DEEPSCALE
forrest@deepscale.ai

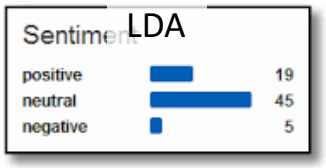
Architecting new deep neural networks for embedded applications

Forrest Iandola

DEEPSCALE
forrest@deepscale.ai

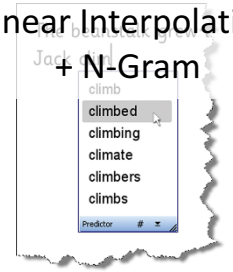
Machine Learning in 2012

Sentiment Analysis



Word Prediction

Linear Interpolation
+ N-Gram



Text Analysis

Audio Analysis

Computer Vision

Object Detection

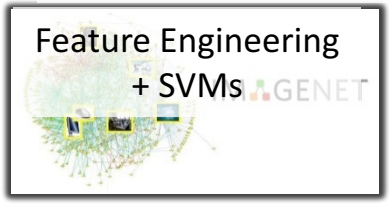
Deformable Parts Model

Semantic Segmentation

Speech Recognition

Audio Concept Recognition

Image Classification

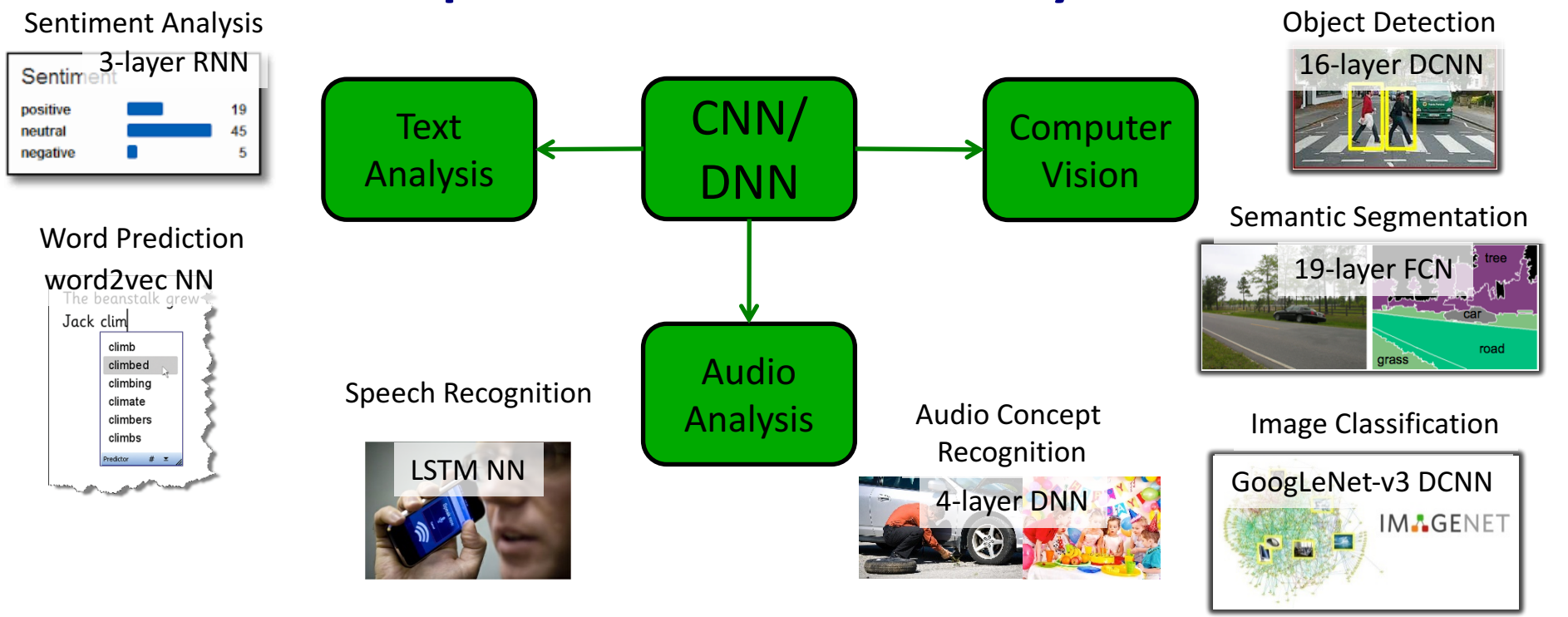


We have 10 years of experience in a broad variety of ML approaches …

[1] B. Catanzaro, N. Sundaram, **K. Keutzer.** Fast support vector machine training and classification on graphics processors. International Conference on Machine Learning (ICML), 2008.
[2] Y. Yi, C.Y. Lai, S. Petrov, **K. Keutzer**. Efficient parallel CKY parsing on GPUs. International Conference on Parsing Technologies, 2011.
[3] K. You, J. Chong, Y. Yi, E. Gonina, C.J. Hughes, Y. Chen, **K. Keutzer**. Parallel scalability in speech recognition. *IEEE Signal Processing Magazine*, 2009.
[4] **F. Iandola**, M. **Moskewicz**, K. **Keutzer**. libHOG: Energy-Efficient Histogram of Oriented Gradient Computation. ITSC, 2015.
[5] N. Zhang, R. Farrell, **F. Iandola**, and T. Darrell. Deformable Part Descriptors for Fine-grained Recognition and Attribute Prediction. ICCV, 2013.
[6] M. Kamali, I. Omer, **F. Iandola**, E. Ofek, and J.C. Hart. Linear Clutter Removal from Urban Panoramas International Symposium on Visual Computing. ISVC, 2011.

DEEPSCALE
forrest@deepscale.ai

By 2016, Deep Neural Networks Give Superior Solutions in Many Areas



Finding the "right" DNN architecture is replacing broad algorithmic exploration for many problems.

[7] K. Ashraf, B. Elizalde, F. **Iandola**, M. **Moskewicz**, J. Bernd, G. Friedland, K. **Keutzer**. Audio-Based Multimedia Event Detection with Deep Neural Nets and Sparse Sampling. ACM ICMR, 2015.

[8] F. **Iandola**, A. **Shen**, P. Gao, K. **Keutzer**. DeepLogo: Hitting logo recognition with the deep neural network hammer. arXiv:1510.02131, 2015.

[9] F. **Iandola**, M. **Moskewicz**, S. Karayev, R. Girshick, T. Darrell, K. **Keutzer**. DenseNet: Implementing Efficient ConvNet Descriptor Pyramids. arXiv: 1404.1869, 2014.

