LAYER-WISE DEEP NEURAL NETWORK PRUNING VIA ITERATIVELY REWEIGHTED OPTIMIZATION

Tao Jiang, Xiangyu Yang, Yuanming Shi, and Hao Wang

School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China.
E-mail: {jiangtao1,yangxy3,shiym,wanghao1}@shanghaitech.edu.cn

ABSTRACT

The huge number of parameters of deep neural network makes it difficult to deploy on embedded devices with limited hardware, computation, storage and energy resources. In this paper, we shall propose a log-sum minimization approach to prune a trained network layer by layer thereby improving the network compression ratio. Specifically, this is achieved by enhancing sparsity for network parameters such that the output of the network after pruning is consistent with the original one. We further present an iteratively reweighted algorithm to solve the nonconvex and nonsmooth log-sum minimization problem with general convex constraints. Furthermore, we show the existence of the cluster points for the iterates and the global convergence of the proposed iteratively reweighted algorithm. Numerical experiments demonstrate that the proposed approach is able to significantly prune the trained neural network while preserving the prediction accuracy.

***Index Terms*—** Deep learning, network pruning, compressive sensing, sparse optimization.

1. INTRODUCTION

Deep neural network (DNN) has shown enormous success in a wide range of applications, from speech processing [1] and image classification [2] to reinforcement learning [3], which become the fundamental services for embedded devices (e.g., IoT devices) [4, 5]. Therefore, the adoption of DNN to the embedded devices becomes a trend especially for the real-time and accurate prediction services [4, 5]. However, the number of parameters of a DNN is typically extremely large (e.g., more than 130M parameters in VGG-16 net [6]) in order to achieve high prediction accuracy. This makes it impractical for most models be directly deployed on embedded devices, which are usually constrained on hardware resources and energy budget. So far a prominent approach is transmitting data to the cloud where large DNN models are deployed, and then returning the prediction results to embedded devices [7]. However, such a solution suffers significant side effects including privacy dangers and dynamic network quality (e.g., bandwidth, latency). As a result, building small yet efficient DNN models becomes critical for direct deployment of DNN models on resource constrained embedded devices.

To produce compact DNN models that can be directly deployed on embedded devices, many approaches have been proposed such as low-rank approximation of network parameters [8], network parameter quantization [9] and network pruning [10, 11, 12]. In particular, network pruning has received notable attention among these methods due to its competitive performance and compatibility. Given a large trained DNN model, the main idea behind network pruning is to trim the connections between neurons according to a certain criterion, thereby reducing the number of nonzero parameters. However, most of the existing pruning criteria are designed heuristically, which give no performance guarantees for a pruned DNN. Instead of considering a whole network, [13] propose to prune the network layer-by-layer via minimizing reconstruction error of each layer. This layer-wise pruning method offers the possibility to provide performance guarantees, i.e., the overall performance drops after pruning can be bounded by the sum of reconstruction error for each layer.

In order to further prune (or sparsify) a DNN, in this paper, we propose to adopt log-sum function to enhance sparsity in network parameters instead of using ℓ_1-norm as adopted in [13]. This is motivated by the fact that the log-sum function is a tighter approximation to ℓ_0-norm compared with ℓ_1-norm. However, the resulting optimization problem is computationally intractable due to the nonconvex and nonsmooth nature of the log-sum function. To address this issue, an iteratively reweighted algorithm [14] is proposed to solve it, which is achieved by solving a sequence of convex subproblems. However, the convergence analysis of an iteratively reweighted algorithm for the original nonconvex and nonsmooth problem is difficult to track [15, 16]. The global convergence of the iteratively reweighted algorithm for unconstrained ℓ_p regularization problem is derived in [17]. For the ℓ_2-ℓ_p minimization problems [15] with linear constrains, the global convergence of the iteratively reweighted algorithm is established in [18]. A recent study [19] has shown the global convergence results for a class of unconstrained nonconvex regularization problems in computer vision based on limiting-subgradient. However, their results are inapplicable in our problem due to the complicated convex constraints for network pruning. To address this issue, in this paper, we prove the global convergence of the presented iteratively reweighted algorithm based on the Fréchet subdifferential [20]. This is achieved by proving that the sequence generated by the iteratively reweighted algorithm must have cluster points, and every cluster point satisfies the first-order optimality condition of the constrained log-sum minimization problem. Numerical experiments demonstrate that the proposed approach achieves high compression ratio while preserving comparable prediction accuracy.

