CSC413/2516 Winter 2022 — Course Information
Neural Networks and Deep Learning

Course web site: http://uoft-csc413.github.io/2022

Overview

It is very hard to hand design programs to solve many real world problems, e.g. distinguishing images of cats v.s. dogs. Machine learning algorithms allow computers to learn from example data, and produce a program that does the job. Neural networks are a class of machine learning algorithm originally inspired by the brain, but which have recently have seen a lot of success at practical applications. They're at the heart of production systems at companies like Google and Facebook for image processing, speech-to-text, and language understanding. This course gives an overview of both the foundational ideas and the recent advances in neural net algorithms.

Roughly the first 2/3 of the course focuses on supervised learning — training the network to produce a specified behavior when one has lots of labeled examples of that behavior. The last 1/3 focuses on unsupervised learning and reinforcement learning.

Schedule

There is both an afternoon section and a night section for the course. Both will cover the same material, and will have the same assignments and final exam. Since both sections are at full enrollment, **please attend your assigned section**. See the course web page for detailed times.

Prerequisites

This is a second course in machine learning, so it has some substantial prerequisites. These prerequisites will be enforced, including for grad students.

- **Multivariable Calculus:**
  MAT235/MAT237/MAT257/MAT291/MAT294/AER210/MAT232/MAT233/MATB41
- **Linear Algebra:**
  MAT221H1/MAT223H1/MAT240H1/MAT185/MAT188/MAT223/MATA23
- **Machine Learning:**
  CSC311/CSC411/STA314/ECE421/ROB/313

Load

There are 24 hours of lectures and 11 hours of tutorials.

Readings

There is no required textbook for the class. A few small readings may be assigned if the need arises. These required readings will all be available on the web, for free.

There are also some relevant resources which are freely available online. We will try to provide links on a lecture-by-lecture basis.

- Video lectures for UofT Professor Geoffrey Hinton's Coursera course. Professor Hinton is one of the fathers of the field, so think of these as the Feynman Lectures of neural nets.
  https://www.youtube.com/playlist?list=PLoRl3Ht4JOcdU872GhiYWf6jwrk_SNhz9
- *Deep Learning*, a textbook by Yoshua Bengio, Ian Goodfellow, and Aaron Courville.
  http://www.deeplearningbook.org/
- Andrej Karpathy's lecture notes on convolutional networks. These are very readable and cover the material in roughly the first half of the course.
  http://cs231n.github.io/
- Richard Socher's lecture notes, focusing on RNNs.
  http://cs224d.stanford.edu/syllabus.html
- *Metacademy*, an online website (which one of the instructors is involved with) which helps you construct personalized learning plans and which has links to lots of resources relevant to particular concepts. We'll post links to relevant Metacademy concepts as the course progresses.
  http://www.metacademy.org
- Video lectures for Hugo Larochelle's neural networks course. These are similar to Professor Hinton's lectures but a bit more mathematical.
  http://info.usherbrooke.ca/hlarochelle/neural_networks/content.html
- David MacKay's excellent textbook, *Information Theory, Inference, and Learning Algorithms*. This isn't focused on neural nets per se, but it has some overlap with this course, especially the lectures on Bayesian models.
  http://www.inference.phy.cam.ac.uk/mackay/itila/
- *Neural Networks and Deep Learning*, a book by physicist Michael Nielsen which covers the basics of neural nets and backpropagation.
  http://neuralnetworksanddeeplearning.com/

Marking Scheme

The undergraduate marking scheme is as follows:

- Midterm test: 10%.
- Final project: 20%.
- Four programming assignments: 40%
  - Total of 4, weighted equally.
- Written homeworks: 30%.
  - Total of 4, your three highest marks will count for 10% each.
  - Lowest mark will be dropped.

The marking scheme for graduate students is identical to the marking scheme for undergraduates, except that the midterm test exam is weighted towards the final project. The requirements and marking scheme for the final project will be posted separately on the course web page.

Academic Integrity

By the time you get to an advanced course like csc413 you've heard this lots of times, so we'll keep it brief: avoid academic offenses (a.k.a. cheating). All graded work in this course is individual work.

Written Homeworks

In order to give you additional practice with the material, we assign written homeworks, which give you additional practice with the course content and encourage you to keep on top of the material. Roughly speaking, there will be one homework due each week that doesn't have another assignment or test. Each one consists of 2-3 conceptual questions and is meant to take a few hours.

Dates. Homeworks will typically be due at 11:59pm on Friday. See the course web page for particular deadlines. Each homework covers material up through the lecture one week prior to the deadline.