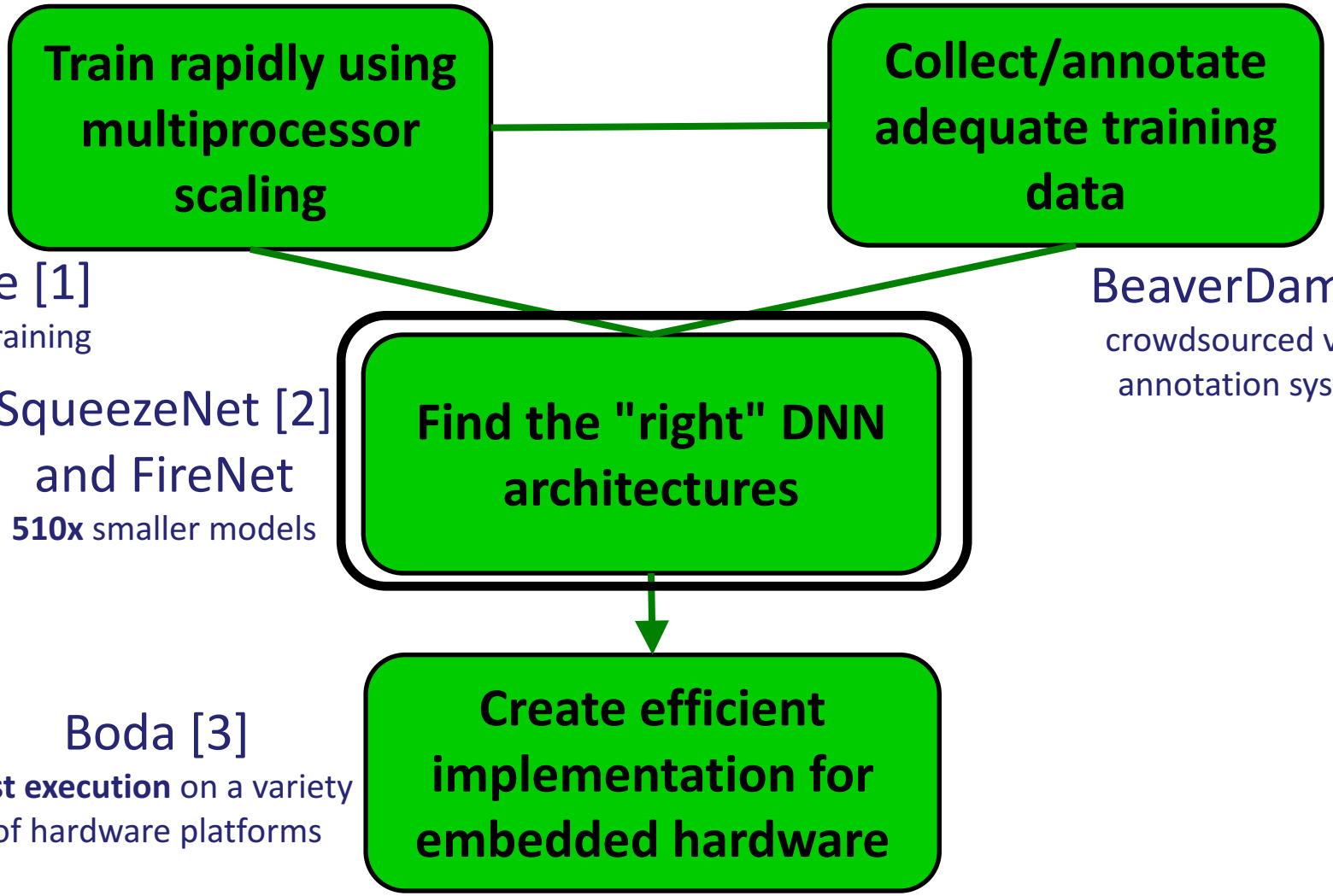
[10] R. Girshick, F. **Iandola**, T. Darrell, J. Malik. Deformable Part Models are Convolutional Neural Networks. CVPR, 2015.

[11] **F. Iandola**, K. Ashraf, **M.W. Moskewicz**, **K. Keutzer**. FireCaffe: near-linear acceleration of deep neural network training on compute clusters. arXiv:1511.00175, 2015. Also, CVPR 2016, pp. 2592–2600.

[12] K. Ashraf, B. Wu, **F.N. Iandola**, **M.W. Moskewicz**, **K. Keutzer**. Shallow Networks for High-Accuracy Road Object-Detection. arXiv:1606.01561, 2016.

DEEPSCALE
forrest@deepscale.ai

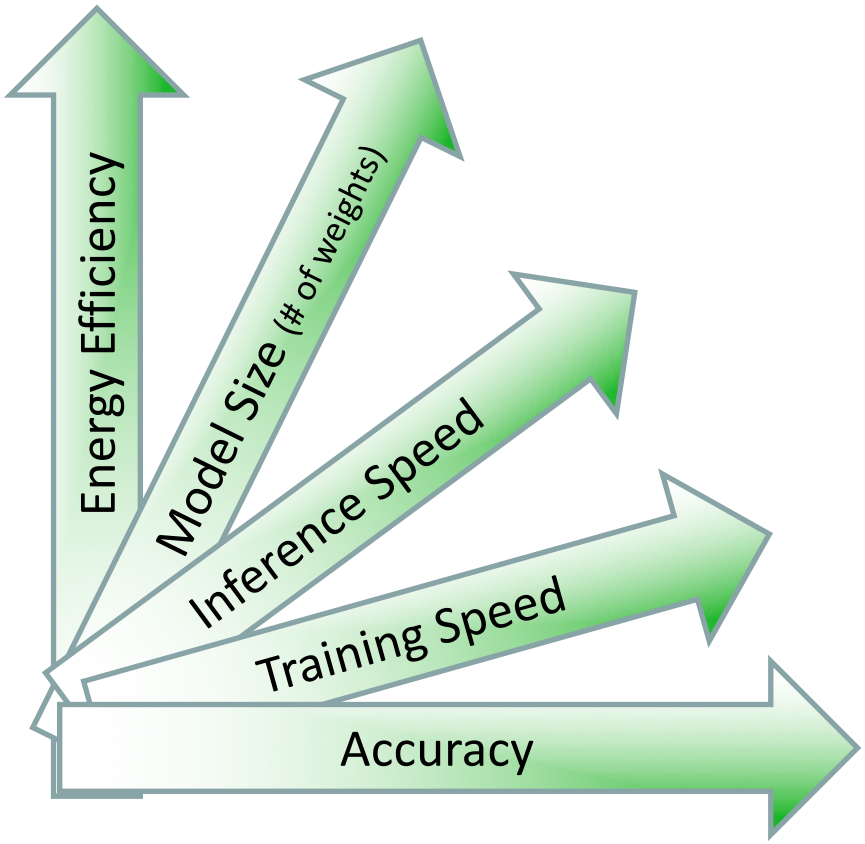
Key ingredients for success with Deep Neural Networks



