Deep Learning

Past, present and future

Prof. Gilles Louppe, Chaire NRB
g.louppe@uliege.be
Past
The Perceptron (Rosenblatt, 1957)

\[ f(x; w) = \begin{cases} 1 & \text{if } w_0 + w^T x > 0 \\ 0 & \text{otherwise} \end{cases} \]
$w = (-2, 3)$

$w = (1, 4)$

$w = (5, 4)$

$w = (0, 2)$

$w = (2, 2)$

$w = (5, 1)$

$w = (1, 0)$

$w = (3, 0)$

$w = (-2, -1)$

$w = (2, -2)$
The Multi-Layer Perceptron (Rumelhart et al, 1986)
Convolutional networks

Hubel and Wiesel, 1962
The Neocognitron (Fukushima, 1987)
Convolutional network (LeCun et al, 1989)

10 output units

**layer H3**
30 hidden units

**layer H2**
12 x 16 = 192 hidden units

**layer H1**
12 x 64 = 768 hidden units

256 input units

- Fully connected
  - ~ 300 links

- Fully connected
  - ~ 6000 links

- ~ 40,000 links
  - from 12 kernels
  - 5 x 5 x 8

- ~ 20,000 links
  - from 12 kernels
  - 5 x 5
LeCun et al, 1993
Learning

$$\theta_{t+1} = \theta_t - \gamma \nabla_{\theta} \mathcal{L}(\theta_t)$$
Present
1980s and 1990s

Accuracy

Scale (data size, model size)

- neural networks
- other approaches
What has changed?

**Algorithms**

\[ F(x) \]

- Weight layer
- ReLU
- Weight layer

\[ F(x) + x \]

- ReLU

**Data**

- Image of data

**Software**

- Caffe2
- CNTK
- TensorFlow
- Theano

**Compute engines**

- Image of compute engine
Deep networks

Szegedy et al, 2014
Applications

(Left) Image classification
(Right) Biomedical image segmentation (Cytomine, 2010-2018, ULiège)
Soccer games analysis (Cioppa et al, 2018, ULiège)
Analysis of scientific data at the LHC (Brehmer et al, 2018)
Autonomous Drone Navigation with Deep Learning. Flight over 250 meter F

Autonomous drones (Smolianskiy et al, 2017)
Learning to play video games (Mnih et al, 2013)
Learning to perform tasks (Levine et al, 2015)
Future
Automatisation : 4 belges sur 10 craignent de perdre leur job. À tort ?

L'automatisation galopante au sein des entreprises inquiète de nombreux travailleurs belges. Quatre Belges sur dix craignent ainsi de voir leur fonction disparaître au cours des dix prochaines années. À tort ?
Neural networks are not just another classifier, they represent the beginning of a fundamental shift in how we write software. They are Software 2.0.

Andrej Karpathy (Director of AI, Tesla, 2017)
Software 1.0

- Programs are written in languages such as Python, C or Java.
- They consist of *explicit instructions* to the computer *written by a programmer*.
- The programmer identifies a *specific point* in program space with some desirable behavior.

```java
string s input;
int length, IN;
double dblTemp;
bool again = true;

while (again) {
    IN = IN - 1;
    again = false;
    getline(s, input);
    system("cls");
    stringstream(input) >> dblTemp;
    length = input.length();
    if (length < 4) {
        again = true;
        continue;
    } else if (input[length - 3]) { 
        again = true;
        continue;
    }
    while (x < length) {
        if (isdigit(input[x])) {
            continue;
        } else if (length - x)) {
```
Software 2.0

- Programs are written in **neural network weights**
- **No human is involved** in writing those weights!
- Instead, specify constraints on the behavior of a desirable program (e.g., through data).
- **Search** the program space through optimization.

```
string sigma;
int length, N;
double delta;
bool again = true;
while (again) {
  if (< 3) {
    again = false;
    getline(cin, sigma);
    system("rm <temp> 2>");
    stringstream(sino) >> bitemp;
    length = sizeof(length);
    if (sznflg) {
      again = true;
      continue;
    } else {
      if (length == 3) { // ...
        if (again == true) {
          continue;
        } else {
          // ...
        }
      }
    }
  }
}
```
For many real-world problems, it is often significantly easier to collect the data than to explicitly write the program.

Therefore,

- programmers of tomorrow do not maintain complex software repositories, write intricate programs or analyze their running times.
- Instead, programmers become teachers. They collect, clean, manipulate, label, analyze and visualize the data that feeds neural nets.
Pixel data (e.g., visual recognition)

- airplane
- automobile
- bird
- cat
- deer
- dog

Audio data (e.g., speech recognition and synthesis)

- Google Assistant

Text data (e.g., machine translation)

- English: I love deep learning
- French: J'adore l'apprentissage en profondeur

System applications (e.g., databases)

- (a) Traditional Hash-Map
- (b) Learned Hash-Map
Growing Use of Deep Learning at Google

# of directories containing model description files

Across many products/areas:
- Android
- Apps
- drug discovery
- Gmail
- Image understanding
- Maps
- Natural language understanding
- Photos
- Robotics research
- Speech
- Translation
- YouTube
  ... many others ...

(Jeff Dean, Lead of Google.ai, 2017)
Benefits

- Computationally homogeneous
- Simple to bake in silicon
- Constant running time
- Constant memory use
- It is highly portable
- It is very agile
- It is better than you
Modules can meld into an optimal whole (Jeff Dean, Lead of Google.ai, 2017)
Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Software 2.0 needs Software 1.0 (Sculley et al, 2015)
People are now building a new kind of software by assembling networks of parameterized functional blocks and by training them from examples using some form of gradient-based optimization.

An increasingly large number of people are defining the networks procedurally in a data-dependent way (with loops and conditionals), allowing them to change dynamically as a function of the input data fed to them. It's really very much like a regular program, except it's parameterized.

Yann LeCun (Director of AI Research, Facebook, 2018)
Any Turing machine can be simulated by a recurrent neural network
(Siegelmann and Sontag, 1995)
Differentiable Neural Computer (Graves et al, 2016)
Summary

- Past: Deep Learning has a long history, fueled by contributions from neuroscience, control and computer science.
- Present: It is now mature enough to enable applications with super-human level performance, as already illustrated in many engineering disciplines.
- Future: Neural networks are not just another classifier. Sooner than later, they will take over increasingly large portions of what Software 1.0 is responsible for today.
The end.