This work was supported in part by the National Nature Science Foundation of China under Grant 61601290 and in part by the Shanghai Sailing Program under Grant 16YF1407700.

2. PROBLEM STATEMENT

Consider a feedforward fully-connected neural network with L layers. n training samples $\boldsymbol{x}_i \in \mathbb{R}^d$ are available. We denote by $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$ as the input training data matrix. The final output of the network is denoted by $\boldsymbol{X}_L \in \mathbb{R}^{d_L\times n}$. We focus on Rectified Linear Units (ReLU) as the activation function. Therefore, the input-output relationship of the ℓ-th layer is given by

$$\boldsymbol{X}_\ell = \max(\boldsymbol{W}_\ell^T \boldsymbol{X}_{\ell-1}, 0),\ \ell = 1, \cdots, L, \tag{1}$$

where $\boldsymbol{X}_0 = \boldsymbol{X}$, $d_0 = d$ and $\boldsymbol{W}_\ell \in \mathbb{R}^{d_{\ell-1}\times d_\ell}$ is the parameter matrix applied to the output of previous layer $\boldsymbol{X}_{\ell-1}$; the activation function $\max(\cdot, 0)$ is applied in an element-wise manner. In model (1), the bias is assumed to be embedded in the parameter matrix $\boldsymbol{W}_\ell$ for notational simplicity.

Network compression including low-rank approximation of network [8], network parameter quantization [9] and network pruning [10, 11, 12] is one of the promising approaches to adopt deep neural network for embedded devices (e.g., IoT devices) with limited hardware resources and tight energy budgets [4, 5]. The network pruning method received notable attention due to their competitive performance and compatibility. Furthermore, the work in [13] proposed a layer-wise network pruning method, namely, Net-trim, which removes connections at each layer by sparse optimization method, which also pervades numerous applications in wireless communication [21] . They show that discrepancy between the pruned network and the original one can be bounded by the sum of the reconstruction error of each layer. In this paper, we choose the compression ratio (CR) and prediction accuracy as our performance metric, where CR is defined as one minus the ratio of the number of parameters of the pruned network to that of the original one.

To further improve the CR, we propose a nonconvex and nonsmooth optimization approach, followed by an iteratively reweighted algorithm with global convergence guarantees to prune a trained neural network layer by layer while maintaining the prediction accuracy.

2.1. Layer-wise Pruning of Network

We consider the ℓ-th layer of a neural network. A natural approach to prune the network is using ℓ_1-norm minimization to promote sparsity while preserving the network's output approximately. This approach takes the form

$$\begin{aligned} \underset{\boldsymbol{W}\in\mathbb{R}^{d_{\ell-1}\times d_\ell}}{\text{minimize}} \quad & \|\boldsymbol{W}\|_1 \\ \text{subject to} \quad & \|\max(\boldsymbol{W}^T\boldsymbol{X}_{\ell-1}, 0) - \boldsymbol{X}_\ell\|_F \le \epsilon, \end{aligned} \tag{2}$$

where $\|\boldsymbol{W}\|_1 = \sum_{i,j}|w_{i,j}|$ with $w_{i,j}$ as the i-th row and the j-th column entry of matrix $\boldsymbol{W}$ and parameter $\epsilon > 0$ controls the reconstruction error of this layer.

Furthermore, the nonconvex constraint in (2) can be approximated by a convex set based on the fact that the entries of $\boldsymbol{X}_\ell$ are either zero or strictly positive [13]. It turns out that we can obtain a pruned network via solving the following convex proxy to (2):

$$\begin{aligned} \underset{\boldsymbol{W}\in\mathbb{R}^{d_{\ell-1}\times d_\ell}}{\text{minimize}} \quad & \|\boldsymbol{W}\|_1 \\ \text{subject to} \quad & \|(\boldsymbol{W}^T\boldsymbol{X}_{\ell-1} - \boldsymbol{X}_\ell)_\Omega\|_F \le \epsilon, \\ & (\boldsymbol{W}^T\boldsymbol{X}_{\ell-1})_{\Omega^c} \le \boldsymbol{0}, \end{aligned} \tag{3}$$

where $\Omega = \operatorname{supp}\boldsymbol{X}_\ell = \{(i,j) : [\boldsymbol{X}_\ell]_{i,j} > 0\}$.

