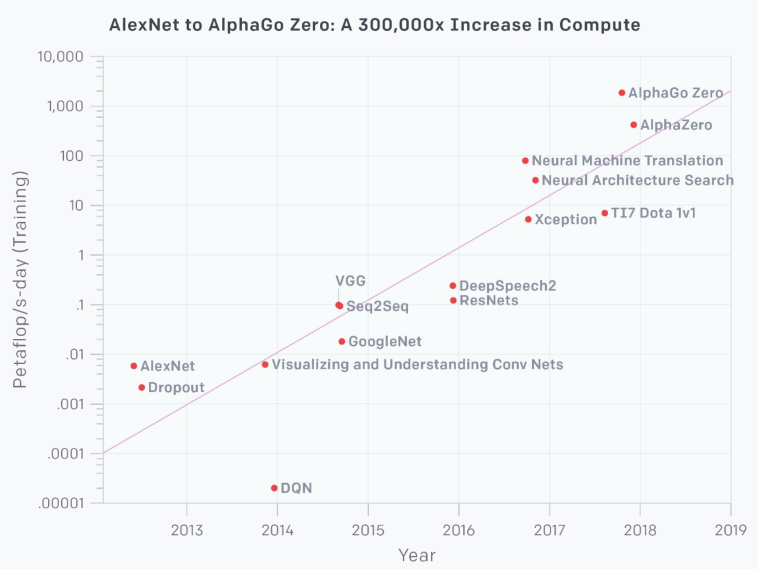
A Theoretical View on Sparsely Activated Networks

Cenk Baykal, Nishanth Dikkala, Rina Panigrahy, Cyrus Rashtchian, Xin Wang

Google Research

Introduction



Increasingly prohibitive computational and environmental costs of modern AI

Moore's law cannot keep up

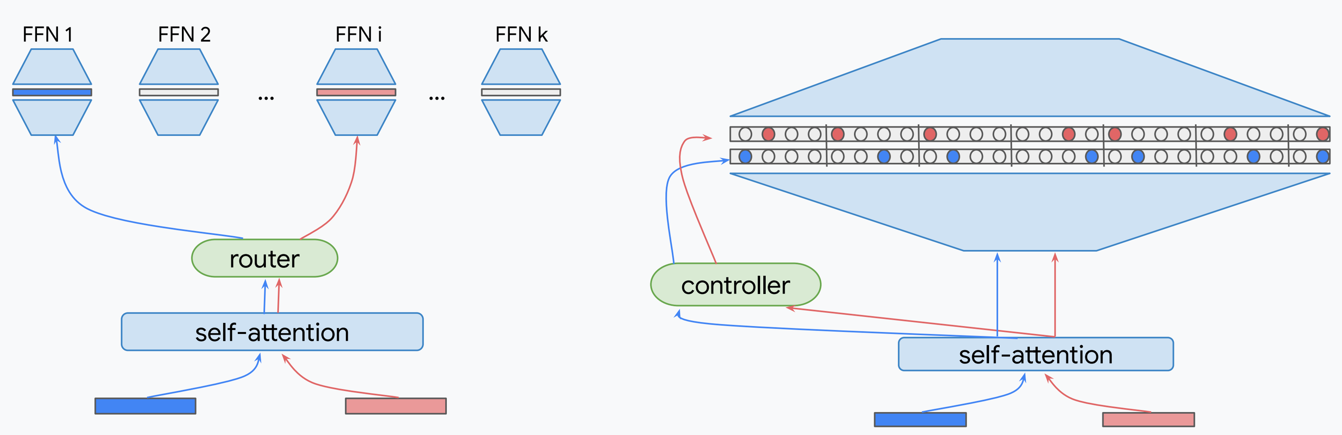
Sparsification & Compression techniques, such as Sparsely Activated Networks, have become essential...

But, they lack theoretical foundations

Sparsely Activated Networks

Idea: increase capacity (# of parameters) without increasing compute

Examples: Switch Transformer, Scaling Transformer, Mixture-of-Experts



Switch transformer

Scaling transformer

***Our work:** theoretically establish the power of sparsely activated networks relative to dense ones*

DSM, LSH Models

Data-dependent Sparse Models (DSM)

Theoretical model of sparsely activated networks

Key idea: routing function specifies the subnetwork (a.k.a., expert)

A = final layer matrix

$\mathrm{mask}(x)$ = routing function for sparsity (zero out most positions)

$\phi(x)$ = representation from non-final layers

Putting it together: composing these gives the sparse network

$$g(x) = (A \circ \mathrm{mask}(x))\phi(x)$$

Lemma: The DSM model captures modern networks, such as Switch Transformers and Scaling Transformers.

