Deep Expander Networks
Efficient Deep Networks from Graph Theory

Ameya Prabhu*, Girish Varma*, Anoop Namboodiri
IIIT Hyderabad
INDIA
Efficient CNNs

<table>
<thead>
<tr>
<th>Operation</th>
<th>GFlops</th>
<th>Accuracy [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-19</td>
<td></td>
<td>90</td>
</tr>
<tr>
<td>ResNet-152</td>
<td></td>
<td>80</td>
</tr>
<tr>
<td>MobileNet</td>
<td></td>
<td>70</td>
</tr>
<tr>
<td>Inception V4</td>
<td></td>
<td>60</td>
</tr>
<tr>
<td>Inception V3</td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>NAS Net-A</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>SE Net</td>
<td></td>
<td>30</td>
</tr>
</tbody>
</table>

Graph showing the relationship between operations (GFlops) and accuracy [%] for different neural network architectures. The graph indicates that as operations increase, accuracy decreases, and vice versa. The graph also shows a trade-off between size (compress) and connectivity.
**Approaches Towards Efficiency**

**Better Network Design**
- Inception Net
- ResNet
- NAS Net
- P-NAS Net

**Efficient Layer Modification**
- Group Convolutions
- Pruning

![Diagram showing network architectures and efficiency improvement](image)

- **↑ Connectivity**
- **↓ Size**

- Diagram before and after pruning, indicating weight layer, relu, synapses pruning.
Better Layer Connections: \textbf{Train} \rightarrow \textbf{Prune}

- Need to train full network
- Need Multiple trainings
- Layer structure specific to given data
  - Not Transferrable

![Diagram of layer connections with arrows from Train to Prune]
**Can We: Prune → Train**

### Train → Prune
- ✗ Need to train full network
- ✗ Need Multiple trainings
- ✗ Layer structure specific to given data
  - Not Transferrable

### Prune → Train
- ✓ Train a compact network
- ✓ Single training
- ✓ Generic layer structure, independent of data
  - Transferrable

![Graph showing error vs. epochs with pruning and training stages](image-url)
**Pruning Without Data**

- Need to sparsify connections
- Need to ensure multi-layer connectivity

---

- Not well connected
- Retraining does not help

---

All this to be done without data!!
Expander Graphs

Combinatorics: Highly connected; Sever many edges to disconnect any large part of the graph.

Probability: Random walk on these converges to its limiting distribution as rapidly as possible.

Algebra: First positive eigenvalue of their Laplace operator is bounded away from zero.

Expander Graphs are simultaneously sparse and highly connected.
**Constructing Expander Layers**

- Pick an input node
- Connect it to $D$ random outputs
- Repeat for every input node
- Repeat for every layer
GUARANTEES ABOUT X-NETS

Theorem 1 (Sensitivity):
Let $n$ be the number of input as well as output nodes in the network and $G_1, G_2, \cdots, G_t$ be $D$-regular bipartite expander graphs with $n$ nodes on both sides. Then every output neuron is sensitive to every input in a Deep X-Net defined by $G_i$'s with depth $t = O(\log_D n)$.

Theorem 2 (Rich Connectivity):
Let $n$ be the number of input as well as output nodes in the network and $G$ be $D$ regular bipartite expander graph with $n$ nodes on both sides. Let $S, T$ be subsets of input and output nodes in the X-Net layer defined by $G$. The number of edges between $S$ and $T$ $|E| \approx \frac{D|S||T|}{n}$

Lots of paths between any $S$ and $T$
NOTE: CONNECTIVITY GRAPH OF CONVOLUTIONS
**Note:** Connectivity Graph of Convolutions

Conv Layers

Top View

Connectivity Graph
Our Convolutional Layer

Red and green represent the subsets that are connected.
Expander vs. Full Convolution

Expander Convolution

- Smaller Filters (Compressed)
- Fewer Computations (Efficient)
- Maintains Overall Connectivity

Full Convolution
IMPLEMENTING X-NETS

OpenAI BlockSparse
EXPERIMENTAL RESULTS

Comparisons with:

- **Layer Connectivity Graphs**: Group Convolution
- **Network Compression**: Pruning
- **Efficient Architectures**: ResNet and DenseNet
Benchmarking with Group Convolution

X-Conv beats G-Conv by ~4-5% on a compact MobileNet-0.5 on Imagenet.

<table>
<thead>
<tr>
<th>Compression</th>
<th>G-Conv</th>
<th>X-Conv (Ours)</th>
<th>Err. Red.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2x</td>
<td>42.55%</td>
<td>41.78%</td>
<td>0.8%</td>
</tr>
<tr>
<td>4x</td>
<td>50.59%</td>
<td>46.00%</td>
<td>4.6%</td>
</tr>
<tr>
<td>8x</td>
<td>54.87%</td>
<td>50.77%</td>
<td>4.1%</td>
</tr>
<tr>
<td>16x</td>
<td>60.97%</td>
<td>55.37%</td>
<td>5.6%</td>
</tr>
</tbody>
</table>
## Comparison with Pruning

### VGG-16 on CIFAR-10

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th># Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>Li et al.</td>
<td>93.4 %</td>
<td>5.4 M (2.8x)</td>
</tr>
<tr>
<td>NW Slimming</td>
<td>93.8 %</td>
<td>2.3 M (6.5x)</td>
</tr>
<tr>
<td>X-VGG 16-1</td>
<td>93.4 %</td>
<td>1.65 M (9x)</td>
</tr>
<tr>
<td>X-VGG 16-2</td>
<td>93.0 %</td>
<td>1.15 M (13x)</td>
</tr>
<tr>
<td>VGG-16 Orig</td>
<td>94.0 %</td>
<td>15.0 M (1.0x)</td>
</tr>
</tbody>
</table>

### AlexNet on ImageNet

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th># Params</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collins et al.</td>
<td>55.1 %</td>
<td>15.2 M (4x)</td>
</tr>
<tr>
<td>Zhou et al.</td>
<td>54.4 %</td>
<td>14.1 M (4.3x)</td>
</tr>
<tr>
<td>Han et al.</td>
<td>57.2 %</td>
<td>6.7 M (9.1x)</td>
</tr>
<tr>
<td>Srinivas et al.</td>
<td>56.9 %</td>
<td>5.9 M (10.3x)</td>
</tr>
<tr>
<td>Guo et al.</td>
<td>56.9 %</td>
<td>3.4 M (18x)</td>
</tr>
<tr>
<td>X-AlexNet-1</td>
<td>55.2 %</td>
<td>7.6 M (8x)</td>
</tr>
<tr>
<td>X-AlexNet-2</td>
<td>56.2 %</td>
<td>9.7 M (6.3x)</td>
</tr>
<tr>
<td>AlexNet-Orig</td>
<td>57.2 %</td>
<td>61 M (1.0x)</td>
</tr>
</tbody>
</table>

**X-Nets are as compressible as the best pruning techniques**
ADVANTAGES OVER PRUNING

Train in 1 cycle

Transferable Architectures

Go Wider / Deeper

OR
Wider/Deeper networks with higher compression achieves same error rate with fewer parameters
ResNet vs X-ResNet on ImageNet
**ResNet vs X-ResNet on ImageNet**

![Graph showing the comparison between ResNet and X-ResNet on ImageNet. The graph plots Error against FLOPs with red dots representing ResNet and blue dots representing X-ResNet.](image-url)
DenseNet vs X-DenseNet on CIFAR-10

Error vs FLOPs graph showing comparison between DenseNet and X-DenseNet.
**In Summary:**

- X-Nets provide a principled way to compress deep networks.
- Single-cycle training of a lighter data-agnostic network.
- Allows training of wider and deeper networks.
- Achieves good error-flops trade-off.
- Highlights the use of global connectivity analysis in network architecture design.
THANK YOU!!

Using our Pytorch Code:

```
from layers import ExpanderLinear, ExpanderConv2d

nn.Conv2d(...) → ExpanderConv2d(..., expandSize=128)
nn.Linear(...) → ExpanderLinear(..., expandSize=256)
```

GitHub Repo: https://github.com/DrImpossible/Deep-Expander-Networks

Visit us @ Poster ID: P-4A-04