11-785 – Deep Learning

Key things to understand.

A pedagogical tool to track the key aspects of the lecture that the instructor emphasises, and as a checklist of things you have judged are important.

Week 1

Lecture 0 - Logistics

Lecture 1 – Introduction

- Understand what neural networks are
- Understand their neurophysiological basis.
- Understand the philosophical and psychological context of neural networks – associationism, proto-connectionism, connectionism.
- Understand recent historical developments – McCulloch-Pitts, Hebbian learning, Rosenblatt perceptrons, Minsky-Papert and XOR controversy.
- Understand the various capabilities of multilayer perceptrons (MLPs) as connectionist computational models, Boolean functions, Boolean machines.

Lecture 2 – Neural networks as universal approximators

- Understand that MLPs are networks of perceptrons.
- Understand the definitions of layer and depth of MLPs.
- Understand MLP capabilities in terms of universal Boolean functions, universal classifiers over real inputs, universal approximators, optimal depth-width.
- MLPs as universal Boolean functions:
    - Logic gates and perceptrons.
    - Truth tables, disjunctive normal form, Boolean functions.
    - Reduced DNF and Karnaugh maps.
    - Understand how these tools yield heuristic arguments regarding depth and width in this context.
    - Be acquainted with relevant research findings in this area.
    - Understand caveats and nuances of these findings.
- MLPs as universal classifiers over real inputs:
    - Understand the heuristic examples regarding the approximation of complex decision boundaries.
- MLPs as universal approximators.
- Optimal depth and width of MLPs:
    - Understand that there are requisite depth/width constraints on expressive power of MLPs.
    - Understand that the above feature as the "sufficiency of architecture".
    - Understand the interpretation of information propagation, loss.
    - Understand the role of appropriate activation functions in the depth-width tradeoff.
    - Be acquainted with various definitions of capacity/expressive power.

- *Make sure you understand the heuristic examples here, lots covered quickly.*

Week 2

Lecture 3 – Training the network (part 1)

- Understand feedforward architecture.
- Understand that learning a neural network corresponds to a statistical estimation problem (of weights and biases) to represent an unknown function.
- Understand that tuning parameters corresponds to minimisation of an integral over divergences.
- Understand the role of training data as noisy sampling from an unknown function we wish to represent.
- Understand the perceptron learning algorithm, and the Novikov's perceptron convergence theorem.
- Appreciate the theoretical difficulties of training a neural network using a series of perceptron learning algorithms as one of an NP-hard combinatorial optimisation problem.
- Understand the role of differentiable activation functions.
- Understand the relation between a perceptron with sigmoid activation and logistic regression.
- Understand the role of expected error, empirical estimates of expected error, empirical risk minimisation.
- Be fluent in scalar function optimisation using elementary calculus.
- Be fluent with tools for evaluating derivatives of multivariate scalar functions, such as gradients, Hessians, and their properties.
- Be fluent in unconstrained optimisation of a multivariate scalar function.
- Understand the role of non-convexity, numerical iterative methods, and gradient descent.

- *Appreciate the historical significance of ADALINE and MADALINE.*
- *Better understand the formal properties of the gradient in context of vector calculus, directional derivatives, and level sets.*

Week 3

Lecture 4 – Backpropagation

- Understand the details of the components of each element of a neural network:
    - Training data as input-output pairs.
    - The loss/error function.
    - Weight and bias parameters.
    - Affine combinations.
    - Activation functions and their derivatives.
    - Vector representation of the neural network.
    - Input representation.
    - Output representations for binary classification, multiclass classification.
- Understand the affinity between the problem setting, output activation function, and error/loss/divergence for regression; and the binary and multiclass classification settings.
- Understand the technique of label-smoothing.
- Understand gradient descent and its pseudocode.
- Understand that derivatives and the chain rule can be viewed as a route of influence from a source to a destination through a topological network with influence diagrams.
- Be fluent with the chain rule for univariate and multivariate functions.
- Be fluent with the forward and backward pass of the backpropagation/reverse-mode automatic differentiation algorithm.

Lecture 4.5 – Backpropagation (part 2)

- Understand a number of issues that have been backgrounded thus far:
- Understand the role of vector activations, and the consequence for derivative calculations.
- Understand that assuming affine combinations does not necessarily hold for other NN architectures.
- Understand the issue of non-differentiable activation functions, and the use of subgradients.
- Be fluent with the use of Jacobian matrices for derivatives of vector functions with respect to vector inputs.
- Understand their differing properties for scalar and vector activations functions.
- Be fluent with the chain rule cast in terms of gradients, Jacobians, and combinations.
- Be fluent with the vector representation of the NN, and the vector representation of forward and backward passes of the backpropagation algorithm.

