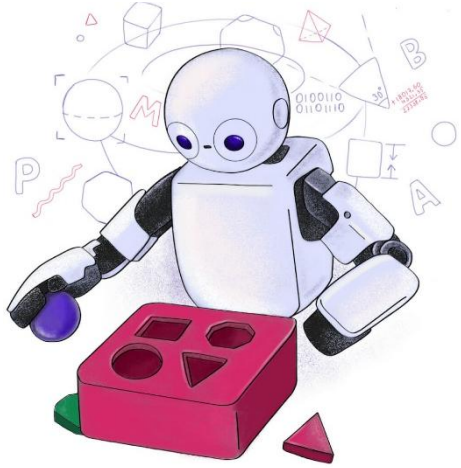
TP558 - Tópicos avançados em Machine Learning:

DEEP COMPRESSION: COMPRESSING DEEP NEURAL NETWORKS WITH PRUNING, TRAINED QUANTIZATION AND HUFFMAN CODING



Inatel

Gabriel Dias Scarpioni
gabriel.scarpioni@inatel.br

Introdução

- REDES NEURAIS PROFUNDAS: Alta performance porém alto custo
  - AlexNet Caffemodel > 200MB
  - VGG-16 Caffemodel > 500MB
- Dificuldade de implementar em sistemas mobile
  - Tamanho dos arquivos
  - Consumo de energia
- Vantagens de redes neurais embarcadas: Maior privacidade, menor largura de banda e processamento em tempo real

Introdução

- Consumo de energia estimado 45 nm CMOS:

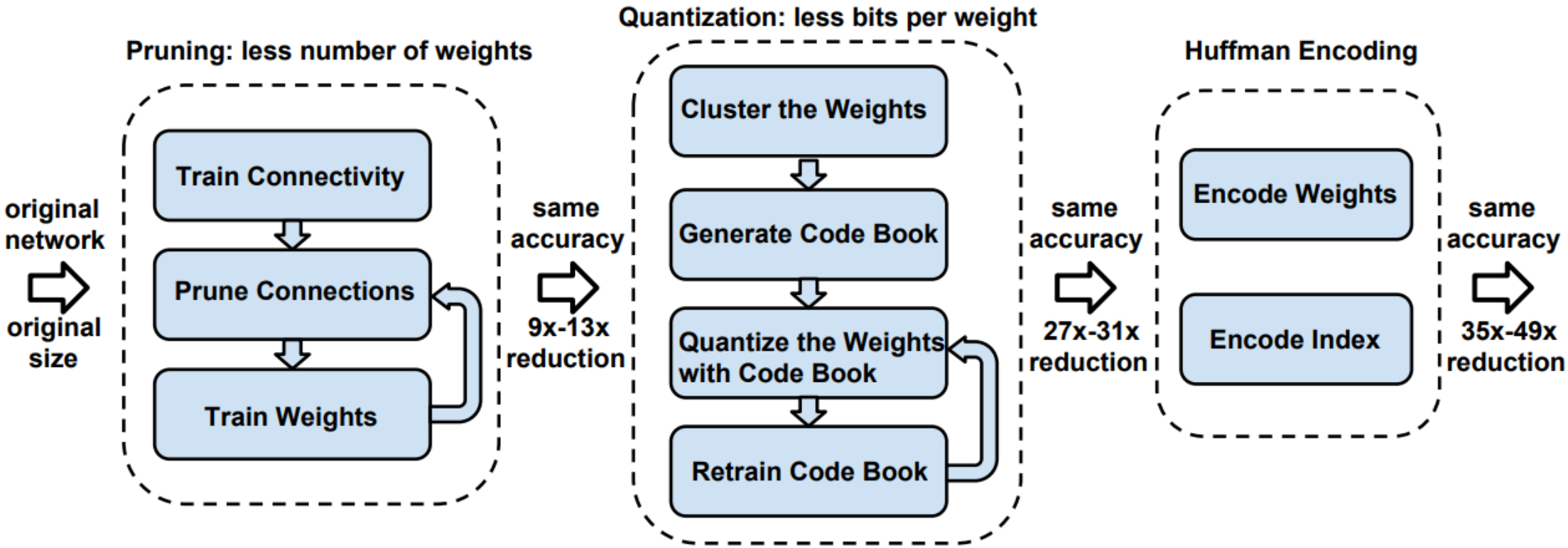
- Consumo de uma operação de adição floating de 32 bits: 0.9 pJ
- Acesso à cache 32 bit SRAM: 5 pJ
- Acesso à memória 32 bit DRAM: 640 pJ

- Executando uma rede com 1 bilhão de conexões com 20 fps, o consumo será:

$$20 \times 1 \times 10^{9} \times \ 640 \times 10^{-12} = 12{,}8\ W$$

Introdução

- Objetivo: Reduzir o armazenamento e a energia necessária para executar inferências em grandes redes neurais



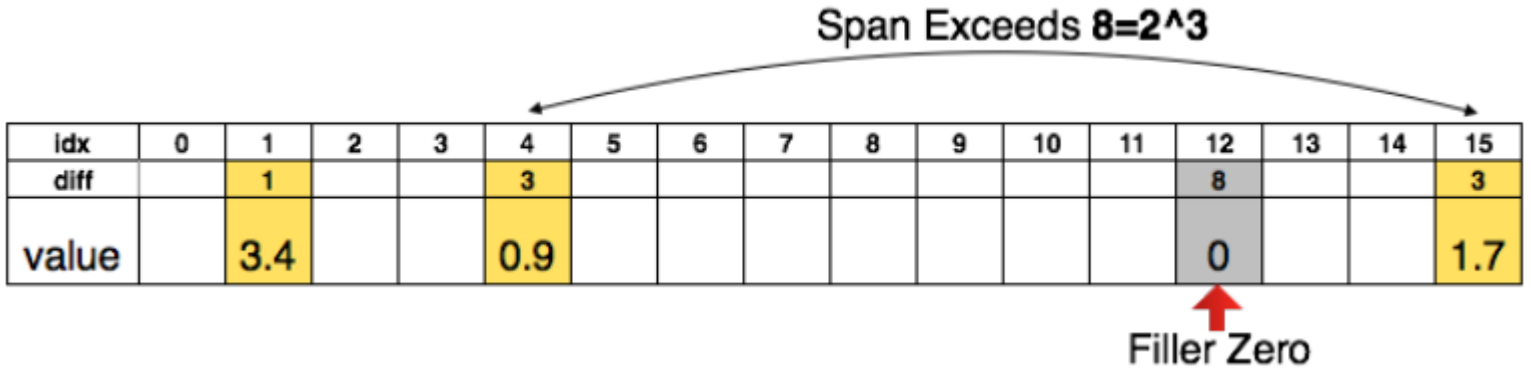
Pipeline: pruning, quantization e Huffman coding

