Proximal Mean-field for Neural Network Quantization

Thalaiyasingam Ajanthan

University of Oxford

ANU, January 2019

Collaborators

Puneet K. Dokania

Richard Hartley

Philip Torr

Neural Network (NN) Quantization

Objective

- Learn a network while the parameters are restricted to a small discrete set.

Why?

- Reduced memory and time complexity at inference time. *E.g.* Binary $\Rightarrow$ 32 times less memory
- Better generalization bounds? [Arora-2018]
- Robustness to adversarial examples?

Idea

- Formulate NN quantization as a discrete labelling problem.

Neural Network (NN) Quantization

Objective

- Learn a network while the parameters are restricted to a small discrete set.

Why?

- Reduced memory and time complexity at inference time. *E.g.* Binary $\Rightarrow$ 32 times less memory
- Better generalization bounds? [Arora-2018]
- Robustness to adversarial examples?

Idea

- Formulate NN quantization as a discrete labelling problem.

Neural Network (NN) Quantization

Objective

- Learn a network while the parameters are restricted to a small discrete set.

Why?

- Reduced memory and time complexity at inference time. *E.g.* Binary $\Rightarrow$ 32 times less memory
- Better generalization bounds? [Arora-2018]
- Robustness to adversarial examples?

Idea

- Formulate NN quantization as a discrete labelling problem.

Neural Network (NN) Quantization

Objective

- Learn a network while the parameters are restricted to a small discrete set.

Why?

- Reduced memory and time complexity at inference time. *E.g.* Binary $\Rightarrow$ 32 times less memory
- Better generalization bounds? [Arora-2018]
- Robustness to adversarial examples?

Idea

- Formulate NN quantization as a discrete labelling problem.

NN quantization as MRF optimization

NN Quantization as Discrete Labelling

$$\min_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}) := \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) \ ,$$
$$\mathbf{w} \in \mathcal{Q}^m \ .$$

- $\mathcal{D}$ Dataset ($\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n}$)
- ℓ Loss function (I/O mapping + cross-entropy)
- $\mathbf{w}$ Learnable parameters (m)
- $\mathcal{Q}$ Set of quantization levels ($\mathcal{Q} = \{-1, 1\}$)

$w_j \in \mathcal{Q}$

Difficulties

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

NN Quantization as Discrete Labelling

$$\min_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) \ ,$$
$$\mathbf{w} \in \mathcal{Q}^m \ .$$

- $\mathcal{D}$ Dataset $(\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{n})$
- ℓ Loss function (I/O mapping + cross-entropy)
- $\mathbf{w}$ Learnable parameters (m)
- $\mathcal{Q}$ Set of quantization levels $(\mathcal{Q} = \{-1, 1\})$

$\boxed{w_j \in \mathcal{Q}}$

Difficulties

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

NN Quantization as Discrete Labelling

$$\min_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}) := \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}; (\mathbf{x}_i, \mathbf{y}_i)) \ ,$$
$$\mathbf{w} \in \mathcal{Q}^m \ .$$

Difficulties

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

Idea

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

Lifting and Relaxation

Lifting: Indicator variables

$$u_{j:\lambda} = 1 \quad \Leftrightarrow \quad w_j = \lambda \in \mathcal{Q}$$

For $\mathbf{w} \in \mathcal{Q}^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q} \ ,$$

$$\text{s.t.} \quad \mathbf{u} \in \mathcal{V} = \left\{ \mathbf{u} \;\middle|\; \begin{array}{ll} \sum_\lambda u_{j:\lambda} = 1, & \forall j \\ u_{j:\lambda} \in \{0,1\}, & \forall j, \lambda \end{array} \right\} ,$$

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

Lifting and Relaxation

Lifting: Indicator variables

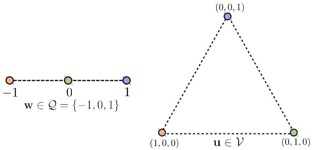
$$u_{j:\lambda} = 1 \quad \Leftrightarrow \quad w_j = \lambda \in \mathcal{Q}$$

