The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Proximal Alternating Direction Network: A Globally Converged Deep Unrolling Framework

Risheng Liu,[1,2,*] **Xin Fan,**[1,2] **Shichao Cheng,**[1,2,3] **Xiangyu Wang,**[1,2] **Zhongxuan Luo**[1,2,3]
[1]DUT-RU International School of Information Science & Engineering, Dalian University of Technology, Dalian, China
[2]Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian, China
[3]School of Mathematical Science, Dalian University of Technology, Dalian, China
{rsliu, xin.fan, zxluo}@dlut.edu.cn, shichao.cheng@outlook.com, wxy9326@gmail.com

Abstract

Deep learning models have gained great success in many real-world applications. However, most existing networks are typically designed in heuristic manners, thus lack of rigorous mathematical principles and derivations. Several recent studies build deep structures by unrolling a particular optimization model that involves task information. Unfortunately, due to the dynamic nature of network parameters, their resultant deep propagation networks do *not* possess the nice convergence property as the original optimization scheme does. This paper provides a novel proximal unrolling framework to establish deep models by integrating experimentally verified network architectures and rich cues of the tasks. More importantly, we *prove in theory* that 1) the propagation generated by our unrolled deep model globally converges to a critical-point of a given variational energy, and 2) the proposed framework is still able to learn priors from training data to generate a convergent propagation even when task information is only partially available. Indeed, these theoretical results are the best we can ask for, unless stronger assumptions are enforced. Extensive experiments on various real-world applications verify the theoretical convergence and demonstrate the effectiveness of designed deep models.

Introduction

In last years, deep models (a.k.a. deep neural networks) have produced the state-of-the-art performance in many application fields, such as image processing, object recognition, natural language processing, and bioinformatics. On the downside, these existing approaches are typically designed based on heuristic understandings of a particular problem, and trained using engineering experience to implement multi-layered feature propagations, e.g., (Krizhevsky, Sutskever, and Hinton 2012; Simonyan and Zisserman 2014; He et al. 2016). Therefore, they lack solid theoretical guidances and interpretations. More importantly, it is challenging to incorporate the mathematical rules and principles of the considered task into these existing networks.

Alternatively, several recent works, e.g., (Gregor and LeCun 2010; Schmidt and Roth 2014; Andrychowicz et al. 2016), build their networks using a specific optimization model and iteration scheme. The main idea is to unroll numerical algorithms and constitute their network architectures based on the resulted iterations. In this way, these approaches successfully incorporate the information of a predefined energy into the network propagation. Nevertheless, due to the dynamical nature of parameterized iterations, existing theoretical results, especially convergence, from the optimization area are not valid at all. Furthermore, the unrolled deep models are often with limited flexibility and adaptability as the basic architectures are restricted by the particular iteration scheme.

To partially overcome limitations in existing approaches, this work attempts to develop a simple, flexible and efficient framework to build deep models for various real-word applications. The launching point of our work is from Energy-Based Models (EBMs) (Teh et al. 2003; Zhao, Mathieu, and LeCun 2016). EBMs are a series of methods, which associate a scalar energy to each configuration of observations and their interested perditions. The inference of EBM consists of searching for a configuration of variables that minimizes the energy. In this work, we consider the following energy minimization formulation

$$\inf_{\mathbf{x}} \{\mathcal{F}(\mathbf{x}) := f(\mathbf{x};\mathbf{y}) + r(\mathbf{x})\}, \tag{1}$$

where $\mathbf{y}$ and $\mathbf{x}$ are the observed and predicted variables, respectively, r reveals the priors of predictions, and f is a measure of compatibility, i.e., fidelity, between $\mathbf{x}$ and $\mathbf{y}$.

We establish a novel proximal framework to unroll the general energy in Eq. (1), and incorporate various experimentally efficient architectures into the resulted deep model. Promising theoretical properties and practical performance will be also demonstrated. The main advantages of our proposed framework against existing optimization-unrolled deep models can be distilled to the following three points.

Insensitive Unrolling Scheme: Most existing iteration-unrolling based deep models are strictly confined to some special types of energy formulations. For example, architectures in (Schmidt and Roth 2014; Chen and Pock 2017) are deduced from the field-of-experts prior while networks in (Gregor and LeCun 2010) are based on ℓ_1-regularizations. In contrast, our unrolling strategy only depends on the separable structure and functional properties of $\mathcal{F}$, but is completely insensitive to particular forms of f and r. We can even design deep models without knowing the form of r so

*Corresponding Author.

that our framework adapts to various challenging tasks and complex data distributions.

Flexible Built-in Architecture: On the one hand, the architectures in existing unrolled networks are deduced from fixed iteration schemes, thus lacking flexibility. On the other hand, it has been revealed in many practical applications that heuristic deep architectures, e.g., ReLU and Batch Normalization, are extremely efficient though absent of theoretic analysis. Our studies show that under some mild conditions, our deep model can incorporate most existing empirically successful network architectures (even built by means of engineering tricks). In other words, we indeed theoretically offer the flexibility for existing deep architectures while taking the advantage of their efficiency.

Convergence Guarantee: A fundamental weakness underlying existing networks is the elusiveness of theoretical analysis. Especially, little to no attention has been paid to the convergence behaviors of deep models[1]. The main reason is that even building networks by unrolling converged optimization algorithms, the dynamic nature of their parameters and heuristic architectures would still break the convergent guarantee in original iteration scheme. Contrarily, this paper proves that our designed deep models do have nice convergence properties. That is, under some mild conditions, the sequence generated by the proposed proximal alternating direction networks (PADNet) can converge to a critical-point of Eq. (1) with relatively simple priors[2]. Furthermore, our propagation guarantees at least fixed-point convergence when handling complex priors, e.g., only with partial task/data information. We argue that *these theoretical results are the best we can ask for, unless stronger assumptions are enforced.*

Proximal Alternating Direction Network

In this section, we develop an alternating direction type unrolling scheme to generate the propagation sequence (denoted as $\{\mathbf{x}^k\}$) based on the energy model in Eq. (1). As shown in the following, rather than directly calculating $\mathbf{x}^{k+1}$ from $\mathbf{x}^k$, we would like to first design cascaded propagations of two auxiliary variables (denoted as $\mathbf{u}^k$ and $\mathbf{v}^k$) corresponding to the fidelity and prior of the task, respectively. The residual type deep architectures are then incorporated for subsequence updating. Finally, a novel proximal error correction process is designed to control our propagation.

