Convergence of a Relaxed Variable Splitting Method for Learning Sparse Neural Networks via ℓ_1, ℓ_0, and transformed-ℓ_1 Penalties

Thu Dinh and Jack Xin [*]

December 17, 2018

Abstract

Sparsification of neural networks is one of the effective complexity reduction methods to improve efficiency and generalizability. We consider the problem of learning a one hidden layer convolutional neural network with ReLU activation function via gradient descent under sparsity promoting penalties. It is known that when the input data is Gaussian distributed, no-overlap networks (without penalties) in regression problems with ground truth can be learned in polynomial time at high probability. We propose a relaxed variable splitting method integrating thresholding and gradient descent to overcome the non-smoothness in the loss function. The sparsity in network weight is realized during the optimization (training) process. We prove that under ℓ_1, ℓ_0, and transformed-ℓ_1 penalties, no-overlap networks can be learned with high probability, and the iterative weights converge to a global limit which is a transformation of the true weight under a novel thresholding operation. Numerical experiments confirm theoretical findings, and compare the accuracy and sparsity trade-off among the penalties.

Keywords: Sparse neural networks, sparse penalties, relaxed

variable splitting, thresholding, gradient descent, convergence.

Running Title: Learning sparse neural networks.

AMS Subject Classifications: 90C26, 97R40, 68T05.

ArXiv preprint arXiv:1812.05719.

[*] Department of Mathematics, University of California at Irvine, Irvine, CA 92697, USA. Email: (thud2, jack.xin)@uci.edu.

1 Introduction

Deep neural networks (DNN) have achieved state-of-the-art performance on many machine learning tasks such as speech recognition (Hinton et al., 2012 [18]), computer vision (Krizhevsky et al., 2016 [20]), and natural language processing (Dauphin et al., 2016 [12]). Training such networks is a problem of minimizing a high-dimensional non-convex and non-smooth objective function, and is often solved by simple first-order methods such as stochastic gradient descent. Nevertheless, the success of neural network training remains to be understood from a theoretical perspective. Progress has been made in simplified model problems. Blum & Rivest (1993) showed that even training a 3-node neural network is NP-hard [2], and Shamir (2016) showed learning a simple one-layer fully connected neural network is hard for some specific input distributions [32]. Recently, several works (Tian, 2017 [34]; Brutzkus & Globerson, 2017 [6]) focused on the geometric properties of loss functions, which is made possible by assuming that the input data distribution is Gaussian. They showed that stochastic gradient descent (SGD) with random or zero initialization is able to train a no-overlap neural network in polynomial time.

Another notable issue is that DNNs contain millions of parameters and lots of redundancies, potentially causing over-fitting and poor generalization [42] besides spending unnecessary computational resources. One way to reduce complexity is to sparsify the network weights using an empirical technique called pruning [21] so that the non-essential ones are zeroed out with minimal loss of performance [17, 36, 24]. Recently a surrogate ℓ_0 regularization approach based on a continuous relaxation of Bernoulli random variables in the distribution sense is introduced with encouraging results on small size image data sets [23]. This motivated our work here to study deterministic regularization of ℓ_0 via its Moreau envelope and related ℓ_1 penalties in a one hidden layer convolutional neural network model [6].

The architecture of the network is illustrated in Figure (1). We consider the convolutional setting in which a sparse filter w is shared among different hidden nodes. Assume that the input sample is $x \in \mathbb{R}^n$. We generate k patches from x, each with size d, and let $w \in \mathbb{R}^d$ denote the filter coefficient. Denote $x[i]$ the i^{th} patch of x. We assume that the patches do not overlap so $k = n/d$ and the input distribution is Gaussian. Finally, let σ denote the ReLU activation function, $\sigma(z) := \max\{0, z\}$. The output of the network in Figure (1) is given by:

$$f(x; w) := \frac{1}{k} \sum_{i=1}^{k} \sigma(w \cdot x[i]) \tag{1}$$

We address the realizable case, where training data is generated from a function as in equation (1) with a ground truth weight vector w^*. Training data is then generated by sampling m training points $x_1, .., x_m$ from a Gaussian input, and

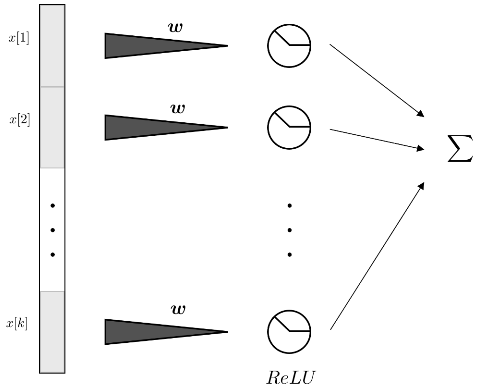


Figure 1: The architecture of a no-overlap neural network

assigning them labels using $y = f(x; w^*)$. The learning problem is then to find a w that minimizes the objective loss function. In other words, solve the optimization problem:

$$\min_w \frac{1}{m} \sum_{j=1}^{m} (f(x_j; w) - y_j)^2 \tag{2}$$

In the limit $m \to \infty$, this is equivalent to minimizing the population risk:

$$f(w) := \mathbb{E}_{x \sim \mathcal{D}} \left[(f(x; w) - f(x; w^*))^2 \right] \tag{3}$$

We note that the iterative thresholding algorithms (IT) are commonly used for retrieving sparse signals ([11, 7, 3, 4, 44] and references therein). In high dimensional setting, IT algorithms provide simplicity and low computational cost, while also promote sparsity of the target vector. We shall investigate the convergence of gradient descent with simultaneous thresholding for the following objective function

$$l(w) = f(w) + \lambda\, P(w) \tag{4}$$

where $f(w)$ is the population loss function of the network, and P is ℓ_0, ℓ_1, or the transformed-ℓ_1 (TL1) function: a one parameter family of bilinear transformations composed with the absolute value function [26, 45]. When acting on vectors, the TL1 penalty interpolates ℓ_0 and ℓ_1 with thresholding in closed analytical form for any parameter value [44]. The ℓ_1 thresholding function is known as soft-thresholding [11, 13], and that of ℓ_0 the hard-thresholding [3, 4]. Due to the non-convex and non-smooth nature of the loss function f, the thresholding part should be properly integrated with gradient descent to be applicable for learning DNNs. As pointed out in [23], it is beneficial to attain sparsity during the optimization (training) process. To this end, we propose a Relaxed Variable Splitting Method (RSVM) with a combined thresholding and proximal gradient descent for minimizing the following augmented objective function

$$\mathcal{L}_\beta(u,w) = f(w) + \lambda\, P(u) + \frac{\beta}{2}\,\|w-u\|^2$$

for a positive parameter β. We note in passing that minimizing $\mathcal{L}_\beta$ in u recovers the original objective (4) with penalty P replaced by its Moreau envelope [25]. We shall show that our algorithm, alternately minimizing u and w, converges for ℓ_0, ℓ_1, and TL1 penalties to a global limit $(\bar{w}, \bar{u})$ with high probability. For a similar model and treatment in a general context, see [1]; and in image processing, see [41]. In the case here, the $\bar{w}$ is a novel thresholded version of the true weight w^* modulo some normalization. The $\bar{u}$ is a sparse approximation of w^*. Furthermore, the ℓ_1 penalty gives the smallest angular error in $\bar{u}$ approximation of w^* and smallest value of $f(\bar{u})$. The ℓ_0 penalty promotes sparsity of $\bar{u}$ most effectively, with smallest angular error in $\bar{w}$ approximation of w^* and smallest value of $f(\bar{w})$. The TL1 penalty gives a middle ground between ℓ_0 and ℓ_1, either in terms of $\bar{u}$ or $\bar{w}$.

The rest of the paper is organized as follows. In Section 2, we briefly overview related mathematical results in the study of neural networks and complexity reduction. Preliminaries are in section 3. In Section 4, we state and discuss the main results. The proofs of the main results and numerical simulations are in Section 5. The acknowledgements are in section 6.

2 Related Work

In recent years, significant progress has been made in the study of convergence in neural network training. From a theoretical point of view, optimizing (training) neural network is a non-convex non-smooth optimization problem. Blum & Rivest; Livni et al.; Shalev-Shwartz et al. showed that training a neural network is hard in the worst cases [2, 22, 31]. Shamir showed that if either the target function or input distribution is "nice", optimization, algorithms used in practice can succeed [32]. Optimization methods in deep neural networks are often categorized into (stochastic) gradient descent methods and others.

