ECE472, Deep Learning – Syllabus, Fall 2023, Room 502, W 6-9

tldr: We will introduce the concepts relevant to so called "deep learning" — our fundamental processes are based on computations performed over differentiable graphs, where nodes correspond to operations and edges correspond to operands. We will use the Microsoft Teams site: "ECE-472-1-Deep Learning-2023FA"

Instructor Chris Curro, EE '15, MEE '16; `professor@curro.cc`

Reference Textbook Ian Goodfellow and Yoshua Bengio and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org` (Note: the book may be a useful reference for the first five weeks of class, and not thereafter.)

Assignments There will be a handful of programming assignments.

Citations Plagiarism will not be tolerated. All cases of suspected plagiarism will be submitted to the Dean's office for investigation. Feel free to ask questions of your peers, but please cite them for any help you receive. Cite any resources utilize.

Quizzes There will be quizzes most weeks. These quizzes will test understanding of assigned research papers. Expect 1-3 papers on most weeks. If you must miss a quiz, please let me know before hand and we will arrange appropriate accommodations, otherwise you receive a zero for that quiz.

Grading Quizzes and Assignments will be graded against a standardized rubric. The rubric contains several attributes. For each assignment and attribute a student can receive up to two points. At the end of the semester all points will be aggregated for each attribute. Given the set of attributes, I will project the scores to a single scalar per student. Letter grades will be drawn from this scalar space.

Attendance We will not take attendance, but there will be in-person quizzes every week.

Office hours We will arrive at an appropriate schedule during the first class. Expect 1 or 2 hours per week. Additional hours by appointment. Office hours will be conducted remotely on Microsoft Teams.

Boilerplate

Required links

- https://cooper.edu/sites/default/files/uploads/assets/site/files/2020/Cooper-Union-Policy-Upholding-Human-Rights-Title-IX-Protections.pdf
- https://cooper.edu/students/student-affairs/disability
- https://cooper.edu/students/student-affairs/health/counseling

Students Outcomes

- Ability to
  - discuss contemporary research in an intelligent way
  - recognize failings in a given experiment and synthesize follow-up experimentation
  - synthesize hypotheses on ablative and compositional experiments
  - argue in an evidence based way and make conclusions
  - communicate mathematical concepts in a narrative
  - identify situations in which deep learning may or may not be appropriate over other machine learning techniques

We will assess the aformentioned abilities through class discussions, quizzes, and assignment submissions.

Prerequisite Skills

- Knowledge of a programming language (Python preferred)
- Knowledge of differentiation in multivariate calculus
- Knowledge of basic linear algebra and probability (e.g., matrix multiplication, distributions)

Approximate list of topics

Introduction Linear regression. Regression with basis functions. Gradient descent. Automatic differention; reverse mode and forward mode. Affine projection. Multi-layer perceptrons. Activation functions. Cross validation. L1 and L2 regularization. Dropout, bath normalization, and friends. Logistic regression. Binary cross entropy, and other entropy based loss functions. Weight initialization.

Convolutions and friends Convolutional layers. Strided convolutions. Pooling. Residual connections. Transposed convolutions.

Transformers and friends Attention. Multi-head attention. Tokenization. CLIP. Generative-pretraining

Excotica Neural ODEs. Diffusion models. Mixture of experts. Large language models. Fine tuning. RHLF. RAG. Emergent abilities.

Applications and other techniques Autoencoders. Super-resolution. Image inpainting. Speech generation. Speech recognition. Music generation. Image generation. Recommender systems. Text classification. Natural language generation. Reinforcement learning. Style transfer. Content transfer.

Getting Started

Study the following code carefully:

`https://gist.github.com/ccurro/491d3e888e06f446ec1ede559adfd47d`

General Homework Requirements

1. Until I indicate otherwise the use of high-level APIs like `tf.keras` is forbidden.
2. Write tightly scoped classes/functions. When working in TensorFlow, inherit from `tf.Module` and test these modules with `pytest`
3. Demonstrate your knowledge of what mathematical properties each module should have by testing them.
4. Homework assignments will be submitted digitally and I will return scores with a handwritten rubric.
5. I may return general class-wide feedback on each assignment.
6. Each assignment should be reproducible. (i.e., running the code twice should return the exact same result back)
7. Submission of "notebooks" is forbidden.
8. The *only* framework references you will need for completing the assignments are the official TensorFlow docs: `https://www.tensorflow.org/guide/core` and `https://www.tensorflow.org/api/stable`. The documentation is high quality and up-to-date. Do not go searching for guides on YouTube, Medium, etc.
9. Use the Python docs liberally as well: `https://docs.python.org/3/`
10. Do not use AI products to write your homework assignments. We are studying how to *make* these products.
11. Practice using `flake8`, `isort`, and `black` to lint and standardize your Python code.

