01 - Introduction

Numerical Methods for Deep Learning

January 7, 2018

Course Overview

- First block: Linear Models
  1. regression and least squares
  2. classification: logistic regression and softmax
  3. numerical optimization: steepest descent, conjugate gradients, Newton's method, SGD
  4. regularization
  5. cross validation
  6. **First project: coding it all and working on a number of data sets**

Course Overview

- Second block: Neural Networks
  1. introduction
  2. single layer and two layer NN and their power
  3. computing derivatives
  4. numerical optimization: steepest descent, Nonlinear CG, and Newton's method, Stochastic gradient descent
  5. **Second project: Coding single and two layer NN**
- Third block: Deep Neural Networks
  1. introduction
  2. Residual Neural Networks and time dependent processes
  3. Solving the DNN problem - The adjoint and back propagation
  4. **Third project: Coding deep ResNN**

Course Outline

- Fourth block: Parametrized Deep Neural Networks
  1. Regularization and parameterization
  2. Convolution Neural Networks (CNNs) and PDE's
  3. Computing derivatives
  4. Programming CNN

- If we have time block
  1. Recurrent Networks and nonlinear data assimilation
  2. Instantaneous control and 4D-Var

Neural Networks - A quick overview

- Neural Networks with a particular (deep) architecture
- Exist for a long time (70's and even earlier), (Lecun, Hinton)
- Recent revolution - computational power and lots of data
- Can perform very well for large amounts of data
- Applications
  - Image classification
  - Face recognition
  - Segmentation
  - Driverless cars

Neural Networks - A quick overview

- Neural Networks with a particular (deep) architecture
- Exist for a long time (70's and even earlier), (Lecun, Hinton)
- Recent revolution - computational power and lots of data
- Can perform very well for large amounts of data
- Applications
  - Image classification
  - Face recognition
  - Segmentation
  - Driverless cars
- A few recent news articles:
  - Apple Is Bringing the AI Revolution to Your iPhone, WIRED 2016
  - Why Deep Learning Is Suddenly Changing Your Life, FORTUNE 2016
  - Data Scientist: Sexiest Job of the 21st Century, Harvard Business Rev 17

NN - A quick overview

Neural Networks is a data interpolator/classifier when the underline model is unknown.
A generic way to write it is

$$\mathbf{c} = f(\mathbf{y}, \boldsymbol{\theta}).$$

- The function f is the computational model.
- $\mathbf{y}$ is the input data (e.g., the example)
- $\mathbf{c}$ is the output (e.g. class of example)
- $\boldsymbol{\theta}$ are parameters of the model f

In a learning process, we have a examples $\{\mathbf{y}_j, \mathbf{c}_j\}$ and the goal is to estimate or "learn" the parameters $\boldsymbol{\theta}$

Learning from data - the core of science

How to choose f?

Fundamental(?) understanding, example: Newton's formula

$$x(t) = \frac{1}{2} g t^2$$

g unknown parameter
To estimate g observe falling object

t	x
0	0
1	4.9
2	20.1
3	44.1

What is the optimal value for g?

Learning from data - the core of science

How to choose f?

Phenomenological models, example: Archie's law - what is the electrical resistivity of a rock and how it relates to its porosity, ϕ and saturation, S_w?

$$\rho(\phi, S_w) = a\phi^{n/2} S_w^p$$

a, n, p unknown parameters

Obtaining parameters from observed data and lab experiments on rocks

Phenomenological vs. Fundamental

Fundamental laws come from understanding(?) the underlying process. They are **assumed invariant** and can therefore be predictive(?).

Phenomenological models are data driven. They "work" on some given data. Hard to know what are the limitations.

But ...

- Models based on understanding can do poorly - weather, economics ...
- Models based on data can sometimes do better
- How do we quantify understanding?

Generalization

Suppose that we have examples $\{\mathbf{y}_j, \mathbf{c}_j\}, \quad j = 1, \ldots, n$, a model $f(\mathbf{y}, \boldsymbol{\theta})$ and some optimal parameter $\boldsymbol{\theta}^*$.
Let $\{\mathbf{y}_j^t, \mathbf{c}_j^t\}, \quad j = 1, \ldots, s$ be some test set, that was not used to compute $\boldsymbol{\theta}$.

