Introduction

Numerical Methods for Deep Learning

Learning From Data: The Core of Science - 1

Given inputs and outputs, how to choose f?

Option 1 (Fundamental(?) understanding): For example, Galileo's law of motion

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

with unknown parameter g.

Learning From Data: The Core of Science - 1

Given inputs and outputs, how to choose f?

Option 1 (Fundamental(?) understanding): For example, Galileo's law of motion

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

with unknown parameter g.
To estimate g observe falling object

t	x
0	0
1	4.9
2	20.1
3	44.1

Goal: Derive model from theory, calibrate it using data.

Learning From Data: The Core of Science - 2

Given inputs and outputs, how to choose f?

Option 2 (Phenomenological models): For 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.

Goal: Find model that consistent with fundamental theory, without directly deriving it from theory.

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 their limitations are.

But ...

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

Deep Neural Networks: History

- Neural Networks with a particular (deep) architecture
- Exist for a long time (70's and even earlier) [11, 12, 9]
- Recent revolution - computational power and lots of data [1, 10, 8]
- Can perform very well when trained with lots of data
- Applications
  - Image recognition [5, 7, 8], segmentation, natural language processing [2, 3, 6]

Deep Neural Networks: History

- Neural Networks with a particular (deep) architecture
- Exist for a long time (70's and even earlier) [11, 12, 9]
- Recent revolution - computational power and lots of data [1, 10, 8]
- Can perform very well when trained with lots of data
- Applications
  - Image recognition [5, 7, 8], segmentation, natural language processing [2, 3, 6]
- 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

Learning Objective: Demystify Deep Learning

Artificial Intelligence / Machine Learning

The Dark Secret at the Heart of AI

No one really knows how the most advanced algorithms do what they do. That could be a problem.

by **Will Knight** Apr 11, 2017

Learning objectives of this minicourse:

- look under the hood of some deep learning examples
- describe deep learning mathematically (see also [4])
- expose numerical challenges / approaches to improve DL

DNN - A Quick Overview - 1

Neural networks are data interpolator/classifier when the underlying model is unknown.

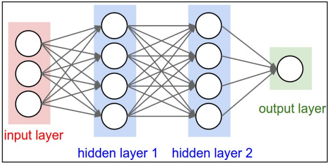
A generic way to write it is

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

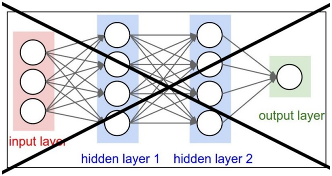
- the function f is the computational model
- $\mathbf{y} \in \mathbb{R}^{n_f}$ is the input data (e.g., an image)
- $\mathbf{c} \in \mathbb{R}^{n_c}$ is the output (e.g. class of the image)
- $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$ are parameters of the model f

In supervised learning we have examples $\{(\mathbf{y}_j, \mathbf{c}_j) \ : \ j = 1, \ldots, n\}$ and the goal is to estimate or "learn" the parameters $\boldsymbol{\theta}$.

DNN - A Quick Overview - 2



DNN - A Quick Overview - 2



$$
\begin{cases}
\mathbf{y}_{l+1} & = & \sigma(\mathbf{K}_l \mathbf{y}_l + \mathbf{b}_l) \\
\mathbf{y}_{l+1} & = & \mathbf{y}_l + \sigma(\mathbf{K}_l \mathbf{y}_l + \mathbf{b}_l) \\
\mathbf{y}_{l+1} & = & \mathbf{y}_l + \sigma\left(\mathbf{L}_l \sigma(\mathbf{K}_l \mathbf{y}_l + \mathbf{b}_l)\right) \\
 & \vdots &
\end{cases}
$$

Here:

- $l = 0, 1, 2, \ldots, N$ is the layer
- $\sigma : \mathbb{R} \to \mathbb{R}$ is the activation function
- $\mathbf{y}_0 = \mathbf{y} \in \mathbb{R}^{n_f}$ is the input data (e.g., an image)
- $\mathbf{c} \in \mathbb{R}^{n_c}$ is the output (e.g. class of the image)
- $\mathbf{L}_l, \mathbf{K}_l, \mathbf{b}_l$ are parameters of the model f

