Cooperative Learning of Disjoint Syntax and Semantics

Serhii Havrylov

Germán Kruszewski
Armand Joulin
Is using linguistic structures for sentence modelling useful? (e.g. syntactic trees)
Is using linguistic structures for sentence modelling useful? (e.g. syntactic trees)

Yes, it is! Let’s create more treebanks!
Is using linguistic structures for sentence modelling useful? (e.g. syntactic trees)

Yes, it is! Let’s create more treebanks!

No! Annotations are expensive to make. Parse trees is just a linguists’ social construct. Just stack more layers and you will be fine!
Recursive neural network

The cat sat on the mat
Recursive neural network

$x$

The cat sat on the mat
Recursive neural network

$((x, t))$

The cat sat on the mat
Recursive neural network

\[ f_\theta(x, t) \]
Recursive neural network

\[(y, f_\theta(x, t))\]

neutral

The cat sat on the mat
Recursive neural network

\[ \ell(y, f_\theta(x, t)) \]

neutral

The cat sat on the mat
Latent tree learning

\[ \ell(y, f_\theta(x, t)) \]
Latent tree learning

\[ \mathbb{E}_{p_{\phi}(t|x)}[\ell(y, f_{\theta}(x, t))] \]
Latent tree learning

$$\mathbb{E}_{p_\phi(t|x)}[\ell(y, f_\theta(x, t))]$$
Latent tree learning

\[ \mathbb{E}_{p(\phi)}(t|x) \left[ \ell(y, f(x, t)) \right] \]
Latent tree learning

\[ \mathbb{E}_{p_{\phi}(t|x)}[\ell(y, f_{\theta}(x, t))] \]
Latent tree learning

- RL-SPINN: Yogatama et al., 2016
- Soft-CYK: Maillard et al., 2017
- Gumbel Tree-LSTM: Choi et al., 2018
Latent tree learning

- RL-SPINN: Yogatama et al., 2016
- Soft-CYK: Maillard et al., 2017
- Gumbel Tree-LSTM: Choi et al., 2018

Recent work has shown that:
- Trees **do not resemble** any semantic or syntactic formalisms (Williams et al. 2018).
Latent tree learning

- RL-SPINN: Yogatama et al., 2016
- Soft-CYK: Maillard et al., 2017
- Gumbel Tree-LSTM: Choi et al., 2018

Recent work has shown that:
- Trees **do not resemble** any semantic or syntactic formalism (Williams et al. 2018).
- Parsing strategies **are not consistent** across random restarts (Williams et al. 2018).
Latent tree learning

- RL-SPINN: Yogatama et al., 2016
- Soft-CYK: Maillard et al., 2017
- Gumbel Tree-LSTM: Choi et al., 2018

Recent work has shown that:
- Trees **do not resemble** any semantic or syntactic formalisms (Williams et al. 2018).
- Parsing strategies **are not consistent** across random restarts (Williams et al. 2018).
- These models **fail to learn the simple context-free grammar** (Nangia et al. 2018).
ListOps (Nangia, & Bowman (2018))

[MIN 1 [MAX [MIN 9 [MAX 10] 2 9 [MED 8 4 3]] [MIN 7 5] 6 9 3]]
[MIN 7 5] [MIN 7 5]
[MIN 7 1 [MAX 6 8 17] [MIN 2 6] 3]
ListOps (Nangia, & Bowman (2018))

\[ \text{MIN 1 [MAX [MIN 9 [MAX 1 0 ] 2 9 [MED 8 4 3 ] ] [MIN 7 5 ] 6 9 3 ] ]} \]
\[ \text{MAX 1 4 0 9 } \]
\[ \text{MAX 7 1 [MAX 6 8 1 7 ] [MIN 2 6 ] 3 ]} \]
ListOps (Nangia, & Bowman (2018))
Tree-LSTM parser (Choi et al., 2018)
Tree-LSTM parser (Choi et al., 2018)

\[ r_{i+1}^k = \text{Tree-LSTM}(r_{i}^k, r_{i+1}^k) \]
Tree-LSTM parser \cite{choi-2018-tree}

\[
s_k(i) = \langle q, r_{i+1}^{k+1} \rangle
\]

\[
r_{i}^{k+1} = \text{Tree-LSTM}(r_{i}^{k}, r_{i+1}^{k})
\]
Tree-LSTM parser (Choi et al., 2018)

\[ s_k(i) = \langle q, r_{i+1}^k \rangle \]

\[ r_{i+1}^k = \text{Tree-LSTM}(r_i^k, r_{i+1}^k) \]
Tree-LSTM parser (Choi et al., 2018)
Tree-LSTM parser (Choi et al., 2018)

\[ \mathbf{r}_{i}^{k+1} = \text{Tree-LSTM}(\mathbf{r}_{i}^{k}, \mathbf{r}_{i+1}^{k}) \]
Tree-LSTM parser (Choi et al., 2018)

\[ s_k(i) = \langle q, r_i^{k+1} \rangle \]