[1] F.N. Iandola, K. Ashraf, M.W. Moskewicz, and K. Keutzer. **FireCaffe**: near-linear acceleration of deep neural network training on compute clusters. CVPR, 2016.
[2] F.N. Iandola, M. Moskewicz, K. Ashraf, S. Han, W. Dally, K. Keutzer. **SqueezeNet**: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.
[3] M.W. Moskewicz, F.N. Iandola, K. Keutzer. "**Boda-RTC**: Productive Generation of Portable, Efficient Code for Convolutional Neural Networks on Mobile Computing Platforms." WiMob, 2016.
[4] http://github.com/antingshen/BeaverDam

DEEPSCALE
forrest@deepscale.ai

Key metrics for specifying CNN/DNN design goals



To achieve the optimal results on these metrics, it's important to design and/or evaluate:

- CNN architectures
- Software/Libraries
- Hardware architectures

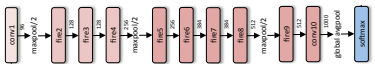
DEEPSCALE
forrest@deepscale.ai

Strategies for evaluating team progress on full-stack CNN/DNN system development

Evaluating individual contributions

Evaluating the overall system

CNN Team



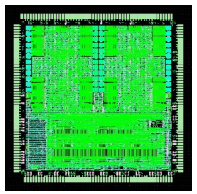
- Accuracy
- Quantity of computation
- Model Size

Software/Libraries Team

`kernel<<< >>>`

- Percent of peak throughput achieved on appropriate hardware

Hardware Team



- Power envelope
- Peak achievable throughput

- Energy per frame
- Inference speed per frame

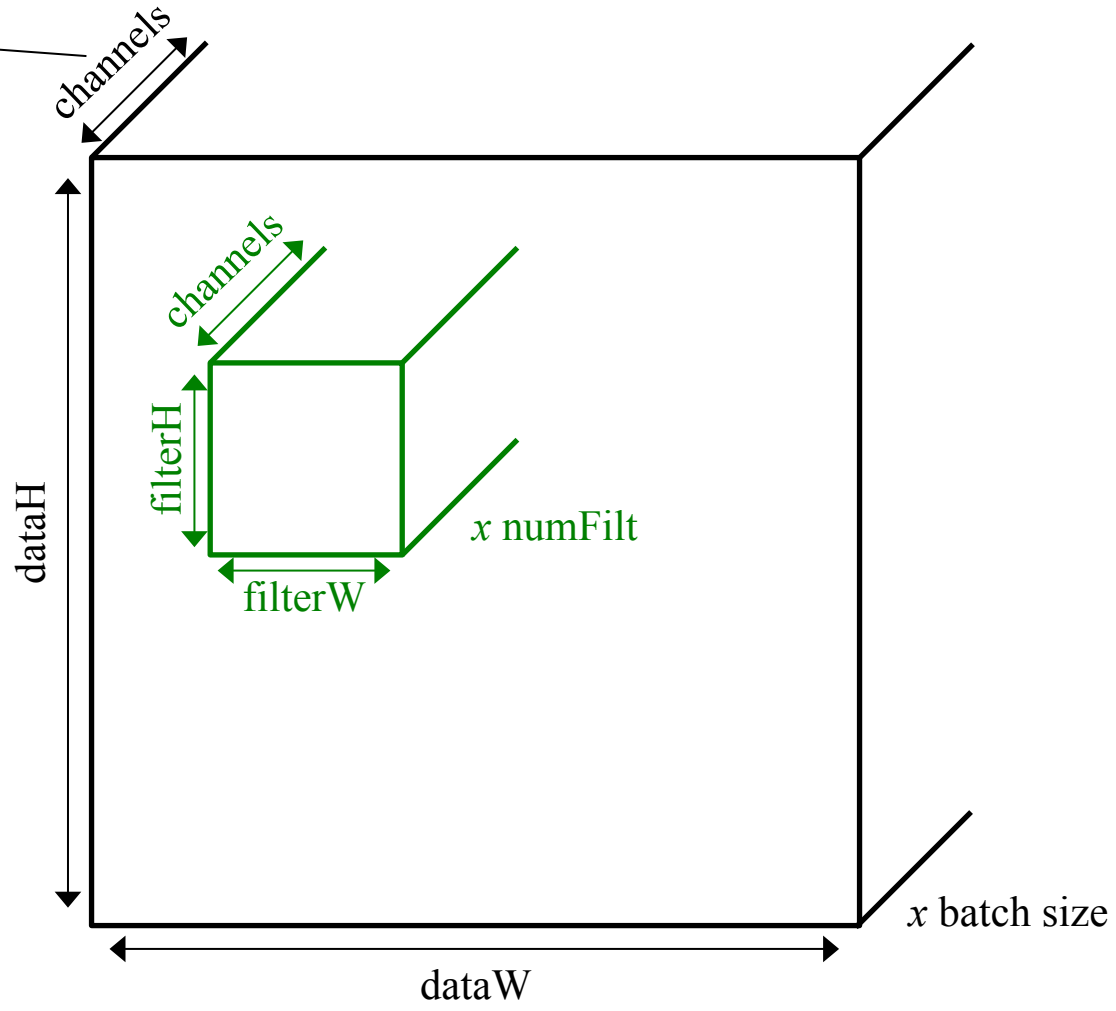
DEEPSCALE
forrest@deepscale.ai

Architecting small DNNs for embedded applications

DEEPSCALE
forrest@deepscale.ai

Reminder: Dimensions of a convolution layer

The number of channels in the current layer is determined by the number of filters (numFilt) in the previous layer.



Bear in mind that CNNs are comprised of *many* layers, L_1 , …, L_n

DEEPSCALE
forrest@deepscale.ai

Local and *Global* changes to CNN architectures

Examples of Local changes to CNNs

- Change the number of channels in the input data
- Change the number of filters in a layer
- Change the resolution of the filters in a layer (e.g. 3x3 → 6x6)
- Change the number of categories to classify

Effect of a Local change:

A Local change to layer L_i only affects the dimensions of layers L_i and L_{i+1}

Examples of Global changes to CNNs

- Change the strides or downsampling/upsampling in a layer
- Change the height and width of the input data (e.g. 640x480 → 1920x1440 image)

Effect of a Global change:

A Global change to layer L_i affects the dimensions of all downstream layers: L_{i+1} , …, L_n

DEEPSCALE
forrest@deepscale.ai

Effect of *Local* and *Global* changes to parallelism during CNN training

Recall the distributed data-parallel approach to training that we described earlier in the talk.

- Local changes involve modifying the dimensions of filters or channels.
  - Consider a local change where we increase the number of filters in a layer
  - Effect on training: This local change increases the number of parameters and therefore **increases the quantity of communication** required during training.