For $\mathbf{w} \in \mathcal{Q}^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q} \ ,$$

$$\text{s.t.} \quad \mathbf{u} \in \mathcal{V} = \left\{ \ \mathbf{u} \ \middle| \ \begin{array}{ll} \sum_\lambda u_{j:\lambda} = 1, & \forall\, j \\ u_{j:\lambda} \in \{0,1\}, & \forall\, j, \lambda \end{array} \right\} ,$$

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



Lifting and Relaxation

Lifting: Indicator variables

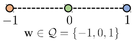
$$u_{j:\lambda} = 1 \quad \Leftrightarrow \quad w_j = \lambda \in \mathcal{Q}$$

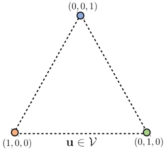
For $\mathbf{w} \in \mathcal{Q}^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q} \ ,$$

$$\text{s.t.} \quad \mathbf{u} \in \mathcal{V} = \left\{ \mathbf{u} \;\middle|\; \begin{array}{ll} \sum_\lambda u_{j:\lambda} = 1, & \forall j \\ u_{j:\lambda} \in \{0,1\}, & \forall j, \lambda \end{array} \right\} \ ,$$

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





$$\mathcal{Q}^m \quad \Leftrightarrow \quad \mathcal{V}$$

Lifting and Relaxation

Relaxation

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

For $\mathbf{w} \in \mathrm{conv}(\mathcal{Q})^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q} \ ,$$

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

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

- $\mathcal{S}$ decomposes over j.

Lifting and Relaxation

Relaxation

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

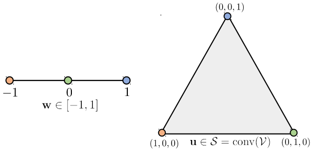
For $\mathbf{w} \in \mathrm{conv}(\mathcal{Q})^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q} \ ,$$

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

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

- $\mathcal{S}$ decomposes over j.



Lifting and Relaxation

Relaxation

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

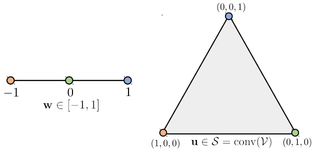
For $\mathbf{w} \in \text{conv}(\mathcal{Q})^m$,

$$\mathbf{w} = \mathbf{u}\mathbf{q}\ ,$$

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

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

- $\mathcal{S}$ decomposes over j.



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

Relaxed Optimization Problem

$$\min_{\mathbf{u}} \tilde{L}(\mathbf{u}; \mathcal{D}) := L(\mathbf{uq}; \mathcal{D}) \ ,$$
$$\mathbf{u} \in \mathcal{S} \ .$$

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

Relaxed Optimization Problem

$$\min_{\mathbf{u}} \tilde{L}(\mathbf{u}; \mathcal{D}) := L(\mathbf{uq}; \mathcal{D}) \ ,$$
$$\mathbf{u} \in \mathcal{S} \ .$$

- Any local minimum in the **u**-space is also a local minimum in the relaxed **w**-space and vice versa.

Relaxed Optimization Problem

$$\min_{\mathbf{u}} \tilde{L}(\mathbf{u}; \mathcal{D}) := L(\mathbf{u}\mathbf{q}; \mathcal{D}) \ ,$$

$$\mathbf{u} \in \mathcal{S} \ .$$

- Any local minimum in the **u**-space is also a local minimum in the relaxed **w**-space and vice versa.

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

Projected (Stochastic) Gradient Descent (PGD)

At iteration k,

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

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

- Any off-the-shelf SGD algrorithm can be used.
- For softmax projection, PGD ≡ Proximal Mean-Field.

- Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].

Projected (Stochastic) Gradient Descent (PGD)

At iteration k,

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

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

- Any off-the-shelf SGD algrorithm can be used.
- For softmax projection, PGD $\equiv$ Proximal Mean-Field.
- Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].

Projected (Stochastic) Gradient Descent (PGD)

At iteration k,

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

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

- Any off-the-shelf SGD algrorithm can be used.
- For softmax projection, PGD $\equiv$ Proximal Mean-Field.

- Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].

Projected (Stochastic) Gradient Descent (PGD)

At iteration k,

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

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

- Any off-the-shelf SGD algroritihm can be used.
- For softmax projection, PGD $\equiv$ Proximal Mean-Field.

- Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].

Softmax Projection and Exploration

For each $j \in \{1 \dots m\}$,

$$\mathbf{u}_j^k = \text{softmax}(\beta \tilde{\mathbf{u}}_j^k) \ ,$$

$$u_{j:\lambda}^k = \frac{e^{\beta\left(\tilde{u}_{j:\lambda}^k\right)}}{\sum_{\mu \in \mathcal{Q}} \ e^{\beta\left(\tilde{u}_{j:\mu}^k\right)}} \quad \forall \, \lambda \in \mathcal{Q} \ ,$$

where $\beta > 0$.

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

Ultimate objective

- A quantized solution $\Rightarrow$ $\mathbf{u} \in \mathcal{V}$ attained when $\beta \to \infty$.

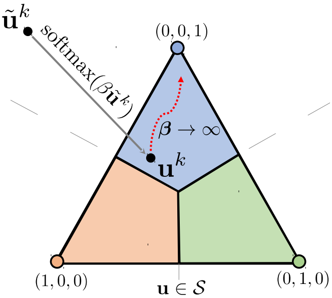
Softmax Projection and Exploration

For each $j \in \{1 \dots m\}$,

$$\mathbf{u}_j^k = \text{softmax}(\beta \tilde{\mathbf{u}}_j^k) \ ,$$

$$u_{j:\lambda}^k = \frac{e^{\beta\left(\tilde{u}_{j:\lambda}^k\right)}}{\sum_{\mu \in \mathcal{Q}} \ e^{\beta\left(\tilde{u}_{j:\mu}^k\right)}} \quad \forall \lambda \in \mathcal{Q} \ ,$$

where $\beta > 0$.



Ultimate objective

- A quantized solution $\Rightarrow$ $\mathbf{u} \in \mathcal{V}$ attained when $\beta \rightarrow \infty$.

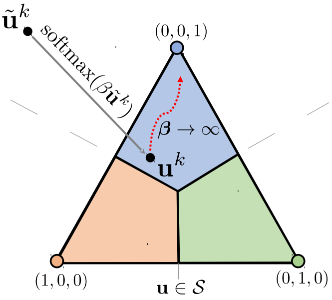
Softmax Projection and Exploration

For each $j \in \{1 \ldots m\}$,

$$\mathbf{u}_j^k = \text{softmax}(\beta \tilde{\mathbf{u}}_j^k) \ ,$$

$$u_{j:\lambda}^k = \frac{e^{\beta\left(\tilde{u}_{j:\lambda}^k\right)}}{\sum_{\mu \in \mathcal{Q}} e^{\beta\left(\tilde{u}_{j:\mu}^k\right)}} \quad \forall \, \lambda \in \mathcal{Q} \ ,$$

where $\beta > 0$.



Ultimate objective

- A quantized solution $\Rightarrow$ $\mathbf{u} \in \mathcal{V}$ attained when $\beta \to \infty$.

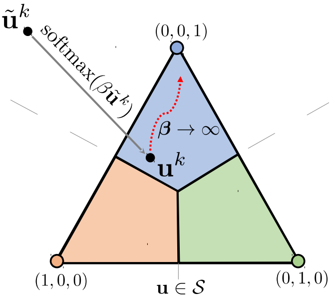
Softmax Projection and Exploration

For each $j \in \{1 \dots m\}$,

$$\mathbf{u}_j^k = \text{softmax}(\beta \tilde{\mathbf{u}}_j^k) \; ,$$

$$u_{j:\lambda}^k = \frac{e^{\beta\left(\tilde{u}_{j:\lambda}^k\right)}}{\sum_{\mu \in \mathcal{Q}} \; e^{\beta\left(\tilde{u}_{j:\mu}^k\right)}} \quad \forall \, \lambda \in \mathcal{Q} \; ,$$

where $\beta > 0$.



