# Lecture 1: Introduction

CS 182 Spring 2021 – Taught by Sergey Levine

Notes by Michael Zhu

## Introduction

Deep learning is powerful because of its ability to automatically learn feature representations of complex inputs (i.e. images, text, etc).

**Example**: Instead of translating from English to French directly, we translate English into some representation of the "thought" which produced the English sentence.

This same "thought" can then be used to translate into many other languages without having to train specifically on the English → other language mapping.

**Note**: The above example is more of a *thought* exercise, in reality a deep learning model doesn't truly understand the semantic meaning of words but it produces a bunch of numbers to represent the words.

We don't know if the numbers actually represent the semantic meaning in the same way humans understand language, but the point is that it is a language-agnostic representation.

## What is machine learning?

**Traditional programming:** input → [computer program] → output

Humans know the mapping and can define the rules of the program by hand.

**Machine learning:** [input, output] → computer program

The rules are too complex to define by hand. We instead provide a bunch of examples and define a program that can figure out the input → output mapping by learning from the data.

The program learns **parameters** $\theta$ that are applied to the input $x$ to produce output $y$

$$f_\theta(x) = y$$

$f_\theta(x)$ can be any **expression** (aka parameterization) of $x$ and $\theta$.

$$\text{Example: } \vec{x}^T\vec{\theta} = y$$

## Shallow (Feature Based) learning

Use a fixed function for extracting features from $x$

Example: Histogram of oriented gradients → go over little patches of the image and find the edges in the image. This seems pretty good since most objects are defined by their shapes

Example: Haar features for face detection [1]

Compromise: don't hand program rules but hand program features

- Hard to come up with useful features

# Deep learning

Instead of hand picking features, the model learns the best features automatically.

There are many layers of features which can be learned, which is why this method is called *deep learning*.

Generally, earlier layers learn low level representations (i.e. lines, curves, basic shapes, etc) which then build up to high level representations (i.e. eyes, mouths, ears, etc) in later layers.

We want higher level features because they are comparatively more abstract and more invariant to nuisances (i.e. different image angles, poor lighting, etc), which make them more useful for predicting the final output.

## What do we need to make this work?

1. Models with high capacity (i.e. lot of layers and large layers)
2. Lots of data
3. Compute power to handle the above (i.e. GPU's, TPU's)

Con: Requires huge amounts of parameters, data, and compute

Pro: DL models scale well; more parameters, data, and compute $\rightarrow$ better performance

## Some Vocab

**Model Capacity**: how many different functions a model class can represent

**Inductive bias**: built-in knowledge/biases in a model

- all knowledge is "bias" in that it makes some solutions more likely than others
- balance of how much built-in knowledge vs how much learning
- Self-driving cars example: learn end-to-end vs human-designed system with learned intermediate steps

**Units (neural network "neurons")**:

- sums up input signals from upstream units $\rightarrow z = \sum_i a_i$
- uses an activation function to decide output signal $\rightarrow a = \sigma(z)$
- activation function choices include Sigmoid, TanH, ReLU, and more

## Brief History of Deep Learning

1950: Turing describes how learning could lead to machine intelligence [2]

1957: Rosenblatt's perceptron is proposed as a practical learning method [3]

1986: Backprop proposed as method for training deep neural nets [4]

1989: LeNet neural network for handwriting recognition [5]

2012: Krizhevsky's AlexNet wins ImageNet competition [6]

[1] https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf
[2] https://www.csee.umbc.edu/courses/471/papers/turing.pdf
[3] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4066017
[4] https://www.iro.umontreal.ca/~vincentp/ift3395/lectures/backprop_old.pdf
[5] http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf
[6] https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf