Rapid Learning or Feature Reuse?  
Towards Understanding the Effectiveness of MAML

Aniruddh Raghu, Maithra Raghu, Samy Bengio, Oriol Vinyals
In this paper

- Where does the superior of MAML come from?
  - Rapid learning
  - Feature reuse
- Almost No Inner Loop (ANIL) and No Inner Loop (NIL)
Rapid Learning vs Feature Reuse

Rapid Learning

Outer loop performs meta-initialization

Feature Reuse

Inner loop performs task-adaptation

Rapid learning
Examine Feature Update

The inner loop mostly change the top layer.

? Any coincidence with vanishing gradient
ANIL

- Inner loop update:
  - Remove params update of the lower layers
  - Remain params update of the top layer

\[
\theta^*_{T_b} = \begin{cases} 
\theta_1 - \alpha \frac{\partial L_{T_b}(\theta)}{\partial \theta_1} \\
\theta_2 - \alpha \frac{\partial L_{T_b}(\theta)}{\partial \theta_2} \\
\theta_{\text{head}} - \alpha \frac{\partial L_{T_b}(\theta)}{\partial \theta_{\text{head}}} 
\end{cases}
\]

\[
\theta^*_{T_c} = \begin{cases} 
\theta_1 \\
\theta_2 \\
\theta_{\text{head}} - \alpha \frac{\partial L_{T_b}(\theta)}{\partial \theta_{\text{head}}} 
\end{cases}
\]
### ANIL

<table>
<thead>
<tr>
<th>Method</th>
<th>Omniglot-20way-1shot</th>
<th>Omniglot-20way-5shot</th>
<th>MiniImageNet-5way-1shot</th>
<th>MiniImageNet-5way-5shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML</td>
<td>93.7 ± 0.7</td>
<td>96.4 ± 0.1</td>
<td>46.9 ± 0.2</td>
<td>63.1 ± 0.4</td>
</tr>
<tr>
<td>ANIL</td>
<td>96.2 ± 0.5</td>
<td>98.0 ± 0.3</td>
<td>46.7 ± 0.4</td>
<td>61.5 ± 0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>HalfCheetah-Direction</th>
<th>HalfCheetah-Velocity</th>
<th>2D-Navigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML</td>
<td>170.4 ± 21.0</td>
<td>-139.0 ± 18.9</td>
<td>-20.3 ± 3.2</td>
</tr>
<tr>
<td>ANIL</td>
<td>363.2 ± 14.8</td>
<td>-120.9 ± 6.3</td>
<td>-20.1 ± 2.3</td>
</tr>
</tbody>
</table>

- The performance of MAML and ANIL are comparable
- Inner loop updates for lower layer is not necessary
### NIL

- **Top layer at inference**
  - Train ANIL/MAML as usual
  - Testing: Replace the top layer by cosine similarity

- **Conclusion:**
  - With no task-specific head, no task specific adaptation, the model is comparable to MAML/NIL
  - The feature learned by MAML/ANIL is good enough

<table>
<thead>
<tr>
<th>Method</th>
<th>Omniglot-20way-1shot</th>
<th>Omniglot-20way-5shot</th>
<th>MiniImageNet-5way-1shot</th>
<th>MiniImageNet-5way-5shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML</td>
<td>93.7 ± 0.7</td>
<td>96.4 ± 0.1</td>
<td>46.9 ± 0.2</td>
<td>63.1 ± 0.4</td>
</tr>
<tr>
<td>ANIL</td>
<td>96.2 ± 0.5</td>
<td>98.0 ± 0.3</td>
<td>46.7 ± 0.4</td>
<td>61.5 ± 0.5</td>
</tr>
<tr>
<td>NIL</td>
<td>96.7 ± 0.3</td>
<td>98.0 ± 0.04</td>
<td>48.0 ± 0.7</td>
<td>62.2 ± 0.5</td>
</tr>
</tbody>
</table>
Reviewer comments

All these datasets are artificially created from the same dataset and hence it might be very easy to reuse features to get good performance.

I am not sure if the same analysis will hold if we consider a dataset where tasks are not this similar (like Meta-dataset, Triantafillou et al 2019)