Generalization

Suppose that we have examples $\{\mathbf{y}_j, \mathbf{c}_j\}, \quad j = 1, \ldots, n$, a model $f(\mathbf{y}, \boldsymbol{\theta})$ and some optimal parameter $\boldsymbol{\theta}^*$.
Let $\{\mathbf{y}_j^t, \mathbf{c}_j^t\}, \quad j = 1, \ldots, s$ be some test set, that was not used to compute $\boldsymbol{\theta}$.
Loosely speaking, if

$$\|f(\mathbf{y}_j^t, \boldsymbol{\theta}^*) - \mathbf{c}_j^t\|_p$$

is small then the model is predictive - it generalizes well

Generalization

Suppose that we have examples $\{\mathbf{y}_j, \mathbf{c}_j\}, \quad j = 1, \ldots, n$, a model $f(\mathbf{y}, \boldsymbol{\theta})$ and some optimal parameter $\boldsymbol{\theta}^*$.
Let $\{\mathbf{y}_j^t, \mathbf{c}_j^t\}, \quad j = 1, \ldots, s$ be some test set, that was not used to compute $\boldsymbol{\theta}$.
Loosely speaking, if

$$\|f(\mathbf{y}_j^t, \boldsymbol{\theta}^*) - \mathbf{c}_j^t\|_p$$

is small then the model is predictive - it generalizes well

For Phenomenological models, there is no reason why the model should generalize, but in practice it often does.

Generalization

Why would a model generalize poorly?

$$1 \ll \| f(\mathbf{y}_j^t, \boldsymbol{\theta}^*) - \mathbf{c}_j^t \|_p$$

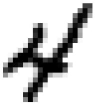
- Our "optimal" $\boldsymbol{\theta}^*$ was optimal for the training but is less so for other data
- The chosen computational model f is poor (e.g. linear model for a nonlinear function).

Example 1 - Classification of hand writing

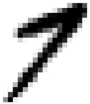
- Let $\mathbf{y}_j \in \mathbb{R}^n$ and let $\mathbf{c}_j \in \mathbb{R}^k$.
- The vector $\mathbf{c}$ is the probability if $\mathbf{y}$ belong to a certain class
- $0 \leq \mathbf{c}_j \leq 1$ and $\sum_j \mathbf{c}_j = 1$.

Examples (MNIST):

$\mathbf{y}_1$



$\mathbf{y}_2$



$\mathbf{c}_1 = [0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]$ $\quad$ $\mathbf{c}_2 = [0\ 0.3\ 0\ 0\ 0\ 0.7\ 0\ 0\ 0]$

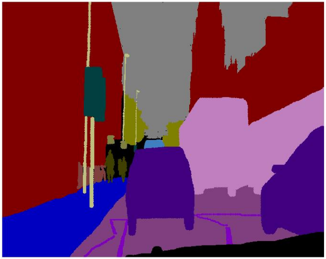
Example 2 - Classification of natural images

Same problem but images are natural images

Example 3 - semantic segmentation

y, input image

c, segmentation (labeled image)



Goal: Find map $\mathbf{c} = f(\mathbf{y}, \boldsymbol{\theta})$

Example 3 - semantic segmentation

Problem: Given image $\mathbf{y}$ and label $\mathbf{c}$ find a map $f(\cdot, \boldsymbol{\theta})$ such that $\mathbf{c} \approx f(\mathbf{y}, \boldsymbol{\theta})$

Example 3 - semantic segmentation

Problem: Given image $\mathbf{y}$ and label $\mathbf{c}$ find a map $f(\cdot, \boldsymbol{\theta})$ such that $\mathbf{c} \approx f(\mathbf{y}, \boldsymbol{\theta})$

Make the problem simpler

- Extract features from the image
- Classify in the feature space

Reduces the problem of learning from the image to feature detection and classification

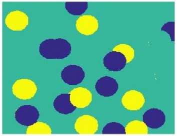
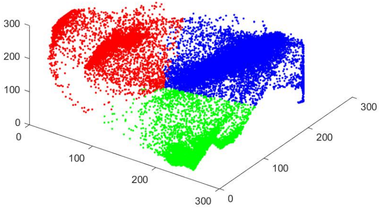
Possible features - color, neighbors, edges ...

Reduce the problem to low dimensional one

Example 3 - semantic segmentation

Simpler setup

- Data, $\mathbf{y}$ is the rgb value of the pixel (and its neighbors?)
- $\mathbf{c}$ is a labeled pixel
- The map $\mathbf{c} = f(\mathbf{y}, \boldsymbol{\theta})$

Data sets (to be downloaded and used for the course)

MNIST

CIFAR10

CamVid (can be downloaded from mathworks web page).