COMP 499/691 Deep Learning

Practical Matters

- Jan 18 to April 19
- Zoom sessions
- Labs - submission required
- 3 Problem sets
- Final Project — to be assigned by Week 5
- Recommended Textbook: Deep Learning by Goodfellow, Courville, Bengio — freely available online

Evaluation

- Assignments: 45%
  - Programming assignments (roughly 80% of assignments)
  - Some theoretical and written questions
- Project: 30%
- Quizzes: 15%
  - 4 Quizzes
- Labs: 10%
  - Each week

Pre-requisites

- Linear Algebra, Multivariable Calculus, Probability and Statistics, Algorithms
- Machine learning class or prior equivalent experience
- Experience with python and scientific toolkits (e.g. numpy,scipy,matplotlib, sklearn) is recommended

Odds and Ends

- Lab 1 is posted, due in 2 weeks. Subsequent labs will be due each week
- Office hours poll
- QA Forum and MS Teams
- Questions in zoom chat (directly to me)

Outline

- Introduction to Neural Networks and Representation Learning
- Backpropagation and Automatic Differentiation Software
- Optimization for Deep Learning
- Regularization and Implicit Regularization
- CNNs and Representation Learning
- Interpretability in Deep Learning (Guest Lecture)
- Generalization, Memorization, and Adversarial Examples

Outline

- RNNs and sequence models
- Attention and Self-Attention
- Multi-Task and Transfer Learning
- Deep Generative Models
- Deep Metric Learning
- Self-supervised Learning
- (If time allots) Basics of Deep Reinforcement Learning

What You Will Learn From This Class

- In-depth practical and theoretical understanding of the building blocks of deep learning models
- Overview of recent research areas in the field and open problems

Deep Learning

- Rebranding of (deep) Neural Networks - universal function approximator
- Representation Learning - avoid hand-crafted features
- Modular but powerful/expressive model framework

Applications

Autonomous Driving

Speech Recognition

Automated Shopping

Image from Wikimedia Commons

Machine Translation

Group Activity

- 10 minutes - In groups of 3-5 discuss
  - Come up with a list of 1-2 Deep Learning application that are not already mentioned
  - In your group decided on an application which you'd like to best understand after this course

Neural Networks

- Simple computation blocks

x_1, x_2 **Inputs**

w_1, w_2 **Weights**

b **Bias**

σ **Activation function**

h_1 **Neuron activation**

$$h(x1,x2) = \sigma(w_1 x_1 + w_2 x_2 + b)$$

Neural Networks

- Simple computation blocks that work together

$$h_1^{(2)} = \sigma(w_{11}^{(2)} h_1^{(1)} + w_{21}^{(2)} h_3^{(1)} + w_{31}^{(2)} h_3^{(1)} + b_1^{(2)})$$

Neural Networks

- Linear models can be thought of as neural networks

- Usually for feedforward case, deep networks refers to those with 2 or more hidden layers

Neural Networks Review

- Can be written as composition of simple linear operator + pointwise non-linearity

$$f_{\mathbf{W}_1,\mathbf{W}_2,\mathbf{W}_3}(x) = \mathbf{W}_3\rho(\mathbf{W}_2\rho(\mathbf{W}_1 x))$$

W_i **Matrix of parameters at layer I**

ρ **Pointwise non-lineartiy**

Neural Networks Review

- Can incorporate bias terms in W to simplify notation

$$h^{(2)} = \rho(\mathbf{W}_1 x)$$

$$\mathbf{W}_1 = \begin{bmatrix} \mathbf{W}_1' & \mathbf{b} \end{bmatrix} \quad x = \begin{bmatrix} x' \\ 1 \end{bmatrix}$$

- Aspects of neural networks have been historically inspired by biological systems

Images from Jean-Louis Queguiner

Biologically Plausible

Careful to not bring the biological analogy too far

- Biological neurons much more complex and have large variety

- Timing is (potentially) important in real neuronal activity (spike timing)

Lobo, J. Et al (2020). Spiking Neural Networks and online learning: An overview and perspectives. Neural Networks.

