTIST})) + p(y_{PERSON})
\]
Learning with Constraints: \textit{Representation of Constraints}

\[ 1 - p(y_{ARTIST}) + p(y_{PERSON}) = 1 \]
Learning with Constraints: *Representation of Constraints*

\[ 1 - p(y_{\text{ARTIST}}) + p(y_{\text{PERSON}}) = 1 \]

\[ 1 - p(y_{\text{ARTIST}}) + p(y_{\text{PERSON}}) \geq 1 \]
Learning with Constraints: *Representation of Constraints*

\[ 1 - p(y_{ARTIST}) + p(y_{PERSON}) = 1 \]

\[ 1 - p(y_{ARTIST}) + p(y_{PERSON}) \geq 1 \]

Equivalently:

\[ p(y_{ARTIST}) - p(y_{PERSON}) \leq 0 \]
Learning with Constraints: Representation of Constraints

Define:

\[ f_k^i = p(y_{ARTIST}) - p(y_{PERSON}) \]

Inequality Constraint:

\[ f_k^i \leq 0 \]
Learning with Constraints: Formulation

Unconstrained Problem

$$\min_w \ L(w)$$

$L(w)$ : Any standard loss function, say Cross Entropy
Learning with Constraints: Formulation

Unconstrained Problem

\[
\min_{\omega} L(\omega)
\]

\(L(\omega)\): Any standard loss function, say Cross Entropy

Constrained Problem

\[
\min_{\omega} L(\omega) \quad \text{subject to} \quad f^i_k(\omega) \leq 0; \quad \forall 1 \leq i \leq m; \quad \forall 1 \leq k \leq K
\]
Learning with Constraints: *Formulation*

**Constrained Problem**

\[
\min_w L(w) \quad \text{subject to} \quad f^i_k(w) \leq 0; \quad \forall 1 \leq i \leq m; \quad \forall 1 \leq k \leq K
\]

*Where:*

\[m: \text{Size of training data}\]

\[K: \text{Number of Constraints}\]
Learning with Constraints: *Formulation*

Constrained Problem

\[
\min_{\mathbf{w}} L(\mathbf{w}) \quad \text{subject to} \quad f_i(\mathbf{w}) \leq 0, \quad 1 \leq i \leq m, \quad 1 \leq k \leq K
\]

\[
\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) = L(\mathbf{w}) + \sum_{i=1}^{m} \sum_{k=1}^{K} \lambda^i_k f_k^i(\mathbf{w})
\]
Learning with Constraints: Formulation

Constrained Problem

\[
\min_{\mathbf{w}} L(w) \quad \text{subject to} \quad f_i(w) \leq 0, \quad \forall i = 1, \ldots, m, \quad \forall k = 1, \ldots, K
\]

\[
\mathcal{L}(w, \Lambda) = L(w) + \sum_{i=1}^{m} \sum_{k=1}^{K} \lambda_k^i f_k^i(w)
\]

\[
\min_{w} \max_{\Lambda} \mathcal{L}(w, \Lambda) \quad \geq \quad \max_{\Lambda} \min_{w} \mathcal{L}(w, \Lambda)
\]
Learning with Constraints: Formulation

Constrained Problem

$$\min_w L(w) \quad \text{subject to} \quad f_k^i(w) \leq 0; \quad \forall 1 \leq i \leq m; \quad \forall 1 \leq k \leq K$$

Where:

- $m$: Size of training data
- $K$: Number of Constraints

Issue:

$O(mK)$ #constraints

i.e. $mK$ Lagrange Multipliers!
Learning with Constraints: Reduce # Constraints

\[ H(c) = \begin{cases} c & \text{for } c \geq 0, \\ 0 & \text{for } c < 0 \end{cases} \]
Learning with Constraints: Reduce # Constraints

\[ H(c) = \begin{cases} 
  c & \text{for } c \geq 0, \\
  0 & \text{for } c < 0 
\end{cases} \]

\[ f^i_k(w) \leq 0 \quad \equiv \quad H(f^i_k(w)) = 0 \]

Equivalent
Learning with Constraints: \textit{Reduce \# Constraints}

\[
H(c) = \begin{cases} 
  c & \text{for } c \geq 0, \\
  0 & \text{for } c < 0 
\end{cases}
\]

\[
f^i_k(w) \leq 0 \quad \equiv \quad H(f^i_k(w)) = 0
\]