\[ r_i^{k+1} = \text{Tree-LSTM}(r_i^k, r_{i+1}^k) \]
Tree-LSTM parser (Choi et al., 2018)

\[ s_k(i) = \langle q, r_i^{k+1} \rangle \]

\[ r_i^{k+1} = \text{Tree-LSTM}(r_i^k, r_{i+1}^k) \]
Tree-LSTM parser (Choi et al., 2018)
Tree-LSTM parser (Choi et al., 2018)

\[ r_{i}^{k+1} = \text{Tree-LSTM}(r_{i}^{k}, r_{i+1}^{k}) \]
Tree-LSTM parser (Choi et al., 2018)

\[ s_k(i) = \langle q, r_{i}^{k+1} \rangle \]

\[ r_{i}^{k+1} = \text{Tree-LSTM}(r_{i}^{k}, r_{i+1}^{k}) \]
Tree-LSTM parser (Choi et al., 2018)

\[ s_k(i) = \langle q, r_{i}^{k+1} \rangle \]

\[ r_{i}^{k+1} = \text{Tree-LSTM}(r_{i}^{k}, r_{i+1}^{k}) \]
Tree-LSTM parser (Choi et al., 2018)
Separation of syntax and semantics

Parser $\phi$

$\text{}$

$s_k(i) = \langle q, \text{Tree-LSTM}(r^k_i, r^k_{i+1}) \rangle$

Compositional Function $\theta$

$\text{}$

$r^{k+1}_i = \text{Tree-LSTM}(r^k_i, r^k_{i+1})$
Parsing as a RL problem

Parser $\phi$

$$p_\phi(t|x) = \prod_{k=0}^{K} \pi_\phi(a^i_k | r^k)$$

Compositional Function $\theta$

$$\ell(f_\theta(x, t), y)$$
Optimization challenges

Size of the search space is

\[ C_n \sim \frac{4^n}{n^{3/2} \sqrt{\pi}} \]
Optimization challenges

Size of the search space is

\[ C_n \sim \frac{4^n}{n^{3/2} \sqrt{\pi}} \]

For a sentence with 20 words, there are \(1,767,263,190\) possible trees.
Optimization challenges

Syntax and semantic has to be learnt simultaneously
model has to infer from examples that \([\text{MIN } 0\ 1] = 0\)
Optimization challenges

Syntax and semantic has to be learnt simultaneously

model has to infer from examples that \([\text{MIN} \ 0 \ 1] = 0\)

– nonstationary environment (i.e. the same sequence of actions can receive different rewards)
Optimization challenges

Typically, the *compositional function* \( \theta \) is learned faster than the *parser* \( \varphi \).
Optimization challenges

Typically, the \textit{compositional function} $\theta$ is learned faster than the \textit{parser} $\phi$.

This fast coadaptation limits the exploration of the search space to parsing strategies similar to those found at the beginning of the training.
 Optimization challenges

- High variance in the estimate of a parser’s gradient $\nabla \varphi$ has to be addressed.

- Learning paces of a parser $\theta$ and a compositional function $\varphi$ have to be levelled off.
Variance reduction

$$\nabla_\phi \mathcal{L} \approx \ell(f_\theta(x, t), y) \frac{\partial \log p_\phi(t|x)}{\partial \phi}$$
Variance reduction

\[ \nabla_\phi \mathcal{L} \approx \ell(f_\theta(x, t), y) \left( \frac{\partial \log p_\phi(t|x)}{\partial \phi} \right) \]
Variance reduction

\[ \nabla_{\phi} \mathcal{L} \approx \ell(f_\theta(x, t), y) \frac{\partial \log p_\phi(t|x)}{\partial \phi} \]

reward

Is this a carrot?
Variance reduction

$$\nabla_{\phi} \mathcal{L} \approx \ell(f_{\theta}(x, t), y) \frac{\partial \log p_{\phi}(t|x)}{\partial \phi}$$

the moving average of recent rewards

$$\nabla_{\phi} \mathcal{L} \approx (\ell(f_{\theta}(x, t), y) - c) \frac{\partial \log p_{\phi}(t|x)}{\partial \phi}$$

new reward
Variance reduction

- \([\text{MIN } 1 \ [\text{MAX } [\text{MIN } 9 \ [\text{MIN } 1 \ 0 \ ] \ 2 \ [\text{MED } 8 \ 4 \ 3 \ ] \ ] \ [\text{MAX } 7 \ 5 \ ] \ 6 \ 9 \ ] \ ] \]
- \([\text{MAX } 1 \ 0 \ ]\)
Variance reduction

- \[ \text{MIN 1 [MAX [MIN 9 [MIN 1 0 ] 2 [MED 8 4 3 ] ] [MAX 7 5 ] 6 9 ] ]} \]
- \[ \text{MAX 1 0 } \]

\[ \nabla_\phi \mathcal{L} \approx (\ell(f_\theta(x, t), y) - c(x)) \frac{\partial \log p_\phi(t|x)}{\partial \phi} \]
Variance reduction

\[ \nabla_{\phi} \mathcal{L} \approx \ell(f_{\theta}(x, t), y) - c(x) \frac{\partial \log p_{\phi}(t|x)}{\partial \phi} \]
Variance reduction

