

Statistical (Machine) Learning

Lecture 8 — Classifier Zoo

Pierpaolo Brutti

Many routes to one destination...

...get close to that Bayes strategy!

- Empirical Risk Minimization vs Plug-in methods vs Local Averaging/Smoothing
- Parametric vs Semi-parametric vs Non-parametric
- Linear vs Non-linear
- Discriminative vs Generative
- Unconstrained vs Constrained vs Penalized/Regularized
- etc.

Plug-in Methods

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Linear Regression

$$\text{Regression: } L(Y, f(\mathbf{X})) = \underset{\text{Square Loss}}{(Y - f(\mathbf{X}))^2} \quad R(f) = \underset{\text{Square Risk}}{\mathbb{E}(Y - f(\mathbf{X}))^2} \quad \widehat{R}(f) = \frac{1}{n} \sum_{i=1}^n (Y_i - f(\mathbf{X}_i))^2 \underset{\text{Empirical Risk (OLS)}}{}$$

$$\text{Linear: } \mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \left\{ f : \mathbb{R}^p \mapsto \mathbb{R} \mid f(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} = \langle \mathbf{x}, \boldsymbol{\beta} \rangle \right\} \quad (\text{Hypothesis Space})$$

$$R(\boldsymbol{\beta}) = \mathbb{E}(Y - \mathbf{X}^T \boldsymbol{\beta})^2 \rightsquigarrow \widehat{R}(\boldsymbol{\beta}) = \frac{1}{n} \sum_{i=1}^n (Y_i - \mathbf{X}_i^T \boldsymbol{\beta})^2 \propto \|\mathbf{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 \quad (\mathbb{X} \rightsquigarrow n \times p \text{ Design Matrix})$$

Algorithm

$$\boldsymbol{\beta}_{\text{emp}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \widehat{R}(\boldsymbol{\beta}) = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbf{Y} \quad \rightsquigarrow \quad f_{\text{emp}}(\mathbf{x}) = \underset{\boldsymbol{\beta} \in \mathcal{F}}{\operatorname{argmin}} \widehat{R}(f) = \mathbf{x}^T \boldsymbol{\beta}_{\text{emp}} \quad \rightsquigarrow \quad h_{\text{emp}}(\mathbf{x}) = \mathbb{I}(f_{\text{emp}}(\mathbf{x}) \geq \frac{1}{2})$$

Probabilistic Model / Likelihood

$$\{Y_i\}_{i=1}^n \stackrel{\text{ind.}}{\sim} \mathcal{N}_1(f_i, \sigma_\epsilon^2) \quad \text{with} \quad f_i = f_i(\boldsymbol{\beta}) = \mathbf{X}_i^T \boldsymbol{\beta} \quad \Leftrightarrow \quad \mathbf{Y} \sim \mathcal{N}_n(\mathbf{f}, \sigma_\epsilon^2 \mathbb{I}_n) \quad \text{with} \quad \mathbf{f} = \mathbb{X}\boldsymbol{\beta}$$

$$\underset{\text{maximize}}{\ell(\boldsymbol{\beta})} = \log \mathcal{L}(\boldsymbol{\beta}) \propto \sum_{i=1}^n \log e^{-\frac{1}{2\sigma_\epsilon^2} (Y_i - \mathbf{X}_i^T \boldsymbol{\beta})^2} = -\frac{1}{2\sigma_\epsilon^2} \|\mathbf{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 \propto -\widehat{R}(\boldsymbol{\beta})$$

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Linear Regression

$$\boldsymbol{\beta}_{\text{emp}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \widehat{R}(\boldsymbol{\beta}) = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbf{Y} \quad \rightsquigarrow \quad f_{\text{emp}}(\mathbf{x}) = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \widehat{R}(f) = \mathbf{x}^T \boldsymbol{\beta}_{\text{emp}} \quad \rightsquigarrow \quad h_{\text{emp}}(\mathbf{x}) = \mathbb{I}(f_{\text{emp}}(\mathbf{x}) \geq \frac{1}{2})$$

Numerical Solutions

- **Problem:** Calculating $\boldsymbol{\beta}_{\text{emp}}$ naively can be prohibitive when p is large
 - Matrix inversion takes $O(p^3)$ by **Gauss-Jordan method**
 - Very *unstable* if $(\mathbb{X}^T \mathbb{X})$ is badly conditioned \rightsquigarrow **multicollinearity** (i.e., correlated features)
- **Solution A:** QR -factorization (improve numerical stability)
 - **Remember:** $\boldsymbol{\beta}_{\text{emp}}$ solves the equation: $(\mathbb{X}^T \mathbb{X}) \boldsymbol{\beta} = \mathbb{X}^T \mathbf{Y}$
 - **Write:** $\mathbb{X} = QR$ with $Q \in \mathbb{R}^{n \times p}$ *orthogonal* matrix ($Q^T Q = \mathbb{I}_p$) and $R \in \mathbb{R}^{p \times p}$ upper triangular.
 - **Substitute:** $R^T (Q^T Q) R \boldsymbol{\beta} = R^T Q^T \mathbf{Y} \Leftrightarrow R^T R \boldsymbol{\beta} = R^T Q^T \mathbf{Y} \Leftrightarrow R \boldsymbol{\beta} = Q^T \mathbf{Y}$
 - **Moral:** just solve a *linear* system with a *triangular* matrix \rightsquigarrow easy \rightsquigarrow **R lm()**

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Linear Regression

$$\boldsymbol{\beta}_{\text{emp}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \widehat{R}(\boldsymbol{\beta}) = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbf{Y} \quad \rightsquigarrow \quad f_{\text{emp}}(\mathbf{x}) = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \widehat{R}(f) = \mathbf{x}^T \boldsymbol{\beta}_{\text{emp}} \quad \rightsquigarrow \quad h_{\text{emp}}(\mathbf{x}) = \mathbb{I}(f_{\text{emp}}(\mathbf{x}) \geq \frac{1}{2})$$

Numerical Solutions

- **Problem:** Calculating $\boldsymbol{\beta}_{\text{emp}}$ naively can be prohibitive when p is large
 - Matrix inversion takes $O(p^3)$ by **Gauss-Jordan method**
 - Very *unstable* if $(\mathbb{X}^T \mathbb{X})$ is badly conditioned \rightsquigarrow **multicollinearity** (i.e., correlated features)
- **Solution B:** Gradient Descent (improve speed, reduce memory footprint \rightsquigarrow big data)
 - Completely bypass the need of matrix inversion or factorization using gradient descent.
 - **Remember:** $\widehat{R}(\boldsymbol{\beta}) = \frac{1}{n} \|\mathbf{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 = \frac{1}{n} (\mathbf{Y} - \mathbb{X}\boldsymbol{\beta})^T (\mathbf{Y} - \mathbb{X}\boldsymbol{\beta}) = \frac{1}{n} (\mathbf{Y}^T \mathbf{Y} - 2 \boldsymbol{\beta}^T \mathbb{X}^T \mathbf{Y} + \boldsymbol{\beta}^T (\mathbb{X}^T \mathbb{X}) \boldsymbol{\beta})$
- $\nabla \widehat{R}(\boldsymbol{\beta}) = \frac{2}{n} ((\mathbb{X}^T \mathbb{X}) \boldsymbol{\beta} - \mathbb{X}^T \mathbf{Y}) \quad \rightsquigarrow \quad \boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \alpha \nabla \widehat{R}(\boldsymbol{\beta}^{(t)})$
- If the data set is much too big \rightsquigarrow go *Stochastic!*

