# DessiLBI: Exploring Structural Sparsity of Deep Networks via Differential Inclusion Paths

**Yanwei Fu** [1] **Chen Liu** [1] **Donghao Li** [1 2] **Xinwei Sun** [3] **Jinshan Zeng** [4 2] **Yuan Yao** [2]

## Abstract

Over-parameterization is ubiquitous nowadays in training neural networks to benefit both optimization in seeking global optima and generalization in reducing prediction error. However, compressive networks are desired in many real world applications and direct training of small networks may be trapped in local optima. In this paper, instead of pruning or distilling over-parameterized models to compressive ones, we propose a new approach based on *differential inclusions of inverse scale spaces*. Specifically, it generates a family of models from simple to complex ones that couples a pair of parameters to simultaneously train over-parameterized deep models and structural sparsity on weights of fully connected and convolutional layers. Such a differential inclusion scheme has a simple discretization, proposed as **De**ep **s**tructurally **spl**itting **L**inearized **B**regman **I**teration (DessiLBI), whose global convergence analysis in deep learning is established that from any initializations, algorithmic iterations converge to a critical point of empirical risks. Experimental evidence shows that DessiLBI achieve comparable and even better performance than the competitive optimizers in exploring the structural sparsity of several widely used backbones on the benchmark datasets. Remarkably, with *early stopping*, DessiLBI unveils "*winning tickets*" in early epochs: the effective sparse structure with comparable test accuracy to fully trained over-parameterized models.

[1]School of Data Science, and MOE Frontiers Center for Brain Science, Shanghai Key Lab of Intelligent Information Processing Fudan University. yanweifu@fudan.edu.cn [2]Hong Kong University of Science and Technology [3]Microsoft Research-Asia [4]Jiangxi Normal University,. Correspondence to: Yuan Yao <yuany@ust.hk>.

*Proceedings of the $37^{th}$ International Conference on Machine Learning*, Online, PMLR 119, 2020. 

## 1 Introduction

The expressive power of deep neural networks comes from the millions of parameters, which are optimized by Stochastic Gradient Descent (SGD) (Bottou, 2010) and variants like Adam (Kingma & Ba, 2015). Remarkably, model over-parameterization helps both optimization and generalization. For optimization, over-parameterization may simplify the landscape of empirical risks toward locating global optima efficiently by gradient descent method (Mei et al., 2018; 2019; Venturi et al., 2018; Allen-Zhu et al., 2018; Du et al., 2018). On the other hand, over-parameterization does not necessarily result in a bad generalization or overfitting (Zhang et al., 2017), especially when some weight-size dependent complexities are controlled (Bartlett, 1997; Bartlett et al., 2017; Golowich et al., 2018; Zhu et al., 2018; Neyshabur et al., 2019).

However, compressive networks are desired in many real world applications, *e.g.* robotics, self-driving cars, and augmented reality. Despite that $\ell_1$ regularization has been applied to deep learning to enforce the sparsity on weights toward compact, memory efficient networks, it sacrifices some prediction performance (Collins & Kohli, 2014). This is because that the weights learned in neural networks are highly correlated, and $\ell_1$ regularization on such weights violates the incoherence or irrepresentable conditions needed for sparse model selection (Donoho & Huo, 2001; Tropp, 2004; Zhao & Yu, 2006), leading to spurious selections with poor generalization. On the other hand, $\ell_2$ regularization is often utilized for correlated weights as some low-pass filtering, sometimes in the form of weight decay (Loshchilov & Hutter, 2019) or early stopping (Yao et al., 2007; Wei et al., 2017). Furthermore, group sparsity regularization (Yuan & Lin, 2006) has also been applied to neural networks, such as finding optimal number of neuron groups (Wen et al., 2016) and exerting good data locality with structured sparsity (Wen et al., 2016; Yoon & Hwang, 2017).

Yet, without the aid of over-parameterization, directly training a compressive model architecture may meet the obstacle of being trapped in local optima in contemporary experience. Alternatively, researchers in practice typically start from training a big model using common task datasets like ImageNet, and then prune or distill such big models to small ones without sacrificing too much of the performance (Jaderberg et al., 2014; Han et al., 2015; Li et al., 2017; Abbasi-Asl & Yu, 2017; Arora et al., 2018). In particular, a recent study (Frankle & Carbin, 2019) created the *lottery ticket hypothesis* based on empirical observations: "dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) that – when trained in isolation – reach test accuracy comparable to the original network in a similar number of iterations". How to effectively reduce an over-parameterized model thus becomes the key to compressive deep learning. Yet, (Liu et al., 2019) raised a question, *is it necessary to fully train a dense, over-parameterized model before finding important structural sparsity*?

This paper provides a novel answer by exploiting a dynamic approach to deep learning with structural sparsity. We are able to establish a family of neural networks, from simple to complex, by following regularization paths as solutions of *differential inclusions of inverse scale spaces*. Our key idea is to design some dynamics that simultaneously exploit over-parameterized models and structural sparsity. To achieve this goal, the original network parameters are lifted to a coupled pair, with one *weight set* $W$ of parameters following the standard gradient descend to explore the over-parameterized model space, while the other set of parameters $\Gamma$ learning structure sparsity in an *inverse scale space*. The large-scale important parameters are learned at faster speed than small unimportant ones. The two sets of parameters are coupled in an $\ell_2$ regularization. This dynamics on highly non-convex (e.g. deep models) setting enjoys a simple discretization, which is proposed as **De**ep **s**tructurally **spli**tting **L**inearized **B**regman **I**teration (DessiLBI) with provable global convergence guarantee in this paper. Here, DessiLBI is a natural extension of SGD with structural sparsity exploration: DessiLBI reduces to the standard gradient descent method when the coupling regularization is weak, while reduces to a sparse mirror descent when the coupling is strong.

Critically, DessiLBI enjoys a nice property that effective subnetworks can be rapidly learned via structural sparsity parameter $\Gamma$ by the iterative regularization path without fully training a dense network first. Particularly, support set of structural sparsity parameter $\Gamma$ learned in the early stage of this inverse scale space discloses important sparse subnetworks. Such architectures can be fine-tuned or re-trained to achieve comparable test accuracy as the dense, over-parameterized networks. As a result, structural sparsity parameter $\Gamma$ may enable us to rapidly find "winning tickets" in early training epochs for the "lottery" of identifying successful subnetworks that bear comparable test accuracy to the dense ones, confirmed empirically by experiments.

**Contributions**. (1) DessiLBI is, for the first time, applied to explore the structural sparsity of over-parameterized deep network via differential inclusion paths. DessiLBI can be interpreted as the discretization of the dynamic approach of differential inclusion paths in the inverse scale space. (2) Global convergence of DessiLBI in such a nonconvex optimization is established based on the Kurdyka-Łojasiewicz framework, that the whole iterative sequence converges to a critical point of the empirical loss function from arbitrary initializations. (3) Stochastic variants of DessiLBI demonstrate the comparable and even better performance than other training algorithms on ResNet-18 in large scale training such as ImageNet-2012, among other datasets, together with additional structural sparsity in successful models for interpretability. (4) Structural sparsity parameters in DessiLBI provide important information about subnetwork architecture with comparable or even better accuracies than dense models before and after retraining – *DessiLBI with early stopping* can provide fast "*winning tickets*" without fully training dense, over-parameterized models.

