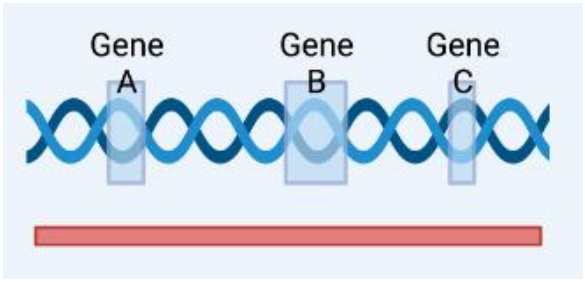
# COMP6258 Differentiable Programming and Deep Learning

Jonathon Hare and Antonia Marcu

**What do Differentiable Programming and Deep Learning give us?**





Differentiable Programming and Deep Learning give us the ability to computationally solve problems without needing to handcraft the solution or engineer the features. As long as we have data we can build and train models to make predictions.

**Machine Learning - A Recap**

| | | |
|---|---|---|
| Data | $\{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^N$ | $\{\boldsymbol{x}_n\}_{n=1}^N$ |
| Function Approximator | $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ | |
| Parameter Estimation | $E_0 = \sum_{n=1}^N \{\|\boldsymbol{y}_n - f(\boldsymbol{x}_n; \boldsymbol{\theta})\|\}^2$ | |
| Prediction | $\hat{\boldsymbol{y}}_{N+1} = f(\boldsymbol{x}_{N+1}, \hat{\boldsymbol{\theta}})$ | |

**What is Deep Learning?**

Deep learning is primarily characterised by function compositions:

- Feedforward networks: $\boldsymbol{y} = f(g(\boldsymbol{x}; \boldsymbol{\theta}_g); \boldsymbol{\theta_f})$
  - Often with relatively simple functions (e.g. $f(\boldsymbol{x}; \boldsymbol{\theta}_f) = \sigma(\boldsymbol{x}^\top \boldsymbol{\theta}_f)$)
- Recurrent networks: $\boldsymbol{y}_t = f(\boldsymbol{y}_{t-1}, \boldsymbol{x}_t; \boldsymbol{\theta}) = f(f(\boldsymbol{y}_{t-2}, \boldsymbol{x}_{t-1}; \boldsymbol{\theta}), \boldsymbol{x}_t; \boldsymbol{\theta}) = \ldots$

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.

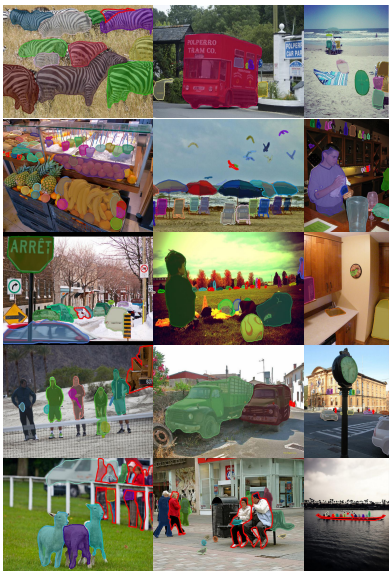
### What is Differentiable Programming?

- Differentiable programming is a term coined by Yann Lecun 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.
- So we need to be able to compute gradients with respect to the parameters of these functional blocks.
- The functional blocks don't need to be direct functions in a mathematical sense; more generally they can be *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?![1]

### 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[2] who are interested in biologically motivated models and learning methods.
    - There's a lot of recent research that computes biological proxies for gradients though!
  - 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.