NETWORK PRUNING

- Amplamente estudada para comprimir modelos de CNN
- Forma válida de reduzir complexidade de rede e o overfitting sem perda de acurácia
- Como funciona:
  - Passo 1: Treinamento normal da rede
  - Passo 2: Todas as conexões com peso abaixo de um limiar são removidas
  - Passo 3: Novo ciclo de treinamento para aprender os pesos finais das conexões esparsas

NETWORK PRUNING

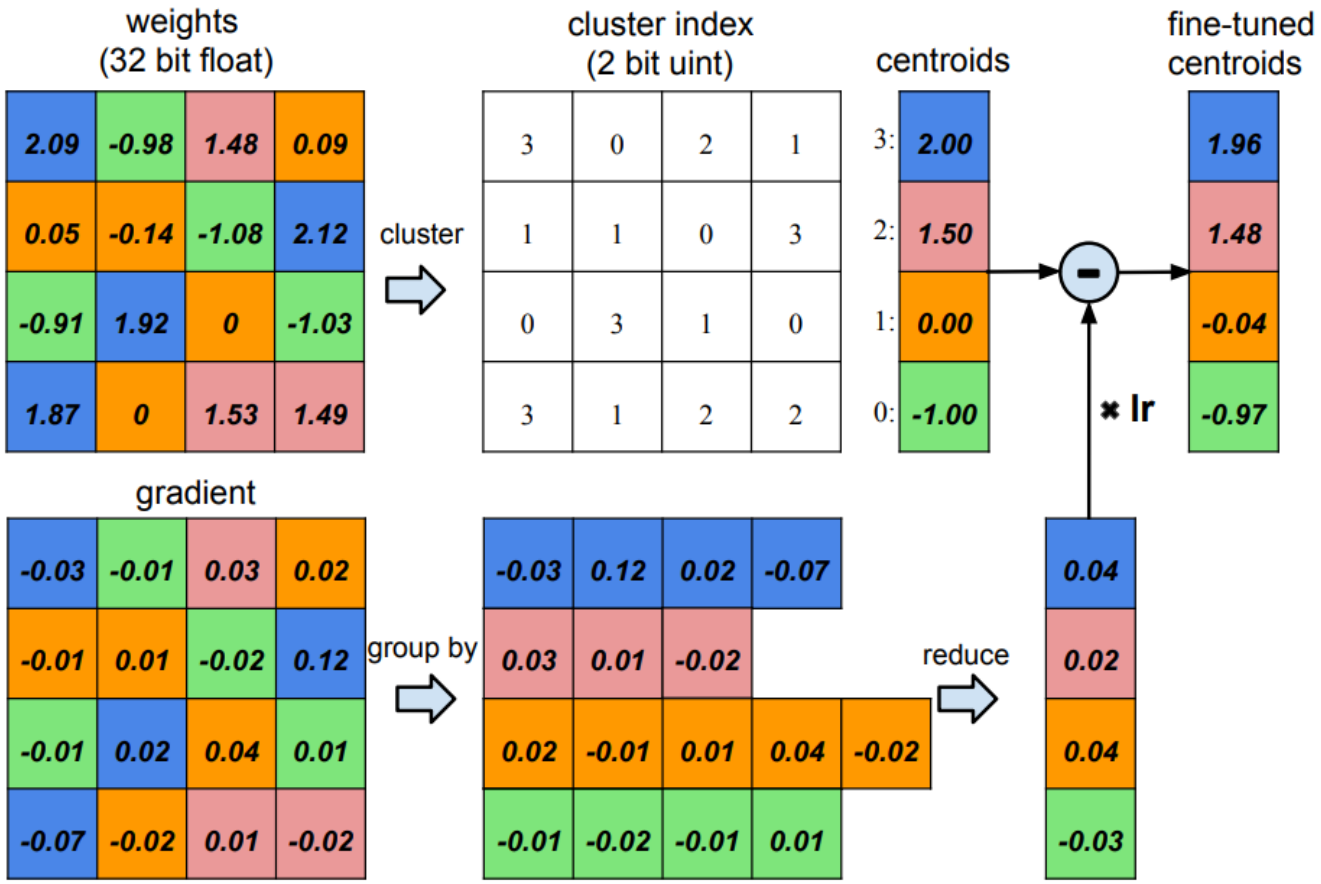
- A matriz esparsa resultante é armazenada utilizando os formatos CSR (Compressed Sparse Row) ou CSC (Compressed Sparse Column)
- É armazenado a diferença entre os índices dos pesos diferentes de zero e não a posição absoluta
    - 8 bits para camadas convolucionais e 5 bits para camadas totalmente conectadas
- Armazenamento: 2 a + n + 1
- Filler Zero



TRAINED QUANTIZATION AND WEIGHT SHARING

- Reduz o número de bits necessários para representar cada peso
- Vários pesos diferentes passam a compartilhar o mesmo valor
- Em vez de usar 32 bits para cada peso, poucos bits são utilizados para indicar um índice, por exemplo 5 ou 8 bits.
- Um novo ciclo de treinamento para refinamento dos pesos compartilhados

TRAINED QUANTIZATION AND WEIGHT SHARING



Compression rate

$$r = \frac{n * b}{(n \log_2 4) + (k * b)}$$

$$r = \frac{16 * 32}{(16 * 2) + (4 * 32)} = 3{,}2$$

TRAINED QUANTIZATION AND WEIGHT SHARING

WEIGHT SHARING

- WCSS (Within-Cluster Sum of Squares)

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{w \in c_i} |w - c_i|^2$$

- Compartilhamento de pesos depois que a rede já foi treinada, favorecendo que os pesos se aproximem dos pesos originais

