On Exact Computation with an Infinitely Wide Neural Net

Sanjeev Arora\textsuperscript{1,2}  Simon S. Du\textsuperscript{2}  Wei Hu\textsuperscript{1}  
Zhiyuan Li\textsuperscript{1}  Ruslan Salakhutdinov\textsuperscript{3}  Ruosong Wang\textsuperscript{3}

\textsuperscript{1}Princeton University  \textsuperscript{2}Institute for Advanced Study  \textsuperscript{3}Carnegie Mellon University
Introduction

• Recent papers [Li and Liang, Du et al., Allen-Zhu et al., Zou et al.] suggested that NNs with sufficiently large width can achieve 0 training error via gradient descent.

• [Jacot et al.] showed that as one increases the width to infinity, a certain limiting behavior, called neural tangent kernel (NTK), can emerge.
Questions we studied

• 1. Can we formally show that the prediction of NNs is equivalent to that of NTKs when width is sufficiently large?

• 2. How does NTK perform?
Theoretical Contribution

Theorem (Arora, Du, Hu, Li, Salakhutdinov, Wang, NeurIPS 2019): When width is sufficiently large (polynomial in number of data, depth and the inverse of target accuracy $\varepsilon$), the predictor learned by applying gradient descent on a neural network is $\varepsilon$-close to the kernel regression predictor of the corresponding neural tangent kernel.
Experimental Contribution

Dynamic programming-based algorithms for calculating NTKs for CNNs (CNTKs) + efficient GPU implementations.

<table>
<thead>
<tr>
<th>Depth</th>
<th>CNN-V</th>
<th>CNTK-V</th>
<th>CNN-GAP</th>
<th>CNTK-GAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>59.97%</td>
<td>64.47%</td>
<td>63.81%</td>
<td>70.47%</td>
</tr>
<tr>
<td>4</td>
<td>60.20%</td>
<td>65.52%</td>
<td>80.93%</td>
<td>75.93%</td>
</tr>
<tr>
<td>6</td>
<td>64.11%</td>
<td>66.03%</td>
<td>83.75%</td>
<td>76.73%</td>
</tr>
<tr>
<td>11</td>
<td>69.48%</td>
<td>65.90%</td>
<td>82.92%</td>
<td>77.43%</td>
</tr>
<tr>
<td>21</td>
<td>75.57%</td>
<td>64.09%</td>
<td>83.30%</td>
<td>77.08%</td>
</tr>
</tbody>
</table>
Future Directions

• Understand the design of neural network architectures and common techniques in deep learning, e.g., batch normalization and residual layers, from the lens of neural tangent kernel.

• Combine NTK with other techniques in kernel methods to further improve the performance.
Thanks!

- Full paper: https://arxiv.org/abs/1904.11955
- Code: https://github.com/ruosongwang/CNTK