SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size

F. Iandola, S. Han, M. Moskewicz, K. Ashraf, W. Dally, K. Keutzer
Stanford and UC Berkeley
April 2016

Why make smaller nets?

1. Less distribution overhead
    - Easier distribution of updates into the field
    - More efficient distributed training
2. Feasible to deploy in memory-limited systems (eg. Robots (also FPGAs))
3. Feasible to run on compute-limited systems (eg. Robots)

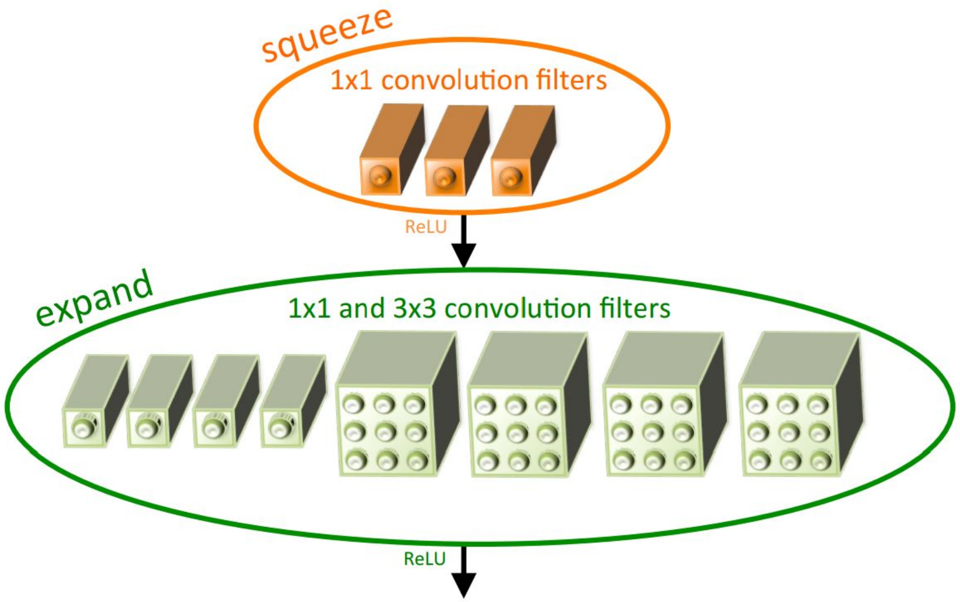
But can accuracy be maintained while reducing net size?

Parameter Reduction Strategies

1. Use mostly 1x1 convolution kernels
   - 1x1 kernel has 9x fewer parameters than 3x3
2. Decrease number of channels in layers before >1x1 layers
   - This limits the number of inputs to large-area filters
3. Downsample late in the network
   - This maintains more spatial information in more of the network at runtime
   - Has been demonstrated to increase network accuracy in other works

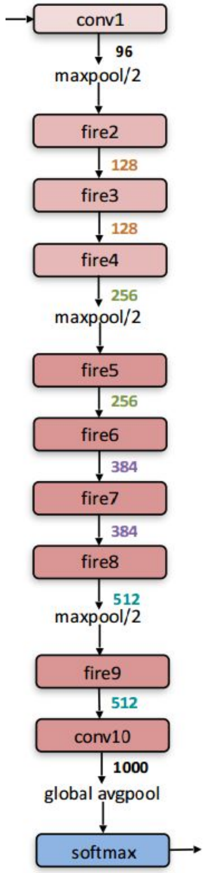
Fire Module

- A layer of 1x1 filter followed by a layer of 1x1 and 3x3 filters
- 3 parameters:
  - Number of 1x1 filters in squeeze layer: s_{1x1}
  - Number of 1x1 filters in expand layer: e_{1x1}
  - Number of 3x3 filters in expand layer: e_{3x3}
- Per strategy 2, they set: $s_{1x1} < e_{1x1} + e_{3x3}$



Macro-architecture

- 224x224x3 input images
- One 3x3 conv+ReLU+maxpool/2 module (96 filters)
- T9 fire modules, 2-stride max-pooling after modules 4 and 8
    - 128 filters in first module, goes up by 128 every two modules
- Then one 1x1 conv+ReLU module (1 filter/class)
- Finally, global average pooling, then softmax