## 2 Preliminaries and Related Work

**Mirror Descent Algorithm (MDA)** firstly proposed by (Nemirovski & Yudin, 1983) to solve constrained convex optimization $L^\star := \min_{W\in K} \mathcal{L}(W)$ ($K$ is convex and compact), can be understood as a generalized projected gradient descent (Beck & Teboulle, 2003) with respect to Bregman distance $B_\Omega(u,v) := \Omega(u) - \Omega(v) - \langle \nabla\Omega(v), u-v\rangle$ induced by a convex and differentiable function $\Omega(\cdot)$,

$$Z_{k+1} = Z_k - \alpha\nabla\mathcal{L}(W_k) \tag{1a}$$

$$W_{k+1} = \nabla\Omega^\star(Z_{k+1}), \tag{1b}$$

where the conjugate function of $\Omega(\cdot)$ is $\Omega^\star(Z) := \sup_W \langle W, Z\rangle - \Omega(W)$. Equation (1) optimizes $W_{k+1} = \arg\min_z \langle z, \alpha\mathcal{L}(W_k)\rangle + B_\Omega(z, W_k)$ (Nemirovski) in two steps: Eq (1a) implements the gradient descent on $Z$ that is an element in dual space $Z_k = \nabla\Omega(W_k)$; and Eq (1b) projects it back to the primal space. As step size $\alpha \to 0$, MDA has the following limit dynamics as ordinary differential equation (ODE) (Nemirovski & Yudin, 1983):

$$\dot{Z}_t = \alpha\nabla\mathcal{L}(W_t) \tag{2a}$$

$$W_t = \nabla\Omega^\star(Z_t), \tag{2b}$$

Convergence analysis with rates have been well studied for convex loss, that has been extended to stochastic version (Ghadimi & Lan, 2012; Nedic & Lee, 2014) and Nesterov acceleration scheme (Su et al., 2016; Krichene et al., 2015). For highly non-convex loss met in deep learning, (Azizan et al., 2019) established the convergence to global optima for *overparameterized* networks, provided that (i) the initial point is close enough to the manifold of global optima; (ii) the $\Omega(\cdot)$ is strongly convex and differentiable. For non-differentiable $\Omega$ such as the Elastic Net penalty in compressed sensing and high dimensional statistics ($\Omega(W) = \|W\|_1 + \frac{1}{2\kappa}\|W\|_F^2$), Eq. (1) is studied as the Linearized Bregman Iteration (LBI) in applied mathematics (Yin et al., 2008; Osher et al., 2016) that follows a discretized solution path of differential inclusions, to be discussed below. Such solution paths play a role of sparse regularization path where early stopped solutions are often better than the convergent ones when noise is present. In this paper, we investigate a varied form of LBI for the highly non-convex loss in deep learning models, exploiting the sparse paths, and establishing its convergence to a KKT point for *general* networks from *arbitrary initializations*.

**Linearized Bregman Iteration (LBI)**, was proposed in (Osher et al., 2005; Yin et al., 2008) that firstly studies Eq. (1) when $\Omega(W)$ involves $\ell_1$ or total variation non-differentiable penalties met in compressed sensing and image denoising. Beyond convergence for convex loss (Yin et al., 2008; Cai et al., 2009), Osher et al. (2016) and Huang et al. (2018) particularly showed that LBI is a discretization of differential inclusion dynamics whose solutions generate iterative sparse regularization paths, and established the statistical model selection consistency for high-dimensional generalized linear models. Moreover, Huang et al. (2016; 2018) further improved this by proposing SplitLBI, incorporating into LBI a variable splitting strategy such that the restricted Hessian with respect to augmented variable ($\Gamma$ in Eq. 3) is orthogonal. This can alleviate the multicollinearity problem when the features are highly correlated; and thus can relax the irrepresentable condition, *i.e.*, the necessary condition for Lasso to have model selection consistency (Tropp, 2004; Zhao & Yu, 2006). However, existing work on SplitLBI is restricted to convex problems in generalized linear modes. It remains unknown whether the algorithm can exploit the structural sparsity in highly non-convex deep networks. To fill in this gap, in this paper, we propose the deep Structural Splitting LBI that simultaneously explores the overparameterized networks and the structural sparsity of the weights of fully connected and convolutional layers in such networks, which enables us to generate an iterative solution path of deep models whose important sparse architectures are unveiled in early stopping.

**Alternating Direction Method of Multipliers (ADMM)** which also adopted variable splitting strategy, breaks original complex loss into smaller pieces with each one can be easily solved iteratively (Wahlberg et al., 2012; Boyd et al., 2011). Equipped with the variable splitting term, (He & Yuan, 2012; Wang & Banerjee, 2013) and (Zeng et al., 2019b) established the convergence result of ADMM in convex, stochastic and non-convex setting, respectively. (Wang & Banerjee, 2014) studied convergence analysis with respect to Bregman distance. Recently, (Franca et al., 2018) derived the limit ODE dynamics of ADMM for convergent analysis. However, one should distinguish the LBI dynamics from ADMM that LBI should be viewed as a discretization of differential inclusion of inverse scale space that generalizes a sparse regularization solution path from simple to complex models where early stopping helps find important sparse models; in contrast, the ADMM, as an optimization algorithm for a given objective function, focuses on convergent property of the iterations.

# 3 Methodology

Supervised learning learns $\Phi_W : \mathcal{X} \to \mathcal{Y}$, from input $\mathcal{X}$ to output space $\mathcal{Y}$, with a parameter $W$ such as weights in neural networks, by minimizing certain loss functions on training samples $\widehat{\mathcal{L}}_n(W) = \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, \Phi_W(x_i))$. For example, a neural network of $l$-layer is defined as $\Phi_W(x) = \sigma_l\left(W^l\sigma_{l-1}\left(W^{l-1}\cdots\sigma_1\left(W^1x\right)\right)\right)$, where $W = \{W^i\}_{i=1}^{l}$, $\sigma_i$ is the nonlinear activation function of the $i$-th layer.

**Differential Inclusion of Inverse Scale Space.** Consider the following dynamics,

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}\left(W_t, \Gamma_t\right) \tag{3a}$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}\left(W_t, \Gamma_t\right) \tag{3b}$$

$$V_t \in \partial\bar{\Omega}(\Gamma_t) \tag{3c}$$

where $V$ is a sub-gradient of $\bar{\Omega}(\Gamma) := \Omega_\lambda(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2$ for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda(\Gamma) = \lambda\Omega_1(\Gamma)$ ($\lambda \in \mathbb{R}_+$) such as Lasso or group Lasso penalties for $\Omega_1(\Gamma)$, $\kappa > 0$ is a damping parameter such that the solution path is continuous, and the augmented loss function is

$$\bar{\mathcal{L}}\left(W, \Gamma\right) = \widehat{\mathcal{L}}_n\left(W\right) + \frac{1}{2\nu}\|W - \Gamma\|_F^2, \tag{4}$$

with $\nu > 0$ controlling the gap admitted between $W$ and $\Gamma$. Compared to the original loss function $\widehat{\mathcal{L}}_n\left(W\right)$, our loss $\bar{\mathcal{L}}\left(W, \Gamma\right)$ additionally uses variable splitting strategy by lifting the original neural network parameter $W$ to $(W, \Gamma)$ with $\Gamma$ modeling the structural sparsity of $W$. For simplicity, we assumed $\bar{\mathcal{L}}$ is differentiable with respect to $W$ here, otherwise the gradient in Eq. (3a) is understood as subgradient and the equation becomes an inclusion.

Differential inclusion system (Eq. 3) is a coupling of gradient descent on $W$ with non-convex loss and mirror descent (LBI) of $\Gamma$ (Eq. 2) with non-differentiable sparse penalty. It may explore dense over-parameterized models $W_t$ in the proximity of structural parameter $\Gamma_t$ with gradient descent, while $\Gamma_t$ records important sparse model structures.