Alternating Direction Scheme: For each $\mathbf{x}^k$, we introduce the Moreau-Yosida regularization (Parikh, Boyd, and others 2014; Xu, Lin, and Zha 2016) of $\mathcal{F}$ with parameter μ^k and auxiliary variable $\mathbf{u}$ to obtain the following regularized energy model

$$\begin{aligned} \mathcal{M}_{\mathcal{F}}^{\mu^k}(\mathbf{x}^k) &= \inf_{\mathbf{u}} \left\{ f_{\mathbf{x}^k}^{\mu^k}(\mathbf{u}) + r(\mathbf{u}) \right\}, \\ \text{where} \quad f_{\mathbf{x}^k}^{\mu^k}(\mathbf{u}) &:= f(\mathbf{u}) + \tfrac{\mu^k}{2}\|\mathbf{u}-\mathbf{x}^k\|^2. \end{aligned} \tag{2}$$

Now the problem is temporarily reduced to calculate $\mathbf{u}$ based on $\mathbf{x}^k$. One common inference strategy for $\mathbf{u}$ in Eq. (2) is to introduce another auxiliary variable $\mathbf{v}$ and a Lagrange multiplier $\boldsymbol{\lambda}$ and then perform alternating minimizations to the corresponding augmented Lagrange function, resulting to the following iteration scheme

$$\mathbf{u}^{k+1} = \arg\min_{\mathbf{u}} f_{\mathbf{x}^k}^{\mu^k}(\mathbf{u}) + \tfrac{\rho^k}{2}\|\mathbf{u}-(\mathbf{v}^k-\boldsymbol{\lambda}^k)\|^2, \tag{3}$$

$$\mathbf{v}^{k+1} \in \arg\min_{\mathbf{v}} r(\mathbf{v}) + \tfrac{\rho^k}{2}\|\mathbf{v}-(\mathbf{u}^{k+1}+\boldsymbol{\lambda}^k)\|^2, \tag{4}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + (\mathbf{u}^{k+1}-\mathbf{v}^{k+1}), \tag{5}$$

where ρ^k is a penalty parameter.

In this way, we actually perform the well-known Alternating Direction Method of Multiplier (ADMM) (Parikh, Boyd, and others 2014; Lin, Liu, and Su 2011) for the Moreau-Yosida regularized approximation of the original energy in Eq. (1) at each iteration.

Built-in Deep Architecture: We then show how to incorporate deep architectures into the above base iteration. Specifically, we consider a residual formulation to replace the subproblem in Eq. (4). That is, we define the propagation of $\mathbf{v}$ as[3]

$$\mathbf{v}_T = \mathcal{N}_\alpha(\mathbf{v}_0; \boldsymbol{\mathcal{W}}_T) := \mathbf{v}_0 - \alpha\left(\sum_{t=0}^{T-1} \mathcal{G}(\mathbf{v}_t; \mathcal{W}_t)\right), \tag{6}$$

where $\boldsymbol{\mathcal{W}}_T = \{\mathcal{W}_t\}_{t=0}^{T-1}$ is the set of learnable parameters, α is a step-size parameter, $\mathcal{G}$ is the basic network unit, $(\mathbf{v}_0, \mathbf{v}_T)$ are the input and output of $\mathcal{N}_\alpha$ (at T-th stage), respectively. Notice that standard training strategies can be directly adopted to optimize parameters of our basic architecture. If necessary, one may further jointly fine-tune parameters of the whole network after the design phase.

It is easy to check that the network in Eq. (6) actually *recursively* performs coordinate descent steps (i.e., $\mathbf{v}_{t+1} = \mathbf{v}_t - \alpha\mathcal{G}(\mathbf{v}_t)$) to propagate $\mathbf{v}$. So from optimization viewpoint, we interpret $\mathcal{G}$ as a descent-direction-estimation architecture for the optimization of the subproblem in Eq. (4). While in more challenging scenario (e.g., hard to define an explicit and solvable r for this subproblem), we can still learn built-in propagation architectures from training data to obtain our desired solution.

Proximal Error Correction: Now it is ready to give the formal updating scheme of $\mathbf{x}^k$. We can see that built-in architectures in Eq. (6) actually do not exactly optimize the original energy in Eq. (1). So it is necessary to introduce an additional step to control our propagation at each iteration. Specifically, denote $\mathbf{v}^{k+1}$ as the output of built-in network in

[1] Notice that the concept of "convergence" in this paper is not only related to the propagation of network parameters in the training phase, but the outputs of network architectures for both training and test phases. That is, we consider the output of the t-th basic architecture as the t-th element of a sequence, and then investigate the convergence on the resulting sequence.

[2] We will formally discuss details of relatively "simple" and "complex" priors in the following sections.

[3] Formally, we should denote the output of t-th residual unit at k-th iteration as $\mathbf{v}_t^k$. But in this paragraph, we temporarily omit the superscript k to simplify the presentation.

Eq. (6) at k-th iteration. Then we adopt a proximal-gradient-like scheme (Wang et al. 2017) to formally update $\mathbf{x}^{k+1}$

$$\begin{aligned}&\mathbf{x}^{k+1} \in \\ &\arg\min_{\mathbf{x}} r(\mathbf{x}) + \tfrac{1}{2}\|\mathbf{x} - (\mathbf{v}^{k+1} - \nabla f^{\mu^k}_{\mathbf{x}^k}(\mathbf{v}^{k+1}))\|^2 \\ &:= \mathtt{prox}_r\left(\mathbf{v}^{k+1} - \nabla f^{\mu^k}_{\mathbf{x}^k}(\mathbf{v}^{k+1})\right),\end{aligned} \tag{7}$$

where $\mathtt{prox}_r$ is Moreau's proximal operator of r.

Overall, our proposed deep model, called **P**roximal **A**lternating **D**irection **Net**work (PADNet), is summarized in Alg. 1. Notice that we actually consider PADNet in two different scenarios, which can be categorized by properties of prior regularization r in Eq. (1). That is:

- *Simple priors*: $\mathtt{prox}_r$ can be computed in closed-form.
- *Complex priors*: $\mathtt{prox}_r$ is intractable or r is unknown.

We perform error correction (i.e., Step 4 in Alg. 1) in the first case (Explicit PADNet or EPADNet for short) but directly propagate the output of built-in networks (i.e., Step 3 in Alg. 1) in the second case (Implicit PADNet or IPADNet for short). Theoretical results for these two different scenarios will be respectively proved in the next section.

Algorithm 1 Proximal Alternating Direction Network

Require: $\mathbf{u}^0$, $\mathbf{v}^0$, $\mathbf{x}^0$, $\boldsymbol{\lambda}^0$ and necessary parameters.

1: **for** $k = 0, 1, 2, \cdots, K-1$ **do**
2: $\mathbf{u}^{k+1} = \arg\min_{\mathbf{u}} f^{\mu^k}_{\mathbf{x}^k}(\mathbf{u}) + \frac{\rho^k}{2}\|\mathbf{u} - (\mathbf{v}^k - \boldsymbol{\lambda}^k)\|^2$. (Preliminary Estimation)
3: $\mathbf{v}^{k+1} = \mathcal{N}_{\alpha^k}(\mathbf{u}^{k+1} + \boldsymbol{\lambda}^k; \boldsymbol{\mathcal{W}}^k_{T_k})$. (Network Propagation)
4: $\mathbf{x}^{k+1} = \mathtt{prox}_r\left(\mathbf{v}^{k+1} - \nabla f^{\mu^k}_{\mathbf{x}^k}(\mathbf{v}^{k+1})\right)$. (Error Correction)
5: $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + (\mathbf{u}^{k+1} - \mathbf{v}^{k+1})$. (Dual Updating)
6: **end for**

Learning with Convergence Guarantee

In general, unrolling task-aware optimization schemes may incorporate rich domain-knowledge into the network structure. Unfortunately, the sequence generated by most existing unrolled deep models will no longer have convergence guarantee, even though nice theoretical results have been proved and verified for their original optimization schemes.

Fortunately, we in this work demonstrate that under some mild conditions, the propagation generated by our PADNet is globally converged[4], even with built-in network architectures designed in heuristic manners.

Convergence Behavior Analysis of PADNet

To make our paper self-contained, some necessary definitions should be presented before the formal analysis. Indeed, these concepts have been widely known in variational analysis and optimization and one may refer to (Rockafellar and Wets 2009; Attouch et al. 2010) and references therein for more details.

Definition 1 *We give necessary definitions, including proper and lower semi-continuous, coercive and semi-algebraic.*

- *A function $r : \mathbb{R}^n \to (-\infty, +\infty]$ is said to be proper and lower semi-continuous if $\mathtt{dom}(r) \neq \emptyset$, where $\mathtt{dom}(r) := \{\mathbf{x} \in \mathbb{R}^n : r(\mathbf{x}) < +\infty\}$ and $\liminf_{\mathbf{x}\to\mathbf{y}} r(\mathbf{x}) \geq r(\mathbf{y})$ at any point $\mathbf{y} \in \mathtt{dom}(r)$.*
- *A function $\mathcal{F}$ is said to be coercive, if $\mathcal{F}$ is bounded from below and $\mathcal{F} \to \infty$ if $\|\mathbf{x}\| \to \infty$, where $\|\cdot\|$ is the ℓ_2 norm.*
- *A subset S of $\mathbb{R}^n$ is a real semi-algebraic set if there exist a finit number of real polynomial functions $g_{ij}, h_{ij} : \mathbb{R}^n \to \mathbb{R}$ such that*

$$S = \bigcup_{j=1}^{p}\bigcap_{i=1}^{q}\{\mathbf{x} \in \mathbb{R}^n : g_{ij}(\mathbf{x}) = 0 \text{ and } h_{ij}(\mathbf{x}) < 0\}. \tag{8}$$

A function $r : \mathbb{R}^n \to (-\infty, \infty]$ is called semi-algebraic if its graph $\{(\mathbf{x}, z) \in \mathbb{R}^{n+1} : r(\mathbf{x}) = z\}$ is a semi-algebraic subset of $\mathbb{R}^{n+1}$.

Remark 1 *Indeed, many functions arising in learning and vision areas, including ℓ_0 norm, rational ℓ_p norms (i.e., $p = p_1/p_2$ with positive integers p_1 and p_2) used in our experimental part and their finite sums or products, are all semi-algebraic.*

In the following, we first analyze PADNet for tasks with simple priors. Specifically, given a variable $\mathbf{x}$, we estimate the discrepancy between it and the optimal solution of Eq. (2) by the function

$$\mathcal{E}^k(\mathbf{x}) := \mathbf{g}_{\mathbf{x}} + \nabla f^{\mu^k}_{\mathbf{x}^k}(\mathbf{x}), \text{ where } \mathbf{g}_{\mathbf{x}} \in \partial r(\mathbf{x}). \tag{9}$$

Here $\mathcal{E}^k$ is deduced based on the first-order optimality condition of Eq. (2) at k-th iteration. Then with the following simple error condition, we prove in Theorem 1 that the propagation of EPADNet indeed globally converges to a critical-point of Eq. (1). Please refer to supplemental materials for necessary preliminaries and all proofs of the proposed theories in this paper.

Condition 1 *(Error Condition) The error function (in Eq. (9)) at k-th iteration should satisfy $\|\mathcal{E}^k(\mathbf{x}^{k+1})\| \leq C_{\mathcal{E}}\|\mathbf{x}^{k+1} - \mathbf{x}^k\|$, where $C_{\mathcal{E}} > 0$ is a universal constant.*

Theorem 1 *(Critical-Point Convergence of Explicit PADNet) Let f be continuous differential, r be proper and lower semi-continuous and $\mathcal{F}$ be coercive. Then EPADNet converges to a critical point of Eq. (1) under Condition 1. That is, $\{\mathbf{x}^k\}$ generated by EPADNet is a bounded sequence and its any cluster point $\mathbf{x}^*$ is a critical point of Eq. (1) (i.e., satisfying $0 \in \partial\mathcal{F}(\mathbf{x}^*)$). Furthermore, if $\mathcal{F}$ is semi-algebraic, then $\{\mathbf{x}^k\}$ globally converges to a critical point of Eq. (1).*

Remark 2 *With the semi-algebraic property of $\mathcal{F}$, we can also obtain convergence rate of EPADNet based on a particular desingularizing function $\phi(s) = \frac{c}{\theta}(s)^\theta$ with a constant*

[4]Notice that "globally converged" in this paper is in the sense that the whole sequence generated by our deep model is converged and this concept has been widely used in non-convex optimization (Attouch et al. 2010) society.