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Logistic Regression

Cross-Entropy: $L(Y, f(\mathbf{X})) \stackrel{\star}{=} -[Y \log f(\mathbf{X}) + (1-Y) \log(1-f(\mathbf{X}))] \stackrel{\heartsuit}{=} -[\log(1-f(\mathbf{X})) + Y \log \frac{f(\mathbf{X})}{1-f(\mathbf{X})}] \quad R(f) = \mathbb{E} L(Y, f(\mathbf{X}))$

$\widehat{R}(f) \stackrel{\star}{=} -\frac{1}{n} \sum_{i=1}^n Y_i \log f(\mathbf{X}_i) + (1-Y_i) \log(1-f(\mathbf{X}_i)) \stackrel{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^n \log(1-f(\mathbf{X}_i)) + Y_i \log \frac{f(\mathbf{X}_i)}{1-f(\mathbf{X}_i)} \quad (\text{Empirical Risk})$

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Logistic Regression

Cross-Entropy: $L(Y, f(\mathbf{X})) \stackrel{\star}{=} -[Y \log f(\mathbf{X}) + (1-Y) \log(1-f(\mathbf{X}))] \stackrel{\heartsuit}{=} -[\log(1-f(\mathbf{X})) + Y \log \frac{f(\mathbf{X})}{1-f(\mathbf{X})}] \quad R(f) = \mathbb{E} L(Y, f(\mathbf{X}))$

$$\widehat{R}(f) \stackrel{\star}{=} -\frac{1}{n} \sum_{i=1}^n Y_i \log f(\mathbf{X}_i) + (1-Y_i) \log(1-f(\mathbf{X}_i)) \stackrel{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^n \log(1-f(\mathbf{X}_i)) + Y_i \log \frac{f(\mathbf{X}_i)}{1-f(\mathbf{X}_i)} \quad (\text{Empirical Risk})$$

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \left\{ f : \mathbb{R}^p \mapsto [0, 1] \mid f(\mathbf{x}) = (\text{act} \circ g)(\mathbf{x}) \text{ with } \text{act}(u) = \frac{1}{1+e^{-u}}, g(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} \right\} \quad (\text{Hypothesis Space})$

$$L(Y, g(\mathbf{X})) \stackrel{\star}{=} Y \log(1+e^{-g(\mathbf{X})}) + (1-Y) \log(1+e^{+g(\mathbf{X})}) \rightsquigarrow \widehat{R}(\boldsymbol{\beta}) \stackrel{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^n \log(1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}) - Y_i \mathbf{X}_i^T \boldsymbol{\beta}$$

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Logistic Regression

Cross-Entropy: $L(Y, f(\mathbf{X})) \stackrel{\star}{=} -[Y \log f(\mathbf{X}) + (1-Y) \log(1-f(\mathbf{X}))] \stackrel{\heartsuit}{=} -[\log(1-f(\mathbf{X})) + Y \log \frac{f(\mathbf{X})}{1-f(\mathbf{X})}] \quad R(f) = \mathbb{E} L(Y, f(\mathbf{X}))$

$$\widehat{R}(f) \stackrel{\star}{=} -\frac{1}{n} \sum_{i=1}^n Y_i \log f(\mathbf{X}_i) + (1-Y_i) \log(1-f(\mathbf{X}_i)) \stackrel{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^n \log(1-f(\mathbf{X}_i)) + Y_i \log \frac{f(\mathbf{X}_i)}{1-f(\mathbf{X}_i)} \quad (\text{Empirical Risk})$$

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \left\{ f : \mathbb{R}^p \mapsto [0, 1] \mid f(\mathbf{x}) = (\text{act} \circ g)(\mathbf{x}) \text{ with } \text{act}(u) = \frac{1}{1+e^{-u}}, g(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} \right\} \quad (\text{Hypothesis Space})$

$$L(Y, g(\mathbf{X})) \stackrel{\star}{=} Y \log(1+e^{-g(\mathbf{X})}) + (1-Y) \log(1+e^{+g(\mathbf{X})}) \rightsquigarrow \widehat{R}(\boldsymbol{\beta}) \stackrel{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^n \log(1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}) - Y_i \mathbf{X}_i^T \boldsymbol{\beta}$$

Algorithm

$$\boldsymbol{\beta}_{\text{emp}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \widehat{R}(\boldsymbol{\beta}) = \{\text{no closed form}\} \rightsquigarrow f_{\text{emp}}(\mathbf{x}) = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \widehat{R}(f) = \text{act}(\mathbf{x}^T \boldsymbol{\beta}_{\text{emp}}) \in [0, 1] \rightsquigarrow h_{\text{emp}}(\mathbf{x}) = \mathbb{I}(f_{\text{emp}}(\mathbf{x}) \geq \frac{1}{2})$$

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Logistic Regression

Cross-Entropy: $L(Y, f(\mathbf{X})) \stackrel{\star}{=} -[Y \log f(\mathbf{X}) + (1-Y) \log(1-f(\mathbf{X}))] \stackrel{\heartsuit}{=} -[\log(1-f(\mathbf{X})) + Y \log \frac{f(\mathbf{X})}{1-f(\mathbf{X})}] \quad R(f) = \mathbb{E} L(Y, f(\mathbf{X}))$

$$\widehat{R}(f) \stackrel{\star}{=} -\frac{1}{n} \sum_{i=1}^n Y_i \log f(\mathbf{X}_i) + (1-Y_i) \log(1-f(\mathbf{X}_i)) \stackrel{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^n \log(1-f(\mathbf{X}_i)) + Y_i \log \frac{f(\mathbf{X}_i)}{1-f(\mathbf{X}_i)} \quad (\text{Empirical Risk})$$

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \left\{ f : \mathbb{R}^p \mapsto [0, 1] \mid f(\mathbf{x}) = (\text{act} \circ g)(\mathbf{x}) \text{ with } \text{act}(u) = \frac{1}{1+e^{-u}}, g(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} \right\} \quad (\text{Hypothesis Space})$

$$L(Y, g(\mathbf{X})) \stackrel{\star}{=} Y \log(1+e^{-g(\mathbf{X})}) + (1-Y) \log(1+e^{+g(\mathbf{X})}) \rightsquigarrow \widehat{R}(\boldsymbol{\beta}) \stackrel{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^n \log(1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}) - Y_i \mathbf{X}_i^T \boldsymbol{\beta}$$

Algorithm

$$\boldsymbol{\beta}_{\text{emp}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \widehat{R}(\boldsymbol{\beta}) = \{\text{no closed form}\} \rightsquigarrow f_{\text{emp}}(\mathbf{x}) = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \widehat{R}(f) = \text{act}(\mathbf{x}^T \boldsymbol{\beta}_{\text{emp}}) \in [0, 1] \rightsquigarrow h_{\text{emp}}(\mathbf{x}) = \mathbb{I}(f_{\text{emp}}(\mathbf{x}) \geq \frac{1}{2})$$

Probabilistic Model / Likelihood

$$\{Y_i\}_{i=1}^n \stackrel{\text{ind}}{\sim} \text{Ber}(f_i) \text{ with } f_i = \frac{1}{1+e^{-\mathbf{X}_i^T \boldsymbol{\beta}}} \rightsquigarrow \underset{\text{maximize}}{\ell(\boldsymbol{\beta})} = \log \mathcal{L}(\boldsymbol{\beta}) \propto \sum_{i=1}^n Y_i \log(1-f_i) + (1-Y_i) \log(1-f_i) \propto -\widehat{R}(\boldsymbol{\beta})$$