Assignment 1 — Due: Sept. 5 at 10 PM

tldr: Perform linear regression of a noisy sinewave using a set of gaussian basis functions with learned location and scale parameters. Model parameters are learned with stochastic gradient descent. Use of automatic differentiation is required. Hint: note your limits!

Problem Statement Consider a set of scalars $\{x_1, x_2, \ldots, x_N\}$ drawn from $\mathcal{U}(0, 1)$ and a corresponding set $\{y_1, y_2, \ldots, y_N\}$ where:

$$y_i = \sin(2\pi x_i) + \epsilon_i \tag{1}$$

and ϵ_i is drawn from $\mathcal{N}(0, \sigma_{\text{noise}})$. Given the following functional form:

$$\hat{y}_i = \sum_{j=1}^{M} w_j \phi_j \left(x_i \mid \mu_j, \sigma_j\right) + b \tag{2}$$

with:

$$\phi(x \mid \mu, \sigma) = \exp \frac{-(x-\mu)^2}{\sigma^2} \tag{3}$$

find estimates $\hat{b}$, $\{\hat{\mu}_j\}$, $\{\hat{\sigma}_j\}$, and $\{\hat{w}_j\}$ that minimize the loss function:

$$J(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2 \tag{4}$$

for all (x_i, y_i) pairs. Estimates for the parameters must be found using stochastic gradient descent. A framework that supports automatic differentiation must be used. Set $N = 50, \sigma_{\text{noise}} = 0.1$. Select M as appropriate. Produce two plots. First, show the data-points, a noiseless sinewave, and the manifold produced by the regression model. Second, show each of the M basis functions. Plots must be of suitable visual quality.

Requirements Create a `Linear` module. Create a `BasisExpansion` module. Write unit tests for each of them. Write an integration tests that combine the two together. Take inspiration from my example linear regression code.

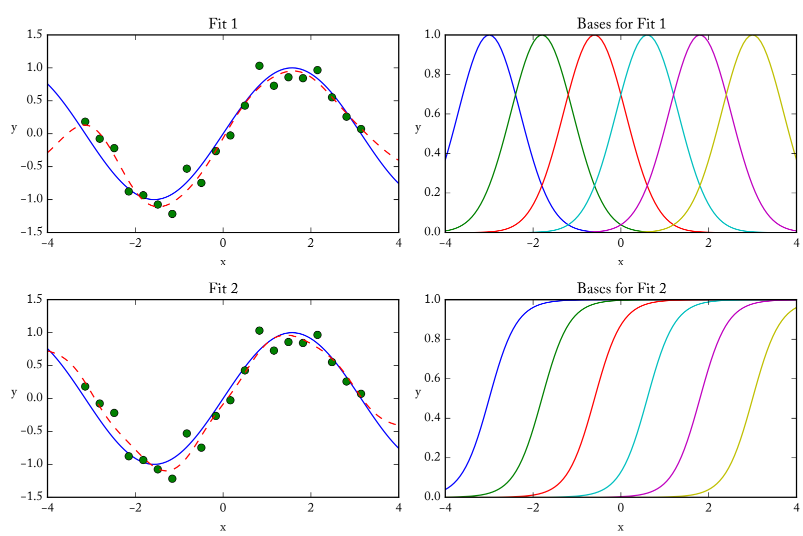

Figure 1: Example plots for models with equally spaced sigmoid and gaussian basis functions.

Assignment 2 — Due: Sept. 12 at 10 PM

tldr: Perform binary classification on the spirals dataset using a multi-layer perceptron. You must generate the data yourself.

Problem Statement Consider a set of examples with two classes and distributions as in Figure 2. Given the vector $x \in \mathbb{R}^2$ infer its target class $t \in \{0, 1\}$. As a model use a multi-layer perceptron f which returns an estimate for the conditional density $p(t = 1 \mid x)$:

$$f : \mathbb{R}^2 \to [0, 1] \tag{5}$$

parametrisized by some set of values θ. All of the examples in the training set should be classified correctedly (i.e. $p(t = 1 \mid x) > 0.5$ if and only if $t = 1$). Impose an L^2 penalty on the set of parameters. Produce one plot. Show the examples and the boundary corresponding to $p(t = 1 \mid x) = 0.5$. The plot must be of suitable visual quality. It may be difficult to to find an appropriate functional form for f, write a few sentences discussing your various attempts.

