title.png

CSE 201 – Advanced Optimization: Theory and Algorithms

Non-convex optimization problems are everywhere!

Examples include:

- Training multi-layer neural networks $\leftarrow$ (focus of this talk)
- Maximum likelihood estimation of latent variable models
- Clustering: k-means, hierarchical

graphmodel.png

cluster.png

Convex vs. Non-convex Optimization

- Unique global optimum

- Multiple local optima, saddle points

Non-convex more prevalent!

What makes training Neural Networks hard?

What makes training Neural Networks hard?

Potential Problems:

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees
- proliferation of local minima

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees
- proliferation of local minima
- saddle points

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees
- proliferation of local minima
- saddle points

Our contribution:

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees
- proliferation of local minima
- saddle points

Our contribution:

- geometric intuition derived from low dimensional spaces, doesn't generalize to high dimensional spaces

What makes training Neural Networks hard?

Potential Problems:

- no convergence guarantees
- proliferation of local minima
- saddle points

Our contribution:

- geometric intuition derived from low dimensional spaces, doesn't generalize to high dimensional spaces
- consequently, we argue that the **proliferation of saddle points**, not local minima is the main source of difficulty

Related work

Prior to this work, there was very little related literature.

- *Neural networks and principal component analysis: Learning from examples without local minima*, Baldi and Hornik (1989)
- *On the saddle point problem for non-convex optimization*, Pascanu, Dauphin, Ganguli and Bengio (arXiv May 2014)

Outline

Introduction

Outline

Introduction

Different Types of Critical Points

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Different Types of Critical Points

1st order critical points ($\nabla f(\theta) = 0$) of $f(\theta)$ can be characterized by the curvature of the function in its vicinity, as described by the eigenvalues of $\boldsymbol{H}$, the Hessian matrix.

Different Types of Critical Points

1st order critical points ($\nabla f(\theta) = 0$) of $f(\theta)$ can be characterized by the curvature of the function in its vicinity, as described by the eigenvalues of $\boldsymbol{H}$, the Hessian matrix.

- if $\forall_i \lambda_i > 0 \rightarrow$ local minimum

Different Types of Critical Points

1st order critical points ($\nabla f(\theta) = 0$) of $f(\theta)$ can be characterized by the curvature of the function in its vicinity, as described by the eigenvalues of $\boldsymbol{H}$, the Hessian matrix.

- if $\forall_i \lambda_i > 0 \rightarrow$ local minimum
- if $\forall_i \lambda_i < 0 \rightarrow$ local maximum

Different Types of Critical Points

1st order critical points ($\nabla f(\theta) = 0$) of $f(\theta)$ can be characterized by the curvature of the function in its vicinity, as described by the eigenvalues of $\boldsymbol{H}$, the Hessian matrix.

- if $\forall_i \lambda_i > 0 \rightarrow$ local minimum
- if $\forall_i \lambda_i < 0 \rightarrow$ local maximum
- if $\exists i$ s.t. $\lambda_i > 0$, $\lambda_i < 0$ and $\lambda_i \neq 0 \rightarrow$ saddle point with min-max structure

Different Types of Critical Points

1st order critical points ($\nabla f(\theta) = 0$) of $f(\theta)$ can be characterized by the curvature of the function in its vicinity, as described by the eigenvalues of $\boldsymbol{H}$, the Hessian matrix.

- if $\forall_i \lambda_i > 0 \rightarrow$ local minimum
- if $\forall_i \lambda_i < 0 \rightarrow$ local maximum
- if $\exists i$ s.t. $\lambda_i > 0$, $\lambda_i < 0$ and $\lambda_i \neq 0 \rightarrow$ saddle point with min-max structure

Different Types of Saddle Points

If $\boldsymbol{H}$ is singular ($\exists \lambda_i = 0$), then the *degenerate* 1st order critical point can also be a saddle point.

