NVIDIA

PRUNING CONVOLUTIONAL NEURAL NETWORKS

Pavlo Molchanov

Stephen Tyree

Tero Karras

Timo Aila

Jan Kautz

GPU TECHNOLOGY CONFERENCE 2017

WHY WE CAN PRUNE CNNS?

WHY WE CAN PRUNE CNNS?

Optimization “failures”:

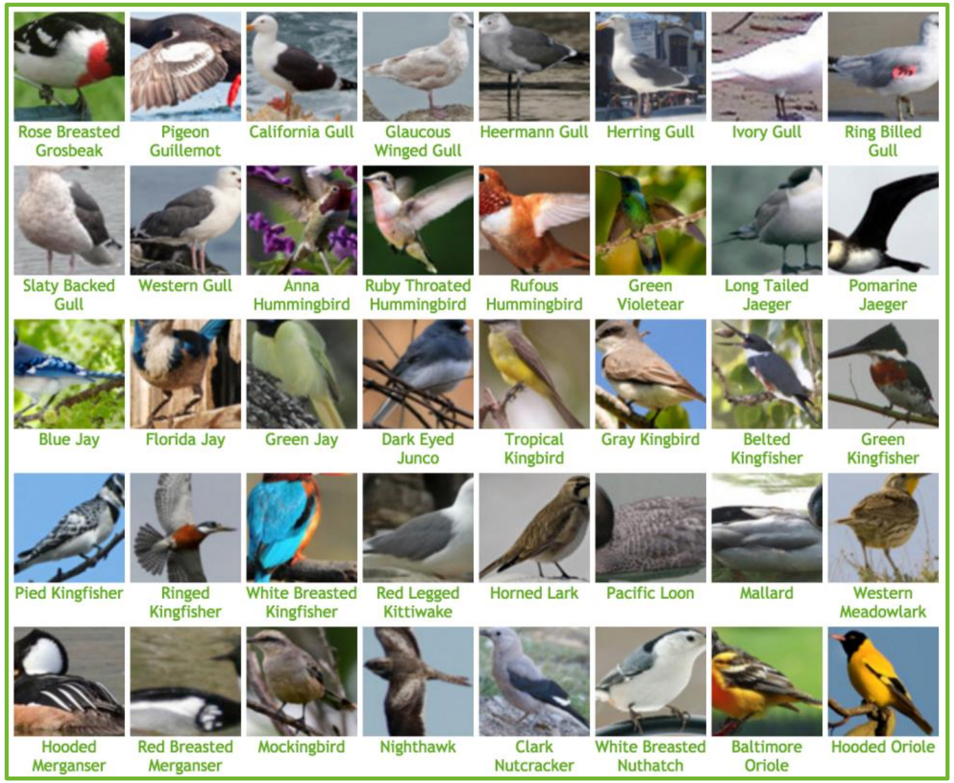
- Some neurons are "dead": little activation
- Some neurons are uncorrelated with output

Modern CNNs are overparameterized:

- **VGG16** has **138M** parameters
- **Alexnet** has **61M** parameters
- **ImageNet** has **1.2M** images

PRUNING FOR TRANSFER LEARNING

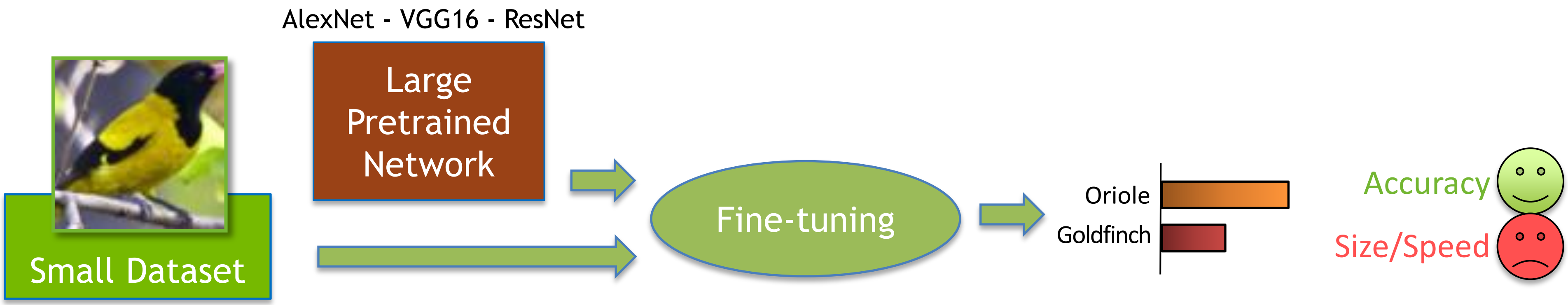
Small Dataset



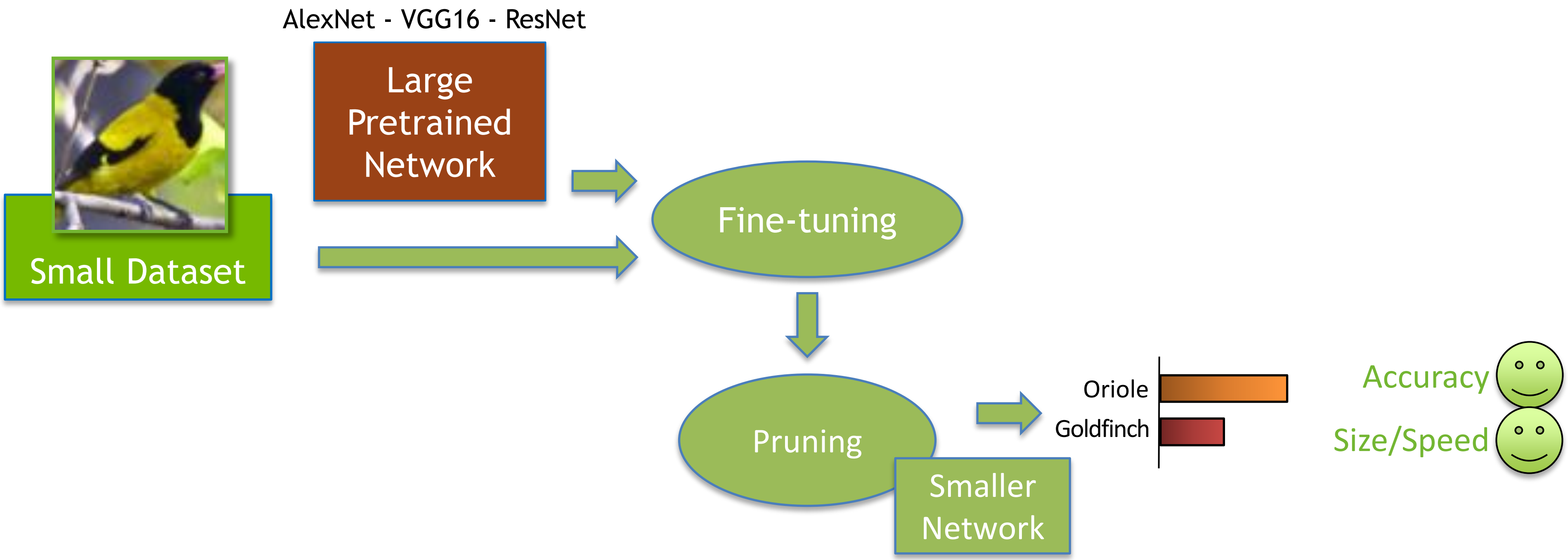
Caltech-UCSD Birds (200 classes, <6000 images)

PRUNING FOR TRANSFER LEARNING

PRUNING FOR TRANSFER LEARNING



PRUNING FOR TRANSFER LEARNING



TYPES OF UNITS

- Convolutional units
  - **Heavy on computation**
  - Small on storage
- Fully connected (dense) units
  - Fast on computations
  - Heavy on storage

Our focus

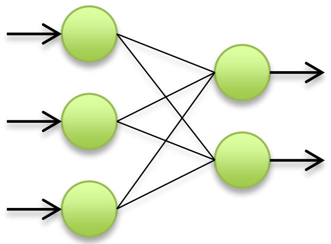
Ratio of floating point operations

	Convolutional layers	Fully connected layers
VGG16	99%	1%
Alexnet	89%	11%
R3DCNN	90%	10%

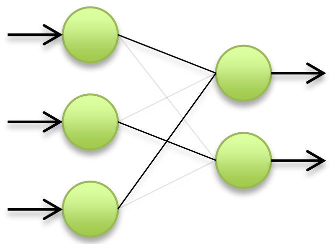
To reduce computation, we focus pruning on convolutional units.

