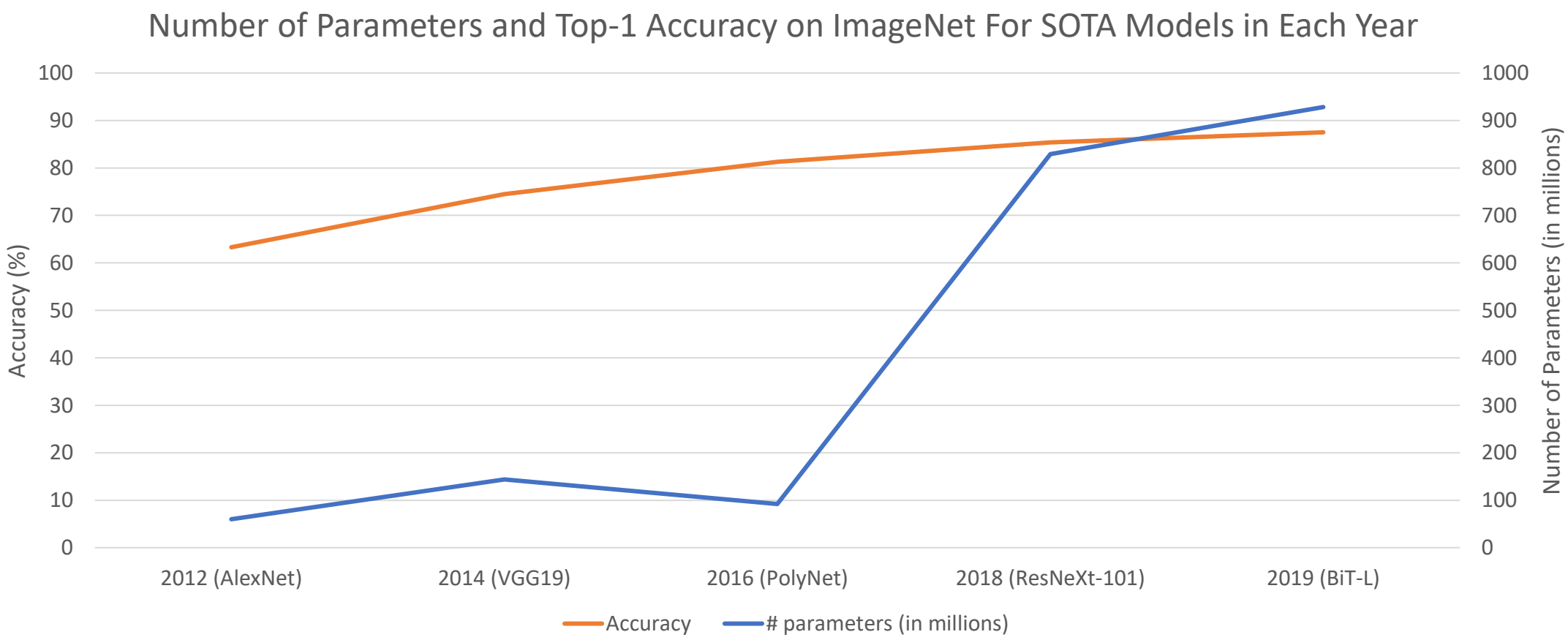
Exploiting Non-Linear Redundancy for Neural Model Compression

Muhammad Ahmed Shah, Raphael Olivier, Bhiksha Raj

Outline

- Motivation and Background
- Method: Lossless Redundancy Elimination
- Evaluation and Results
- Analysis
- Conclusion