- Global changes involve modifying the dimensions of data or activations (i.e. the output data from the layers)
  - Consider a global change where we decrease the stride of a layer (increasing the number of activations it produces)
  - Effect on training: This global change does not affect the number of parameters and therefore **does not change the quantity of communication** required during training.

DEEPSCALE
forrest@deepscale.ai

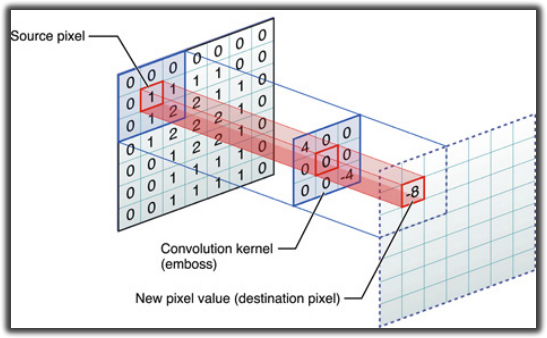
Effect of *Local* and *Global* changes to parallelism during CNN training

modification	type of modification	Δ Qty of output	Δ Qty of params	Δ Qty of computation
Initial CNN (NiN [82])	none	1x	1x	1x
4x more input channels	Local	1x	1x	1.3x
4x more filters in conv8	Local	1.1x	1.1x	1.1x
4x larger filter resolution in conv7	Local	1x	1.3x	1.3x
4x more categories to classify	Local	1x	1.4x	1.1x
remove pool3 downsampling layer	Global	2.6x	1x	3.8x
4x larger input data resolution	Global	4.2x	1x	4.3x

DEEPSCALE
forrest@deepscale.ai

FireNet: CNN architectures with few weights

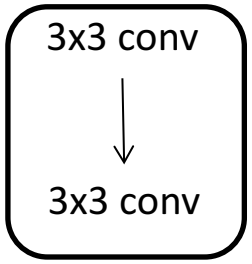
Image convolution in 2D



- Fewer weights in model → more scalability in training
- Observation: 3x3 filters contain 9x more weights and require 9x more computation than 1x1 filters.
- SqueezeNet is one example of a FireNet architecture

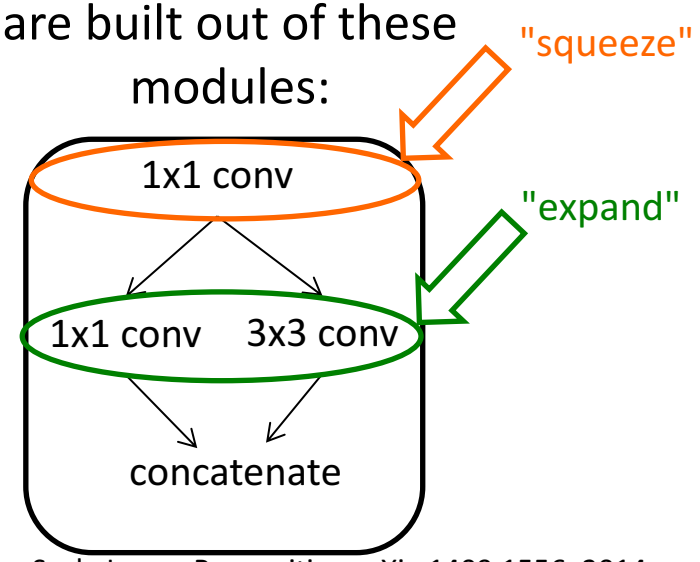
VGG [1] CNNs

are built out of these modules:



Our FireNet CNNs

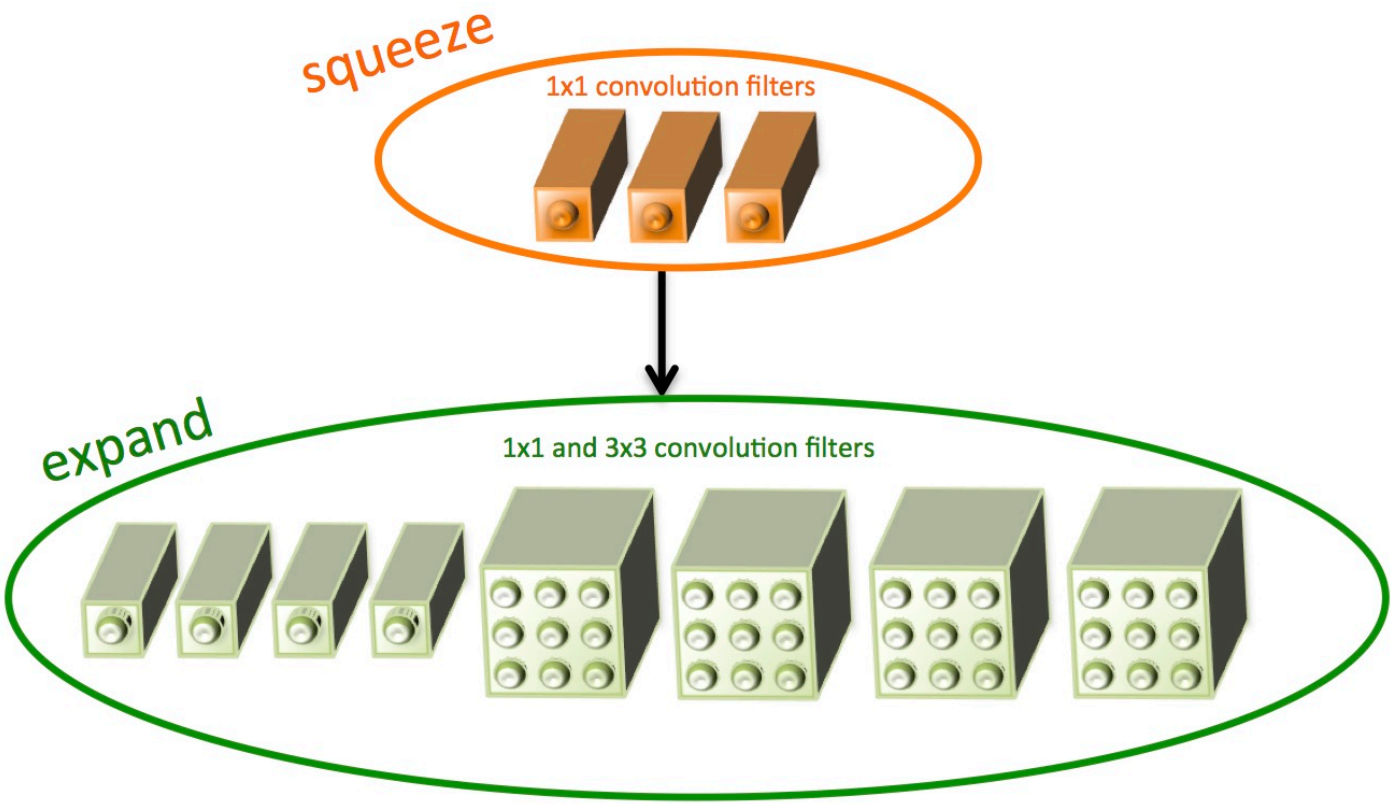
are built out of these modules:



[1] K. Simonyan, A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition arXiv:1409.1556, 2014.