Ultimate objective

- A quantized solution $\Rightarrow$ $\mathbf{u} \in \mathcal{V}$ attained when $\beta \to \infty$.

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

Mean-Field Method

Let $L(\mathbf{w})$ be the energy (or loss), then

$$P(\mathbf{w}) = \frac{1}{Z} e^{-L(\mathbf{w})} \ .$$

- Mean-field approximates $P(\mathbf{w})$ with a fully-factorized distribution $U(\mathbf{w}) = \prod_{j=1}^{m} U_j(w_j)$.
- From the probabilistic interpretation of $\mathbf{u} \in \mathcal{S}$, $U_j(w_j = \lambda) = u_{j:\lambda}$, for each $j \in \{1 \ldots m\}$.

Objective

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

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

Mean-Field Method

Let $L(\mathbf{w})$ be the energy (or loss), then

$$P(\mathbf{w}) = \frac{1}{Z} e^{-L(\mathbf{w})} \ .$$

- Mean-field approximates $P(\mathbf{w})$ with a fully-factorized distribution $U(\mathbf{w}) = \prod_{j=1}^{m} U_j(w_j)$.
- From the probabilistic interpretation of $\mathbf{u} \in \mathcal{S}$, $U_j(w_j = \lambda) = u_{j:\lambda}$, for each $j \in \{1 \dots m\}$.

Objective

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

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

Mean-Field Method

Let $L(\mathbf{w})$ be the energy (or loss), then

$$P(\mathbf{w}) = \frac{1}{Z} e^{-L(\mathbf{w})} \ .$$

- Mean-field approximates $P(\mathbf{w})$ with a fully-factorized distribution $U(\mathbf{w}) = \prod_{j=1}^{m} U_j(w_j)$.
- From the probabilistic interpretation of $\mathbf{u} \in \mathcal{S}$, $U_j(w_j = \lambda) = u_{j:\lambda}$, for each $j \in \{1 \ldots m\}$.

Objective

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

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

Mean-Field Method

Let $L(\mathbf{w})$ be the energy (or loss), then

$$P(\mathbf{w}) = \frac{1}{Z} e^{-L(\mathbf{w})} \ .$$

- Mean-field approximates $P(\mathbf{w})$ with a fully-factorized distribution $U(\mathbf{w}) = \prod_{j=1}^{m} U_j(w_j)$.
- From the probabilistic interpretation of $\mathbf{u} \in \mathcal{S}$, $U_j(w_j = \lambda) = u_{j:\lambda}$, for each $j \in \{1 \dots m\}$.

Objective

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

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

Mean-Field Method

Objective

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

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

Difficulty

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

Simple Idea

- Replace $L(\mathbf{w})$ with its first-order approximation $\hat{L}^k(\mathbf{w})$.

Mean-Field Method

Objective

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

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

Difficulty

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

Simple Idea

- Replace $L(\mathbf{w})$ with its first-order approximation $\hat{L}^k(\mathbf{w})$.

Mean-Field Method

Objective

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

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

Difficulty

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

Simple Idea

- Replace $L(\mathbf{w})$ with its first-order approximation $\hat{L}^k(\mathbf{w})$.

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \mathrm{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right)\;, \quad \text{PGD}$$

$$\mathbf{u}^{k+1} = \operatorname*{argmin}_{\mathbf{u}\in\mathcal{S}}\; \eta\,\mathbb{E}_{\mathbf{u}}\left[\hat{L}^k(\mathbf{w})\right] - \left\langle \mathbf{u}^k, \mathbf{u}\right\rangle_F - \frac{1}{\beta}H(\mathbf{u})\;, \quad \text{PMF}$$

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \text{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right)\;, \quad \text{PGD}$$

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

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \text{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right)\ , \quad \text{PGD}$$

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

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \text{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right)\;, \quad \text{PGD}$$

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

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \text{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right)\ , \quad \text{PGD}$$

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

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax update is an exact fixed point of the PMF objective

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration k,

$$\mathbf{u}^{k+1} = \text{softmax}\left(\beta\left(\mathbf{u}^k - \eta\,\mathbf{g}^k\right)\right) \ , \qquad \text{PGD}$$

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

where $\eta > 0$ and $\beta > 0$.