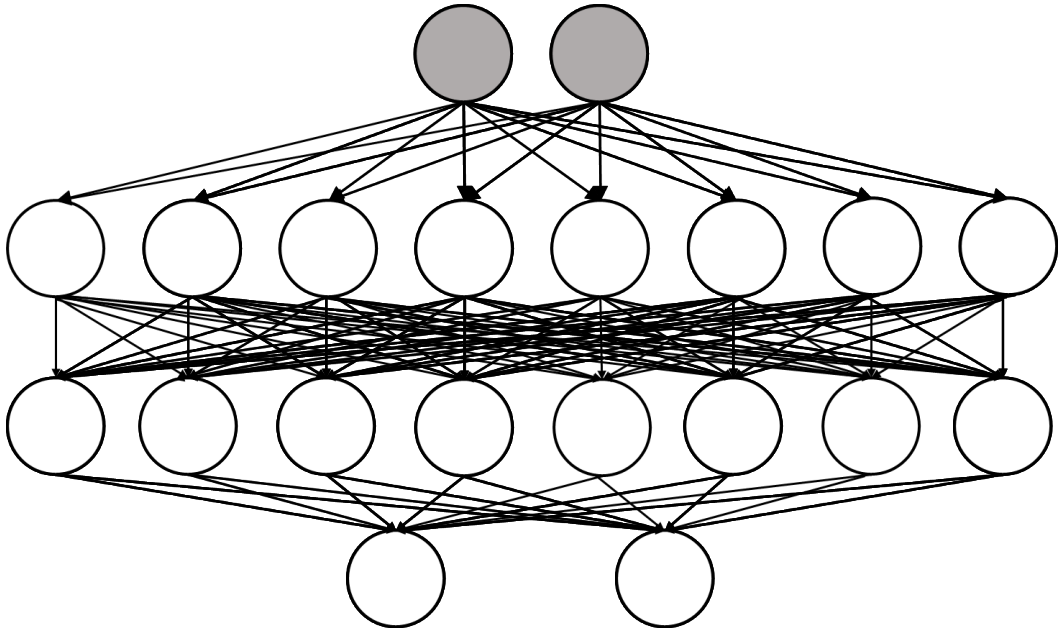
Overparameterized Models – The New Norm



Data from: https://paperswithcode.com/sota/image-classification-on-imagenet

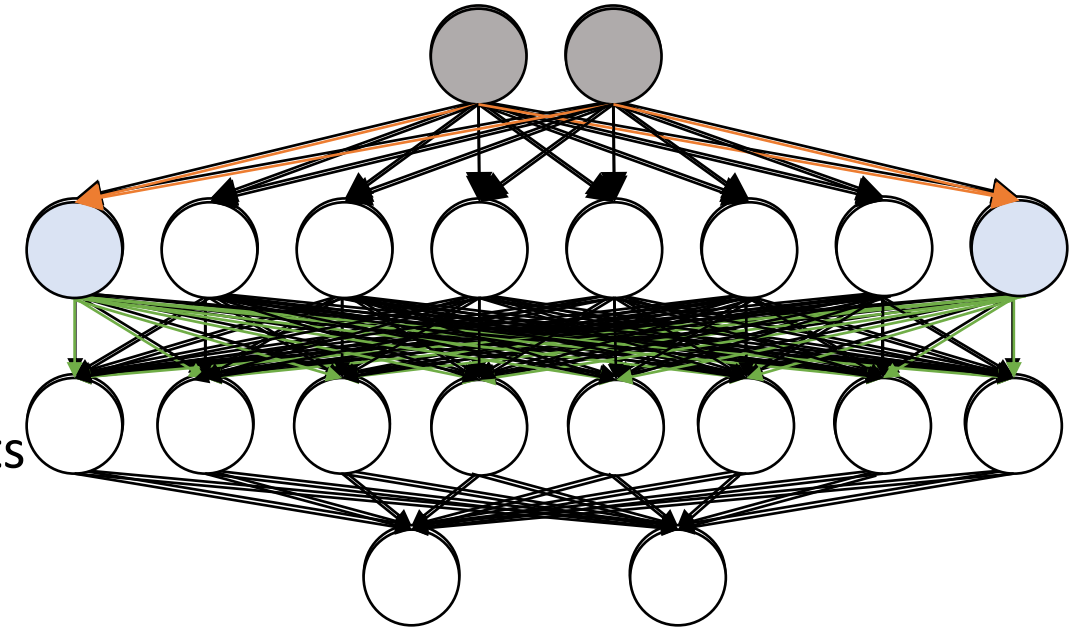
Current Methods For Model Compression

- Structural Pruning
  - Remove structural units from the model
  - Does not rely on sparse matrix operations – operationally more efficient
  - Produces compact representations
  - Selection Criteria:
    - Heuristics – magnitude of the unit's output
    - Regularization - assign weights to filter, prune filters with low weights
  - Heuristics can be misleading



Current Methods For Model Compression

- Structural Pruning
  - For example – two units have identical outputs and identical incoming and outgoing weights
  - Suppose they have sufficiently high activations (relative to the other units)
  - Both neurons will have an equal and significant impact on downstream outputs
  - Thus most magnitude-based heuristics would retain both neurons
  - Ideally, achieve lossless compression by removing one of the neurons and doubling the other neuron's outgoing weights



The significance of a neuron is how (un)predictable its output is given the outputs of the other neurons in the layer!