TRAINED QUANTIZATION AND WEIGHT SHARING

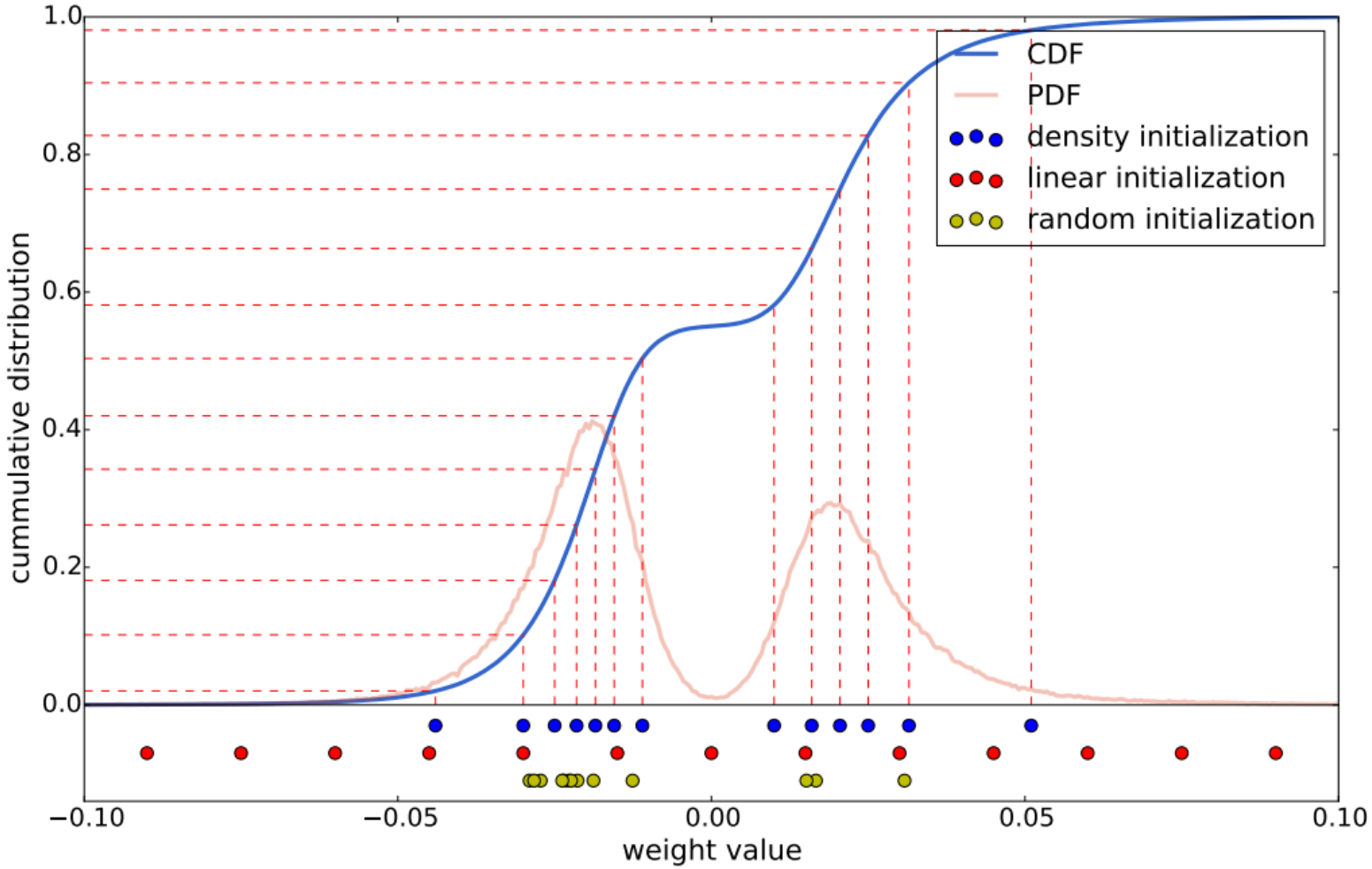
INITIALIZATION OF SHARED WEIGHTS

- Forgy (aleatório): Escolhe alguns pesos originais ao acaso como centróides iniciais;

- Density initialization: Leva em conta a densidade da distribuição dos pesos

- Linear initialization: Distribui os centróides uniformemente entre o menor e o maior valor dos pesos, independente da densidade.

