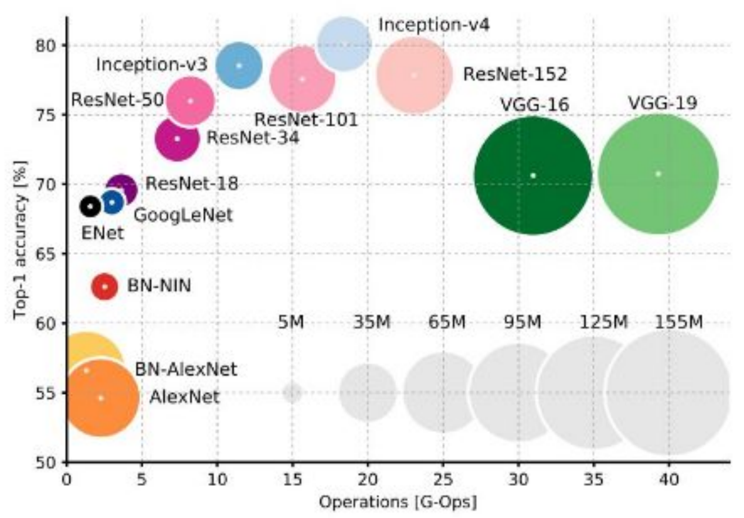
Deep Expander Networks: Efficient Deep Networks from Graph Theory

Ameya Prabhu*, Girish Varma*, Anoop Namboodiri
Center for Visual Information Technology, Kohli Center for Intelligent Systems, IIIT Hyderabad, India

EFFICIENT CNNS

DNNs have great accuracies but are resource intensive. Hence important to study speed/accuracy tradeoffs.

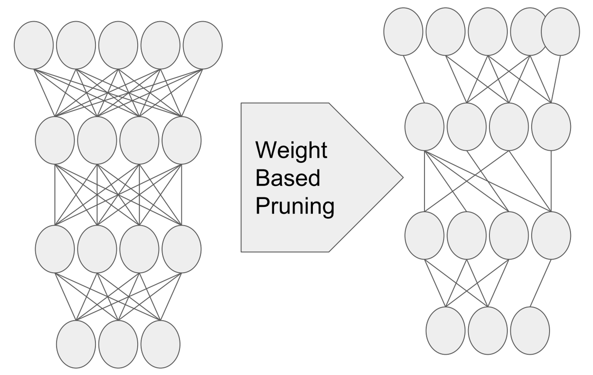
CNNs are especially runtime heavy. Essential to make CNNs efficient for making them applicable in real-time and embedded systems.



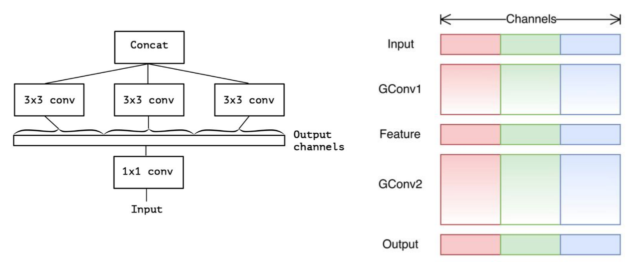
A. Canziani, A. Paszke, and E. Culurciello. An analysis of deep neural network models for practical applications.arXiv preprint arXiv:1605.07678 , 2016.

MAJOR APPROACHES & CHALLENGES

Pruning

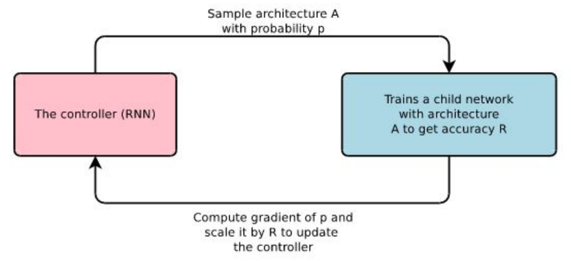


Architecture Design



Depthwise Seperable, Grouped Convolutions

Architecture Search

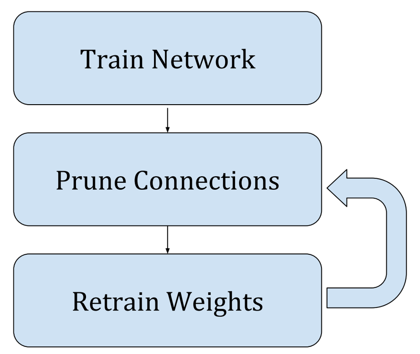


Trained using RL or Evolutionary Strategy.

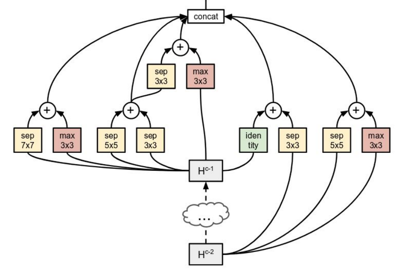
Challenges

Training process gets more complicated with newer hyperparameters.

Ideally should allow training of novel architectures themselves.



Trial and Error methods will not scale to large datasets.



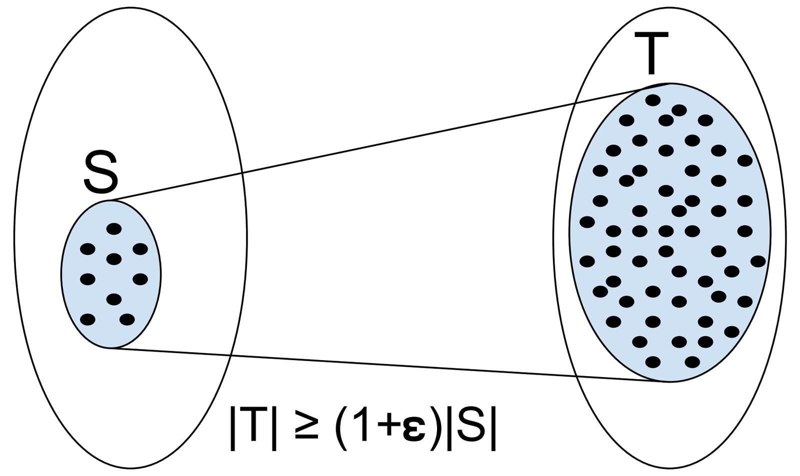
EXPANDER GRAPHS

Expander Graphs:Graphs such that neighbourhood of every subset of vertices expands.

Well studied theory for over 50 years in theoretical computer science.

There are sparse graphs with O(n) number of edges that has the expander properties.

A random D-regular graph for D>2, is an expander with high probability.



OUR APPROACH

Model CNNs using Graphs.
sparsity = efficiency

Hypothesise that
expressivity = connectivity

Propose to use expander graphs that are simultaneously sparse and well connected

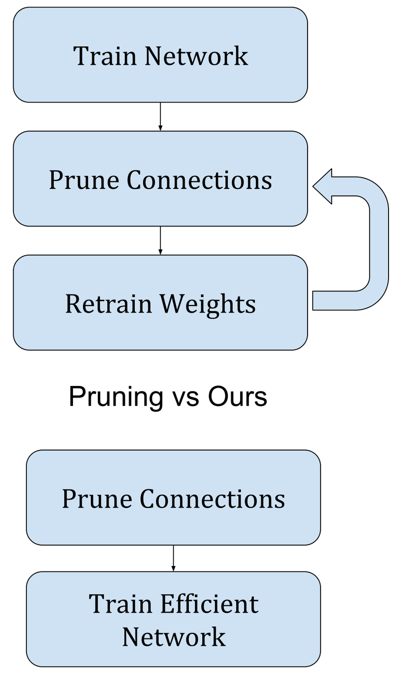
Advantages

Compact, fast in train time

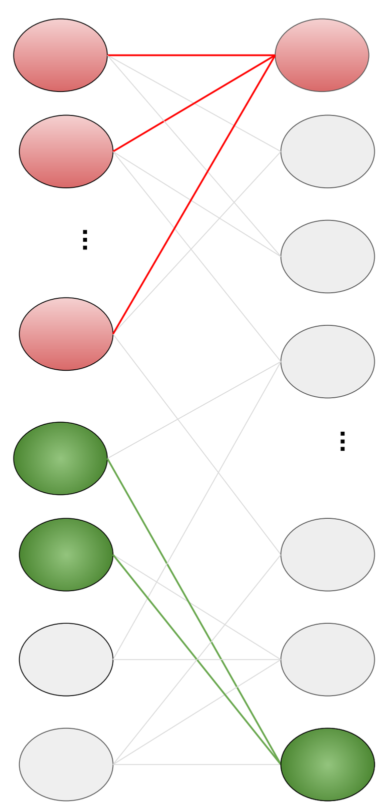
Training in one cycle/phase, similar to original models.