Requirements

1. Generate data using an instance of `numpy.random.Generator`
2. Create an `MLP` class. The `MLP` class should inherit from `tf.Module` and can use the `Linear` class from the previous assignment. It should have the following interface:

```
MLP(
  num_inputs,
  num_outputs,
  num_hidden_layers,
  hidden_layer_width,
  hidden_activation=tf.identity,
  output_activation=tf.identity,
)
```

3. Write unit tests for the `MLP` class.
4. Learn how to use `sklearn.inspection.DecisionBoundaryDisplay`
5. Your network must operate on Cartesian coordinates. Do not transform the coordinates to be polar.

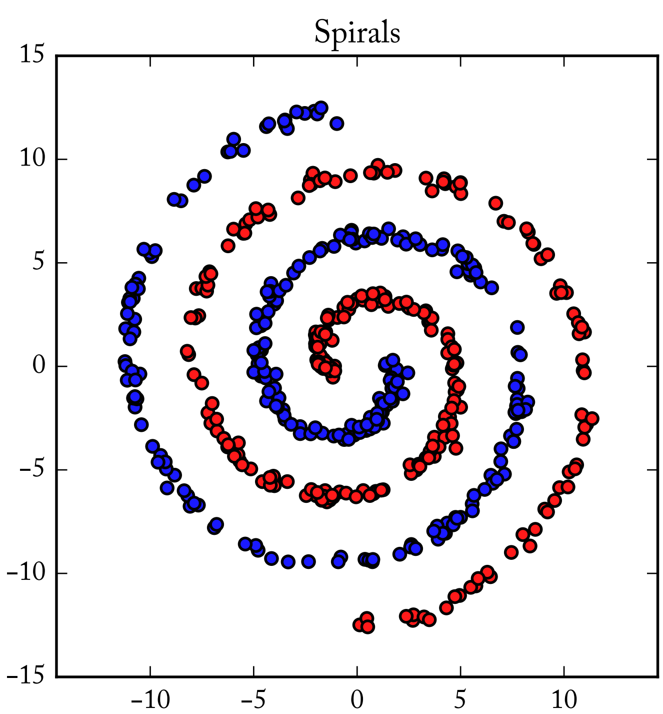

Figure 2: Sample spiral data.

Assignment 3 — Due: Sept. 19 at 10 PM

tldr: Classify MNIST digits with a convolutional neural network. Get at least 95.5% accuracy on the test test.

Problem Statement Consider the MNIST dataset consisting of 50,000 training images, and 10,000 test images. Each instance is a 28×28 pixel handwritten digit zero through nine. Train a (optionally convolutional) neural network for classification using the training set that achieves at least 95.5% accuracy on the test set. Do not explicitly tune hyperparameters based on the test set performance, use a validation set taken from the training set as discussed in class. Use dropout and an L^2 penalty for regularization. Note: if you write a sufficiently general program the next assignment will be very easy.

Do not use the built in MNIST data class from TensorFlow.

Requirements

1. Use `tf.nn.conv2d` for this assignment. *Do not try to write your convolution implementation.*
2. Create a `Conv2d` class that inherits from `tf.Module` and wraps `tf.nn.conv2d`
3. Create a `Classifier` class that inherits from `tf.Module`. The interface for `Classifier` should at a minimum be:

```
Classifier(
  input_depth: int,
  layer_depths: list[int],
  layer_kernel_sizes: list[tuple[int, int]],
  num_classes: int,
)
```

4. Write unit tests for all of your classes.

Extra challenge (optional) In addition to the above, the student with the fewest number of parameters for a network that gets at least 80% accuracy on the test set will receive a prize. There will be an extra prize if any one can achieve 80% on the test set with a single digit number of parameters. For this extra challenge you can make your network have any crazy kind of topology you'd like, it just needs to be optimized by a gradient based algorithm.

Assignment 4 — Due: Oct. 3 at 10 PM

tldr: Classify CIFAR10. Achieve performance similar to the state of the art. Classify CIFAR100. Achieve a top-5 accuracy of 90%.

Problem Statement Consider the CIFAR10 and CIFAR100 datasets which contain 32×32 pixel color images. Train a classifier for each of these with performance similar to the state of the art (for CIFAR10). It is your task to figure out what is state of the art. Feel free to adapt any techniques from papers you read. Write a paragraph or two summarizing your experiments. Hopefully you'll be able to resuse your MNIST program.

Requirements

1. Experiment with data augmentation.
2. Use your `Conv2d` class from the previous assignment
3. Create a `GroupNorm` class.
4. Create a `ResidualBlock` class around your `Conv2d` and `GroupNorm` classes.
5. Modify your `Classifier` class to use the new `ResidualBlock` class.
6. Write unit tests for all of your classes.