TRAINED QUANTIZATION AND WEIGHT SHARING

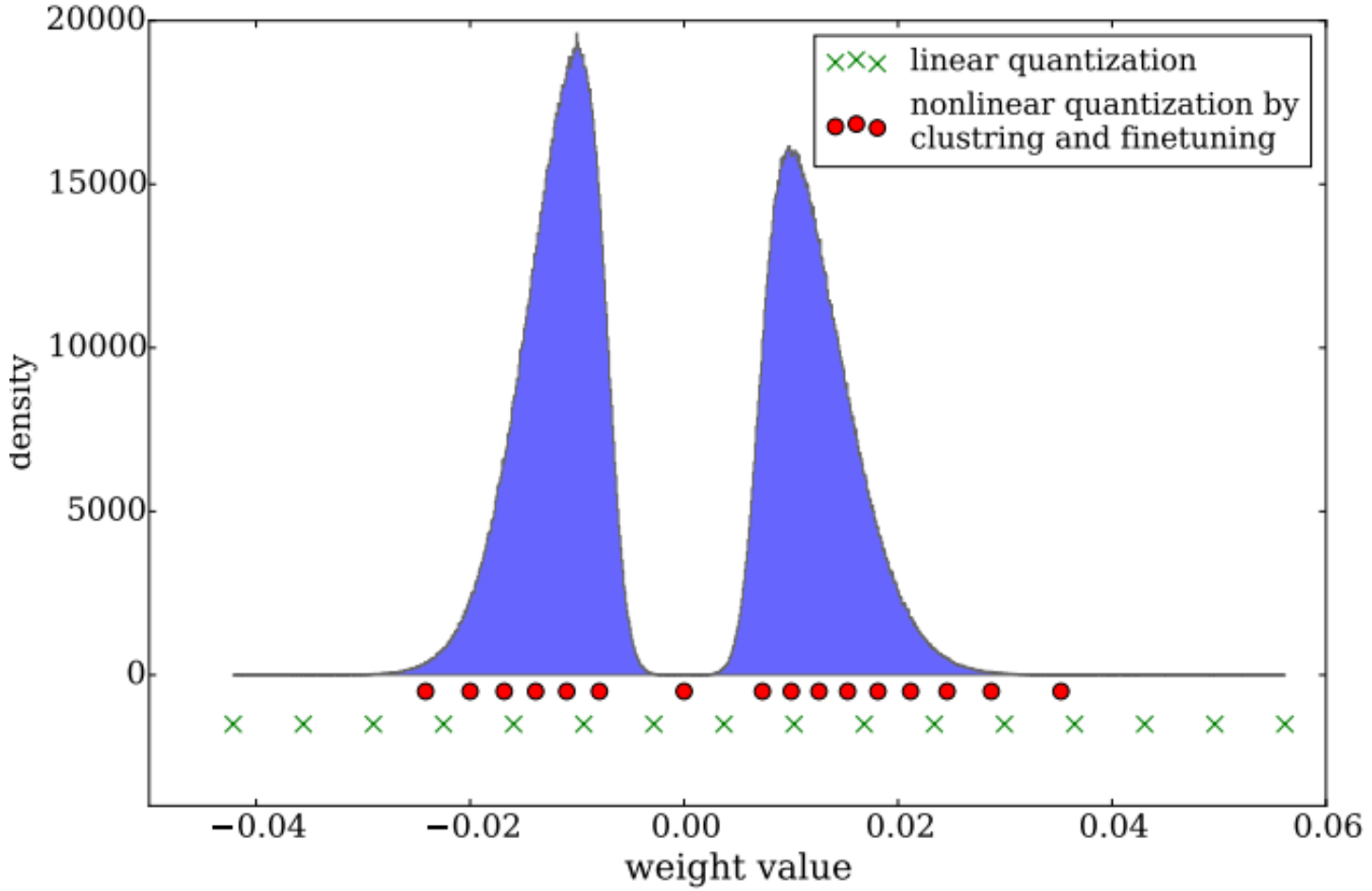


TRAINED QUANTIZATION AND WEIGHT SHARING

INITIALIZATION OF SHARED WEIGHTS

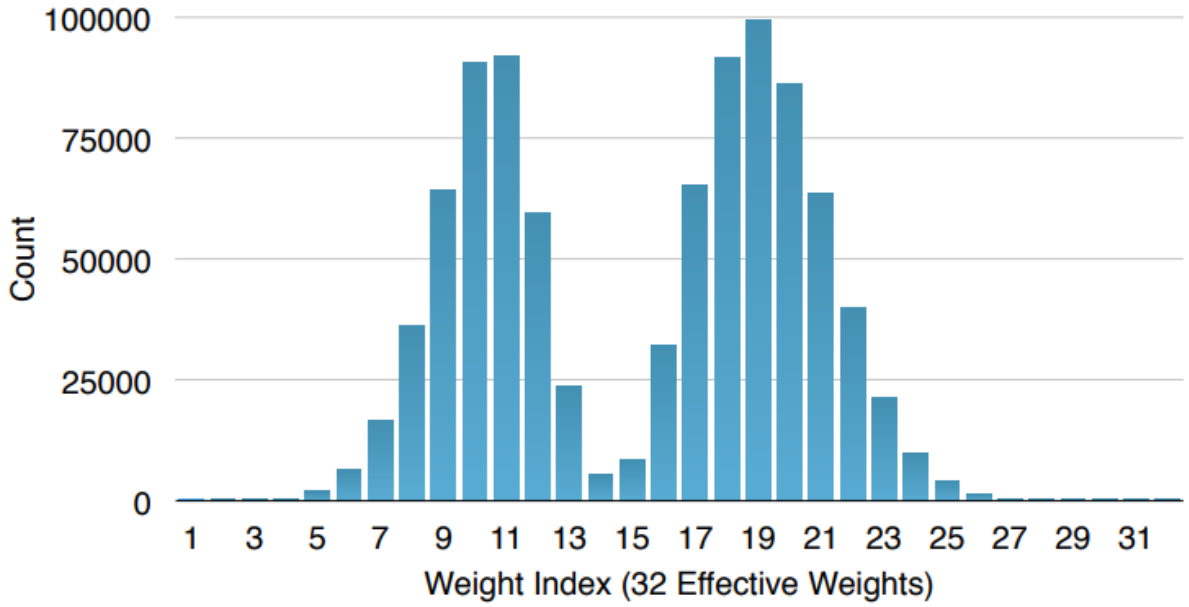
- Pesos com valores absolutos maiores geralmente influenciam mais a saída da rede
- Forgy é aleatório, então poucos centróides acabam escolhendo valores grandes.
- Density-based foca nas regiões densas da distribuição (onde a maioria dos pesos está concentrada, geralmente perto de zero)
- Linear: Garante que mesmo os pesos grandes tenham centróides próximos, melhorando a representação de toda a faixa de valores

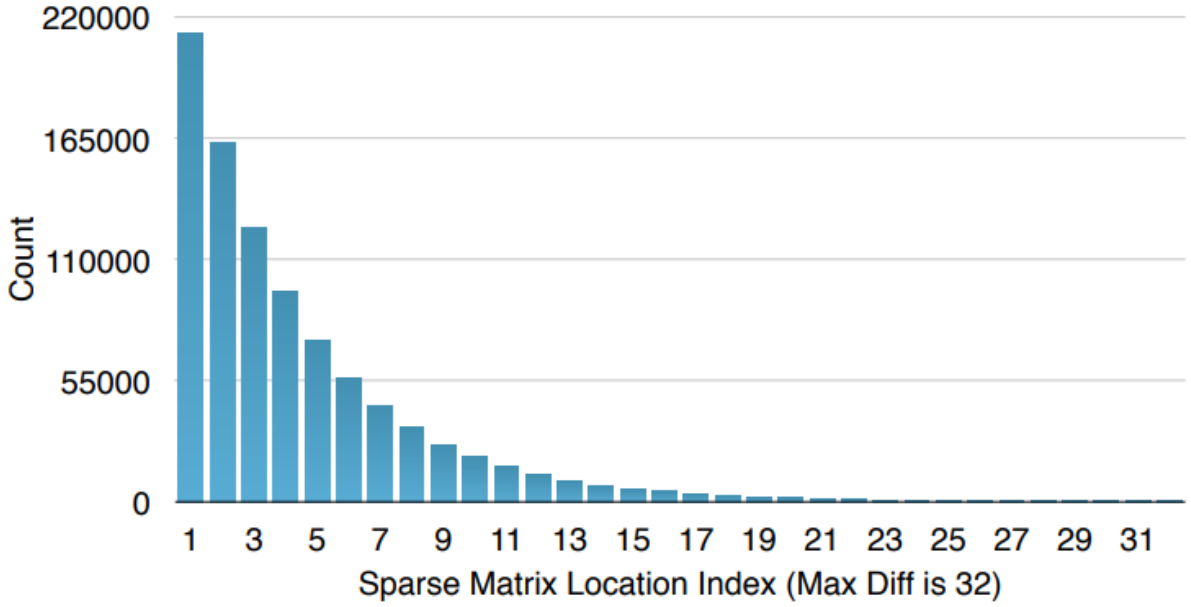
TRAINED QUANTIZATION AND WEIGHT SHARING



Huffman coding

- Palavras-código de comprimento variável para codificar os símbolos de origem
- Símbolos mais comuns são representados com menos bits
- Economia de 20% a 30% de armazenamento para distribuições não uniformes





Experimentos

Table 1: The compression pipeline can save 35× to 49× parameter storage with no loss of accuracy.

Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40**×
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39**×
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35**×
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×

Experimentos

Table 2: Compression statistics for LeNet-300-100. P: pruning, Q:quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
ip1	235K	8%	6	4.4	5	3.7	3.1%	2.32%
ip2	30K	9%	6	4.4	5	4.3	3.8%	3.04%
ip3	1K	26%	6	4.3	5	3.2	15.7%	12.70%
Total	266K	8%(12×)	6	5.1	5	3.7	3.1% (**32**×)	2.49% (**40**×)

Experimentos

Table 3: Compression statistics for LeNet-5. P: pruning, Q:quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	0.5K	66%	8	7.2	5	1.5	78.5%	67.45%
conv2	25K	12%	8	7.2	5	3.9	6.0%	5.28%
ip1	400K	8%	5	4.5	5	4.5	2.7%	2.45%
ip2	5K	19%	5	5.2	5	3.7	6.9%	6.13%
Total	431K	8%(12×)	5.3	4.1	5	4.4	3.05% (**33**×)	2.55% (**39**×)

Experimentos

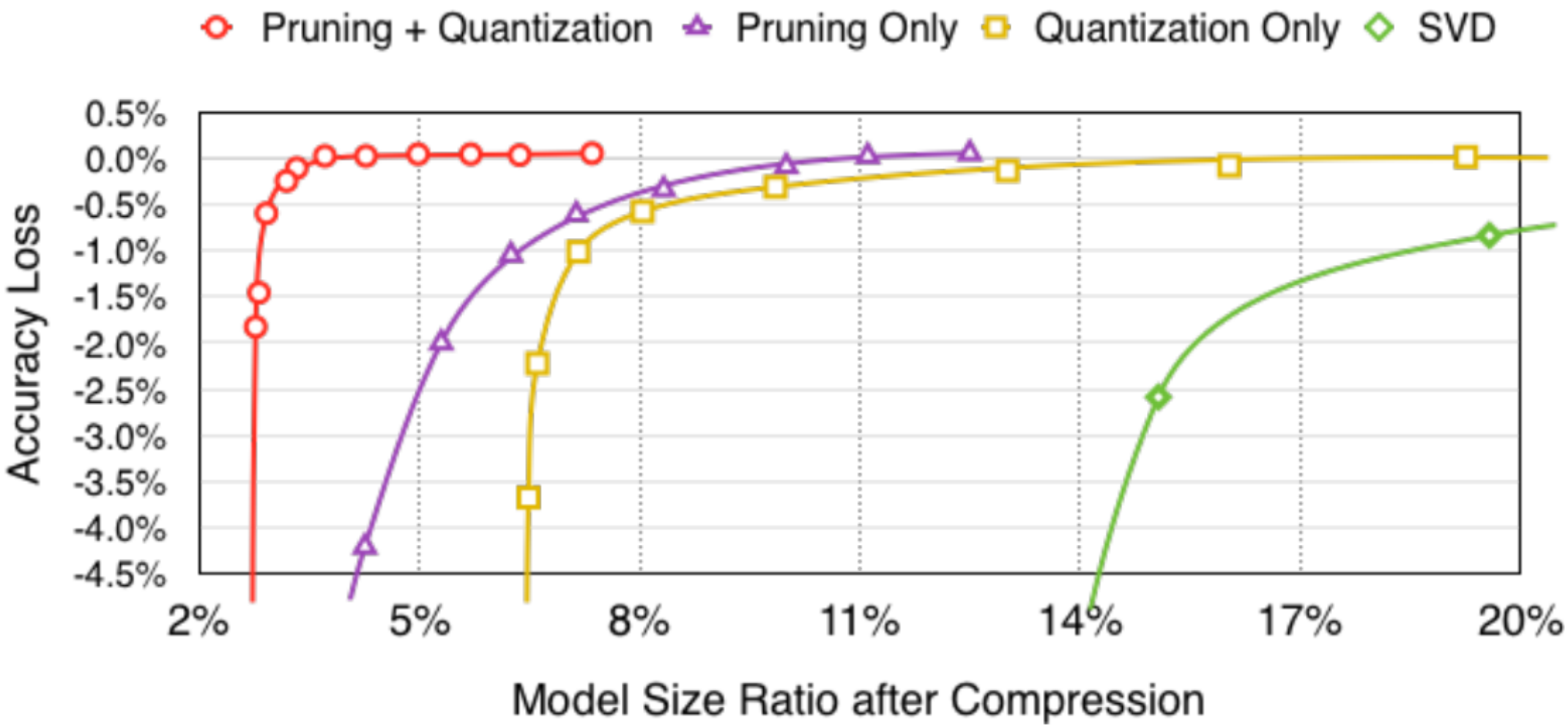
Table 4: Compression statistics for AlexNet. P: pruning, Q: quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	35K	84%	8	6.3	4	1.2	32.6%	20.53%
conv2	307K	38%	8	5.5	4	2.3	14.5%	9.43%
conv3	885K	35%	8	5.1	4	2.6	13.1%	8.44%
conv4	663K	37%	8	5.2	4	2.5	14.1%	9.11%
conv5	442K	37%	8	5.6	4	2.5	14.0%	9.43%
fc6	38M	9%	5	3.9	4	3.2	3.0%	2.39%
fc7	17M	9%	5	3.6	4	3.7	3.0%	2.46%
fc8	4M	25%	5	4	4	3.2	7.3%	5.85%
Total	61M	11%(9×)	5.4	4	4	3.2	3.7% (**27**×)	2.88% (**35**×)

