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

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Linear Regression

Regression: $\underset{\texttt{Square Loss}}{L\big(Y, f(\boldsymbol{X})\big) = \big(Y - f(\boldsymbol{X})\big)^2} \qquad \underset{\texttt{Square Risk}}{R(f) = \mathbb{E}\big(Y - f(\boldsymbol{X})\big)^2} \qquad \underset{\texttt{Empirical Risk (OLS)}}{\widehat{R}(f) = \tfrac{1}{n} \textstyle\sum_{i=1}^n \big(Y_i - f(\boldsymbol{X}_i)\big)^2}$

Linear: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \Big\{ f : \mathbb{R}^p \mapsto \mathbb{R} \;\Big|\; f(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta} = \langle \boldsymbol{x}, \boldsymbol{\beta} \rangle \Big\}$ (Hypothesis Space)

$$R(\boldsymbol{\beta}) = \mathbb{E}\big(Y - \boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\big)^2 \;\rightsquigarrow\; \widehat{R}(\boldsymbol{\beta}) = \tfrac{1}{n} \textstyle\sum_{i=1}^n \big(Y_i - \boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}\big)^2 \propto \|\boldsymbol{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 \qquad (\mathbb{X} \rightsquigarrow n \times p \text{ Design Matrix})$$

Algorithm

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

Probabilistic Model / Likelihood

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

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

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Linear Regression

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

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 $(\mathbb{X}^{\mathsf{T}}\mathbb{X})$ is badly conditioned $\rightsquigarrow$ **multicollinearity** (i.e., correlated features)
- **Solution A:** QR-factorization (improve numerical stability)
  - **Remember:** $\boldsymbol{\beta}_{\text{emp}}$ solves the equation: $(\mathbb{X}^{\mathsf{T}}\mathbb{X})\,\boldsymbol{\beta} = \mathbb{X}^{\mathsf{T}}\boldsymbol{Y}$
  - **Write:** $\mathbb{X} = QR$ with $Q \in \mathbb{R}^{n \times p}$ *orthogonal* matrix ($Q^{\mathsf{T}}Q = \mathbb{I}_p$) and $R \in \mathbb{R}^{p \times p}$ upper triangular.
  - **Substitute:** $R^{\mathsf{T}}\big(Q^{\mathsf{T}}Q\big)R\,\boldsymbol{\beta} = R^{\mathsf{T}}Q^{\mathsf{T}}\boldsymbol{Y} \quad \Leftrightarrow \quad R^{\mathsf{T}}R\,\boldsymbol{\beta} = R^{\mathsf{T}}Q^{\mathsf{T}}\boldsymbol{Y} \quad \Leftrightarrow \quad R\,\boldsymbol{\beta} = Q^{\mathsf{T}}\boldsymbol{Y}$
  - **Moral:** just solve a *linear* system with a *triangular* matrix $\rightsquigarrow$ easy $\overset{\texttt{R}}{\rightsquigarrow}$ `lm()`

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Linear Regression

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

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 $(\mathbb{X}^\intercal \mathbb{X})$ 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:** $\widehat{R}(\boldsymbol{\beta}) = \frac{1}{n} \|\boldsymbol{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 = \frac{1}{n} (\boldsymbol{Y} - \mathbb{X}\boldsymbol{\beta})^\intercal (\boldsymbol{Y} - \mathbb{X}\boldsymbol{\beta}) = \frac{1}{n} \big(\boldsymbol{Y}^\intercal \boldsymbol{Y} - 2\, \boldsymbol{\beta}^\intercal \mathbb{X}^\intercal \boldsymbol{Y} + \boldsymbol{\beta}^\intercal (\mathbb{X}^\intercal \mathbb{X}) \boldsymbol{\beta}\big)$

$$\nabla \widehat{R}(\boldsymbol{\beta}) = \tfrac{2}{n} \big((\mathbb{X}^\intercal \mathbb{X}) \boldsymbol{\beta} - \mathbb{X}^\intercal \boldsymbol{Y}\big) \quad \rightsquigarrow \quad \boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \alpha\, \nabla \widehat{R}\big(\boldsymbol{\beta}^{(t)}\big)$$

  - If the data set is much too big $\rightsquigarrow$ go *Stochastic*!