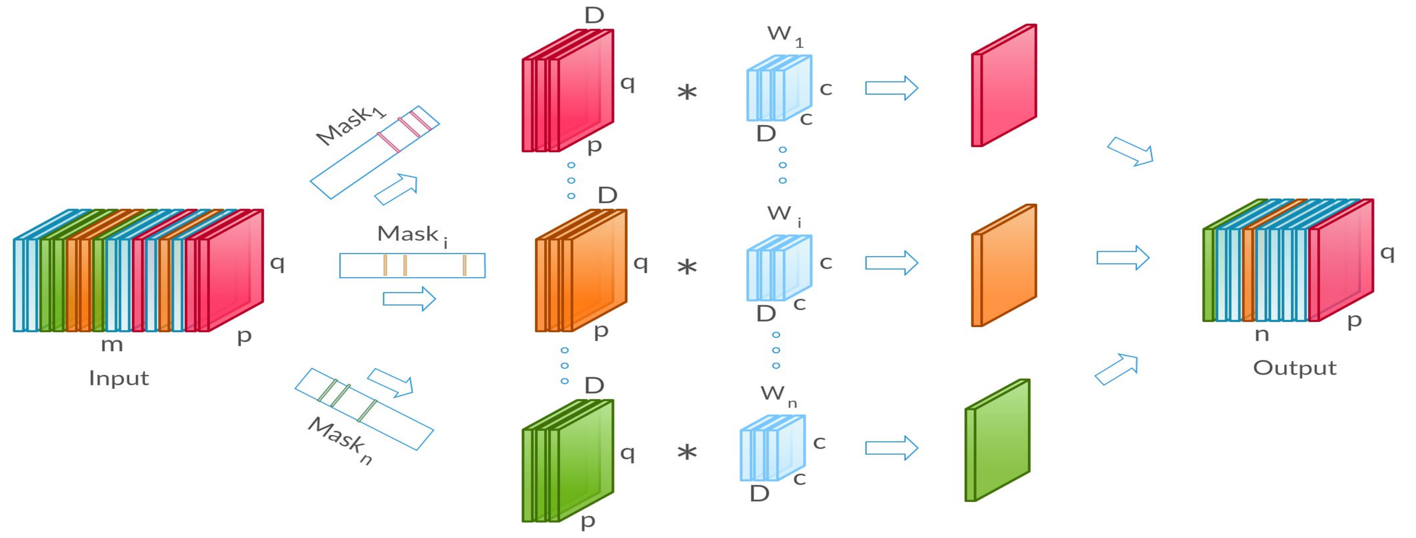
Bulky full model need not be trained.

Task-independent architectures. Generalizable.



X-CONV LAYERS & X-NETS



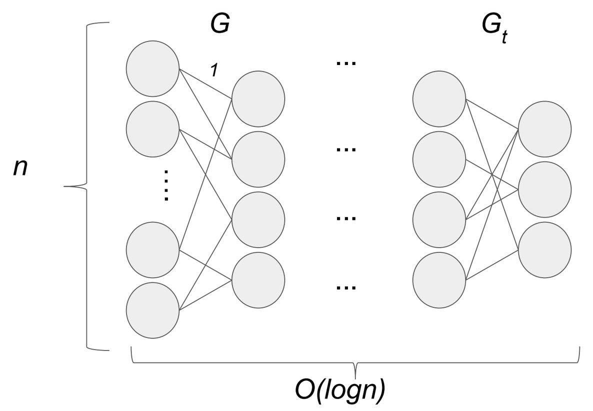


The connections are fixed according to an expander graph structure. This is a good prior to form a compact networks before training that is efficiently implementable.

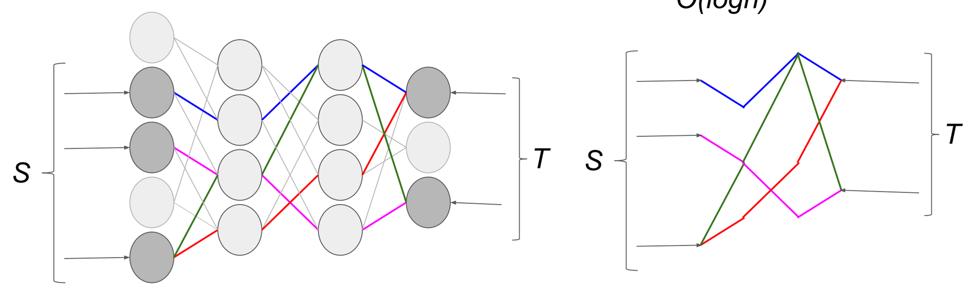
We study X-MobileNet, X-DenseNet, X-ResNet, X-VGG and X-AlexNet where the Conv layers are replaced by X-Conv layers.

THEORETICAL PROPERTIES

Theorem 1 (Sensitivity of X-Nets): $G_1, G_2, \cdots, G_t$ 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(logn).



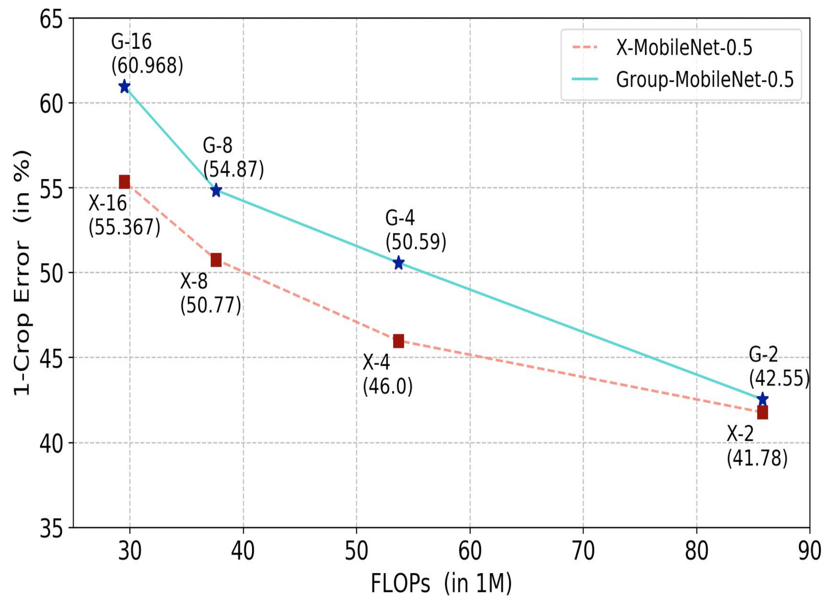
Theorem 2 (Mixing in X-Nets): Let S,T be subsets of input and output nodes in the X-Net layer defined by G. The number of edges between S and T is ≈ D |S||T| / n