Specifically, the solution path of $\Gamma_t$ exhibits the following property in the separation of scales: starting at the zero, important parameters of large scale will be learned fast, popping up to be nonzeros early, while unimportant parameters of small scale will be learned slowly, appearing to be nonzeros late. In fact, taking $\Omega_\lambda(\Gamma) = \|\Gamma\|_1$ and $\kappa \to \infty$ for simplicity, $V_t$ as the subgradient of $\bar{\Omega}_t$, undergoes a gradient descent flow before reaching the $\ell_\infty$-unit box, which implies that $\Gamma_t = 0$ in this stage. The earlier a component in $V_t$ reaches the $\ell_\infty$-unit box, the earlier a corresponding component in $\Gamma_t$ becomes nonzero and rapidly evolves toward a critical point of $\bar{\mathcal{L}}$ under gradient flow. On the other hand, the $W_t$ follows the gradient descent with a standard $\ell_2$-regularization. Therefore, $W_t$ closely follows dynamics

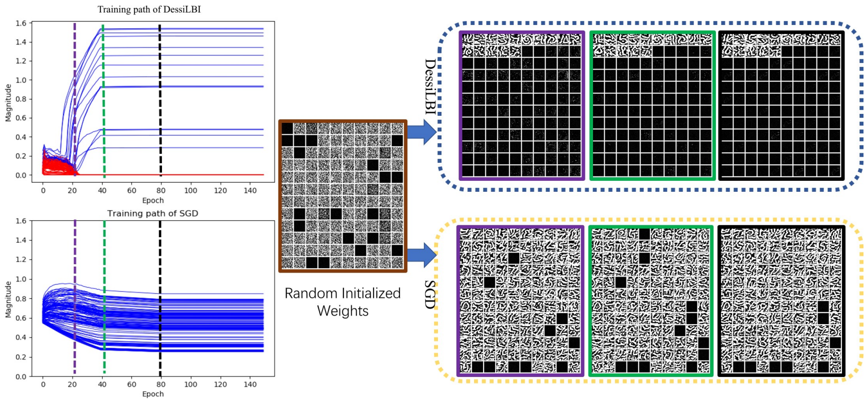



*Figure 1.* Visualization of solution path and filter patterns in the third convolution al layer (i.e., conv.c5) of LetNet-5, trained on MNIST. The left figure shows the magnitude changes for each filter of the models trained by DessiLBI and SGD, where $x$-axis and $y$-axis indicate the training epochs, and filter magnitudes ($\ell_2$-norm), respectively. The DessiLBI path of filters selected in the support of $\Gamma$ are drawn in blue color, while the red color curves represent the filters that are not important and outside the support of $\Gamma$. We visualize the corresponding learned filters by Erhan et al. (2009) at 20 (blue), 40 (green), and 80 (black) epochs, which are shown in the right figure with the corresponding color bounding boxes, *i.e.*, blue, green, and black, respectively. It shows that our DessiLBI enjoys a sparse selection of filters without sacrificing accuracy (see Table 1).

of $\Gamma_t$ whose important parameters are selected.

Compared with directly enforcing a penalty function such as $\ell_1$ or $\ell_2$ regularization

$$\min_{W} \widehat{\mathcal{R}}_n(W) := \widehat{\mathcal{L}}_n\left(W\right) + \Omega_\lambda\left(W\right), \quad \lambda \in \mathbb{R}_+. \tag{5}$$

dynamics Eq. 3 can relax the irrepresentable conditions for model selection by Lasso (Huang et al., 2016), which can be violated for highly correlated weight parameters. The weight $W$, instead of directly being imposed with $\ell_1$-sparsity, adopts $\ell_2$-regularization in the proximity of the sparse path of $\Gamma$ that admits simultaneously exploring highly correlated parameters in over-parameterized models and sparse regularization.

The key insight lies in that differential inclusion of Eq. 3c drives the important features in $\Gamma_t$ that earlier reaches the $\ell_\infty$-unit box to be selected earlier. Hence, the importance of features is related to the "time scale" of dynamic hitting time to the $\ell_\infty$ unit box, and such a time scale is inversely proportional to lasso regularization parameter $\lambda = 1/t$ (Osher et al., 2016). Such a differential inclusion is firstly studied in (Burger et al., 2006) with Total-Variation sparsity for image reconstruction, where important features in early dynamics are coarse-grained shapes with fine details appeared later. This is in contrast to wavelet scale space that coarse-grained features appear in large scale spaces, thus named "inverse scale space". In this paper, we shall see that Eq. 3 inherits such an inverse scale space property empirically even for the highly nonconvex neural network training. Figure 1 shows a LeNet trained on MNIST by the discretized dynamics, where important sparse filters are selected in early epochs while the popular SGD returns dense filters.

**Deep Structural Splitting Linearized Bregman Iteration.** Eq. 3 admits an extremely simple discrete approximation, using Euler forward discretization of dynamics and called DessiLBI in the sequel:

$$W_{k+1} = W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}\left(W_k, \Gamma_k\right), \tag{6a}$$

$$V_{k+1} = V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}\left(W_k, \Gamma_k\right), \tag{6b}$$

$$\Gamma_{k+1} = \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}\left(V_{k+1}\right), \tag{6c}$$

where $V_0 = \Gamma_0 = 0$, $W_0$ can be small random numbers such as Gaussian initialization. For some complex networks, it can be initialized as common setting. The proximal map in Eq. (6c) that controls the sparsity of $\Gamma$,

$$\mathrm{Prox}_{\Omega_\lambda}(V) = \arg\min_{\Gamma} \left\{ \frac{1}{2}\|\Gamma - V\|_2^2 + \Omega_\lambda\left(\Gamma\right) \right\}, \tag{7}$$

Such an iterative procedure returns a sequence of sparse networks from simple to complex ones whose global convergence condition to be shown below, while solving Eq. (5) at various levels of $\lambda$ might not be tractable, especially for over-parameterized networks.

Our DessiLBI explores structural sparsity in fully connected and convolutional layers, which can be unified in framework of group lasso penalty, $\Omega_1(\Gamma) = \sum_g \|\Gamma^g\|_2$, where $\|\Gamma^g\|_2 = \sqrt{\sum_{i=1}^{|\Gamma^g|}\left(\Gamma_i^g\right)^2}$ and $|\Gamma^g|$ is the number of weights in $\Gamma^g$. Thus Eq. (6c) has a closed form solution $\Gamma^g = \kappa \cdot \max\left(0, 1 - 1/\|V^g\|_2\right) V^g$. Typically,

(1) For a convolutional layer, $\Gamma^g = \Gamma^g(c_{in}, c_{out}, \texttt{size})$ denote the convolutional filters where $\texttt{size}$ denotes the kernel size and $c_{in}$ and $c_{out}$ denote the numbers of input channels and output channels, respectively. When we regard each group as each convolutional filter, $g = c_{out}$; otherwise for

weight sparsity, $g$ can be every element in the filter that reduces to the Lasso.
(2) For a fully connected layer, $\Gamma = \Gamma(c_{in}, c_{out})$ where $c_{in}$ and $c_{out}$ denote the numbers of inputs and outputs of the fully connected layer. Each group $g$ corresponds to each element $(i, j)$, and the group Lasso penalty degenerates to the Lasso penalty.

In addtion, we can take the group of incoming weights $\Gamma^g = \Gamma^g(c_{in}, g)$ denoting the incoming weights of the $g$-th neuron of fc layers. This will be explored in future work.