$c > 0$ *and parameter* $\theta \in (0, 1]$ *(defined in (Chouzenoux, Pesquet, and Repetti 2016)). Specifically, the sequence converges after finite iterations if* $\theta = 1$. *The linear and sublinear rates can be obtained if choosing* $\theta \in [1/2, 1)$ *and* $\theta \in (0, 1/2)$, *respectively.*

It has been verified that broad class of functions arising in learning problems (even *nonconvex* and *nonsmooth*) satisfy assumptions in Theorem 1. For example, both ℓ_0 norm and rational ℓ_p norm with $p > 0$ (i.e., $p = p_1/p_2$, p_1 and p_2 are positive integers) are proper, lower semi-continuous and semi-algebraic.

Based on above analysis, we propose a learning framework (summarized in Alg. 2) to adaptively design and train globally converged deep models for different learning tasks.

Algorithm 2 The Learning Framework of PADNet

Require: $\mathbf{u}^0$, $\mathbf{v}^0$, $\mathbf{x}^0$, $\boldsymbol{\lambda}^0$, $\mathcal{G}$, $t_{\max} \geq 1$, $k_{\max} \geq 1$, $\epsilon > 0$, $C_{\mathcal{E}} > 0$, $\{\mu^k | 2C_{\mathcal{E}} < \mu^k < \infty\}$, $\{\rho^k | \rho^k = (\gamma)^k \rho^0, \rho^0 > 0, \gamma > 1\}$ ($(\gamma)^k$ denotes k-th power of γ) and $\{\alpha^k | \alpha^k = \sqrt{1/\rho^k}\}$.

1: **for** $k = 0, 1, 2, \cdots$ **do**
2: $\quad \mathbf{u}^{k+1} = \arg\min_{\mathbf{u}} f_{\mathbf{x}^k}^{\mu^k}(\mathbf{u}) + \frac{\rho^k}{2}\|\mathbf{u} - (\mathbf{v}^k - \boldsymbol{\lambda}^k)\|^2$.
3: $\quad$ **for** $t = 0, 1, 2, \cdots$ **do**
4: $\quad\quad$ Train $\mathcal{N}_{\alpha^k}$ with $\boldsymbol{\mathcal{W}}_{0:t}^k = \{\boldsymbol{\mathcal{W}}_{0:t-1}^k, \mathcal{W}_t^k\}$ (i.e., $t+1$ basic units).
5: $\quad\quad$ $\mathbf{v}_{t+1}^{k+1} = \mathcal{N}_{\alpha^k}(\mathbf{u}^{k+1} + \boldsymbol{\lambda}^k; \boldsymbol{\mathcal{W}}_{0:t}^k)$.
6: $\quad\quad$ $\mathbf{x}_{t+1}^{k+1} = \mathtt{prox}_r(\mathbf{v}_{t+1}^{k+1} - \nabla f_{\mathbf{x}^k}^{\mu^k}(\mathbf{v}_{t+1}^{k+1}))$.
7: $\quad\quad$ **if** $\|\mathcal{E}^k(\mathbf{x}_{t+1}^{k+1})\| \leq C_{\mathcal{E}}\|\mathbf{x}_{t+1}^{k+1} - \mathbf{x}^k\|$ or $t \geq t_{\max}$ **then**
8: $\quad\quad\quad$ $\mathbf{x}^{k+1} = \mathbf{x}_{t+1}^{k+1}$, $\mathbf{v}^{k+1} = \mathbf{v}_{t+1}^{k+1}$, $T_k = t$, break.
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad$ $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + (\mathbf{u}^{k+1} - \mathbf{v}^{k+1})$.
12: $\quad$ **if** $\frac{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|}{\|\mathbf{x}^k\|} \leq \epsilon$ or $k \geq k_{\max}$ **then**
13: $\quad\quad$ $K = k$, break.
14: $\quad$ **end if**
15: **end for**

Remark 3 *Theorem 1 together with Alg. 2 actually provides a flexible framework with solid theoretical guarantee for deep model design and we only need to check whether built-in networks satisfy Condition 1 during their design phase. Furthermore, in general, any architectures satisfying this condition (even designed in engineering manner) can be incorporated into our deep models.*

In contrast, when handling tasks with complex priors, neither error checking (i.e., Step 6 in Alg. 2) during design and training nor error correction (i.e., Step 4 in Alg. 1) during test will be performed. Therefore, we cannot obtain the same convergence results as that in Theorem 1. Fortunately, by enforcing another easily satisfied condition to built-in architectures, we would still prove a fixed-point convergence guarantee for IPADNet.

Condition 2 *(Architecture Condition) For any given input* $\mathbf{v}$, *the architecture* $\mathcal{N}_\alpha$ *should satisfy* $\|\mathcal{N}_\alpha(\mathbf{v}) - \mathbf{v}\| \leq C_{\mathcal{N}}\alpha$, *where* $C_{\mathcal{N}} > 0$ *is a universal constant.*

Notice that this bound condition is relatively weak and we can check that most commonly used linear and nonlinear operations in existing deep networks satisfy it.

Theorem 2 *(Fixed-Point Convergence of Implicit PADNet) Let* f *be continuous differential with bounded gradients. Then IPADNet is converged under Condition 2. That is,* $\{(\mathbf{u}^k, \mathbf{v}^k, \boldsymbol{\lambda}^k)\}$ *generated by IPADNet is a Cauchy sequence, so that is globally converged to a fixed-point.*

Remark 4 *Theorem 2 actually provides a theoretically guaranteed paradigm to fuse both analytical and empirical informations to build deep models for challenging learning tasks. That is to say, we can simultaneously design model-based fidelity function* f *to reveal our theoretical understandings of the problem and learn complex priors from training data by model-free network architecture* $\mathcal{N}_\alpha$.

To end our analysis, we emphasize that the above convergence results are the best we can ask for unless other stronger assumptions are made on the given learning task.