2.2. Proposed Nonconvex Pruning Approach

To further enhance sparsity for the network parameters, instead of applying the convex ℓ_1-minimization approach, we propose to use log-sum sparsity inducing function

$$f_p(\boldsymbol{W}) = \sum_{i,j}\log(p|w_{ij}| + 1), \quad \text{with} \quad p > 0, \tag{4}$$

to seek a sparser solution in problem (3). This is motivated by the fact that function $f_p(\boldsymbol{W})$ tends to $\|\boldsymbol{W}\|_0$ as $p \to \infty$, where $\|\boldsymbol{W}\|_0$ is the ℓ_0-norm that counts the number of non-zero entries of $\boldsymbol{W}$. Intuitively, the log-sum function has p slop at the origin as $p \to \infty$ while the ℓ_1-norm only has unit slop. Therefore, $f_p(\boldsymbol{X})$ places relatively larger penalty on small nonzero network parameters, thereby encouraging them to be set to zero more strongly. Let $\mathcal{W}$ denote the convex feasible set in problem (3). Adopting the log-sum function results in the following optimization problem:

$$\underset{\boldsymbol{W}\in\mathbb{R}^{d_{\ell-1}\times d_\ell}}{\text{minimize}} \quad f_p(\boldsymbol{W}), \quad \text{subject to} \quad \boldsymbol{W} \in \mathcal{W}. \tag{5}$$

However, the nonconvex and nonsmooth nature of log-sum function makes (5) computationally intractable. In the following section, we present an iteratively reweighted algorithm with global convergence guarantees for solving problem (5).

3. ITERATIVELY REWEIGHTED ALGORITHM

In this section, we shall present an iteratively reweighted algorithm to solve the constrained nonconvex and nonsmooth log-sum minimization problem (5), which can be written as

$$\underset{\boldsymbol{W}\in\mathbb{R}^{d_{\ell-1}\times d_\ell}}{\text{minimize}} \quad J(\boldsymbol{W}) := f_p(\boldsymbol{W}) + \delta_{\mathcal{W}}(\boldsymbol{W}). \tag{6}$$

Here, $\delta_{\mathcal{W}}(\boldsymbol{W})$ is the characteristic function defined as

$$\delta_{\mathcal{W}}(\boldsymbol{W}) = \begin{cases} 0 & \text{if } \boldsymbol{W} \in \mathcal{W}, \\ +\infty & \text{otherwise.} \end{cases}$$

Notice that the log-sum function can be written as a composite function of h_p and the absolute value function

$$f_p(\boldsymbol{W}) = \sum_{i,j} h_p(z_{ij}),$$

with $z_{ij} = |w_{ij}|$ and $h_p(z) := \log(pz + 1)$. The major difficulty of minimizing f_p comes from the nonconvex and nonsmooth nature of function h_p with respect to w_{ij}. To circumvent this, at a given point $\hat{\boldsymbol{W}}$, we approximate f_p by replacing h_p using its linearization at $|\hat{w}_{ij}|$, i.e.,

$$h_p(z_{ij}) \approx h_p'(\hat{z}_{ij})(z_{ij} - \hat{z}_{ij}) \quad \text{with } \hat{z}_{ij} = |\hat{w}_{ij}|.$$

