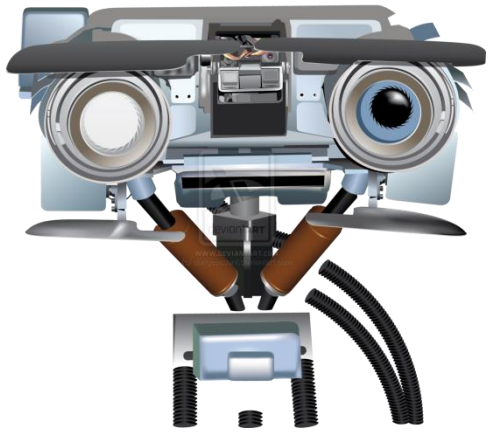
Announcements

Reminder: Class challenge out! Ends December 10th

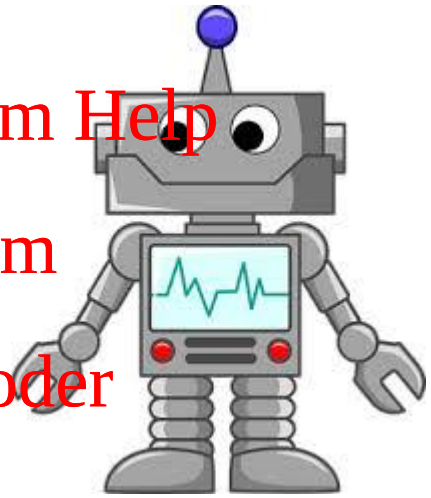
- Lab this week: final review, class survey



On-device Machine Learning

CS542 Fall 2020

Creating an AI Model



model development

model deployment

What if the model you trained is too big/takes too long to run?

Reminder: memory representation

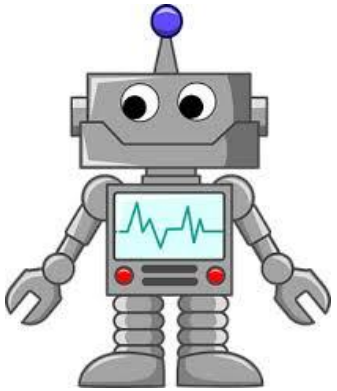
- Int – 4 bytes
- Float – 4 bytes
- Double – 8 bytes
- Binary variable – 1 byte

Aha! We can reduce memory by 1/4th if we can binarize the weights/features!

Binary features enable us to use hamming distance for fast similarity computation!

Why should I care?

resource scarce systems



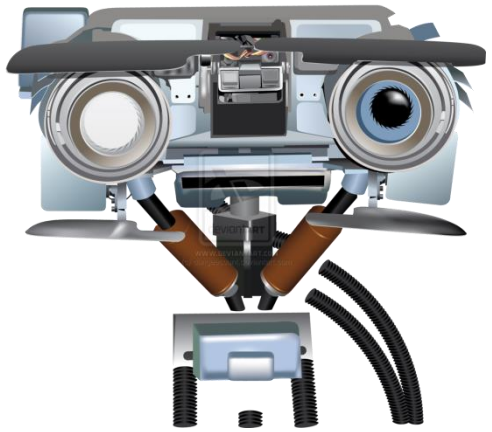
efficient features for search

Goals

- Efficiency
  - Inference speed
  - Memory reduction (either RAM or disk space)
- Limit Performance loss
- ~~Training time?~~