Locality Sensitive Hashing (LSH)-based Sparse Networks

Locality Sensitive Hashing

Hash function that maps similar points into similar buckets

Hyperplane LSH: form buckets based on multiple random hyperplanes

$$h_i(x) = \left\lfloor \frac{a_i^\top x + b_i}{\varepsilon} \right\rfloor$$

LSH Networks

Data-dependent routing via LSH

Main Theoretical Result

Large family of functions where sparse networks are as powerful as dense ones

***Theorem 1 (informal):** Sparsely Activated Models of the same total size as Dense models can represent Lipschitz functions to the same accuracy as dense models while using exponentially fewer operations during training and inference.*

Computational Efficiency of DSMs

***Theorem 2:** For learning a Lipschitz function in d-dimensions using a sparse network with size $O(\sqrt{d}^d/\epsilon^d)$ we can learn to error ϵ, where each forward pass takes time $O(d^2 \log(1/\epsilon))$. On the other hand, a dense model requires time $\Omega(\sqrt{d}^d/\epsilon^d)$.*

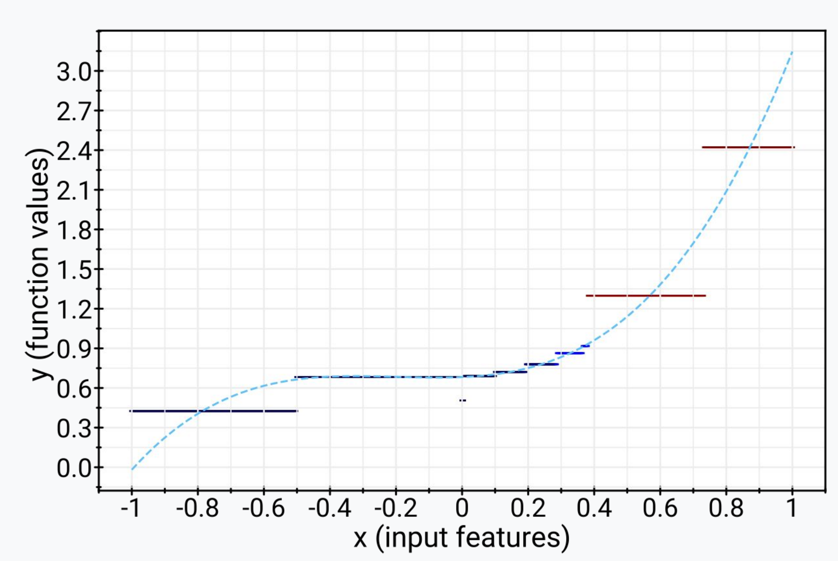
Proof Overview

- Lipschitz functions map nearby inputs to similar values
- Use a separate expert for each "small" region of input space
- Enough regions → small error in function approximation