Therefore, we can obtain the next iterate $\tilde{\boldsymbol{W}}$ via solving the following convex optimization problem, which is a convex and smooth local approximation model at each iterate $\hat{\boldsymbol{W}}$:

$$\underset{\boldsymbol{W}\in\mathbb{R}^{d_{\ell-1}\times d_\ell}}{\text{minimize}} \quad G_{\hat{\boldsymbol{W}}}(\boldsymbol{W}) := \sum_{i,j}\hat{a}_{ij}|w_{ij}| + \delta_{\mathcal{W}}(\boldsymbol{W}), \tag{7}$$

where the weights are given by $\hat{a}_{ij} = h_p'(\hat{z}_{ij}) = \frac{p}{p\hat{z}_{ij}+1} = \frac{p}{p|\hat{w}_{ij}|+1}$. The smaller parameter $\hat{w}_{ij}$ is, the larger weight $\hat{a}_{ij}$ will

be, which shall place relatively larger penalty on the small but nonzero network parameters $\hat{w}_{ij}$ and enforce them to be zero more aggressively.

The presented iteratively reweighted algorithm updates the weight matrix $\boldsymbol{A}^k := [a_{ij}^k] \in \mathbb{R}^{d_{\ell-1} \times d_\ell}$ to obtain the new iterate $\boldsymbol{W}^{k+1}$. As suggested in [14], we solve the unweighted problem (3) to obtain the initial points $\boldsymbol{W}^0$.

Algorithm 1: Iteratively Reweighted Algorithm for Solving the Network Pruning Problem (5)

Input : Initial points $\boldsymbol{W}^0$

for $k = 1, 2, \cdots$ **do**

  Compute new iterate:
  $\boldsymbol{W}^k \in \underset{\boldsymbol{W} \in \mathbb{R}^{d_{\ell-1} \times d_\ell}}{\arg\min} G_{\boldsymbol{W}^{k-1}}(\boldsymbol{W})$.

  Updates weigths: $a_{ij}^k = \frac{p}{p|w_{ij}^k|+1}$, $\forall i, j \in [d_1] \times [d_2]$.

end

However, the convergence analysis of the iteratively reweighted algorithm for the nonconvex optimization problem is difficult to establish. The work in [17] showed the global convergence of the iteratively reweighted algorithm for the ℓ_p regularization problem but no constraints are involved, and [18] showed the global convergence for the ℓ_2-ℓ_p minimization problem with linear constrains. A recent study [19] showed the global convergence results for a class of unconstrained nonconvex regularization problems in computer vision based on limiting-subgradient. However, their results are normally inapplicable for our log-sum minimization problem with complicated constraints.

Before proceeding to the global convergence analysis, we summarize the benefits of choosing log-sum function among various types of sparsity inducing functions, which are mainly based on the boundedness of h_p' and the coercivity of h_p.

- Note that $a_{ij} = h_p'(z_{ij})$ is bounded above by p as $z_{ij} \to 0$, which can prevent the subproblems $G(\cdot)$ from becoming overwhelmingly ill-conditioned and intractable.
- Since $h_p'(0+)$ is bounded above, it is easy to characterize the generalized subdifferentials of $h_p \circ |\cdot|$ at 0, where $h_p \circ |\cdot|$ denotes the function h_p composited with $|\cdot|$. As a result, it is straightforward to characterize the optimality condition of (6). This also makes the convergence analysis simple.
- The coercivity of $J(\cdot)$ guarantees the level set of $J(\cdot)$ is always bounded. As shown in the analysis, $J(\cdot)$ is steadily decreasing from any feasible starting point over the iterates, so that the iterates always possess cluster points—a simplification for the analysis as well as the selection of the initial points.

4. GLOBAL CONVERGENCE

In this section, we show the global convergence of our proposed algorithms. Specifically, we shall provide the first-order optimality condition for the problem (5) and show that the sequence generated by our proposed algorithm must have cluster points, and every cluster point of the iterates satisfies the first-order optimal condition for problem (6).

Definition 1 (Fréchet subdifferential [20])**.** *Let $\mathcal{X}$ be a real Banach space and $\mathcal{X}^*$ denotes the corresponding topological duals and f be a function from $\mathcal{X}$ into an extended real line $\bar{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$ and finite at $\boldsymbol{x}$. The Fréchet subdifferential of f at $\boldsymbol{x}$, denoted as $\partial_F f(\boldsymbol{x})$, is a set defined by*

$$\partial_F f(\boldsymbol{x}) = \left\{ \boldsymbol{z} \in \mathcal{X}^* : \liminf_{\boldsymbol{u} \to \boldsymbol{x}} \frac{f(\boldsymbol{u}) - f(\boldsymbol{x}) - \langle \boldsymbol{z}, \boldsymbol{u} - \boldsymbol{x} \rangle}{\|\boldsymbol{u} - \boldsymbol{x}\|} \geq 0 \right\}.$$

Its elements are referred to as Fréchet subgradients.