## 4 Global Convergence of DessiLBI

We present a theorem that guarantees the *global convergence* of DessiLBI, *i.e.* from any intialization, the DessiLBI sequence converges to a critical point of $\bar{\mathcal{L}}$. Our treatment extends the block coordinate descent (BCD) studied in (Zeng et al., 2019a), with a crucial difference being the mirror descent involved in DessiLBI. Instead of the splitting loss in BCD, a new Lyapunov function is developed here to meet the Kurdyka-Łojasiewicz property (Łojasiewicz, 1963). (Xue & Xin, 2018) studied convergence of variable splitting method for single hidden layer networks with Gaussian inputs.

Let $P := (W, \Gamma)$. Following (Huang & Yao, 2018), the DessiLBI algorithm in Eq. (6a-6c) can be rewritten as the following standard Linearized Bregman Iteration,

$$P_{k+1} = \arg\min_P \left\{ \langle P - P_k, \alpha \nabla \bar{\mathcal{L}}(P_k) \rangle + B_\Psi^{p_k}(P, P_k) \right\} \tag{8}$$

where

$$\begin{aligned} \Psi(P) &= \Omega_\lambda(\Gamma) + \frac{1}{2\kappa}\|P\|_2^2 \\ &= \Omega_\lambda(\Gamma) + \frac{1}{2\kappa}\|W\|_2^2 + \frac{1}{2\kappa}\|\Gamma\|_2^2, \end{aligned} \tag{9}$$

$p_k \in \partial\Psi(P_k)$, and $B_\Psi^q$ is the Bregman divergence associated with convex function $\Psi$, defined by

$$B_\Psi^q(P, Q) := \Psi(P) - \Psi(Q) - \langle q, P - Q \rangle. \tag{10}$$

for some $q \in \partial\Psi(Q)$. Without loss of generality, consider $\lambda = 1$ in the sequel. One can establish the global convergence of DessiLBI under the following assumptions.

**Assumption 1.** *Suppose that: (a) $\widehat{\mathcal{L}}_n(W) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, \Phi_W(x_i))$ is continuous differentiable and $\nabla\widehat{\mathcal{L}}_n$ is Lipschitz continuous with a positive constant Lip; (b) $\widehat{\mathcal{L}}_n(W)$ has bounded level sets; (c) $\widehat{\mathcal{L}}_n(W)$ is lower bounded (without loss of generality, we assume that the lower bound is 0); (d) $\Omega$ is a proper lower semi-continuous convex function and has locally bounded subgradients, that is, for every compact set $\mathcal{S} \subset \mathbb{R}^n$, there exists a constant $C > 0$ such that for all $\Gamma \in \mathcal{S}$ and all $g \in \partial\Omega(\Gamma)$, there holds $\|g\| \le C$; and (e) the Lyapunov function*

$$F(P, \tilde{g}) := \alpha\bar{\mathcal{L}}(W, \Gamma) + B_\Omega^{\tilde{g}}(\Gamma, \tilde{\Gamma}), \tag{11}$$

*is a Kurdyka-Łojasiewicz function on any bounded set, where $B_\Omega^{\tilde{g}}(\Gamma, \tilde{\Gamma}) := \Omega(\Gamma) - \Omega(\tilde{\Gamma}) - \langle \tilde{g}, \Gamma - \tilde{\Gamma} \rangle$, $\tilde{\Gamma} \in \partial\Omega^*(\tilde{g})$, and $\Omega^*$ is the conjugate of $\Omega$ defined as*

$$\Omega^*(g) := \sup_{U \in \mathbb{R}^n} \{ \langle U, g \rangle - \Omega(U) \}.$$

**Remark 1.** *Assumption 1 (a)-(c) are regular in the analysis of nonconvex algorithm (see, (Attouch et al., 2013) for instance), while Assumption 1 (d) is also mild including all Lipschitz continuous convex function over a compact set. Some typical examples satisfying Assumption 1(d) are the $\ell_1$ norm, group $\ell_1$ norm, and every continuously differentiable penalties. By Eq. (11) and the definition of conjugate, the Lyapunov function $F$ can be rewritten as follows,*

$$F(W, \Gamma, g) = \alpha\bar{\mathcal{L}}(W, \Gamma) + \Omega(\Gamma) + \Omega^*(g) - \langle \Gamma, g \rangle. \tag{12}$$

Now we are ready to present the main theorem.

**Theorem 1.** *[Global Convergence of DessiLBI] Suppose that Assumption 1 holds. Let $(W_k, \Gamma_k)$ be the sequence generated by DessiLBI (Eq. (6a-6c)) with a finite initialization. If*

$$0 < \alpha_k = \alpha < \frac{2}{\kappa(Lip + \nu^{-1})},$$

*then $(W_k, \Gamma_k)$ converges to a critical point of $\bar{\mathcal{L}}$ defined in Eq. (4), and $\{W^k\}$ converges to a critical point of $\widehat{\mathcal{L}}_n(W)$.*

Applying to the neural networks, typical examples are summarized in the following corollary.

**Corollary 1.** *Let $\{W_k, \Gamma_k, g_k\}$ be a sequence generated by DessiLBI (18a-18c) for neural network training where (a) $\ell$ is any smooth definable loss function, such as the square loss ($t^2$), exponential loss ($e^t$), logistic loss $\log(1+e^{-t})$, and cross-entropy loss; (b) $\sigma_i$ is any smooth definable activation, such as linear activation ($t$), sigmoid ($\frac{1}{1+e^{-t}}$), hyperbolic tangent ($\frac{e^t - e^{-t}}{e^t + e^{-t}}$), and softplus ($\frac{1}{c}\log(1+e^{ct})$ for some $c > 0$) as a smooth approximation of ReLU; (c) $\Omega$ is the group Lasso. Then the sequence $\{W_k\}$ converges to a stationary point of $\widehat{\mathcal{L}}_n(W)$ under the conditions of Theorem 1.*

## 5 Experiments

This section introduces some stochastic variants of DessiLBI, followed by four set of experiments revealing the insights of DessiLBI exploring structural sparsity of deep networks.

**Batch DessiLBI**. To train networks on large datasets, stochastic approximation of the gradients in DessiLBI over the mini-batch $(\mathbf{X}, \mathbf{Y})_{\text{batch}_t}$ is adopted to update the parameter $W$,

$$\widetilde{\nabla}_W^t = \nabla_W \widehat{\mathcal{L}}_n(W) \mid_{(\mathbf{X}, \mathbf{Y})_{\text{batch}_t}}. \tag{13}$$

**DessiLBI with momentum (Mom)**. Inspired by the variants of SGD, the momentum term can be also incorporated to the standard DessiLBI that leads to the following updates of $W$ by replacing Eq (6a) with,

$$v_{t+1} = \tau v_t + \widetilde{\nabla}_W \bar{\mathcal{L}}(W_t, \Gamma_t) \quad (14a)$$
$$W_{t+1} = W_t - \kappa \alpha v_{t+1} \quad (14b)$$

where $\tau$ is the momentum factor, empirically setting as 0.9. **DessiLBI with momentum and weight decay (Mom-Wd).** The update formulation is ($\beta = 1e^{-4}$)

$$v_{t+1} = \tau v_t + \widetilde{\nabla}_W \bar{\mathcal{L}}(W_t, \Gamma_t) \quad (15)$$
$$W_{t+1} = W_t - \kappa \alpha v_{t+1} - \beta W_t \quad (16)$$

**Implementation.** Experiments are conducted over various backbones, *e.g.*, LeNet, AlexNet, VGG, and ResNet. For MNIST and Cifar-10, the default hyper-parameters of DessiLBI are $\kappa = 1$, $\nu = 10$ and $\alpha_k$ is set as 0.1, decreased by 1/10 every 30 epochs. In ImageNet-2012, the DessiLBI utilizes $\kappa = 1$, $\nu = 1000$, and $\alpha_k$ is initially set as 0.1, decays 1/10 every 30 epochs. We set $\lambda = 1$ in Eq. (7) by default, unless otherwise specified. On MNIST and Cifar-10, we have batch size as 128; and for all methods, the batch size of ImageNet 2012 is 256. The standard data augmentation implemented in pytorch is applied to Cifar-10 and ImageNet-2012, as (He et al., 2016). The weights of all models are initialized as (He et al., 2015). In the experiments, we define *sparsity* as percentage of non-zero parameters, *i.e.*, the number of non-zero weights dividing the total number of weights in consideration. Runnable codes can be downloaded[1].

