COMP307/AIML420
INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Tutorial on neural networks

Neural Network Basics

- NN is a function; maps input to output: $\mathrm{y} = h(x)$
- It is an *adjustable* function, with parameters denoted θ or W
  - To make this explicit we could also write $y = h(\theta, x)$
- Learning adjusts the function by the adjusting parameters:
  - Based on a set of input-output examples (COMP307, AIML420)
  - (Based on structure of input only)
- To learn we define a loss/objective function $f(\theta)$
  - L2 example: $f(\theta) = \sum_{x \in A} \| h(\theta, x) - d \|^2$ with database A (and hence the x) fixed
  - In what respect we want the function to match the examples
    - For regression: usually squared error (L2) but also L1 is common
    - For classification: usually cross entropy
- We ensure $f(\theta)$ is differentiable with respect to θ
  - Ensure $h(\theta, x)$ is differentiable with respect to θ
  - We compute the gradient of $f(\theta)$ a the current θ and find a better θ by traveling downhill (gradient descent), reducing the loss function

What are NN good for

- Classification
  - images, documents, …
- Regression
  - medicine, agriculture, manufacturing, …
- Generation
  - chatGPT, image generation, superresolution, coding, …

NN Context: Structures/Components

- Fully connected neural networks (FCNN)
  - For more complex problems, do not work so well alone
- CNNs, convolutional neural networks
  - Each layer is a nonlinear *filter,*
    - Not everything is connected to everything reduces parameter no
  - Usually many filters in parallel: channels / feature maps
- Resnet:
  - FCNN or CNN layer(s) with bypass
  - Representations often changes slowly as a function of layer
  - Facilitates very deep networks
  - Can be interpreted as an approximation to a differential equation
- Unet
  - Decomposes then recomposes signal at various resolution levels
  - Commonly used in diffusion

NN Context: Structures/Components

- Recurrent neural networks (RNNs)
  - Compute output and state based on input and state
  - Remember in time
  - LSTM (long-term short-term memory) most common
  - Somewhat superseded by transformers

- Transformers
  - Fundamental component: *attention*
    - Self-attention and cross attention
    - Have a set of queries, keys, and values. Key and value come as a pair. The query asks the key how important the corresponding value is with as outcome a weight. Then sum the accordingly weighted values for each query (so get an "answer" for each query)
  - *Major component of LLMs*

Context: Current SOTA Systems

- Large language models (LLMs)
  - Text generation and more
  - Learn structure of language by predicting masked information
  - Predict the next word from the previous text
- Diffusion
  - Images and video generation
  - Idea:
    - Making an image into noise is easy
    - Making an image into noise with very small steps can be described as a (stochastic) differential equation, SDE
    - Invert the differential equation
      - See simulation above eqs. (8) and (10) in Song blog
- Methodology still changing rapidly
  - Knowledge of the basics allows you to keep up

JAX and Pytorch programs

- Colab link to JAX regression example
  - A program that does regression:
    - The output is the input multiplied by a specific matrix
    - The neural network has to learn to do multiply the input with that matrix
  - Which of the two tests at the end will give a better result?

- Colab link to PyTorch classification example
  - Simple example, but does use CNN
  - Unfortunately, this code no longer runs (should be easy to fix)
  - The original TensorFlow 2 version is here

- Colab link to JAX classification example
  - Uses perceptron