Today

- Network/Feature Quantization
- Parameter Pruning
- Knowledge Distillation

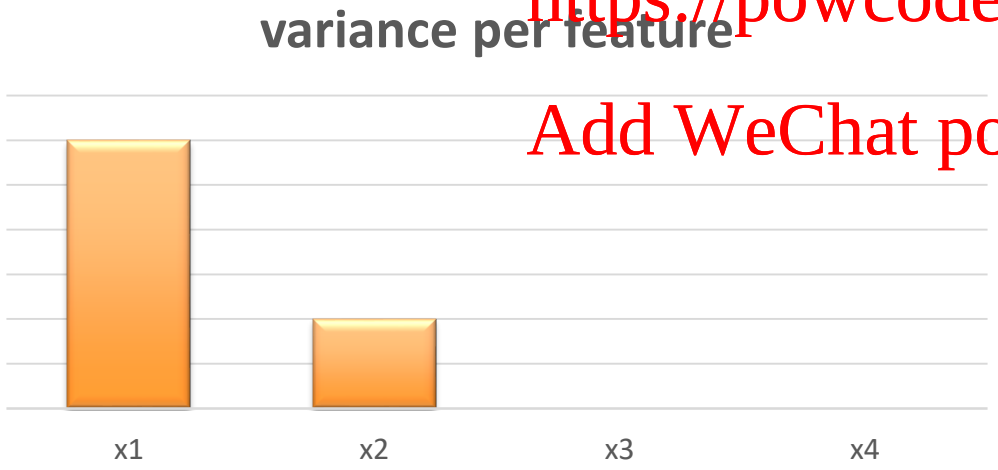


Feature Quantization

CS542 Fall 2020

Simple feature quantization method - threshold

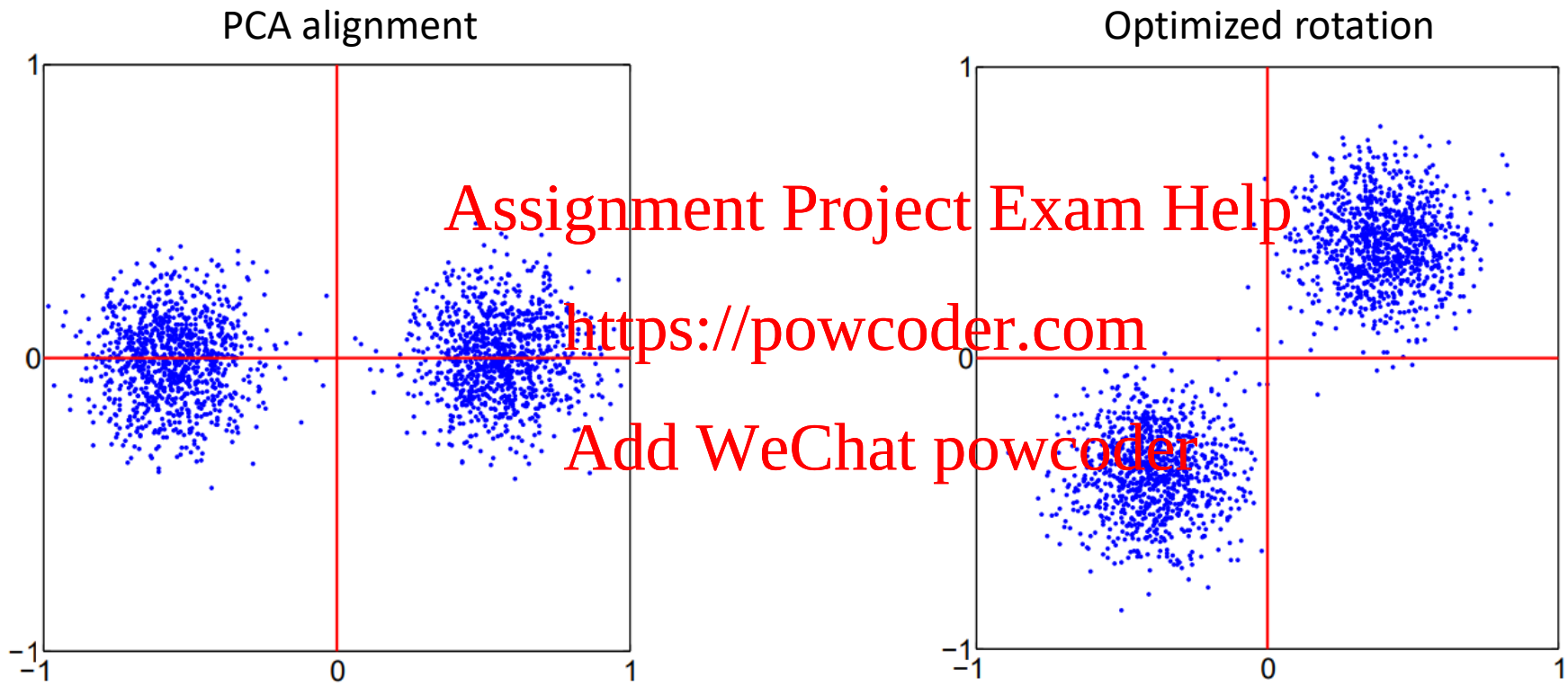
- Find the mean μ of each feature x_i in the training set, then binarize using $\mu > x_i$



Some features have more information than others

Also assumes the initial feature representation size is small enough

Iterative quantization



Paper reference: Iterative Quantization: A Procrustean Approach to Learning Binary Codes. Y. Gong and S. Lazebnik, CVPR 2011.

Quantization optimization

For orthogonal rotation matrix R and projected features V, minimize:

$$Q(B, R) = \|B - VR\|_F^2$$

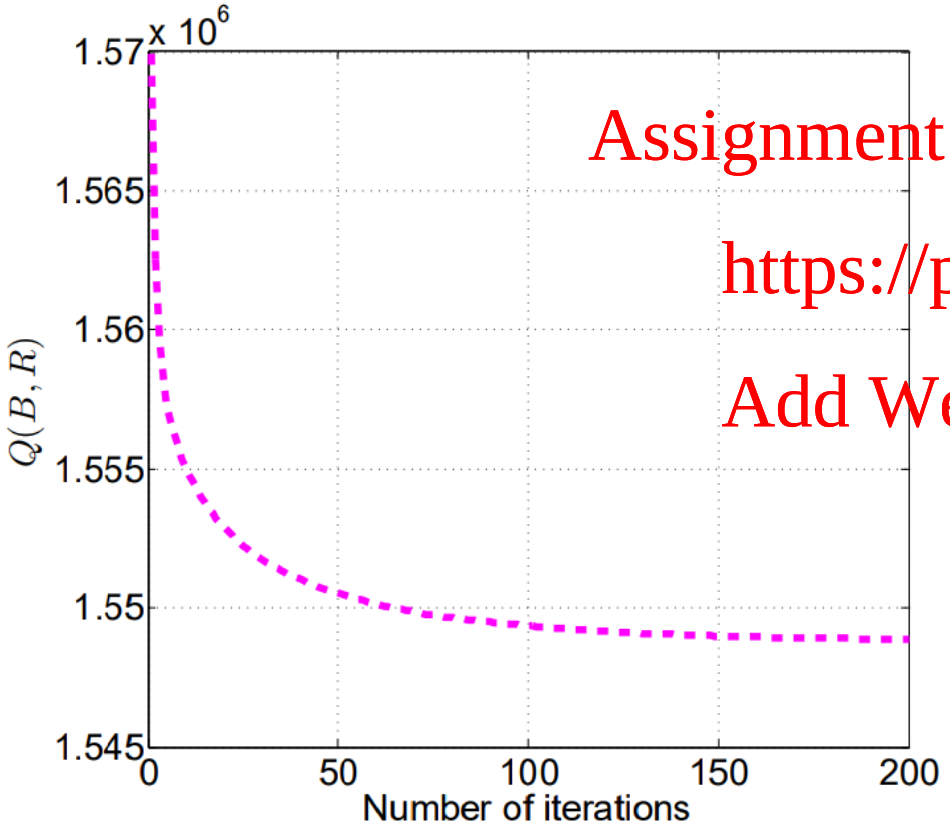
where B is a binary code matrix.

But it is dependent on two variables!