Discriminative Methods

$$p(y, \mathbf{x}) = p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\mathbf{x}) = \mathbb{I}(f_{\text{opt}}(\mathbf{x}) \geq \frac{1}{2}) \quad \text{with} \quad f_{\text{opt}}(\mathbf{x}) = \mathbb{E}(Y \mid \mathbf{X} = \mathbf{x}) \stackrel{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

Logistic Regression

Cross-Entropy: $L(Y, f(\mathbf{X})) \stackrel{\star}{=} -[Y \log f(\mathbf{X}) + (1-Y) \log(1-f(\mathbf{X}))] \stackrel{\heartsuit}{=} -[\log(1-f(\mathbf{X})) + Y \log \frac{f(\mathbf{X})}{1-f(\mathbf{X})}] \quad R(f) = \mathbb{E} L(Y, f(\mathbf{X}))$

$$\widehat{R}(f) \stackrel{\star}{=} -\frac{1}{n} \sum_{i=1}^n Y_i \log f(\mathbf{X}_i) + (1-Y_i) \log(1-f(\mathbf{X}_i)) \stackrel{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^n \log(1-f(\mathbf{X}_i)) + Y_i \log \frac{f(\mathbf{X}_i)}{1-f(\mathbf{X}_i)} \quad (\text{Empirical Risk})$$

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \left\{ f : \mathbb{R}^p \mapsto [0, 1] \mid f(\mathbf{x}) = (\text{act} \circ g)(\mathbf{x}) \text{ with } \text{act}(u) = \frac{1}{1+e^{-u}}, g(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} \right\} \quad (\text{Hypothesis Space})$

$$L(Y, g(\mathbf{X})) \stackrel{\star}{=} Y \log(1+e^{-g(\mathbf{X})}) + (1-Y) \log(1+e^{+g(\mathbf{X})}) \rightsquigarrow \widehat{R}(\boldsymbol{\beta}) \stackrel{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^n \log(1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}) - Y_i \mathbf{X}_i^T \boldsymbol{\beta}$$

Numerical Solutions

- Newton's method/Fisher's scoring \rightsquigarrow Iteratively Reweighted Least Square (IRLS) $\stackrel{\text{R}}{\rightsquigarrow}$ `glm()`
- Gradient descent:** apply *chain-rule* in reverse, from output/predictions to input/features (*backprop!*)

$$\frac{\partial \widehat{R}}{\partial \beta_j} \stackrel{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^n \frac{1}{1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}} \frac{\partial}{\partial \beta_j} (1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}) - Y_i X_{i,j} = \frac{1}{n} \sum_{i=1}^n \frac{e^{\mathbf{X}_i^T \boldsymbol{\beta}}}{1+e^{\mathbf{X}_i^T \boldsymbol{\beta}}} \frac{\partial}{\partial \beta_j} (\mathbf{X}_i^T \boldsymbol{\beta}) - Y_i X_{i,j} = \frac{1}{n} \sum_{i=1}^n \text{act}(\mathbf{X}_i^T \boldsymbol{\beta}) X_{i,j} - Y_i X_{i,j}$$

$$\nabla \widehat{R}(\boldsymbol{\beta}) = \frac{1}{n} \mathbb{X}^T (\mathbf{f} - \mathbf{Y}) \quad \text{with} \quad f_i = \text{act}(\mathbf{X}_i^T \boldsymbol{\beta}) \quad (\text{as in OLS, at min, residuals are orthogonal to col-space of } \mathbb{X})$$

Discriminative Methods

Neural Network / Multilayer Perceptron

- The **backwards propagation** of errors, or just *backpropagation*, is the standard algorithm for computing gradients in a neural network.
- It is conceptually based on applying the **chain rule** to each layer of the network in **reverse order**.
- The conceptual idea behind backpropagation can be applied to any hierarchical model described by a directed acyclic graph (DAG) \rightsquigarrow here focus on dense neural networks.

Discriminative Methods

Neural Network / Multilayer Perceptron

- The **backwards propagation** of errors, or just *backpropagation*, is the standard algorithm for computing gradients in a neural network.
- It is conceptually based on applying the **chain rule** to each layer of the network in **reverse order**.
- The conceptual idea behind backpropagation can be applied to any hierarchical model described by a directed acyclic graph (DAG) \rightsquigarrow here focus on dense neural networks.
- **Step 1**
 - Insert an input x into the first layer and then propagating the outputs of each hidden layer through to the final output.
 - All of the intermediate outputs are stored.
 - Derivatives with respect to parameters in the **last** layer are calculated **directly**.

Discriminative Methods

Neural Network / Multilayer Perceptron

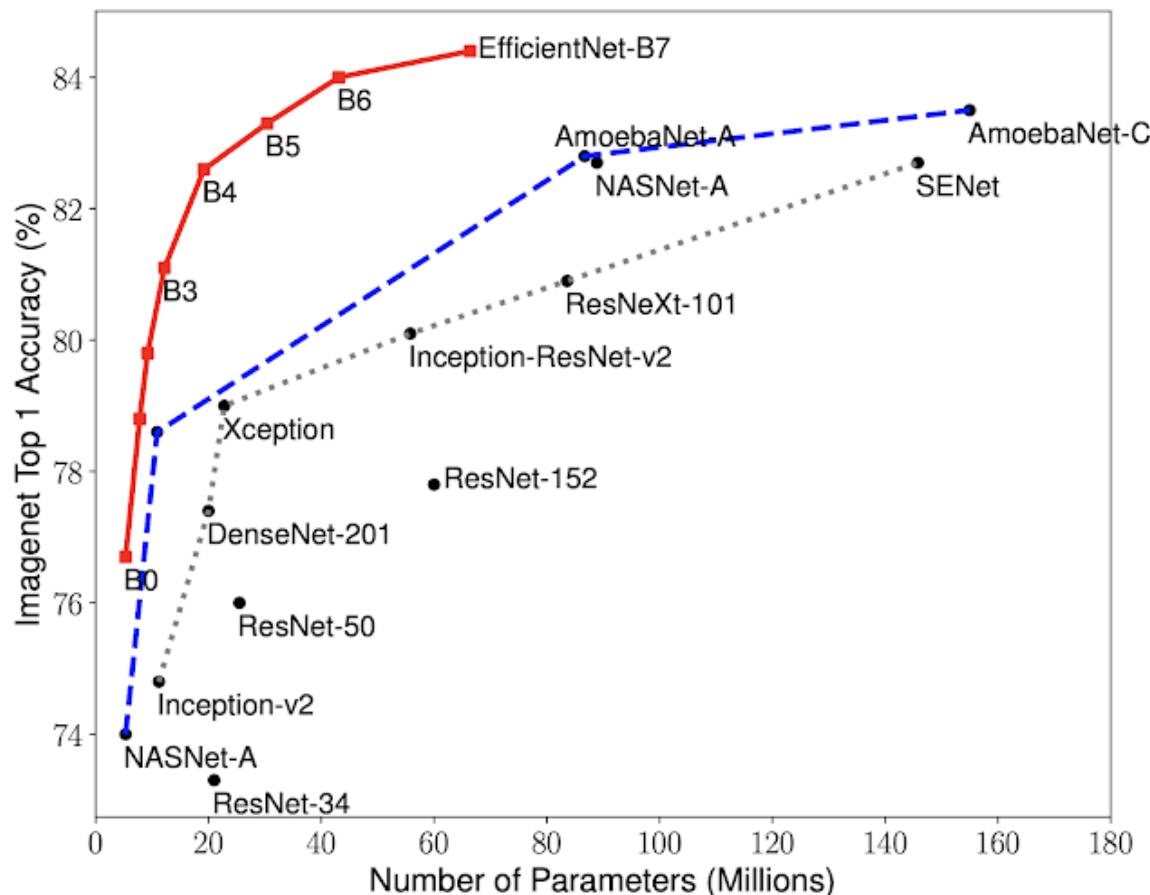
- The **backwards propagation** of errors, or just *backpropagation*, is the standard algorithm for computing gradients in a neural network.
- It is conceptually based on applying the **chain rule** to each layer of the network in **reverse order**.
- The conceptual idea behind backpropagation can be applied to any hierarchical model described by a directed acyclic graph (DAG) \rightsquigarrow here focus on dense neural networks.
- **Step 1**
 - Insert an input x into the first layer and then propagating the outputs of each hidden layer through to the final output.
 - All of the intermediate outputs are stored.
 - Derivatives with respect to parameters in the **last** layer are calculated **directly**.
- **Step 2**
 - Derivatives are now calculated to quantify how changing the parameters in any internal layer affect the output of just that layer.
 - The **chain rule** is then used to compute the full gradient in terms of these intermediate quantities with one pass backwards through the network.