### 5.1 Image Classification

**Settings**. We compare different variants of SGD and Adam in the experiments. By default, the learning rate of competitors is set as 0.1 for SGD and its variant and 0.001 for Adam and its variants, and gradually decreased by 1/10 every 30 epochs. (1) Naive SGD: the standard SGD with batch input. (2) SGD with $l_1$ penalty (Lasso). The $l_1$ norm is applied to penalize the weights of SGD by encouraging the sparsity of learned model, with the regularization parameter of the $l_1$ penalty term being set as $1e^{-3}$ (3) SGD with momentum (Mom): we utilize momentum 0.9 in SGD. (4) SGD with momentum and weight decay (Mom-Wd): we set the momentum 0.9 and the standard $l_2$ weight decay with the coefficient weight $1e^{-4}$. (5) SGD with Nesterov (Nesterov): the SGD uses nesterov momentum 0.9. (6) Naive Adam: it refers to standard Adam[2].

The results of image classification are shown in Tab. 1. Our DessiLBI variants may achieve comparable or even better performance than SGD variants in 100 epochs, indicating the efficacy in learning dense, over-parameterized models.

| Dataset | | ImageNet-2012 | |
|---|---|---|---|
| Models | Variants | AlexNet | ResNet-18 |
| *SGD* | Naive | –/– | 60.76/79.18 |
| | $l_1$ | 46.49/65.45 | 51.49/72.45 |
| | Mom | 55.14/78.09 | 66.98/86.97 |
| | Mom-W$d^{\star}$ | 56.55/79.09 | 69.76/89.18 |
| | Nesterov | -/- | 70.19/89.30 |
| *Adam* | Naive | –/– | 59.66/83.28 |
| *DessiLBI* | Naive | 55.06/77.69 | 65.26/86.57 |
| | Mom | 56.23/78.48 | 68.55/87.85 |
| | Mom-Wd | **57.09/79.86** | **70.55/89.56** |

*Table 1.* Top-1/Top-5 accuracy(%) on ImageNet-2012. $^{\star}$: results from the official pytorch website. We use the official pytorch codes to run the competitors. More results on MNIST/Cifar-10, please refer Tab. 2 in supplementary.

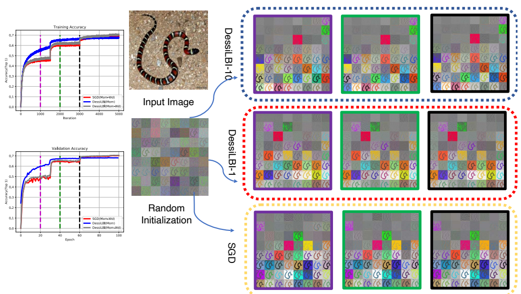



*Figure 2.* Visualization of the first convolutional layer filters of ResNet-18 trained on ImageNet-2012. Given the input image and initial weights visualized in the middle, filter response gradients at 20 (purple), 40 (green), and 60 (black) epochs are visualized by (Springenberg et al., 2014). The "DessiLBI-10" ("DessiLBI-1") in the right figure refers to DessiLBI with $\kappa = 10$ and $\kappa = 1$, respectively. Please refer to Fig. 6 in the Appendix for larger size figure.

### 5.2 Learning Sparse Filters for Interpretation

In DessiLBI, the structural sparsity parameter $\Gamma_t$ explores important sub-network architectures that contributes significantly to the loss or error reduction in early training stages. Through the $\ell_2$-coupling, structural sparsity parameter $\Gamma_t$ may guide the weight parameter to explore those sparse models in favour of improved interpretability. Figure 1 visualizes some sparse filters learned by DessiLBI of LeNet-5 trained on MNIST (with $\kappa = 10$ and weight decay every 40 epochs), in comparison with dense filters learned by SGD. The activation pattern of such sparse filters favours high order global correlations between pixels of input images. To further reveal the insights of learned patterns of DessiLBI, we visualize the first convolutional layer of ResNet-18 on ImageNet-2012 along the training path of our DessiLBI as in Fig. 2. The left figure compares the training and val-

[1]https://github.com/DessiLBI2020/DessiLBI

[2]In the Appendix of Tab. 2, we further give more results for Adabound, Adagrad, Amsgrad, and Radam, which, we found, are difficulty trained on ImageNet-2012 in practice.

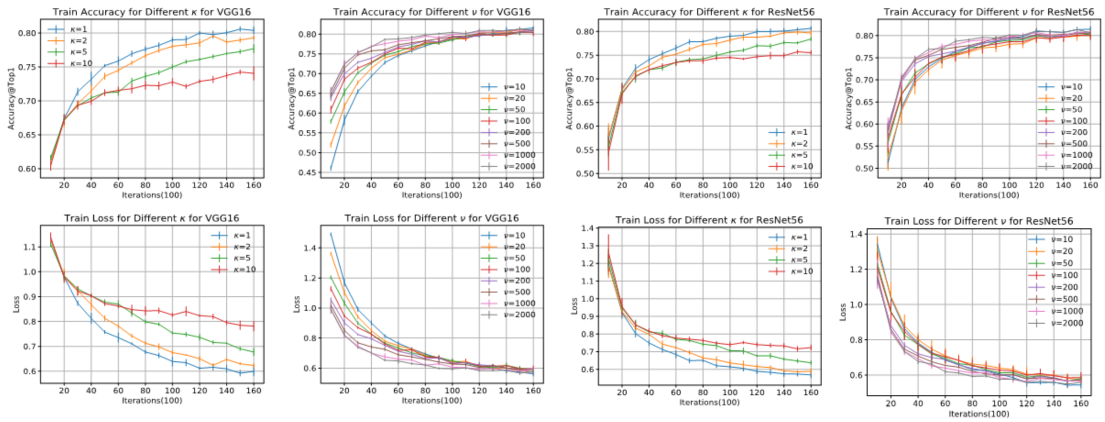



*Figure 3.* Training loss and accuracy curves at different $\kappa$ and $\nu$. The X-axis and Y-axis indicate the training epochs, and loss/accuracy. The results are repeated for 5 rounds, by keeping the exactly same initialization for each model. In each round, we use the same initialization for every hyperparameter. For all models, we train for 160 epochs with initial learning rate (lr) of 0.1 and drop by 0.1 at epoch 80 and 120.