First of all, we derive conditions to characterize the first-order necessary optimality condition of (6) in the following theorem.

Theorem 1 (Fermat's rule [22])**.** *Let $\boldsymbol{W} \in \mathcal{W}$ be a local minimum of* (6)*. Then the following condition is satisfied:*

$$0 \in \partial_F J(\boldsymbol{W}) = \partial_F f_p(\boldsymbol{W}) + N(\boldsymbol{W}|\mathcal{W}), \tag{8}$$

where $N(\boldsymbol{W}|\mathcal{W})$ is the normal cone [23] to $\mathcal{W}$ at a point $\boldsymbol{W}$.

We now investigate $\partial_F f_p(\boldsymbol{W})$ in Lemma 1, which can simplify our subsequent analysis. We omit the detailed proofs of Lemma 1, 2, 3, 4 due to space limitation.

Lemma 1. *It holds true that $[\gamma_{ij}\zeta_{ij}] \in \partial_F f_p(\boldsymbol{W})$ for $\gamma_{ij} = h_p'(z_{ij})$, where $\zeta_{ij} \in \partial z_{ij}(w_{ij})$. In particular, $\partial_F h_p \circ z_{ij}(0) = [-p, p]$.*

The following Lemma 2 shows the existence of the solution to (7) and characterizes the first-order optimality of (6).

Lemma 2. *The optimal solution of subproblem* (7) *is always nonempty for any $\hat{\boldsymbol{W}} \in \mathcal{W}$. Furthermore, $\hat{\boldsymbol{W}}$ also satisfies the first-order optimality condition of* (6) *if and only if*

$$\hat{\boldsymbol{W}} \in \arg\min_{\boldsymbol{W}} G_{\hat{\boldsymbol{W}}}(\boldsymbol{W}).$$

The next Lemma 3 indicates that the optimal solution of the subproblem causes a decrease in $J(\boldsymbol{W})$.

Lemma 3. *Let $\hat{\boldsymbol{W}} \in \mathcal{W}$ and $\hat{a}_{ij} = a_{ij}(\hat{w}_{ij})$ for $i, j \in [d_1] \times [d_2]$. Suppose that $\tilde{\boldsymbol{W}} \in \arg\min_{\boldsymbol{W}} G_{\hat{\boldsymbol{W}}}(\boldsymbol{W})$. It holds true that*

$$J(\tilde{\boldsymbol{W}}) - J(\hat{\boldsymbol{W}}) \leq G_{\hat{\boldsymbol{W}}}(\tilde{\boldsymbol{W}}) - G_{\hat{\boldsymbol{W}}}(\hat{\boldsymbol{W}}) \leq 0.$$

At the k-th iterate, define the model reduction caused by the new iterate $\boldsymbol{W}^{k+1}$

$$\Delta G_{\boldsymbol{W}^k}(\boldsymbol{W}^{k+1}) = G_{\boldsymbol{W}^k}(\boldsymbol{W}^k) - G_{\boldsymbol{W}^k}(\boldsymbol{W}^{k+1}).$$

Notice that $\Delta G_{\boldsymbol{W}^k}(\boldsymbol{W}^{k+1}) \geq 0$. We show this model reduction diminishes in the following Lemma 4.

Lemma 4. *Suppose $\{\boldsymbol{W}^k\}$ is generated by Algorithm 1 with $\boldsymbol{W}^1 \in \mathcal{W}$. It holds true that $\{J(\boldsymbol{W}^k)\}$ is monotonically decreasing and $\lim_{k \to \infty} \Delta G_{\boldsymbol{W}^k}(\boldsymbol{W}^{k+1}) = 0$.*

We now provide our main result on the global convergence in the following theorem.

