Escaping the Deep Saddle Points

Ankit Goyal   Enayat Ullah

Indian Institute of Technology Kanpur
Kanpur, India

14 April 2016

Outline

Why bother about saddle points?

- Proliferation of saddle points, in non-convex high dimensional setting. [1]
- Symmetry leads to saddle points. [2]
- In high-dimensions, sufficient to converge to local minima.
- Hessian based approaches to escape saddle points. [1]
- Strict-saddle property, SGD escape saddle point without second-order information. [3]

Strict Saddle

Definition

A twice differentiable function $f(w)$ is strict-saddle, if all its local minima have $\nabla^2 f(w) \succ 0$ and all its other stationary points satisfy $\lambda_{min}(\nabla^2 f(w)) < 0$.

Definition (Formal)

A twice differentiable function $f(w)$ is $(\alpha, \gamma, \epsilon, \delta)$ strict-saddle, if for any point w at least one of the following is true

1. $\|\nabla f(w)\| \geq \epsilon$.
2. $\lambda_{min}(\nabla^2 f(w)) \leq -\gamma$.
3. There is a local minimum $w^\star$ such that $\|w - w^\star\| \leq \delta$, and the function $f(w')$ restricted to 2δ neighborhood of $w^\star$ ($\|w' - w^\star\| \leq 2\delta$) is α-strongly convex.

Noisy SGD

Algorithm 1 Noisy Stochastic Gradient

Require: SG oracle $SG(w)$, initial point w_0, desired accuracy κ.
Ensure: w_t that is close to some local minimum $w^\star$.

Choose $\eta = \min\{\tilde{O}(\kappa^2/\log(1/\kappa)), \eta_{\max}\}$, $T = \tilde{O}(1/\eta^2)$
for $t = 0$ to $T - 1$ **do**
    Sample noise n uniformly from unit sphere.
    $w_{t+1} \leftarrow w_t - \eta(SG(w) + n)$
end for

Main Theorem

Theorem

Suppose $f(w)$ is strict-saddle, then Noisy Gradient Descent outputs a point that is close to a local minimum in polynomial number of steps.

Theorem (Formally)

Suppose a function $f(w) : \mathbb{R}^d \to \mathbb{R}$ that is $(\alpha, \gamma, \epsilon, \delta)$ strict-saddle, and has a stochastic gradient oracle with radius at most Q. Further, suppose the function is bounded by $|f(w)| \leq B$, is β-smooth and has ρ-Lipschitz Hessian. Then there exists a threshold $\eta_{\max} = \tilde{\Theta}(1)$, so that for any $\zeta > 0$, and for any $\eta \leq \eta_{\max} / \max\{1, \log(1/\zeta)\}$, with probability at least $1 - \zeta$ in $t = \tilde{O}(\eta^{-2} \log(1/\zeta))$ iterations. NGD outputs a point w_t that is $\tilde{O}(\sqrt{\eta \log(1/\eta\zeta)})$-close to some local minimum $w^\star$.

Tensor

- A 4-th order tensor $T \in \mathbb{R}^{d^4}$ has an orthogonal decomposition if it can be written as
$$T = \sum_{i=1}^{d} a_i^{\otimes 4}, \tag{1}$$
where a_i's are orthonormal vectors that satisfy $\|a_i\| = 1$ and $a_i^T a_j = 0$ for $i \neq j$.
- Multilinear form for a p-th order tensor $T \in \mathbb{R}^{d^p}$, where matrices $M_i \in \mathbb{R}^{d \times n_i}$ $i \in [p]$, we define
$$[T(M_1, M_2, ..., M_p)]_{i_1, i_2, \ldots, i_p} = \sum_{j_1, j_2, \ldots, j_p \in [d]} T_{j_1, j_2, \ldots, j_p} \prod_{t \in [p]} M_t[i_t, j_t]. \tag{2}$$

Tensor Decomposition

The orthogonal tensor decomposition objective function can be equivalently be expressed as follows:

$$\min_{\forall i, \|u_i\|^2=1} \quad \sum_{i \neq j} T(u_i, u_i, u_j, u_j), \tag{3}$$

Intuition: Expanding u_k along the orthogonal basis $\{a_i\}$'s, we get $u_k = \sum_{i=1}^{d} z_k(i) a_i$. Therefore, $T(u_k, u_k, u_l, u_l) = \sum_{i=1}^{d} (z_k(i))^2 (z_l(i))^2$. This is nonnegative, and is equal to 0 only when the support of z_k and z_l do not intersect.