Implementable Error Calculation

It can be observed in Eq (9) that directly calculating $\mathcal{E}^k$ using its theoretical definition is challenging due to the subgradient term $\mathbf{g}_{\mathbf{x}}$. So we provide a calculable formulation based on the following derivations. Specifically, using Eq. (7), we have

$$\begin{aligned}\mathbf{x}^{k+1} &= \mathtt{prox}_r\left(\mathbf{v}^{k+1} - \nabla f_{\mathbf{x}^k}^{\mu^k}(\mathbf{v}^{k+1})\right)\\ &= \mathtt{prox}_r\Big(\mathbf{x}^{k+1} - \nabla f_{\mathbf{x}^k}^{\mu^k}(\mathbf{x}^{k+1}) + (\mu^k - 1)(\mathbf{x}^{k+1}\\ &\quad -\mathbf{v}^{k+1}) - \nabla f(\mathbf{v}^{k+1}) + \nabla f(\mathbf{x}^{k+1})\Big).\end{aligned} \tag{10}$$

By setting $\mathbf{e}^{k+1} = (\mu^k - 1)(\mathbf{x}^{k+1} - \mathbf{v}^{k+1}) - \nabla f(\mathbf{v}^{k+1}) + \nabla f(\mathbf{x}^{k+1})$ in Eq. (10) and following Theorem 1, we directly have that if $k \to \infty$, then

$$\mathbf{e}^{k+1} - \nabla f_{\mathbf{x}^k}^{\mu^k}(\mathbf{x}^{k+1}) \in \partial r(\mathbf{x}^{k+1}). \tag{11}$$

Therefore, we actually obtain the following implementable error calculation formulation for $\mathcal{E}^k$

$$\begin{aligned}&\mathcal{E}^k(\mathbf{x}^{k+1}) := \mathbf{e}^{k+1}\\ &= (\mu^k - 1)(\mathbf{x}^{k+1} - \mathbf{v}^{k+1}) - \nabla f(\mathbf{v}^{k+1}) + \nabla f(\mathbf{x}^{k+1}).\end{aligned} \tag{12}$$

Discussions

Intuitively, one may argue that building a deeper network should definitely result good performance. But unfortunately, many empirical evidences (Simonyan and Zisserman 2014) have suggested that the improvement cannot be trivially gained by simply adding more layers, or worse, deeper networks even suffer from a decline on performance in some applications (Shen, Lin, and Huang 2016). Therefore, it is particularly worthy of investigating the intrinsic propagation behaviors for networks with different topological structures and architectures from more solid theoretical perspective.

Indeed, our above theories have built intrinsic theoretical connections between unrolled deep models and original numerical schemes. We also investigate conditions for incorporating heuristic architectures into the proposed deep model. Therefore, the studies in this paper should provide a new perspective and introduce several powerful tools from optimization area to address the challenging but fundamental issues discussed in above paragraph.

Experiments

To verify our theoretical results and demonstrate the effectiveness of our deep models in application fields, we apply PADNet on two real-world applications, i.e., non-blind deconvolution and single image haze removal. All experiments are conduced on a PC with Intel Core i7 CPU at 3.4 GHz, 32 GB RAM and a NVIDIA GeForce GTX 1050 Ti GPU.

Non-blind Deconvolution

We first consider non-blind deconvolution, which is an important task in learning and vision areas. Specifically, given an observation $\mathbf{y}$ (e.g., image), the latent signal $\mathbf{x}$ can be processed in a filtered domain as follows (Krishnan and Fergus 2009; Schmidt and Roth 2014) $\mathcal{D}(\mathbf{y}) = \mathcal{D}(\mathbf{x}) \otimes \mathbf{k} + \boldsymbol{\varepsilon}$, where $\mathcal{D}$ is a set of filters (e.g., horizontal and vertical gradient operations), $\otimes$ denotes convolution, $\mathbf{k}$ is a point spread function and $\boldsymbol{\varepsilon}$ denotes errors/noises. This problem can be formulated as the maximum-a-posteriori estimation

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}) = \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}). \tag{13}$$

Here we follow typical choices to consider ℓ_2-fidelity (i.e., $p(\mathbf{y}|\mathbf{x}) \propto \exp(-\frac{1}{2}\|\mathcal{D}(\mathbf{x}) \otimes \mathbf{k} - \mathcal{D}(\mathbf{y})\|^2)$) and ℓ_p-regularization (i.e., $p(\mathbf{x}) \propto \exp(-\lambda\|\mathcal{D}(\mathbf{x})\|_p^p)$, $0 \leq p \leq 1$), where λ is the parameter. We adopt results in (Zuo et al. 2013) to calculate the proximal operation of general ℓ_p-minimization. In following deconvolution experiments, we always use the set of 400 images of size 180×180 built in (Chen and Pock 2017) as our training data. Two commonly used image deblurring benchmarks respectively collected by Levin *et. al.* (Levin et al. 2009) (32 blurry images of size 255×255) and Sun *et. al.* (Sun et al. 2013) (640 blurry images with 1% Gaussian noises, sizes range from 620×1024 to 928×1024) are used for testing.

Convergence Behaviors on Gradient Domain The gradient of images plays very important role in image structure analysis. Here we first consider deconvolution on gradient domain to verify the convergence behaviors of our designed deep models to a given energy with a simple prior. Specifically, the energy in gradient domain is defined as

$$\min_{\mathbf{g}} \frac{1}{2}\|\mathbf{g} \otimes \mathbf{k} - \mathcal{D}(\mathbf{y})\|^2 + \lambda\|\mathbf{g}\|_p^p, \tag{14}$$

where $\mathbf{g}$ denotes the gradient of the latent image. We first build the basic architecture $\mathcal{G}$ as cascade of two convolutions with one RBF nonlinearity (Schmidt and Roth 2014) between them. Then we perform Alg. 2 based on Eq. (14) with $p = 0, 0.8, 1$ to respectively design three EPADNet models. We also establish an IPADNet model from Alg. 2 with only the fidelity $f = \frac{1}{2}\|\mathbf{g} \otimes \mathbf{k} - \mathcal{D}(\mathbf{y})\|^2$. To compare iteration behaviors with conventional optimization strategies, we also perform popular ADMM and Half-Quadratic Splitting (HQS) (Zuo et al. 2013) algorithms on Eq. (14) with the same ℓ_p-regularizer and parameters.

The averaged convergence results of compared algorithms on Levin *et. al.*' benchmark are reported in Tab. 1. As IPDANet does not depend on ℓ_p functions, we just repeated its results for three cases (i.e., $p = 0, 0.8, 1$) in this table.

It can be seen that our designed deep models (i.e., one IPADNet and three EPADNets) need extremely less iterations but obtain more accurate estimations than conventional optimization schemes. Moreover, the performance of IPDANet is better than EPADNets regularized by ℓ_0 and ℓ_1, but a little worse than the $\ell_{0.8}$ energy. These results make sense because the prior learned from training data should perform better than the relatively improper handcrafted priors (e.g., ℓ_0 and ℓ_1 norms in this task). If the prior function can fit the data distribution well (e.g., $\ell_{0.8}$ norm here), the critical-point convergence guarantee of EPADNet will definitely result better performance, compared with the relatively weak fixed-point convergence of IPADNet.