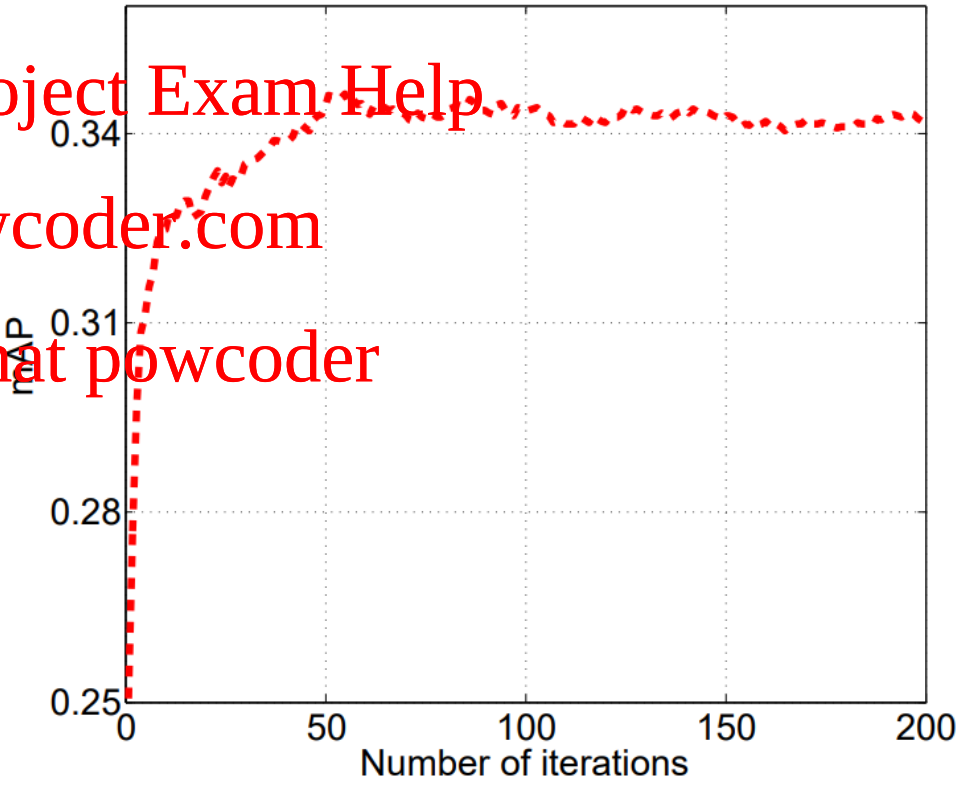
Fix one, update the other

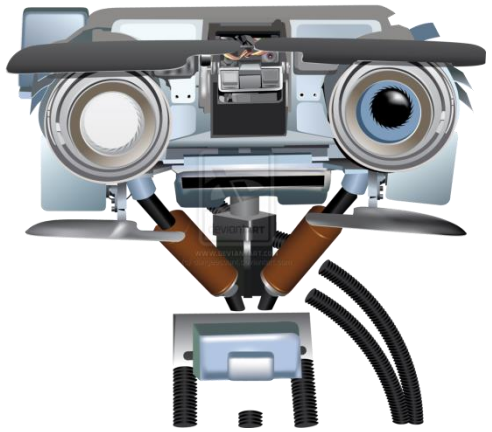
Performance over time

Cost function value



Euclidean neighbor precision





Network Quantization

CS542 Fall 2020

Convolutions in binary networks

figure credit: https://arxiv.org/pdf/2004.03333.pdf

Runtime comparison

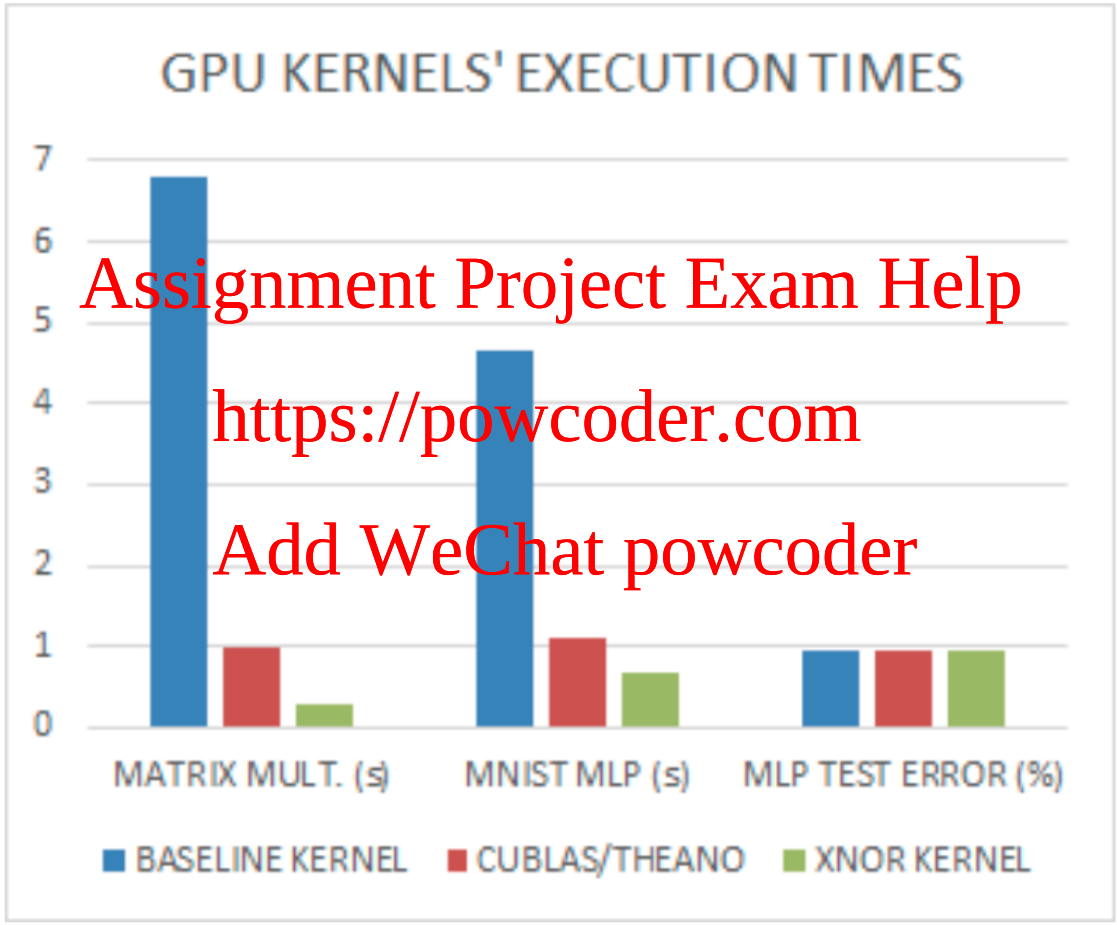
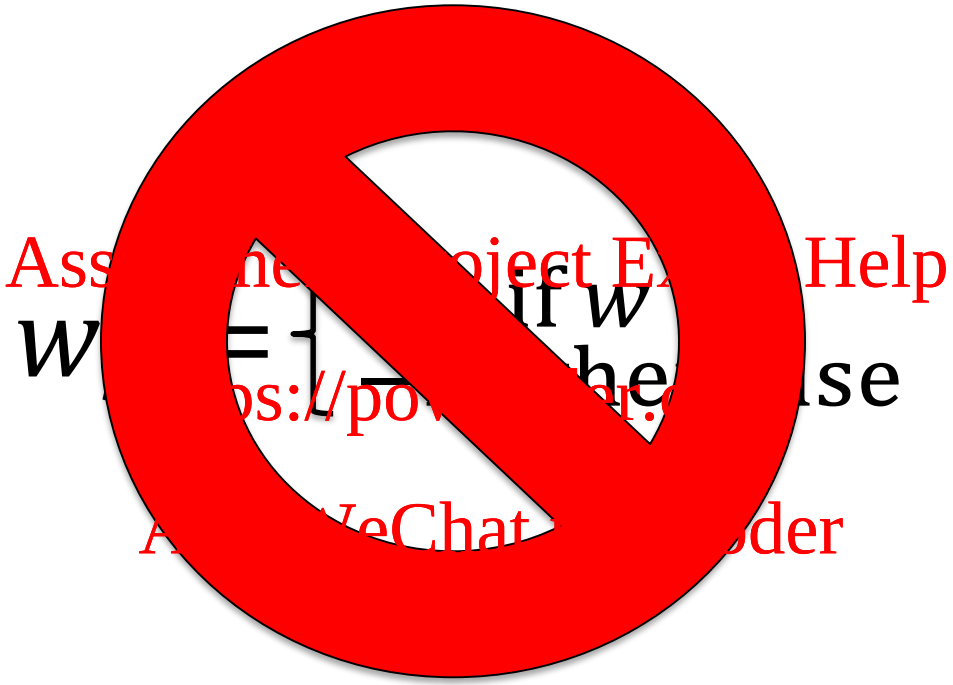