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Logistic Regression

Cross-Entropy: $L\big(Y, f(\boldsymbol{X})\big) \overset{\bigstar}{=} -\big[Y \log f(\boldsymbol{X}) + (1-Y) \log\big(1 - f(\boldsymbol{X})\big)\big] \overset{\heartsuit}{=} -\Big[\log\big(1-f(\boldsymbol{X})\big) + Y \log \frac{f(\boldsymbol{X})}{1-f(\boldsymbol{X})}\Big] \qquad R(f) = \mathbb{E}\, L\big(Y, f(\boldsymbol{X})\big)$

$$\widehat{R}(f) \overset{\bigstar}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^{n} Y_i \log f(\boldsymbol{X}_i) + (1-Y_i)\big(1 - f(\boldsymbol{X}_i)\big) \overset{\heartsuit}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^{n} \log\big(1 - f(\boldsymbol{X}_i)\big) + Y_i \log \frac{f(\boldsymbol{X}_i)}{1-f(\boldsymbol{X}_i)} \qquad \texttt{(Empirical Risk)}$$

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Logistic Regression

Cross-Entropy: $L\big(Y, f(\boldsymbol{X})\big) \overset{\bigstar}{=} -\big[Y \log f(\boldsymbol{X}) + (1-Y) \log\big(1 - f(\boldsymbol{X})\big)\big] \overset{\heartsuit}{=} -\Big[\log\big(1 - f(\boldsymbol{X})\big) + Y \log \frac{f(\boldsymbol{X})}{1 - f(\boldsymbol{X})}\Big] \qquad R(f) = \mathbb{E}\, L\big(Y, f(\boldsymbol{X})\big)$

$\widehat{R}(f) \overset{\bigstar}{=} -\frac{1}{n} \sum_{i=1}^{n} Y_i \log f(\boldsymbol{X}_i) + (1 - Y_i)\big(1 - f(\boldsymbol{X}_i)\big) \overset{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^{n} \log\big(1 - f(\boldsymbol{X}_i)\big) + Y_i \log \frac{f(\boldsymbol{X}_i)}{1 - f(\boldsymbol{X}_i)}$ `(Empirical Risk)`

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \Big\{ f : \mathbb{R}^p \mapsto [0,1] \;\Big|\; f(\boldsymbol{x}) = \big(\mathsf{act} \circ g\big)(\boldsymbol{x}) \;\text{ with }\; \mathsf{act}(u) = \frac{1}{1 + \mathrm{e}^{-u}},\; g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\beta} \Big\}$ `(Hypothesis Space)`

$L\big(Y, g(\boldsymbol{X})\big) \overset{\bigstar}{=} Y \log\big(1 + \mathrm{e}^{-g(\boldsymbol{X})}\big) + (1 - Y) \log\big(1 + \mathrm{e}^{+g(\boldsymbol{X})}\big) \;\rightsquigarrow\; \widehat{R}(\boldsymbol{\beta}) \overset{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^{n} \log\big(1 + \mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}}\big) - Y_i \boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}$

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Logistic Regression

Cross-Entropy: $L\big(Y, f(\boldsymbol{X})\big) \overset{\bigstar}{=} -\big[Y \log f(\boldsymbol{X}) + (1-Y) \log\big(1 - f(\boldsymbol{X})\big)\big] \overset{\heartsuit}{=} -\Big[\log\big(1 - f(\boldsymbol{X})\big) + Y \log \frac{f(\boldsymbol{X})}{1 - f(\boldsymbol{X})}\Big] \qquad R(f) = \mathbb{E}\, L\big(Y, f(\boldsymbol{X})\big)$

$\widehat{R}(f) \overset{\bigstar}{=} -\frac{1}{n} \sum_{i=1}^{n} Y_i \log f(\boldsymbol{X}_i) + (1 - Y_i)\big(1 - f(\boldsymbol{X}_i)\big) \overset{\heartsuit}{=} -\frac{1}{n} \sum_{i=1}^{n} \log\big(1 - f(\boldsymbol{X}_i)\big) + Y_i \log \frac{f(\boldsymbol{X}_i)}{1 - f(\boldsymbol{X}_i)}$ `(Empirical Risk)`

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \Big\{ f : \mathbb{R}^p \mapsto [0,1] \;\Big|\; f(\boldsymbol{x}) = \big(\mathsf{act} \circ g\big)(\boldsymbol{x}) \;\text{ with }\; \mathsf{act}(u) = \frac{1}{1 + \mathrm{e}^{-u}},\; g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\beta} \Big\}$ `(Hypothesis Space)`

$L\big(Y, g(\boldsymbol{X})\big) \overset{\bigstar}{=} Y \log\big(1 + \mathrm{e}^{-g(\boldsymbol{X})}\big) + (1 - Y) \log\big(1 + \mathrm{e}^{+g(\boldsymbol{X})}\big) \;\rightsquigarrow\; \widehat{R}(\boldsymbol{\beta}) \overset{\heartsuit}{=} \frac{1}{n} \sum_{i=1}^{n} \log\big(1 + \mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}}\big) - Y_i \boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}$