Different Types of Saddle Points

If $\boldsymbol{H}$ is singular ($\exists \lambda_i = 0$), then the *degenerate* 1st order critical point can also be a saddle point.

minmaxsaddle.png

(c) Min-max saddle ($\lambda_i > 0$, $\lambda_i < 0$ and $\lambda_i \neq 0$)

monkeysaddle.png

(d) Monkey saddle (min-max structure and 0 eigenvalue)

Different Types of Saddle Points

If $\boldsymbol{H}$ is singular ($\exists \lambda_i = 0$), then the *degenerate* 1st order critical point can also be a saddle point.

minmaxsaddle.png

(e) Min-max saddle ($\lambda_i > 0$, $\lambda_i < 0$ and $\lambda_i \neq 0$)

monkeysaddle.png

(f) Monkey saddle (min-max structure and 0 eigenvalue)

For purposes of this work, we focus on non-degenerate 1st order saddle points for which the Hessian is not singular.

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Saddle points are prevalent in high dimensions

Saddle points are prevalent in high dimensions

Claim

The ratio of the number of saddle points to local minima increases exponentially with dimensionality N.

Saddle points are prevalent in high dimensions

Claim

The ratio of the number of saddle points to local minima increases exponentially with dimensionality N.

Intuition

simulation.png

Terminology

We plan to use some results from statistical physics, but first we define two key terms.

Terminology

We plan to use some results from statistical physics, but first we define two key terms.

Index of a critical point (α)

\# of negative eigenvalues of $\boldsymbol{H}$ at a given critical point

Terminology

We plan to use some results from statistical physics, but first we define two key terms.

Index of a critical point (α)

$\#$ of negative eigenvalues of $\boldsymbol{H}$ at a given critical point

Gaussian Random Field

set of normally distributed random variables $Y(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^n$, with a collection of distribution functions $F(Y(\boldsymbol{x_1}) \leq y_1, \ldots, Y(\boldsymbol{x_n}) \leq y_n)$ where the $\boldsymbol{x_i}$ can be points on some manifold

gaussrandomfield.png

Related Studies

In *Bray & Dean 2007*, the authors calculate the average number of critical points of a Gaussian field on high-dimensional space as a function of their energy and index.

Related Studies

In *Bray & Dean 2007*, the authors calculate the average number of critical points of a Gaussian field on high-dimensional space as a function of their energy and index.

We use two specific theoretical results from their work:

Related Studies

In *Bray & Dean 2007*, the authors calculate the average number of critical points of a Gaussian field on high-dimensional space as a function of their energy and index.

We use two specific theoretical results from their work:

1. eigenvalues of $\boldsymbol{H}$ at a critical point are distributed according to Wigner's famous semicircular law but shifted by an amount determined by ε

Theoretical Prediction for GRF

2. In the ε (training error) vs. α (index of critical point) plane, the critical points concentrate on a monotonically increasing curve as α ranges from 0 to 1

monotone.png

Experimental Validation

Experimental Validation

We want to experimentally test whether the theoretical predictions for Gaussian random fields presented by Bray and Dean (2007) hold for neural networks.

Experimental Validation

We want to experimentally test whether the theoretical predictions for Gaussian random fields presented by Bray and Dean (2007) hold for neural networks.

Goals

Experimental Validation

We want to experimentally test whether the theoretical predictions for Gaussian random fields presented by Bray and Dean (2007) hold for neural networks.

Goals

1. explore how critical points of a single layer MLP are distributed in the ε-α plane

Experimental Validation

We want to experimentally test whether the theoretical predictions for Gaussian random fields presented by Bray and Dean (2007) hold for neural networks.

Goals

1. explore how critical points of a single layer MLP are distributed in the ε-α plane
2. explore how the eigenvalues of $\boldsymbol{H}$ at these critical points are distributed

Experimental Methodology

Experimental Methodology

We use a small MLP trained on a down-sampled version of MNIST and CIFAR-10

Experimental Methodology

We use a small MLP trained on a down-sampled version of MNIST and CIFAR-10

MNIST

- Use a single layer MLP and first use our ideal algorithm (SFN) to define an ideal training path (store all parameters, repeat many times)
- Using the stored parameters, repeat the process adding some noise and use the Newton method to discover nearby critical points along the ideal training path