TYPES OF PRUNING

No pruning

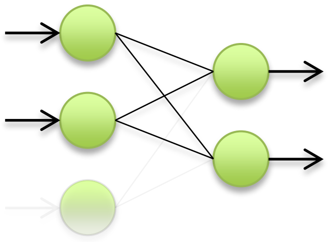


Fine pruning



- Remove connections between neurons/feature maps
- May require special SW/HW for full speed-up

Coarse pruning



- Remove entire neurons/feature maps
- Instant speed-up
- No change to HW/SW

Our focus

NETWORK PRUNING

NETWORK PRUNING

Training:

$$\min_{W} C(W, \mathcal{D})$$

C: training cost function

$\mathcal{D}$: training data

W: network weights

$\widehat{W}$: *pruned* network weights

NETWORK PRUNING

Training:

$$\min_{W} C(W, \mathcal{D})$$

C: training cost function

$\mathcal{D}$: training data

W: network weights

$\widehat{W}$: *pruned* network weights

Pruning:

$$\min_{\widehat{W}} \left| C(\widehat{W}, \mathcal{D}) - C(W, \mathcal{D}) \right|$$

$$s.t.\ \widehat{W} \subset W, \left|\widehat{W}\right| < B$$

NETWORK PRUNING

Training:

$$\min_{W} C(W, \mathcal{D})$$

C: training cost function
$\mathcal{D}$: training data
W: network weights
$\widehat{W}$: *pruned* network weights

Pruning:

$$\min_{\widehat{W}} \left| C(\widehat{W}, \mathcal{D}) - C(W, \mathcal{D}) \right|$$

$$\cancel{s.t.\ \widehat{W} \subset W, |\widehat{W}| < B}$$

$$s.t.\ \|\widehat{W}\|_0 \leq B$$

$\|\cdot\|_0$ $- \ell_0$ norm, number of non zero elements

NETWORK PRUNING

Exact solution: combinatorial optimization problem – too computationally expensive

- VGG-16 has $|W| = 4224$ convolutional units

2^W=3553871205531788502027616705177895234326962283811349000683834453551638494934980826570988629674816508671333937997942971545498 563185784012615902725922028388957693142
1862796735241131771064707150729404513525374011172364491439311003809147986212244125583682040173009664289254204672705377527023751838969121362871174353608981432683121364
5491611587700632287226757360106388212811709391049243449409694131581866174894684285426551148222434459277138467708468356441728767115601429026774386653664558802884798090
6965876098883394994207765939795994221495102245529321358133169053471175098438846379813927963588224649996889912395677448659534869881828474761387469462375439163452354234
5894518795402778976197641675203085270364961383790287738178866981707575145292010325953635643917893687322226855341345293028465563634475713300900704784609781200491091266
5177085470491781920811732083028359068442910422663939383012657211605418802586239081536469961410441163264284259407567601349688157128480106842375724875121706906188815680
8417681026874596048633568575893047553712713299830093139608694750348505494684606129671946123873358658490052333372765817334544824122023280282312402650277313912908677267
4199580978427901948940349864646863071403137640248862807464745563583993330788235800894899276294310469436651968921 5

NVIDIA

NETWORK PRUNING

Exact solution: combinatorial optimization problem – too computationally expensive

- VGG-16 has $|W| = 4224$ convolutional units

2^W=3553871205531788502027616705177895234326962283811349000683834453551638494934980826570988629674816508671333937997942971545498 563185784012615902725922028388957693142
1862796735241131771064707150729404513525374011172364491439311003809147986212244125583682040173009664289254204672705377527023751838969121362871174353608981432683121364
5491611587700632287226757360106388212811709391049243449409694131581866174894684285426551148222434459277138467708468356441728767115601429026774386653664558802884798090
6965876098883394994207765939795994221495102245529321358133169053471175098438846379813927963588224649996889912395677448659534869881828474761387469462375439163452354234
5894518795402778976197641675203085270364961383790287738178866981707575145292010325953635643917893687322226855341345293028465563634475713300900704784609781200491091266
5177085470491781920811732083028359068442910422663939383012657211605418802586239081536469961410441163264284259407567601349688157128480106842375724875121706906188815680
8417681026874596048633568575893047553712713299830093139608694750348505494684606129671946123873358658490052333372765817334544824122023280282312402650277313912908677267
41995809784279019489403498646468630714031376402488628074647455635839933307882358008948992762943104694366519689215

Greedy pruning:

- Assumes all neurons are independent
  (same assumption for back propagation)

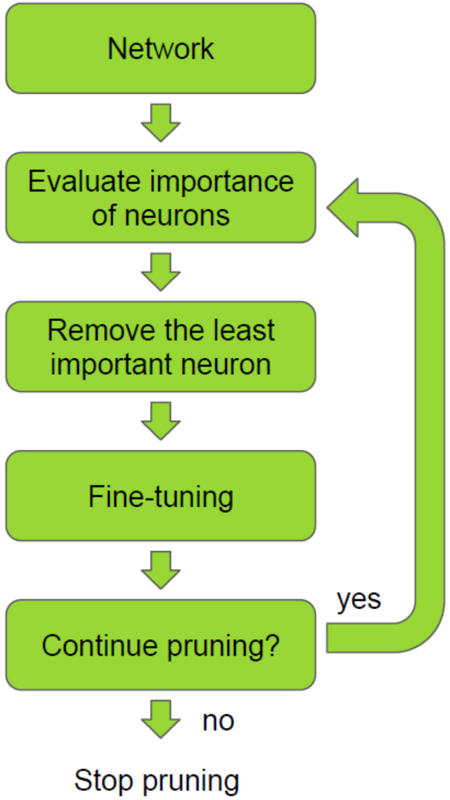
- Iteratively, remove neuron with the smallest contribution

GREEDY NETWORK PRUNING

Iterative pruning

Algorithm:

1) Estimate importance of neurons (units)
2) Rank units
3) Remove the least important unit
4) Fine tune network for K iterations
5) Go back to step1)



