Introduction to Deep-Learning and TDA

Outline

Some material

Book

- Deeplearning book, Goodfellow et al, 2016, MIT Press.
- Thousands of papers. Each years.

Alternative material

- Blog of C.Olah[a]
- Videos of 3Blue1Brown about DL on Youtube[b]

[a] Link for Colah's blog
[b] Click here to see the first video

What is a neural network?

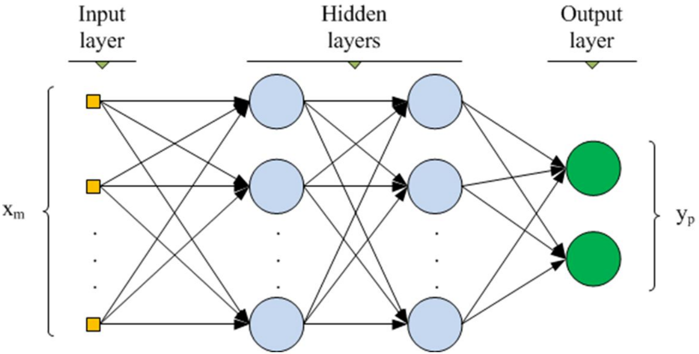
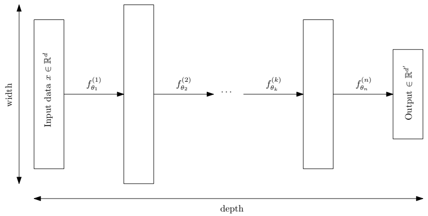


Figure: A multi-layer perceptron, the most standard neural network model.

What is a neural network?



Mathematical formulation

A multi-layer perceptron (MLP) is a class of functions $\mathcal{F}_\Theta = \{f_\theta = f^{(n)}_{\theta_n} \circ \ldots \circ f^{(1)}_{\theta_1}\}$ with:

- $\theta_k = (W_k, b_k)$ with $W_k \in \mathbb{R}^{d_{k+1} \times d_k}$, $b_k \in \mathbb{R}^{d_{k+1}}$
- $f^{(k)}_{\theta_k} : x \in \mathbb{R}^{d_k} \mapsto \sigma^{(k)}(W_k \cdot x + b_k) \in \mathbb{R}^{d_{k+1}}$:
- σ is an *activation* function, e.g. $\sigma(x) = \max(0, x)$ (ReLU) or $\sigma(x) = \frac{1}{1+e^{-x}}$ (sigmoid)

Deep-learning (supervised) problem

Framework

We have *labeled* data $(x_1, y_1)..(x_N, y_N) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_n}$, and we consider the optimization problem:

$$\text{minimize}\left\{\mathcal{L}\left((f_\theta(x_1)..f_\theta(x_N)), (y_1..y_n)\right) : \theta \in \Theta\right\} \tag{1}$$

where $\mathcal{L} : \mathbb{R}^{d_n} \times \mathbb{R}^{d_n} \to \mathbb{R}_+$ is a loss function.

Example: classification

We have K class, each label (y_i) has the form $(0..0, 1, 0..0) \in \mathbb{R}^K$, and we want to solve:

$$\text{minimize}\left\{\ell(\theta) := \sum_{i=1}^{N} ||f_\theta(x_i) - y_i||^2 : \theta \in \Theta\right\} \tag{2}$$

Why does deep-learning work?

Theoretically:

Very few results.

- *Universal approximation theorem*, states that with sufficiently high $(d_k)_k$, $\mathcal{F}_\Theta$ can approximate any continuous function.
- In some cases, it can be shown that there is no bad local minima, despite $\theta \mapsto \ell(\theta)$ not being convex. It legitimates gradient descent approach in optimization process (empirically verified).[a]

[a] See this paper, Kawaguchi, NIPS 2016, for example.

Why does deep-learning work?

Theoretically:

Very few results.

- *Universal approximation theorem*, states that with sufficiently high $(d_k)_k$, $\mathcal{F}_\Theta$ can approximate any continuous function.
- In some cases, it can be shown that there is no bad local minima, despite $\theta \mapsto \ell(\theta)$ not being convex. It legitimates gradient descent approach in optimization process (empirically verified).[a]

[a] See this paper, Kawaguchi, NIPS 2016, for example.

In practice:

- Involves only easy-to-compute functions, and **differentiable**, with easy-to-compute gradients.
- Structural form which allows to handle a **lot** of parameters. E.G. AlexNet (2012) has about 60M parameters.

