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 ⇒ 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

\[
\min_{w} L(w; D) := \frac{1}{n} \sum_{i=1}^{n} \ell(w; (x_i, y_i)), \quad w \in Q^m.
\]

\(D\) Dataset (\(D = \{x_i, y_i\}_{i=1}^{n}\))

\(\ell\) Loss function (I/O mapping + cross-entropy)

\(w\) Learnable parameters (\(m\))

\(Q\) Set of quantization levels (\(Q = \{-1, 1\}\))

Difficulties

▶ Exponentially many feasible points: \(|Q|^m\) with \(m \approx 10^6\).

▶ \(L\) is highly non-convex.
NN Quantization as Discrete Labelling

$$\min_{\mathbf{w}} L(\mathbf{w}; \mathcal{D}) := \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}; (x_i, y_i)) ,$$

$$\mathbf{w} \in Q^m .$$

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

**Difficulties**

- Exponentially many feasible points: $|Q|^m$ with $m \approx 10^6$.
- $L$ is highly non-convex.
Quantization as Discrete Labelling

$$\min_w L(w; D) := \frac{1}{n} \sum_{i=1}^{n} \ell(w; (x_i, y_i)) ,$$

$$w \in Q^m .$$

Difficulties

▶ Exponentially many feasible points: $|Q|^m$ 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

\[ u_{j: \lambda} = 1 \iff w_j = \lambda \in \mathcal{Q} \]

For \( w \in \mathcal{Q}^m \),
\[ w = uq, \]

s.t. \( u \in \mathcal{V} = \left\{ u \mid \sum_{\lambda} u_{j: \lambda} = 1, \quad \forall j \right\} \]

where \( q \) is the vector of quantization levels.
Lifting: Indicator variables

\[ u_j : \lambda = 1 \iff w_j = \lambda \in Q \]

For \( w \in Q^m \),

\[ w = uq \]

s.t. \( u \in V = \left\{ u \left| \sum_{\lambda} u_j : \lambda = 1, \; \forall j \right. \right. \left. \left. u_j : \lambda \in \{0, 1\}, \; \forall j, \lambda \right\} \),

where \( q \) is the vector of quantization levels.
Lifting and Relaxation

Lifting: Indicator variables

\[ u_j: \lambda = 1 \iff w_j = \lambda \in Q \]

For \( w \in Q^m \),

\[ w = uq \]

s.t. \( u \in \mathcal{V} = \left\{ u \mid \sum_\lambda u_j: \lambda = 1, \quad \forall j \right\} \),

where \( q \) is the vector of quantization levels.

\[ Q^m \iff \mathcal{V} \]
Lifting and Relaxation

Relaxation

\( u_{j:\lambda} = \{0, 1\} \implies u_{j:\lambda} = [0, 1] \)

For \( w \in \text{conv}(Q)^m \),
\[
w = uq,
\]

s.t. \( u \in S = \left\{ u \mid \sum_{\lambda} u_{j:\lambda} = 1, \quad \forall j \right\} \),

where \( q \) is the vector of quantization levels.

\( S \) decomposes over \( j \).
Lifting and Relaxation

Relaxation

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

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

\[ w = uq \]

s.t. \( u \in \mathcal{S} = \left\{ u \middle| \sum_{\lambda} u_{j: \lambda} = 1, \quad \forall j \right\} \),

where \( q \) is the vector of quantization levels.

- \( \mathcal{S} \) decomposes over \( j \).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{diagram.png}
\caption{Diagram illustrating the decomposition of \( \mathcal{S} \) over \( j \).}
\end{figure}
Lifting and Relaxation

Relaxation

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

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

$$w = uq,$$

s.t. $$u \in S = \left\{ u \mid \sum_{\lambda} u_{j:\lambda} = 1, \quad \forall j \atop u_{j:\lambda} \in [0, 1], \quad \forall j, \lambda \right\},$$

where $$q$$ is the vector of quantization levels.

- $$S$$ decomposes over $$j$$.

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

\[
\min_u \tilde{L}(u; \mathcal{D}) := L(\mathbf{u} \mathbf{q}; \mathcal{D}) ,
\]
\[u \in \mathcal{S} .\]

▶ Any local minimum in the \(u\)-space is also a local minimum in the relaxed \(w\)-space and vice versa.
Relaxed Optimization Problem

\[
\min_u \tilde{L}(u; D) := L(uq; D) ,
\]
\[u \in S .\]

▶ Any local minimum in the \(u\)-space is also a local minimum in the relaxed \(w\)-space and vice versa.
Relaxed Optimization Problem

\[
\min_u \tilde{L}(u; D) := L(uq; D), \\
u \in S.
\]