We also plot curves of relative errors (i.e., iteration error $\|\mathbf{g}^{k+1} - \mathbf{g}^k\|/\|\mathbf{g}^k\|$ and reconstruction error $\|\mathbf{g}^k - \mathbf{g}^{\mathtt{gt}}\|/\|\mathbf{g}^{\mathtt{gt}}\|$) and error condition (referring to $\|\mathcal{E}^k(\mathbf{g}^{k+1})\|$ and $C_{\mathcal{E}}\|\mathbf{g}^{k+1} - \mathbf{g}^k\|$) on an example image from this benchmark in Fig. 1, where $\mathbf{g}^{\mathtt{gt}}$ denotes the ground-truth image gradient. To provide more readable illustrations of convergence behaviors, here all relative errors are plotted starting from $k = 1$. We also show zoomed in curve comparisons of our two deep models in Fig. 1 (b). Notice that we indeed only have one implicit deep model for this task. But to compare its performance with methods based on different ℓ_p energies, we just repeatedly plot its relative errors (as green curves) in multiple subfigures.

It is observed in Fig. 1 (a) that our deep models always converged within 5-6 iterations, while both ADMM and HQS needed dozens of steps to stop their iterations. The dashdot curves in Fig. 1 (a) show that the designed EPADNet satisfied the constraint of errors in Condition 1 all the time, thus the global convergence to the critical-point of Eq. (14) can be experimentally guaranteed. All these results verified our proved theories. We can further see in Fig. 1 (b) that propagations of our two deep models (solid red and green curves) had obtained significantly lower reconstruction errors than conventional algorithms even just after the first iteration (i.e., the initial points of these curves). This is because built-in networks actually learned a direct descent direction toward the desired solutions, which demonstrated the superiority of our framework again.

Explicit / Implicit PADNet on Image Domain Non-blind deconvolution on image domain is commonly formulated as the following energy minimization task (Li et al. 2013; Krishnan and Fergus 2009; Schmidt and Roth 2014)

$$\min_{\mathbf{x},\mathbf{g}} \frac{1}{2} f_{\mathtt{im}}(\mathbf{x}, \mathbf{g}; \mathbf{y}) + \lambda\|\mathbf{g}\|_p^p, \tag{15}$$

Table 1: Averaged convergence results of ADMM, HQS, IPADNet and EPADNet on Eq. (14).

Ave.	Number of Iterations (denoted as K)				$\|\mathbf{g}^K - \mathbf{g}^{\mathrm{gt}}\|/\|\mathbf{g}^{\mathrm{gt}}\|$			
Alg.	ADMM	HQS	IPADNet	EPADNet	ADMM	HQS	IPADNet	EPADNet
ℓ_1	29	33	6	**5**	1.3146	1.3354	**0.4291**	0.4516
$\ell_{0.8}$	28	32	6	**5**	1.2069	1.2215	0.4291	**0.4140**
ℓ_0	40	54	**6**	**6**	1.7114	1.7598	**0.4291**	0.6005

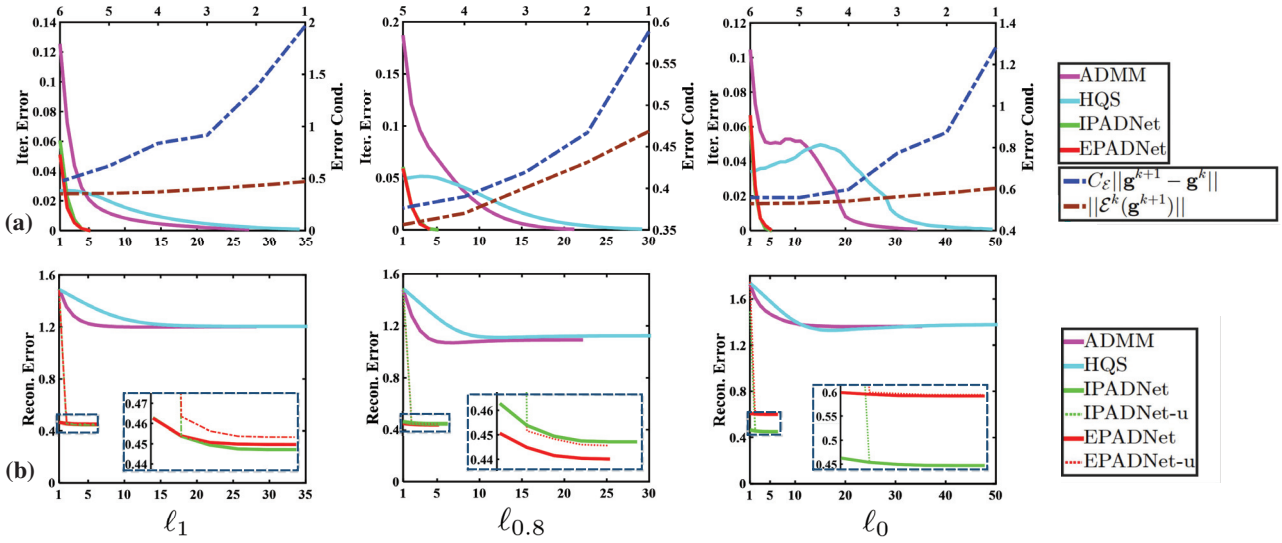

Figure 1: Convergence curves of ADMM, HQS, IPADNet and EPADNet on Eq. (14). (a) Iteration Error ("Iter. Error"). (b) Reconstruction Error ("Recon. Error"). Error Condition ("Error Cond.") referring to EPADNet is illustrated by dashdot curves with right and top axises on subfigures in (a). "Recon. Error" of the auxiliary variable (i.e., $\mathbf{u}$) of our deep models are plotted by relatively thin dotted curves on subfigures in (b). All horizontal axises denote the number of iterations.

in which the fidelity $f_{\mathtt{im}}$ can be formulated as

$$f_{\mathtt{im}}(\mathbf{x}, \mathbf{g}; \mathbf{y}) := \inf_{\mathbf{x}} \left\{ \|\mathbf{x} \otimes \mathbf{k} - \mathbf{y}\|^2 + \beta \|\mathcal{D}(\mathbf{x}) - \mathbf{g}\|^2 \right\}. \tag{16}$$

Here β is a penalty parameter, $\mathbf{x}$ and $\mathbf{g}$ are variables in image and gradient domains, respectively.