Back-propagation

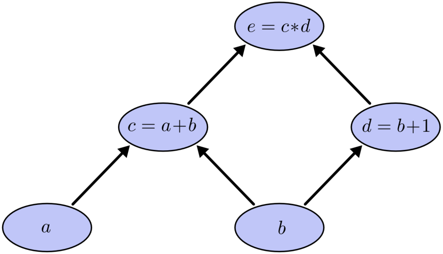


Figure: Backpropagation scheme on a computational graph, from Colah's blog

Back-propagation

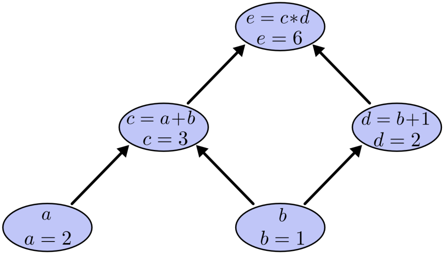


Figure: Backpropagation scheme on a computational graph, from Colah's blog

Back-propagation

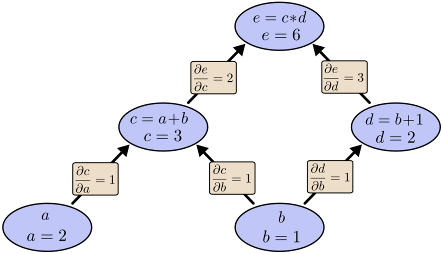


Figure: Backpropagation scheme on a computational graph, from Colah's blog

Back-propagation

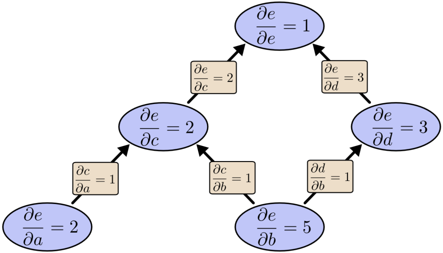


Figure: Backpropagation scheme on a computational graph, from Colah's blog

Back-propagation

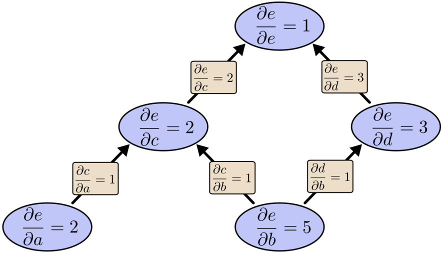


Figure: Backpropagation scheme on a computational graph, from Colah's blog

Take home message

Computing the gradient of $\theta \mapsto \ell(\theta) \in \mathbb{R}$ according to **all** parameters (variables) can be done with the same complexity as computing $\ell(\theta)$.

A word about convolution

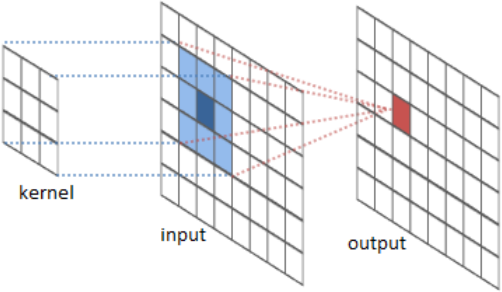


Figure: Classic way to depict convolution in NN. From Colah's blog

Why?

- Leverage intrinsic geometry in your data ("stationarity in the signal").
- Reduce the number of parameters (compared to a fully-connected one)

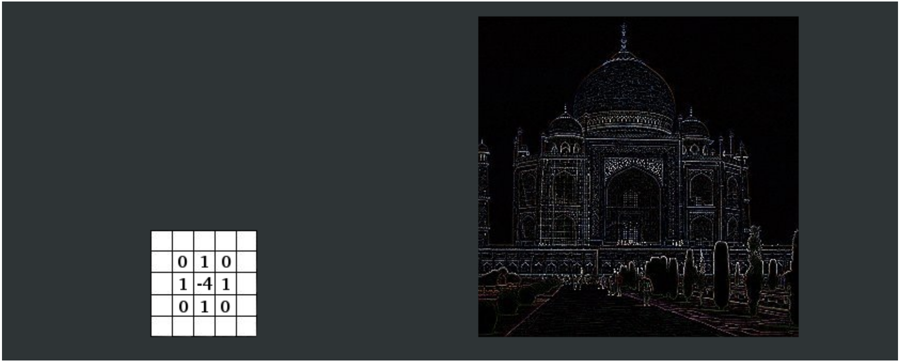


Figure: An illustration of 2D convolution, from Gimp documentation

Why?

- Leverage intrinsic geometry in your data ("stationarity in the signal").
- Reduce the number of parameters (compared to a fully-connected one)

Outline

1. Generalities about deep learning
2. Deep-learning and TDA

Limitations and motivations

Why is it hard to merge TDA and DL?

- Different mathematical approach: theoretical vs experimental.
- TDA objects (eg PDs) are not deep-learning-friendly:
  - Non-linear space, not in $\mathbb{R}^d$
  - Non-differentiable metrics

Why is it interesting?

- Use topological descriptors in deep-learning pipelines.
- Deep-learning could help TDA pipelines.
- TDA could help understanding deep-learning ; share some vocabulary.
- Deep-learning is everywhere, well-developed, huge community, etc.

Upcoming sessions

Some potentially interesting references

- Deep Learning with Topological Signatures, *Hofer et al, NIPS 2017.*
- Applying Topo. Pers. in CNN for Music Audio Signals, *Liu et al. Arxiv 2016.*
- Persistent homology of time-dependent functional networks constructed from coupled time series, *Stolz et al. AIP 2017*
- TDA in NLP (not exactly DL, but use w2v):
  - Does the geometry of Word embedding help document classification? *P.Michel et al. arxiv 2017*
  - Persistent homology, an introduction and a new text representation for NLP, *Zhu, Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*
  - A Topological collapse for Document Summarization, *Guan et al. IEEE 2016.*