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

\[
\min_{u \in S} \tilde{L}(u; D) \equiv \min_{w \in \text{conv}(Q)^m} L(w; D).
\]
Projected (Stochastic) Gradient Descent (PGD)

At iteration $k$,

$$
\tilde{u}^{k+1} = u^k - \eta g^k, \quad \text{SGD}
$$

$$
u^{k+1} = P_S(\tilde{u}^{k+1}),
$$

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

- Any off-the-shelf SGD algorithm 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 $k$,

\[ \tilde{u}^{k+1} = u^k - \eta g^k, \quad \text{SGD} \]
\[ u^{k+1} = P_S \left( \tilde{u}^{k+1} \right), \]

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

▶ Any off-the-shelf SGD algorithm 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 $k$,

$$
\tilde{u}^{k+1} = u^k - \eta g^k, \quad \text{SGD}
$$

$$
u^{k+1} = \mathcal{P}_S \left( \tilde{u}^{k+1} \right),
$$

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

▶ Any off-the-shelf SGD algorithm can be used.
▶ For softmax projection, $\text{PGD} \equiv \text{Proximal Mean-Field}$.

▶ Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].
Projected (Stochastic) Gradient Descent (PGD)

At iteration $k$,

$$\tilde{u}^{k+1} = u^k - \eta g^k,$$  \hspace{1cm} \text{SGD}

$$u^{k+1} = P_S \left( \tilde{u}^{k+1} \right),$$

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

▶ Any off-the-shelf SGD algorithm can be used.
▶ For softmax projection, $\text{PGD} \equiv \text{Proximal Mean-Field}$.
▶ Projection free algorithms may also be employed [Lacoste-2012, Ajanthan-2017].
Softmax Projection and Exploration

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

$$u^k_j = \text{softmax}(\beta \tilde{u}^k_j) ,$$

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

where $\beta > 0$.

Ultimate objective

- Preserves relative order of $u^k_{j: \lambda}$.
- Differentiable.

A quantized solution $\Rightarrow u \in \mathcal{V}$ attained when $\beta \to \infty$. 
Softmax Projection and Exploration

For each \( j \in \{1 \ldots m\} \),

\[
    u^k_j = \text{softmax}(\beta \tilde{u}^k_j),
\]

\[
    u^k_{j: \lambda} = \frac{e^{\beta(\tilde{u}^k_{j: \lambda})}}{\sum_{\mu \in Q} e^{\beta(\tilde{u}^k_{j: \mu})}} \forall \lambda \in Q,
\]

where \( \beta > 0 \).

Ultimate objective

- A quantized solution \( u \in V \) attained when \( \beta \to \infty \).
Softmax Projection and Exploration

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

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

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

where $\beta > 0$.

Ultimate objective

- A quantized solution $\Rightarrow$ $u \in V$ attained when $\beta \to \infty$. 
Softmax Projection and Exploration

For each \( j \in \{1 \ldots m\} \),

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

\[
u_{j:\lambda}^k = \frac{e^{\beta(\tilde{\mathbf{u}}_{j:\lambda}^k)}}{\sum_{\mu \in Q} e^{\beta(\tilde{\mathbf{u}}_{j:\mu}^k)}} \quad \forall \lambda \in Q,
\]

where \( \beta > 0 \).

Ultimate objective

- A quantized solution \( \Rightarrow \quad \mathbf{u} \in \mathcal{V} \) 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 \( u \in S \), \( U_j(w_j = \lambda) = u_j;\lambda \), for each \( j \in \{1\ldots m\} \).

Objective

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

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

Let $L(w)$ be the energy (or loss), then

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

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

Objective

$$\arg\min_{u \in S} \text{KL}(u\|P) = \arg\min_{u \in S} E_u[L(w)] - H(u),$$

where $E_u[\cdot]$ is the expectation over $u$ and $H(u)$ is the entropy.
Mean-Field Method

Let $L(w)$ be the energy (or loss), then

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

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

Objective

$$\arg\min_{u \in S} KL(u\|P) = \arg\min_{u \in S} \mathbb{E}_u [L(w)] - H(u),$$

where $\mathbb{E}_u [\cdot]$ is the expectation over $u$ and $H(u)$ is the entropy.
Mean-Field Method

Let $L(w)$ be the energy (or loss), then

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

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

Objective

$$\arg\min_{u \in S} KL(u\|P) = \arg\min_{u \in S} \mathbb{E}_u[L(w)] - H(u),$$

where $\mathbb{E}_u[\cdot]$ is the expectation over $u$ and $H(u)$ is the entropy.
Mean-Field Method

Objective

\[
\arg\min_{u \in S} \ KL(u \| P) = \arg\min_{u \in S} \ \mathbb{E}_u[L(w)] - H(u),
\]

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

Difficulty

- For a neural network \( L(w) \) has no explicit factorization.