- Learning algorithm in brains is unknown! May not be anything like existing methods for NN
- Brains have feedback and recurrence
- Many other huge differences

Representational Power of Linear Models

$$f_W(x) = W_1 x$$

$$f_W(x) = W_2 \rho W_1 x$$

Linear Deep Network

$$f_{W_1,W_2,W_3}(x) = W_3\rho(W_2\rho(W_1x))$$

W_i **Matrix of parameters at layer I**

ρ **Identity function**

Can we approximate this?

Non-linear 1-hidden layer network

From Pascal Vincent and Hugo Larochelle's slides

Playground Tensorflow

https://playground.tensorflow.org/

Universal Approximation Theorem

$f_{target} : \mathscr{R}^d \rightarrow \mathscr{R}$ **Function we want to approximate**

$f_W(x) = W_2 \circ \rho \circ W_1 x$ **Approximation parametrized by** W_1, W_2

Under conditions on ρ **(satisfied for e.g. for sigmoid), there exists** W_1, W_2

which obtains at most ϵ **error**

$$\sup_{x \in [0,1]^d} |f_W(x) - f_{target}(x)| < \epsilon$$

Cybenko 1989

Why Deep?

Most functions of interest can be approximated with a single hidden layer network:

Why do we need multiple layers?

- Deeper networks can *much more efficiently* represent many functions (number of parameters)

1-hidden layer runs on

N-hidden layer

- Deeper networks, for some data, may bias the learning process towards more general solutions

Images from Wikimedia Commons

Capacity of Deep NN

- Several results exist showing that Deep NN are more parameter efficient
- Existing results are for restricted settings (on the target or network) or make use of assumptions

Eldan, Ronen, and Ohad Shamir. "The power of depth for feedforward neural networks." *Conference on learning theory*. 2016.

Montufar, Guido F., et al. "On the number of linear regions of deep neural networks." *Advances in neural information processing systems*. 2014.

Capacity of Deep NN

- (Montufar et al NIPS 2014) studies number of piecewise regions represented by ReLU networks and followed up by many works

Relu non-linearity

Relu network

$$
\begin{aligned}
\mathbf{h}^1 &= \max\{0, W^1\mathbf{x} + b^1\} \\
\mathbf{h}^l &= \max\{0, W^l\mathbf{h}^{l-1} + b^l\} \\
\mathbf{y} &= W^{L+1}\mathbf{h}^{\mathbf{L}}
\end{aligned}
$$

Another example

Example in 2d

(c)

Montufar, Guido F., et al. "On the number of linear regions of deep neural networks." *Advances in neural information processing systems*. 2014.

Serra T. et al Bounding and Counting Linear Regions of Deep Neural Networks. Neurips 2017

Capacity of Deep NN

- Relu networks create a piecewise linear functions

Eldan, Ronen, and Ohad Shamir. "The power of depth for feedforward neural networks." *Conference on learning theory*. 2016.

Montufar, Guido F., et al. "On the number of linear regions of deep neural networks." *Advances in neural information processing systems*. 2014.

Serra T. et al Bounding and Counting Linear Regions of Deep Neural Networks. Neurips 2017

Capacity of Deep NN

- (Montufar et al NIPS 2014)

- Number of linear regions in the output increase exponentially with depth and polynomial in width

$$\Omega((\frac{n}{d})^{(L-1)d} \; n^d)$$

L hidden layers

d input dimension

n width -- number of hidden units per layer

Hanin, Boris, and David Rolnick. "Complexity of linear regions in deep networks." *arXiv preprint arXiv:1901.09021* (2019).

Montufar, Guido F., et al. "On the number of linear regions of deep neural networks." *Advances in neural information processing systems*. 2014.

Depth Helps Generalization (Optional)

- Common conjecture based on empirical observations with some existing, but limited theoretical backing

- Some relevant work:

Urban, Gregor, et al. "Do deep convolutional nets really need to be deep and convolutional?." *arXiv preprint arXiv:1603.05691* (2016).

Arora, Sanjeev, et al. "Implicit regularization in deep matrix factorization." *Advances in Neural Information Processing Systems*. 2019.