- *Understand how techniques are adapted to accommodate multiplicative, rather than affine combinations.*

<u>Lecture 5 – Convergence</u>

- Understand the neural network training algorithm pseudocode.
- Understand heuristic comparisons of the performance of backpropagation and the perceptron learning algorithm when additional training instances are added.
- Understand the implications of these arguments for the MLPs.
- Understand how these are framed in terms of bias-variance.
- Understand comparisons of the backpropagation trained classifier and optimal classifiers.
- Understand visually that the loss or error surface can be visualised as a complex multi-dimensional manifold.
- Be aware that visualisation of an error surface for various neural networks is an open question.
- Understand that neural network error surfaces are highly non-convex
- Understand the distinction between global, local minima and maxima; and saddle points.
- Understand the role of convex optimisation as well-studied and as providing a useful reference point.
- Understand the how convergence of an iterative algorithm is quantified theoretically through a convergence rate, and the properties of differing rates.
- Understand how contour plots in parameter space can help visualise convergence, oscillating convergence, and divergence.
- Understand how the learning rate or step-size parameter in gradient descent relates to convergence for quadratic error surfaces.
- Understand the relation between Taylor expansions, Newton's method, and the relation between the learning rate and 2nd derivative for generic differentiable, convex objectives.
- Understand the nuances and heuristic arguments on convergence for differing characterisations of convex functions, and as dimensions increase.
- Understand the role of rescaling axes to provide a normalised gradient descent rule, using Hessians.
- Understand the computational infeasibility of using Hessians.
- Be aware of research relating to 2nd order methods to approximate Hessians.
- Understand the use of a single learning rate for all parameters, and implications for convergence.
- Understand that heuristic arguments presented may not hold for non-convex error surfaces.
- Understand why decaying learning rates may be used.

- *Review paper on comparisons of backpropagation and the perceptron learning algorithm by Brady et al. (1989).*
- *Weight normalisation - Salimans, Kingma (2016)*

Week 4

Lecture 6 – Convergence in neural networks

- Understand the motivations for the use of resilient propagation (Rprop) and Quickprop algorithms.
- In particular, understand the motivation of the above as updating each component of the parameter separately.
- Understand the mechanics, properties, and pseudocode for Rprop and Quickprop.
- Understand that Rprop and Quickprop can be viewed as making use of approximate information from the gradient and Hessian respectively; and as derivative-inspired algorithms.
- Understand the issues related to making component-wise updates of the parameter, such as those of divergence and convergence in distinct directions.
- Understand the motivation for the use of momentum methods.
- Understand that momentum methods use running averages.
- Understand the mechanics, properties and pseudocode of momentum methods.
- Understand the motivation for the use of Nesterov's accelerated gradient method, together with mechanics, properties and pseudocode.
- Understand that the methods can be viewed as means of improve on the convergence of gradient descent.

- Understand the that there are design choices associated with the size of the training set to be processed before parameter updates are carried out.
- Have a heuristic understanding of the computational issues and algorithmic (mis)behaviour of choosing to do full-batch update of parameters and incremental updates as a means of motivating stochastic gradient descent (SGD).
- Understand the pseudocode for stochastic gradient descent.
- Understand heuristic reasons for why the learning rate should be reduced when implementing SGD.
- Understand formal results regarding sufficient conditions for the convergence of SGD for convex and non-convex loss functions.
- Understand formal convergence properties of batch gradient descent and SGD.
- Understand the statistical relation between the empirical and expected error i.e. through bias and variance of an estimator.
- Understand batch, single-instance SGD, and mini-batch SGD in terms of variance.
- Understand the motivations of, pseudocode for, and formal convergence properties of mini-batch gradient descent.

Lecture 7 – Stochastic gradient decent, overfitting, tricks

- Understand the motivations for recent trend-based methods as building on mini-batch SGD.
- Understand how trend-based methods such as momentum and Nesterov's accelerated gradient can be combined with incremental updates to perform a variance smoothing function.

- Understand the motivations and design principles of more recent trend-based methods to smooth out the variation for mini-batch SGD.
- Understand the properties, mechanics and pseudocode for RMS Prop.
- Understand the properties, mechanics and pseudocode for ADAM (RMS Prop with momentum).
- Understand the properties of desirable loss functions, and that loss surfaces in deep learning will often be very complex.
- Understand the problem of covariate shift as a motivation for batch normalisation.
- Understand the procedure of batch normalisation.
- Understand in detail how batch normalisation affects backpropagation calculations.
- Understand the nuances of dealing with high-dimensional data in context of the number of training instances required to represent a function; and related issues such as over-fitting.
- Understand the motivations for the introduction of smoothing constraints.
- Understand, at the level of the individual unit in a simple neural network, how overfitting happens.
- Understand that L2 and L1 regularisation, or weight decay can be viewed as imposing smoothness constraints.
- Be aware of results that deeper neural networks may impose smoothness constraints, and the heuristic arguments for why this may be the case.

- *Review papers on ADAM, ADAGrad, batch-normalisation*