AlexNet

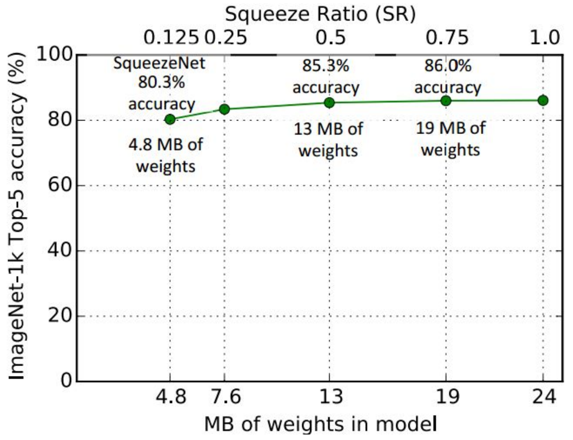
- 5 convolutional layers
  - 11x11, 5x5, and 3x3 kernels
- 3 fully-connected layers
- 240 MB model w/ 32-bit weights
  - Approximately 60 million parameters
- Top-1 ImageNet accuracy: 57.2%
- Top-5 ImageNet accuracy: 80.3%

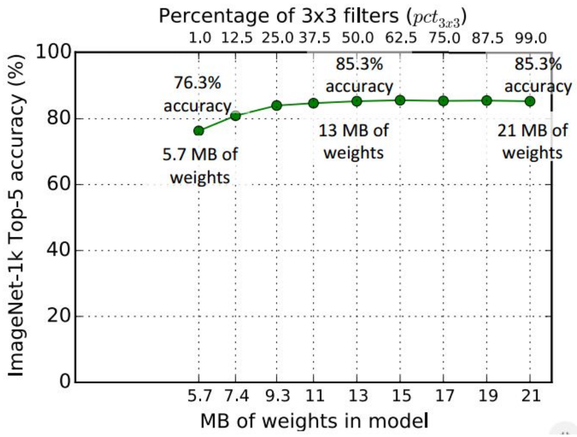
SqueezeNet

- 18 convolutional layers
  - 1x1 and 3x3 kernels
- **Zero** fully-connected layers
- 4.8 MB model w/ 32-bit weights (50x less)
  - Approximately 1.2 million parameters
- Top-1 ImageNet accuracy: **57.5%**
- Top-5 ImageNet accuracy: 80.3%
- With deep compression, 0.47 MB model, and no decline in accuracy!

Architectural experiments

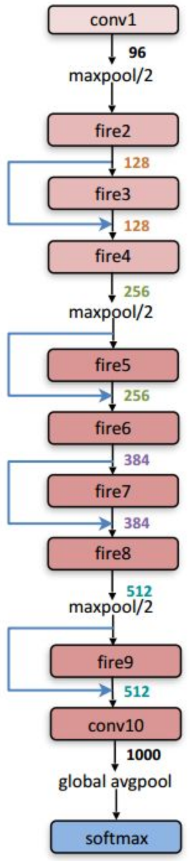
- Modified ratio of # squeeze filters to # expand filters (from 0.125 to 1)
  - Top-5 Accuracy plateaus at 86.0% for ratio of 0.75, model size 19 MB
- Modified ratio of 1x1 to 3x3 filters in expand layers (holding sqeeze:expand ratio at 0.5)
  - Top-5 accuracy plateaus at 85.3% with 50% 3x3 filters, model size 13 MB
  - In the prior experiment, ratio was held at 50%, so max. accuracy across these parameters is 86.0%

