Robust Binary Models by Pruning Randomly-initialized Networks

Chen Liu*, Ziqi Zhao*, Sabine Süsstrunk, Mathieu Salzmann
École Polytechnique Fédérale de Lausanne, Switzerland

NEURAL INFORMATION PROCESSING SYSTEMS

Adversarial Robustness

Given the training set $\{(\mathbf{x}_i, y_i)\}_{i=0}^N$, an adversarial budget $\mathcal{S}_\epsilon = \{\Delta | \|\Delta\|_\infty \leq \epsilon\}$, the loss objective $\mathcal{L}$ and a classifier f parameterized by $\boldsymbol{w} \in \mathbb{R}^n$, adversarial training is solving the min-max problem below:

$$\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} \max_{\Delta_i \in \mathcal{S}_\epsilon} \mathcal{L}(f(\boldsymbol{w}, \boldsymbol{x}_i + \Delta_i), y_i). \quad (1)$$

Adversarial training can produce models that are robust to adversarial attacks.

Robust Subnetwork

- The **Strong Lottery Ticket Hypothesis**: within a random overparameterized network, there exists a subnetwork achieving performance similar to that of trained networks with the same number of parameters.
- We extend the Strong Lottery Ticket Hypothesis in the case of **robust binary networks**.
- **Pruning as a way of adversarial training**: given the pruning rate r, we find the optimal pruning mask $\boldsymbol{m}$:

$$\min_{\boldsymbol{m}} \frac{1}{N} \sum_{i=1}^{N} \max_{\Delta_i \in \mathcal{S}_\epsilon} \mathcal{L}(f(\boldsymbol{w} \odot \boldsymbol{m}, \boldsymbol{x}_i + \Delta_i), y_i), \quad (2)$$
$$s.t.\ \boldsymbol{m} \in \{0,1\}^n, sum(\boldsymbol{m}) = (1-r)n.$$

Code on GitHub:

https://github.com/IVRL/RobustBinarySubNet

Framework

Assign a real-valued score s to each model parameter, then the pruning mask $\boldsymbol{m}$ is a function of $\boldsymbol{s}$ and r: $\boldsymbol{m} = M(\boldsymbol{s}, r)$. Parameters with high scores are retained after pruning.

In the forward pass, networks are pruned based on pruning mask $\boldsymbol{m}$ generated by function M

In the backward pass, M is treated as an identity function to obtain the gradient of $\boldsymbol{s}$.

Adaptive Pruning

Pruning Strategies: Allocate $\{m_i\}_{i=1}^L$ parameters in each layer of an L-layer network with $\{n_i\}_{i=1}^L$ parameters such that $\sum_i m_i / \sum_i n_i = 1 - r$

Two extreme pruning strategies:

1. *fixed pruning rate*: $1 - r = \frac{m_1}{n_1} = \frac{m_2}{n_2} = ... = \frac{m_L}{n_L}$

   Too few parameters for small layers with a big r.

2. *fixed number of parameters*: $m_1 = m_2 = ... = m_L$

   Small layers totally unpruned.

We propose **adaptive pruning**

$$1 - r = \frac{\sum_{i=1}^L m_i}{\sum_{i=1}^L n_i}, \frac{m_1}{n_1^p} = \frac{m_2}{n_2^p} = ... = \frac{m_L}{n_L^p} \quad (3)$$

A higher proportion of parameters retained in the smaller layers.

$p \in [0, 1]$: coefficient controlling the trade-off between the two extreme cases. $p = 1$ is the *fixed pruning rate* strategy; $p = 0$ is the *fixed number of parameters* strategy.

Binary Initialization

Binary Initialization: All parameters are initialized as either $+1$ or -1. This saves 45% and 32% FLOPs in training phase and inference phase on a ResNet34 model.

Last Batch Normalization Layer (LBN): An additional batch normalization layer on top of the network to stabilize training.

LBN can 1) avoid gradient explosion and gradient vanishing; 2) make the performance less sensitive to hyper-parameter selection.