- $[\text{MIN } 1 \ [\text{MAX } [\text{MIN } 9 \ [\text{MIN } 1 \ 0 \ ] \ 2 \ [\text{MED } 8 \ 4 \ 3 \ ] \ ] \ [\text{MAX } 7 \ 5 \ ] \ 6 \ 9 \ ]]$
- $[\text{MAX } 1 \ 0 \ ]$

\[
\nabla_{\phi} \mathcal{L} \approx (\ell(f_{\theta}(x, t), y) - c(x)) \frac{\partial \log p_{\phi}(t|x)}{\partial \phi}
\]

self-critical training (SCT) baseline Rennie et al. (2017)

\[
c(x) = \ell(f_{\theta}(x, \hat{t}), y)
\]

\[
\hat{t} = \arg \max p_{\phi}(t|x)
\]
Synchronizing syntax and semantics learning

Syntax \( p_\phi(t|x) \)  

Semantics \( f_\theta(x, t) \)
Synchronizing syntax and semantics learning

Parameter space

\[ \phi - \nabla \phi \]

\[ \phi \]

Distribution space

\[ \hat{p}_\phi(. \mid x) \]

\[ p_{\phi - \nabla \phi}(. \mid x) \]

\[ p_\phi(. \mid x) \]
Synchronizing syntax and semantics learning

Parameter space

\[ \phi - \nabla \phi \]

Distribution space

\[ p_{\hat{\phi}}(. | x) \quad p_{\phi - \nabla \phi}(. | x) \]

\[ \frac{p_{\phi}(t | x)}{p_{\phi_{\text{old}}}(t | x)} \in [1 - \epsilon; 1 + \epsilon] \]
Synchronizing syntax and semantics learning

Proximal Policy Optimization (PPO) of Schulman et al. (2017)
Optimization challenges

● High variance in the estimate of a parser’s gradient $\nabla \varphi$ is addressed by using **self-critical training** (SCT) baseline of Rennie et al. (2017).

● Learning paces of a parser $\varphi$ and a compositional function $\theta$ is levelled off by controlling parser’s updates using **Proximal Policy Optimization** (PPO) of Schulman et al. (2017).
ListOps results

[\text{MAX 29} \text{ [MIN 47]} 0]
ListOps results

- SCT/PPO: 99.2%
- SCT/no PPO: 64.0%
- MA/ PPO: 67.5%

[Max 29 [Min 47] 0]
ListOps results

- SCT/PPO: 99.2%
- SCT/no PPO: 64.0%
- MA/ PPO: 67.5%
- RL-SPINN: 60.7%

[MAX 29 [MIN 47] 0]
Sentiment Analysis (SST-2)

88.0%
Sentiment Analysis (SST-2)

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree-LSTM</td>
<td>88.0%</td>
</tr>
<tr>
<td>RL-SPINN</td>
<td>86.5%</td>
</tr>
<tr>
<td>ST-Gumbel</td>
<td>90.3%</td>
</tr>
<tr>
<td>Ours</td>
<td>90.2%</td>
</tr>
</tbody>
</table>
Natural language inference (MultiNLI)

- LSTM: 69.1%
- RL-SPINN: 67.4%
- ST-Gumbel: 69.5%
- Ours: 70.7%
<table>
<thead>
<tr>
<th>Method</th>
<th>Time complexity</th>
<th>Space complexity</th>
<th>ListOps</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL-SPINN: Yogatama et al., 2016</td>
<td>(O(n d^2))</td>
<td>(O(n d^2))</td>
<td>✗</td>
</tr>
<tr>
<td>Soft-CYK: Maillard et al., 2017</td>
<td>(O(n^3d + n^2d^2))</td>
<td>(O(n^3d))</td>
<td>✗</td>
</tr>
<tr>
<td>Gumbel Tree-LSTM: Choi et al., 2018</td>
<td>(O(n^2d + nd^2))</td>
<td>(O(n^2d))</td>
<td>✗</td>
</tr>
<tr>
<td>Ours</td>
<td>(O(Knd^2))</td>
<td>(O(nd^2))</td>
<td>✓</td>
</tr>
</tbody>
</table>

\(n\) – sentence length  
\(d\) – tree-LSTM dimensionality  
\(K\) – number of updates in PPO
Conclusions

- The *separation* between syntax and semantics allows *coordination* between optimisation schemes for each module.
- Self-critical training *mitigates credit assignment* problem by *distinguishing* “hard” and “easy” to solve datapoints.
- The model *can recover* a simple context-free grammar of mathematical expressions.
- The model *performs competitively* on several real natural language tasks.

[github.com/facebookresearch/latent-treelstm](https://github.com/facebookresearch/latent-treelstm)