Stochastic gradient descent methods were first proposed by Robins and Monro in 1951 [29]. Rumelhart et al. introduced the popular back-propagation algorithm in 1986 [30]. Since then, many well-known SGD methods with adaptive learning rates were proposed and applied in practice, such as the Polyak momentum [27], AdaGrad [16], RMSProp [35], Adam [19], and AMSGrad [28]. The behavior of gradient descent methods in neural networks is better understood when the input has *Gaussian* distribution. In 2017, Tian showed the population gradient descent can recover the true weight vector with random initialization for one-layer one-neuron model [34]. Brutzkus & Globerson (2017) showed that a convolution filter with non-overlapping input can be learned in polynomial time [6]. Du et al. showed (stochastic) gradient descent with random initialization can learn the convolutional filter in polynomial time and the convergence rate depends on the smoothness of the input distribution and the closeness of patches [15]. Du et al. also analyzed the polynomial convergence guarantee of randomly initialized gradient descent algorithm for learning a one-hidden-layer convolutional neural network [14]. A hybrid projected SGD (so called BinaryConnect) is widely used for training various weight quantized DNNs [10, 37]. Recently, a Moreau envelope based relaxation method (BinaryRelax) is proposed and analyzed to advance weight quantization in DNN training [38]. Also a blended coarse gradient descent method [39] is introduced to train fully quantized DNNs in weights and activation functions, and overcome vanishing gradients.

Non-SGD methods for deep learning were also studied in the recent years. Taylor et al. proposed the Alternating Direction Method of Multipliers (ADMM) to transform a fully-connected neural network into an equality-constrained problem to solve [33]. Zhang et al. [43] handled deep supervised hashing (VDSH) problem by an ADMM algorithm to overcome vanishing gradients. Carreira and Wang proposed a method of auxiliary coordinates (MAC) to replace a nested neural network with a constrained problem without nesting [8].

3 Preliminaries

In this paper, the input feature $x \in \mathbb{R}^n$ is i.i.d. Gaussian random vector with zero mean and unit variance. Let $\mathcal{G}$ denote this distribution. We assume that there exists a true w^* by which the training data is generated. The population risk is then:

$$f(w) = \mathbb{E}_{\mathcal{G}}[(f(x;w) - f(x;w^*))^2]. \tag{5}$$

We define

$$g(u,v) = \mathbb{E}_{\mathcal{G}}[\sigma(u \cdot x)\sigma(v \cdot x)]. \tag{6}$$

Then the population loss can be simplified as

$$f(w) = \frac{1}{k^2} \sum_{i,j} [g(w_i, w_i) - 2g(w_i, w_j^*) + g(w_i^*, w_j^*)]. \tag{7}$$

Furthermore, the next two lemmas show that $g(u,v)$ and $l(w)$ can be simplified even more.

Lemma 3.0.1. *(Cho & Saul, 2009 [9]) Assume $x \in \mathbb{R}^d$ is a vector where the entries are i.i.d. Gaussian random variables with mean 0 and variance 1. Given $u, v \in \mathbb{R}^d$, denote by $\theta_{u,v}$ the angle between u and v. Then*

$$g(u,v) = \frac{1}{2\pi} \|u\| \|v\| \left(\sin\theta_{u,v} + (\pi - \theta_{u,v}) \cos\theta_{u,v} \right).$$

In the case of no-overlap networks, the loss function is then simplified to:

$$f(w) = \frac{1}{k^2}\big[\gamma\|w\|^2 + \gamma\|w^*\|^2 - 2kg(w, w^*) - 2\beta\|w\|\|w^*\|\big]. \tag{8}$$

Consider a simple gradient descent update rule for minimizing $f(w)$. Let $\eta > 0$ denote the step size. Then the update at iteration t is:

$$w^{t+1} = w^t - \eta\nabla f(w^t). \tag{9}$$

Next, we will introduce ℓ_0 and ℓ_1 regularization to the population loss function, and the modified gradient update for each case.

3.1 ℓ_1 Penalty and Relaxed Variable Splitting Method

Consider the minimization problem

$$l(w) = f(w) + \lambda\|w\|_1. \tag{10}$$

This problem is equivalent to minimizing $f(w)$ under the constraint $\|w\| \leq t$. We propose a different approach to solve this minimization problem, using the Relaxed Variable Splitting Method (RVSM). We first convert (10) into an equation of two variables

$$l(u, w) = f(w) + \lambda\|u\|_1.$$

and consider the Lagrangian

$$\mathcal{L}_\beta(u, w) = f(w) + \lambda\|u\|_1 + \frac{\beta}{2}\|w - u\|^2. \tag{11}$$

The RSVM is defined as follows:

```
Algorithm 1: RVSM
Initialize $u^0, w^0$;
while stopping criteria not satisfied do
    $u^{t+1} \leftarrow \arg\min_u \mathcal{L}_\beta(u, w^t)$
    $\hat{w}^{t+1} \leftarrow w^t - \eta\nabla f(w^t) - \eta\beta(w^t - u^{t+1})$
    $w^{t+1} \leftarrow \frac{\hat{w}^{t+1}}{\|\hat{w}^{t+1}\|}$
end
Output: $u^t, w^t$
```

Here the update of w^t has the form $w^{t+1} = C^t(w^t - \eta\nabla f(w^t) - \eta\beta(w^t - u^{t+1}))$, where C^t is a normalization constant. This normalization process is unique to our proposed algorithm, and is distinct from other common descent algorithms, for example the Alternate Direction Method of Multipliers (ADMM), where the update of w has the form $w^{t+1} \leftarrow \arg\min_w \mathcal{L}_\beta(u^{t+1}, w, z^t)$ and z_t is the Lagrange multiplier. Since f is non-convex and only Lipschitz differentiable away from zero, convergence analysis of ADMM is beyond the current theory [40]. Here we circumvent the problem by updating w via a simple gradient descent and then normalizing.

3.2 ℓ_0 Penalty

In practice, often time the ground truth w^* is sparse. In this section, we attempt an algorithm to speed up the rate of convergence of the regular gradient descent. Let $H_s(\cdot)$ be the hard-thresholding operator, i.e. H_s keeps the largest s components (in magnitude) and zeros out the rest. Let $\Gamma^t := supp(H_s(w^t))$ and let $\Omega^t := supp(H_s(w^t - \eta\nabla f(w^t)))$. We propose the following algorithm for gradient descent when w^* is s-sparse:

Algorithm 2: Adaptive Hard-thresholding Algorithm

```
Initialize $s$-sparse $w^0$;
while stopping criteria not satisfied do
    $\hat{w}^{t+1} \leftarrow w^t - \eta\nabla f(w^t)$
    if $\Gamma^t \neq \Omega^t$ then
        $w^{t+1} \leftarrow H_s(\hat{w}^{t+1})$
    else
        $w^{t+1} \leftarrow \hat{w}^{t+1}$
    end
end
Output: $w^t$
```

4 Main Results

Theorem 1. *Suppose η is small so that $\|\eta\nabla f(w^t) + \eta\beta(w^t - u^{t+1})\| \leq \frac{1}{2}$, for all t. Suppose also that the initialization satisfies $\theta(w^0, w^*) \leq \pi - \delta$, for some $\delta > 0$, with $\beta \leq \frac{\delta \sin\delta}{k\pi}$ and $\frac{\lambda}{\beta} < \frac{1}{\sqrt{d}}$. Then the RVSM Algorithm converges to a critical point $(\overline{u}, \overline{w})$ when $\eta \leq \frac{1}{\beta+L}$. Let $\theta := \theta(w^*, \bar{w})$, the angle between w^* and $\bar{w}$. Then $\theta < \delta$. Moreover, almost surely, the critical point $\bar{w}$ satisfies*

$$w^* - \frac{k\pi}{\pi - \theta}\beta(\bar{w} - S_{\lambda/\beta}(\bar{w})) = C\bar{w} \tag{12}$$

where $S_{\lambda/\beta}(\cdot)$ is the soft-thresholded operator, for some constant C such that $0 < C \leq \frac{1}{1-2k\lambda\sqrt{d}}$; and

$$\|w^* - \bar{w}\| \leq 4k\beta \sin\gamma \tag{13}$$

where $\gamma := \theta(\bar{w}, S_{\lambda/\beta}(\bar{w}))$. The sparse approximation $\bar{u} = S_{\lambda/\beta}(\bar{w})$ satisfies the estimate:

$$\|w^* - \bar{u}\| \leq 4k\beta \sin\gamma + \frac{\lambda}{\beta}\sqrt{d}. \tag{14}$$

Notice that the sign of $(\bar{w} - S_{\lambda/\beta}(\bar{w}))$ agrees with $\bar{w}$, and $k \leq \frac{k\pi}{\pi-\theta} \leq 2k$. Thus $\bar{w}$ is some soft-thresholded version of w^*, after some normalization.