Algorithm

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

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\texttt{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\texttt{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\texttt{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Logistic Regression

$$\textsf{Cross-Entropy:}\ L\big(Y, f(\boldsymbol{X})\big) \overset{\bigstar}{=} -\big[Y \log f(\boldsymbol{X}) + (1-Y) \log\big(1 - f(\boldsymbol{X})\big)\big] \overset{\heartsuit}{=} -\Big[\log\big(1 - f(\boldsymbol{X})\big) + Y \log \tfrac{f(\boldsymbol{X})}{1 - f(\boldsymbol{X})}\Big] \qquad R(f) = \mathbb{E}\, L\big(Y, f(\boldsymbol{X})\big)$$

$$\widehat{R}(f) \overset{\bigstar}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^n Y_i \log f(\boldsymbol{X}_i) + (1 - Y_i)\big(1 - f(\boldsymbol{X}_i)\big) \overset{\heartsuit}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^n \log\big(1 - f(\boldsymbol{X}_i)\big) + Y_i \log \tfrac{f(\boldsymbol{X}_i)}{1 - f(\boldsymbol{X}_i)} \qquad \texttt{(Empirical Risk)}$$

$$\textsf{Linear-Logistic:}\ \mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \Big\{ f : \mathbb{R}^p \mapsto [0,1] \;\Big|\; f(\boldsymbol{x}) = \big(\mathsf{act} \circ g\big)(\boldsymbol{x}) \ \text{ with } \ \mathsf{act}(u) = \tfrac{1}{1 + \mathrm{e}^{-u}},\ g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\beta} \Big\} \qquad \texttt{(Hypothesis Space)}$$

$$L\big(Y, g(\boldsymbol{X})\big) \overset{\bigstar}{=} Y \log\big(1 + \mathrm{e}^{-g(\boldsymbol{X})}\big) + (1 - Y) \log\big(1 + \mathrm{e}^{+g(\boldsymbol{X})}\big) \ \rightsquigarrow \ \widehat{R}(\boldsymbol{\beta}) \overset{\heartsuit}{=} \tfrac{1}{n} \textstyle\sum_{i=1}^n \log\big(1 + \mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}}\big) - Y_i \boldsymbol{X}_i^{\mathsf{T}} \boldsymbol{\beta}$$

Algorithm

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

Probabilistic Model / Likelihood

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

Discriminative Methods

$$p(y, \boldsymbol{x}) = p(y \mid \boldsymbol{x}) \cdot p(\boldsymbol{x}) \quad \rightsquigarrow \quad h_{\text{opt}}(\boldsymbol{x}) = \mathbb{I}\big(f_{\text{opt}}(\boldsymbol{x}) \geqslant \tfrac{1}{2}\big) \quad \text{with} \quad f_{\text{opt}}(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) \overset{Y \in \{0,1\}}{=} \mathbb{P}(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})$$

Logistic Regression

Cross-Entropy: $L\big(Y, f(\boldsymbol{X})\big) \overset{\bigstar}{=} -\big[Y \log f(\boldsymbol{X}) + (1-Y) \log\big(1 - f(\boldsymbol{X})\big)\big] \overset{\heartsuit}{=} -\Big[\log\big(1 - f(\boldsymbol{X})\big) + Y \log \frac{f(\boldsymbol{X})}{1 - f(\boldsymbol{X})}\Big] \qquad R(f) = \mathbb{E}\, L\big(Y, f(\boldsymbol{X})\big)$

$$\widehat{R}(f) \overset{\bigstar}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^{n} Y_i \log f(\boldsymbol{X}_i) + (1 - Y_i)\big(1 - f(\boldsymbol{X}_i)\big) \overset{\heartsuit}{=} -\tfrac{1}{n} \textstyle\sum_{i=1}^{n} \log\big(1 - f(\boldsymbol{X}_i)\big) + Y_i \log \frac{f(\boldsymbol{X}_i)}{1 - f(\boldsymbol{X}_i)} \qquad \texttt{(Empirical Risk)}$$

Linear-Logistic: $\mathcal{F} = \mathcal{F}(\boldsymbol{\beta}) = \Big\{ f : \mathbb{R}^p \mapsto [0,1] \;\Big|\; f(\boldsymbol{x}) = \big(\mathsf{act} \circ g\big)(\boldsymbol{x}) \text{ with } \mathsf{act}(u) = \frac{1}{1+\mathrm{e}^{-u}},\ g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta} \Big\} \qquad \texttt{(Hypothesis Space)}$

$$L\big(Y, g(\boldsymbol{X})\big) \overset{\bigstar}{=} Y \log\big(1 + \mathrm{e}^{-g(\boldsymbol{X})}\big) + (1-Y) \log\big(1 + \mathrm{e}^{+g(\boldsymbol{X})}\big) \quad \rightsquigarrow \quad \widehat{R}(\boldsymbol{\beta}) \overset{\heartsuit}{=} \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \log\big(1 + \mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}}\big) - Y_i \boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}$$