Assignment 5 — Due: Oct. 10 at 10 PM

tldr: Classify the AG News dataset.

Problem Statement Consider the AG News dataset at `https://huggingface.co/datasets/ag_news` which contains headlines and descriptions for a large set of news articles. Perform proper cross validation. You may use pretrained models; for example, `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

Assignment 6 — Due: Oct. 31 at 10 PM

Problem Statement Implement a `MultiHeadAttention` class and a `TransformerBlock` class. Assume 1-D case only. Provide a sufficient set of tests to prove that they work correctly.

Assignment 7 — Due: Nov. 28 at 10 PM

Problem Statement See SIREN at `https://www.vincentsitzmann.com/siren/`. Implement their method for "Image Fitting" on a single image: Test Card F. Demonstrate something "interesting" that you can do using the trained network. Take inspiration from the paper.

Papers

This paper list, for Fall 2023, is up to date as of November 9, 2023. Expect a few adjustments in the later weeks.

Week 1

1. Atilim Gunes Baydin, Barak A. Pearlmutter, and Alexey Andreyevich Radul. "Automatic differentiation in machine learning: a survey". In: *CoRR* abs/1502.05767 (2015). arXiv: `1502.05767`. URL: `http://arxiv.org/abs/1502.05767`
2. Leon Bottou. "Stochastic Gradient Descent Tricks". In: *Neural Networks, Tricks of the Trade, Reloaded*. Neural Networks, Tricks of the Trade, Reloaded. Vol. 7700. Lecture Notes in Computer Science (LNCS). Springer, Jan. 2012, pp. 430–445. URL: `https://www.microsoft.com/en-us/research/publication/stochastic-gradient-tricks/`

Week 2

3. Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: `1412.6980 [cs.LG]`
4. Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: `1711.05101 [cs.LG]`
5. Adam Pearce, Asma Ghandeharioun, and Nada Hussein. *Do machine learning models memorize or generalize?* URL: `https://pair.withgoogle.com/explorables/grokking/`

Week 3

6. Kaiming He et al. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. 2015. arXiv: `1502.01852 [cs.CV]`
7. Andre F. de Araújo, Wade Norris, and Jack Sim. "Computing Receptive Fields of Convolutional Neural Networks". In: *Distill* (2019). URL: `https://distill.pub/2019/computing-receptive-fields`
8. Kaiming He et al. *Identity Mappings in Deep Residual Networks*. 2016. arXiv: `1603.05027 [cs.CV]`

Week 4

9. Tomas Mikolov et al. "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems*. Ed. by C.J. Burges et al. Vol. 26. Curran Associates, Inc., 2013. URL: `https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf`
10. Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: `1706.03762`. URL: `http://arxiv.org/abs/1706.03762`
11. Taku Kudo and John Richardson. "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing". In: *CoRR* abs/1808.06226 (2018). arXiv: `1808.06226`. URL: `http://arxiv.org/abs/1808.06226`

Week 5

12. Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929. URL: https://arxiv.org/abs/2010.11929
13. Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: (2019)
14. Jordan Hoffmann et al. *Training Compute-Optimal Large Language Models*. 2022. DOI: 10.48550/ARXIV.2203.15556. URL: https://arxiv.org/abs/2203.15556
15. Aakanksha Chowdhery et al. *PaLM: Scaling Language Modeling with Pathways*. 2022. DOI: 10.48550/ARXIV.2204.02311. URL: https://arxiv.org/abs/2204.02311

Week 6

16. Michael Poli et al. *Hyena Hierarchy: Towards Larger Convolutional Language Models*. 2023. arXiv: 2302.10866 [cs.LG]
17. Manzil Zaheer et al. *Big Bird: Transformers for Longer Sequences*. 2021. arXiv: 2007.14062 [cs.LG]
18. Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903 [cs.CL]
19. Jason Wei et al. *Finetuned Language Models Are Zero-Shot Learners*. 2022. arXiv: 2109.01652 [cs.CL]
20. Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]

Week 7

21. Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. DOI: 10.48550/ARXIV.2103.00020. URL: https://arxiv.org/abs/2103.00020
22. Preetum Nakkiran et al. "Deep Double Descent: Where Bigger Models and More Data Hurt". In: *CoRR* abs/1912.02292 (2019). arXiv: 1912.02292. URL: http://arxiv.org/abs/1912.02292
23. Samuel L. Smith et al. *Don't Decay the Learning Rate, Increase the Batch Size*. 2018. arXiv: 1711.00489 [cs.LG]
24. Leslie N. Smith and Nicholay Topin. *Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates*. 2018. arXiv: 1708.07120 [cs.LG]