In this part, we build an explicit PADNet using Eq. (15) to pursuit $\mathbf{x}$, in which we set $p = 0.8$ and introduce an additional linear layer derived by $\nabla_{\mathbf{x}} f_{\mathtt{im}} = 0$ to transfer variables from gradient domain to image domain. In contrast, by simply defining $f_{\mathtt{ex}}(\mathbf{x}; \mathbf{y}) = \frac{1}{2}\|\mathbf{x} \otimes \mathbf{k} - \mathbf{y}\|^2$ and discarding explicit ℓ_p-priors, we can also design an implicit PADNet to learn priors from training data for this task. Here the basic architecture $\mathcal{G}$ (used in our deep models) consists of 7 convolution layers. The ReLU nonlinearities are added between each two linear layers accordingly and batch normalizations (BN) (Ioffe and Szegedy 2015) are also introduced for convolution operations from 2-nd to 6-th linear layers.

We compare performances of our two deep models against state-of-the-art algorithms, including TV (Li et al. 2013), HL (Krishnan and Fergus 2009), EPLL (Zoran and Weiss 2011), IDD-BM3D (Danielyan, Katkovnik, and Egiazarian 2012), MLP (Schuler et al. 2013) and CSF (Schmidt and Roth 2014) on both standard Levin *et. al.*' and more challenging Sun *et. al.*' benchmarks. The averaged quantitative results (i.e., PSNR and SSIM), are reported in Tab. 2, in which "(E)" and "(I)" denote algorithms based on explicit and implicit PADNet, respectively. We can recognize that "Ours (E)" and the works in (Li et al. 2013; Krishnan and Fergus 2009) actually all address this task by optimizing Eq. (15). Thanks to built-in networks, we achieved much better performance than conventional optimization approaches. We further observed that discriminative learning approaches (Schmidt and Roth 2014; Schuler et al. 2013) also performed well as they learn adaptive networks from training data. Overall, the results of our two algorithms are better than other compared approaches. The PSNR score of "Ours (E)" is even higher than that of "Ours (I)" on standard Levin *et. al.*'s dataset. We argue that this is reasonable because $\ell_{0.8}$ prior actually has been powerful enough for relatively simple test images. While "Ours (I)" obtained the best quantitative results on Sun *et. al.*'s dataset, which demonstrated that our prior-and-data aggregated framework is especially more efficient on real-world challenging applications (see Remark 4).

Table 2: Averaged non-blind deconvolution results on Levin *et. al.*' and Sun *et. al.*' benchmarks.

	Alg.	TV	HL	Ours (E)	EPLL	IDD-BM3D	MLP	CSF	Ours (I)
Levin	PSNR	29.38	30.12	**33.41**	31.65	31.53	31.32	32.74	33.37
	SSIM	0.88	0.90	**0.95**	0.93	0.90	0.90	0.93	**0.95**
Sun	PSNR	30.67	31.03	32.69	32.44	30.79	31.47	31.55	**32.71**
	SSIM	0.85	0.85	**0.89**	0.88	0.87	0.86	0.87	**0.89**

Table 3: Averaged single image haze removal results on Fattal's benchmark.

Alg.	He *et. al.*	Meng *et. al.*	Chen *et. al.*	Berman *et. al.*	Li *et. al.*	Ren *et. al.*	Cai *et. al.*	Ours
PSNR	27.11	26.13	26.47	26.09	25.54	24.40	21.63	**28.47**
SSIM	0.96	0.95	0.93	0.95	0.94	0.94	0.89	**0.96**
Time (s)	17.20	6.95	272.07	3.73	62.67	5.87	5.77	**2.74**

Single Image Haze Removal

Finally, we evaluate PADNet on the task of single image haze removal, which is a challenging real-world vision application. Most existing works address this task as estimating the latent scene radiance $\mathbf{J}$ from given hazy observation $\mathbf{I}$ from the following linear interpolation formula

$$\mathbf{I}(x) = t(x)\mathbf{J}(x) + (1 - t(x))\mathbf{A}, \tag{17}$$

where t is transmission, $\mathbf{A}$ is global atmospheric light and x denotes the pixel index.

It is known that transmission t expresses the relative portion of light that managed to survive the entire path between the observer and a surface point in the scene without being scattered (Fattal 2014). With Eq. (17), we have that estimating accurate transmission map t plays the core role in this task. However, due to multiple solutions exist for a single hazy image, the problem is highly ill-posed. Recent works often design their models based on different perspectives on transmissions within the following prior regularized energy

$$\begin{aligned} &\min_{\mathbf{t}} \tfrac{1}{2}\|\mathbf{t} - \tilde{\mathbf{t}}\|^2 + \lambda r(\mathbf{t}), \\ &\text{e.g.,} \begin{cases} r_{\texttt{TGV}}(\mathbf{t}) = \|\nabla\mathbf{t} - \mathbf{z}\|_1 + \beta\|\nabla\mathbf{z}\|_1, \\ \text{(Chen, Do, and Wang 2016)}, \\ r_{\texttt{MRF}}(\mathbf{t}) = \sum_j \|\mathbf{w}_j \odot (\mathbf{d}_j \otimes \mathbf{t})\|_1, \\ \text{(Meng et al. 2013)}, \end{cases} \end{aligned} \tag{18}$$

where $\mathbf{t}$ and $\tilde{\mathbf{t}}$ respectively denote the discrete transmission vector and its propagation guidance. The regularization r can be derived based on different tools, e.g., Total Generalized Variation (TGV) (Chen, Do, and Wang 2016) and Markov Random Field (MRF) (Meng et al. 2013). In $r_{\texttt{MRF}}$, $\mathbf{z}$ denotes an auxiliary variable, $\odot$ is Hadamard product and $\{\mathbf{w}_j\}$ are weight vectors for local filters $\{\mathbf{d}_j\}$.

In this part, we first utilize implicit strategy to design PADNet based on fidelity $f(\mathbf{t}) = \frac{1}{2}\|\mathbf{t} - \tilde{\mathbf{t}}\|^2$ (with the same guidance $\tilde{\mathbf{t}}$ defined in (Meng et al. 2013)) to estimate transmission and then recover the latent scene radiance from Eq. (17) as that in (He, Sun, and Tang 2011; Chen, Do, and Wang 2016; Meng et al. 2013). We build basic architecture $\mathcal{G}$ with 17 convolution layers (ReLU and BN operations are incorporated using the same strategy as that in above image deconvolution task) and train it on synthetic hazy images (Ren et al. 2016) for our deep model.