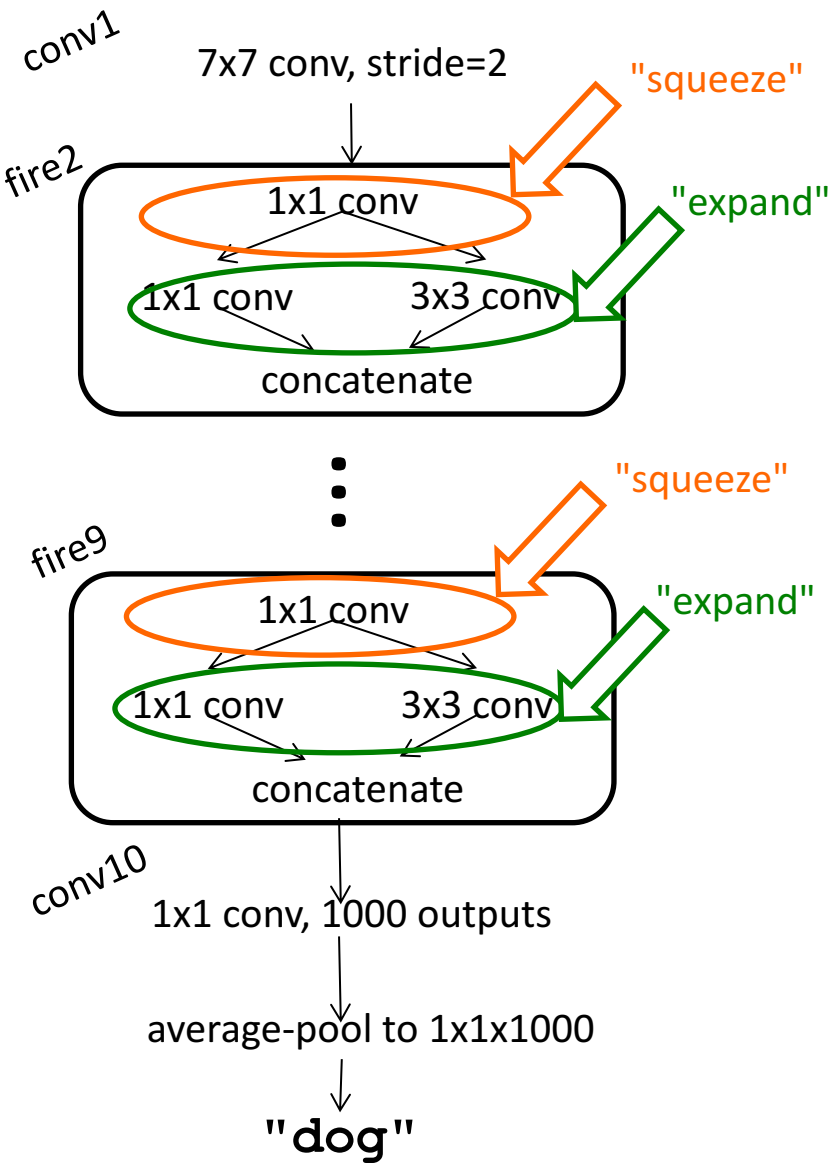
DEEPSCALE
forrest@deepscale.ai

Fire Module in Detail



DEEPSCALE
forrest@deepscale.ai

An Example FireNet CNN Architecture



Convolution

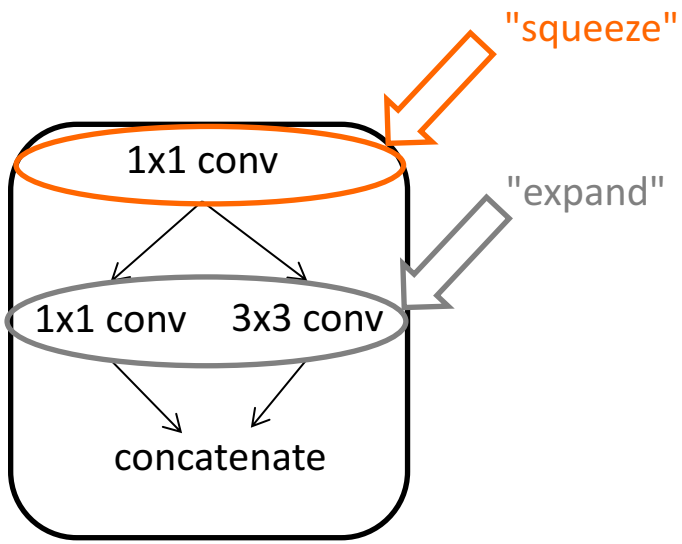
- In "expand" layers half the filters are 1x1; half the filters are 3x3

Pooling

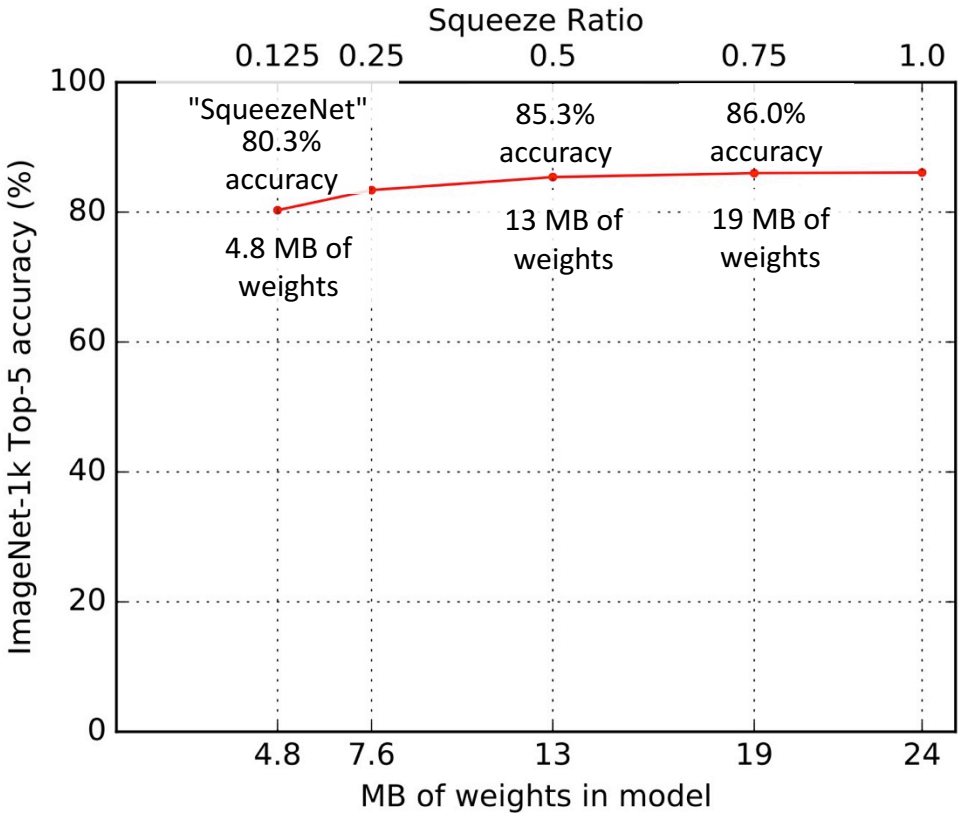
- Maxpool after conv1, fire4, fire8 (3x3 kernel, stride=2)
- Global Avgpool after conv10 down to 1x1x1000

