Parameter Prediction Paper: Decoupled Neural Interfaces Using Synthetic Gradients

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver Koray Kavukcuoglu

Reviewed by: Arshdeep Sekhon

1Department of Computer Science, University of Virginia
https://qdata.github.io/deep2Read/

August 25, 2018
Locking in Neural Networks

To update Layer 1:

1. Forward Propagation through Layer 2 and Layer 3
2. Backward Propagation through Layer 2 and Layer 3
Why is locking a problem?

1. Updates in sequential and synchronous manner
2. A distributed system: Updates depend on the slowest part
3. parallelizing training of neural network modules can speed up training.
1. Layer 1 will be updated before Layer 2 and Layer 3 have even been executed.

2. No longer locked to the rest of the network.
1. Decoupled Neural Interfaces predict gradients: synthetic gradients from previous layer outputs or activations.
2. do not rely on backpropagation to get error gradients.
A network with $N$ layers $f_i, i \in \{1, \cdots, N\}$

For the $i_{th}$ layer, input $h_{i-1}$, output $h_i = f_i(h_{i-1})$

The complete graph is represented by $F^N_1$
Gradients for FeedForward Networks

\[ \theta_i \leftarrow \theta_i - \alpha \delta_i \frac{\delta h_i}{\delta \theta_i}; \quad \delta_i = \frac{\delta L}{\delta h_i} \]  

(1)
\[ \theta_n \leftarrow \theta_n - \alpha \hat{\delta}_i \frac{\delta h_i}{\delta \theta_n}; \quad \delta_i = M_{i+1}(h_i) \]
Synthetic Gradients for FeedForward Networks

\[ \theta_n \leftarrow \theta_n - \alpha \hat{\delta}_i \frac{\delta h_i}{\delta \theta_n}; \quad \delta_i = M_{i+1}(h_i) \]  

1. \[ \theta_n \leftarrow \theta_n - \alpha \hat{\delta}_i \frac{\delta h_i}{\delta \theta_n}; \quad \delta_i = M_{i+1}(h_i) \]  

2. \( n \in \{1, \cdots, n\} \)

3. To train \( M_{i+1}(h_i) \),

4. wait for true error gradient to be computed

5. after a full forwards and backwards pass of \( F_N^{i+1} \)

6. Minimize \( \|\hat{\delta}_i - \delta_i\|_2^2 \)
use backpropagated $\hat{\delta}_{i+1}$ instead of the true gradients
Synthetic Gradients for RNNs

The task: stream prediction; possibly infinite

![Diagram of unrolled RNN]

Unrolling the recurrent network:
Forward Graph: $F^\infty_1$ made up of $f_i$ where $i$ varies from 1 to $\infty$
At a particular point in time $t$, minimise Loss over the next steps

$$\inf_{\tau=t} \sum L_{\tau}$$ (4)

$$\theta \leftarrow \theta - \alpha \sum_{\tau=t}^{\infty} \frac{\delta L_{\tau}}{\delta \theta}$$ (5)
Truncated Backpropagation

At a particular point in time $t$, minimise Loss over the next steps

$$\theta \leftarrow \theta - \alpha \left( \sum_{\tau=t}^{T} \frac{\delta L_{\tau}}{\delta \theta} + \left( \inf_{\tau=T+1} \sum_{\tau=T+1}^{\infty} \frac{\delta L_{\tau}}{\delta h_T} \right) \frac{\delta h_T}{\delta \theta} \right)$$  \hspace{1cm} (6)

$$\theta \leftarrow \theta - \alpha \left( \sum_{\tau=t}^{T} \frac{\delta L_{\tau}}{\delta \theta} + \left( \delta_T \right) \frac{\delta h_T}{\delta \theta} \right)$$  \hspace{1cm} (7)

truncated BPTT: $\delta_T = 0$; limits temporal dependency learnt by rnn
Truncated Backpropagation

\[ \theta \leftarrow \theta - \alpha \left( \sum_{\tau=t}^{T} \frac{\delta L_\tau}{\delta \theta} + (\hat{\delta}_T) \frac{\delta h_T}{\delta \theta} \right) \] (8)

\( \hat{\delta}_T = M_T(h_T) \); learned approximation of the future loss gradients
divide unrolled rnn into subnetworks of length T
insert a DNI between \( F_{t+T-1} \) and \( F_{t+2T-1} \)
Truncation Backpropagation

\[ \theta \leftarrow \theta - \alpha \left( \sum_{\tau=t}^{T} \frac{\delta L_\tau}{\delta \theta} + \left( \hat{\delta}_T \right) \frac{\delta h_T}{\delta \theta} \right) \]  

Train \( M_T \) by minimizing \( d(\delta_T, \hat{\delta}_T) \)

True \( \delta_T \) not available: Bootstrapping

\[ \delta_T = \sum_{\tau=T+1}^{2T} \frac{\delta L_\tau}{\delta h_T} + \hat{\delta}_{2T+1} \frac{h_{2T}}{h_T} \]
Other applications

1. Add an auxiliary task
2. Combine with true backpropagation gradients
3. Arbitrary Network Graphs
Results: Penn Tree Bank Language Modeling

![Graph showing BPC vs Data Time for different T values](image)

- Blue line: $T = 8.0$ (1.354)
- Red line: $T = 20.0$ (1.349)
- Black line: $T = 40.0$ (1.344)

Reviewed by: Arshdeep Sekhon (University of Virginia)
Results: Copy Repeat Copy Task

1. **Copy**: Copy a sentence of length N

2. **Repeat Copy**: Copy a sentence of length N R times

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>8</th>
<th>20</th>
<th>40</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy</td>
<td>7</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>16</td>
<td>14</td>
<td>18</td>
<td>18</td>
<td>-</td>
</tr>
<tr>
<td>Repeat Copy</td>
<td>7</td>
<td>5</td>
<td>19</td>
<td>23</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>39</td>
<td>33</td>
<td>39</td>
<td>59</td>
<td>-</td>
</tr>
</tbody>
</table>

3. Max sequence length successfully modeled increases with DNI for the same T in BPTT