ORACLE

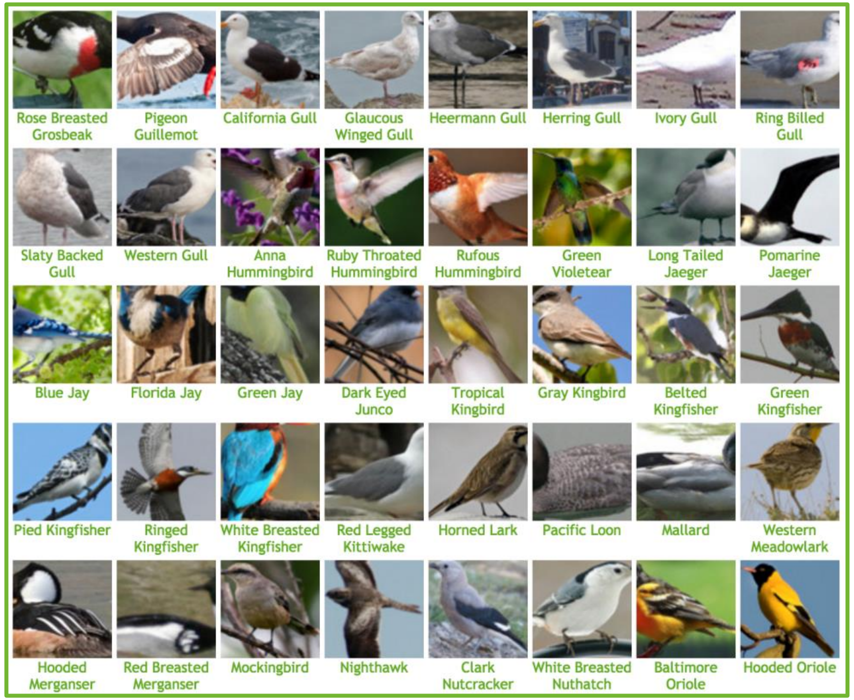
ORACLE

Caltech-UCSD Birds-200-2011 Dataset

- 200 classes
- <6000 training images

	Method	Test accuracy
S. Belongie et al	*SIFT+SVM	19%
From scratch	CNN	25%
S. Razavian et al	*OverFeat+SVM	62%
Our baseline	**VGG16 finetuned**	**72.2%**
N. Zhang et al	R-CNN	74%
S. Branson et al	*Pose-CNN	76%
J. Krause et al	*R-CNN+	82%

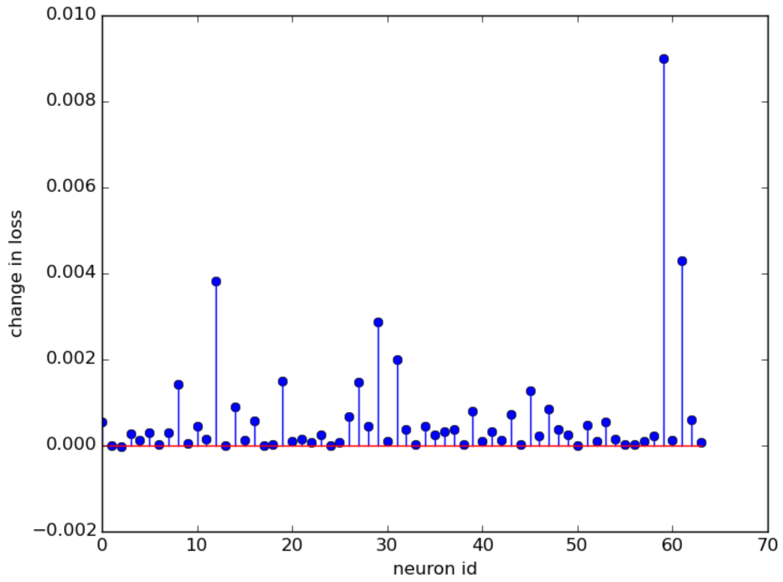
*require additional attributes



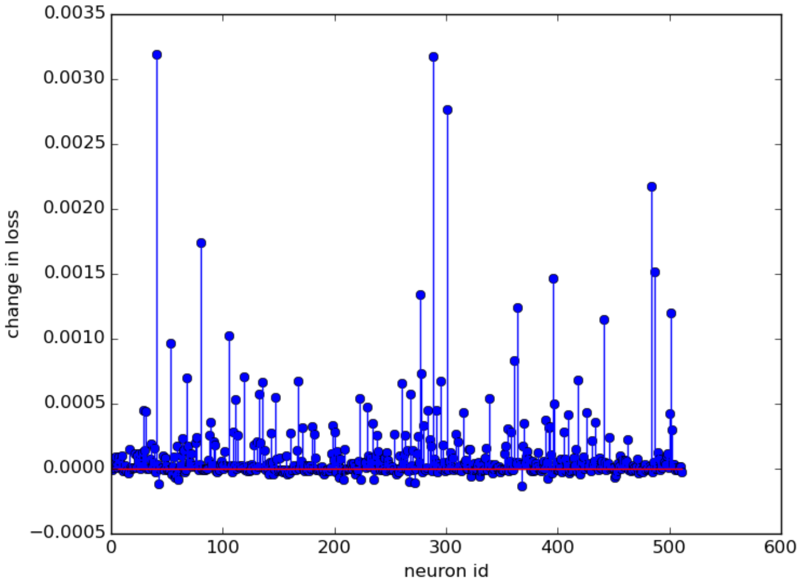
ORACLE

VGG16 on Birds-200 dataset

- Exhaustively computed change in loss by removing one unit



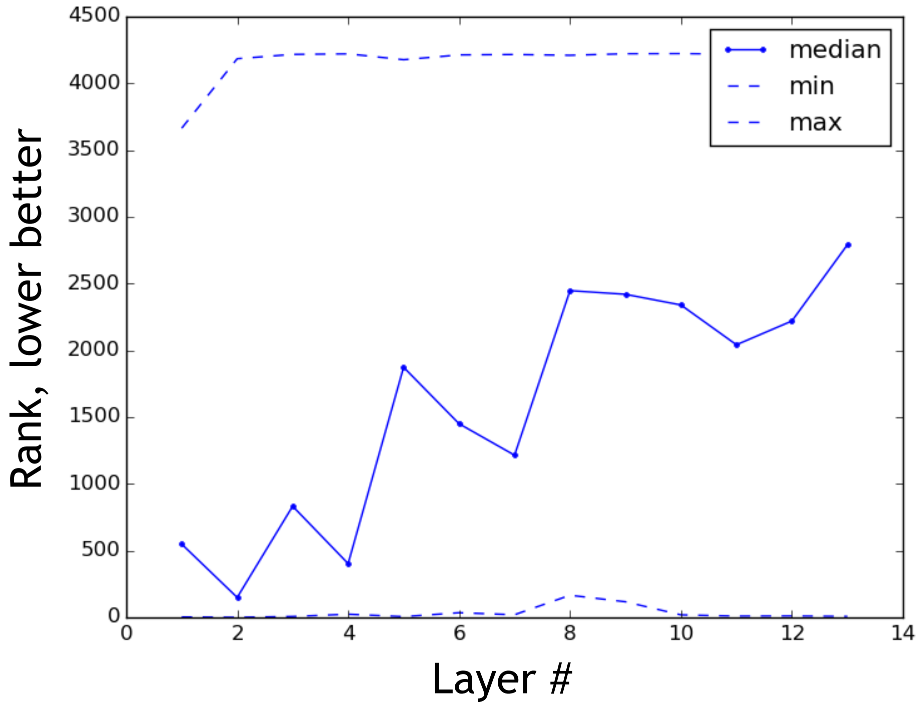
First layer



Last layer

ORACLE

VGG-16 on Birds-200



*only convolutional layers

- On average first layers are more important
- Every layer has **very important** units
- Every layer has **non important** units
- Layers with pooling are more important

APPROXIMATING THE ORACLE

APPROXIMATING THE ORACLE

Candidate criteria

- Average activation (discard lower activations)
- Minimum weight (discard lower l_2 of weight)
- With first-order Taylor expansion (TE):

$$\mathcal{C}(\mathcal{D}, h_i = 0) \quad = \quad \mathcal{C}(\mathcal{D}, h_i) - \frac{\delta \mathcal{C}}{\delta h_i} h_i + \underbrace{R_1(h_i = 0)}_{\text{ignore}}$$

Absolute difference in cost by removing a neuron:

$$\left|\Delta \mathcal{C}(h_i)\right| = \left|\mathcal{C}(\mathcal{D}, h_i) - \frac{\delta \mathcal{C}}{\delta h_i} h_i - \mathcal{C}(\mathcal{D}, h_i)\right| = \left|\frac{\delta \mathcal{C}}{\delta h_i} h_i\right|$$

$\frac{\delta \mathcal{C}}{\delta h_i}$	Gradient of the cost wrt. activation h_i
h_i	Unit's output
$+$	Both computed during standard backprop.