DEEPSCALE
forrest@deepscale.ai

Tradeoffs in "squeeze" modules



- The "squeeze" modules get their name because they have fewer filters (i.e. fewer output channels) than the "expand" modules
- A natural question: what tradeoffs occur when we vary the degree of squeezing (number of filters) in the "squeeze" modules?

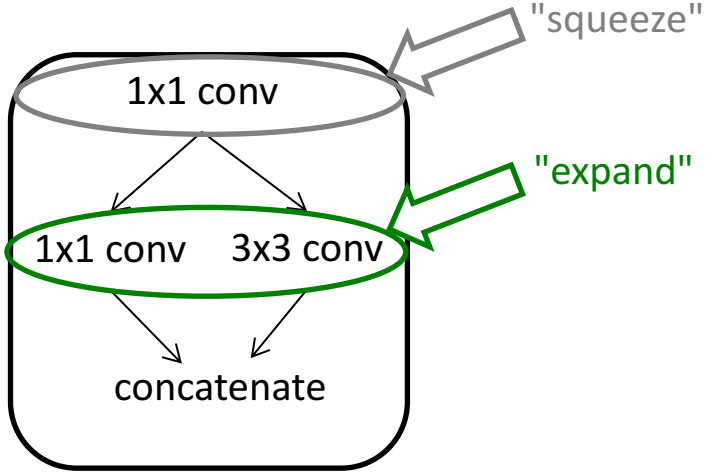


S = number of filters in "squeeze"
E = number of filters in "expand"
***Squeeze Ratio** = S/E*
for a predetermined number of filters in E