Quick & Shallow

(...a first tiny step into the Deep...)

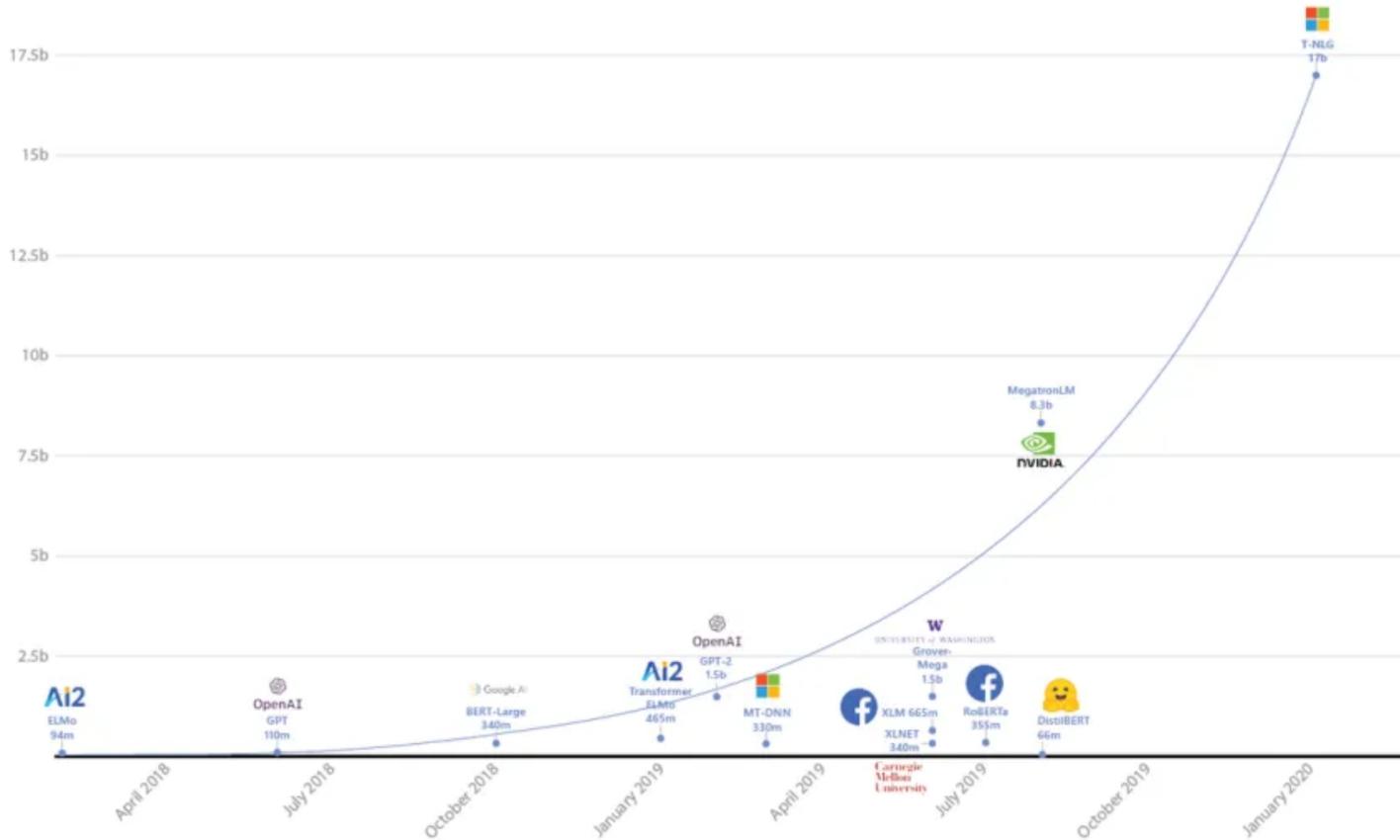
How Deep?

"Million" Deep



How Deep?

"Billion" or even "Trillion" Deep



Deep Networks: Revolution

- Neural networks are and always will be a cause of "friction" between scientific communities
- Neural networks are and always will be highly *non-linear* models in the parameters whose most "mythical" progenitor is...*logistic regression!* (^_^')
- Neural networks are and always will be models that essentially **claim** to dispense with specific *domain knowledge* to address a learning problem.

Deep Networks: Revolution

- Neural networks are and always will be a cause of "friction" between scientific communities
- Neural networks are and always will be highly *non-linear* models in the parameters whose most "mythical" progenitor is...*logistic regression!* (^_^')
- Neural networks are and always will be models that essentially **claim** to dispense with specific *domain knowledge* to address a learning problem.
- Neural networks are and always will be *over-parameterized* models that need:
 1. huge amounts of data to be decently estimated
 2. implicit or explicit regularization schemes
 3. highly efficient and specialized (optimization) algorithms and heuristics

Deep Networks: Revolution

- Neural networks are and always will be a cause of "friction" between scientific communities
- Neural networks are and always will be highly *non-linear* models in the parameters whose most "mythical" progenitor is...*logistic regression!* (^_^')
- Neural networks are and always will be models that essentially **claim** to dispense with specific *domain knowledge* to address a learning problem.
- Neural networks are and always will be *over-parameterized* models that need:
 1. huge amounts of data to be decently estimated
 2. implicit or explicit regularization schemes
 3. highly efficient and specialized (optimization) algorithms and heuristics
- In the 1990s there was essentially a lack of infrastructure to run these models at full speed, today...well...it's (somewhat) different.
- Given the appalling size of current models, the use of **pre-trained** nets simply **fine-tuned** on the specific task at hand is the (industrial) standard.
- A complete, satisfactory theoretical treatment is still missing...our men are working on it!

Deep Networks: Revolution

[Home](#) > [Latest Awards News](#) > [2018 Turing Award](#)

Fathers of the Deep Learning Revolution Receive ACM A.M. Turing Award

Bengio, Hinton and LeCun Ushered in Major Breakthroughs in Artificial Intelligence

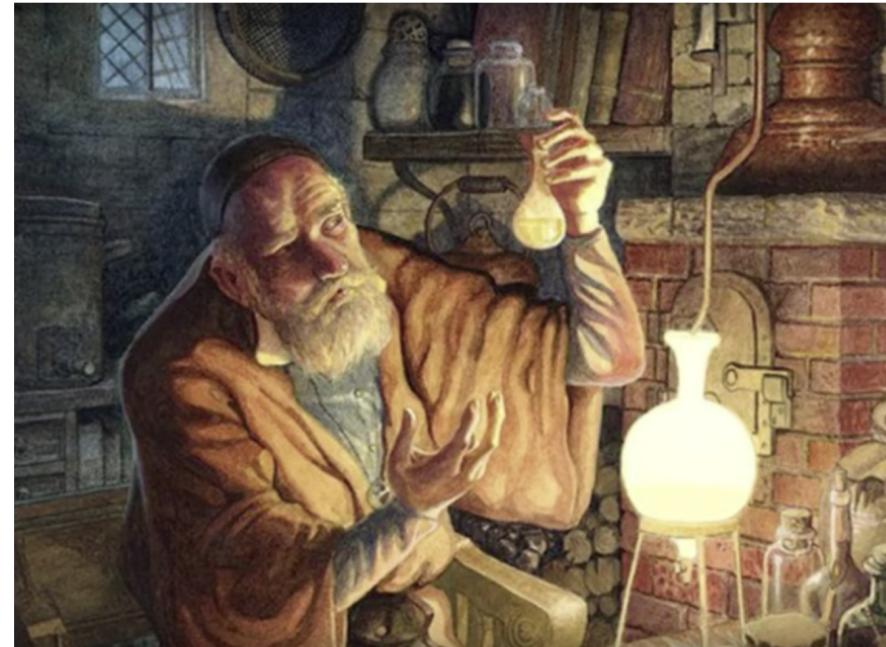


Deep Networks: Alchemy

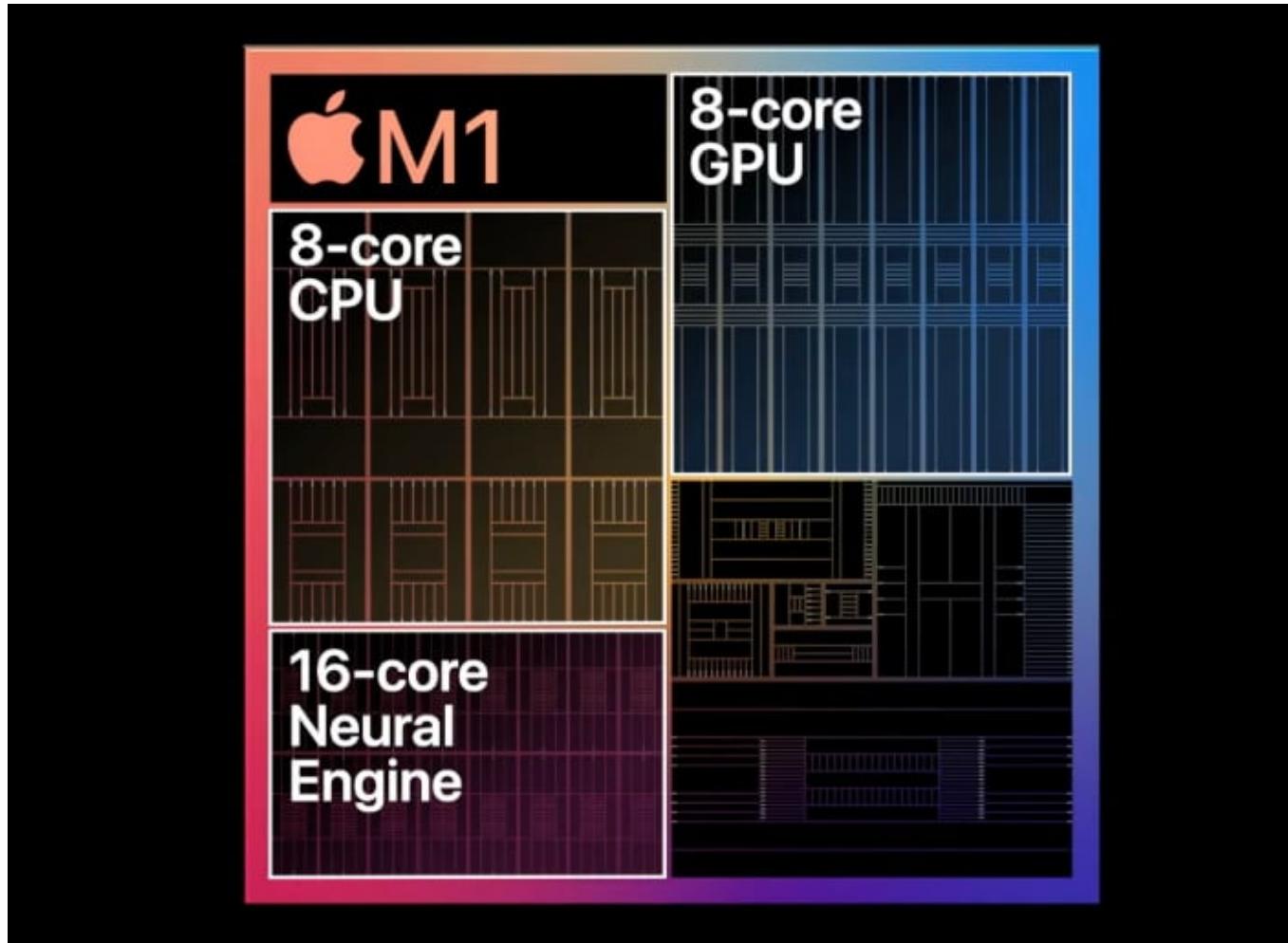
LeCun vs Rahimi: Has Machine Learning Become Alchemy?



Synced Dec 13, 2017 · 4 min read ★



Deep Networks: Revolution



Deep Networks: Revolution (?)

[] DeepAI

RESEARCHERS DEVELOPERS PRODUCTS ...

LOGIN SIGNUP

DeepAI: The front page of A.I.

The most popular research, guides, news and more in artificial intelligence

⌚ Trending ⚖ Research 📰 News 📜 Definitions 📈 Jobs ...

popular | newest

Automatic deep-learning AI tool measures volume of cerebral ventricles on MRIs in children

news

26 hours ago · Sadiq Zayn Almasi · ❤️ 14 · ⚡ share

read it

KD-Lib: A PyTorch library for Knowledge Distillation, Pruning and Quantization

research

44 hours ago · by Het Shah, et al. · ❤️ 35 · ⚡ share

read it

Deep Networks: Revolution (?)