Dashed curve is the target function graph

Piecewise constant curve is the learned LSH model output

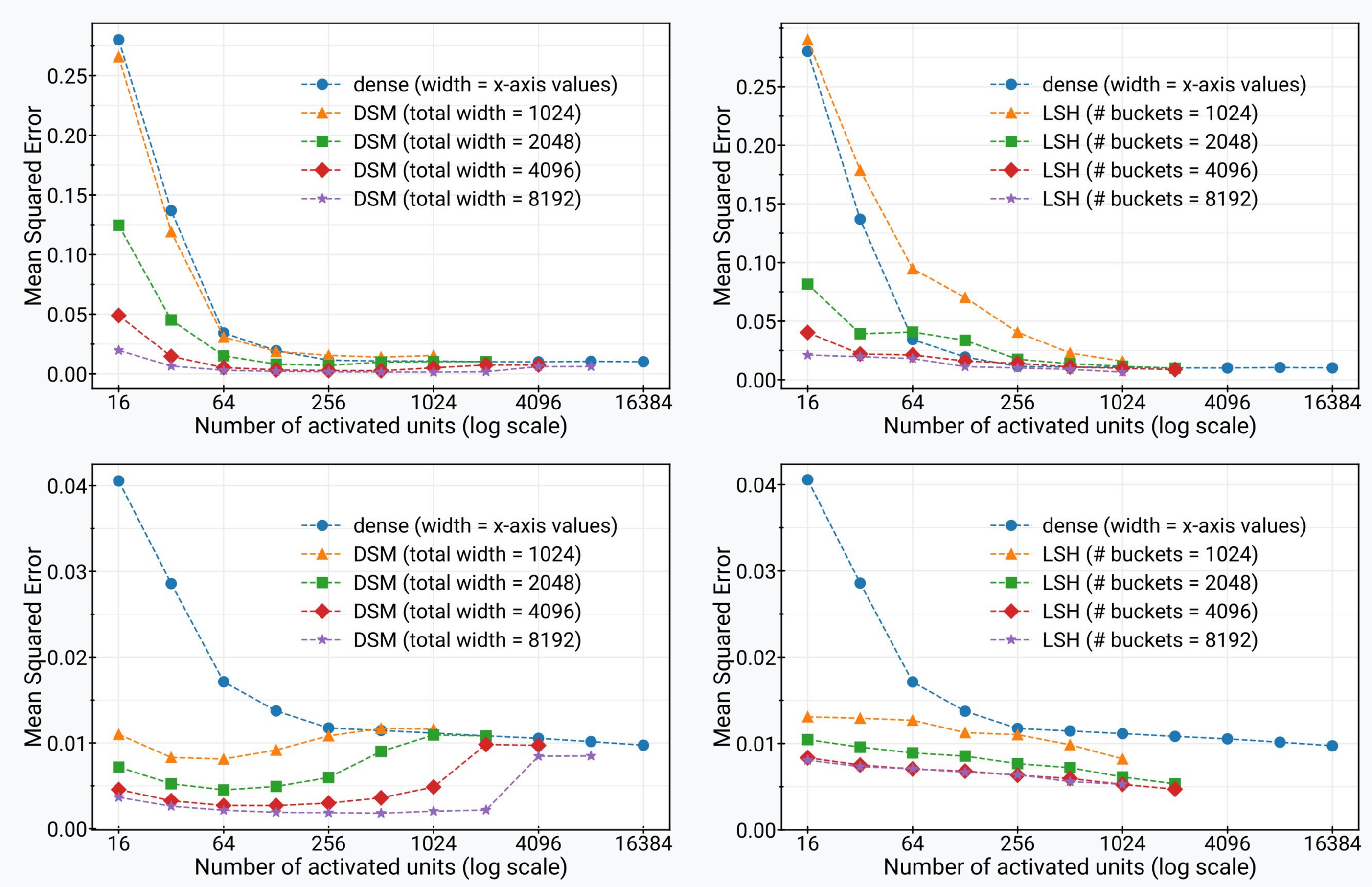
Different colors correspond to different LSH buckets



Experiments - Synthetic

Target function: random degree 4 polynomial on 8-dim input space

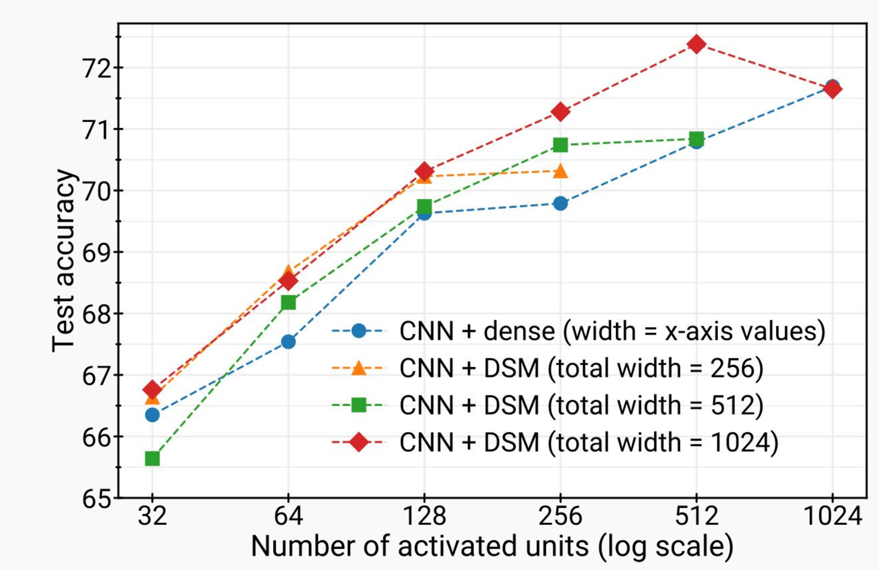
Random Boolean function interpolated over a high-dimensional hypercube



***Takeaway:** both DSM and LSH-based sparse models outperform or match dense models*

Experiments - CIFAR-10

DSM outperforms dense models given the same number of activated units



CIFAR-10 test accuracy for dense & DSM models

Model \ # activated units	256	512
Dense	69.79	70.79
DSM (50% sparse)	**70.74**	71.33
DSM (25% sparse)	69.8	**71.68**

***Observation:** Wide and sparse models generally outperform narrow and dense ones*