In these experiments: the *expand* modules have 50% 1x1 and 50% 3x3 filters

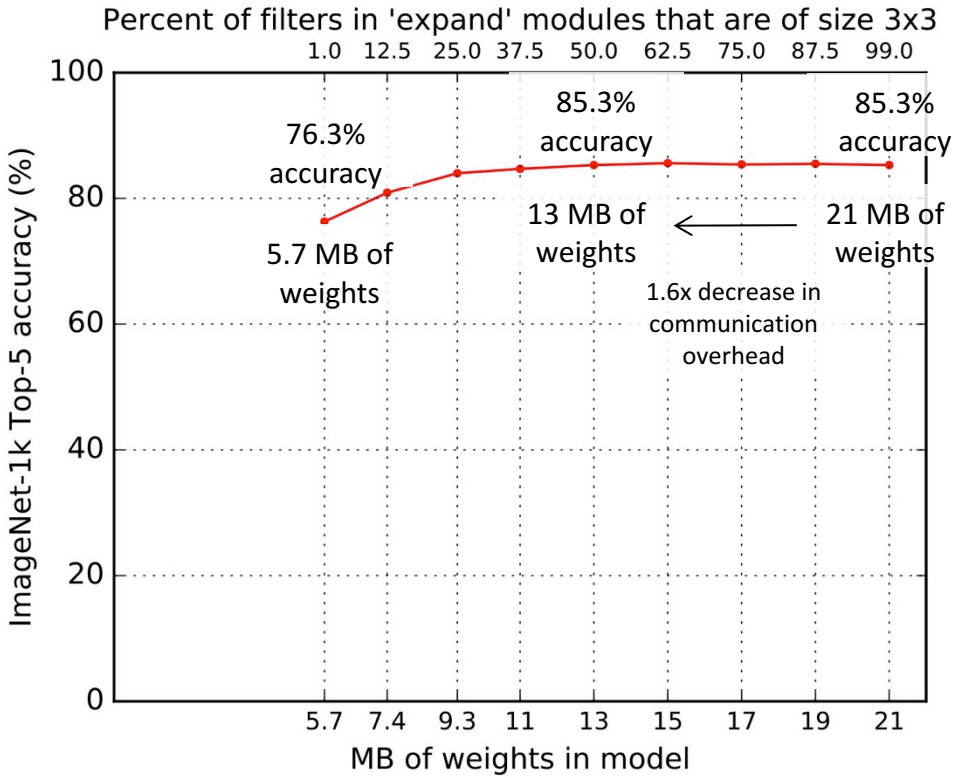
DEEPSCALE
forrest@deepscale.ai

Judiciously using 3x3 filters in "expand" modules



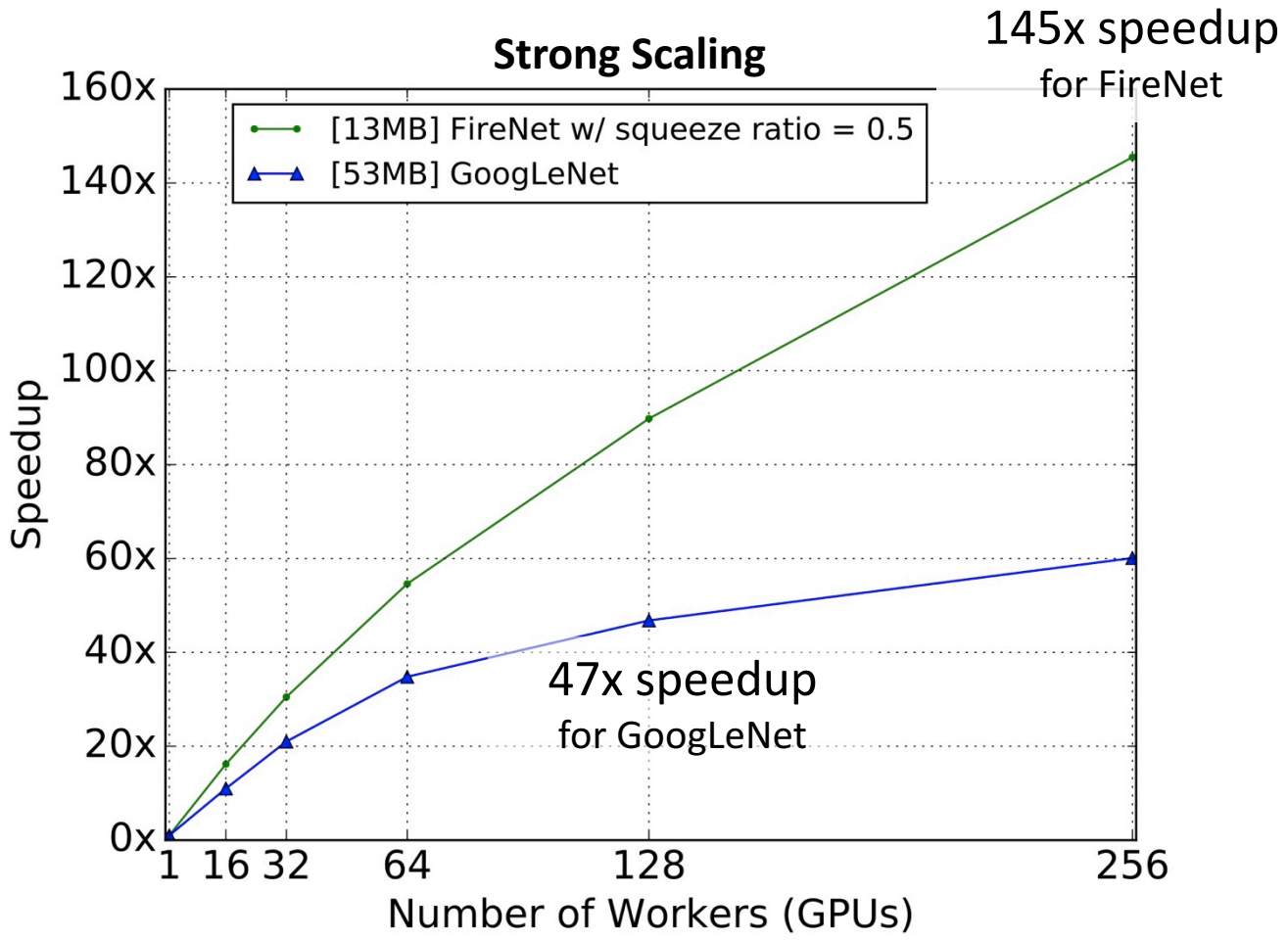
- In the "expand" modules, what are the tradeoffs when we turn the knob between mostly 1x1 and mostly 3x3 filters?
- Hypothesis: if having more weights leads to higher accuracy, then having *all* 3x3 filters should give the highest accuracy
- Discovery: accuracy plateaus with 50% 3x3 filters

Each point on this graph is the result of training a unique CNN architecture on ImageNet.



DEEPSCALE
forrest@deepscale.ai

Fewer Weights in CNN→ More Scalability in Training

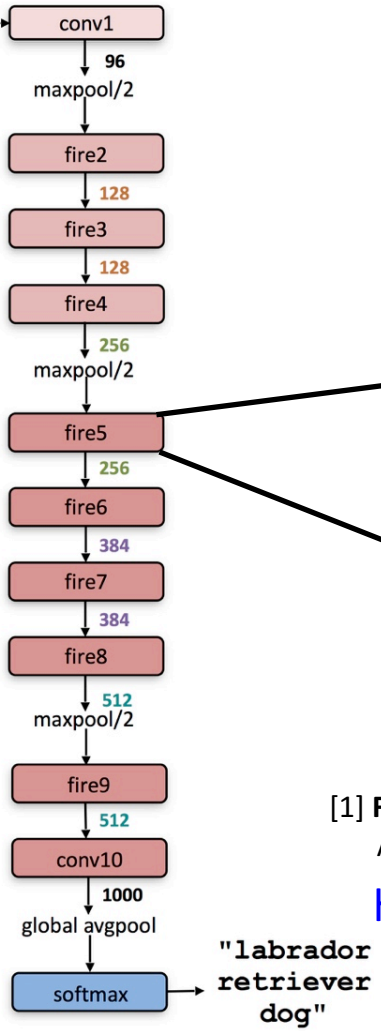


Using FireCaffe on the Titan cluster

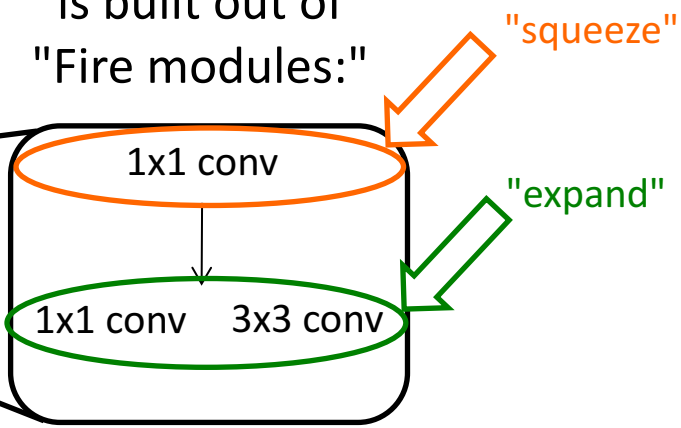
DEEPSCALE
forrest@deepscale.ai

One of our new CNN architectures: *SqueezeNet*

The SqueezeNet Architecture [1]



SqueezeNet is built out of "Fire modules:"

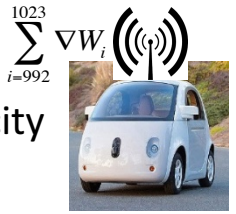


[1] **F.N. Iandola**, M. Moskewicz, K. Ashraf, S. Han, W. Dally, K. Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.

http://github.com/DeepScale/SqueezeNet

DEEPSCALE
forrest@deepscale.ai

SqueezeNet vs. Related Work



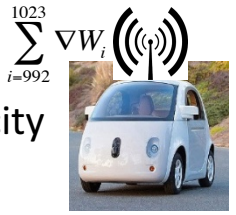
- **Small DNN models** are important if you…
  - Are deploying DNNs on devices with limited memory bandwidth or storage capacity
  - Plan to push frequent model updates to clients (e.g. self-driving cars or handsets)
- The model compression community often targets AlexNet as a DNN model to compress

Compression Approach	DNN Architecture	Original Model Size	Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
None (baseline)	AlexNet [1]	240MB	240MB	1x	57.2%	80.3%

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.
[2] E.L .Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. NIPS, 2014.
[3] S. Han, J. Pool, J. Tran, W. Dally. Learning both Weights and Connections for Efficient Neural Networks, NIPS, 2015.
[4] S. Han, H. Mao, W. Dally. Deep Compression…, arxiv:1510.00149, 2015.
[5] **F.N. Iandola**, **M. Moskewicz**, K. Ashraf, S. Han, W. Dally, **K. Keutzer**. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.