NVIDIA.

APPROXIMATING THE ORACLE

Candidate criteria

- Alternative: Optimal Brain Damage (OBD) by Y. LeCun et al., 1990
  - Use second order derivatives to estimate importance of neurons:

$$\Delta C(h_i) = \underbrace{\frac{\delta C}{\delta h_i}}_{=0} h_i + 0.5\frac{\delta^2 C}{\delta h_i^2} + \underbrace{R_2(h_i = 0)}_{\text{ignore}}$$

− Needs extra comp of second order derivative

APPROXIMATING THE ORACLE

Comparison to OBD

OBD: second-order expansion:

$$\Delta C(h_i) = \underbrace{\frac{\delta C}{\delta h_i}}_{=0} h_i + 0.5 \frac{\delta^2 C}{\delta h_i^2}$$

we propose: abs of first-order expansion:

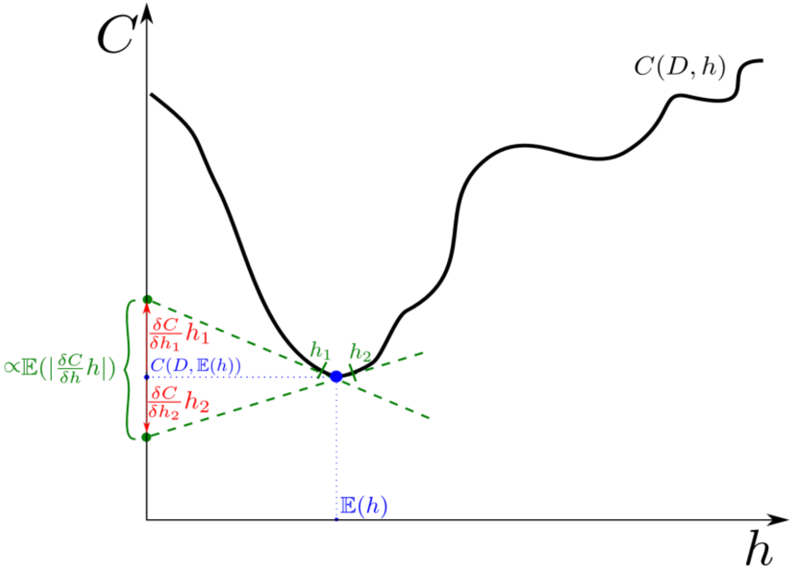
$$\left|\Delta C(h_i)\right| = \left|\frac{\delta C}{\delta h_i} h_i\right|$$

Assuming $y = \frac{\delta C}{\delta h_i} h_i$

For perfectly trained model:

$$\mathbb{E}(y) = 0$$

$$\mathbb{E}(|y|) = std(y)\sqrt{2}/\sqrt{\pi}$$ if y is Gaussian



APPROXIMATING THE ORACLE

Comparison to OBD

OBD: second-order expansion:

$$\Delta C(h_i) = \underbrace{\frac{\delta C}{\delta h_i}}_{=0} h_i + 0.5 \frac{\delta^2 C}{\delta h_i^2}$$

we propose: abs of first-order expansion:

$$\left|\Delta \mathcal{C}(h_i)\right| = \left|\frac{\delta \mathcal{C}}{\delta h_i} h_i\right|$$

Assuming $y = \frac{\delta C}{\delta h_i} h_i$

For perfectly trained model:

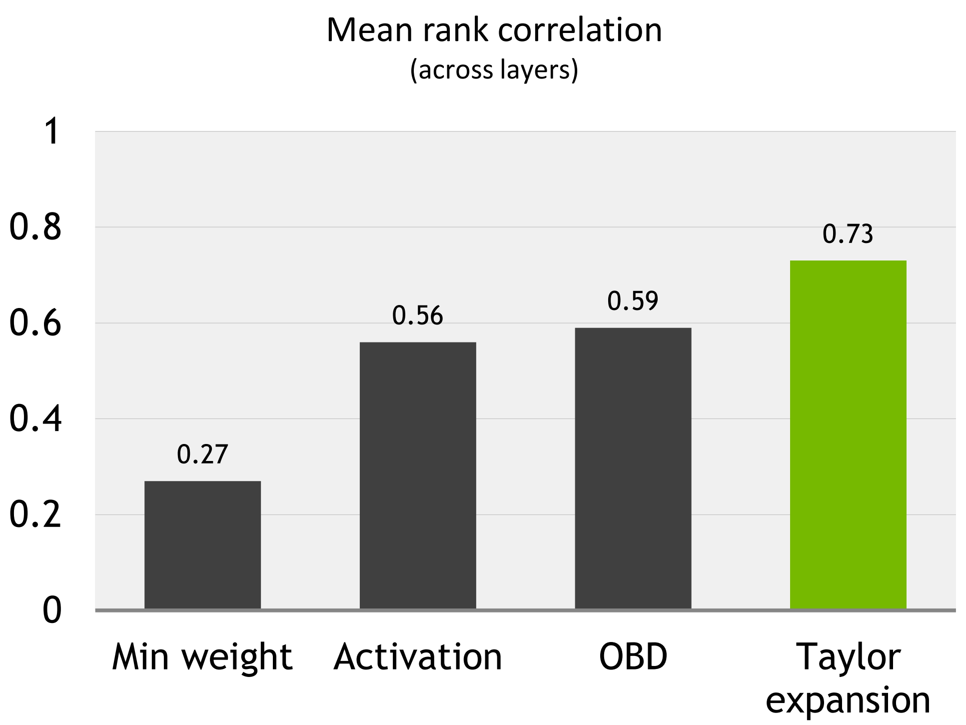
$$\mathbb{E}(y) = 0$$

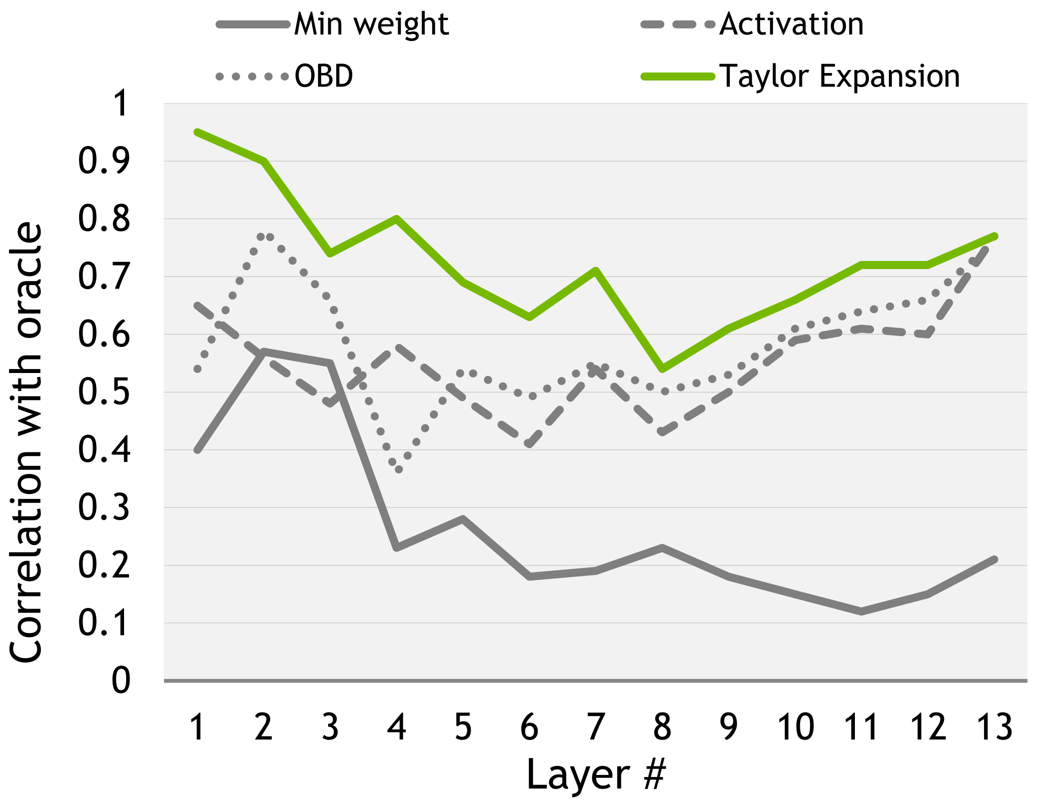
$$\mathbb{E}(|y|) = std(y)\sqrt{2}/\sqrt{\pi}$$ if y is Gaussian