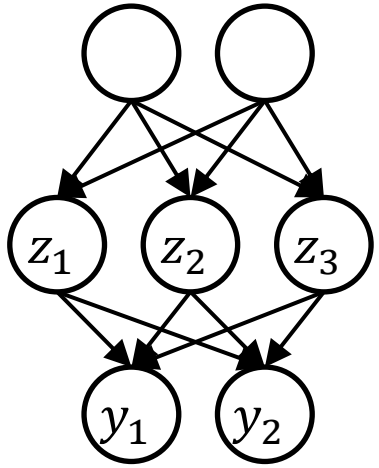
Outline

- Motivation and Background
- Method: Lossless Redundancy Elimination
- Evaluation and Results
- Analysis
- Conclusion

Lossless Redundancy Elimination

- Consider this network

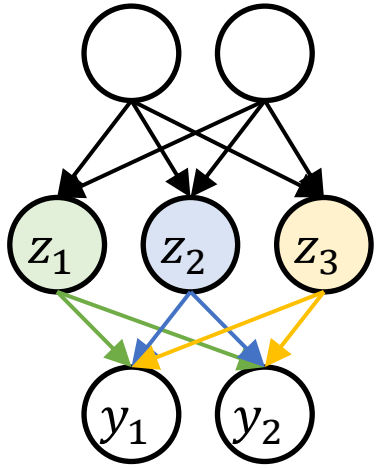
$$y_1 = w_{11}z_1 + w_{12}z_2 + w_{13}z_3$$
$$y_2 = w_{21}z_1 + w_{22}z_2 + w_{23}z_3$$



Lossless Redundancy Elimination

- Suppose $z_1 = \alpha z_2 + \beta z_3$
- In this case

$$y_1 = (w_{12} + \alpha w_{11})z_2 + (w_{13} + \beta w_{11})z_3$$
$$y_2 = (w_{22} + \alpha w_{21})z_2 + (w_{23} + \beta w_{21})z_3$$

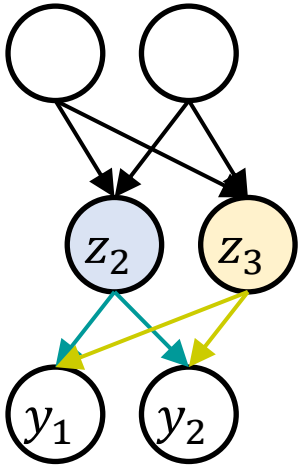


Lossless Redundancy Elimination

- Suppose $z_1 = \alpha z_2 + \beta z_3$
- In this case

$$y_1 = (w_{12} + \alpha w_{11})z_2 + (w_{13} + \beta w_{11})z_3$$
$$y_2 = (w_{22} + \alpha w_{21})z_2 + (w_{23} + \beta w_{21})z_3$$

- We can remove z_1
- And readjust weights
  - $w_{12} \leftarrow w_{12} + \alpha w_{11}$ $\quad$ $w_{13} \leftarrow w_{13} + \beta w_{11}$
  - $w_{22} \leftarrow w_{22} + \alpha w_{21}$ $\quad$ $w_{23} \leftarrow w_{23} + \beta w_{21}$



LRE-AMC

- We modify an exiting technique called Annealed Model Contraction (AMC)
- We do the following to compress a single layer:
  1. Compute the predictability of the units (neurons/conv filters) using OLS regression
  2. Remove $\gamma\%$ of the most predictable units.
  3. Fine tune the network
  4. Measure the accuracy of the model and repeat if accuracy is recovered