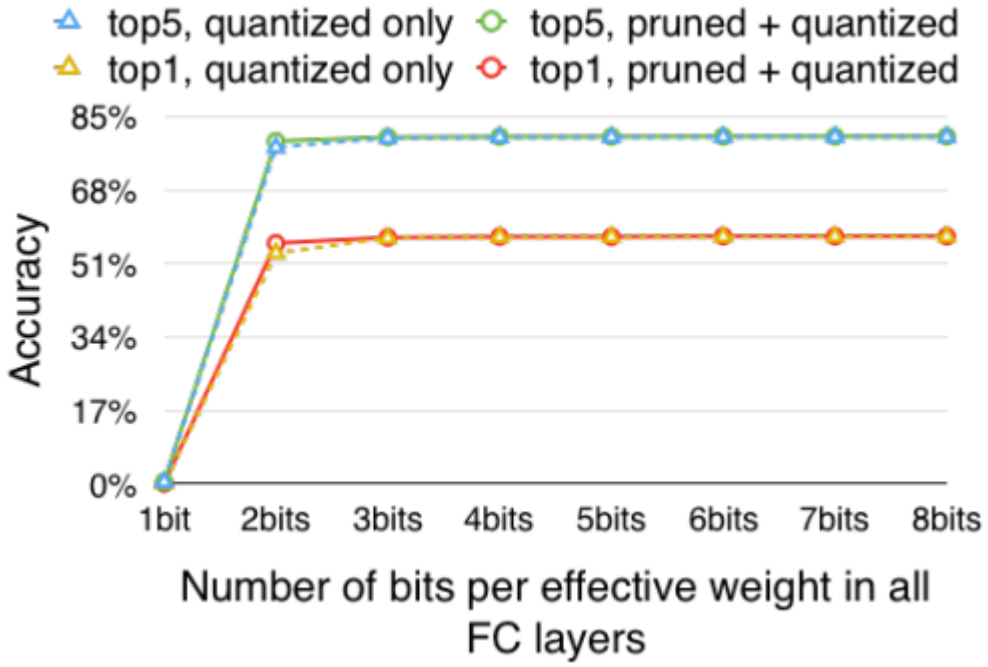
Experimentos

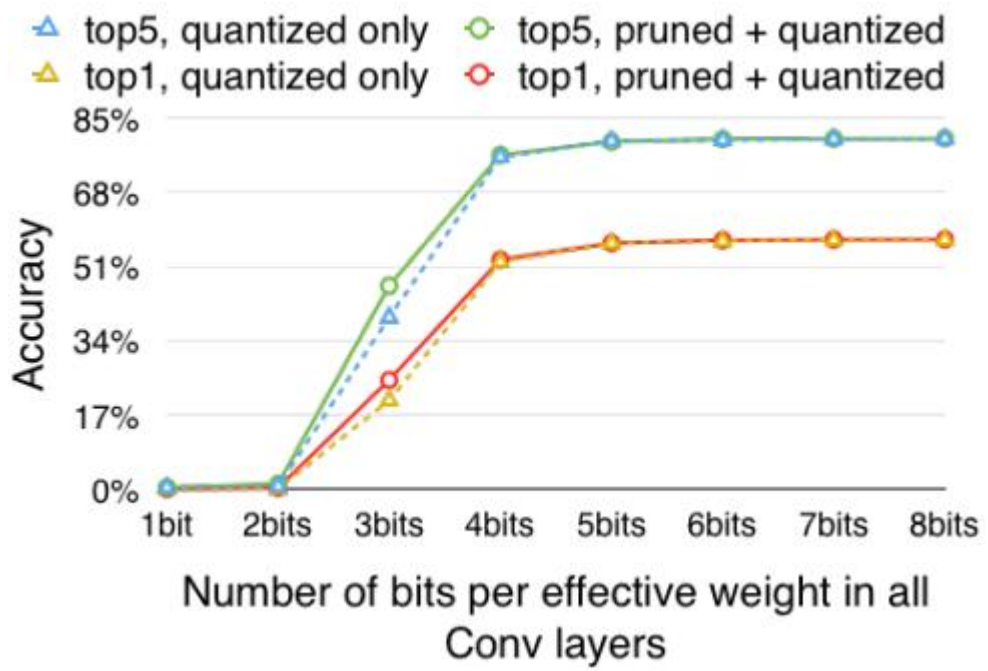
Layer	#Weights	Weights% (P)	Weigh bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1_1	2K	58%	8	6.8	5	1.7	40.0%	29.97%
conv1_2	37K	22%	8	6.5	5	2.6	9.8%	6.99%
conv2_1	74K	34%	8	5.6	5	2.4	14.3%	8.91%
conv2_2	148K	36%	8	5.9	5	2.3	14.7%	9.31%
conv3_1	295K	53%	8	4.8	5	1.8	21.7%	11.15%
conv3_2	590K	24%	8	4.6	5	2.9	9.7%	5.67%
conv3_3	590K	42%	8	4.6	5	2.2	17.0%	8.96%
conv4_1	1M	32%	8	4.6	5	2.6	13.1%	7.29%
conv4_2	2M	27%	8	4.2	5	2.9	10.9%	5.93%
conv4_3	2M	34%	8	4.4	5	2.5	14.0%	7.47%
conv5_1	2M	35%	8	4.7	5	2.5	14.3%	8.00%
conv5_2	2M	29%	8	4.6	5	2.7	11.7%	6.52%
conv5_3	2M	36%	8	4.6	5	2.3	14.8%	7.79%
fc6	103M	4%	5	3.6	5	3.5	1.6%	1.10%
fc7	17M	4%	5	4	5	4.3	1.5%	1.25%
fc8	4M	23%	5	4	5	3.4	7.1%	5.24%
Total	138M	7.5%(13×)	6.4	4.1	5	3.1	3.2% (**31**×)	2.05% (**49**×)

Discussões - Pruning and Quantization Working Together

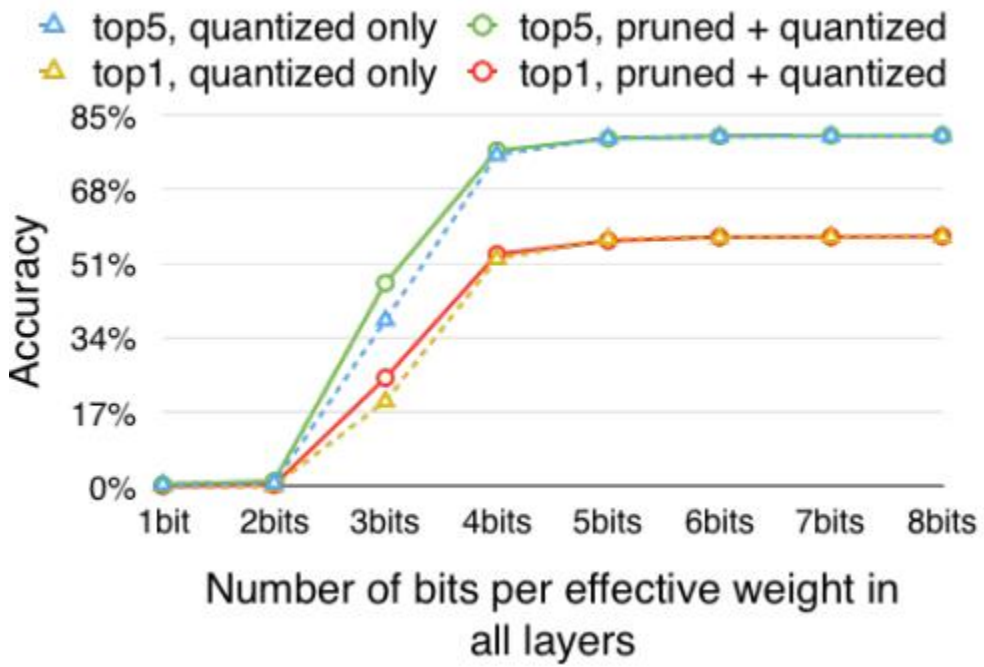


Discussões - Pruning and Quantization Working Together





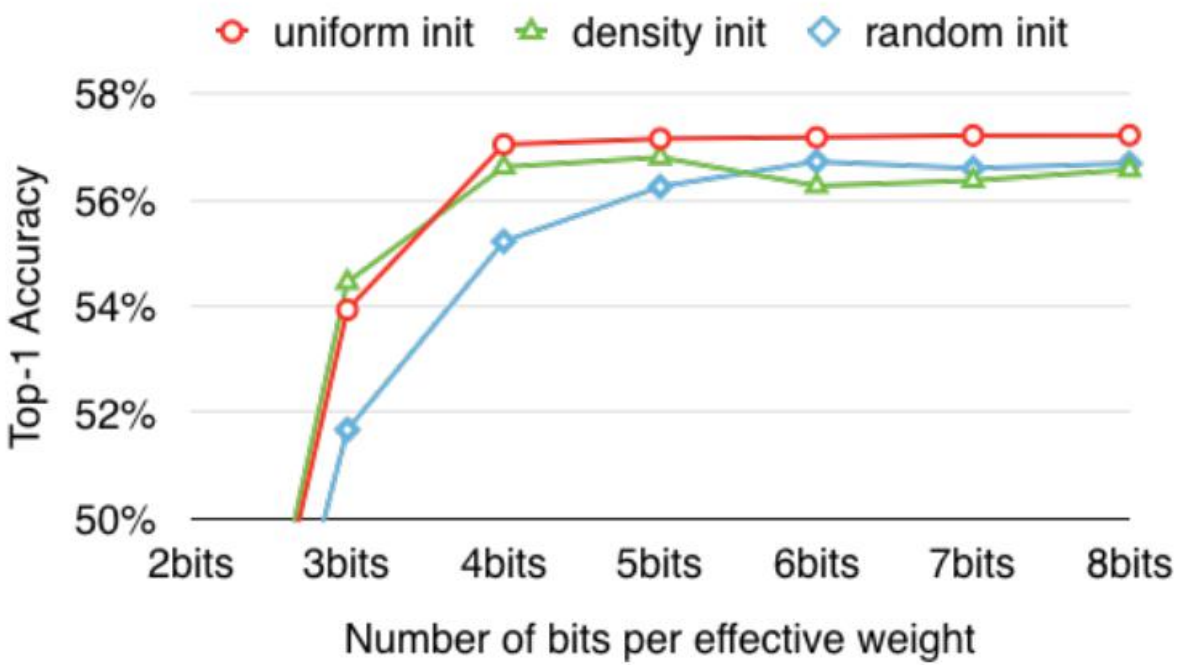
Discussões - Pruning and Quantization Working Together

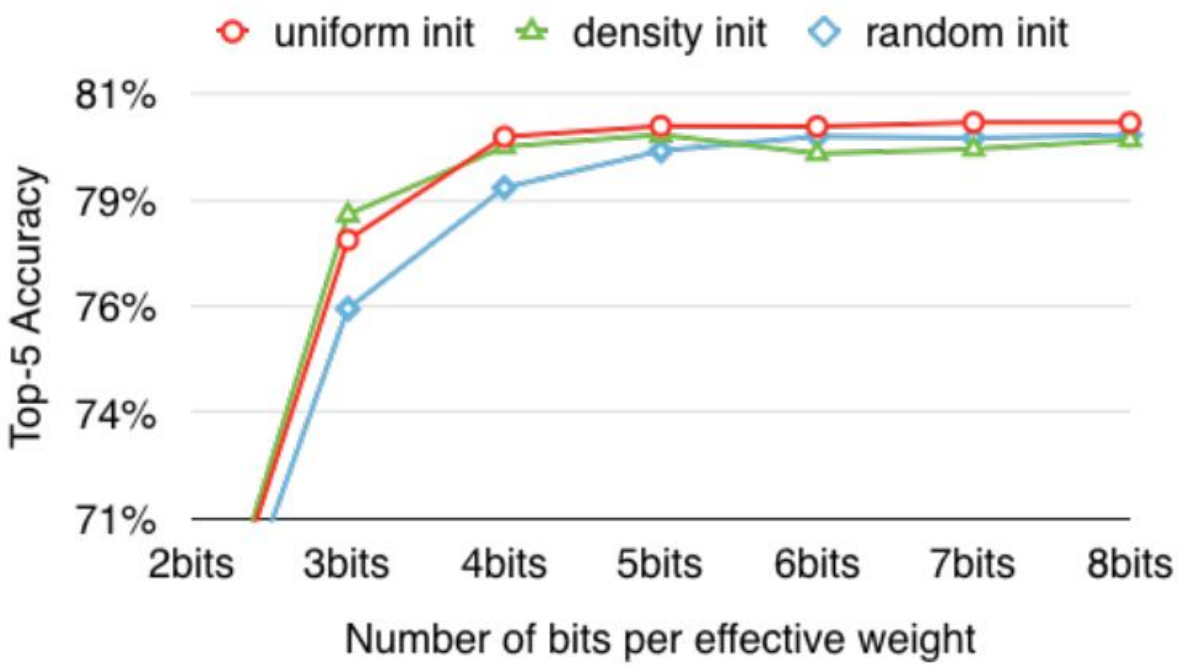


Experimentos

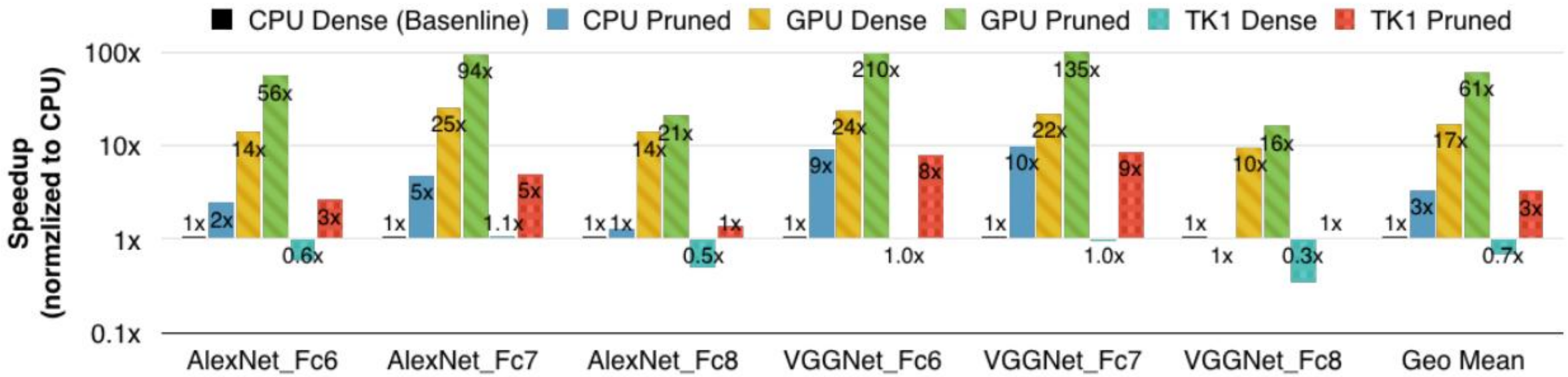
#CONV bits / #FC bits	Top-1 Error	Top-5 Error	Top-1 Error Increase	Top-5 Error Increase
32bits / 32bits	42.78%	19.73%	-	-
8 bits / 5 bits	42.78%	19.70%	0.00%	-0.03%
8 bits / 4 bits	42.79%	19.73%	0.01%	0.00%
4 bits / 2 bits	44.77%	22.33%	1.99%	2.60%