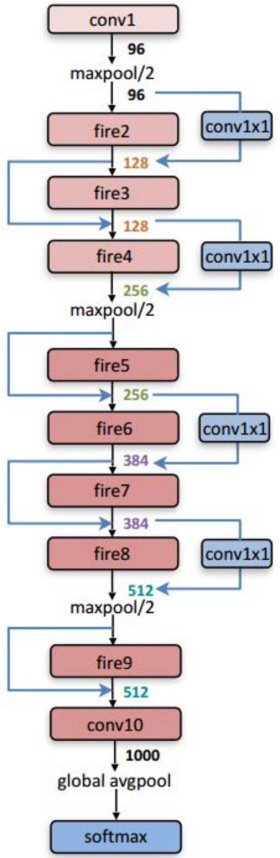




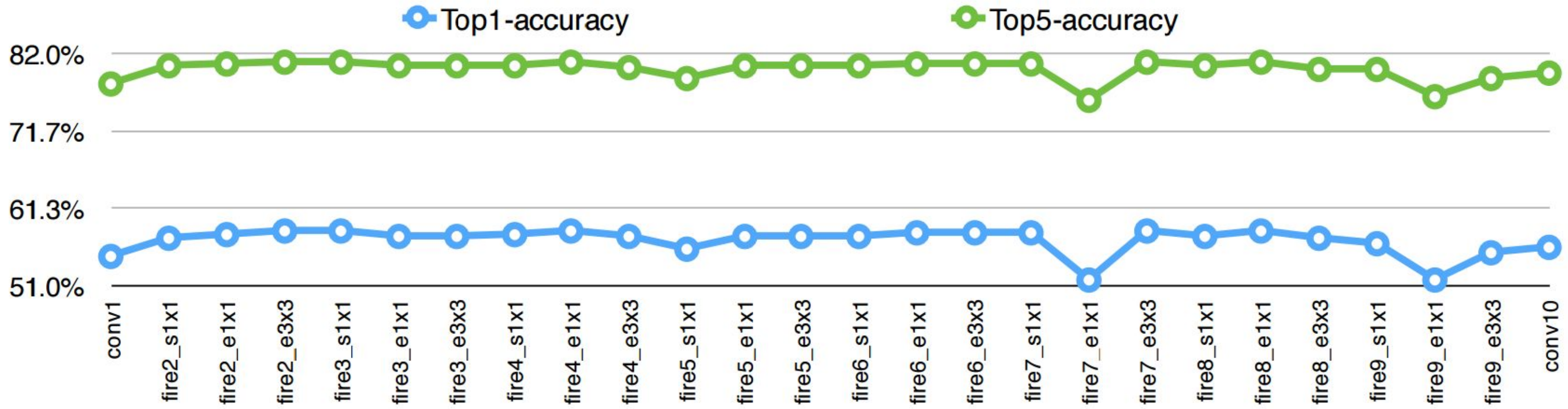
Architectural experiments

- Following ResNet, added simple bypasses between fire modules with same # of filters
  - Gave top-1 accuracy of 60.4%, top-5 accuracy of 82.5%, no change in model size
  - Allows info to flow “around” the squeeze layers?
- Also added convolutional bypasses between modules with different # of filters
  - Gave top-1 accuracy of 58.8%, top-5 accuracy of 82.0%, increased model size to 7.7 MB
  - Better than no bypasses, but worse than only simple bypasses! No analysis offered...





Compression experiments



- Pruned smallest 50% weights for each layer in isolation, measured accuracy
- 1x1 layers more sensitive than 3x3 layers
- This informed their compression scheme: dropped more weights from 3x3 filters
- Adding more channels to most sensitive layers gives 1-2% accuracy boost, 48% increase in model size

Dense-Sparse-Dense Training

- After training, prune 2/3rds of weights according to the final compression scheme
- Reinitialize pruned weights to 0 and continue training another 20 epochs
- Top-1 accuracy +4.3% improvement
- Top-5 accuracy +3.2% improvement
- Hypothesis: enforcing sparsity midway through acts as a form of regularization
- May be applicable to CNNs more broadly, not just SqueezeNet

What do I like about this paper?

- Lots of interesting ideas
- Exploring reduction while maintaining quality, rather than expansion to increase quality
- Scientific approach to exploring the very large parameter space

Conclusions and questions

- Most of the techniques for improving accuracy significantly increase network size
    - Exceptions: simple bypass, dense-sparse-dense training
- Accuracy/size tradeoff can be tuned to the application

- How does making the network deeper affect accuracy? Size?
- What are the time-performance effects of the techniques?

Time performance

- Experiments run using Caffe
- GPU: NVidia GTX 960, 4 GB memory, 1.127 GHz, 1024 CUDA cores
- CPU: Intel Core i7-5820K, 3.30 GHz, 6 double-threaded cores

Architecture	Model size	GPU forward time	CPU forward time
AlexNet	233 MB	4.78 ms = 209 Hz	267.9 ms = 3.73 Hz
ResNet-50	98 MB	34.29 ms = 29 Hz	443.4 ms = 2.26 Hz
SqueezeNet v1.1	4.8 MB	**3.80 ms = 263 Hz**	**58.8 ms = 17.0 Hz**