RemoveAndAdjust(A, W, j): adjust the weight matrix after the removal of the j^{th} neuron
Function `LREShrink(`F, l, γ`):`
  $Z \leftarrow F_{1:l}(\mathcal{X})$ `// compute the activations of the` l^{th} `layer.`
  $A \leftarrow \min_A \|ZA - Z\|^2$ s.t $\text{diag}(A) = \mathbf{0}$
  $\mathcal{E} \leftarrow \arg\text{sort}(\|ZA - Z\|^2)[: \lfloor \gamma * \text{sizeof}(F[l]) \rfloor]$
  $\bar{W}^{(l+1)} \leftarrow [W^{(l+1)}; b^{(l+1)}]$ `// Concatenate the weights and bias.`
  for $j \in \mathcal{E}$ **do**
    $W^{(l)} \leftarrow W^{(l)}_{-j.}$ `//drop the` j^{th} `row of` $W^{(l)}$
    $W^{(l+1)} \leftarrow \text{RemoveAndAdjust}(A, \bar{W}^{(l+1)}, j)$
  end
$Acc \leftarrow \text{evaluate}(F_t)$
$F'_s[i_B] \leftarrow \text{LREShrink}(F_s, i_B, \gamma)$
$Acc' \leftarrow \text{evaluate}(F'_s)$
while $Acc - Acc' \leqslant \epsilon$ **do**
  $F_s \leftarrow F'_s$
  $F'_s[i_B] \leftarrow \text{LREShrink}(F'_s, i_B, \gamma)$
  $Acc' \leftarrow \text{evaluate}(F'_s)$
  if $Acc - Acc' > \epsilon$ **then**
    $F'_s \leftarrow \text{distill}(F'_s)$
    $Acc' \leftarrow \text{evaluate}(F'_s)$
  end
end

Outline

- Motivation and Background
- Method: Lossless Redundancy Elimination
- Evaluation and Results
- Analysis
- Conclusion

Experimental Setup

- In each compression iteration we remove 25% of the neurons in the layer.
- We keep compressing as long as the validation accuracy does not deteriorate by more than ϵ %.

Results

- LRE-AMC can drastically shrink the model
- Effective on large/complex datasets as well like Image Net
- TD shrinking is more effective for param reduction
- RR shrinking is more effective for FLOP reduction
- Comparison with prior work:
  - Closest competitor [1] removes 4% fewer params but 5% more FLOPs
  - LRE-AMC preferable if memory is constrained.

Dataset	Param Reduction (%)	FLOP Reduction (%)	Accuracy Reduction (%)	ϵ
CIFAR10	97.4	80.2	1.8	0
CIFAR10-[1]	94.3	**85.0**	**0.5**	-
CIFAR10-[2]	93.6	65.0	0.6	-
CIFAR10-[3]	64.0	64.0	2.1	-
Caltech256	81.7	65.2	1.7	1
ImageNet	11.8	20.0	**1.6**	2
ImageNet	**19.1**	**26.0**	3.0	3

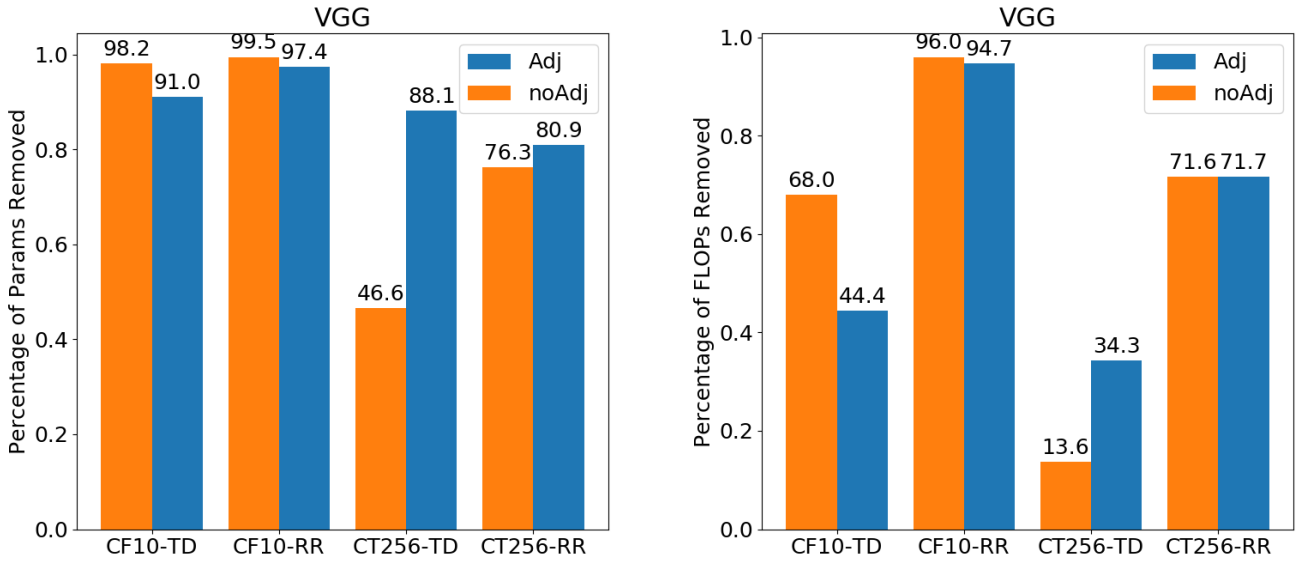
CF10 = CIFAR 10 **CT256** = Caltech 256 **IN**=ImageNet