Theorem 2. *Suppose sequence $\{\boldsymbol{W}^k\}_{k=1}^\infty$ is generated by Algorithm 1 with initial point $\boldsymbol{W}^0 \in \mathcal{W}$. It holds true that $\{\boldsymbol{W}^k\}$ must have cluster points and every cluster point of $\{\boldsymbol{W}^k\}$ satisfies the first-order optimality condition for* (5)*.*

Proof. Assume by contradiction that $\{\boldsymbol{W}^k\}$ is unbounded. There must exist a subsequence $\{\boldsymbol{W}^k\}_{k\in\mathcal{S}}$, $\mathcal{S}\subset\mathbb{N}$ such that $\|\boldsymbol{W}^k\|\xrightarrow{k\in\mathcal{S}}+\infty$. It follows that $J(\boldsymbol{W}^k)\xrightarrow{k\in\mathcal{S}}+\infty$ due to the coercivity of J, contradicting Lemma 4. Hence $\{\boldsymbol{W}^k\}$ must be bounded. By Bolzano-Weierstrass theorem, we show that the existence of the cluster points.

Now suppose $\boldsymbol{W}^*$ is a cluster point of $\{\boldsymbol{W}^k\}$. According to Lemma 2, it suffices to show that $\boldsymbol{W}^*\in\arg\min_{\boldsymbol{W}} G_{\boldsymbol{W}^*}(\boldsymbol{W})$. We assume by contradiction that $\boldsymbol{W}^*$ is not optimal and there exists $\bar{\boldsymbol{W}}$ such that $\epsilon:=G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)-G_{\boldsymbol{W}^*}(\bar{\boldsymbol{W}})>0$. Consider a subsequence $\mathcal{S}\subset\mathbb{N}$ with $\{\boldsymbol{W}^k\}_{\mathcal{S}}\to\boldsymbol{W}^*$. By Lemma 4, there exists $k_1>0$, such that for all $k>k_1$

$$G_{\boldsymbol{W}^k}(\boldsymbol{W}^k)-G_{\boldsymbol{W}^k}(\boldsymbol{W}^{k+1})\le\frac{\epsilon}{4}. \tag{9}$$

To derive a contradiction, notice that $w_{ij}^k\xrightarrow{\mathcal{S}}w_{ij}^*$ and $a_{ij}^k\xrightarrow{\mathcal{S}}a_{ij}^*$. There exists k_2 such that for all $k>k_2$, $\sum_{ij}(a_{ij}^*-a_{ij}^k)|\bar{w}_{ij}|>-\frac{\epsilon}{12}$, and $\sum_{ij}(a_{ij}^k|w_{ij}^k|-a_{ij}^*|w_{ij}^*|)>-\frac{\epsilon}{12}$. Therefore, for all $k>k_2$

$$\begin{aligned}
&G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)-G_{\boldsymbol{W}^k}(\bar{\boldsymbol{W}})\\
&=\sum_{ij}a_{ij}^*|w_{ij}^*|-\sum_{ij}(a_{ij}^*-(a_{ij}^*-a_{ij}^k))|\bar{w}_{ij}|\\
&=[G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)-G_{\boldsymbol{W}^*}(\bar{\boldsymbol{W}})]+\sum_{ij}(a_{ij}^*-a_{ij}^k)|\bar{w}_{ij}|\\
&\ge[G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)-G_{\boldsymbol{W}^*}(\bar{\boldsymbol{W}})]-\frac{\epsilon}{12}\\
&=\epsilon-\frac{\epsilon}{12}=\frac{11\epsilon}{12},
\end{aligned}$$

and that

$$G_{\boldsymbol{W}^k}(\boldsymbol{W}^k)-G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)=\sum_{ij}a_{ij}^k|w_{ij}^k|-\sum_{ij}a_{ij}^*|w_{ij}^*|>-\frac{\epsilon}{12}$$