Experimental Methodology

We use a small MLP trained on a down-sampled version of MNIST and CIFAR-10

MNIST

- Use a single layer MLP and first use our ideal algorithm (SFN) to define an ideal training path (store all parameters, repeat many times)
- Using the stored parameters, repeat the process adding some noise and use the Newton method to discover nearby critical points along the ideal training path

CIFAR-10

- Train multiple 3-layer deep neural networks using SGD and save the parameters for each epoch
- We then train using the Newton method to discover nearby critical points along the ideal training path

Saddle Point Experiments

saddlepoint_experiment1_mnist.png

Saddle Point Experiments

saddlepoint_experiment1_cifar10.png

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Dynamics of Optimization Algorithms near Saddle Points

Dynamics of Optimization Algorithms near Saddle Points

- Given the prevalence of saddle points, we want to understand how various optimization algorithms behave near them

Dynamics of Optimization Algorithms near Saddle Points

- Given the prevalence of saddle points, we want to understand how various optimization algorithms behave near them
- We focus on non-degenerate saddle points for which $\boldsymbol{H}$ is not singular

Dynamics of Optimization Algorithms near Saddle Points

- Given the prevalence of saddle points, we want to understand how various optimization algorithms behave near them
- We focus on non-degenerate saddle points for which $\boldsymbol{H}$ is not singular

To locally analyze these critical points we reparametrize our function f:

$$f(\theta_0 + \Delta\theta) = f(\theta_0) + \frac{1}{2}\sum_{i=1}^{n_\theta} \lambda_i \Delta \boldsymbol{v_i^2}$$

λ_i - ith eigenvalue of $\boldsymbol{H}$

$\Delta \boldsymbol{v_i} = (\boldsymbol{e_i}^T \Delta\theta)$

Gradient Descent near saddle points

Gradient Descent near saddle points

- A step of gradient descent always points away from the saddle close to it

Gradient Descent near saddle points

- A step of gradient descent always points away from the saddle close to it
- Drawback near saddles is not the direction but the step size $-\lambda_i \Delta \boldsymbol{v_i}$ as small steps are taken in directions corresponding to small eigenvalues

Gradient Descent near saddle points

- A step of gradient descent always points away from the saddle close to it
- Drawback near saddles is not the direction but the step size $-\lambda_i \Delta \boldsymbol{v_i}$ as small steps are taken in directions corresponding to small eigenvalues

gradient_descent1.png

(g) SGD (red dots) slows down near the saddle point

gradient_descent2.png

(h) SGD (green dots) particularly good at exploiting the unstable nature of a saddle point

Newton's method near saddle points

Newton's method near saddle points

- Newton's method solves the slowness problem by rescaling gradients in each direction with inverse of the corresponding eigenvalue (yielding step $-\Delta \boldsymbol{v_i}$)

Newton's method near saddle points

- Newton's method solves the slowness problem by rescaling gradients in each direction with inverse of the corresponding eigenvalue (yielding step $-\Delta \boldsymbol{v_i}$)
- if however, $\lambda_i < 0$ then

$$GD : -(-\lambda_i)\Delta \boldsymbol{v_i} = \lambda_i \Delta \boldsymbol{v_i}$$

$$Newton : -(-\lambda_i)\frac{1}{-\lambda_i}\Delta \boldsymbol{v_i} = -\Delta \boldsymbol{v_i}$$

Newton's method near saddle points

- Newton's method solves the slowness problem by rescaling gradients in each direction with inverse of the corresponding eigenvalue (yielding step $-\Delta \boldsymbol{v_i}$)
- if however, $\lambda_i < 0$ then

$$GD : -(-\lambda_i)\Delta \boldsymbol{v_i} = \lambda_i \Delta \boldsymbol{v_i}$$

$$Newton : -(-\lambda_i)\frac{1}{-\lambda_i}\Delta \boldsymbol{v_i} = -\Delta \boldsymbol{v_i}$$

newtonmethod.png

Figure: In (b) θ_0 becomes an attractor for the Newton method, which can get stuck in this saddle point and not converge to a local minima

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

How would an optimal algorithm behave?

How would an optimal algorithm behave?