The assumption on η is achievable: For each t, $\|w^t\| = 1$, thus by [6], $\|\nabla f(t)\|$ is bounded. Moreover, u^{t+1} is a soft-thresholded version of w^t, since it's a regular Lasso regression. Thus $\|\beta(w^t - u^{t+1})\|$ is also bounded.

Definition 1. The transformed l_1 (TL1) function $\rho_a(x)$ is defined as

$$\rho_a(x) = \frac{(a+1)|x|}{a+|x|},$$

for some parameter $a \in (0, +\infty)$. With the change of parameter $'a'$, TL1 interpolates l_0 and l_1 norms:

$$\lim_{a \to 0^+} \rho_a(x) = I_{\{x \neq 0\}}, \quad \lim_{a \to +\infty} \rho_a(x) = |x|.$$

For a vector $x \in \mathbb{R}^d$, we define

$$P_a(x) = \sum_{i=1}^{d} \rho_a(x_i).$$

In Figure below, level lines of TL1 on the plane are shown at small and large values of parameter a, resembling those of l_1 (at $a = 100$), $l_{1/2}$ (at $a = 1$), and l_0 (at $a = 0.01$).

Corollary 1.1. *Under similar conditions, the RVSM Algorithm also converges to a critical point when the l_1 penalty term is replaced with an ℓ_0 or TL1 penalty. That is, the RVSM algorithm converges for the following loss functions:*

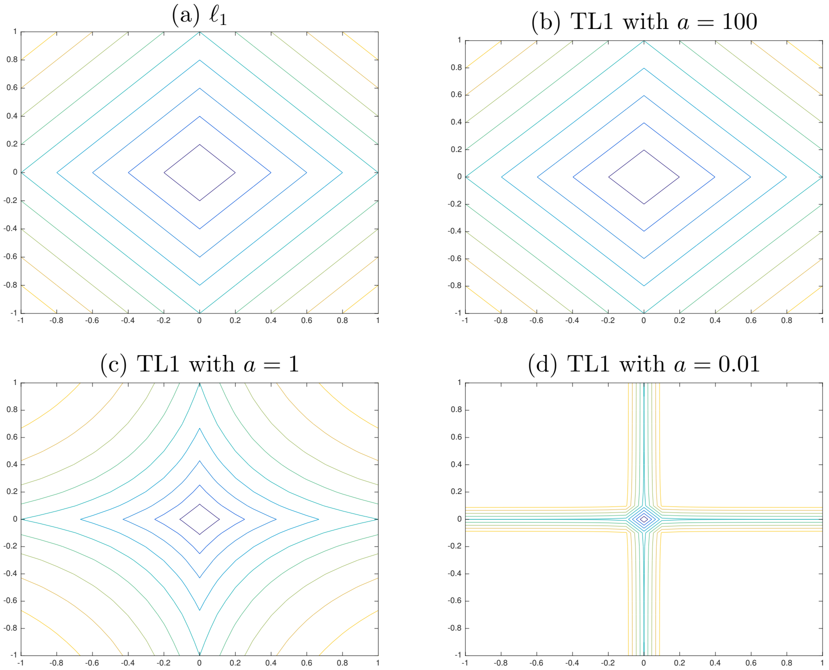
$$\mathcal{L}_{\beta,TL1}(u, w) = f(w) + \lambda P_a(x) + \frac{\beta}{2}\|w - u\|^2,$$

$$\mathcal{L}_{\beta,0}(u, w) = f(w) + \lambda\|u\|_0 + \frac{\beta}{2}\|w - u\|^2$$

and the limit point $\bar{w}$ also satisfies equation (13).

Theorem 2. *Assume $w^0\big|_{\Gamma^*} \neq 0$. If the initialization satisfies $\theta\left(w^0\big|_{\Gamma^*}, w^*\right) \neq \pi$, then the Adaptive Hard-thresholding Algorithm converges to w^*.*

The assumption in theorem 2 is reasonable. We will show that if $w^0\big|_{\Gamma^*} = 0$, then there exists $t \geq 0$ such that the hard-thresholding function is applied in the t^{th} iteration, and $w^{t+1}\big|_{\Gamma^*} \neq 0$. The theorem can then be applied by treating w^{t+1} as w^0.


(a) ℓ_1

(b) TL1 with $a = 100$

(c) TL1 with $a = 1$

(d) TL1 with $a = 0.01$

Figure 2: Level lines of TL1 with different parameters: $a = 100$ (figure b), $a = 1$ (figure c), $a = 0.01$ (figure d). For large parameter a, the graph looks almost the same as l_1 (figure a). While for small value of a, it tends to the axis.

5 Proof of Main Results

5.1 Proof of Theorem 1

Lemma 5.1.1 ([6])**.** *Let $f(w)$ be defined as in Eq. 1. Then the following holds:*
1. $f(w)$ is differentiable if and only if $w \neq 0$.
2. For $k > 1, l(w)$ has three critical points:
(a) A local maximum at $w = 0$.
(b) A unique global minimum at $w = w^*$.
(c) A degenerate saddle point at $w = -\left(\frac{k^2-k}{k^2+(\pi-1)k}\right) w^*$.
For $k = 1, w = 0$ is not a local maximum and the unique global minimum w^* is the only differentiable critical point.

Lemma 5.1.2 ([6])**.** *Assume $\|w_1\|, \|w_2\| \geq M$, w_1, w_2 and w^* are on the same two dimensional half-plane defined by w^*, then*

$$\|\nabla f(w_1) - f(w_2)\| \leq L\|w_1 - w_2\|$$

for $L = 1 + \frac{3\|w^*\|}{M}$.

Lemma 5.1.3 ([6])**.** *Let $g : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function on a set $D \subseteq \mathbb{R}^n$ and $x, y \in D$ such that for all $0 \leq \tau \leq 1, x + \tau(y - x) \in D$ and $\|\nabla f(x + \tau(y - x)) - \nabla f(x)\| \leq L\|x - y\|$. Then we have*

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2}\|x - y\|^2$$

Lemmas 5.1.1, 5.1.2, and 5.1.3 follow directly from [6].

Lemma 5.1.4. *(Descent of $\mathcal{L}_\beta$ due to u update)*
The iteration in Algorithm 1 satisfies

$$\mathcal{L}_\beta(u^{t+1}, w^t) \leq \mathcal{L}_\beta(u^t, w^t)$$

This lemma follows directly from the update rule of u^t.