1. L. Liebenwein +, “Provable filter pruning for efficient neural networks,” 2019.
2. Z. Zhuang+, “Discrimination-aware channel pruning for deep neural networks,” in NeurIPS 2018,
3. J.-H. Luo+, “Thinet: A filter level pruning method for deep neural network compression,” ICCV, 2017,

Outline

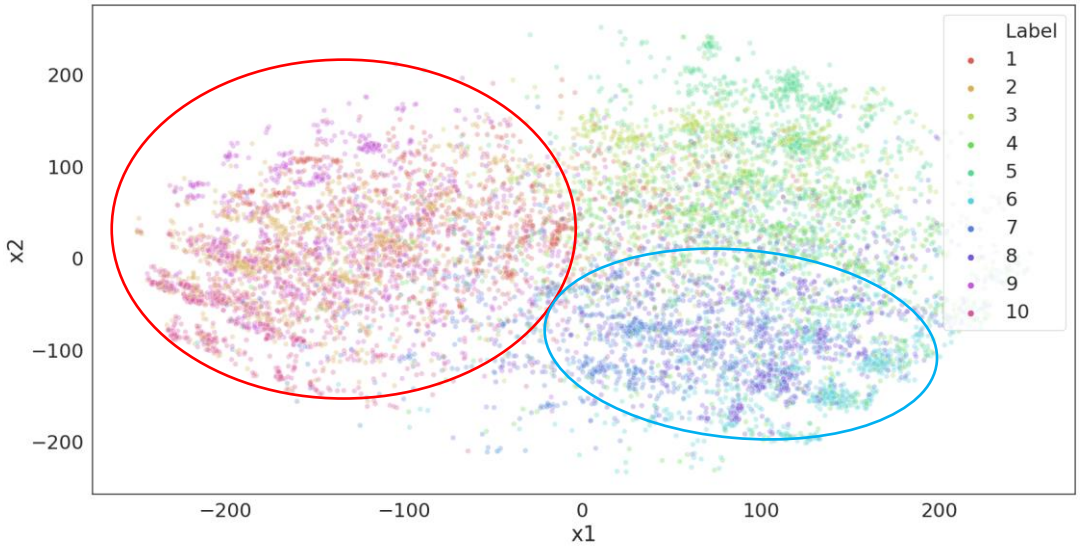
- Motivation and Background
- Method: Lossless Redundancy Elimination
- Evaluation and Results
- Analysis
- Conclusion