figure credit: Hubara et al., Binarized Neural Networks, NeurIPS, 2016

A simple approach



Results in significant loss of information!

Example of binarization

Simple method:

$$w = [0.1, 0.1, 0.1, 0.1]$$
$$= [1, 1, 1, 1]$$

A better approach

$$w_b = \begin{cases} +1 \text{ with probability } p = \sigma(w) \\ -1 \text{ with probability } 1 - p \end{cases}$$

$$\sigma(x) = \max\left(0, \min\left(1, \frac{x+1}{2}\right)\right)$$

Example of binarization

Reminder: $\sigma(x) = \max\left(0, \min\left(1, \frac{x+1}{2}\right)\right)$

Simple method:

$$w = [0.1, 0.1, 0.1, 0.1]$$

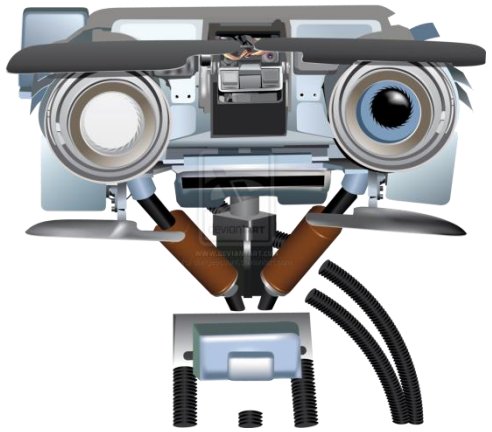
$$= [\ 1,\ \ 1,\ \ 1,\ \ 1]$$

Better approach:

$$w = [0.1, 0.1, 0.1, 0.1]$$

$$= [\ 1, -1, -1,\ \ 1]$$

Additional details: Courbariaux et al. BinaryConnect: Training Deep Neural Networks with binary weights during propagations. NeurIPS, 2015.



Parameter Pruning

CS542 Fall 2020

Remove unimportant weights

- Many weights may not affect performance much (if at all)
- Can save space and computation since you can skip operations*

Optimal Brain Damage (LeCun et al., NeurIPS, 1989)

- Find parameters that don't affect the training/validation error of the network.

1. Train your network
2. Compute the second-order derivatives h_{kk} for each parameter
3. Compute parameter saliencies $s_k = \frac{h_{kk}u_k}{2}$ ($u_k =$ next layer parameters)
4. Delete some low saliency parameters
5. Repeat

Does this really save space/time?

It depends….

1	1	0
0	1	0
1	1	1

For CNNs we can just prune entire filters instead!