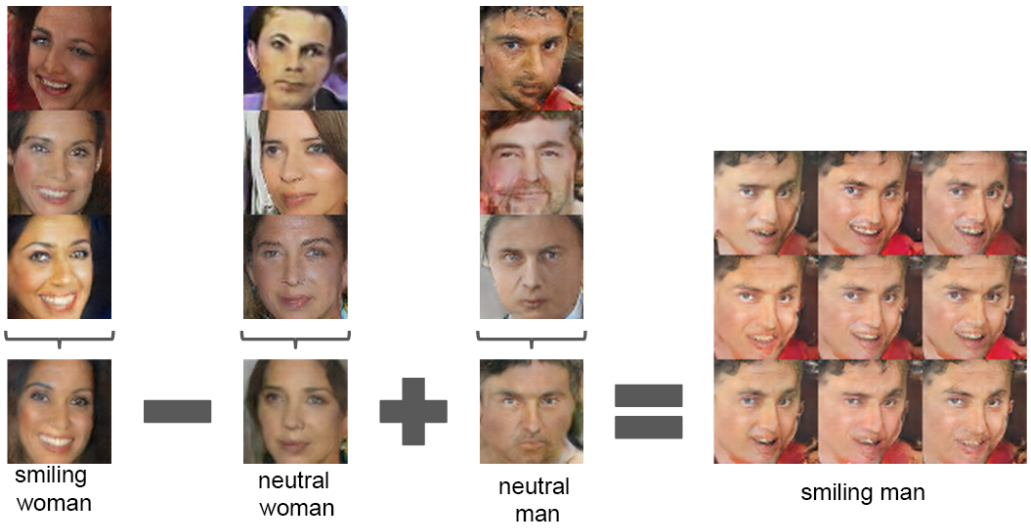
### Success stories - Object detection and segmentation



Pinheiro, Pedro O., et al. "Learning to refine object segments." European Conference on Computer Vision. Springer, 2016.

### Success stories - Image generation

[1] See our ICLR 2019 paper: https://arxiv.org/abs/1812.03928 and NeurIPS 2019 paper: https://arxiv.org/abs/1906.06565
[2] including at least myself, my PhD students and Geoff Hinton!




Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).

### Success stories - Translation

**GPT-4**

**Prompt:**
*Translate the following Irish text to English:*
*"Bhí mé ag siúl sa choill nuair a chuala mé ceol álainn ón adharc."*

**Output (With LLM Integration):**
*I was walking in the forest when I heard beautiful music from the horn.*
**Output (Traditional MT):**
*I was in the wood when I heard nice sound from the horn.*

Lyu, C., Du, Z., Xu, J., Duan, Y., Wu, M., Lynn, T., Aji, A.F., Wong, D.F., Liu, S. and Wang, L., 2023. A Paradigm Shift: The Future of Machine Translation Lies with Large Language Models. LREC-COLING 2024

Language translation has made massive leaps over the last 10 years as a result of deep learning. Initially this was because of large aligned datasets and translation models that targeted transforming a sequence of words in one language to another. As a result of large language models we are increasingly seeing translation quality improve in terms of the nuance and subjective feel of the translation — translations are starting to becoming more natural and less mechanistic (but possibly less objectively accurate as a result).