Simple Idea

- Replace \( L(w) \) with its first-order approximation \( \hat{L}^k(w) \).
Mean-Field Method

Objective

\[ \arg\min_{u \in S} \text{KL}(u || P) = \arg\min_{u \in S} \mathbb{E}_u[L(w)] - H(u), \]

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

Difficulty

- For a neural network \( L(w) \) has no explicit factorization.

Simple Idea

- Replace \( L(w) \) with its first-order approximation \( \hat{L}^k(w) \).
Mean-Field Method

Objective

\[ \arg\min_{u \in S} \text{KL}(u \| P) = \arg\min_{u \in S} \mathbb{E}_u[L(w)] - H(u), \]

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

Difficulty

- For a neural network \( L(w) \) has no explicit factorization.

Simple Idea

- Replace \( L(w) \) with its first-order approximation \( \hat{L}^k(w) \).
Softmax based PGD as Proximal Mean-Field (PMF)

At iteration $k$,

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

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

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

- $\hat{L}^k (\mathbf{w})$ is the first-order Taylor approximation of $L$ at $\mathbf{w}^k = \mathbf{u}^k q$.

- Negative of cosine similarity $\Rightarrow$ proximal term.

- Entropy term vanishes when $\beta \to \infty$. 

Softmax based PGD as Proximal Mean-Field (PMF)

At iteration $k$, 

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

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

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

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

At iteration $k$, 

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

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

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

- $\hat{L}^k(w)$ is the first-order Taylor approximation of $L$ at $w^k = u^k q$.

- Negative of cosine similarity $\Rightarrow$ proximal term.

- Entropy term vanishes when $\beta \to \infty$. 
Softmax based PGD as Proximal Mean-Field (PMF)

At iteration $k$,

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

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

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

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

At iteration $k$,

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

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

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

- $\hat{L}^k(w)$ is the first-order Taylor approximation of $L$ at $w^k = u^k q$.
- 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 $k$,

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

$$
\mathbf{u}^{k+1} = \arg\min_{\mathbf{u} \in \mathcal{S}} \eta \mathbb{E}_{\mathbf{u}} \left[ \hat{L}^k (\mathbf{w}) \right] - \left< \mathbf{u}^k, \mathbf{u} \right>_F - \frac{1}{\beta} H(\mathbf{u}), \quad \text{PMF}
$$

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

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 estimation.
Why MRF Perspective?

- Encoding parameter dependency:
  - Tree-structured entropy [Ravikumar-2008].
  - Second-order approximation of $L$.
- Connection to Bayesian deep learning methods.
- Uncertainty estimation.
Why MRF Perspective?

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

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

Classification accuracies on the test set for different methods.
## Results

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

**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

pmf is less noisy and closely resembles the reference network.
PMF is less noisy and closely resembles the reference network.
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 $|Q|$.
- No theoretical proof for the convergence to a vertex when $\beta \rightarrow \infty$. 
Thank you!
At iteration $k$,

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

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

First-order Approximation and Optimization
Proximal Mean-Field (PMF)

Algorithm

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

Ensure: \( w^* \in Q^m \)

1: \( \tilde{u}^0 \in \mathbb{R}^{m \times d}, \quad \beta \leftarrow 1 \) \hspace{1cm} \( \triangleright \) Initialization
2: \( \text{for } k \leftarrow 0 \ldots K \text{ do} \)
3: \( \tilde{u}^k \leftarrow \text{softmax}(\beta \tilde{u}^k) \) \hspace{1cm} \( \triangleright \) Projection
4: \( \mathcal{D}^b = \{(x_i, y_i)\}_{i=1}^b \sim \mathcal{D} \)
5: \( g_u^k \leftarrow \nabla_u \tilde{L}(u; \mathcal{D}^b)|_{u=\tilde{u}^k} \) \hspace{1cm} \( \triangleright \) Sample a mini-batch
6: \( g_{\tilde{u}}^k \leftarrow g_u^k \frac{\partial u}{\partial \tilde{u}}|_{\tilde{u}=\tilde{u}^k} \) \hspace{1cm} \( \triangleright \) Gradient w.r.t. \( u \) at \( u^k \)
7: \( \tilde{u}^{k+1} \leftarrow \tilde{u}^k - \eta^k g_{\tilde{u}}^k \) \hspace{1cm} \( \triangleright \) Gradient w.r.t. \( \tilde{u} \) at \( u^k \)
8: \( \beta \leftarrow \rho \beta \) \hspace{1cm} \( \triangleright \) Increase \( \beta \)
9: \( w^* \leftarrow \text{hardmax}(\tilde{u}^K q) \) \hspace{1cm} \( \triangleright \) Quantization