Week 8

25. Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. *A Neural Algorithm of Artistic Style*. 2015. arXiv: 1508.06576 [cs.CV]
26. Xun Huang and Serge Belongie. *Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization*. 2017. arXiv: 1703.06868 [cs.CV]
27. Tero Karras et al. *Analyzing and Improving the Image Quality of StyleGAN*. 2020. arXiv: 1912.04958 [cs.CV]
28. Tero Karras et al. "Alias-Free Generative Adversarial Networks". In: *CoRR* abs/2106.12423 (2021). arXiv: 2106.12423. URL: https://arxiv.org/abs/2106.12423

29. Patrick Esser, Robin Rombach, and Björn Ommer. "Taming Transformers for High-Resolution Image Synthesis". In: *CoRR* abs/2012.09841 (2020). arXiv: 2012.09841. URL: https://arxiv.org/abs/2012.09841

Week 8

30. Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. DOI: 10.48550/ARXIV.2204.06125. URL: https://arxiv.org/abs/2204.06125

31. Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2021. arXiv: 2112.10752 [cs.CV]. URL: https://arxiv.org/abs/2112.10752

32. Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. DOI: 10.48550/ARXIV.2205.11487. URL: https://arxiv.org/abs/2205.11487

33. Rinon Gal et al. *An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion*. 2022. DOI: 10.48550/ARXIV.2208.01618. URL: https://arxiv.org/abs/2208.01618

Week 10

34. Kaiming He et al. *Masked Autoencoders Are Scalable Vision Learners*. 2021. arXiv: 2111.06377 [cs.CV]

35. Aäron van den Oord et al. "WaveNet: A Generative Model for Raw Audio". In: *CoRR* abs/1609.03499 (2016). arXiv: 1609.03499. URL: http://arxiv.org/abs/1609.03499

36. Ron J. Weiss et al. "Wave-Tacotron: Spectrogram-free end-to-end text-to-speech synthesis". In: *CoRR* abs/2011.03568 (2020). arXiv: 2011.03568. URL: https://arxiv.org/abs/2011.03568

37. Aäron van den Oord et al. "Parallel WaveNet: Fast High-Fidelity Speech Synthesis". In: *CoRR* abs/1711.10433 (2017). arXiv: 1711.10433. URL: http://arxiv.org/abs/1711.10433

Week 11

38. Andrew Jaegle et al. *Perceiver: General Perception with Iterative Attention*. 2021. arXiv: 2103.03206 [cs.CV]

39. Andrew Jaegle et al. *Perceiver IO: A General Architecture for Structured Inputs and Outputs*. 2021. arXiv: 2107.14795 [cs.LG]

40. Muzammal Naseer et al. "Intriguing Properties of Vision Transformers". In: *CoRR* abs/2105.10497 (2021). arXiv: 2105.10497. URL: https://arxiv.org/abs/2105.10497

Week 12

41. Kai Arulkumaran et al. "A Brief Survey of Deep Reinforcement Learning". In: *CoRR* abs/1708.05866 (2017). arXiv: 1708.05866. URL: http://arxiv.org/abs/1708.05866

42. Julian Schrittwieser et al. "Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model". In: *CoRR* abs/1911.08265 (2019). arXiv: 1911.08265. URL: http://arxiv.org/abs/1911.08265

43. Lili Chen et al. “Decision Transformer: Reinforcement Learning via Sequence Modeling”. In: *CoRR* abs/2106.01345 (2021). arXiv: 2106.01345. URL: https://arxiv.org/abs/2106.01345

44. Danijar Hafner et al. “Mastering Atari with Discrete World Models”. In: *CoRR* abs/2010.02193 (2020). arXiv: 2010.02193. URL: https://arxiv.org/abs/2010.02193

Week 13

45. John Jumper et al. “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* 596.7873 (July 2021), pp. 583–589. DOI: 10.1038/s41586-021-03819-2. URL: https://doi.org/10.1038%2Fs41586-021-03819-2

46. Jonas Degrave et al. “Magnetic control of tokamak plasmas through deep reinforcement learning”. In: *Nature* 602.7897 (Feb. 2022), pp. 414–419. DOI: 10.1038/s41586-021-04301-9. URL: https://doi.org/10.1038%2Fs41586-021-04301-9

47. Julien Perolat et al. “Mastering the game of Stratego with model-free multiagent reinforcement learning”. In: *Science* 378.6623 (Dec. 2022), pp. 990–996. DOI: 10.1126/science.add4679. URL: https://arxiv.org/abs/2206.15378