- $\hat{L}^k(\mathbf{w})$ 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 $\beta \to \infty$.

Softmax update is an exact fixed point of the PMF objective

Binary Connect [Courbariaux-2015] is a special case of PMF

Why MRF Perspective?

- Encoding parameter dependency:
  - Tree-structured entropy [Ravikumar-2008].
  - Second-order approximation of L.
- Connection to Bayesian deep learning methods.
- Uncertainty estimatation.

Why MRF Perspective?

- Encoding parameter dependency:
  - Tree-structured entropy [Ravikumar-2008].
  - Second-order approximation of L.
- Connection to Bayesian deep learning methods.
- Uncertainty estimatation.

Why MRF Perspective?

- Encoding parameter dependency:
  - Tree-structured entropy [Ravikumar-2008].
  - Second-order approximation of L.
- Connection to Bayesian deep learning methods.
- Uncertainty estimatation.

Results

Dataset	Architecture	REF (32 bit) Top-1/5 (%)	BC (1 bit) Top-1/5 (%)	PMF (1 bit) Top-1/5 (%)
MNIST	LeNet-300	98.55/99.93	98.05/99.93	98.24/99.97
	LeNet-5	99.39/99.98	99.30/99.98	99.44/100.0
CIFAR-10	VGG-16	93.01/99.38	86.40/98.43	90.51/99.56
	ResNet-18	94.64/99.78	91.60/99.74	92.55/99.80
CIFAR-100	VGG-16	70.33/88.58	43.70/73.43	61.52/85.83
	ResNet-18	73.85/92.49	69.93/90.75	71.85/91.88
TinyImageNet	ResNet-18	56.41/79.75	49.33/74.13	50.78/75.01

Classification accuracies on the test set for different methods.

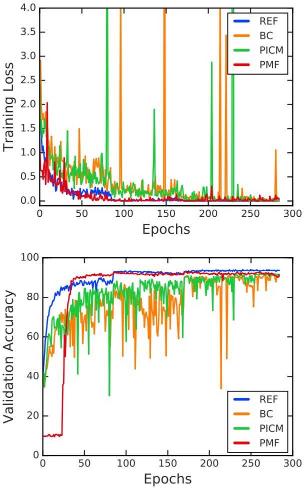
Results

Dataset	Architecture	REF (32 bit) Top-1/5 (%)	BC (1 bit) Top-1/5 (%)	PMF (1 bit) Top-1/5 (%)
MNIST	LeNet-300	98.55/99.93	98.05/99.93	98.24/99.97
	LeNet-5	99.39/99.98	99.30/99.98	99.44/100.0
CIFAR-10	VGG-16	93.01/99.38	86.40/98.43	90.51/99.56
	ResNet-18	94.64/99.78	91.60/99.74	92.55/99.80
CIFAR-100	VGG-16	70.33/88.58	43.70/73.43	61.52/85.83
	ResNet-18	73.85/92.49	69.93/90.75	71.85/91.88
TinyImageNet	ResNet-18	56.41/79.75	49.33/74.13	50.78/75.01

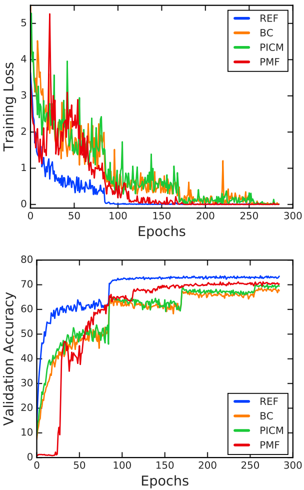
Classification accuracies on the test set for different methods.