Pérez, Guillermo Valle, Chico Q. Camargo, and Ard A. Louis. "Deep learning generalizes because the parameter-function map is biased towards simple functions." ICLR 2019

Deep Learning

- Rebranding of (deep) Neural Networks - universal function approximator
- Representation learning - avoid hand-crafted features
- Modular but powerful model framework

Supervised Representation Learning

- Linear models are fast and easy to use
- Designing hand crafted feature is thus tempting
- Difficult for humans to fully describe models of perception and relevant features they use — representation learning can fill this gap

Representation Learning

- We can interpret intermediate hidden layer outputs as learned features

Photo from Wikimedia Commons

Representation Learning

- A key feature of representation learning is reusability, features from one category can be relevant in another suggesting generality

Photo from Wikimedia Commons

Representation Learning

- In supervised settings we can consider final layer as encouraging features that linearly separate the data

Representation Learning

- In order to optimize the learning objectives with a given deep architecture the learning process can discover relevant features of the data

- Intermediate representations in deep networks can find features similar to those designed by humans

- However they can also learn ones we wouldn't think of

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013] From Y Lecun slides

Representation Learning Re-usability

- A goal of representation learning is to be useful on new tasks and datasets
- With enough varied data representation learning can often outperform hand crafted feature
    - Imagenet learned features more useful in most computer vision tasks than hand-crafted features
- Can reuse representations in multiple ways

Unsupervised Rep. Learning

- Unsupervised representation learning simple example, auto encoders

Photo from Wikimedia Commons

Unsupervised Rep. Learning

- Unsupervised representation learning has been a major driver of deep learning research since 2006

- Starting to show promising results since 2018

Deep Learning

- Rebranding of (deep) Neural Networks - universal function approximator
- Representation learning - avoid hand-crafted features
- Modular framework for high capacity models

A Modular Framework

- Gradient descent and back-propagation algorithm leads to set of tools for jointly adapting modular system for a single objective

A Modular Framework

Toolbox

Assemble model components from toolbox

Photo from Wikimedia Commons

A Modular Framework

- Software frameworks for adjusting all model components is set
- Adding priors is easy and learning is formulaic
- Flexible framework to build models with strong approximation ability

Large Data and Compute

- Deep learning models and frameworks are ripe to take advantage of larger datasets and increasing compute

Images from Nvidia Inc

More data + high capacity model

10,000 samples

1 Million samples

Bigger Models Keep Being Better

- Original Imagenet dataset of ~1.2 million images released in 2010

- For architecture class bigger model increases performance — power law
- More efficient architectures shift the curve over time

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

Mingxing Tan [1] Quoc V. Le [1]

Neural Scaling “Laws” (Hypothesis)

- Empirically observed that models continue to improve with more data and bigger (parameters) versions of existing model classes
- Performance gains follow a power law
- Recent language models continue to improve performance
- Observations have been consistent even a decade since the first large dataset results with deep learning models

More Data Image Recognition

Revisiting Unreasonable Effectiveness of Data in Deep Learning Era

Chen Sun[1], Abhinav Shrivastava[1,2], Saurabh Singh[1], and Abhinav Gupta[1,2]

[1]Google Research
[2]Carnegie Mellon University

Representations keep improving

Language Modeling

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
OpenAI
sam@openai.com

Tom Henighan
OpenAI
henighan@openai.com

Tom B. Brown
OpenAI
tom@openai.com

Benjamin Chess
OpenAI
bchess@openai.com

Rewon Child
OpenAI
rewon@openai.com

Scott Gray
OpenAI
scott@openai.com

Alec Radford
OpenAI
alec@openai.com

Jeffrey Wu
OpenAI
jeffwu@openai.com

Dario Amodei
OpenAI
damodei@openai.com

Where do these empirical scaling observations break down?

Summary of some Trends

- More compute/acc efficient models due to modularity/ease of use -> continues to show improve results on many tasks
- More data + compute -> show improved results on many tasks
- Bigger models + compute -> show improved results on many tasks

When do we not use Deep Learning

- Many machine learning problems do not require deep learning or huge models
- Training speed — deep learning is slow
  - Logistic regression, Random Forests, Gradient boosting, and related can often be fit on a laptop for mid-sized problems, including hyperparameter search
- Linearly separable problems

When do we not use Deep Learning

- Smaller datasets, without external related data, sometimes more easily solved with hand-crafted features + simpler classifiers
- Good feature extractors exist
  - Deep Learning methods particularly excel on perceptual data (images, speech, language)
- Interpretability is critical
- Tabular data can often be solved equally well by other methods
- This is a non-exhaustive list, there are many other situations as well!