We evaluate the performance of our deep model together with five existing handcrafted-prior based algorithms (i.e., (He, Sun, and Tang 2011; Meng et al. 2013; Chen, Do, and Wang 2016; Berman, Avidan, and others 2016; Li et al. 2014)) and two empirically designed deep networks (i.e., (Ren et al. 2016; Cai et al. 2016))[5] on the commonly used Fattal's benchmark (Fattal 2008), which consists of 11 challenging hazy images, including architecture, natural scenery and indoor scene. The averaged quantitative results, including PSNR, SSIM and running time in seconds (denoted as "Time (s)"), are given in Tab. 3. Two empirically designed networks in (Cai et al. 2016; Ren et al. 2016) performed better than most conventional prior-based methods. Though obtained good dehazing results, the work in (Chen, Do, and Wang 2016) has the longest running time. Our proposed deep model achieved the best performance among all compared algorithms on this benchmark. This is mainly because that PADNet can successfully fuse cues from both human perspectives and training data to estimate haze distributions. Furthermore, the speed of PADNet is the fastest among all compared methods, which also verified the efficiency of our framework.

Conclusions

This paper proposed a novel framework, named proximal alternating direction network (PADNet), to design deep models for different learning tasks. Our theoretical results first showed that we can utilize empirically designed architectures to build globally converged PADNet for the given energy minimization model. We further proved that a converged PADNet can also be designed by learning priors from training data. At last we experimentally verified our analysis and demonstrated promising results of PADNet on different real-world applications.

[5]In this subsection, we always denote these methods as He *et. al.*, Meng *et. al.*, Chen *et. al.*, Berman *et. al.*, Li *et. al.*, Ren *et. al.* and Cai *et. al.*, respectively.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (Nos. 61672125, 61632019, 61432003, 61572096 and 61733002), and the Hong Kong Scholar Program (No. XJ2015008). Dr. Liu is also a visiting researcher with Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060

References

Andrychowicz, M.; Denil, M.; Gomez, S.; Hoffman, M. W.; Pfau, D.; Schaul, T.; and de Freitas, N. 2016. Learning to learn by gradient descent by gradient descent. In *NIPS*, 3981–3989.

Attouch, H.; Bolte, J.; Redont, P.; and Soubeyran, A. 2010. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality. *Mathematics of Operations Research* 35(2):438–457.

Berman, D.; Avidan, S.; et al. 2016. Non-local image dehazing. In *CVPR*, 1674–1682.

Cai, B.; Xu, X.; Jia, K.; Qing, C.; and Tao, D. 2016. Dehazenet: An end-to-end system for single image haze removal. *IEEE TIP* 25(11):5187–5198.

Chen, Y., and Pock, T. 2017. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE TPAMI* 39(6):1256–1272.

Chen, C.; Do, M. N.; and Wang, J. 2016. Robust image and video dehazing with visual artifact suppression via gradient residual minimization. In *ECCV*, 576–591.

Chouzenoux, E.; Pesquet, J.-C.; and Repetti, A. 2016. A block coordinate variable metric forward–backward algorithm. *Journal of Global Optimization* 66(3):457–485.

Danielyan, A.; Katkovnik, V.; and Egiazarian, K. 2012. Bm3d frames and variational image deblurring. *IEEE TIP* 21(4):1715–1728.

Fattal, R. 2008. Single image dehazing. *ACM Transactions on Graphics (TOG)* 27(3):72.

Fattal, R. 2014. Dehazing using color-lines. *ACM Transactions on Graphics (TOG)* 34(1):13.

Gregor, K., and LeCun, Y. 2010. Learning fast approximations of sparse coding. In *ICML*, 399–406.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, K.; Sun, J.; and Tang, X. 2011. Single image haze removal using dark channel prior. *IEEE TPAMI* 33(12):2341–2353.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456.

Krishnan, D., and Fergus, R. 2009. Fast image deconvolution using hyper-laplacian priors. In *NIPS*, 1033–1041.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, 1097–1105.

Levin, A.; Weiss, Y.; Durand, F.; and Freeman, W. T. 2009. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, 1964–1971.

Li, C.; Yin, W.; Jiang, H.; and Zhang, Y. 2013. An efficient augmented lagrangian method with applications to total variation minimization. *Computational Optimization and Applications* 56(3):507–530.

Li, Y.; Guo, F.; Tan, R. T.; and Brown, M. S. 2014. A contrast enhancement framework with jpeg artifacts suppression. In *ECCV*, 174–188.

Lin, Z.; Liu, R.; and Su, Z. 2011. Linearized alternating direction method with adaptive penalty for low-rank representation. In *NIPS*, 612–620.

Meng, G.; Wang, Y.; Duan, J.; Xiang, S.; and Pan, C. 2013. Efficient image dehazing with boundary constraint and contextual regularization. In *ICCV*, 617–624.

Parikh, N.; Boyd, S.; et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1(3):127–239.

Ren, W.; Liu, S.; Zhang, H.; Pan, J.; Cao, X.; and Yang, M.-H. 2016. Single image dehazing via multi-scale convolutional neural networks. In *ECCV*, 154–169.

Rockafellar, R. T., and Wets, R. J.-B. 2009. *Variational analysis*, volume 317. Springer Science & Business Media.

Schmidt, U., and Roth, S. 2014. Shrinkage fields for effective image restoration. In *CVPR*, 2774–2781.

Schuler, C. J.; Christopher Burger, H.; Harmeling, S.; and Scholkopf, B. 2013. A machine learning approach for non-blind image deconvolution. In *CVPR*, 1067–1074.

Shen, L.; Lin, Z.; and Huang, Q. 2016. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*, 467–482. Springer.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, L.; Cho, S.; Wang, J.; and Hays, J. 2013. Edge-based blur kernel estimation using patch priors. In *ICCP*.

Teh, Y. W.; Welling, M.; Osindero, S.; and Hinton, G. E. 2003. Energy-based models for sparse overcomplete representations. *JMLR* 4(Dec):1235–1260.

Wang, Y.; Liu, R.; Song, X.; and Su, Z. 2017. An inexact proximal alternating direction method for non-convex and non-smooth matrix factorization and beyond. *arXiv preprint arXiv:1702.08627*.

Xu, C.; Lin, Z.; and Zha, H. 2016. Relaxed majorization-minimization for non-smooth and non-convex optimization. In *AAAI*, 812–818.

Zhao, J.; Mathieu, M.; and LeCun, Y. 2016. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*.

Zoran, D., and Weiss, Y. 2011. From learning models of natural image patches to whole image restoration. In *ICCV*, 479–486.

Zuo, W.; Meng, D.; Zhang, L.; Feng, X.; and Zhang, D. 2013. A generalized iterated shrinkage algorithm for non-convex sparse coding. In *ICCV*, 217–224.