PMF obtains accuracies very close to floating point counterparts

Results

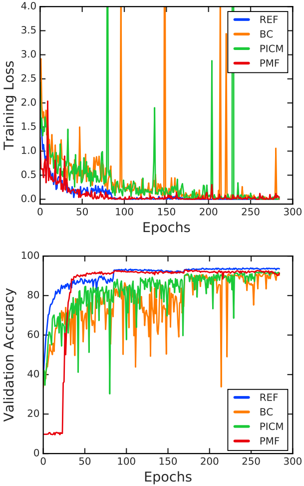


CIFAR-10, ResNet-18

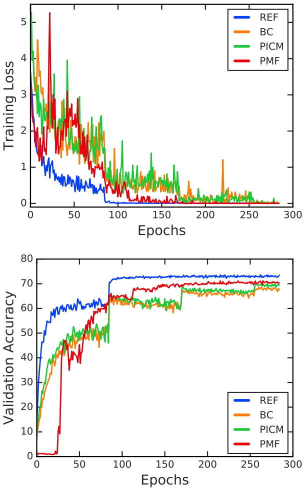


CIFAR-100, ResNet-18

Results



CIFAR-10, ResNet-18



CIFAR-100, ResNet-18

PMF is less noisy and closely resembles the reference network

Summary

- We have introduced a projected stochastic gradient descent algorithm to optimize the NN quantization problem.
- By showing our algorithm as a proximal version of mean-field, we have also provided an MRF optimization perspective to NN quantization.

Current Limitations

- Training time memory complexity is linear in $|\mathcal{Q}|$.
- No theoretical proof for the convergence to a vertex when $\beta \to \infty$.

Thank you!

First-order Approximation and Optimization

At iteration k,

$$\mathbf{u}^{k+1} = \underset{\mathbf{u}\in\mathcal{S}}{\operatorname{argmin}}\ \tilde{L}(\mathbf{u}^k) + \left\langle \mathbf{g}^k, \mathbf{u} - \mathbf{u}^k \right\rangle_F + \frac{1}{2\eta} \left\| \mathbf{u} - \mathbf{u}^k \right\|_F^2,$$

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

Proximal Mean-Field (PMF)

Algorithm

Require: $K, b, \{\eta^k\}, \rho > 1, \mathcal{D}, \tilde{L}$

Ensure: $\mathbf{w}^* \in \mathcal{Q}^m$

$\tilde{\mathbf{u}}^0 \in \mathbb{R}^{\mathrm{m}\times\mathrm{d}}, \quad \beta \leftarrow 1$ ▷ Initialization

for $k \leftarrow 0 \dots K$ **do**

  $\mathbf{u}^k \leftarrow \mathrm{softmax}(\beta\tilde{\mathbf{u}}^k)$ ▷ Projection

  $\mathcal{D}^b = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^b \sim \mathcal{D}$ ▷ Sample a mini-batch

  $\mathbf{g}_{\mathbf{u}}^k \leftarrow \nabla_{\mathbf{u}}\tilde{L}(\mathbf{u}; \mathcal{D}^b)\big|_{\mathbf{u}=\mathbf{u}^k}$ ▷ Gradient w.r.t. $\mathbf{u}$ at $\mathbf{u}^k$

  $\mathbf{g}_{\tilde{\mathbf{u}}}^k \leftarrow \mathbf{g}_{\mathbf{u}}^k \frac{\partial \mathbf{u}}{\partial \tilde{\mathbf{u}}}\big|_{\tilde{\mathbf{u}}=\tilde{\mathbf{u}}^k}$ ▷ Gradient w.r.t. $\tilde{\mathbf{u}}$ at $\mathbf{u}^k$

  $\tilde{\mathbf{u}}^{k+1} \leftarrow \tilde{\mathbf{u}}^k - \eta^k \mathbf{g}_{\tilde{\mathbf{u}}}^k$ ▷ Gradient descent on $\tilde{\mathbf{u}}$

  $\beta \leftarrow \rho\beta$ ▷ Increase β

$\mathbf{w}^* \leftarrow \mathrm{hardmax}(\tilde{\mathbf{u}}^K)\mathbf{q}$ ▷ Quantization