Effect of Weight Re-adjustment

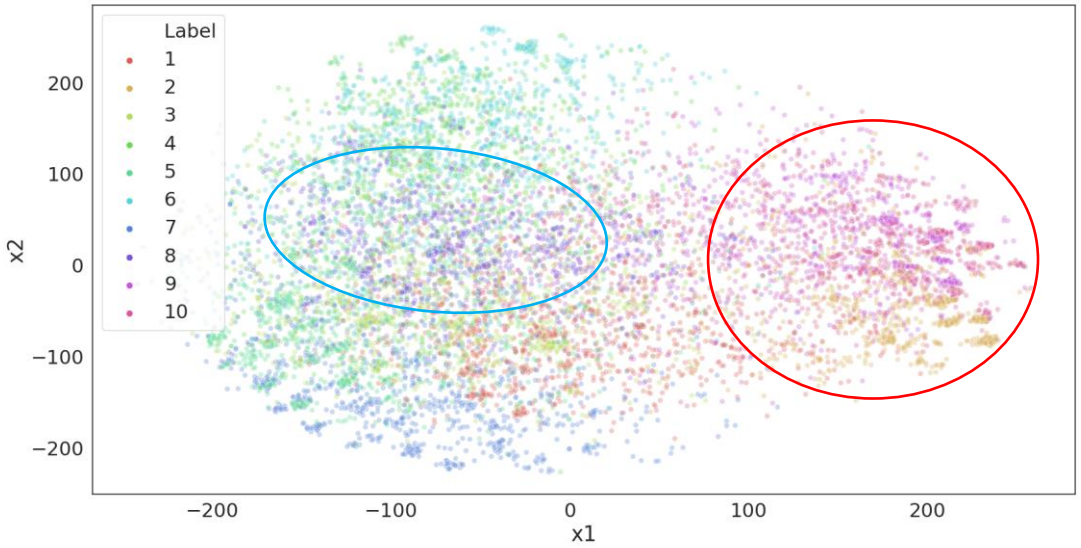


- $\epsilon = 5\%$
- Improves param and FLOP reduction for Caltech-256
  - More complex data, fewer redundant neurons
- More beneficial under Top Down
  - Allows us to compress under low redundancy

Effect of Weight Re-adjustment



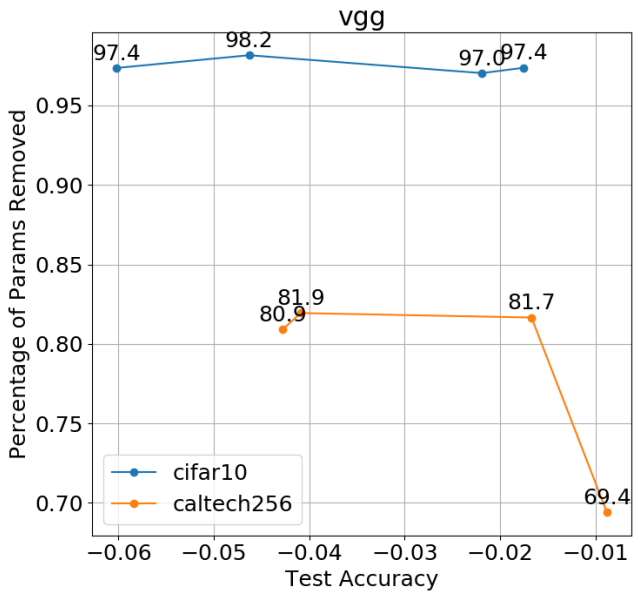
Without weight adjustment

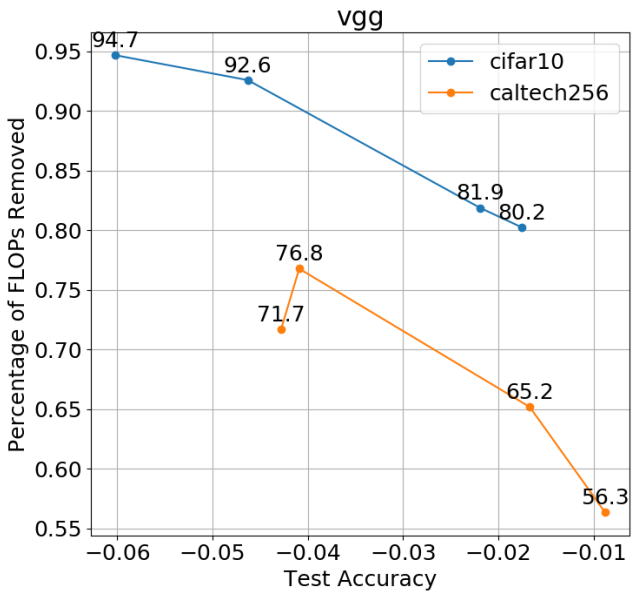


With weight adjustment

- Output of the final convolutional layer in VGG-16 on CIFAR10
- Weight readjustment separates classes more cleanly
  - Classes 1, 2, 7 and 8

Accuracy vs. Compression Trade Off





- Almost linear relationship between reduction in FLOPs and Accuracy
- Accuracy is more resistant to reduction in parameters
  - Accuracy does not depend on how many parameters are removed, rather which parameters are removed.

Outline

- Motivation and Background
- Method: Lossless Redundancy Elimination
- Evaluation and Results
- Analysis
- Conclusion

Conclusion

- We have presented LRE-AMC, a technique to identify and eliminate non-linear dependencies between neurons.
- LRE-AMC can remove more than 97% of the model parameters and 80% of the FLOPs from a VGG-16 trained on CIFAR-10
- Our analysis indicates that our weight adjustment technique, LRE, yields better compression and maintains the intermediate representations.

Thank You

Questions?