Equivalent

\[
\forall i : H(f^i_k(w)) = 0 \quad \equiv \quad \sum_i H(f^i_k(w)) = 0
\]
Learning with Constraints: *Reduce # Constraints*

Originally:

$$\min_w L(w) \quad \text{subject to} \quad f_k^i(w) \leq 0; \quad \forall 1 \leq i \leq m; \quad \forall 1 \leq k \leq K$$
Learning with Constraints: \textit{Reduce \# Constraints}

Originally:

$$\min_{w} L(w) \text{ subject to } f_k^i(w) \leq 0; \ \forall 1 \leq i \leq m; \ \forall 1 \leq k \leq K$$

Now:

Define: \( h_k(w) = \sum_i H(f_k^i(w)) \)

$$\min_{w} L(w) \text{ subject to } h_k(w) = 0; \ \forall 1 \leq k \leq K$$
Learning with Constraints: Reduce # Constraints

Originally:

\[
\min_w L(w) \text{ subject to } f_k^i(w) \leq 0; \quad \forall 1 \leq i \leq m; \quad \forall 1 \leq k \leq K
\]

Now:

Define: \( h_k(w) = \sum_i H(f_k^i(w)) \) \( \quad \text{O(K) #constraints} \)

\[
\min_w L(w) \text{ subject to } h_k(w) = 0; \quad \forall 1 \leq k \leq K
\]
Learning with Constraints: *Primal-Dual Formulation*

\[
\min_{w} L(w) \quad \text{subject to} \quad h_k(w) = 0; \quad \forall 1 \leq k \leq K
\]

**Lagrangian**

\[
\mathcal{L}(w; \Lambda) = L(w) + \sum_{k=1}^{K} \lambda_k h_k(w)
\]
Learning with Constraints: *Primal-Dual Formulation*

\[
\min_w L(w) \quad \text{subject to} \quad h_k(w) = 0; \quad \forall 1 \leq k \leq K
\]

**Lagrangian**

\[
\mathcal{L}(w; \Lambda) = L(w) + \sum_{k=1}^{K} \lambda_k h_k(w)
\]

**Primal**

\[
\min_w \max_{\Lambda} \mathcal{L}(w, \Lambda) \geq \max_{\Lambda} \min_w \mathcal{L}(w, \Lambda)
\]
Learning with Constraints: Parameter Update

\[ x \downarrow \quad w \quad \rightarrow \quad y_p(w) \quad \downarrow \quad y_g \quad \rightarrow \quad L(w) \]

\[ L(w) \]

\( w \) Update
Learning with Constraints: Parameter Update

\[ C(w, \Lambda) = \sum_{k=1}^{K} \lambda_k h_k(w) \]
Learning with Constraints: Parameter Update

\[ y_g = y_p(w) \]

\[ L(w) = \mathcal{L}(w, \Lambda) = L(w) + C(w, \Lambda) \]

\[ \min_w \mathcal{L}(w, \Lambda) \]

\[ C(w, \Lambda) = \sum_{k=1}^{K} \lambda_k h_k(w) \]

\[ w \text{ Update} \]

\[ \Lambda \text{ Fixed} \]
Learning with Constraints: Parameter Update

\[ L(w) \]

\[ \mathcal{L}(w, \Lambda) = L(w) + C(w, \Lambda) \]

\[ \max_{\Lambda} \min_{w} \mathcal{L}(w, \Lambda) \]

\[ C(w, \Lambda) = \sum_{k=1}^{K} \lambda_k h_k(w) \]
Learning with Constraints: *Parameter Update*

\[ C(w, \Lambda) = \sum_{k=1}^{K} \lambda_k h_k(w) \]

\( x \)

\[ \downarrow \]

\[ \text{NN} \]

\[ w \]

\[ \uparrow \]

\[ y_p(w) \]

\[ \text{CV} \]

\[ \Lambda \]

\( w \) Fixed

\( \Lambda \) Update
Learning with Constraints: Training Algorithm

Start
\( \Lambda = 0 \)

\( w \) Update

warmup
Learning with Constraints: *Training Algorithm*

- **Start** \( \Lambda = 0 \)
- **\( w \) Update**
- **\( \Lambda \) Update**
- **Warmup**
Learning with Constraints: \textit{Training Algorithm}

\begin{itemize}
\item \textbf{Start} \quad \Lambda = 0
\item \textbf{\(\Lambda\) Update}
\item \textbf{\(w\) Update}
\end{itemize}