Machine Learning in 3 slides

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. (wiki)

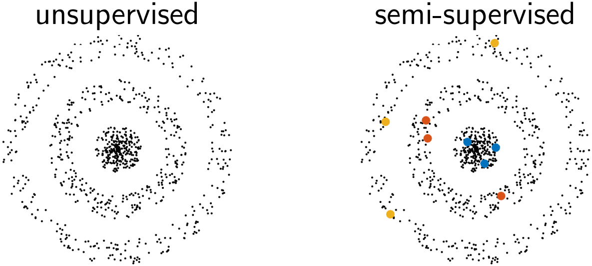
Machine Learning in 3 slides

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. (wiki)

Two common tasks in machine learning:

- given data, cluster it and detect patterns in it (unsupervised learning)
- given data and labels, find a functional relation between them (supervised learning)

Machine Learning in 3 slides



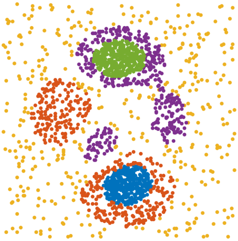
Unsupervised learning - given the data set $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]$ cluster the data into "similar" groups (labels).

- helps find hidden patterns
- often explorative and open-ended

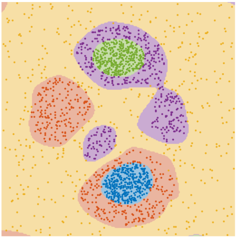
Semisupervised - label the data based on a few examples

Machine Learning in 3 slides

training data



trained model



Supervised learning - given the data set $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n] \in \mathcal{Y}$ and their labels $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_n] \in \mathcal{C}$, find the relation $f : \mathcal{Y} \to \mathcal{C}$

- models range in complexity
- older models based on support vector machines (SVM) and kernel methods
- recently, deep neural networks (DNNs) dominate

Generalization - 1

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 $\{(\mathbf{y}_j^t, \mathbf{c}_j^t) \ : \ j = 1, \ldots, s\}$ be some test set, that was not used to compute $\boldsymbol{\theta}^*$.

Generalization - 1

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 $\{(\mathbf{y}_j^t, \mathbf{c}_j^t) \ : \ j = 1, \ldots, s\}$ be some test set, that was not used to compute $\boldsymbol{\theta}^*$.

Loosely speaking, if

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

then the model is predictive - it generalizes well

Generalization - 1

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 $\{(\mathbf{y}_j^t, \mathbf{c}_j^t) \ : \ j = 1, \ldots, s\}$ be some test set, that was not used to compute $\boldsymbol{\theta}^*$.

Loosely speaking, if

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

then the model is predictive - it generalizes well

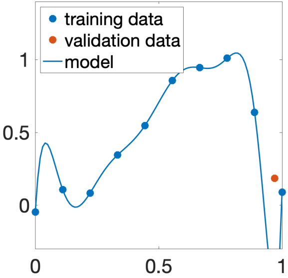
For phenomenological models, there is no reason why the model should generalize, but in practice it often does.

Generalization - 2

Why would a model generalize poorly?

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

Generalization - 2



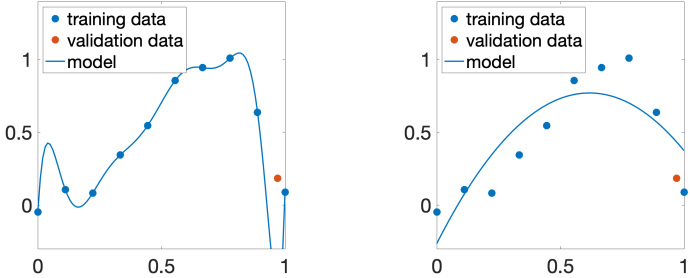
Why would a model generalize poorly?

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

Two common reasons:

1. Our “optimal” $\boldsymbol{\theta}^*$ was optimal for the training but is less so for other data

Generalization - 2



Why would a model generalize poorly?