- Analysis of Newton's method suggests a simple heuristic solution:

$$\text{Newton method: } x_{t+1} = x_t - \boldsymbol{H}^{-1} \Delta f(x_t)$$

How would an optimal algorithm behave?

- Analysis of Newton's method suggests a simple heuristic solution:

$$\text{Newton method:} \quad x_{t+1} = x_t - \boldsymbol{H}^{-1} \Delta f(x_t)$$

$$\text{Proposed solution:} \quad x_{t+1} = x_t - |\boldsymbol{H}|^{-1} \Delta f(x_t)$$

where $|\boldsymbol{H}| = \sum_i |\lambda_i| \boldsymbol{e_i}$

How would an optimal algorithm behave?

- Analysis of Newton's method suggests a simple heuristic solution:

$$\text{Newton method:} \quad x_{t+1} = x_t - \boldsymbol{H}^{-1}\Delta f(x_t)$$

$$\text{Proposed solution:} \quad x_{t+1} = x_t - |\boldsymbol{H}|^{-1}\Delta f(x_t)$$

where $|\boldsymbol{H}| = \sum_i |\lambda_i| \boldsymbol{e_i}$

- We show that this heuristic solution arises naturally from a generalized trust region approach

Generalized Trust Region Approach

In order to attack the saddle point problem, we define a class of generalized trust region methods (think constrained optimization problem)

Generalized Trust Region Approach

In order to attack the saddle point problem, we define a class of generalized trust region methods (think constrained optimization problem)

- Letting $\tau_k(f, \theta, \Delta\theta)$ be the k-order Taylor series expansion of f around θ evaluated at $\theta + \Delta\theta$, we get:

$$\Delta\theta = \operatorname*{argmin}_{\Delta\theta} \tau_k(f, \theta, \Delta\theta)$$

$$\text{with } k \in \{1, 2\} \text{ s.t. } d(\theta, \theta + \Delta\theta) \leq \Delta$$

Generalized Trust Region Approach

- We consider minimizing the 1st order Taylor expansion, and pick a suitable distance measure d that aims to give us some curvature information for f

Generalized Trust Region Approach

- We consider minimizing the 1st order Taylor expansion, and pick a suitable distance measure d that aims to give us some curvature information for f
- We do this by bounding the discrepancy between the first and second order taylor expansions of f:

$$d(\theta, \theta + \Delta\theta) = \left| f(\theta) + \nabla f \Delta\theta + \frac{1}{2}\Delta\theta^T \boldsymbol{H} \Delta\theta - f(\theta) - \nabla f \Delta\theta \right|$$

$$= \frac{1}{2}\left| \Delta\theta^T \boldsymbol{H} \Delta\theta \right| \leq \Delta$$

Generalized Trust Region Approach

- We bound the distance measure further by $\Delta\theta^T|\boldsymbol{H}|\Delta\theta$ which results in the following generalized trust region method:

$$\Delta\theta = \operatorname*{argmin}_{\Delta\theta} f(\theta) + \nabla f \Delta\theta \text{ s.t. } \Delta\theta^T|\boldsymbol{H}|\Delta\theta \leq \Delta$$

Generalized Trust Region Approach

- We bound the distance measure further by $\Delta\theta^T|\boldsymbol{H}|\Delta\theta$ which results in the following generalized trust region method:

$$\Delta\theta = \operatorname*{argmin}_{\Delta\theta} f(\theta) + \nabla f \Delta\theta \text{ s.t. } \Delta\theta^T|\boldsymbol{H}|\Delta\theta \leq \Delta$$

- Solving the constrained optimization using Lagrange multiplier yields a step of the form:

$$\Delta\theta = -\nabla f|\boldsymbol{H}|^{-1}$$

which is exactly our proposed heuristic!