- ✓ No extra computations
- ✓ We look at absolute difference
- — Can’t predict exact change in loss

EVALUATING PRUNING CRITERIA

Spearman's rank correlation with oracle: VGG16 on Birds-200





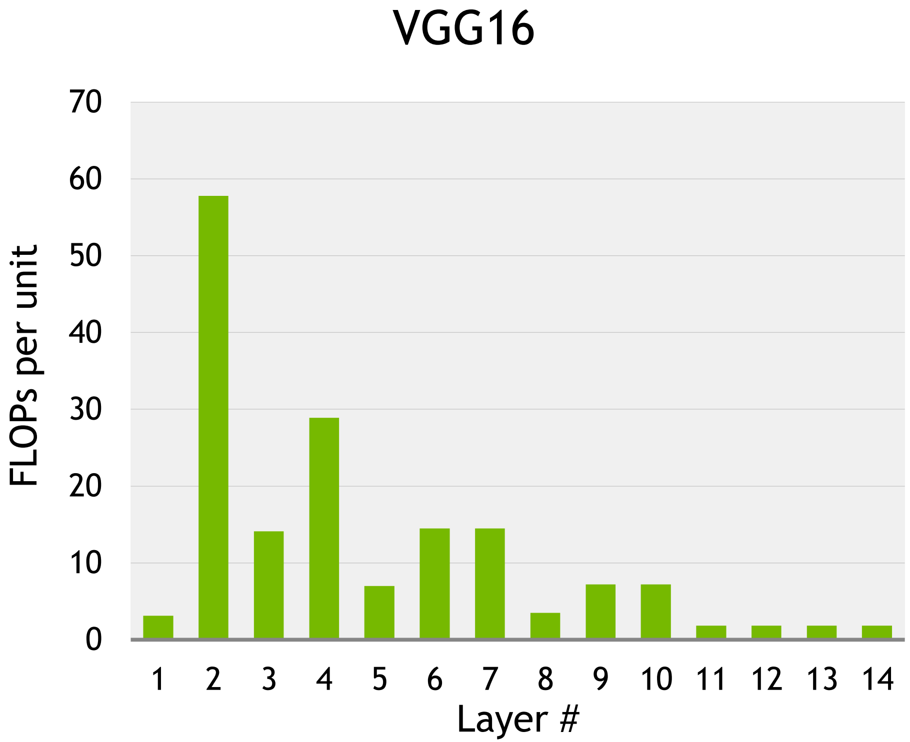
EVALUATING PRUNING CRITERIA

Pruning with objective

- Regularize criteria with objective:

$$\Theta(h) = \Theta(h) - \lambda\mathcal{R}(h)$$

- Regularizer can be:
  - FLOPs
  - Memory
  - Bandwidth
  - Target device
  - Exact inference time