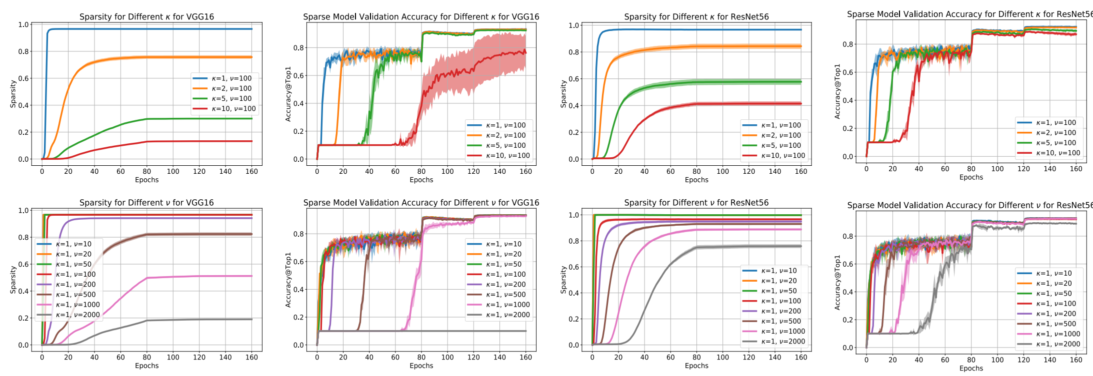
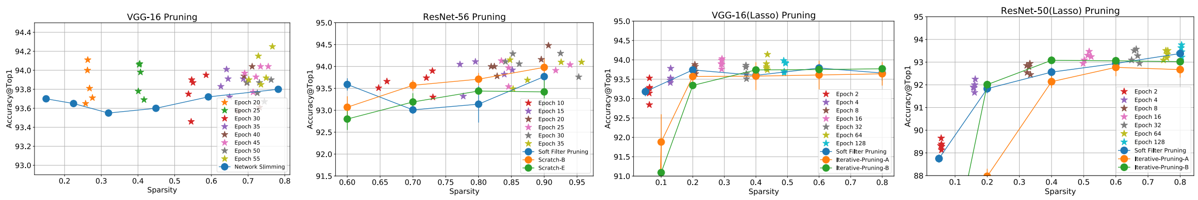



*Figure 4.* Sparsity and validation accuracy by different $\kappa$ and $\nu$ show that moderate sparse models may achieve comparable test accuracies to dense models without fine-tuning. Sparsity is obtained as the percentage of nonzeros in $\Gamma_t$ and sparse model at epoch $t$ is obtained by projection of $W_t$ onto the support set of $\Gamma_t$, i.e. pruning the weights corresponding to zeros in $\Gamma_t$. The best accuracies achieved are recorded in comparison with full networks in Tab. 3 and 5 of Appendix for different $\kappa$ and $\nu$, respectively. X-axis and Y-axis indicate the training epochs, and sparsity/accuracy. The results are repeated for 5 times. Shaded area indicates the variance; and in each round, we keep the exactly same initialization for each model. In each round, we use the same initialization for every hyperparameter. For all the model, we train for 160 epochs with initial learning rate (lr) of 0.1 and decrease by 0.1 at epoch 80 and 120.

idation accuracy of DessiLBI and SGD. The right figure compares visualizations of the filters learned by DessiLBI and SGD.

**Visualization.** To be specific, denote the weights of an $l$-layer network as $\{W^1, W^2, \cdots, W^l\}$. For the $i-$th layer weights $W^i$, denote the $j-$th channel $W^i_j$. Then we compute the gradient of the sum of the feature map computed from each filter $W^i_j$ with respect to the input image (here a snake image). We further conduct the min-max normalization to the gradient image, and generate the final visualization map. The right figure compares the visualized gradient images of first convolutional layer of 64 filters with $7 \times 7$ receptive fields. We visualize the models parameters at 20 (purple), 40 (green), and 60 (black) epochs, respectively, which corresponds to the bounding boxes in the right figure annotated by the corresponding colors, *i.e.*, purple, green, and black. We order the gradient images produced from 64 filters by the descending order of the magnitude ($\ell_2$-norm) of filters, *i.e.*, images are ordered from the upper left to the bottom right. For comparison, we also provide the visualized gradient from random initialized weights.

**DessiLBI learns sparse filters for improved interpreation**. Filters learned by ImageNet prefer to non-semantic texture rather than shape and color. The filters of high norms mostly focus on the texture and shape information, while color information is with the filters of small magnitudes. This phenomenon is in accordance with observation of (Abbasi-Asl & Yu, 2017) that filters mainly of




(a) VGG-16 (b) ResNet-56 (c) VGG-16 (Lasso) (d) ResNet-50 (Lasso)

*Figure 5.* DessiLBI with early stopping finds sparse subnets whose test accuracies (stars) after retrain are comparable or even better than the baselines (Network Slimming (reproduced by the released codes from (Liu et al., 2019) ) , Soft-Filter Pruning (Tab. 10), Scratch-B (Tab. 10), Scratch-E (Tab. 10), and "Rethinking-Lottery" (Tab. 9a) as reported in (Liu et al., 2019), Iterative-Pruning-A (Han et al., 2015) and Iterative-Pruning-B (Zhu & Gupta, 2017) (reproduced based on our own implementation)). Sparse filters of VGG-16 and ResNet-56 are show in (a) and (b), while sparse weights of VGG-16 and ResNet-50 are shown in (c) and (d).

color information can be pruned for saving computational cost. Moreover, among the filters of high magnitudes, most of them capture non-semantic textures while few pursue shapes. This shows that the first convolutional layer of ResNet-18 trained on ImageNet learned non-semantic textures rather than shape to do image classification tasks, in accordance with recent studies (Geirhos et al., 2019). How to enhance the semantic shape invariance learning, is arguably a key to improve the robustness of convolutional neural networks.

### 5.3 Training Curves and Structural Sparsity at $(\kappa, \nu)$

On Cifar-10, we use VGG-16 and ResNet-56 to show the influence of hyperparameters ($\kappa$ and $\nu$) on: (i) training curves (loss and accuracies); and (ii) structural sparsity learned by $\Gamma_t$.

**Implementation.** We use DessiLBI with momentum and weight decay, due to the good results in Sec. 5.1. Specifically, we have these experiments, repeated for 5 times: (1) we fix $\nu = 100$ and vary $\kappa = 1, 2, 5, 10$, where training curves of $W_t$ are shown in Fig. 3, sparsity of $\Gamma_t$ and validation accuracies of sparse models are shown in top row of Fig. 4. Note that we keep $\kappa \cdot \alpha_k = 0.1$ in Eq (3a), to make comparable learning rate of each variant, and also consistent with SGD. Thus $\alpha_k$ will be adjusted by different $\kappa$. (2) we fix $\kappa = 1$, and change $\nu = 10, 20, 50, 100, 200, 500, 1000, 2000$ as in Fig. 3 and the second row of Fig. 4 ($\alpha_k = 0.1$)[3].

**Influence of $\kappa$ and $\nu$ on training curves**. Training loss $(\widehat{\mathcal{L}}_n)$ and accuracies in Fig. 3 converge at different speeds when $\kappa$ and $\nu$ changes. In particular, larger $\kappa$ cause slower convergence, agreeing with the convergence rate in inverse proportion to $\kappa$ suggested in Lemma A.5. Increasing $\nu$ however leads to faster convergence in early epochs, with the advantage vanishing eventually.

**DessiLBI finds good sparse structure.** Sparse subnetworks achieve comparable performance to dense models without fine-tuning or retraining. In Fig. 4, the sparsity of $\Gamma$ grows as $\kappa$ and $\nu$ increase. While large $\kappa$ may cause a small number of important parameters growing rapidly, large $\nu$ will decouple $W_t$ and $\Gamma_t$ such that the growth of $W_t$ does not affect $\Gamma_t$ that may over-sparsify and deteriorate model accuracies. Thus a moderate choice of $\kappa$ and $\nu$ is preferred in practice. In Fig. 4, Tab. 3 and 5 in Appendix, one can see that moderate sparse models may achieve comparable predictive power to dense models, even without fine-tuning or retraining. This shows that structural sparsity parameter $\Gamma_t$ can indeed capture important weight parameter $W_t$ through their coupling.

