Proximal Mean-field for Neural Network Quantization

Thalaiyasingam Ajanthan[1,2] Puneet K. Dokania[2] Richard Hartley[1] Philip H. S. Torr[2]
[1]Australian National University [2]University of Oxford

Neural Network (NN) Quantization

Objective: Learn a network while the parameters are restricted to a small discrete set:

$$\min_{\mathbf{w}\in\mathcal{Q}^m} L(\mathbf{w};\mathcal{D})\ ,$$

where the set $\mathcal{Q}$ is usually binary, *i.e.*, $\mathcal{Q}=\{-1,1\}$.

Why?

- Reduced memory and time complexity at inference time.
  E.g., Binary $\Rightarrow$ 32 times less memory.
- Better generalization bounds [1].

Approach: Formulate NN quantization as discrete labelling.

Difficulties:

- Exponentially many feasible points ($|\mathcal{Q}|^m$ with $m\approx 10^6$).
- L is highly non-convex.

Relaxations:

- Continuous relaxation of the solution space.
- Iteratively optimize the first-order approximation of L.

Relaxed NN Quantization

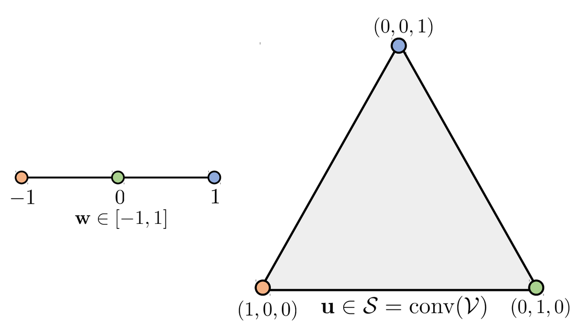
Lifting: Introduce indicator variables. $u_{j:\lambda}=1 \quad\Leftrightarrow\quad w_j=\lambda\in\mathcal{Q}$

Relaxation:

$u_{j:\lambda}=\{0,1\} \quad\Rightarrow\quad u_{j:\lambda}=[0,1]$

For $\mathbf{w}\in\text{conv}(\mathcal{Q})^m$, $\mathbf{w}=\mathbf{u}\mathbf{q}$, where

$$\mathbf{u}\in\mathcal{S}=\left\{\mathbf{u}\;\middle|\;\begin{array}{ll}\sum_\lambda u_{j:\lambda}=1, & \forall j\\ u_{j:\lambda}\in[0,1], & \forall j,\lambda\end{array}\right\}\ ,$$



where $\mathbf{q}$ is the vector of quantization levels.

$u_{j:\lambda}\in\mathcal{S}$ is the probability of parameter w_j taking label $\lambda\in\mathcal{Q}$

- Any local minimum in the $\mathbf{u}$-space is also a local minimum in the relaxed $\mathbf{w}$-space and vice versa.

$$\min_{\mathbf{u}\in\mathcal{S}} L(\mathbf{u}\mathbf{q};\mathcal{D}) \equiv \min_{\mathbf{w}\in\text{conv}(\mathcal{Q})^m} L(\mathbf{w};\mathcal{D})$$

Projected (Stochastic) Gradient Descent (PGD)

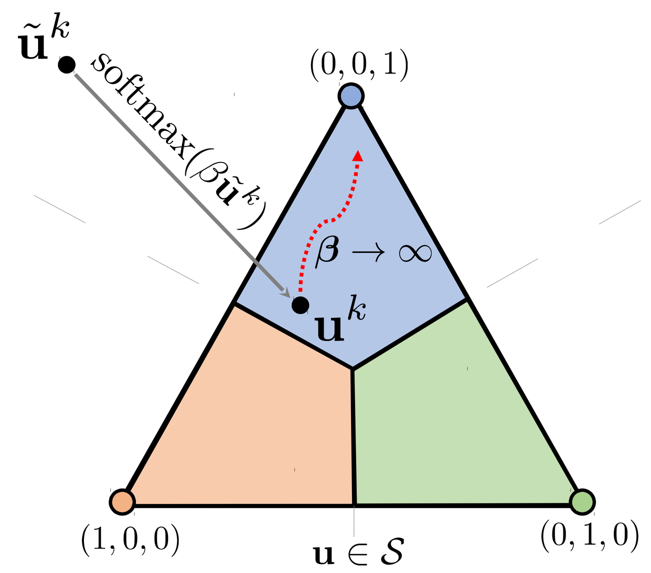
At iteration k,

$$\tilde{\mathbf{u}}^{k+1}=\mathbf{u}^k-\eta\,\mathbf{g}^k\ ,\qquad \text{SGD}$$
$$\mathbf{u}^{k+1}=P_{\mathcal{S}}\left(\tilde{\mathbf{u}}^{k+1}\right)\ ,$$

where $\eta>0$ and $\mathbf{g}^k$ is the (stochastic) gradient of L evaluated at $\mathbf{u}^k$.

- Any off-the-shelf SGD algorithm can be used.

Softmax Projection and Exploration



- Preserves relative order of $u_{j:\lambda}$.
- Differentiable.

Ultimate objective: A quantized solution
$\Rightarrow$ $\mathbf{u}\in\mathcal{V}$ is attained when $\beta\to\infty$.

Softmax $\Rightarrow$ noisy projection to $\mathcal{V}$

Softmax based PGD as Proximal Mean-field (PMF)

Mean-field:

$$\underset{\mathbf{u}\in\mathcal{S}}{\text{argmin}}\ \text{KL}(\mathbf{u}\|\text{P})=\underset{\mathbf{u}\in\mathcal{S}}{\text{argmin}}\ \mathbb{E}_{\mathbf{u}}[L(\mathbf{w})]-H(\mathbf{u})\ ,$$

where $\mathbb{E}_{\mathbf{u}}[\cdot]$ is the expectation over $\mathbf{u}$ and $H(\mathbf{u})$ is the entropy.

