Differentiate your Objective

COMP6248 Differentiable Programming
(and some Deep Learning)

Jonathon Hare
Vision, Learning and Control
University of Southampton
Machine Learning - A Recap

Data \( \{x_n, y_n\}_{n=1}^{N} \quad \{x_n\}_{n=1}^{N} \)

Function Approximator \( y = f(x, \theta) + \nu \)

Parameter Estimation \( E_0 = \sum_{n=1}^{N} \|y_n - f(x_n; \theta)\|^2 \)

Prediction \( \hat{y}_{N+1} = f(x_{N+1}, \hat{\theta}) \)

Regularisation \( E_1 = \sum_{n=1}^{N} \|y_n - f(x_n; \theta)\|^2 + r(\|\theta\|) \)

Modelling Uncertainty \( p(\theta|\{x_n, y_n\}_{n=1}^{N}) \)

Probabilistic Inference \( \mathbb{E}[g(\theta)] = \int g(\theta)p(\theta)d\theta = \frac{1}{N_s} \sum_{n=1}^{N_s} g(\theta^{(n)}) \)

Sequence Modelling \( x_n = f(x_{n-1}, \theta) \)

What is Deep Learning?

Deep learning is primarily characterised by function compositions:

- Feedforward networks: \( y = f(g(x, \theta_g), \theta_f) \)
  - Often with relatively simple functions (e.g. \( f(x, \theta_f) = \sigma(x^\top \theta_f) \))

- Recurrent networks:
  \( y_t = f(y_{t-1}, x_t, \theta) = f(f(y_{t-2}, x_{t-1}, \theta), x_t, \theta) = \ldots \)

In the early days the focus of deep learning was on learning functions for classification. Nowadays the functions are much more general in their inputs and outputs.
What is Differentiable Programming?

- Differentiable programming is a term coined by Yann Lecun\(^1\) to describe a superset of Deep Learning.
- Captures the idea that computer programs can be constructed of parameterised functional blocks in which the parameters are learned using some form of gradient-based optimisation.
  - The implication is that we need to be able to compute gradients with respect to the parameters of these functional blocks. We’ll start explore this in detail next week...
  - The idea of Differentiable Programming also opens up interesting possibilities:
    - The functional blocks don’t need to be direct functions in a mathematical sense; more generally they can be \textit{algorithms}.
    - What if the functional block we’re learning parameters for is itself an algorithm that optimises the parameters of an internal algorithm using a gradient based optimiser?!\(^2\)

\(^1\)https://www.facebook.com/yann.lecun/posts/10155003011462143

Is all Deep Learning Differentiable Programming?

- Not necessarily!
  - Most deep learning systems are trained using first order gradient-based optimisers, but there is an active body of research on gradient-free methods.
  - There is an increasing interest in methods that use different styles of learning, such as Hebbian learning, within deep networks. More broadly there are a number of us\(^3\) who are interested in biologically motivated models and learning methods.
  - This course will primarily focus on differentiable methods, but we’ll look at how relaxations can be made to make non-differentiable operators learnable with gradient-based optimisers.

\(^3\)including at least myself, my PhD students and Geoff Hinton!
Why should we care about this?

![Graph showing performance vs. amount of data for Traditional Learning and Large Neural Networks.]

Reference: Andrew Ng

Success stories - Object detection and segmentation

Success stories - Image generation


Success stories - Translation

ENGLISH TEXT
The reason Boeing are doing this is to cram more seats in to make their plane more competitive with our products,” said Kevin Keniston, head of passenger comfort at Europe's Airbus.

TRANSLATED TO FRENCH
La raison pour laquelle Boeing fait cela est de crier plus de sièges pour rendre son avion plus compétitif avec nos produits”, a déclare Kevin Keniston, chef du confort des passagers chez Airbus.

What is the objective of this module?

A word of warning: This is not a module about how to apply someone else’s deep network architecture to a task, or how to train existing models!

You will learn some of that along the way of course, but the real objective is for you to graduate knowing how to understand, critique and implement new and recent research papers on deep learning and associated topics.