Discussões - Centroid Initialization

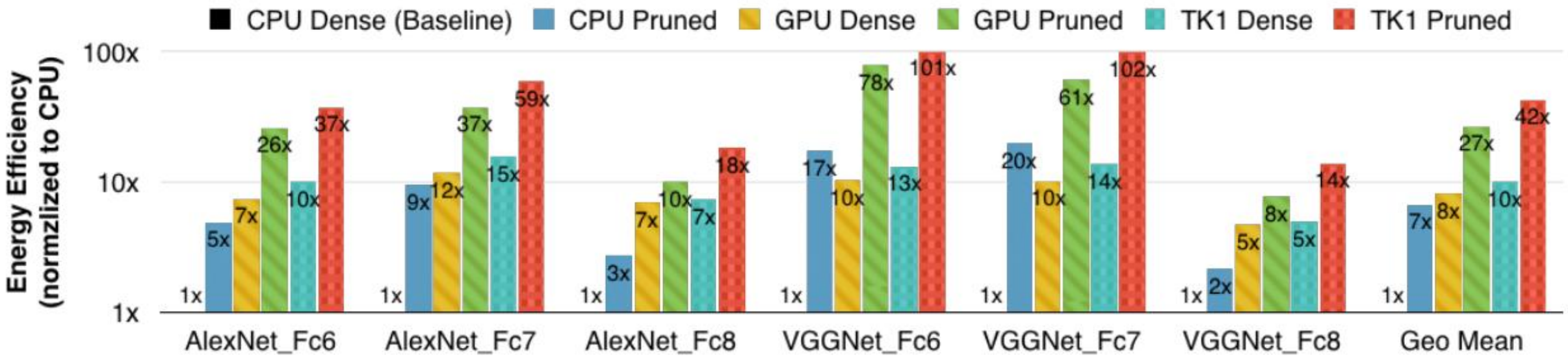




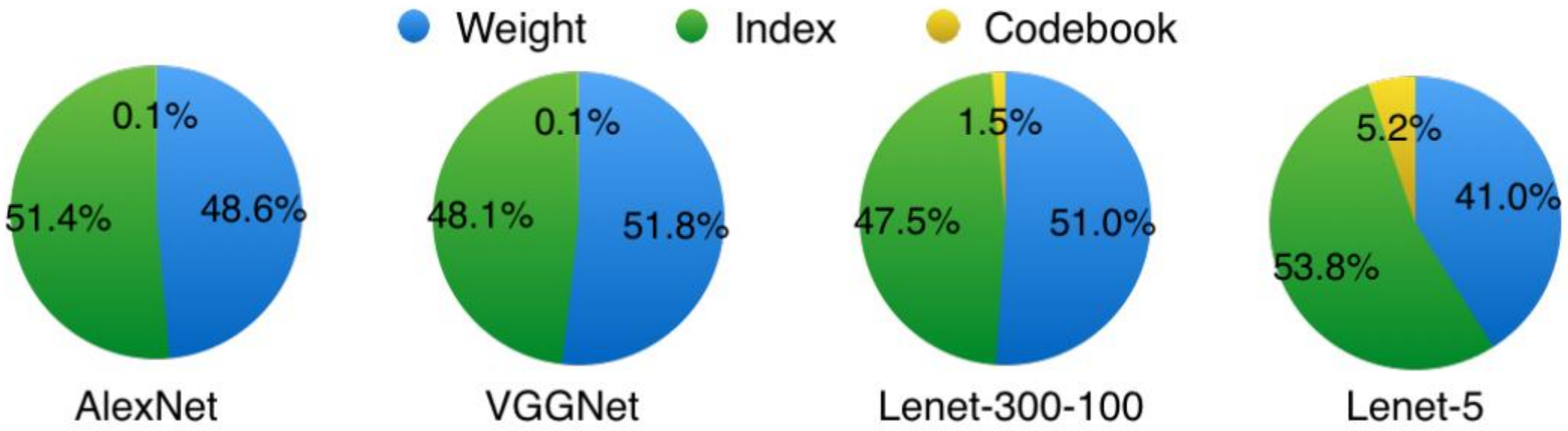
Discussões - Speedup and Energy Efficiency



Discussões - Speedup and Energy Efficiency



Discussões - Ratio of Weights, Index and Codebook



Trabalhos relacionados

Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
Baseline Caffemodel (BVLC)	42.78%	19.73%	240MB	1×
Fastfood-32-AD (Yang et al., 2014)	41.93%	-	131MB	2×
Fastfood-16-AD (Yang et al., 2014)	42.90%	-	64MB	3.7×
Collins & Kohli (Collins & Kohli, 2014)	44.40%	-	61MB	4×
SVD (Denton et al., 2014)	44.02%	20.56%	47.6MB	5×
Pruning (Han et al., 2015)	42.77%	19.67%	27MB	9×
Pruning+Quantization	42.78%	19.70%	8.9MB	27×
Pruning+Quantization+Huffman	**42.78%**	**19.70%**	**6.9MB**	**35×**

Trabalhos futuros e conclusões

- Trabalhos futuros:
  - Pruning avaliado; quantização com weight sharing ainda limitada por bibliotecas padrão.
  - Software: kernels GPU personalizados.
  - Hardware: ASICs especializados com suporte a quantização customizada.

- Resultados:
  - AlexNet: 35× menor.
  - VGG-16: 49× menor.
  - LeNet: 39× menor.

- Benefícios
  - Modelos cabem em SRAM on-chip → menor consumo de energia.
  - Facilita redes complexas em aplicações móveis e embarcadas

Obrigado!

QUIZ

- https://forms.gle/EXJodUTWsvJdTcaE7