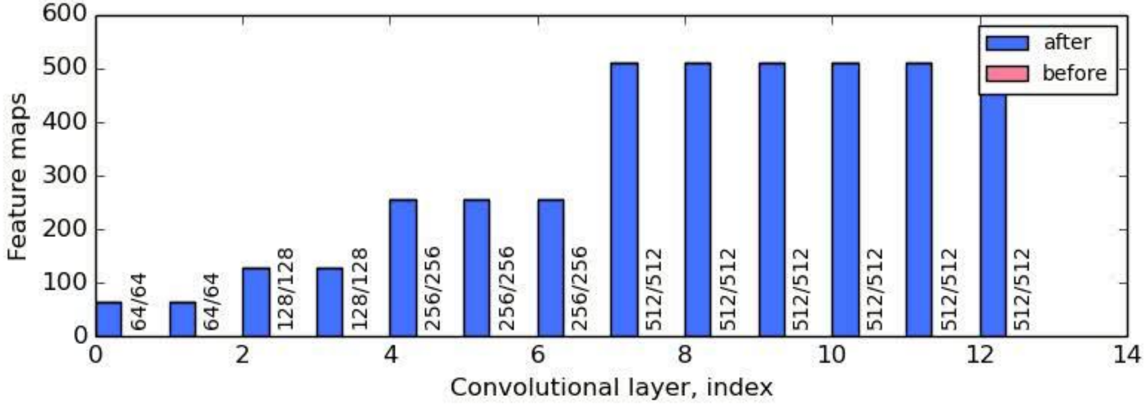


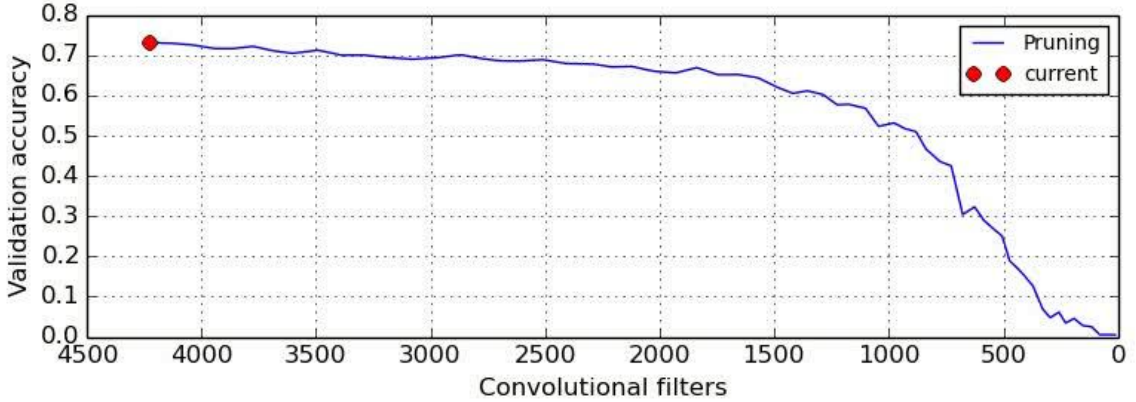
RESULTS

RESULTS

VGG16 on Birds 200 dataset

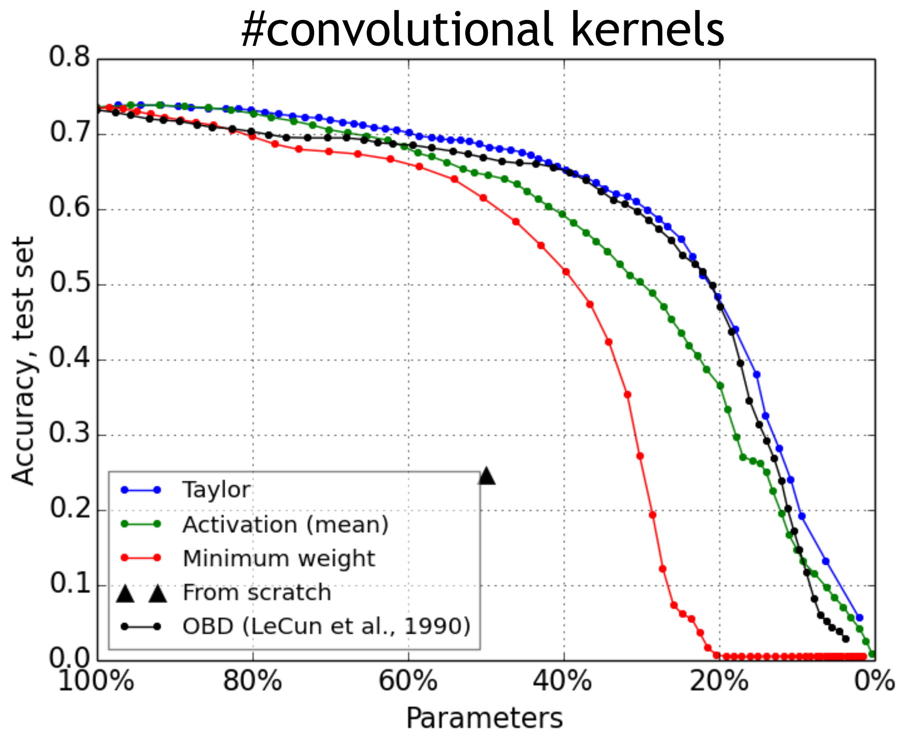
- Remove 1 conv unit every 30 updates

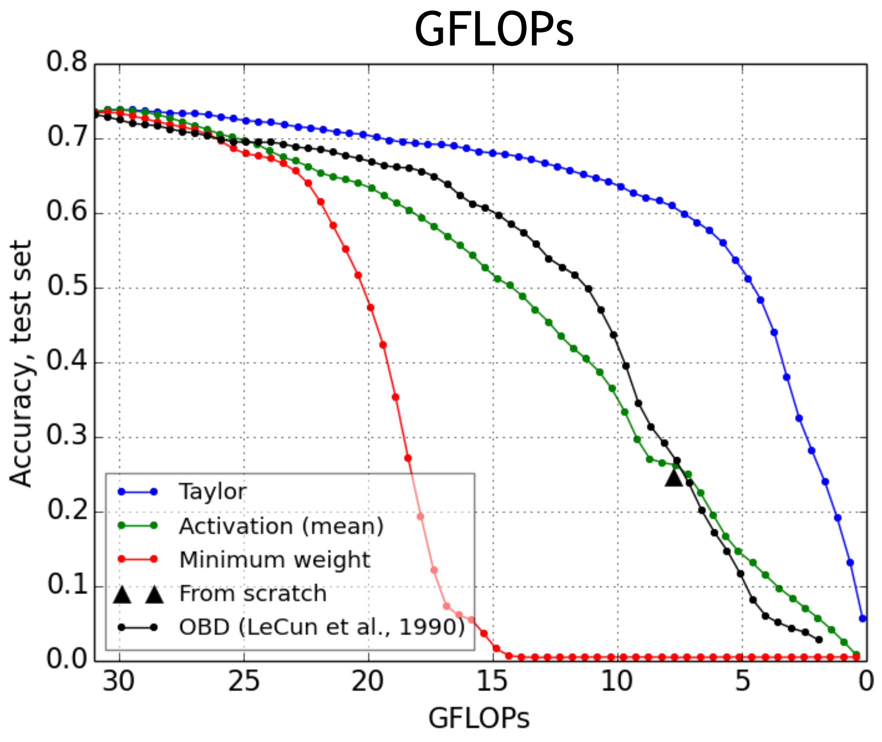




RESULTS

VGG16 on Birds 200 dataset





- Training from scratch doesn't work
- Taylor shows the best result vs any other metric for pruning

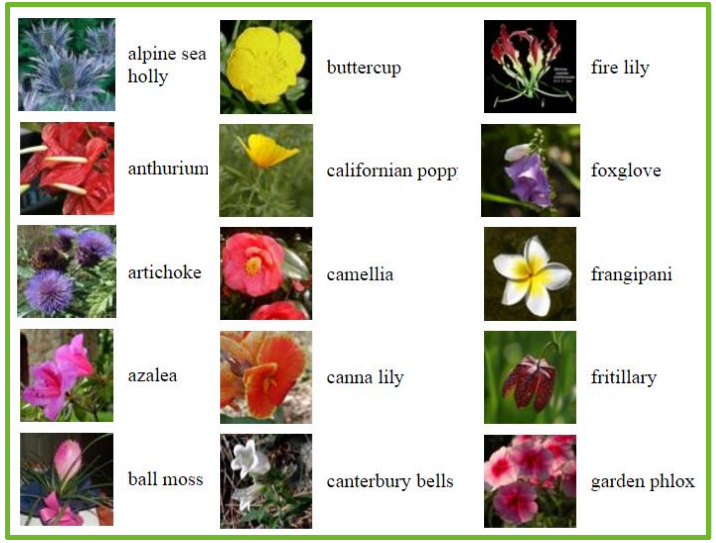
RESULTS

AlexNet on Oxford Flowers102

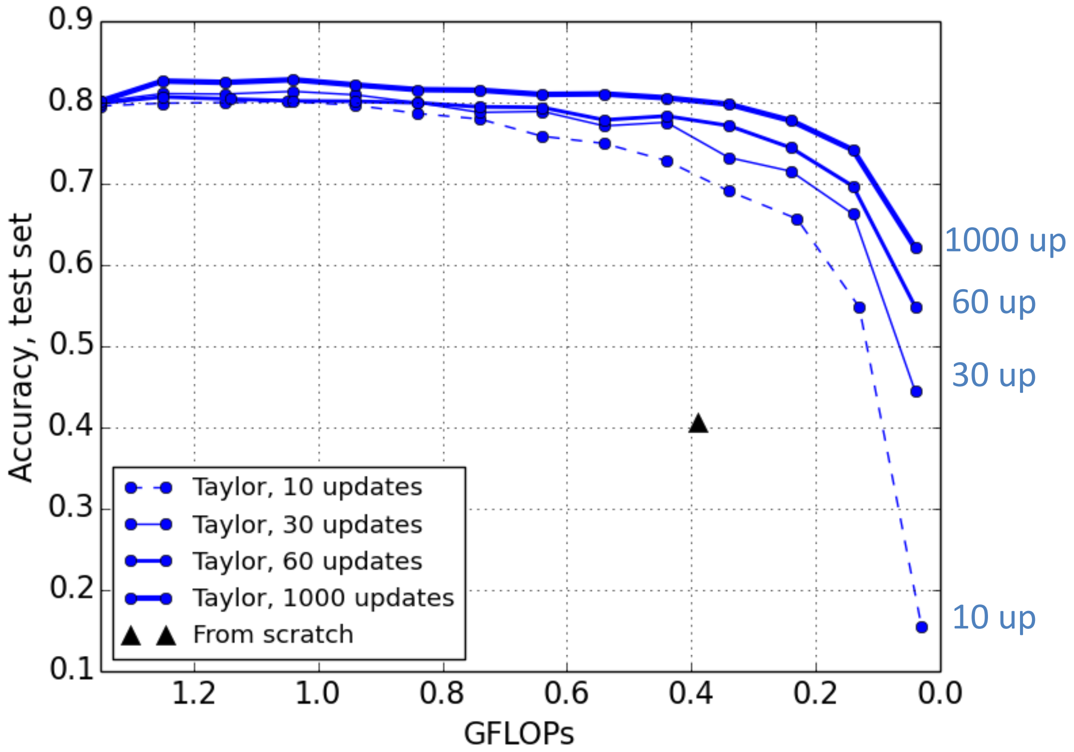
102 classes

~2k training images

~6k testing images



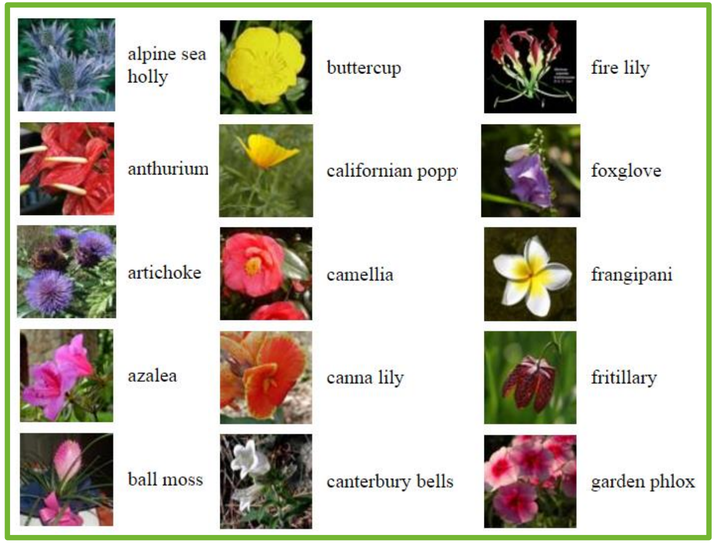
Changing number of updates between pruning iterations