Lemma 5.1.5. *(Descent of $\mathcal{L}_\beta$ due to w update)*
For η small such that $\eta \leq \frac{2}{\beta}$, we have

$$\mathcal{L}_\beta(u^{t+1}, w^{t+1}) \leq \mathcal{L}_\beta(u^t, w^t).$$

Proof. First notice that since

$$w^{t+1} = C^t(w^t - \eta\nabla f(w^t) - \eta\beta(w^t - u^{t+1}))$$

we have

$$\nabla f(w^t) = \frac{1}{\eta}\left(w^t - \frac{w^{t+1}}{C^t}\right) - \beta(w^t - u^{t+1})$$

For a fixed $u := u^{t+1}$ we have

$$\begin{aligned}
&\mathcal{L}_\beta(u, w^{t+1}) - \mathcal{L}_\beta(u, w^t) \\
&= f(w^{t+1}) - f(w^t) + \frac{\beta}{2}\left(\|w^{t+1} - u\|^2 - \|w^t - u\|^2\right) \\
&\leq \langle \nabla f(w^t), w^{t+1} - w^t \rangle + \frac{L}{2}\|w^{t+1} - w^t\|^2 + \frac{\beta}{2}\left(\|w^{t+1} - u\|^2 - \|w^t - u\|^2\right) \\
&= \frac{1}{\eta}\langle w^t - \frac{w^{t+1}}{C^t}, w^{t+1} - w^t \rangle - \beta\langle w^t - u, w^{t+1} - w^t \rangle \\
&+ \frac{L}{2}\|w^{t+1} - w^t\|^2 + \frac{\beta}{2}\left(\|w^{t+1} - u\|^2 - \|w^t - u\|^2\right) \\
&= \frac{1}{\eta}\langle w^t - \frac{w^{t+1}}{C^t}, w^{t+1} - w^t \rangle + \left(\frac{L}{2} + \frac{\beta}{2}\right)\|w^{t+1} - w^t\|^2 + \frac{\beta}{2}\|w^{t+1} - u\|^2 \\
&- \frac{\beta}{2}\|w^t - u\|^2 - \beta\langle w^t - u, w^{t+1} - w^t \rangle - \frac{\beta}{2}\|w^{t+1} - w^t\|^2 \\
&= \frac{1}{\eta}\langle w^t - \frac{w^{t+1}}{C^t}, w^{t+1} - w^t \rangle + \left(\frac{L}{2} + \frac{\beta}{2}\right)\|w^{t+1} - w^t\|^2
\end{aligned}$$

Since $\|w^t\|, \|w^{t+1}\| = 1$, we know $(w^{t+1} - w^t)$ bisects the angle between w^{t+1} and $-w^t$. The assumption $\|\eta\nabla f(w^t) + \eta\beta(w^t - u^{t+1})\| \leq \frac{1}{2}$ guarantees $\frac{2}{3} \leq C^t \leq 2$ and $\theta(-w^t, w^{t+1}) < \pi$. It follows that $\theta(w^{t+1} - w^t, w^t)$ and $\theta(w^{t+1} - w^t, w^{t+1})$ are strictly less than $\frac{\pi}{2}$. On the other hand, $\left(\frac{w^{t+1}}{C^t} - w^t\right)$ also lies in the plane bounded by w^{t+1} and $-w^t$. Therefore $\theta\left(\frac{w^{t+1}}{C^t} - w^t, w^{t+1} - w^t\right) < \frac{\pi}{2}$. This implies $\langle \frac{w^{t+1}}{C^t} - w^t, w^{t+1} - w^t \rangle \geq 0$. Moreover, when $C^t \geq 1$:

$$\begin{aligned}
&\langle \frac{w^{t+1}}{C^t} - w^t, w^{t+1} - w^t \rangle \\
&= \langle \frac{w^{t+1}}{C^t} - \frac{w^t}{C^t}, w^{t+1} - w^t \rangle - \langle \frac{C^t - 1}{C^t} w^t, w^{t+1} - w^t \rangle \\
&\geq \frac{1}{C^t}\|w^{t+1} - w^t\|^2
\end{aligned}$$

And when $\frac{2}{3} \leq C^t \leq 1$:

$$\begin{aligned}
&\langle \frac{w^{t+1}}{C^t} - w^t, w^{t+1} - w^t \rangle \\
&= \langle w^{t+1} - w^t, w^{t+1} - w^t \rangle + \langle \frac{1 - C^t}{C^t} w^{t+1}, w^{t+1} - w^t \rangle \\
&\geq \|w^{t+1} - w^t\|^2
\end{aligned}$$

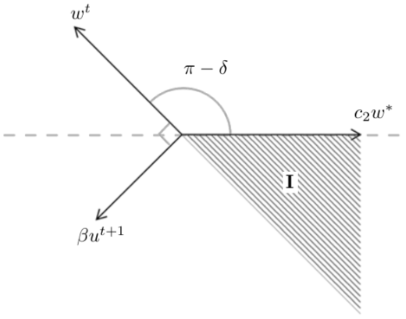


Figure 3: Worst case of the update on w^t

Thus we have

$$\begin{aligned}
&\mathcal{L}_\beta(u, w^{t+1}) - \mathcal{L}_\beta(u, w^t) \\
&\leq \frac{1}{\eta}\langle w^t - \frac{w^{t+1}}{C^t}, w^{t+1} - w^t\rangle + \left(\frac{L}{2} + \frac{\beta}{2}\right)\|w^{t+1} - w^t\|^2 \\
&\leq \left(\frac{L}{2} + \frac{\beta}{2} - \frac{1}{\eta C^t}\chi_{\{C^t \geq 1\}} - \frac{1}{\eta}\chi_{\{\frac{2}{3} \leq C^t \leq 1\}}\right)\|w^{t+1} - w^t\|^2
\end{aligned}$$

Therefore, if η is small so that $\eta \leq \frac{2}{C^t(\beta+L)}$ and $\eta \leq \frac{2}{\beta+L}$, the update on w will decrease $\mathcal{L}_\beta$. Since $C^t \leq 2$, the condition is satisfied when $\eta \leq \frac{1}{\beta+L}$. □

Proof of Theorem 1. We will first show that if $\theta(w^0, w^*) \leq \pi - \delta$, then $\theta(w^t, w^*) \leq \pi - \delta$, for all t. We will show $\theta(w^1, w^*) \leq \pi - \delta$, the statement is then followed by induction. To this end, by the update of w^t, one has

$$\begin{aligned}
w^1 &= C^1 w^0 + \left(\eta\frac{\pi - \theta(w^0, w^*)}{k\pi}\right)w^* + \eta\beta u^1 \\
&= C^1 w^0 + \eta\frac{\delta}{k\pi}w^* + \eta\beta u^1
\end{aligned}$$

for some constant $C > 0$. Since $u^1 = S_{\lambda/\beta}(w^0), \theta(u^1, w^0) \leq \frac{\pi}{2}$. Notice that the sum of the first two terms on the RHS brings the vector closer to w^*, while the last term may behave unexpectedly. Consider the worst case scenario: w^0, w^*, u^1 are co-planar with $\theta(u^1, w^0) = \frac{\pi}{2}$, and w^*, u^1 are on two sides of w^0 (See Figure 3). We need $\frac{\delta}{k\pi}w^* + \beta u^1$ to be in region I. This condition is satisfied when β is small such that

$$\sin\delta \geq \frac{\beta\|u^1\|}{\frac{\delta}{k\pi}\|w^*\|} = \frac{k\pi\beta\|u^1\|}{\delta}$$

or

$$\beta \leq \frac{\delta \sin\delta}{k\pi\|u^1\|}$$

since $u^1 = S_\lambda/\beta(w^0)$, it is sufficient to have

$$\beta \leq \frac{\delta \sin \delta}{k\pi}.$$

Since $\mathcal{L}_\beta(u^t, w^t)$ is non-negative, by Lemma 5.1.4, 5.1.5, $\mathcal{L}_\beta$ converges to some limit $\mathcal{L}$. This implies (u^t, w^t) converges to some stationary point $(\overline{u}, \overline{w})$. By Lemma 5.4.2 and the update of w^t, we have

$$\overline{w} = \bar{C}(c_1\overline{w} + \eta c_2 w^* + \eta\beta\overline{u}) \tag{15}$$

for some constant $\bar{C}, c_1 > 0, c_2 \geq 0$, where $c_2 = \eta\frac{\pi-\theta}{k\pi}$, with $\theta := \theta(\bar{w}, w^*)$, and $\bar{u} = S_{\lambda/\beta}(\bar{w})$. If $c_2 = 0$, then we must have $\bar{w} \mathbin{/\!/} \bar{u}$. But since $\bar{u} = S_{\lambda/\beta}$, this implies all non-zero components of $\bar{w}$ are either equal in magnitude, or all have magnitude smaller than $\frac{\lambda}{\beta}$. The latter case is not possible when $\frac{\lambda}{\beta} < \frac{1}{\sqrt{d}}$. Furthermore, $c_2 = 0$ when $\theta(\bar{w}, w^*) = \pi$ or 0. We have shown that $\theta(\bar{w}, w^*) \leq \pi - \delta$, thus $\theta(\bar{w}, w^*) = 0$. Thus, $\bar{w} = w^*$, and all non-zero components of w^* are equal in magnitude. This has probability zero if we assume w^* is initiated uniformly on the unit circle. Hence we will assume that almost surely, $c_2 > 0$. For expression (15) to hold, we need