### 5.4 Effective Subnetworks by Early Stopping

With early stopping, $\Gamma_t$ in early epochs may learn effective subnetworks (*i.e.* "winning tickets" (Frankle & Carbin, 2019)) that after retraining achieve comparable or even better performance than existing pruning strategies by SGD.

**Settings.** On Cifar-10, we adopt one-shot pruning strategy with the backbones of VGG–16, ResNet-50, and ResNet-56 as (Frankle & Carbin, 2019), which firstly trains a dense over-parameterized model by SGD for $T = 160$ epochs and find the sparse structure by pruning weights or filters (Liu et al., 2019), then secondly retrains the structure from the scratch with $T$ epochs from the same initialization as the first step. For DessiLBI, instead of pruning weights/filters from dense models, we directly utilize structural sparsity $\Gamma_t$ at different training epochs to define the subnet architecture, followed by retrain-from-scratch[4]. In particular, we set $\lambda = 0.1$, and $0.05$ for VGG-16, and ResNet-56 respectively, since ResNet-56 has less parameters than VGG-16. We further introduce another variant of our DessiLBI by using Lasso rather than group lasso penalty for $\Gamma_t$ to sparsify the weights of convolutional filters[5], denoting as VGG-16 (Lasso) and ResNet-50 (Lasso), individually. The results are reported over five rounds, as in Fig. 5 Note that in different runs of DessiLBI, the sparsity of $\Gamma_t$ slightly varies.

[3]Figure. 7 in Appendix shows validation accuracies of full models learned by $W_t$.

[4]Preliminary results of fine-tuning is in Appendix Sec. D.

[5]DessiLBI uses momentum and weight decay with hyperparameters shown in Tab. 11 in Appendix.

**Sparse subnets found by early stopping of DessiLBI is effective.** It achieves remarkably good accuracy after retrain from scratch. In Fig.5 (a-b), sparse filters discovered by $\Gamma_t$ at different epochs are compared against the methods of Network Slimming (Liu et al., 2017), Soft Filter Pruning (Yang et al., 2018), Scratch-B, and Scratch-E, whose results are reported from (Liu et al., 2019). At similar sparsity levels, DessiLBI can achieve comparable or even better accuracy than competitors, even with sparse architecture learned from very early epochs (e.g. $t = 20$ or 10). Moreover in Fig.5 (c-d), we can draw the same conclusion for the sparse weights of VGG-16 (Lasso) and ResNet-50 (Lasso), against the results reported in (Liu et al., 2019), Iterative-Pruning-A (Han et al., 2015) and Iterative-Pruning-B (Zhu & Gupta, 2017) (reproduced based on our own implementation)) . These results shows that structural sparsity $\Gamma_t$ found by early stopping of DessiLBI already discloses important subnetwork that may achieve remarkably good accuracy after retraining from scratch. Therefore, it is not necessary to fully train a dense model to find a successful sparse subnet architecture with comparable performance to the dense ones, *i.e.*, one can early stop DessiLBI properly where the structural parameter $\Gamma_t$ unveils "*winning tickets*" (Frankle & Carbin, 2019).

## 6 Conclusion

This paper presents a novel algorithm – DessiLBI in exploring structural sparsity of deep network. It is derived from differential inclusions of inverse scale space, with a proven global convergence to KKT points from arbitrary initializations. Extensive experiments reveal the effectiveness of our algorithm in training over-parameterized models and exploring effective sparse architecture of deep models.

## Acknowledgement

The authors would like to thank helpful discussions with Yizhou Wang, Bao Wang, Stanley Osher, Jack Xin, and Wotao Yin. This work was supported in part by NSFC Projects (61977038), Science and Technology Commission of Shanghai Municipality Projects (19511120700, 19ZR1471800), and Shanghai Research and Innovation Functional Program (17DZ2260900). Dr. Zeng is supported by the Two Thousand Talents Plan of Jiangxi Province. The research of Yuan Yao was supported in part by Hong Kong Research Grant Council (HKRGC) grant 16303817, ITF UIM/390, as well as awards from Tencent AI Lab, Si Family Foundation, and Microsoft Research-Asia.

## References

Abbasi-Asl, R. and Yu, B. Structural compression of convolutional neural networks based on greedy filter pruning. *arXiv preprint arXiv:1705.07356*, 2017. 1, 5.2

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. 2018. arXiv:1811.03962. 1

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018. 1

Attouch, H., Bolte, J., and Svaiter, B. F. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137:91–129, 2013. 1, A.1, A.3

Azizan, N., Lale, S., and Hassibi, B. Stochastic mirror descent on overparameterized nonlinear models: Convergence, implicit regularization, and generalization. *arXiv preprint arXiv:1906.03830*, 2019. 2

Bartlett, P., Foster, D. J., and Telgarsky, M. Spectrally-normalized margin bounds for neural networks. In *The 31st Conference on Neural Information Processing Systems*. 2017. 1

Bartlett, P. L. For valid generalization the size of the weights is more important than the size of the network. In Mozer, M. C., Jordan, M. I., and Petsche, T. (eds.), *Advances in Neural Information Processing Systems 9*, pp. 134–140. MIT Press, 1997. 1

Beck, A. and Teboulle, M. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003. 2

Benning, M., Betcke, M. M., Ehrhardt, M. J., and Schönlieb, C.-B. Choose your path wisely: gradient descent in a bregman distance framework. *arXiv preprint arXiv:1712.04045*, 2017. A.3

Bochnak, J., Coste, M., and Roy, M.-F. *Real algebraic geometry*, volume 3. Ergeb. Math. Grenzgeb. Springer-Verlag, Berlin, 1998. 3, A.1

Bolte, J., Daniilidis, A., and Lewis, A. The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. *SIAM Journal on Optimization*, 17:1205–1223, 2007a. A.1, A.1

Bolte, J., Daniilidis, A., Lewis, A., and Shiota, M. Clark subgradients of stratifiable functions. *SIAM Journal on Optimization*, 18:556–572, 2007b. A.1, A.1, A.2, A.2

Bottou, L. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, 2010. 1

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011. 2

Burger, M., Gilboa, G., Osher, S., and Xu, J. Nonlinear inverse scale space methods. *Communications in Mathematical Sciences*, 4(1):179–212, 2006. 3

Cai, J.-F., Osher, S., and Shen, Z. Convergence of the linearized bregman iteration for l1-norm minimization. *Mathematics of Computation*, 2009. 2

Collins, M. and Kohli, P. Memory bounded deep convolutional networks. In *arXiv preprint arXiv:1412.1442, 2014*, 2014. 1

Coste, M. *An introduction to o-minimal geometry*. RAAG Notes, 81 pages, Institut de Recherche Mathematiques de Rennes, 1999. A.2

Donoho, D. L. and Huo, X. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001. 1

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. 2018. arXiv:1811.03804. 1

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. Visualizing higher-layer features of a deep network. *University of Montreal, Technical Report*, 1341, 2009. 1

Franca, G., Robinson, D. P., and Vidal, R. Admm and accelerated admm as continuous dynamical systems. *arXiv preprint arXiv:1805.06579*, 2018. 2

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *International Conference on Learning Representations*, 2019. arXiv preprint arXiv:1803.03635. 1, 5.4, D

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. The lottery ticket hypothesis at scale. *arXiv preprint arXiv:1903.01611*, 2019. D

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *International Conference on Learning Representations*, 2019. arXiv preprint arXiv:1811.12231. 5.2

Ghadimi, S. and Lan, G. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework. *SIAM Journal on Optimization*, 22(4):1469–1492, 2012. 2