DEEPSCALE
forrest@deepscale.ai

SqueezeNet vs. Related Work



- **Small DNN models** are important if you…
  - Are deploying DNNs on devices with limited memory bandwidth or storage capacity
  - Plan to push frequent model updates to clients (e.g. self-driving cars or handsets)
- The model compression community often targets AlexNet as a DNN model to compress

Compression Approach	DNN Architecture	Original Model Size	Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
None (baseline)	AlexNet [1]	240MB	240MB	1x	57.2%	80.3%
SVD [2]	AlexNet	240MB	48MB	5x	56.0%	79.4%
Network Pruning [3]	AlexNet	240MB	27MB	9x	57.2%	80.3%
Deep Compression [4]	AlexNet	240MB	6.9MB	35x	57.2%	80.3%

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.
[2] E.L .Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. NIPS, 2014.
[3] S. Han, J. Pool, J. Tran, W. Dally. Learning both Weights and Connections for Efficient Neural Networks, NIPS, 2015.
[4] S. Han, H. Mao, W. Dally. Deep Compression…, arxiv:1510.00149, 2015.
[5] **F.N. Iandola**, **M. Moskewicz**, K. Ashraf, S. Han, W. Dally, **K. Keutzer**. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.

DEEPSCALE
forrest@deepscale.ai

SqueezeNet vs. Related Work

- **Small DNN models** are important if you…
  - Are deploying DNNs on devices with limited memory bandwidth or storage capacity
  - Plan to push frequent model updates to clients (e.g. self-driving cars or handsets)
- The model compression community often targets AlexNet as a DNN model to compress

Compression Approach	DNN Architecture	Original Model Size	Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
None (baseline)	AlexNet [1]	240MB	240MB	1x	57.2%	80.3%
SVD [2]	AlexNet	240MB	48MB	5x	56.0%	79.4%
Network Pruning [3]	AlexNet	240MB	27MB	9x	57.2%	80.3%
Deep Compression [4]	AlexNet	240MB	6.9MB	35x	57.2%	80.3%
None	**SqueezeNet [5] (ours)**	**4.8MB**	**4.8MB**	**50x**	57.5%	80.3%

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.
[2] E.L .Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. NIPS, 2014.
[3] S. Han, J. Pool, J. Tran, W. Dally. Learning both Weights and Connections for Efficient Neural Networks, NIPS, 2015.
[4] S. Han, H. Mao, W. Dally. Deep Compression…, arxiv:1510.00149, 2015.
[5] **F.N. Iandola**, **M. Moskewicz**, K. Ashraf, S. Han, W. Dally, **K. Keutzer**. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.

DEEPSCALE
forrest@deepscale.ai

SqueezeNet vs. Related Work

- **Small DNN models** are important if you…
  - Are deploying DNNs on devices with limited memory bandwidth or storage capacity
  - Plan to push frequent model updates to clients (e.g. self-driving cars or handsets)
- The model compression community often targets AlexNet as a DNN model to compress

Compression Approach	DNN Architecture	Original Model Size	Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
None (baseline)	AlexNet [1]	240MB	240MB	1x	57.2%	80.3%
SVD [2]	AlexNet	240MB	48MB	5x	56.0%	79.4%
Network Pruning [3]	AlexNet	240MB	27MB			.3%
Deep Compression [4]	AlexNet					80.3%
None	**S**		**..8MB**	**50x**	57.5%	80.3%
Deep Compression [4]	**SqueezeNet [5] (ours)**	**4.8MB**	**0.47MB**	**510x**	57.5%	80.3%

By combining CNN architectural innovation with model compression, we achieved a **510x** reduction in model size.

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.
[2] E.L .Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. NIPS, 2014.
[3] S. Han, J. Pool, J. Tran, W. Dally. Learning both Weights and Connections for Efficient Neural Networks, NIPS, 2015.
[4] S. Han, H. Mao, W. Dally. Deep Compression…, arxiv:1510.00149, 2015.
[5] **F.N. Iandola**, **M. Moskewicz**, K. Ashraf, S. Han, W. Dally, **K. Keutzer**. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. arXiv, 2016.

DEEPSCALE
forrest@deepscale.ai

Broader Takeaways

- Neural networks comprise an infinite design space (number of layers, type of layers, dimensions of layers)
- There is nothing particularly special about today's popular DNN architectures (e.g. AlexNet, GoogLeNet, etc)

Therefore…

- Don't: Waste your time tuning your hardware to accelerate AlexNet
- Do: In addition to being a software and hardware architect, become a neural network architect!

- Don't: Design your hardware around a specific DNN
- Do: Design DNN around the strengths and limitations of your hardware (this is easier and cheaper)

DEEPSCALE
forrest@deepscale.ai

DEEPSCALE

Perception for autonomous vehicles

We're hiring for:

- Embedded Software Development
- Deep Learning
- RADAR Software Engineering

`http://deepscale.ai/jobs`

DEEPSCALE
forrest@deepscale.ai

Extras

DEEPSCALE
forrest@deepscale.ai

Adapting SqueezeNet for object detection

- We originally trained SqueezeNet on the problem of *object classification*
- The highest-accuracy results on *object detection* involve pre-training a CNN on *object classification* and then fine-tuning it for *object detection*
  - Modern approaches (e.g. Faster R-CNN, YOLO) typically modify design of the CNN's final layers so that it can *localize* as well as *classify* objects

Next: Let's give this a try with SqueezeNet!

DEEPSCALE
forrest@deepscale.ai

Adapting SqueezeNet for object detection

Method	Average Precision on KITTI [1] pedestrian detection	Mean Average Precison on KITTI [1] object detection dataset	Model Size	Speed (FPS) on Titan X GPU
MS-CNN [2]	85.0	78.5	-	2.5 FPS
Pie [3]	84.2	69.6	-	10 FPS
Shallow Faster R-CNN [4]	82.6	-	240 MB	2.9 FPS
SqueezeDet [5] (ours)	82.9	76.7	**7.90 MB**	**57.2 FPS**
SqueezeDet+ [5] (ours)	**85.4**	**80.4**	26.8 MB	32.1 FPS

[1] A. Geiger, P. Lenz, R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. CVPR, 2012
[2] Z. Cai, Q. Fan, R. Feris, N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. ECCV, 2016.
[3] Anonymous submission on the KITTI leaderboard.
[4] K. Ashraf, B. Wu, **F.N. Iandola**, M.W. Moskewicz, K. Keutzer. **Shallow Networks** for High-Accuracy Road Object-Detection. arXiv:1606.01561, 2016.
[5] Bichen Wu, **Forrest Iandola**, Peter Jin, Kurt Keutzer, "**SqueezeDet**: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving," In Review, 2016.