Hence, for all $k>\max(k_1,k_2)$, it holds that

$$\begin{aligned}
&G_{\boldsymbol{W}^k}(\boldsymbol{W}^k)-G_{\boldsymbol{W}^k}(\bar{\boldsymbol{W}})\\
&=G_{\boldsymbol{W}^k}(\boldsymbol{W}^k)-G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)+G_{\boldsymbol{W}^*}(\boldsymbol{W}^*)-G_{\boldsymbol{W}^k}(\bar{\boldsymbol{W}})\\
&\ge\frac{11\epsilon}{12}-\frac{\epsilon}{12}=\frac{5\epsilon}{6},
\end{aligned}$$

contradicting with (9). Therefore, $\boldsymbol{W}^*\in\arg\min_{\boldsymbol{W}} G_{\boldsymbol{W}^*}(\boldsymbol{W})$. By Lemma 2, $\boldsymbol{W}^*$ satisfies the first-order optimality for (5). □

5. SIMULATION RESULTS

In this section, we demonstrate the effectiveness of our proposed method in terms of CR, prediction accuracy with fine-tuning (FT) and prediction accuracy without fine-tuning on various architectures of deep neural networks. The fine tuning step uses the compressed network parameters as an initialization for network retraining, which often improves the prediction accuracy without changing the sparsity of the network.

We apply our compression approaches to the problem of classifying hand-written digits of the MNIST database, which has a training set of $60,000$ examples, and a prediction set of $10,000$ examples. We consider a fully connected neural network (namely NN3) of size $784\times300\times1000\times300\times10$ following the setting in [13], and train it with 60,000 samples in MNIST. We also compare our compression approach with Net-trim for a convolutional network on CIFAR10 database, which contains $60,000$ samples of size of 32×32 color images in 10 different classes. The convolutional network contains two convolutional layers composed of 64 filters of size of $5\times5\times3$ for the first layer and 64 filters of size of $5\times5\times64$ for the second layer, both followed by a max pooling layer, and three fully connected layers $2304\times384\times192\times10$.

Table 1. Comparison with Net-trim [13] for NN3 network on MNIST dataset. (Initial model prediction accuracy = 98.62%)

	CR(%)		prediction accuracy without FT(%)		prediction accuracy with FT(%)	
	Net-trim	Proposed	Net-trim	Proposed	Net-trim	Proposed
$\epsilon=0.01$	13.71	**25.38**	98.63	**98.65**	**98.65**	**98.65**
$\epsilon=0.04$	31.19	**49.94**	98.63	**98.66**	98.64	**98.66**
$\epsilon=0.08$	45.23	**64.24**	98.35	**98.42**	98.45	**98.53**
$\epsilon=0.10$	49.84	**68.55**	98.25	**98.34**	98.45	**98.51**
$\epsilon=0.20$	62.72	**79.92**	**97.11**	97.04	**98.05**	97.93
$\epsilon=0.30$	69.52	**85.25**	**95.93**	94.42	**97.70**	97.46

Table 2. Comparison with Net-trim [13] for CNN network on CIFAR10 dataset. (Initial model prediction accuracy = 77.44%)

	CR(%)		prediction accuracy without FT(%)		prediction accuracy with FT(%)	
	Net-trim	Proposed	Net-trim	Proposed	Net-trim	Proposed
$\epsilon=0.01$	61.64	**70.42**	**76.50**	76.14	**76.52**	76.39
$\epsilon=0.04$	65.81	**77.27**	**77.23**	76.50	**76.85**	76.32
$\epsilon=0.08$	69.95	**82.05**	**77.00**	75.92	**76.84**	76.30
$\epsilon=0.10$	71.61	**83.79**	**76.77**	75.94	**76.87**	76.24
$\epsilon=0.20$	77.96	**89.13**	**74.05**	73.38	**76.52**	75.09
$\epsilon=0.30$	82.20	**92.07**	**68.60**	66.88	**75.92**	74.17

We set the parameter $p=100$ in Algorithm 1 and terminate it when the iterates $k=5$, and the subproblem (7) is solved by the ADMM algorithm [24] to enjoy scalability. Table 1 and Table 2 illustrates the results of network compression experiments by varying the parameter ϵ in problem (5). As can be seen from Table 1 and Table 2, the proposed method achieves significantly higher compression ratio with comparable prediction accuracy (with or without fine tuning) compared to Net-trim [13]. For example, Table 1 illustrate that by setting $\epsilon=0.1$, the proposed method outperforms 37.54% in terms of CR compared to Net-trim.