Golowich, N., Rakhlin, A., and Shamir, O. Size-independent sample complexity of neural networks. *Conference on Learning Theory*, 2018. arXiv preprint arXiv:1712.06541. 1

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, 2015. 1, 5, 5.4

He, B. and Yuan, X. On the o(1/n) convergence rate of the douglas–rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012. 2

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, 2015. 5

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 5

Huang, C. and Yao, Y. A unified dynamic approach to sparse model selection. In *The 21st International Conference on Artificial Intelligence and Statistics*, Lanzarote, Spain, 2018. 2, 4

Huang, C., Sun, X., Xiong, J., and Yao, Y. Split lbi: An iterative regularization path with structural sparsity. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, pp. 3369–3377. 2016. 2, 3

Huang, C., Sun, X., Xiong, J., and Yao, Y. Boosting with structural sparsity: A differential inclusion approach. *Applied and Computational Harmonic Analysis*, 2018. arXiv preprint arXiv:1704.04833. 2

Jaderberg, M., Vedaldi, A., and Zisserman, A. Speeding up convolutional neural networks with low rank expansions. In *BMVC*, 2014. 1

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 1

Krantz, S. and Parks, H. R. *A primer of real analytic functions*. Birkhäuser, second edition, 2002. 2, A.2

Krichene, W., Bayen, A., and Bartlett, P. L. Accelerated mirror descent in continuous and discrete time. In *Advances in neural information processing systems*, pp. 2845–2853, 2015. 2

Kurdyka, K. On gradients of functions definable in o-minimal structures. *Annales de l'institut Fourier*, 48: 769–783, 1998. A.1, A.1, A.2, A.2

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017. 1

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. Learning efficient convolutional networks through network slimming. In *IEEE International Conference on Computer Vision*, 2017. 5.4

Liu, Z., Sun, M., Zhou, T., Huang, G., and Darrell, T. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019. 1, 5, 5.4

Łojasiewicz, S. Une propriété topologique des sous-ensembles analytiques réels. In: Les Équations aux dérivées partielles. *Éditions du centre National de la Recherche Scientifique, Paris*, pp. 87–89, 1963. 4, A.1

Łojasiewicz, S. *Ensembles semi-analytiques*. Institut des Hautes Etudes Scientifiques, 1965. A.1

Łojasiewicz, S. Sur la geometrie semi-et sous-analytique. *Annales de l'institut Fourier*, 43:1575–1595, 1993. A.1

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019. 1

Mei, S., Montanari, A., and Nguyen, P.-M. A mean field view of the landscape of two-layers neural network. *Proceedings of the National Academy of Sciences (PNAS)*, 2018. 1

Mei, S., Misiakiewicz, T., and Montanari, A. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. *Conference on Learning Theory*, 2019. 1

Mordukhovich, B. S. *Variational analysis and generalized differentiation I: Basic Theory*. Springer, 2006. A.1

Nedic, A. and Lee, S. On stochastic subgradient mirror-descent algorithm with weighted averaging. *SIAM Journal on Optimization*, 24(1):84–107, 2014. 2

Nemirovski, A. Tutorial: Mirror descent algorithms for large-scale deterministic and stochastic convex optimization. 2

Nemirovski, A. and Yudin, D. *Problem complexity and Method Efficiency in Optimization*. New York: Wiley, 1983. Nauka Publishers, Moscow (in Russian), 1978. 2, 2

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*. 2019. 1

Osher, S., Burger, M., Goldfarb, D., Xu, J., and Yin, W. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, 4 (2):460–489, 2005. 2

Osher, S., Ruan, F., Xiong, J., Yao, Y., and Yin, W. Sparse recovery via differential inclusions. *Applied and Computational Harmonic Analysis*, 2016. 2, 3

Rockafellar, R. T. and Wets, R. J.-B. *Variational analysis*. Grundlehren Math. Wiss. 317, Springer-Verlag, New York, 1998. A.1

Shiota, M. *Geometry of subanalytic and semialgebraic sets*, volume 150 of *Progress in Mathematics*. Birkhäuser, Boston, 1997. A.1

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 2, 6

Su, W., Boyd, S., and Candes, E. J. A differential equation for modeling nesterov's accelerated gradient method: theory and insights. *The Journal of Machine Learning Research*, 17(1):5312–5354, 2016. 2

Tropp, J. A. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50(10): 2231–2242, 2004. 1, 2

van den Dries, L. A generalization of the tarski-seidenberg theorem and some nondefinability results. *Bull. Amer. Math. Soc. (N.S.)*, 15:189–193, 1986. A.2

van den Dries, L. and Miller, C. Geometric categories and o-minimal structures. *Duke Mathematical Journal*, 84: 497–540, 1996. A.2

Venturi, L., Bandeira, A. S., and Bruna, J. Spurious valleys in two-layer neural network optimization landscapes. 2018. arXiv:1802.06384. 1

Wahlberg, B., Boyd, S., Annergren, M., and Wang, Y. An admm algorithm for a class of total variation regularized estimation problems. *IFAC Proceedings Volumes*, 45(16): 83–88, 2012. 2

Wang, H. and Banerjee, A. Online alternating direction method (longer version). *arXiv preprint arXiv:1306.3721*, 2013. 2

Wang, H. and Banerjee, A. Bregman alternating direction method of multipliers. In *Advances in Neural Information Processing Systems*, pp. 2816–2824, 2014. 2

Wang, Y., Yin, W., and Zeng, J. Global convergence of admm in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78(1):29–63, 2019. A.1

Wei, Y., Yang, F., and Wainwright, M. J. Early stopping for kernel boosting algorithms: A general analysis with localized complexities. *The 31st Conference on Neural Information Processing Systems*, 2017. 1

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. Learning the number of neurons in deep networks. In *Advances in neural information processing systems*, 2016. 1

Xue, F. and Xin, J. Convergence of a relaxed variable splitting method for learning sparse neural networks via $\ell_1$, $\ell_0$, and transformed-$\ell_1$ penalties. *arXiv:1812.05719v2*, 2018. URL http://arxiv.org/abs/1812.05719. 4

Yang, H., Kang, G., Dong, X., Fu, Y., and Yang, Y. Soft filter pruning for accelerating deep convolutional neural networks. In *International Joint Conference on Artificial Intelligence*, 2018. 5.4

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007. 1

Yin, W., Osher, S., Darbon, J., and Goldfarb, D. Bregman iterative algorithms for compressed sensing and related problems. *SIAM Journal on Imaging sciences*, 1(1):143–168, 2008. 2

Yoon, J. and Hwang, S. J. Combined group and exclusive sparsity for deep neural networks. In *International Conference on Machine Learning*, 2017. 1

Yuan, M. and Lin, Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68 (1):49–67, 2006. 1

Zeng, J., Lau, T. T.-K., Lin, S.-B., and Yao, Y. Global convergence of block coordinate descent in deep learning. In *Proceedings of the 36th International Conference on Machine Learning, Long Beach, California*, 2019a. URL https://arxiv.org/abs/1803.00225. 4, A.2

Zeng, J., Lin, S.-B., and Yao, Y. A convergence analysis of nonlinearly constrained admm in deep learning. *arXiv preprint arXiv:1902.02060*, 2019b. 2

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations*, 2017. arXiv:1611.03530. 1

Zhao, P. and Yu, B. On model selection consistency of lasso. *J. Machine Learning Research*, 7:2541–2567, 2006. 1, 2

Zhu, M. and Gupta, S. To prune, or not to prune: exploring the efficacy of pruning for model compression, 2017. 5, 5.4

Zhu, W., Huang, Y., and Yao, Y. On breiman's dilemma in neural networks: Phase transitions of margin dynamics. *arXiv:1810.03389*, 2018. 1