Saddle-free Newton (SFN) Method

Subsequently, we propose the saddle-free Newton method which is identical to the Newton method when $\boldsymbol{H}$ is positive definite, but unlike the Newton method, it can escape saddle points.

saddlefree.png

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Existence of Saddle Points in Neural Networks

Existence of Saddle Points in Neural Networks

Goals

Existence of Saddle Points in Neural Networks

Goals

- validate existence of saddle points in neural networks

Existence of Saddle Points in Neural Networks

Goals

- validate existence of saddle points in neural networks
- want to observe behavior of algorithms we described earlier

Existence of Saddle Points in Neural Networks

Goals

- validate existence of saddle points in neural networks
- want to observe behavior of algorithms we described earlier

Methodology

Existence of Saddle Points in Neural Networks

Goals

- validate existence of saddle points in neural networks
- want to observe behavior of algorithms we described earlier

Methodology

- Train on down-sampled versions of MNIST and CIFAR-10, where we can compute the update directions by each algorithm exactly

Existence of Saddle Points in Neural Networks

Goals

- validate existence of saddle points in neural networks
- want to observe behavior of algorithms we described earlier

Methodology

- Train on down-sampled versions of MNIST and CIFAR-10, where we can compute the update directions by each algorithm exactly
- Compare MSGD, damped Newton and SFN

Experimental validation of the saddle-free Newton method

Experiment1.png

Effectiveness of saddle-free Newton

Effectiveness of saddle-free Newton

Goals

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets
- Exact implementation of SFN is intractable in high-dimensional problem, need a proxy to test effectiveness

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets
- Exact implementation of SFN is intractable in high-dimensional problem, need a proxy to test effectiveness

Methodology

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets
- Exact implementation of SFN is intractable in high-dimensional problem, need a proxy to test effectiveness

Methodology

- We test effectiveness of SFN on a deep autoencoder (using all MNIST data) and recurrent neural net (using Penn Treebank corpus)

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets
- Exact implementation of SFN is intractable in high-dimensional problem, need a proxy to test effectiveness

Methodology

- We test effectiveness of SFN on a deep autoencoder (using all MNIST data) and recurrent neural net (using Penn Treebank corpus)
- In each case we train the model with SGD and wait until learning stalls, we then continue training with SFN

Effectiveness of saddle-free Newton

Goals

- Want to test effectiveness of saddle-free Newton method on larger neural nets
- Exact implementation of SFN is intractable in high-dimensional problem, need a proxy to test effectiveness

Methodology

- We test effectiveness of SFN on a deep autoencoder (using all MNIST data) and recurrent neural net (using Penn Treebank corpus)
- In each case we train the model with SGD and wait until learning stalls, we then continue training with SFN
- We can't exactly compute $\boldsymbol{H}$ in the high-dimensional problem so we optimize in a lower-dimensional Krylov subspace

Experimental validation of the saddle-free Newton method

Experiment2.png

Outline

Introduction

Different Types of Critical Points

Prevalence of Saddle Points in High Dimensions

Dynamics of Optimization Algorithms near Saddle Points

Attacking the Saddle Point Problem

Experiments

Conclusion

Conclusion

Conclusion

- Argued about the prevalence of saddle points and proposed a very simple algorithm saddle-free Newton method to deal with them

Conclusion

- Argued about the prevalence of saddle points and proposed a very simple algorithm saddle-free Newton method to deal with them
- Method is highly impractical as exact implementation is intractable in a high dimensional problem

Conclusion

- Argued about the prevalence of saddle points and proposed a very simple algorithm saddle-free Newton method to deal with them
- Method is highly impractical as exact implementation is intractable in a high dimensional problem
- Experimental evidence limited and performed only using small networks

Subsequent Work

Subsequent Work

- Despite issues, rather important paper that accelerated work in saddle point behavior for non-convex optimization

Subsequent Work

- Despite issues, rather important paper that accelerated work in saddle point behavior for non-convex optimization
- Choromanska et. al (AISTATS 2015) showed a connection between deep networks with ReLU and spherical spin-glass model

Subsequent Work

- Despite issues, rather important paper that accelerated work in saddle point behavior for non-convex optimization
- Choromanska et. al (AISTATS 2015) showed a connection between deep networks with ReLU and spherical spin-glass model
- Ge. et al.(COLT 2015) introduced notion of strict saddle property for non-convex problem and showed that stochastic gradient descent converges to a local minimum in a polynomial number of iterations