Theorem

The optimization problem (9) is $(\alpha, \gamma, \epsilon, \delta)$-strict-saddle, for $\alpha = 1$ and $\gamma, \epsilon, \delta = 1/poly(d)$. Moreover, all its local minima have the form $u_i = \kappa_i a_{\pi(i)}$ for some $\kappa_i = \pm 1$ and permutation $\pi(i)$.

Two-layer feed-forward neural network

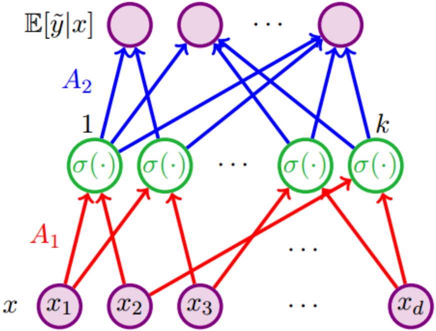

Figure: Graphical representation of a neural network, Image Courtesy: "Beating the perils of non-convexity: Guaranteed training of neuralnetworks using tensor methods"

$$\tilde{f}(x) := \mathbb{E}[\tilde{y}|x] = \langle a_2, \sigma(A_1^\top x + b_1)\rangle + b_2, \tag{4}$$

Score Function

- Score functions are a class of features, which capture local variations in the probability density function of the input.
- The m-th order score function $\mathcal{S}_m(x) \in \bigotimes^m \mathbb{R}^d$ is defined as follows:[4]

$$\mathcal{S}_m(x) := (-1)^m \frac{\nabla_x^{(m)} p(x)}{p(x)}, \tag{5}$$

where $p(x)$ is the probability density function of random vector $x \in \mathbb{R}^d$.

Tensor Decomposition for feed-forward neural network

- For label-function $f(x) := \mathbb{E}[y|x]$, Janzamin et al. [4] showed that

$$\mathbb{E}[y \cdot \mathcal{S}_3(x)] = \mathbb{E}[\nabla_x^{(3)} f(x)] \tag{6}$$

- For a feed-forward neural network, Janzamin et al. [5] showed that

$$\mathbb{E}\left[\tilde{y} \cdot \mathcal{S}_3(x)\right] = \sum_{j \in [k]} \lambda_j \cdot (A_1)_j \otimes (A_1)_j \otimes (A_1)_j \in \mathbb{R}^{d \times d \times d}, \tag{7}$$

NN LIFT Algorithm

Algorithm 2 NN-LIFT (Neural Network LearnIng using Feature Tensors)

Require: Labeled samples $\{(x_i, y_i) : i \in [n]\}$, parameter $\tilde{\epsilon}_1$, parameter λ.
Require: Third order score function $\mathcal{S}_3(x)$ of the input x

1: Compute $\widehat{T} := \frac{1}{n} \sum_{i\in[n]} y_i \cdot \mathcal{S}_3(x_i)$.
2: $\{(\hat{A}_1)_j\}_{j\in[k]} =$ tensor decomposition$(\widehat{T})$
3: $\hat{b}_1 =$ Fourier method$(\{(x_i, y_i) : i \in [n]\}, \hat{A}_1, \tilde{\epsilon}_1)$
4: $(\hat{a}_2, \hat{b}_2) =$ Ridge regression$(\{(x_i, y_i) : i \in [n]\}, \hat{A}_1, \hat{b}_1, \lambda)$