TECH \ ARTIFICIAL INTELLIGENCE

TL;DR

Use this cutting-edge AI text generator to write stories, poems, news articles, and more

2

TalkToTransformer.com offers an accessible version of OpenAI's text generator

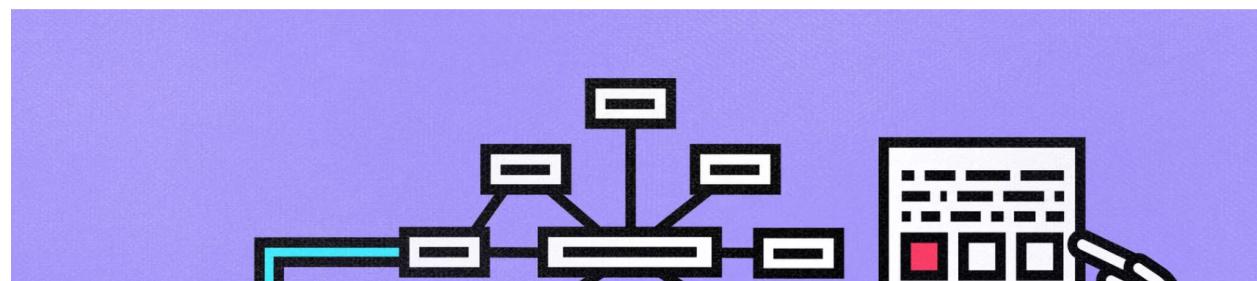
By James Vincent | May 13, 2019, 10:52am EDT



Listen to this article



SHARE



Deep Networks: Revolution (??)

The violin you hear
was generated by AI.

▶ ▶️ 🔊 0:03 / 11:45

Turning BASS into violin (using AI)

221.320 visualizzazioni • 20 ott 2020

16.201 123 CONDIVIDI SALVA ...

28 / 38

Tools: Neural Network Zoo

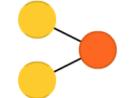
A mostly complete chart of

Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org

- Backfed Input Cell
- Input Cell
- △ Noisy Input Cell
- Hidden Cell
- Probabilistic Hidden Cell
- △ Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- △ Different Memory Cell
- Kernel
- Convolution or Pool

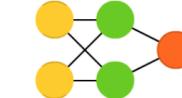
Perceptron (P)



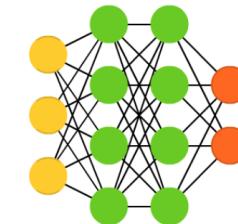
Feed Forward (FF)



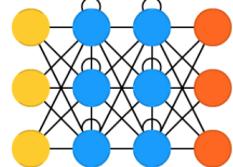
Radial Basis Network (RBF)



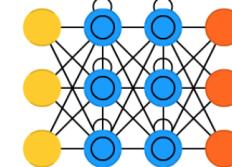
Deep Feed Forward (DFF)



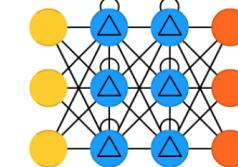
Recurrent Neural Network (RNN)



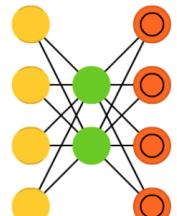
Long / Short Term Memory (LSTM)



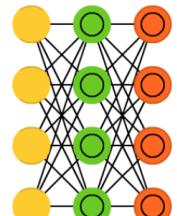
Gated Recurrent Unit (GRU)



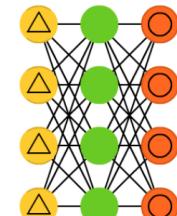
Auto Encoder (AE)



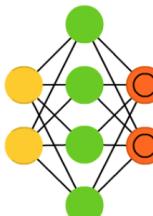
Variational AE (VAE)



Denoising AE (DAE)



Sparse AE (SAE)



Tools: Neural Network Playground

Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.

Epoch 000,000 Learning rate 0.03 Activation Tanh Regularization None Regularization rate 0 Problem type Classification

DATA
Which dataset do you want to use?

Ratio of training to test data: 50%
Noise: 0
Batch size: 10

FEATURES
Which properties do you want to feed in?
 X_1
 X_2
 X_1^2
 X_2^2
 $X_1 X_2$
 $\sin(X_1)$
 $\sin(X_2)$

2 HIDDEN LAYERS
+ -
4 neurons
2 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron. Hover to see it larger.

OUTPUT
Test loss 0.502
Training loss 0.549

Colors shows data, neuron and weight values.

Tools: TensorFlow



Tools: TensorFlow for R

The screenshot shows the homepage of the TensorFlow for R website. At the top, there is a navigation bar with links: Home (which is underlined), Installation, Tutorials, Guide, Deploy, Tools, API, Learn, and Blog. To the right of the navigation bar are icons for GitHub and a search bar. Below the navigation bar, there is a large section titled "R Interface to Tensorflow" with a sub-section "Build, deploy and experiment easily with TensorFlow from R". To the right of this text are the R logo (a blue "R" inside a grey circle) and the TensorFlow logo (an orange stylized "T" and "F").

R Interface to
Tensorflow

Build, deploy and experiment easily with
TensorFlow from R

Installation

Get started with TensorFlow by following
our detailed installation guide.

Tutorials

In the tutorials section you will find
documentation for solving common
Machine Learning problems using
TensorFlow.

Guide

The guide section contains documents
with in depth explanations of how
TensorFlow works.

Tools: Colab



Tools: Colab for R

The screenshot shows the Colab for R interface. At the top, there's a toolbar with icons for file operations, a search bar, and user profile. Below the toolbar is a menu bar with options like File, Modifica, Visualizza, Inserisci, Runtime, Strumenti, Guida, and a message indicating all changes are saved. The main area contains a code editor with the following R code:

```
print(installed.packages())
```

	Package	LibPath	Version
IRdisplay	"IRdisplay"	"/usr/local/lib/R/site-library"	"1.0"
IRkernel	"IRkernel"	"/usr/local/lib/R/site-library"	"1.1.1"
pbdZMQ	"pbdZMQ"		
repr	"repr"		
uuid	"uuid"		
askpass	"askpass"		
assertthat	"assertthat"		
backports	"backports"		
base64enc	"base64enc"		
BH	"BH"		
blob	"blob"		
brew	"brew"		
brio	"brio"		
broom	"broom"		
cachem	"cachem"		
callr	"callr"		
cellranger	"cellranger"		
cli	"cli"	"/usr/lib/R/site-library"	"2.4.0"
clipr	"clipr"	"/usr/lib/R/site-library"	"0.7.1"
colorspace	"colorspace"	"/usr/lib/R/site-library"	"2.0-0"
commonmark	"commonmark"	"/usr/lib/R/site-library"	"1.7"
covr	"covr"	"/usr/lib/R/site-library"	"3.5.1"
cpp11	"cpp11"	"/usr/lib/R/site-library"	"0.2.7"
crayon	"crayon"	"/usr/lib/R/site-library"	"1.4.1"
credentials	"credentials"	"/usr/lib/R/site-library"	"1.3.0"
crosstalk	"crosstalk"	"/usr/lib/R/site-library"	"1.1.1"
curl	"curl"	"/usr/lib/R/site-library"	"4.3"
DBI	"DBI"	"/usr/lib/R/site-library"	"1.1.1"
dbplyr	"dbplyr"	"/usr/lib/R/site-library"	"2.1.0"