### Why should we care about this?

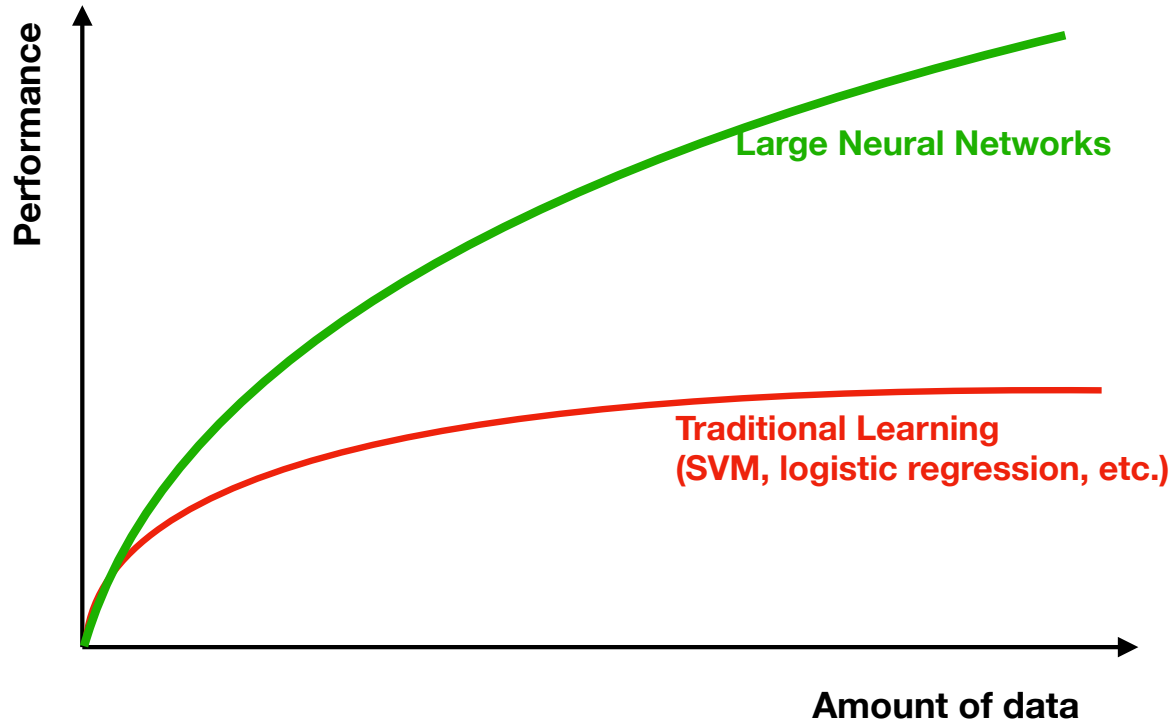



Image courtesy of Andrew Ng

The above picture shows how model performance has been shown to be linked to the amount of data. The story of the last 10-15 years has been one of increasingly growing data. Traditional approaches to machine learning have largely not scaled in performance in the same way large deep neural networks have. This is predominantly because of the way features are extracted from the data; traditional machine learning often used hand-crafted features, whereas the story of deep learning is all about feature learning — giving the model essentially raw data and letting it learn to appropriate features to make decisions from.

We should recognise however that traditional methods still have their place - when datasets are small, and perhaps there isn't much feature engineering to do, they can and do outperform deeper models. At the same time, a large proportion of deep network architectures are themselves nothing more than a learned linear model (linear regression, logistic regression, SVM, ...) attached to a learned feature extractor, so those very basic traditional methods are still very much a big part of the picture today!

**Key takeaways**

- **More data availability** in all domains...
- ...allows for **bigger/deeper/more complex models**
- ...models that can **learn the features**
- and these can have far **greater performance**

**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!** [1em] 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.

**What is the objective of this module?**

- Understand the underlying mathematical and algorithmic principles of deep learning.
- Apply deep learning methods to real datasets.
- Learn how to train models using deep learning libraries.
- Think critically about upcoming research in deep learning.

**How is this module going to be delivered?**

- Lectures (3 per week except when we have seminars)
  - Note: We are refreshing some material from last year, but the website may have old links.
  - You need to read the suggested papers/links before the lectures!
  - There is maybe a little room for some flexibility later in the course on topics - tell us what you're interested in!
  - Lectures will be face to face, but also recorded for the website when possible (do not rely on this!).

**How is this module going to be delivered?**

- Seminars (4)
  - We'll look at a particular paper, papers or ideas in some detail and have an open discussion
  - You'll need to prepare in advance & be ready to ask questions and share your thoughts
  - Not recorded, but contents examinable in the quizzes

**How is this module going to be delivered?**

- Labs (1x 2 hour session per week for 8 weeks + additional help sessions if required)
  - Labs consist of a number of Juypter notebooks you will work though.
    - 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 or newer in the cloud).
  - Labs are in-person (AICE lab in B32) with a team of PhD student demonstrators & both of us.
  - Please ask lots of questions and use this time to get help on the labs and coursework.
  - After each lab you will have to do a follow-up problem-sheet exercise that will be marked.

**What will we cover in the module?**

`https://moodle.ecs.soton.ac.uk/course/view.php?id=469`

**Lab session plan**

| Lab | Date | Topic |
|---|---|---|
| Lab 1 | 3/02/25 | Introducing PyTorch |
| Lab 2 | 10/02/25 | Automatic Differentiation |
| Lab 3 | 17/02/25 | Optimisation |
| Lab 4 | 24/02/25 | NNs with PyTorch and Torchbearer |
| Lab 5 | 03/03/25 | CNNs with PyTorch and Torchbearer |
| Lab 6 | 10/03/25 | Transfer Learning |
| Lab 7 | 17/03/25 | RNNs, Sequence Prediction and Embeddings |
| Lab 8 | 21/04/25 | Deep Generative Models |

**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**

- Interim project plan - Handin in week 5 (25th Feb, 4PM)
- Lab work 40% - Handin in week 11 (6th May, 4PM)
- Online quizzes 20% - Planned for week 6 (4th March) and week 12 (13th May)
- Final project 40% - Handin in week 12 (15th May, 4PM)

**The Main Assignment**

*The COMP6258 Reproducibility Challenge*

`https://moodle.ecs.soton.ac.uk/mod/page/view.php?id=23063`