$$c_2 w^* + \beta\bar{u} \mathbin{/\!/} \bar{w} \tag{16}$$

Expression (16) implies $\bar{w}, \bar{u}$, and w^* are co-planar. Let $\gamma := \theta(\bar{u}, \bar{w})$. From expression (16), and the fact that $\|\bar{w}\| = \|w^*\| = 1$, we have

$$\begin{aligned}
(\langle c_2 w^* + \beta\bar{u}, \bar{w}\rangle)^2 &= \|c_2 w^* + \beta\bar{u}\|^2\|\bar{w}\|^2 \\
(c_2\langle w^*, \bar{w}\rangle + \beta\langle\bar{u}, \bar{w}\rangle)^2 &= \langle c_2 w^* + \beta\bar{u}, c_2 w^* + \beta\bar{u}\rangle \\
c_2^2\cos^2\theta + 2c_2\beta\|\bar{u}\|\cos\theta\cos\gamma + \beta^2\|\bar{u}\|^2\cos^2\gamma &= c_2^2 + 2c_2\beta\|\bar{u}\|\cos(\theta+\gamma) + \beta^2\|\bar{u}\|^2
\end{aligned}$$

Recall $\cos(a+b) = \cos a\cos b - \sin a\sin b$. Thus

$$\begin{aligned}
c_2^2\sin^2\theta - 2c_2\beta\|\bar{u}\|\sin\theta\sin\gamma + \beta^2\|\bar{u}\|^2\sin^2\gamma &= 0 \\
(c_2\sin\theta - \beta\|\bar{u}\|\sin\gamma)^2 &= 0 \\
\frac{\pi-\theta}{k\pi}\sin\theta &= \beta\|\bar{u}\|\sin\gamma \qquad (17)
\end{aligned}$$

By the initialization of β, we have

$$\frac{\pi-\theta}{k\pi}\sin\theta < \frac{\delta}{k\pi}\sin\delta$$

this implies $\theta < \delta$.

Finally, expression (15) can also be written as

$$\left(w^* - \frac{k\pi}{\pi-\theta}\beta(\bar{w} - \bar{u})\right) \mathbin{/\!/} \bar{w} \tag{18}$$

As θ is close to zero, we can assume $k \le \frac{k\pi}{\pi-\theta} \le 2k$. From expression (18), we see that w^*, after subtracting some vector whose signs agree with $\bar{w}$, and whose non-zero components have the same magnitude between $k\lambda$ and $2k\lambda$, is parallel to $\bar{w}$. This implies $\bar{w}$ is some soft-thresholded version of w^*, after some normalization. Moreover, since $\left\| \frac{k\pi}{\pi-\theta}\beta(\bar{w}-\bar{u}) \right\| \le 2k\lambda\sqrt{d}$, for small λ such that $2k\lambda\sqrt{d} < 1$, we must have

$$\theta\left(w^* - \frac{k\pi}{\pi-\theta}\beta(\bar{w}-\bar{u}), \bar{w}\right) = 0$$

On the other hand,

$$\left\|w^* - \frac{k\pi}{\pi-\theta}\beta(\bar{w}-\bar{u})\right\| \ge \|w^*\| - \left\|\frac{k\pi}{\pi-\theta}\beta(\bar{w}-\bar{u})\right\| \ge 1 - 2k\lambda\sqrt{d}$$

therefore,

$$w^* - \frac{k\pi}{\pi-\theta}\beta(\bar{w}-\bar{u}) = C\bar{w} \tag{19}$$

for some constant C such that $0 < C \le \frac{1}{1-2k\lambda\sqrt{d}}$.

Finally, consider the equilateral triangle with sides $w^*, \bar{w}$, and $w^* - \bar{w}$. By the law of sines,

$$\frac{\|w^*-\bar{w}\|}{\sin\theta} = \frac{\|w^*\|}{\sin\theta(\bar{w}, w^*-\bar{w})} = \frac{1}{\sin\theta(\bar{w}, w^*-\bar{w})}$$

as θ is small, $\theta(\bar{w}, w^*-\bar{w})$ is near $\frac{\pi}{2}$. We can assume $\sin\theta(\bar{w}, w^*-\bar{w}) \ge \frac{1}{2}$. Together with expression (17), we have

$$\|w^*-\bar{w}\| \le 2\sin\theta = \frac{2k\pi\beta\|\bar{u}\|\sin\gamma}{\pi-\theta} \le 4k\beta\|\bar{u}\|\sin\gamma \le 4k\beta$$

The bound on $\|w^* - \bar{u}\|$ follows directly from triangle inequality. □

5.2 Proof of Corollary 1.1.

Lemma 5.2.1 ([44]). *Define a function $f_{\lambda,x}(\cdot) : \mathbb{R} \to \mathbb{R}$,*

$$f_{\lambda,x}(y) = \frac{1}{2}(y-x)^2 + \lambda\rho_a(y)$$

and define

$$g_\lambda(x) = sgn(x)\left\{\frac{2}{3}(a+|x|)\cos\left(\frac{\phi(x)}{3}\right) - \frac{2a}{3} + \frac{|x|}{3}\right\}$$

where $\phi(x) = \arccos\left(1 - \frac{27\lambda a(a+1)}{2(a+|x|)^3}\right)$.
Then the optimal solution $y^*_\lambda(x) = \arg\min_y f_{\lambda,x}(y)$ is a threshold function with threshold value t:

$$y^*_\lambda(x) = \begin{cases} 0, & |x| \le t \\ g_\lambda(x), & |x| > t \end{cases}$$

where the threshold parameter t depends on the regularization parameter λ,
(1) if $\lambda \leq \frac{a^2}{2(a+1)}$ (sub-critical),

$$t = t_2^* = \lambda \frac{a+1}{a}$$

(2) if $\lambda > \frac{a^2}{2(a+1)}$ (super-critical),

$$t = t_3^* = \sqrt{2\lambda(a+1)} - \frac{a}{2}.$$

Lemma 5.2.2 ([44])**.** *Define a function $f_{\lambda,x}(\cdot) : \mathbb{R} \to \mathbb{R}$,*

$$f_{\lambda,x}(y) = \frac{1}{2}(y-x)^2 + \lambda \|y\|_0.$$

Then the optimal solution $y_\lambda^*(x) = \arg\min_y f_{\lambda,x}(y)$ is a threshold function

$$y_\lambda^*(x) = \begin{cases} 0, & |x| < \lambda \\ x, & |x| \geq \lambda \end{cases}$$

Proof of Corollary 1.1: By an outline similar to the proof of Theorem 1:
1/ First we show that $L_{\beta,TL1}(u^t, w^t)$ and $L_{\beta,0}(u^t, w^t)$ both decrease under the update of u^t and w^t. To see this, notice that the update on u^t decreases $L_{\beta,TL1}(u^t, w^t)$ and $L_{\beta,0}(u^t, w^t)$ by definition. Then, for a fixed $u = u^{t+1}$, the update on w^t decreases $L_{\beta,TL1}(u^t, w^t)$ and $L_{\beta,0}(u^t, w^t)$ by a similar argument to that found in Theorem 1.
2/ Next, we show $\theta(w^t, w^*) \leq \pi - \delta$, for some $\delta > 0$, for all t, with initialization $\theta(w^0, w^*) = \pi - \delta$. For $L_{\beta,TL1}(u^t, w^t)$, by Lemma 5.2.1, we have

$$u^{t+1} = (g_{\lambda/\beta}(w_1^t), g_{\lambda/\beta}(w_2^t), ..., g_{\lambda/\beta}(w_d^t))$$

And for $L_{\beta,0}(u^t, w^t)$, by Lemma 5.2.2,

$$u^{t+1} = (w_1^t \chi_{\{|w_1^t| \geq t\}}, w_2^t \chi_{\{|w_2^t| \geq t\}}, ...)$$