Pruning entire filters in CNNs

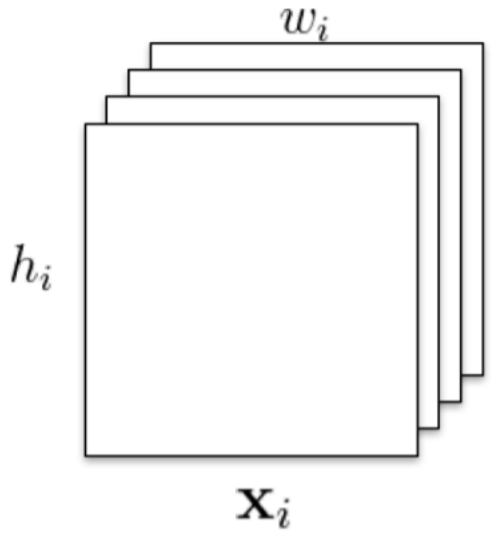


figure credit: Li et al., Pruning Filters for Efficient Convnets, ICLR, 2017

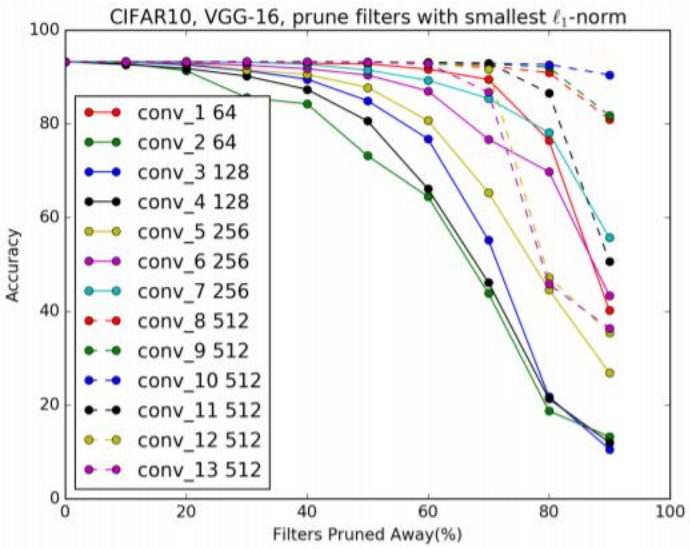
Pruning process

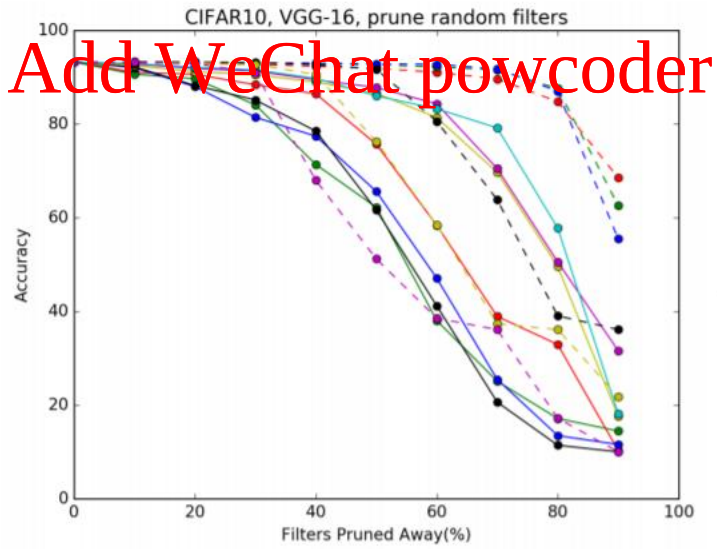
1. Train your neural network
2. Prune filters for layer l_k
3. Retrain your network
4. Repeat for layer l_{k+1}

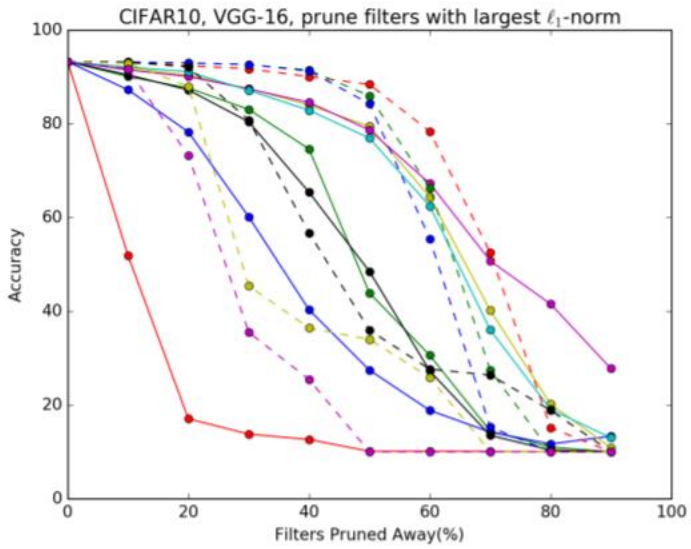
Why not all at once?

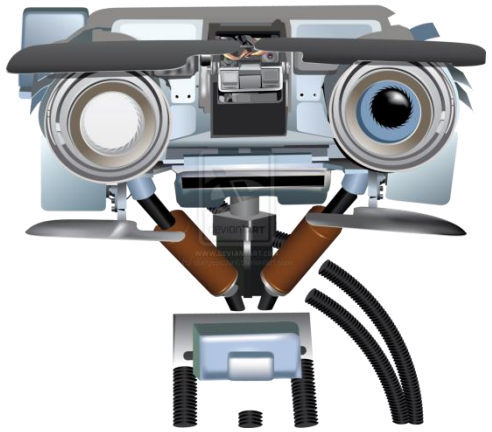
How to choose which filters to prune?