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

Two common reasons:

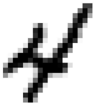
1. Our "optimal" $\boldsymbol{\theta}^*$ was optimal for the training but is less so for other data
2. The chosen computational model f is poor (e.g. quadratic model for a nonlinear function).

Example: Classification of Hand-written Digits

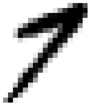
- Let $\mathbf{y}_j \in \mathbb{R}^{n_f}$ and let $\mathbf{c}_j \in \mathbb{R}^{n_c}$.
- The vector $\mathbf{c}$ is the probability of $\mathbf{y}$ belonging to a certain class. Clearly, $0 \leq \mathbf{c}_j \leq 1$ and $\sum_{j=1}^{n_c} \mathbf{c}_j = 1$.

Examples (MNIST):

$\mathbf{y}_1$



$\mathbf{y}_2$



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

Example: Classification of Natural Images

Image classification of natural images

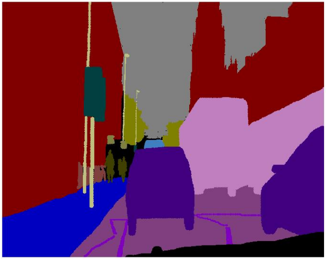
Examples (CIFAR-10):

Example: Semantic Segmentation - 1

- let $\mathbf{y}_j \in \mathbb{R}^n$ be an RGB or grey valued image.
- let the pixels in $\mathbf{c}_j \in \{1, 2, 3, \ldots\}^k$ denote the labels.

$\mathbf{y}$, input image

$\mathbf{c}$, segmentation (labeled image)



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

Example: Semantic Segmentation - 2

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: Semantic Segmentation - 2

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

First step: Reduce the dimensionality of problem.

- extract features from the image
- classify in the feature space

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

Example: Semantic Segmentation - 2

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

First step: Reduce the dimensionality of problem.

- extract features from the image
- classify in the feature space

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

Possible features: Color, neighbors, edges ...

Example: Semantic Segmentation - 3

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: Semantic Segmentation - 3

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

First step: Reduce the dimensionality of problem.

- extract features from the image
- classify in the feature space

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

Example: Semantic Segmentation - 3

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

First step: Reduce the dimensionality of problem.

- extract features from the image
- classify in the feature space

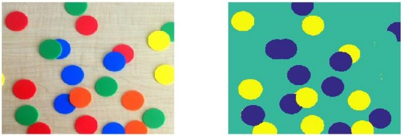
Reduce the problem of learning from the image to feature detection and classification

Possible features: Color, neighbors, edges ...

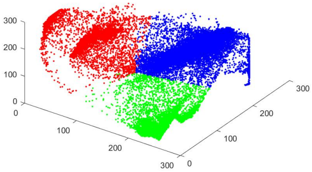
Example: Semantic Segmentation - 3

Simpler setup

- input: $\mathbf{y}$ is the RGB value of the pixel (and its neighbors?)
- output: $\mathbf{c}$ is a labeled pixel
- goal: map $\mathbf{c} = f(\mathbf{y}, \boldsymbol{\theta})$



input image and segmentation



3D representation of RGB values

References

[1] Y. Bengio et al. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[2] A. Bordes, S. Chopra, and J. Weston. Question Answering with Subgraph Embeddings. *arXiv preprint arXiv:1406.3676*, 2014.

[3] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[4] C. F. Higham and D. J. Higham. Deep Learning: An Introduction for Applied Mathematicians. *arXiv.org*, Jan. 2018.

[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[6] S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On Using Very Large Target Vocabulary for Neural Machine Translation. *arXiv preprint arXiv:1412.2007*, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 61:1097–1105, 2012.

[8] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

References (cont.)

[9] Y. LeCun, B. E. Boser, and J. S. Denker. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[10] R. Raina, A. Madhavan, and A. Y. Ng. Large-scale deep unsupervised learning using graphics processors. In *the 26th Annual International Conference*, pages 873–880. ACM, June 2009.

[11] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

[12] D. Rumelhart, G. Hinton, and J. Williams, R. Learning representations by back-propagating errors. *Nature*, 323(6088):533–538, 1986.