In both cases, each component of u^{t+1} is a thresholded version of the corresponding component of w^t. This implies $\theta(u^{t+1}, w^t) \leq \frac{\pi}{2}$, and thus the argument in Theorem 1 follows through, and we have $\theta(w^t, w^*) \leq \pi - \theta$, for all t.
3/ Finally, the equilibrium condition from equation (16) still holds for the critical point, and a similar argument shows that $\theta(\bar{w}, w^*) < \delta$. □

5.3 Comparing Bounds, Sparsity and Population Loss under ℓ_1/TL1/ℓ_0 Penalties - Numerical Simulations

It can be seen from Theorem 1 and Corollary 1.1 that

$$\|w^* - \bar{w}\| \leq 4k\beta \sin\gamma$$

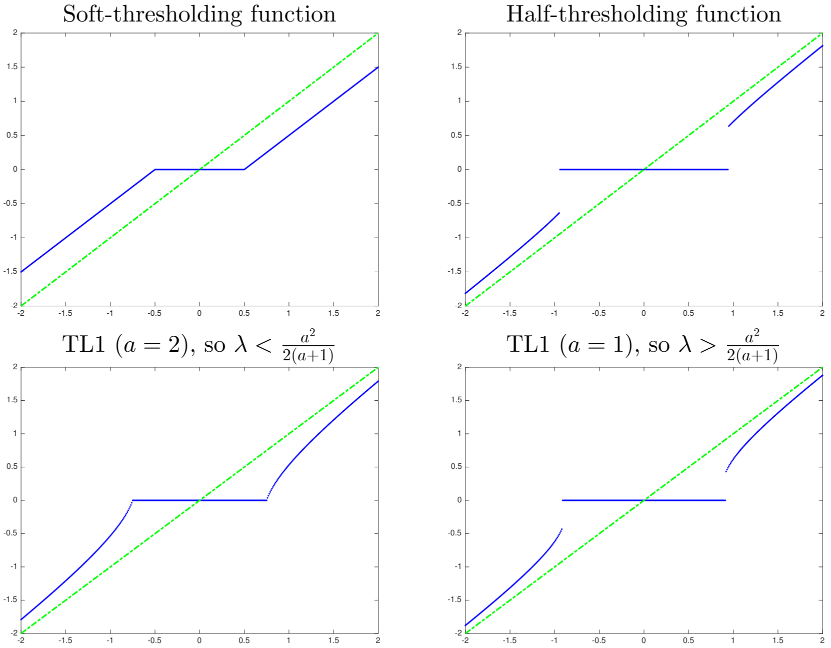


Figure 4: Soft/half (top left/right), TL1 (sub/super critical, lower left/right) thresholding functions at $\lambda = 1/2$.

where $\gamma = \theta(\bar{w}, T_{\lambda/\beta}(\bar{w}))$, for some thresholding function $T_{\lambda/\beta}(\cdot)$ depending on the penalty term. In this section, we investigate how the angle γ changes with respect to different penalty terms.
First we compare the TL1 thresholding function with the hard/soft thresholding function of ℓ_0/ℓ_1 regularization. The plot of these functions are shown in Figure (4). The TL1 thresholding function is continuous when $\lambda < \frac{a^2}{2(a+1)}$, and has a jump discontinuity at threshold otherwise.

The corresponding Huber function $f_{\lambda,x} := \{\frac{1}{2}(y_* - x)^2 + \lambda\, g(y_*)\}$, where $y_* = \text{argmin}_{y \in \mathbb{R}} \{\frac{1}{2}(y - x)^2 + \lambda g(y)\}$ is shown in Figure (5), where $g(y)$ is the ℓ_0, ℓ_1, and $TL1$ norm, respectively.

Let X be a random unit vector, uniformly distributed in $\mathbb{R}^d$ and consider the thresholded vector $T_{\lambda/\beta}(X)$. We run some simulations for a few different penalty terms on the values of $\theta(X, T_{\lambda/\beta}(X))$ and $f(\bar{w}), f(\bar{u})$, where f is the original population loss function of the Neural network.

From the simulation data (see Figure (6)), it can be seen that the ℓ_1 penalty gives the smallest angle change after thresholding. By equation (13), we can expect the ℓ_1 penalty to give the smallest bound on $\|w^* - \bar{w}\|$.
It should be noted that in comparing the errors between ℓ_0, ℓ_1, and TL_1 penalties, if we keep the same parameters λ and β for these runs, the limit $\bar{w}$ under

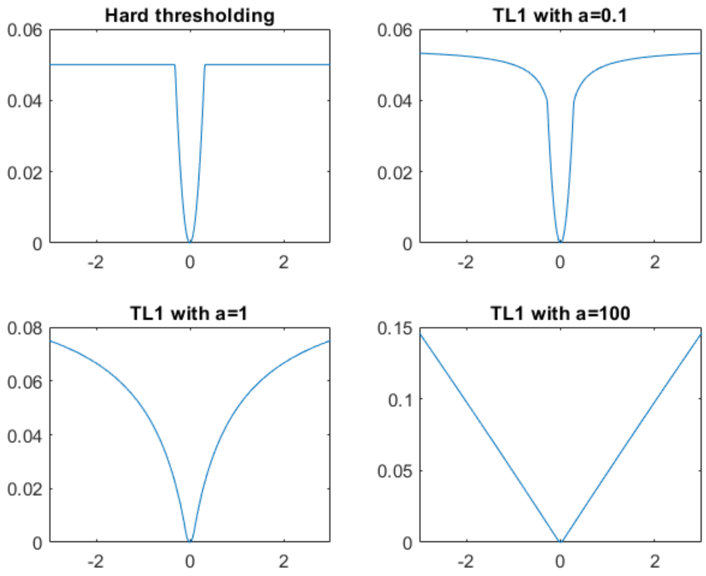


Figure 5: Huber function $f_{\lambda,x}$ in x for different penalties, with $\lambda = 0.05$

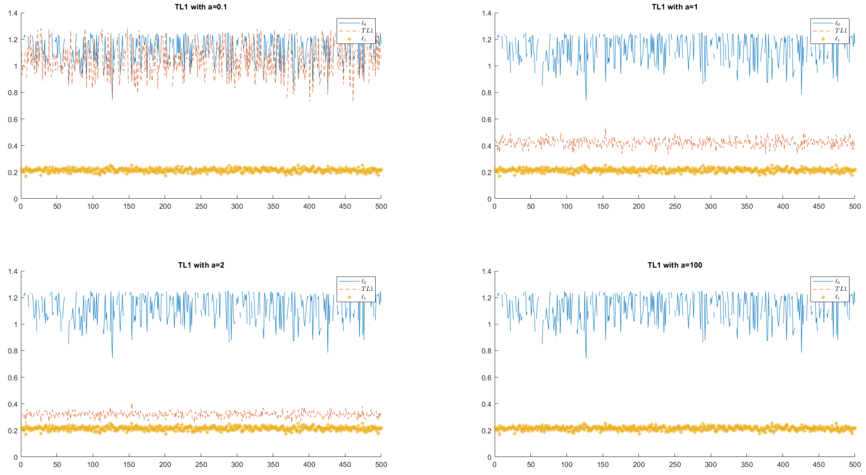


Figure 6: Angle $\gamma = \theta(X, T_{\lambda/\beta}(X))$ vs. realizations of uniformly distributed unit random vector X for different penalties, with $\lambda/\beta = 0.05$

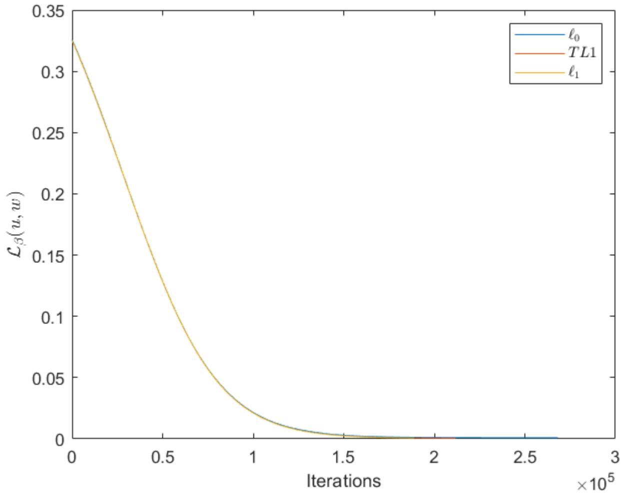


Figure 7: Plot of $\mathcal{L}_\beta(u,w)$ with iterations, with $k = 2, \lambda = 0.0001, \beta = 0.004$, and $d = 30$.

