Chapter 1

Simon Ward-Jones
simonwardjones16@gmail.com

July 31, 2020

Link to chapter

1 Motivating example

- How to identify "Hey Siri" without machine learning?
- The (supervised) ML process is broadly as follows - Pick a model and initialise parameters then repeat:
  1. Measure how good the model is against data
  2. Change params (to ideally improve the error)
- Deep models are deep in precisely the sense that they learn many layers of computation

2 The Key Components: Data, Models, and Algorithms

- **Data** - what we can learn from
  - examples, targets and features
  - dimensionality - fixed length Vs variable-length
  - garbage in, garbage out
- **Model** - how to transform the data.
- **Loss function** - quantifies the badness of our model.
  - minimize the loss to give the best parameters
  - training and testing error and overfitting
  - $\mathbf{L}_1$, $\mathbf{L}_2$ and cross entropy
- **Algorithm** - how to adjust the model's parameters to minimize the loss (gradient descent)

3 Models

- **Supervised**
  - **regression** - real number target e.g. predicting house prices
  - **classification** - discrete number of targets e.g. predicting heart attacks $y \in \{0, 1\}$
  - find f_θ such that $f_\theta(\mathbf{x}_i) = \hat{y}_i \sim y_i$ given data $\{\mathbf{x}_i, y_i\}_{i=1}^n$
  - for classification we are (often) interested in $P(y|\mathbf{x})$
  - **tagging** - extending classification to multi-label classification (dog + cat, AWS, medical journal tagging)
  - **search and ranking** - ordered relevant subset retrieval
  - **recommender systems** - personalised recommendation subset retrieval
  - **sequence learning** - using sequential nature of input features
    - tagging and parsing - annotating a text sequence with attributes (e.g. named entities)
    - automatic speech recognition - from audio waves to text
    - text to speech - from text to audio waves
    - machine translation
- **Unsupervised**
  - **clustering** - grouping "similar" samples
  - **subspace estimation** - dimensionality reduction
  - **representation learning** - representing in $\mathbb{R}^n$
  - **probabilistic graphical models** - defining causality
  - **generative adversarial networks (GANs)** - synthesize data
- **Interacting with an Environment**
  - offline learning doesn't interact with environment
  - intelligent agents are trained to make actions
  - **Reinforcement learning** - e.g. self driving car. The agent interacts with the environment via actions and observations over a period of time steps. The agent receives a reward after each action (defining this reward is a problem itself). The policy (i.e. what action to take in a certain scenario) is learnt balancing exploration and exploitation.

4 Exercises

1. *Which parts of code that you are currently writing could be "learned", i.e., improved by learning and automatically determining design choices that are made in your code? Does your code include heuristic design choices?*

   The syntax and potentially styling could be learned. First you would need a large set of labelled data (e.g. samples with correct or incorrect syntax could be collected).

A classification algorithm could be used to predict whether there was a syntax error. The business logic of algorithms is much harder to capture as part of a model as it is more general and varies more between code samples. For this we would need GAI.

2. *Which problems that you encounter have many examples for how to solve them, yet no specific way to automate them? These may be prime candidates for using deep learning.*

   - Winning a sporting match
   - Responding to emails
   - Deleting blurry photos or shots with the lens cap on

3. *Viewing the development of artificial intelligence as a new industrial revolution, what is the relationship between algorithms and data? Is it similar to steam engines and coal (what is the fundamental difference)?*

   The steam engine was powered by coal much as modern ML algorithms are powered by data.

4. *Where else can you apply the end-to-end training approach? Physics? Engineering? Econometrics?*

   The benefit of end-to-end machine learning is that complex domain specific feature engineering can be captured instead by the deep layering effect of deep neural networks. Due to deep neural networks flexibility they have the ability to be applied in Physics, Engineering, Econometrics as well as many other fields. As a few examples: a CNN could be used to analyse faults in engineering parts using image recognition and an LSTM network could be used to identify certain patterns in sound recorded during physics re-search.