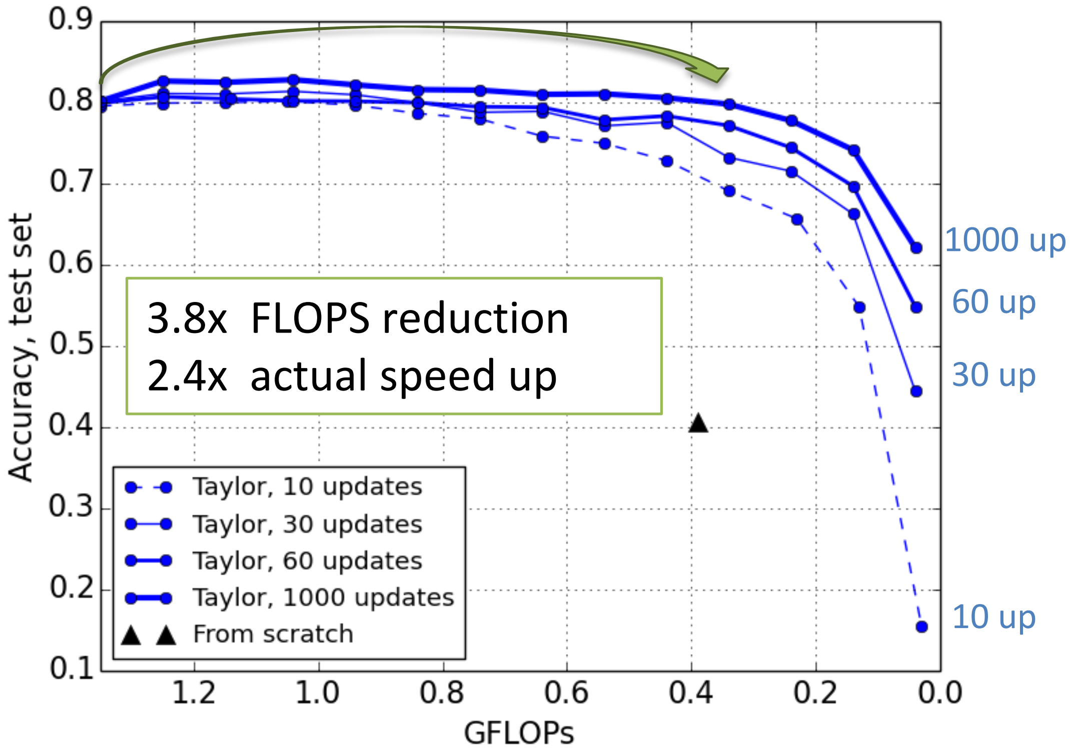
RESULTS

AlexNet on Oxford Flowers102

102 classes
~2k training images
~6k testing images



Changing number of updates between pruning iterations

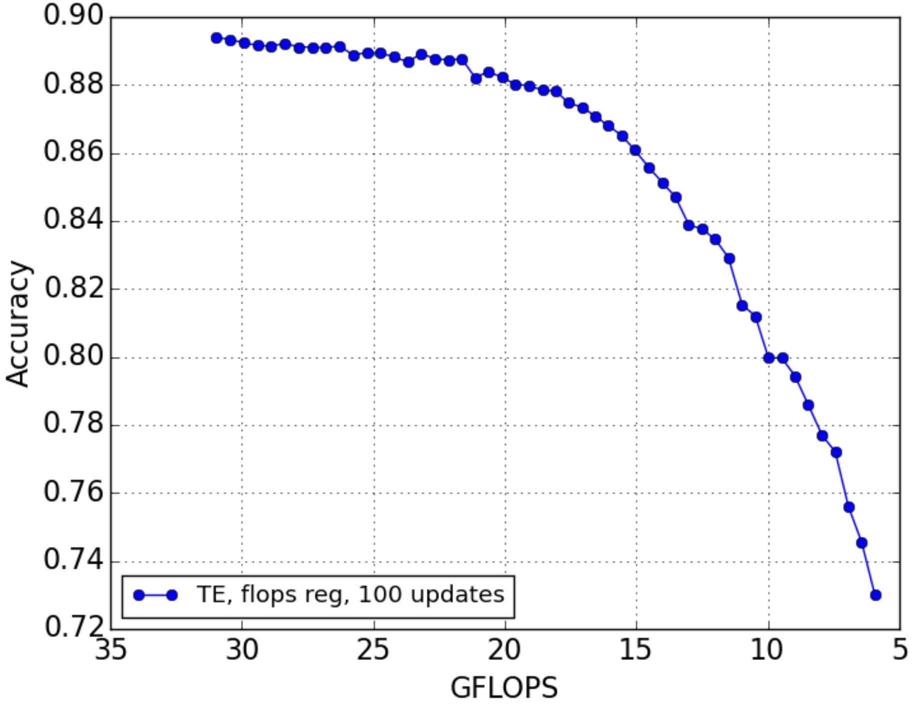


RESULTS

VGG16 on ImageNet

- Pruned over 7 epochs

Top-5 validation set



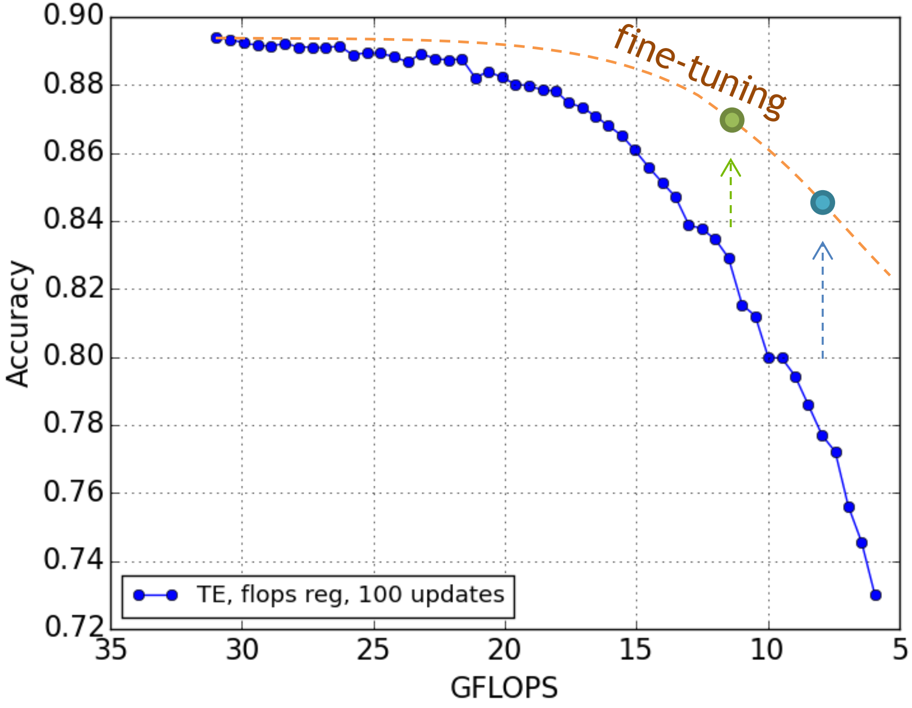
RESULTS

VGG16 on ImageNet

- Pruned over 7 epochs
- Fine-tuning 7 epochs

GFLOPs	FLOPS reduction	Actual speed up	Top-5
31	**1x**		**89.5%**
12	2.6x	2.5x	-2.5%
8	3.9x	3.3x	-5.0%