ℓ_0 thresholding is more likely to be killed off and become the zero vector. This is because the thresholding of the ℓ_0 penalty is $(2\lambda/\beta)^{1/2}$, much larger than that of the ℓ_1 penalty, which is λ/β. In other words, if the threshold of the ℓ_1 penalty is about $O(10^{-2n})$, then the threshold of the ℓ_0 penalty is about $O(10^{-n})$, $n = d/k$. Thus as d increases, the hard-thresholding operator is more likely to kick in and kill off all the non-zero components of $\bar{w}$.
Interestingly, the population loss function also appears to behave in a similar manner (see Tables (1), (2), (3)). In other words, the ℓ_1 penalty gives the smallest population loss at the limit point $\bar{w}$, follows by TL_1 and ℓ_0 penalties. It can be seen that there is a trade-off between accuracy and sparsity: The ℓ_0 penalty promotes sparsity at the cost of accuracy, while the ℓ_1 penalty gives better accuracy at the cost of sparsity; and TL_1, depending on the parameter a, gives a middle ground between the two approaches.

As a final remark, the algorithm is observed to converge regardless of normalization, although a theoretical proof seems more difficult without it, which we shall leave as a future work.

5.4 Proof of Theorem 2.

We will show that by iterating algorithm 2, Γ^t converges to $\Gamma^* := supp(w^*)$. The algorithm then reduces to the regular gradient descent on a smaller subspace $\mathbb{R}^s$. Let θ^t denote the angle between w^t and w^*.

Lemma 5.4.1 ([6]). *If $0 < \phi^t < \pi$, $\eta < 1$, and the update w^{t+1} does not apply*

Table 1: Simulation for ℓ_0 penalty with $k = 2, \lambda = 0.0001, \beta = 0.004$

d	10	20	30	40	50
$\theta(\bar{w}, w^*)$	8.75685E-03	6.68412E-03	6.04336E-03	5.32728E-03	4.94954E-03
$f(\bar{w})$	1.91348E-05	1.11535E-05	9.11881E-06	7.08694E-06	6.11803E-06
$\theta(\bar{u}, w^*)$	3.37294E-01	5.28306E-01	6.58853E-01	7.03679E-01	8.68654E-01
$f(\bar{u})$	2.57150E-02	5.83579E-02	8.56018E-02	9.56203E-02	1.34753E-01
$f(\bar{u}/\|\bar{u}\|)$	2.61605E-02	6.05623E-02	9.01282E-02	1.01181E-01	1.44854E-01
$d - \|\bar{u}\|_0$	6	15	25	34	46

Table 2: Simulation for TL_1 penalty with $a = 1, k = 2, \lambda = 0.0001, \beta = 0.004$

d	10	20	30	40	50
$\theta(\bar{w}, w^*)$	2.89468E-02	2.03823E-02	2.03790E-02	1.64539E-02	1.37370E-02
$f(\bar{w})$	2.08177E-04	1.03407E-04	1.03373E-04	6.74448E-05	4.70379E-05
$\theta(\bar{u}, w^*)$	1.05764E-01	1.42955E-01	1.70723E-01	1.93306E-01	2.09155E-01
$f(\bar{u})$	3.94756E-03	8.38579E-03	1.25855E-02	1.74556E-02	2.18136E-02
$f(\bar{u}/\|\bar{u}\|)$	2.73123E-03	4.94569E-03	7.00566E-03	8.93097E-03	1.04133E-02
$d - \|\bar{u}\|_0$	1	4	9	10	13

Table 3: Simulation for ℓ_1 penalty with $k = 2, \lambda = 0.0001, \beta = 0.004$

d	10	20	30	40	50
$\theta(\bar{w}, w^*)$	5.57468E-02	3.98867E-02	3.57740E-02	2.95779E-02	2.56507E-02
$f(\bar{w})$	7.67536E-04	3.94319E-04	3.17482E-04	2.17325E-04	1.63585E-04
$\theta(\bar{u}, w^*)$	7.52479E-02	9.48354E-02	1.00794E-01	1.01500E-01	1.02384E-01
$f(\bar{u})$	2.51967E-03	4.44921E-03	5.79414E-03	7.15888E-03	8.44975E-03
$f(\bar{u}/\|\bar{u}\|)$	1.39230E-03	2.20154E-03	2.48343E-03	2.51794E-03	2.56144E-03
$d - \|\bar{u}\|_0$	0	1	2	4	6

the H_s function, then $\theta^{t+1} < \theta^t$. Furthermore, if the update w^{t+1} does not apply the H_s function, then $||w^{t+1}|| \geq ||w^t||$.

Lemma 5.4.2 ([6])**.** *Let f be defined as in Equation 1, then $\nabla f(w) =$*

$$\frac{1}{k^2}\left[\left(k + \frac{k^2-k}{\pi} - \frac{k\|w^*\|}{\pi\|w\|}\sin\theta_{w,w^*} - \frac{k^2-k}{\pi}\frac{\|w^*\|}{\|w\|}\right)w - \frac{k}{\pi}(\pi - \theta_{w,w^*})w^*\right]$$

Lemma 5.4.3 ([6])**.** *Let f be defined as in Equation 1. Consider the gradient descent $w^{t+1} = w^t - \eta\nabla f(w^t)$ and assume $\theta_{w^0,w^*} \neq \pi$. Let $\epsilon > 0$. Then after $O(\frac{1}{\epsilon^2})$ iterations, w^t is $O(\sqrt{\epsilon})$ close to w^*.*

Lemmas 5.4.1, 5.4.2, and 5.4.3 follow directly from [6].

Proof of Theorem 3. Since the initialization w^0 is s-sparse, for every t, the update w^t is at most $2s$-sparse. We will show that if $\Gamma^t \neq \Gamma^*$, the algorithm will eventually change Γ^t to be closer to Γ^*. In other words, there exists j such that $\Gamma^{t+j} \neq \Gamma^t$, and $\Gamma^{t+j} \setminus \Gamma^t \subset \Gamma^*$.

For entries in $\Gamma^* \setminus \Gamma^t$, the update has the form

$$w_i^{t+1} = c_1(t)w^t + c_2(t)w^*$$

for some $c_1(t) \geq -1, c_2(t) \geq 0$. By [6], this implies $w^t\big|_{\Gamma^*\setminus\Gamma^t}$ converges to $w^*\big|_{\Gamma^*\setminus\Gamma^t}$. And for entries in $\Gamma^t \setminus \Gamma^*$,

$$w_i^{t+1} = \left[1 - \frac{\eta}{k^2}\left(k + \frac{k^2-k}{\pi} - \frac{k\|w^*\|}{\pi\|w\|}\sin\theta_{w,w^*} - \frac{k^2-k}{\pi}\frac{\|w^*\|}{\|w\|}\right)\right]w^t$$

Notice that $c_1(t) = 1 - \frac{1}{k^2}\left(k + \frac{k^2-k}{\pi} - \frac{k\|w^*\|}{\pi\|w\|}\sin\theta_{w,w^*} - \frac{k^2-k}{\pi}\frac{\|w^*\|}{\|w\|}\right)$ is bounded between 0 and 1. Thus the magnitudes of the entries in $\Gamma^t \setminus \Gamma^*$ are decreasing after each update. If the update w^{t+1} does not apply the H_s function, the same argument applies, and after n updates, we

$$w^{t+n}\big|_{\Gamma^t\setminus\Gamma^*} = \left(\prod_{i=1}^{n} c_i(t)\right) w^*\big|_{\Gamma^t\setminus\Gamma^*} \tag{20}$$

where $0 < c_i < 1$, for $i = 1, ..., n$. By equation 20, the entries in $\Gamma^t \setminus \Gamma^*$ converge to zero. So if the n^{th} update applies the H_s function, it must replace one of the entries from $\Gamma^{t+n} \setminus \Gamma^*$ by an entry from Γ^*.

After the H_s function has been applied, the process starts again, and eventually another entry from Γ^* will be included in an update where the H_s function is applied. As $|\Gamma^*| < \infty$, eventually the algorithm reduces to a regular gradient descent on $\mathbb{R}^s$.