Approach

1. $\mathbb{E}\left[\tilde{y} \cdot \mathcal{S}_3(x)\right] = \sum_{j\in[k]} \lambda_j \cdot (A_1)_j \otimes (A_1)_j \otimes (A_1)_j \in \mathbb{R}^{d\times d\times d}$
2. Problem : May not yield orthogonal decomposition.
   Solution: Whitening, we get $T(W, W, W) \in \mathbb{R}^{K\times k\times k}$
3. The new objective $T(u, u, u)$ is strict-saddle [Proof in report]
4. Modified the Tensor Power method to use Noisy Gradient Descent for Tensor Decomposition

Robust Tensor Power Method

Algorithm 3 Robust tensor power method

Require: Symmetric tensor $\tilde{T} \in \mathbb{R}^{d' \times d' \times d'}$, number of iterations N

Ensure: the estimated eigenvector/eigenvalue pair; the deflated tensor.

1: **for** $\tau = 1$ to R **do**

2:     **for** $t = 1$ to N **do**

3:         Compute power iteration update

$$\widehat{v}_t^{(\tau)} := \frac{\tilde{T}(I, \widehat{v}_{t-1}^{(\tau)}, \widehat{v}_{t-1}^{(\tau)})}{\|\tilde{T}(I, \widehat{v}_{t-1}^{(\tau)}, \widehat{v}_{t-1}^{(\tau)})\|} \quad (8)$$

4:     **end for**

5: **end for**

6: Let $\tau^* := \arg\max_{\tau \in [R]}\{\tilde{T}(\widehat{v}_N^{(\tau)}, \widehat{v}_N^{(\tau)}, \widehat{v}_N^{(\tau)})\}$.

7: Do N power iteration updates (8) starting from $\widehat{v}_N^{(\tau^*)}$ to obtain $\widehat{v}$, and set $\hat{\mu} := \tilde{T}(\widehat{v}, \widehat{v}, \widehat{v})$.

8: **return** the estimated eigenvector/eigenvalue pair $(\widehat{v}, \hat{\mu})$; the deflated tensor $\tilde{T} - \hat{\mu} \cdot \hat{v}^{\otimes 3}$.

Modified Tensor Decomposition Using Noisy Gradient Descent

Algorithm 4 Tensor Decomposition Using Noisy Gradient Descent

Require: Symmetric tensor $\tilde{T} \in \mathbb{R}^{d' \times d' \times d'}$, number of iterations N
Ensure: the estimated eigenvector/eigenvalue pair; the deflated tensor.

1: Compute $\widehat{v}$ using by using Noisy Gradient Descent on the following objective function

$$\max_{\|u\|^2=1} \quad \tilde{T}(u, u, u), \tag{9}$$

2: Set $\hat{\mu} := \tilde{T}(\widehat{v}, \widehat{v}, \widehat{v})$.

3: **return** the estimated eigenvector/eigenvalue pair $(\widehat{v}, \hat{\mu})$; the deflated tensor $\tilde{T} - \hat{\mu} \cdot \hat{v}^{\otimes 3}$.

Input: Tensor $T = \sum_{i \in [k]} \lambda_i u_i^{\otimes 3}$

$\downarrow$

Whitening

$\downarrow$

Tensor Decomposition using Noising Gradient Descent

$\downarrow$

Output: $\{u_i\}_{i \in [k]}$

$\downarrow$

Unwhitening

Figure: Overview of tensor decomposition algorithm for third order tensor.

Table: Results

Dimension	Average Error(Tensor Decomposition)	Average Error(Standard NN)
d=3	1.82	0.564
d=10	2.79	0.682
d=20	5.67	1.237

Details of the dataset created in the report.

References I

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio.
Identifying and attacking the saddle point problem in high-dimensional non-convex optimization.
In *Advances in Neural Information Processing Systems*, pages 2933–2941, 2014.

Andrew M Saxe, James L McClelland, and Surya Ganguli.
Exact solutions to the nonlinear dynamics of learning in deep linear neural networks.
arXiv preprint arXiv:1312.6120, 2013.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan.
Escaping from saddle points—online stochastic gradient for tensor decomposition.
arXiv preprint arXiv:1503.02101, 2015.

References II

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar.
Score function features for discriminative learning: Matrix and tensor framework.
arXiv preprint arXiv:1412.2863, 2014.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar.
Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods.
CoRR abs/1506.08473, 2015.