Selected Experiments

Comparison with SOTA methods

Method	Architecture	Pruning Strategy	ResNet34 CIFAR10 FP	ResNet34 CIFAR10 Binary	ResNet34 CIFAR100 FP	ResNet34 CIFAR100 Binary	ResNet34 ImageNet100 FP	ResNet34 ImageNet100 Binary	ResNet18 CIFAR10 FP	ResNet18 CIFAR10 Binary	ResNet50 CIFAR10 FP	ResNet50 CIFAR10 Binary
AT	RN	Not Pruned	43.26	40.34	36.63	26.49	53.92	34.20	41.50	39.13	43.24	31.18
AT	RN-LBN	Not Pruned	42.39	39.58	35.15	32.98	55.14	35.36	42.25	39.86	44.33	37.25
AT	Small RN	Not Pruned	38.81	26.03	27.68	15.85	25.40	10.44	28.13	30.35	26.03	32.25
FlyingBird	RN	Dynamic	45.86	34.37	35.91	23.32	37.70	9.54	42.15	27.08	35.91	26.33
FlyingBird+	RN	Dynamic	44.57	33.33	34.30	22.64	37.70	9.52	38.55	27.84	29.54	25.40
BCS	RN	Dynamic	43.51	-	31.85	-	-	-	39.60	-	41.85	-
RST	RN	$p = 1.0$	34.95	-	21.96	-	17.54	-	31.98	-	35.40	-
RST	RN-LBN	$p = 1.0$	37.23	-	23.14	-	15.36	-	33.27	-	34.71	-
HYDRA	RN	$p = 0.1$	42.73	29.28	33.00	23.60	43.18	18.22	40.20	30.90	44.14	22.36
ATMC	RN	Global	34.14	25.62	25.10	11.09	22.18	5.78	32.21	17.73	25.23	6.82
ATMC	RN	$p = 0.1$	34.58	24.62	25.37	11.04	23.52	4.58	32.31	19.67	33.61	16.12
Ours	RN-LBN	$p = 0.1$	-	**45.06**	-	**34.83**	-	**33.04**	-	**39.65**	-	**42.72**
Ours(fast)	RN-LBN	$p = 0.1$	-	40.77	-	34.45			-	30.86	-	37.93

Table 1: Robust accuracy (in %) on the CIFAR10, CIFAR100 and ImageNet100 test sets for the baselines and our proposed method. "RN" represents the ResNet model. "RN-LBN" represents the ResNet with a last batch normalization layer. "Small RN" is a smaller dense ResNet with roughly 1% of the parameters of the ResNet model. The pruning rate r is set to 0.99 except for the not-pruned methods. Among the pruned models, the best results for the full-precision (FP) models are underlined; the best results for the binary models are marked in bold. The adversarial budgets ϵ for CIFAR10, CIFAR100 and ImageNet100 are $8/255$, $4/255$ and $2/255$, respectively. "-" means not applicable or trivial performance.

Pruning Strategies and Pruning Rates

Pruning Strategy	r = 0.5	0.8	0.9	0.95	0.99	0.995	0.998
$p = 0.0$	2.16	6.86	23.01	41.61	44.60	40.70	**34.97**
$p = 0.1$	4.35	15.03	28.12	42.65	**44.88**	**40.97**	33.09
$p = 0.2$	8.01	19.21	27.99	43.72	42.92	40.52	32.99
$p = 0.5$	9.21	32.70	42.84	43.62	42.45	40.55	30.08
$p = 0.8$	28.90	41.51	**43.64**	**43.88**	39.12	33.61	28.07
$p = 0.9$	39.09	41.71	43.07	42.28	38.68	33.89	17.43
$p = 1.0$	**42.85**	**43.23**	42.13	41.12	34.57	26.67	20.56

Table 2: Robust accuracy (in %) of pruned ResNet34 models on the CIFAR10 test set under different pruning rates r and values of p. The adversarial budget is $\epsilon = 8/255$. The best result for each r is marked in bold.

Binary Initialization and LBN Layer

Prune Strategy	Signed KC no LBN	Signed KC LBN	Binary no LBN	Binary LBN
$p = 0.0$	39.38	42.83	40.94	44.65
$p = 0.1$	39.62	45.01	**41.01**	**45.06**
$p = 0.2$	36.66	**45.04**	37.85	41.58
$p = 0.5$	**39.98**	42.64	40.61	39.95
$p = 0.8$	37.96	41.71	35.15	38.95
$p = 0.9$	34.75	40.14	35.64	35.81
$p = 1.0$	36.88	39.32	30.02	30.62

Table 3: Robust accuracy (in %) of pruned ResNet34 on the CIFAR10 test set with the *Signed Kaiming Constant* (Signed KC) and the binary initialization. We include models both with and without the last batch normalization layer (LBN). The best results are marked in bold.

Subnetwork Patterns - Irregular Pruning Leads to Structured Subnetworks

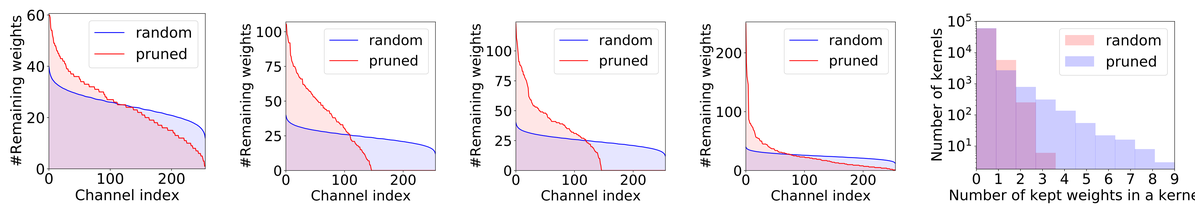

Figure 1: Number of retained parameters in each input and output channel of the first layer (L1) and the second layer (L2) in a random residual block. From left to right: L1 input channel, L1 output channel, L2 input channel, L2 output channel. We sort the numbers and plot the curves from the largest on the left to the smallest on the right. The red curves represent the mask obtained by our method; the blue curves depict what happens when randomly pruning the corresponding layer.

Figure 2: Distribution of the number of retained parameters in each kernel after pruning. The y-axis is in log-scale.