- Often the focus of many research papers.
- The approach of Li et al., Pruning Filters for Efficient Convnets, ICLR, 2017:
  - Rank filters $w_1, w_2, \ldots$ using $s_k = \|w_k\|_1$







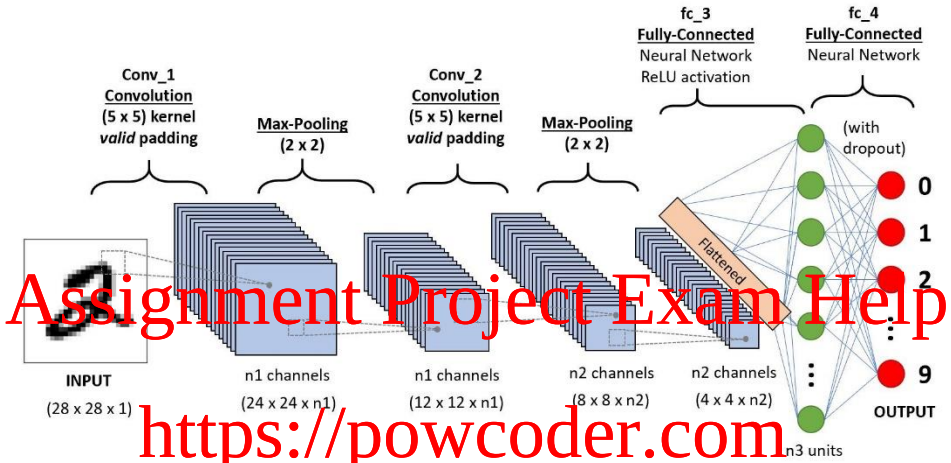


Knowledge Distillation

CS542 Fall 2020

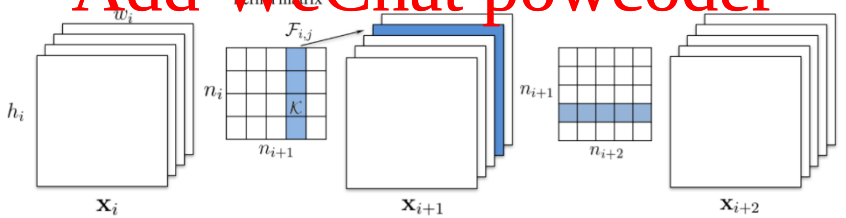
Creating an efficient network

Trained CNN



Why not train a more efficient network to begin with?

Pruning



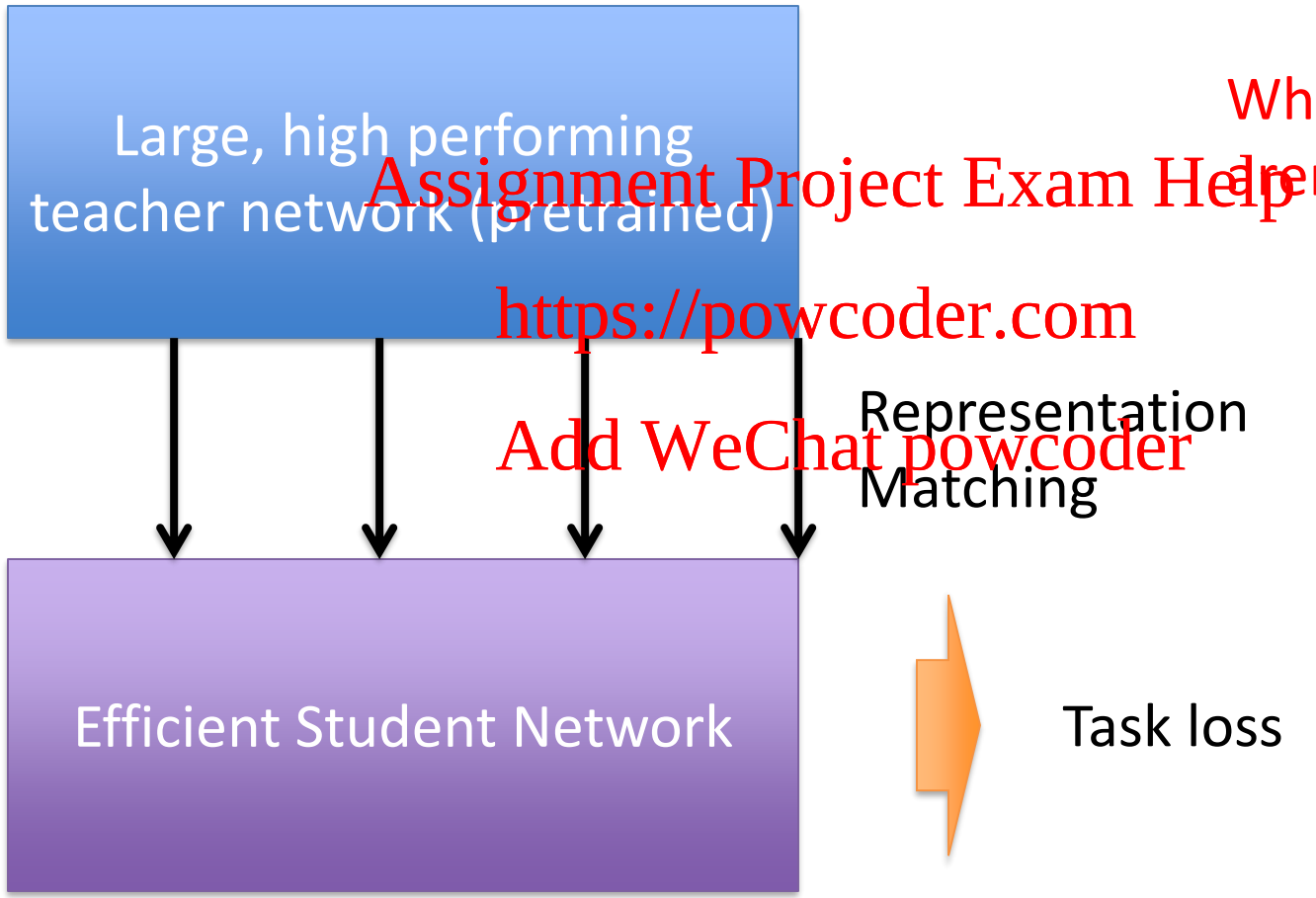
Efficient CNN!