- To gain an in-depth theoretical and practical understanding of modern deep neural networks and their applications.
- Understand the underlying mathematical and algorithmic principles of deep learning.
- Understand the key factors that have made differentiable programming successful for various applications.
- Apply existing deep learning models to real datasets.
- Gain facility in working with deep learning libraries in order to create and evaluate network architectures.
- Critically appraise the merits and shortcomings of model architectures on specific problems.
How is this module going to be delivered?

- Lectures (2 per week)
  - Note: I am refreshing some material from last year, but the website still has links to the old slides.
  - You need to read the suggested papers/links before the lectures!
  - There is a little room for some flexibility later in the course on topics - tell me what you’re interested in!

- Labs (1x 2 hour session per week for 8 weeks + additional help sessions)
  - Labs consist of a number of Jupyter notebooks you will work through.
    - You’ll be using PyTorch as the primary framework, with Torchbearer to help out.
    - You will need to utilise GPU-compute for the later labs (we provide Google Colab links so you can use NVidia K80s in the cloud, but you’ll also be able to use the RTX2070 in the lab machines if you wish).
  - I have scheduled lab sessions so you can get help and ask questions from myself and the demonstrator team.
  - After each lab you will have to do a follow-up exercise that will be marked.

What will we cover in the module?

http://comp6248.ecs.soton.ac.uk/
Lab session plan

<table>
<thead>
<tr>
<th>Lab</th>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lab 1</td>
<td>05/02/20</td>
<td>Introducing PyTorch</td>
</tr>
<tr>
<td>Lab 2</td>
<td>12/02/20</td>
<td>Automatic Differentiation</td>
</tr>
<tr>
<td>Lab 3</td>
<td>19/02/20</td>
<td>Optimisation</td>
</tr>
<tr>
<td>Lab 4</td>
<td>26/02/20</td>
<td>NNs with PyTorch and Torchbearer</td>
</tr>
<tr>
<td></td>
<td>04/03/20</td>
<td>NO LAB</td>
</tr>
<tr>
<td>Lab 5</td>
<td>11/03/20</td>
<td>CNNs with PyTorch and Torchbearer</td>
</tr>
<tr>
<td>Lab 6</td>
<td>18/03/20</td>
<td>Transfer Learning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Break</td>
</tr>
<tr>
<td>Lab 7</td>
<td>22/04/20</td>
<td>RNNs, Sequence Prediction and Embeddings</td>
</tr>
<tr>
<td>Lab 8</td>
<td>29/04/19</td>
<td>Deep Generative Models</td>
</tr>
<tr>
<td></td>
<td>06/05/19</td>
<td>Coursework Help and Advice</td>
</tr>
<tr>
<td></td>
<td>13/05/19</td>
<td>Coursework Help and Advice</td>
</tr>
</tbody>
</table>

What do we expect you already know?

- COMP3223 or COMP6245 (fundamentals of statistical learning, MLPs, gradient descent, how to train and evaluate learning machines, supervised-vs-unsupervised)
- Fundamentals of:
  - Matrix Algebra (matrix-matrix, matrix-vector and matrix-scalar operations, inverse, determinant, rank, Eigendecomposition, SVD);
  - Probability & Statistics (1st-order summary statistics, simple continuous and discrete probability distributions, expected values, etc); and,
  - Multivariable Calculus (partial differentiation, chain-rule).
- Programming in Python
What might you already know?

- How to use a deep learning framework (Keras, Tensorflow, PyTorch)?
- How to train an existing model architecture using a GPU?
- How to perform transfer learning?
- How to perform differentiable sampling of a Multivariate Normal Distribution?

Assessment Structure

- Lab work 40% - Handin in week 11 (6th May, 4PM)
- Final project 40% - Handin in Week 12 (15th May, 4PM) (+ interim handin in week 6)
- Online quizzes 20% - Planned for Thursdays in Weeks 6 (5th-Mar) and 10 (30-Apr)
http://comp6248.ecs.soton.ac.uk/coursework.html