SparseBFA: Attacking Sparse Deep Neural Networks with the Worst-Case Bit Flips on Coordinates

Kyungmi Lee, Anantha P. Chandrakasan
Massachusetts Institute of Technology

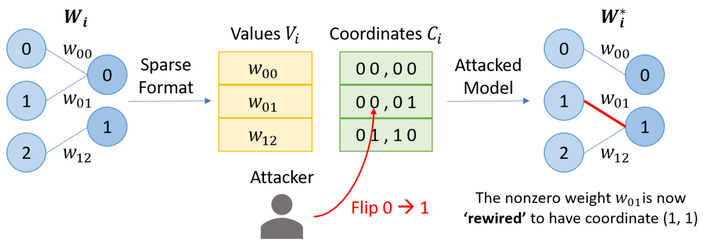
Energy Efficient Circuits and Systems @ MIT

icassp 2022 Singapore

Poster MLSP-40.2

Introduction

- Exploiting **hardware-level faults** to attack DNNs
  - Bit Flip Attack (BFA): flip a few bits among the parameters of a DNN
  - Fault injection attacks (e.g., RowHammer) can induce precise bit flips
- **Sparse DNNs** storing their parameters in sparse matrix formats, which can be exploited by BFAs
  - Bit flip in the coordinates of nonzero weights results in "rewiring" of connections, although it does not change the value itself

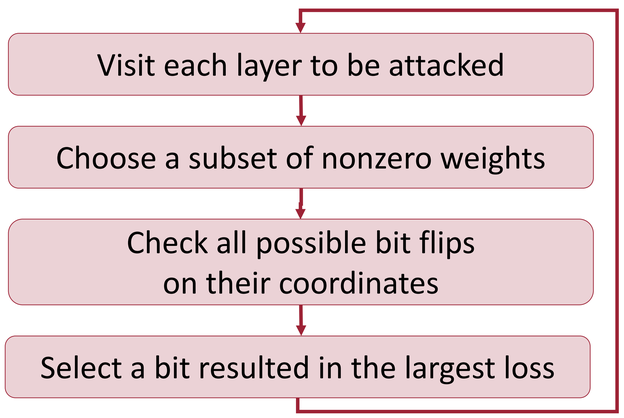


Algorithm: SparseBFA

- Iteratively searches for a single bit in the coordinate lists that maximally increase the loss when it is flipped

Algorithm 1 SparseBFA for the COO format

$m \leftarrow 0$
while $m < M$ **do** ▷ M: the maximum number of iterations
  loss_dict $\leftarrow \emptyset$
  for $i \in S_L$ **do**
    $C_i, V_i \leftarrow$ CONVERT_SPARSE(W_i)
    $S_j \leftarrow$ GET_CANDIDATES_INDEX(C_i)
    for $j \in S_j$ **do**
      for $k \leftarrow 0, d$ **do**
        for $n \leftarrow 0, n_c$ **do**
          $C_i^\star \leftarrow C_i$
          $C_i^\star[j][k][n] \leftarrow 1 - C_i^\star[j][k][n]$
          if IS_VALID($C_i^\star$) **then**
            $W_i^\star \leftarrow$ RECONSTRUCT($C_i^\star, V_i$)
            $\Theta^\star \leftarrow \Theta \setminus \{W_i\} \cup \{W_i^\star\}$
            loss $\leftarrow l(f(\mathbf{x}; \Theta^\star), \mathbf{t})$
            loss_dict$[(i, j, k, n)] \leftarrow$ loss
  $(i_m, j_m, k_m, n_m) \leftarrow$ GET_MAX_KEY(loss_dict)
  $C_{i_m}, V_{i_m} \leftarrow$ CONVERT_SPARSE(W_{i_m})
  $C_{i_m}[j_m][k_m][n_m] \leftarrow 1 - C_{i_m}[j_m][k_m][n_m]$
  $W_{i_m} \leftarrow$ RECONSTRUCT(C_{i_m}, V_{i_m})
  $m \leftarrow m + 1$



- **Complexity of SparseBFA** depends on the second step for choosing a subset

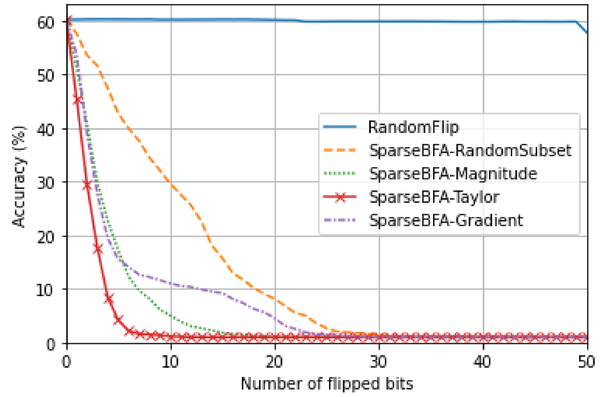
$$W_i = [N_1, \ldots, N_d] \longrightarrow r \cdot \prod_{i=1}^{d} N_i \cdot \sum_{i=1}^{d} \log N_i \quad \text{forward propagations}$$

r: sparsity

  - Choosing all nonzero weights: exhaustive search → Exact, but slow
  - Approximation methods to speed up: quantify importance of nonzero weights similar to pruning (i.e., magnitude, 1st order Taylor's)

Results

- Effectiveness of SparseBFA



  - ResNet-50, TinyImageNet, excluding 1st/last layers from attack
  - Random bit flips do not cause significant accuracy degradation
  - SparseBFA using 1st order Taylor's term as an importance metric shows the best performance (avg. 11.8 bit flips to reach <1% top-1 accuracy)

- DNNs with various architectures and datasets are vulnerable to SparseBFA

Dataset	Model	Sparsity	Total Bits	Original Accuracy	Target Accuracy	# Bit Flips to Reach Target	
						All Layer	Except First, Last
CIFAR-10	Conv4FC2	98.66%	14.26M	86.83%	< 11%	1.4	3.0
	WRN	94.37%	8.34M	93.14%		1.0	12.8
	MobileV2	94.37%	12.98M	90.06%		6.8	8.8
TinyImageNet	ResNet50	94.37%	36.50M	60.18%	< 1% (top-1)	3.2	11.8
	MobileV2	92.50%	17.73M	43.88%		1.4	2.8
PASCAL-VOC	SSD300	82.20%	137.38M	0.70 mAP	0.10 mAP	34.2	-

- First and last layers are more critical: often a single bit flip in the first layer causes degradation to the random guess level
- Sparser DNNs are more vulnerable since SparseBFA has larger search space as sparsity grows
  - Example: Conv4FC2, CIFAR-10, excluding 1st/last layers