\textbf{warmup} \quad \textbf{\(l\) iterations}
Learning with Constraints: \textit{Training Algorithm}
Learning with Constraints: *Training Algorithm*

- **Start**: \( \Lambda = 0 \)
- **\( w \) Update**
- **\( \Lambda \) Update**
- **Increment \( l \)**
  - Adjust \( \eta \)

Crucial for convergence guarantees!
Learning with Constraints: *Experiments* Typenet

<table>
<thead>
<tr>
<th>Scenario</th>
<th>MAP Scores</th>
<th>Constraint Violations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5% Data</td>
<td>10% Data</td>
</tr>
<tr>
<td>B</td>
<td>68.6</td>
<td></td>
</tr>
<tr>
<td>B+H</td>
<td>68.71</td>
<td></td>
</tr>
<tr>
<td>B+C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B+S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Learning with Constraints: *Experiments*  
Typenet

<table>
<thead>
<tr>
<th>Scenario</th>
<th>MAP Scores</th>
<th></th>
<th>Constraint Violations</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5% Data</td>
<td>10% Data</td>
<td>100% Data</td>
<td>5% Data</td>
</tr>
<tr>
<td>B</td>
<td>68.6</td>
<td></td>
<td></td>
<td>22,715</td>
</tr>
<tr>
<td>B+H</td>
<td>68.71</td>
<td></td>
<td></td>
<td>22,928</td>
</tr>
<tr>
<td>B+C</td>
<td>80.13</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>B+S</td>
<td>82.22</td>
<td></td>
<td></td>
<td>41</td>
</tr>
</tbody>
</table>
Learning with Constraints: *Experiments*  
Typenet

<table>
<thead>
<tr>
<th>Scenario</th>
<th>MAP Scores</th>
<th></th>
<th></th>
<th>Constraint Violations</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5% Data</td>
<td>10% Data</td>
<td>100% Data</td>
<td>5% Data</td>
<td>10% Data</td>
<td>100% Data</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>68.6</td>
<td>69.2</td>
<td>70.5</td>
<td>22,715</td>
<td>21,451</td>
<td>22,359</td>
</tr>
<tr>
<td><strong>B+H</strong></td>
<td>68.71</td>
<td>69.31</td>
<td>71.77</td>
<td>22,928</td>
<td>21,157</td>
<td>24,650</td>
</tr>
<tr>
<td><strong>B+C</strong></td>
<td>80.13</td>
<td>81.36</td>
<td>82.80</td>
<td>25</td>
<td>45</td>
<td>12</td>
</tr>
<tr>
<td><strong>B+S</strong></td>
<td>82.22</td>
<td>83.81</td>
<td></td>
<td>41</td>
<td>26</td>
<td></td>
</tr>
</tbody>
</table>


Learning with Constraints: *Experiments*

**NER**

**Task:** Named Entity Recognition

**Auxiliary Task:** Part of Speech Tagging
Learning with Constraints: *Experiments*

**NER**

**Task:** Named Entity Recognition

**Auxiliary Task:** Part of Speech Tagging

**Architecture:** Common LSTM encoder and task specific classifier
Learning with Constraints: Experiments

**NER**

**Task:** Named Entity Recognition

**Auxiliary Task:** Part of Speech Tagging

**Architecture:** Common LSTM encoder and task specific classifier

**Constraints:** 16 constraints of type: *Person* => *Noun*
Learning with Constraints: *Experiments*  
NER

(a) Avg. Gain in F1 Score Over Baseline.  
(b) Avg. number of Constrained Violations
Learning with Constraints: *Experiments*

**Task:** Semantic Role Labelling

**Auxiliary Info:** Syntactic Parse Trees
Learning with Constraints: Experiments
SRL

• For each clause, determine the semantic role played by each noun phrase that is an argument to the verb.
  
  agent  patient  source  destination  instrument  

  – John drove Mary from Austin to Dallas in his Toyota Prius.  
  – The hammer broke the window.

• Also referred to a “case role analysis,” “thematic analysis,” and “shallow semantic parsing”

Slide Credit: Ray Mooney
Learning with Constraints: Experiments

**SRL**

**Task:** Semantic Role Labelling

**Auxiliary Info:** Syntactic Parse Trees

**Architecture:** State-of-the-art based on ELMo embeddings
Learning with Constraints: *Experiments*

**SRL**

**Task:** Semantic Role Labelling

**Auxiliary Info:** Syntactic Parse Trees

**Architecture:** State-of-the-art based on ELMo embeddings

**Constraints:** Transition Constraints & span constraints
Learning with Constraints: Experiments

SRL

Constraints:

Transition Constraints

\[ \text{e.g. } \text{B-Arg}(i) \Rightarrow \text{I-Arg}(i+1) \]

Span Constraints: Semantic spans should be

subset of syntactic spans
Learning with Constraints: *Experiments*

**SRL:** Syntactic Parse Tree for span constraints

```
 NP_{sg}  
  |     |  
 Det  N  PP  
    The man Prep 

 VP_{sg}  
    ate the apple.  

 NP_{pl}  
    by  
      the store near the dog
```

“The man by the store near the dog ate an apple.”

“The man” is the agent of “ate” not “the dog”.

*Slide Credit: Ray Mooney*
# Learning with Constraints: Experiments

## SRL

<table>
<thead>
<tr>
<th>Scenario</th>
<th>1% Data</th>
<th>5% Data</th>
<th>10% Data</th>
<th>Total Constraint Violations</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>62.99</td>
<td></td>
<td></td>
<td>14,857</td>
</tr>
<tr>
<td>CL</td>
<td>66.21</td>
<td></td>
<td></td>
<td>9,406</td>
</tr>
<tr>
<td>B+CI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CL + CI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Learning with Constraints: Experiments

## SRL

<table>
<thead>
<tr>
<th>Scenario</th>
<th>1% Data</th>
<th>5% Data</th>
<th>10% Data</th>
<th>1% Data</th>
<th>5% Data</th>
<th>10% Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>62.99</td>
<td>72.64</td>
<td>76.04</td>
<td>14,857</td>
<td>9,708</td>
<td>7,704</td>
</tr>
<tr>
<td>CL</td>
<td>66.21</td>
<td>74.27</td>
<td>77.19</td>
<td>9,406</td>
<td>7,461</td>
<td>5,836</td>
</tr>
<tr>
<td>B+CI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CL + CI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Learning with Constraints: *Experiments*

### SRL

<table>
<thead>
<tr>
<th>Scenario</th>
<th>1% Data</th>
<th>5% Data</th>
<th>10% Data</th>
<th>1% Data</th>
<th>5% Data</th>
<th>10% Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>62.99</td>
<td>72.64</td>
<td>76.04</td>
<td>14,857</td>
<td>9,708</td>
<td>7,704</td>
</tr>
<tr>
<td>CL</td>
<td>66.21</td>
<td>74.27</td>
<td>77.19</td>
<td>9,406</td>
<td>7,461</td>
<td>5,836</td>
</tr>
<tr>
<td>B+CI</td>
<td>67.9</td>
<td>75.96</td>
<td>78.63</td>
<td>5,737</td>
<td>4,247</td>
<td>3,654</td>
</tr>
<tr>
<td>CL + CI</td>
<td>68.71</td>
<td>76.51</td>
<td>78.72</td>
<td>5,039</td>
<td>3,963</td>
<td>3,476</td>
</tr>
</tbody>
</table>
Reviews

Doubt

1. Why constraint violations even though they are hard.
Reviews

Weakness

1. Design of constrain function requires significant background knowledge about the task. [Jigyasa]

2. I think we cannot model constraints that are dependent on surrounding generated text. Like a sorting task, with unknown no. of numbers. Generated sequence should have $t_i < t_j$ if $i < j$. 
1. **Other Domains:** robotics (physical constraints like reachability, physical properties of objects etc).

2. **Learning Constraints:** Latent representation over the space of logical symbols to fill 3 slots like $A \rightarrow B$. Now, whatever this latent representation is suggesting as a constraint, take that as a hard constraint over the next epoch. This can be extended to have a fixed number of constraints in the model. This would be like learning constraints from the given sample of data, whether that is good or bad, I am not sure because a dataset usually consists of biases in various forms.
References

2. C. Jin, P. Netrapalli, & M. I. Jordan. Minmax optimization: Stable limit points of gradient descent ascent are locally optimal, arxiv 2019
4. S. Murty, P. Verga, L. Vilnis, I. Radovanovic, and A. McCallum. Hierarchical losses and new resources for fine-grained entity typing and linking, ACL 2018
5. J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. ICML 2018
Thank You!