It remains to show that if $w^0\big|_{\Gamma^*} \neq 0$ and $\theta\left(w^0\big|_{\Gamma^*}, w^*\right) \neq \pi$, then for all subsequent updates, $\theta\left(w^t\big|_{\Gamma^*}, w^*\right) \neq \pi$. If an update does not apply the hard-thresholding function, then the statement is true by Lemma 5.4.1. Otherwise, we have shown that the hard-thresholding function removes an entry from Γ^t and replaces it with an entry from Γ^*. All the entries in $w^t\big|_{\Gamma^*}$ that are non-zero since the last hard-thresholding remains unchanged, and therefore $\theta\left(w^t\big|_{\Gamma^*}, w^*\right) \neq \pi$. □

6 Acknowledgments

The authors would like to thank Dr. Yingyong Qi for suggesting reference [23], and Dr. Penghang Yin for helpful discussions. The work was partially supported by NSF grants DMS-1522383 and IIS-1632935.

References

[1] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal Alternating Minimization and Projection Methods for Nonconvex Problems: An Approach Based on the Kurdyka-Lojasiewicz Inequality. Mathematics of Operations Research, vol. 35, no. 2, 2010, pp. 438-457, doi:10.1287/moor.1100.0449.

[2] A. Blum and R.L. Rivest. Training a 3-node neural network is NP-complete. In Advances in neural information processing systems, pp. 494-501, 1989.

[3] T. Blumensath and M. Davies. Iterative thresholding for sparse approximations, J. Fourier Anal. and Appl., 14(5-6):629-654, 2008.

[4] T. Blumensath. Accelerated iterative hard thresholding, Signal Process., 92(3):752-756, 2012.

[5] J. Bolte, S. Sabach, and M. Teboulle. Proximal Alternating Linearized Minimization for Nonconvex and Nonsmooth Problems. Mathematical Programming, vol. 146, no. 1-2, 2013, pp. 459-494., doi:10.1007/s10107-013-0701-9.

[6] A. Brutzkus, and A. Globerson. "Globally Optimal Gradient Descent for a ConvNet with Gaussian Inputs." ArXiv preprint, Arxiv: 1702.07966, 2017.

[7] E. Candès, J. Romberg, and T. Tao, Stable signal recovery from incomplete and inaccurate measurements, Comm. Pure Applied Math., 59(8):1207-1223, 2006.

[8] M. Carreira-Perpinan and W. Wang. Distributed optimization of deeply nested systems. In Artificial Intelligence and Statistics, pages 10–19, 2014.

[9] Y. Cho and L.K. Saul. Kernel methods for deep learning. In Advances in neural information processing systems, pp. 342–350, 2009.

[10] M. Courbariaux, Y. Bengio, and J. David. Binaryconnect: Training deep neural networks with binary weights during propagations. NIPS, 2015, pp. 3123-3131.

[11] I. Daubechies, M. Defrise, and C.D. Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, Comm. Pure Applied Math., 57(11):1413-1457, 2004.

[12] Y.N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. arXiv preprint arXiv:1612.08083, 2016.

[13] D. Donoho. Denoising by soft-thresholding, IEEE Trans. Info. Theory, 41(3):613-627, 1995.

[14] S.S. Du, J.D. Lee, Y. Tian, B. Poczos, and A. Singh. Gradient Descent Learns One-hidden-layer CNN: Dont be Afraid of Spurious Local Minima. International Conference on Machine Learning (ICML), 2018.

[15] S.S. Du, J.D. Lee, and Y.Tian. When is a convolutional filter easy to learn? arXiv preprint arXiv:1709.06129, 2017b.

[16] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12(Jul):2121-2159, 2011.

[17] S. Han, H. Mao, W.J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding, arXiv preprint arXiv:1510.00149, 2015.

[18] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine, 29(6):82-97, 2012.

[19] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[20] A. Krizhevsky, I. Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pp. 1097-1105, 2012.

[21] Y. LeCun, J.S. Denker, S.A. Solla. Optimal Brain Damage. NIPS 1989, vol. 2, pp. 598-605.

[22] R. Livni, S. Shalev-Shwartz, and O. Shamir. On the computational efficiency of training neural networks. In Advances in Neural Information Processing Systems, pp. 855-863, 2014.

[23] C. Louizos, M. Welling, D.P. Kingma. Learning Sparse Neural Networks Through ℓ_0 Regularization. ICLR 2018. arXiv preprint arXiv:1712.01312v2, June 22, 2018.

[24] D. Molchanov, A. Ashukha, D. Vetrov. Variational dropout sparsifies deep neural networks. arXiv preprint arXiv:1701.05369, 2017.

[25] J.-J. Moreau. Proximité et dualité dans un espace hilbertien, Bulletin de la Société Mathématique de France, 93 (1965), pp. 273–299.

[26] M. Nikolova. Local strong homogeneity of a regularized estimator. SIAM Journal on Applied Mathematics 61(2), 2000, pp. 633-658.

[27] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 4(5):1-17, 1964.

[28] S.J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. In International Conference on Learning Representations, 2018.

[29] H. Robbins and S. Monro. A stochastic approximation method. Annals Math. Statistics, 22:400-407, 1951.

[30] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. Nature, 323(6088):533, 1986.

[31] S. Shalev-Shwartz, O. Shamir, and S. Shammah. Failures of gradient-based deep learning. In International Conference on Machine Learning, pp. 3067-3075, 2017a.

[32] O. Shamir. Distribution-specific hardness of learning neural networks. arXiv preprint arXiv:1609.01037, 2016.

[33] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein. Training neural networks without gradients: A scalable admm approach. In International Conference on Machine Learning, pages 2722-2731, 2016.

[34] Y. Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. arXiv preprint arXiv:1703.00560, 2017.

[35] T. Tieleman and G. Hinton. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. Technical report, Technical Report. Available online: https://zh. coursera. org/learn/neuralnetworks/lecture/YQHki/rmsprop-divide-the-gradientby-a-running-average-of-its-recent-magnitude (accessed on 21 April 2017).

[36] K. Ullrich, E. Meeds, M. Welling. Soft weight-sharing for neural network compression, ICLR 2017.

[37] P. Yin, S. Zhang, Y. Qi, J. Xin. Quantization and Training of Low Bit-Width Convolutional Neural Networks for Object Detection. arXiv:1612.06052v2; J. Computational Mathematics, 37(3), 2019, pp. 1–12. Online August 16, 2018: doi:10.4208/jcm.1803-m2017-0301.

[38] P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, and J. Xin. BinaryRelax: A Relaxation Approach for Training Deep Neural Networks with Quantized Weights. To appear in SIAM Journal on Imaging Sciences; arXiv preprint arXiv:1801.06313, 2018.

[39] P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, and J. Xin. Blended Coarse Gradient Descent for Full Quantization of Deep Neural Networks. Research in the Mathematical Sciences to appear, DOI: 10.1007/s40687-018-0177-6; arXiv preprint arXiv:1808.05240, 2018.

[40] Y. Wang, J. Zeng, W. Yin. Global Convergence of ADMM in Nonconvex Nonsmooth Optimization. Journal of Scientific Computing, online June 2018. doi:10.1007/s10915-018-0757-z.

[41] T. Wu. Variable Splitting Based Method for Image Restoration with Impulse Plus Gaussian Noise. Mathematical Problems in Engineering, vol. 2016, 2016, pp. 1-16. doi:10.1155/2016/3151303.

[42] C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530, 2016.

[43] Z. Zhang, Y. Chen, and V. Saligrama. Efficient training of very deep neural networks for supervised hashing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1487-1495, 2016.

[44] S. Zhang and J. Xin. Minimization of Transformed l_1 Penalty: Closed Form Representation and Iterative Thresholding Algorithms. Communications in Mathematical Sciences, vol. 15, no. 2, 2017, pp. 511-537. doi:10.4310/cms.2017.v15.n2.a9.

[45] S. Zhang and J. Xin. Minimization of Transformed l_1 Penalty: Theory, Difference of Convex Function Algorithm, and Robust Application in Compressed Sensing. Mathematical Programming, Series B, 169(1), pp. 307–336, 2018. doi.org/10.1007/s10107-018-1236-x