#Paths ≈ D |S||T|/n

RESULTS

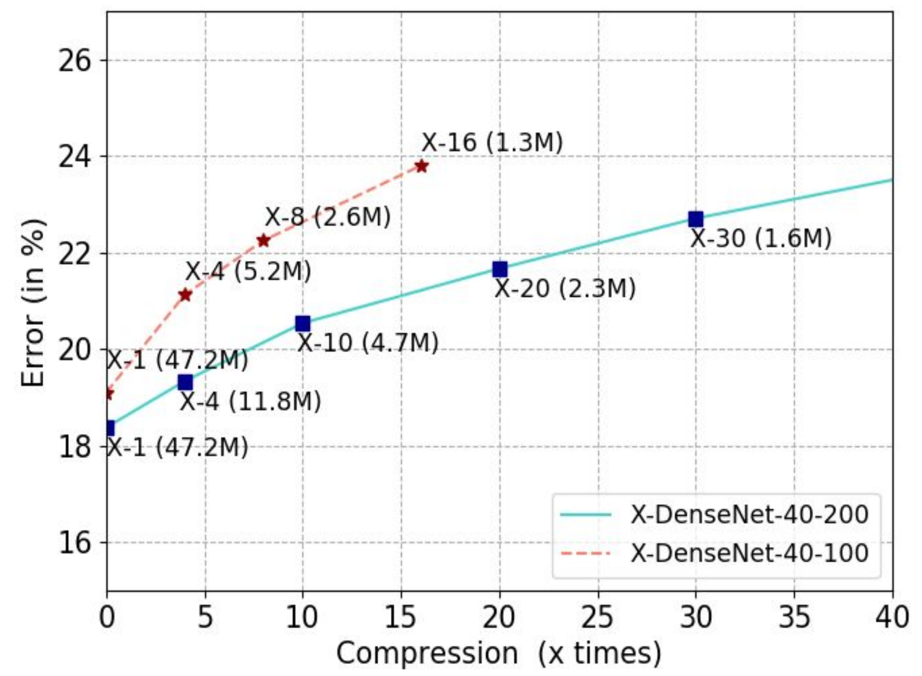
Comparison with Grouped Convolution (G-Conv) with same Sparsity

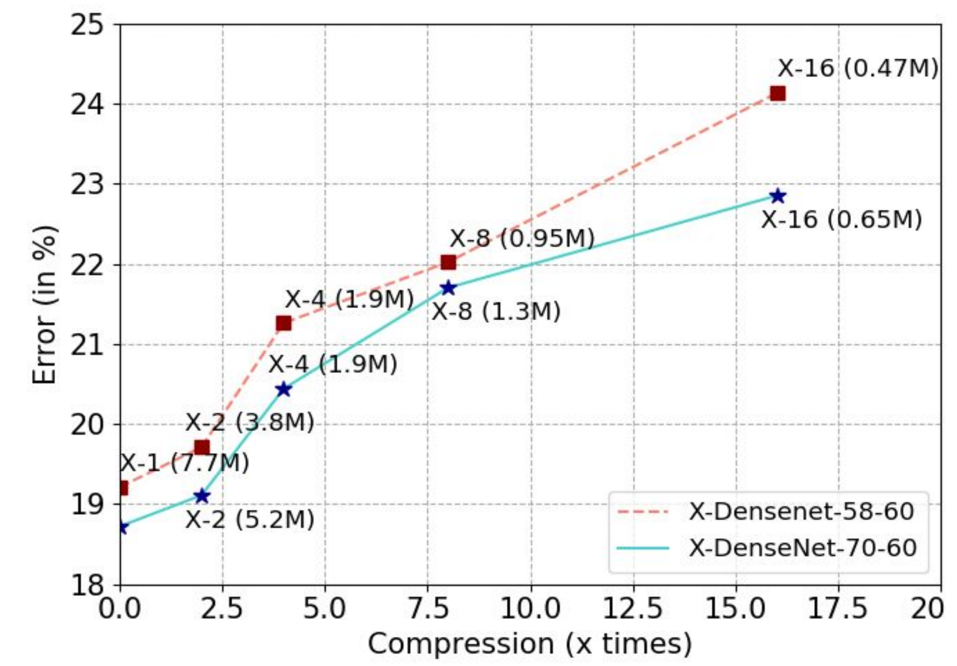


Compression	G-Conv Error	X-Conv (Ours) Error
x2	42.55%	41.78%
x4	50.59%	46.00%
x8	54.87%	50.77%
x16	60.97%	55.37%

X-Conv beats G-Conv by 4-5% on MobileNet-0.5

Wider and Deeper X-DenseNets





Wider or Deeper Compressed Networks give better parameter efficiency and accuracy.

PYTORCH IMPLEMENTATION

Convert your code to use XConv2d and XLinear layers:

```
from layers import XLinear, XConv2d

nn.Conv2d(...)  ➡  XConv2d(..., expandSize=128)
nn.Linear(...)  ➡  XLinear(..., expandSize=256)
```

Email: ameya.pandurang.prabhu@gmail.com
Code: https://github.com/DrImpossible/Deep-Expander-Networks