- For a neural network $L(\mathbf{w})$ has no explicit factorization.

Proximal Mean-field: Replace $L(\mathbf{w})$ with the first-order approximation $\hat{L}^k(\mathbf{w})$ augmented by a proximal term.

Softmax based PGD $\equiv$ Proximal Mean-field

At iteration k,

$$\mathbf{u}^{k+1}=\text{softmax}\left(\beta\left(\mathbf{u}^k-\eta\,\mathbf{g}^k\right)\right)\ ,\qquad \text{PGD}$$
$$\mathbf{u}^{k+1}=\underset{\mathbf{u}\in\mathcal{S}}{\text{argmin}}\ \eta\,\mathbb{E}_{\mathbf{u}}\left[\hat{L}^k(\mathbf{w})\right]-\left\langle\mathbf{u}^k,\mathbf{u}\right\rangle_F-\frac{1}{\beta}H(\mathbf{u})\ .\qquad \text{PMF}$$

- $\hat{L}^k(\mathbf{w})$ is the first-order Taylor approximation of L at $\mathbf{w}^k=\mathbf{u}^k\mathbf{q}$.
- Negative of cosine similarity $\Rightarrow$ proximal term.
- Entropy term vanishes when $\beta\to\infty$.

Binary Connect (BC) [3] is a special case of PMF

Final PMF Algorithm

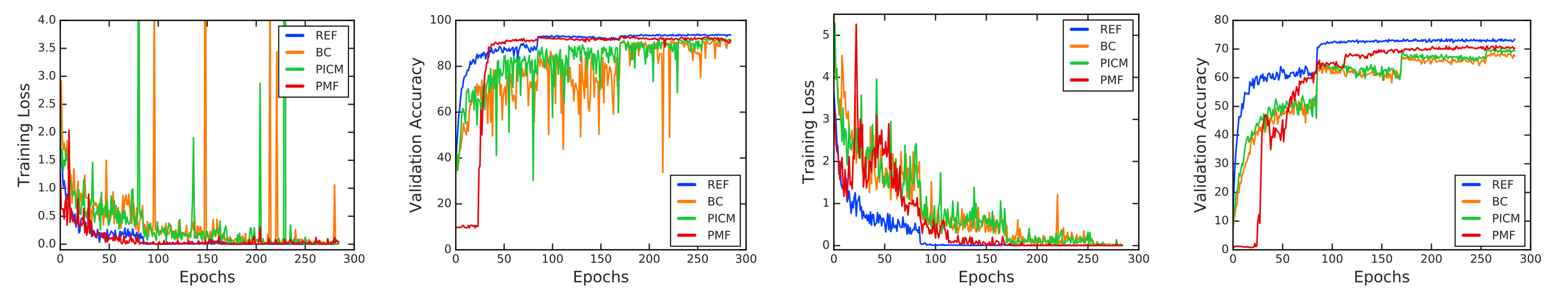
Algorithm One iteration of PMF

$\mathbf{u}^k\leftarrow\text{softmax}(\beta\tilde{\mathbf{u}}^k)$	$\triangleright$ Projection
$\mathcal{D}^b=\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^b\sim\mathcal{D}$	$\triangleright$ Sample a mini-batch
$\mathbf{g}_{\mathbf{u}}^k\leftarrow\nabla_{\mathbf{u}}\tilde{L}(\mathbf{u};\mathcal{D}^b)\big\rvert_{\mathbf{u}=\mathbf{u}^k}$	$\triangleright$ Gradient w.r.t. $\mathbf{u}$ at $\mathbf{u}^k$
$\mathbf{g}_{\tilde{\mathbf{u}}}^k\leftarrow\mathbf{g}_{\mathbf{u}}^k\frac{\partial\mathbf{u}}{\partial\tilde{\mathbf{u}}}\big\rvert_{\tilde{\mathbf{u}}=\tilde{\mathbf{u}}^k}$	$\triangleright$ Gradient w.r.t. $\tilde{\mathbf{u}}$ at $\mathbf{u}^k$
$\tilde{\mathbf{u}}^{k+1}\leftarrow\tilde{\mathbf{u}}^k-\eta^k\mathbf{g}_{\tilde{\mathbf{u}}}^k$	$\triangleright$ Gradient descent on $\tilde{\mathbf{u}}$
$\beta\leftarrow\rho\beta$	$\triangleright$ Increase β, initialized to 1

Results

Dataset	Architecture	REF (32 bit)	BC [3] (1 bit)	PQ [2] (1 bit)*	PMF (1 bit)
MNIST	LeNet-300	98.55	98.05	98.13	**98.24**
	LeNet-5	99.39	99.30	99.27	**99.44**
CIFAR-10	VGG-16	93.01	86.40	90.11	**90.51**
	ResNet-18	94.64	91.60	92.32	**92.73**
CIFAR-100	VGG-16	70.33	43.70	55.10	**61.52**
	ResNet-18	73.85	69.93	68.35	**71.85**
TinyImageNet	ResNet-18	56.41	49.33	49.97	**51.00**

Classification accuracies on the test set for different methods.



CIFAR-10, ResNet-18 CIFAR-100, ResNet-18

PMF is less noisy and closely resembles the reference network

Code: `https://github.com/tajanthan/pmf`

References

[1] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. *ICML*, 2018.
[2] Y. Bai, Y.-X. Wang, and E. Liberty. Proxquant: Quantized neural networks via proximal operators. *ICLR*, 2019.
[3] M. Courbariaux, Y. Bengio, and J.-P. David. Binaryconnect: Training deep neural networks with binary weights during propagations. *NIPS*, 2015.