CNN image credit: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Distillation idea

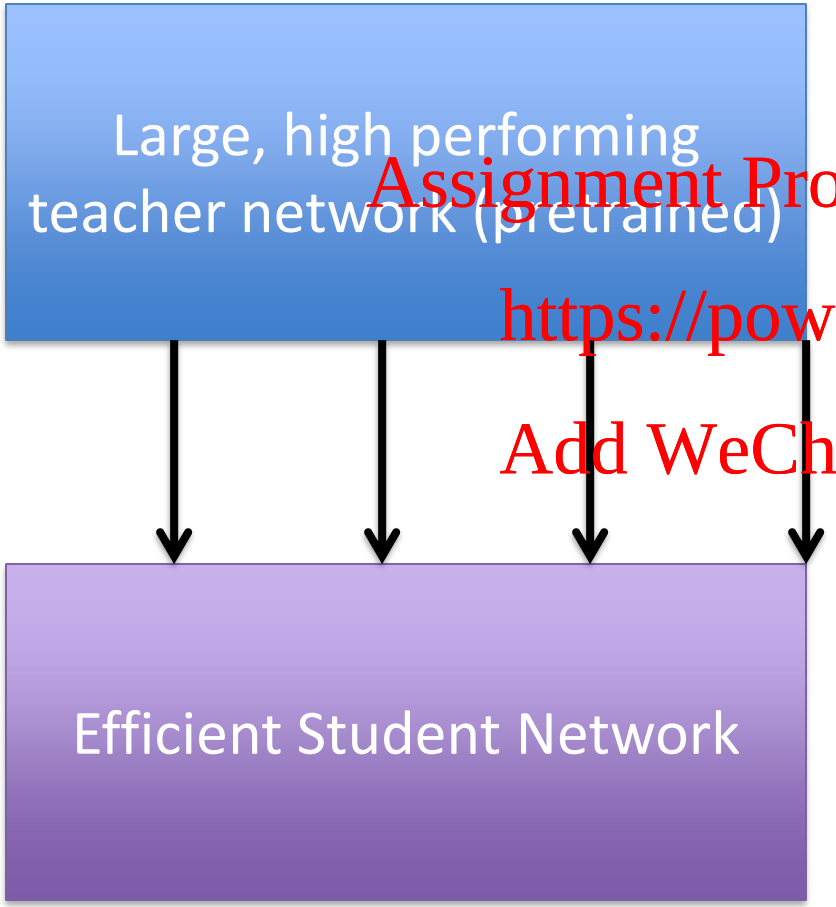
- Smaller, more efficient networks may not perform well
- We have an example of a network that achieves better performance
- Try to get the small network to mimic the large network

Learning with a student-teacher framework



What if the activations aren't the same size?

Learning with a student-teacher framework



What if the activations aren't the same size?

Representation Matching

Task loss + Representation Matching loss

Distillation loss (i.e. matching representations)

$$L_{distill}(a_t, a_s) = \|a_t - a_s\|_2^2$$

More generally considering a case where relevant information is extracted through some transformation function f, g for the teacher, student, respectively

$$L_{distill}(a_t, a_s) = \|f(a_t) - g(a_s)\|_2^2$$

Applications of distillation

- Creating an efficient student using a teacher network
- Can distill ensemble results into a single network

Method	Speech recognition accuracy
Baseline	58.9
Ensemble of 10 models	61.1
Distilled single model	60.8

reference: Hinton et al. Distilling the Knowledge in a Neural Network. NeurIPS 2014 Deep Learning Workshop
another useful paper: Heo et al. A Comprehensive Overhaul of Feature Distillation. ICCV, 2019.

Next Class

Final Review:

expect questions on material covered in the entire course in lectures, problem sets, LABs, and assigned reading

Course evaluations available (5-10 minutes to complete).
Please fill it out at: https://bu.campuslabs.com/courseeval