Top-5 validation set



IMAGENET

RESULTS

R3DCNN for gesture recognition

3D-CNN with recurrent layers fine-tuned for 25 dynamic gestures

P. Molchanov, *Gesture recognition with 3D CNNs*, GTC 2016

RESULTS

R3DCNN for gesture recognition

3D-CNN with recurrent layers fine-tuned for 25 dynamic gestures

12.6x

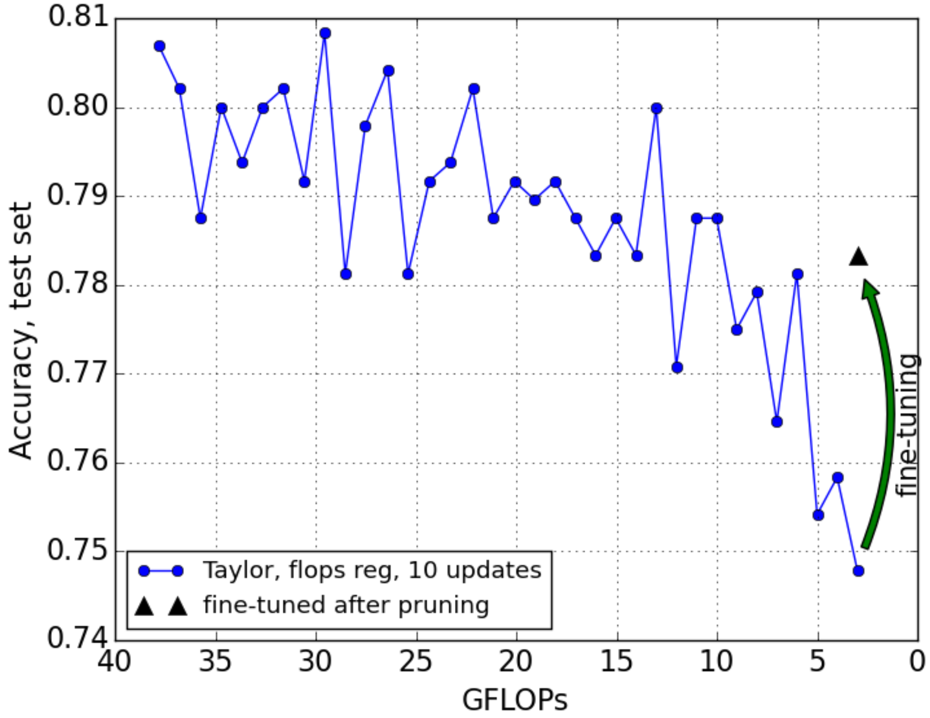
Reduction in FLOPs

Drop in accuracy

2.5%

5.2x

Speed-up



P. Molchanov, *Gesture recognition with 3D CNNs*, GTC 2016

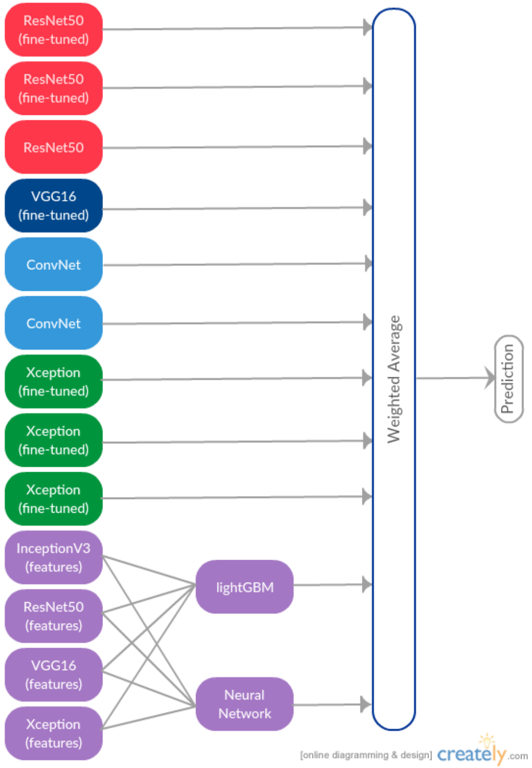
How many neurons we need to classify a cat?

DOGS VS. CATS

Dogs vs. Cats classification

25,000 images

Marco Lugo's solution, 3rd place :



DOGS VS. CATS

Fine-tuned ResNet-101

Full network

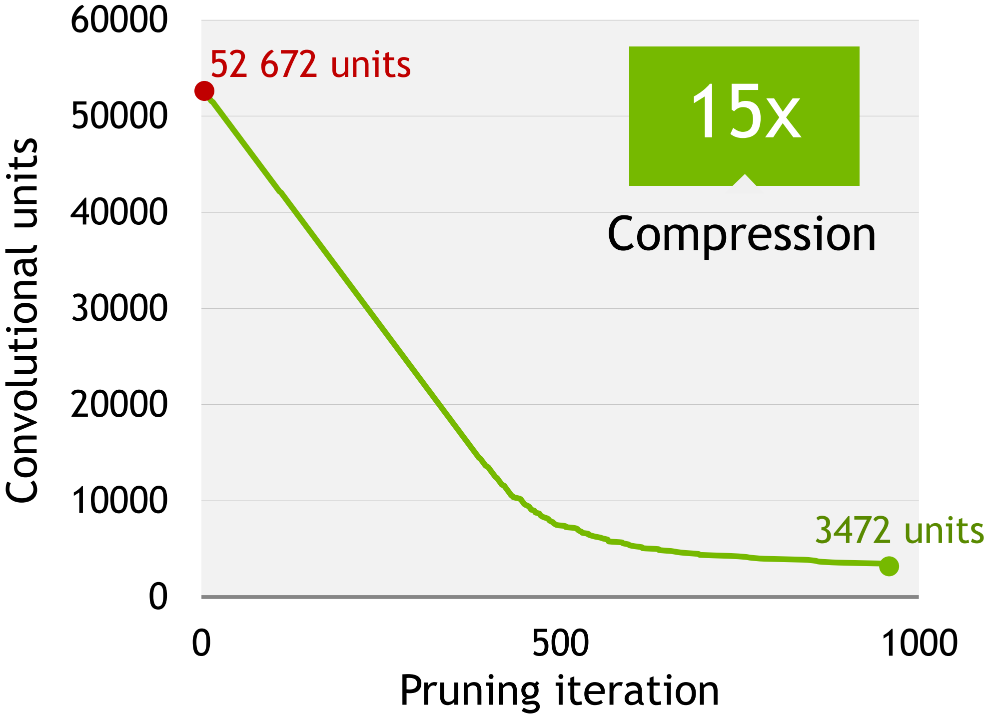
99.2%

Pruned network

99.0 %

DOGS VS. CATS

Fine-tuned ResNet-101



Full network

99.2%

Pruned network

99.0 %

CONCLUSIONS

- Pruning as greedy feature selection
- New criteria based on Taylor expansion
- Pruning is especially effective (and necessary!) for transfer learning
- Pruning can incorporate desired objectives (such as FLOPs)
- Read more in our ICLR2017 paper: https://openreview.net/pdf?id=SJGCiw5gl

THANK YOU!