6. CONCLUSION

In this paper, we propose a log-sum minimization approach to prune a large trained DNN layer by layer via enhancing the sparsity during network trimming. To address the resulting nonconvex and nonsmooth optimization problem, we present an iteratively reweighted algorithm to solve it with global convergence guarantees. Specifically, we prove that the sequence generated by our presented algorithm must have cluster points, and every cluster points satisfies the first-order optimality condition of the constrained log-sum minimization problem. Simulation results demonstrate that the proposed method achieves higher network compression ratio with comparable prediction accuracy compared to Net-trim method.

7. REFERENCES

[1] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoustics Speech Signal Process. (ICASSP)*, 2013, pp. 6645–6649.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[3] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[4] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 126–136, Jan. 2018.

[5] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani, "Deep Learning for IoT Big Data and Streaming Analytics: A Survey," *IEEE Commun. Surveys Tut.*, 2018.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. on Learn. Representations (ICLR)*, 2015.

[7] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[8] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 1269–1277.

[9] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *Proc. Eur. Conf. on Comput. Vision (ECCV)*, 2016, pp. 525–542.

[10] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 1135–1143.

[11] Yi Sun, Xiaogang Wang, and Xiaoou Tang, "Sparsifying neural network connections for face recognition," in *Proc. Comput. Vision and Pattern Recognition (CVPR)*, 2016, pp. 4856–4864.

[12] Yiwen Guo, Anbang Yao, and Yurong Chen, "Dynamic network surgery for efficient DNNs," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1379–1387.

[13] Alireza Aghasi, Afshin Abdi, Nam Nguyen, and Justin Romberg, "Net-trim: Convex pruning of deep neural networks with performance guarantee," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3177–3186.

[14] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. of Fourier Anal. and Appl.*, vol. 14, no. 5-6, pp. 877–905, 2008.

[15] Yuanming Shi, Jinkun Cheng, Jun Zhang, Bai Bo, Chen Wei, and Khaled B. Letaief, "Smoothed l_p-minimization for green Cloud-RAN with user admission control," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 4, pp. 1022–1036, Apr. 2016.

[16] Yuanming Shi, Jun Zhang, Chen Wei, and Khaled B. Letaief, "Enhanced group sparse beamforming for green Cloud-RAN: A random matrix approach," *IEEE Trans. Wireless Commun.*, vol. PP, no. 99, pp. 1–1, Nov. 2017.

[17] Zhaosong Lu, "Iterative reweighted minimization methods for ℓ_p regularized unconstrained nonlinear programming," *Math. Programm.*, vol. 147, no. 1-2, pp. 277–307, 2014.

[18] Xiaojun Chen and Weijun Zhou, "Convergence of the reweighted ℓ_1 minimization algorithm for ℓ_2-ℓ_p minimization," *Comput. Optimization and Appl.*, vol. 59, no. 1-2, pp. 47–61, 2014.

[19] Peter Ochs, Alexey Dosovitskiy, Thomas Brox, and Thomas Pock, "On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision," *SIAM J. on Imag. Sci.*, vol. 8, no. 1, pp. 331–372, 2015.

[20] A Ya Kruger, "On fréchet subdifferentials," *J. of Math. Sci.*, vol. 116, no. 3, pp. 3325–3358, 2003.

[21] Yuanming Shi, Jun Zhang, Wei Chen, and Khaled B Letaief, "Generalized sparse and low-rank optimization for ultra-dense networks," *IEEE Commun. Mag.*, vol. 56, no. 6, pp. 42–48, Jun. 2018.

[22] R Tyrrell Rockafellar and Roger J-B Wets, *Variational analysis*, vol. 317, Springer Sci. & Bus. Media, 2009.

[23] Ralph Tyrell Rockafellar, *Convex analysis*, Princeton university press, 2015.

[24] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. and Trends® in Mach. learn.*, vol. 3, no. 1, pp. 1–122, 2011.