Format. Homeworks must be submitted in PDF format through MarkUs. We encourage typesetting using LaTeX, but scans of handwritten solutions are also acceptable.

Lateness. Homeworks will be accepted up to 3 days late, but 10% will be deducted for each day late, rounded up to the nearest day. Any exceptions require the absence declaration form on ACORN to request special consideration.

Weighting. In aggregate, the homeworks count for 30% of the total grade for the course, so individually they count for roughly 10% each. The lowest homework grade for the term will be dropped.

Collaboration policy. You are expected to work on the homeworks by yourself. You should not discuss them with anyone except the tutors or the instructor. The report you hand in should be entirely your own work and you may be asked to demonstrate how you got any results that you report.

Programming Assignments

A typical assignment will require you to write (or modify) and use some Python code that implements a simple version of a learning procedure that has recently been covered in the course. You will have to submit a brief report (roughly two pages plus figures) that describes the results you obtained.

Dates. Programming assignments will typically be due at 11:59pm on Friday. See the course web page for particular deadlines. Each assignment uses lecture material up through the lecture one week prior to the deadline.

Format. All programming assignment reports must be handed in as PDFs through MarkUs. They must be typed.

Lateness. Programming assignments will be accepted up to 3 days late, but 10% will be deducted for each day late, rounded up to the nearest day. Any exceptions require the absence declaration form on ACORN to request special consideration.

Weighting. The programming assignment marks will count for 10% each for a total of 40%.

Collaboration policy. You are expected to work on the assignments by yourself. You should not discuss them with anyone except the tutors or the instructor. The report you hand in should be entirely your own work and you may be asked to demonstrate how you got any results that you report.

Computation Resources

Many of the deep learning success stories in the recent years rely on the advances of modern GPU computing. The programming assignments here are lightweight comparing to the state of the art deep learning models in terms of their computation requirement. But we highly recommend you to debug your models and to complete the experiments on a modern GPU. Here are the list of free computation resources you have access to:

Colab **(Recommended)** Google Colab is a web-based iPython Notebook service that has access to a free Nvidia K80 GPU per Google account. Although it was initially developed for TensorFlow usage, Colab can easily be configured to run PyTorch, see tutorial here: https://medium.com/@chsafouane/getting-started-with-pytorch-on-google-colab-811c59a656b6

GCE **(Recommended)** Google Compute Engine delivers virtual machines running in Google's data center. You get $300 free credit when you sign up. They provide some of the latest GPUs on the market.

AWS-EC2 Amazone Elastic Compute Cloud (EC2) is a popular cloud platform. You may get free credit somewhere online.

CS Teaching Lab There are some very old GPUs in our CS Teaching Labs / CDF labs, see https://www.teach.cs.toronto.edu/faq.html#ABOUT5 for details.

Tests

Midterm. The midterm test (worth 10% of the course grade) will be held during usual lecture time. It covers material up through Lecture 6 (one week prior to the test).

Missed tests. Missed tests will get a score of 0 except in the case of submitting a university-wide absence declaration form on ACORN. In the event of illness, students should fill out the absence declaration form on ACORN to request special consideration before the test date and approved by the instructor.

submitted at least one week before the test date and approved by the instructor.

Online forum

We'll use Piazza for the course forum, http://piazza.com/utoronto.ca/winter2022/csc4132516.

Auditing

If you are not registered in the class, it is possible for you to audit it (sit in on the lectures). Here are the official university rules on auditors (taken from the Department of Computer Science instructor's advice page):

> To audit a course is to sit and listen to the lectures, and perhaps to the tutorials, without formally enrolling. Auditing is acceptable *if the auditor is a student at U of T, and no University resources are to be committed to the auditor.* The "must be a student" condition means that students of other universities, employees of outside organizations (or even of U of T itself!), or any other non-students, are not permitted to be auditors. (If we did not have this rule, the University would require us to collect auditing fees, and we are not willing to do that.)
>
> The "no resources used" condition means that auditors do not get computing accounts, cannot have term work marked, and cannot write exams. In other words, they cannot use instructors time, TA time, or administrative resources of any kind.
>
> An auditor may not attend class *unless there is an empty seat after the last regularly-enrolled student has sat down.* That sounds frivolous, but in fact it is an aspect of an important point: if enrollment in a course has been closed because the room size has been reached, then there may well be physical seats for auditors, because it is rare for every student to appear for a lecture, but auditors will not be allowed to enroll later on in the course, even if some students drop it. Neither instructors nor the department can waive this rule.
>
> Often these conditions are perfectly acceptable to auditors; we don't mean to ban the practice, but only to live within the University's rules.