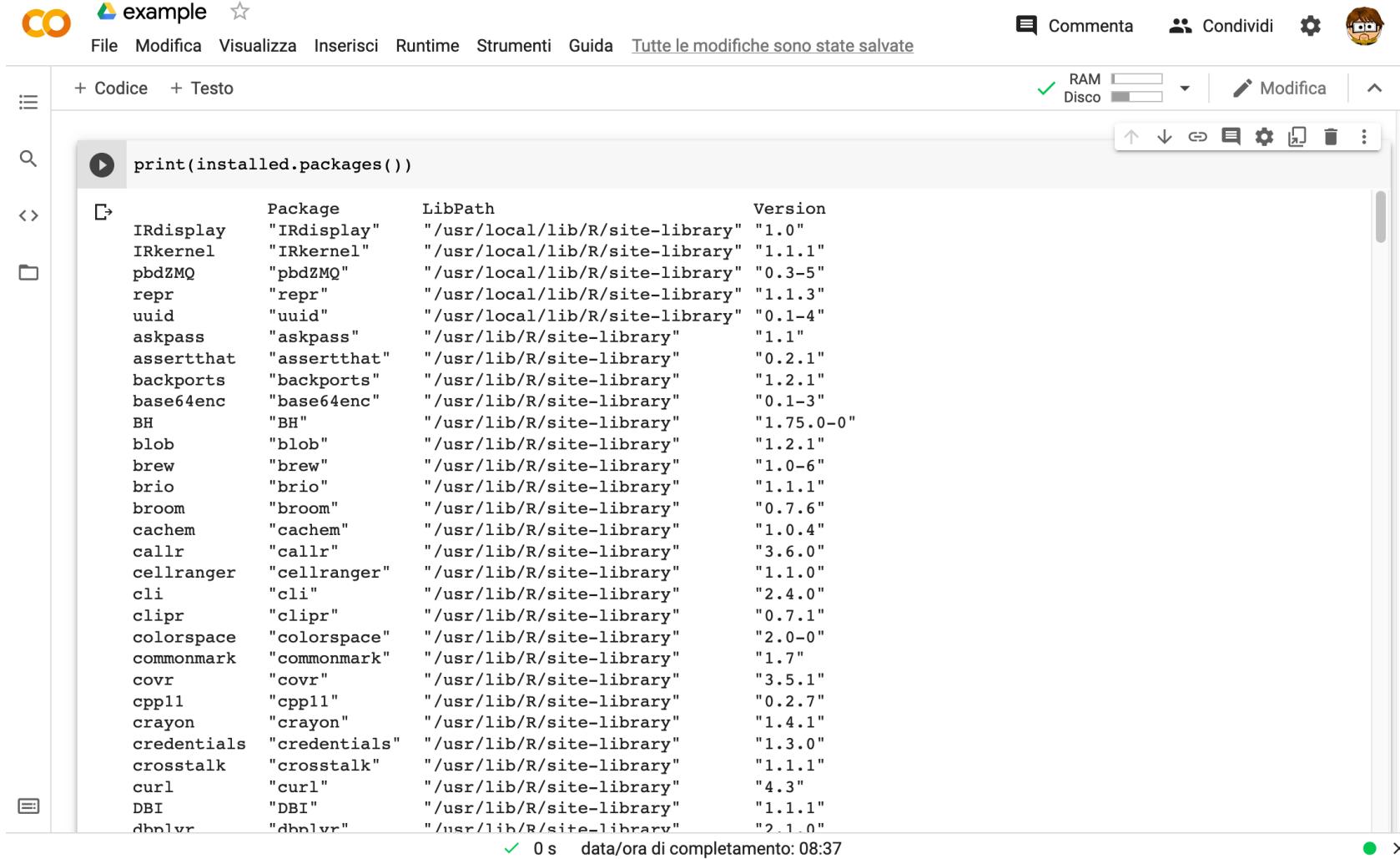
A modal dialog titled "Impostazioni blocco note" (Notebook settings) is open in the foreground. It contains the following options:

- "Tipo di runtime": A dropdown menu set to "R".
- "Accelerazione hardware": A dropdown menu set to "None".
- "Escludi output delle celle di codice durante il salvataggio del blocco note": A checkbox that is unchecked.

At the bottom of the dialog are two buttons: "ANNULLA" (Cancel) and "SALVA" (Save).

At the very bottom of the screen, there's a status bar with the text "✓ 0 s data/ora di completamento: 08:37".

Tools: Colab for R

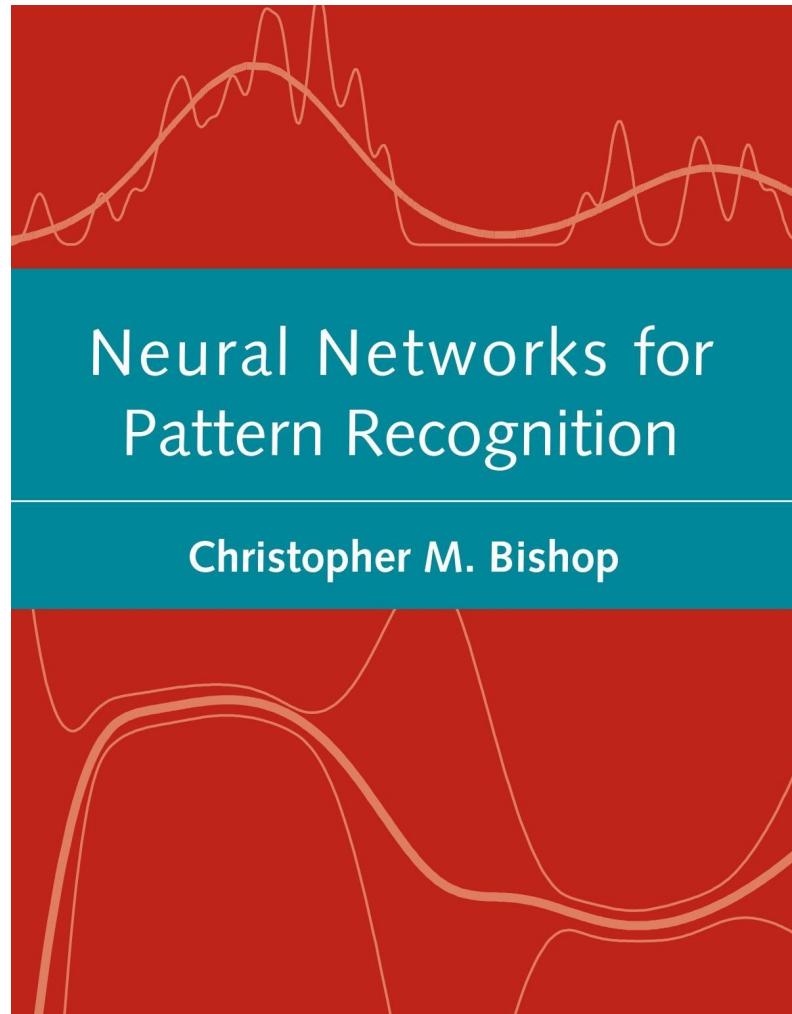


The screenshot shows the Colab for R interface. At the top, there's a toolbar with icons for CO, example, and a star. Below it is a menu bar with File, Modifica, Visualizza, Inserisci, Runtime, Strumenti, Guida, and a message that says "Tutte le modifiche sono state salvate". To the right of the menu are buttons for Commenta, Condividi, and settings. The main area has tabs for + Codice and + Testo. A code cell contains the command `print(installed.packages())`. The output of this command is a table listing various R packages with their details:

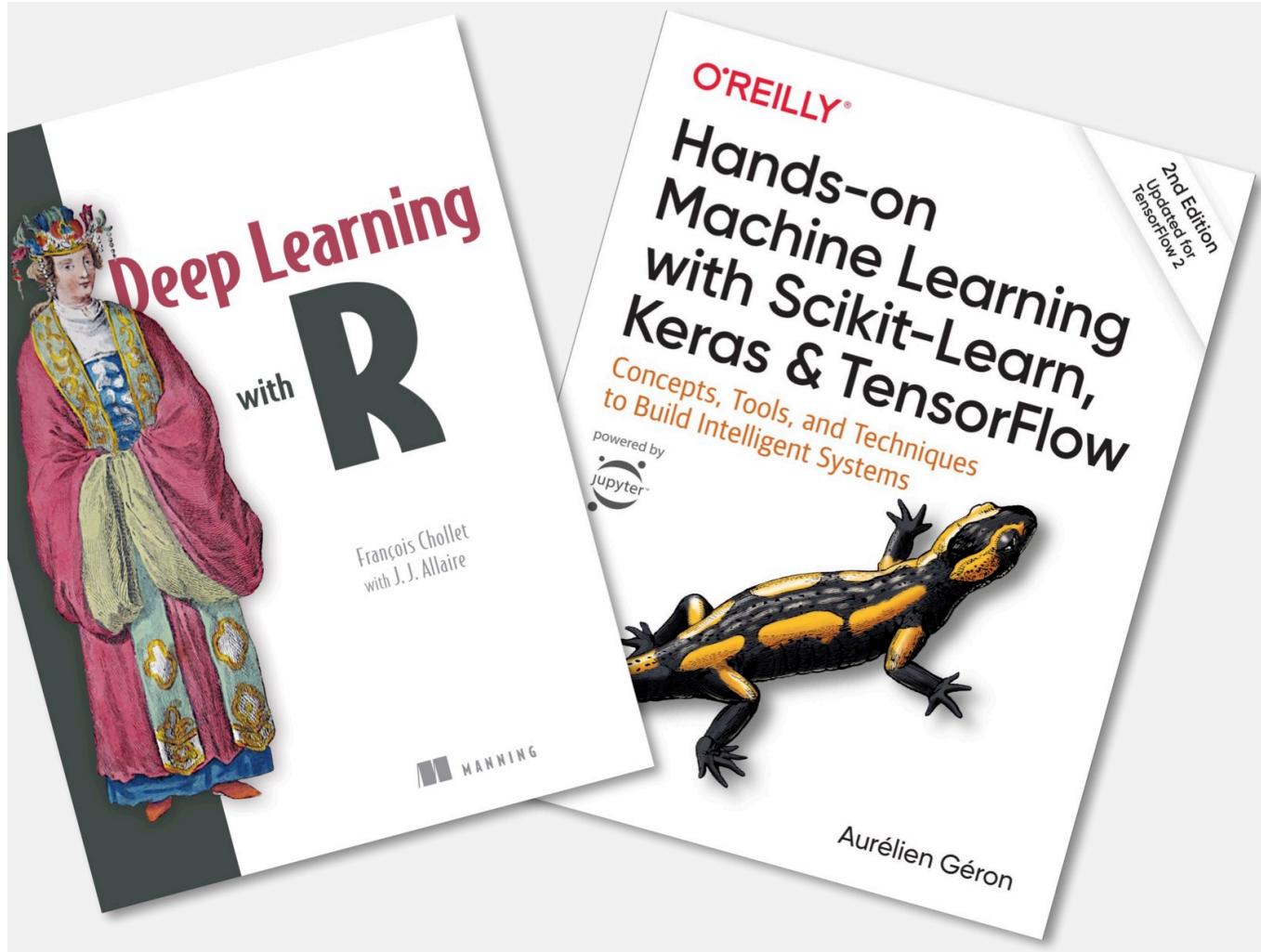
	Package	LibPath	Version
IRdisplay	"IRdisplay"	"/usr/local/lib/R/site-library"	"1.0"
IRkernel	"IRkernel"	"/usr/local/lib/R/site-library"	"1.1.1"
pbdZMQ	"pbdZMQ"	"/usr/local/lib/R/site-library"	"0.3-5"
repr	"repr"	"/usr/local/lib/R/site-library"	"1.1.3"
uuid	"uuid"	"/usr/local/lib/R/site-library"	"0.1-4"
askpass	"askpass"	"/usr/lib/R/site-library"	"1.1"
assertthat	"assertthat"	"/usr/lib/R/site-library"	"0.2.1"
backports	"backports"	"/usr/lib/R/site-library"	"1.2.1"
base64enc	"base64enc"	"/usr/lib/R/site-library"	"0.1-3"
BH	"BH"	"/usr/lib/R/site-library"	"1.75.0-0"
blob	"blob"	"/usr/lib/R/site-library"	"1.2.1"
brew	"brew"	"/usr/lib/R/site-library"	"1.0-6"
brio	"brio"	"/usr/lib/R/site-library"	"1.1.1"
broom	"broom"	"/usr/lib/R/site-library"	"0.7.6"
cachem	"cachem"	"/usr/lib/R/site-library"	"1.0.4"
callr	"callr"	"/usr/lib/R/site-library"	"3.6.0"
cellranger	"cellranger"	"/usr/lib/R/site-library"	"1.1.0"
cli	"cli"	"/usr/lib/R/site-library"	"2.4.0"
clipr	"clipr"	"/usr/lib/R/site-library"	"0.7.1"
colorspace	"colorspace"	"/usr/lib/R/site-library"	"2.0-0"
commonmark	"commonmark"	"/usr/lib/R/site-library"	"1.7"
covr	"covr"	"/usr/lib/R/site-library"	"3.5.1"
cpp11	"cpp11"	"/usr/lib/R/site-library"	"0.2.7"
crayon	"crayon"	"/usr/lib/R/site-library"	"1.4.1"
credentials	"credentials"	"/usr/lib/R/site-library"	"1.3.0"
crosstalk	"crosstalk"	"/usr/lib/R/site-library"	"1.1.1"
curl	"curl"	"/usr/lib/R/site-library"	"4.3"
DBI	"DBI"	"/usr/lib/R/site-library"	"1.1.1"
dhnpvr	"dhnpvr"	"/usr/lib/R/site-library"	"2.1.0"

At the bottom, there's a status bar with a green checkmark, 0 s, and the text "data/ora di completamento: 08:37".

Resources: A Classic



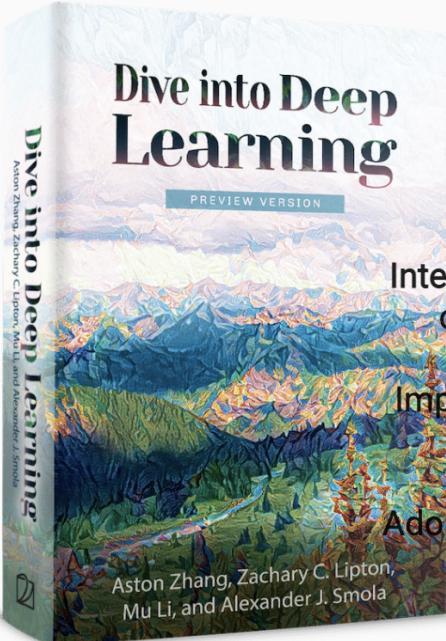
Resources: Deep in R and beyond



Resources: Deep Dive

Dive into Deep Learning

Courses PDF All Notebooks Discuss GitHub 中文版



Dive into Deep Learning
PREVIEW VERSION
Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola

Dive into Deep Learning

Interactive deep learning book with code, math, and discussions

Implemented with NumPy/MXNet, PyTorch, and TensorFlow

Adopted at 175 universities from 40 countries