Numerical Solutions

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

$$\frac{\partial \widehat{R}}{\partial \beta_j} \overset{\heartsuit}{=} \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \frac{1}{1+\mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}}} \frac{\partial}{\partial \beta_j}\big(1+\mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}}\big) - Y_i X_{i,j} = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \frac{\mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}}}{1+\mathrm{e}^{\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}}} \frac{\partial}{\partial \beta_j}\big(\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}\big) - Y_i X_{i,j} = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \mathsf{act}\big(\boldsymbol{X}_i^{\mathsf{T}}\boldsymbol{\beta}\big) X_{i,j} - Y_i X_{i,j}$$

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

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 $\boldsymbol{x}$ 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

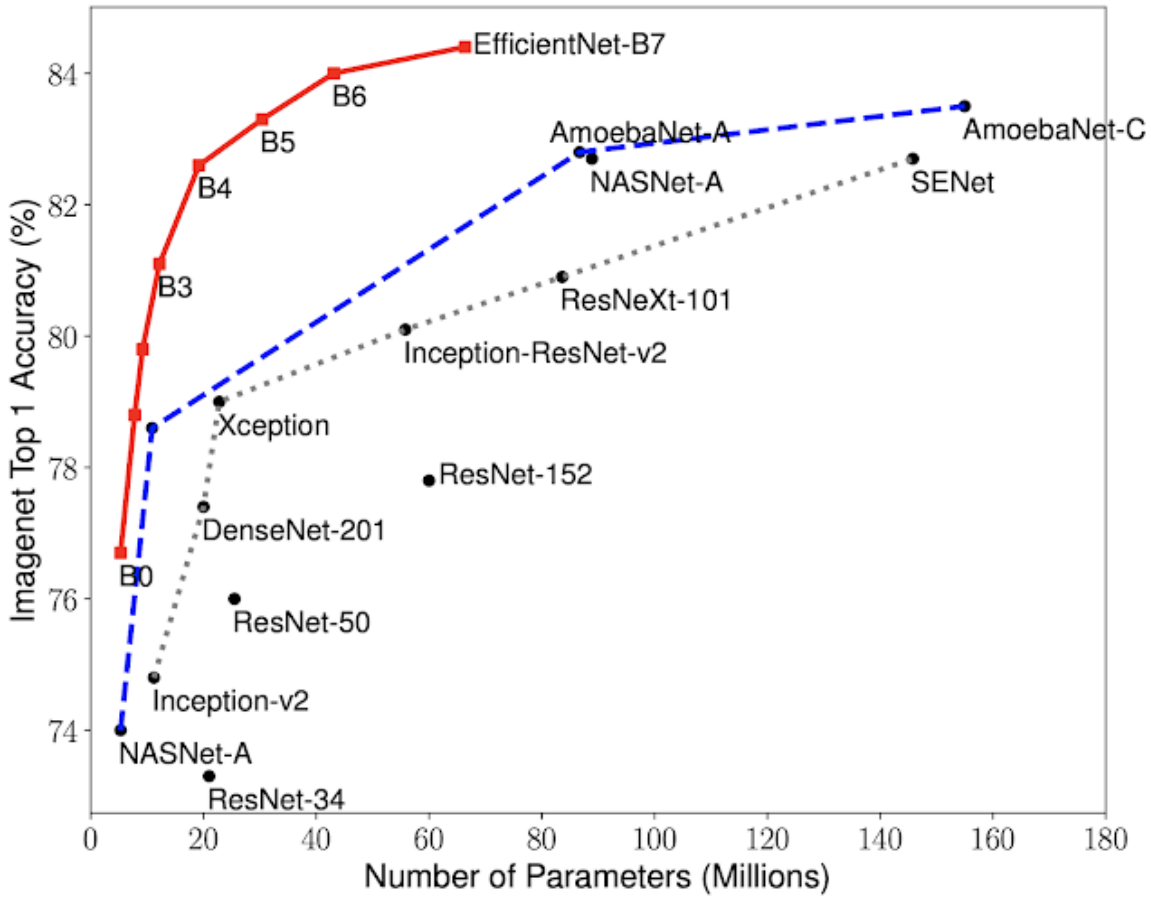
- The **backwards propagation** of errors, or just *backpropagation*, is the standard algorithm for computing gradients in a neural network.
- It is conceptually based on applying the **chain rule** to each layer of the network in **reverse order**.
- The conceptual idea behind backpropagation can be applied to any hierarchical model described by a directed acyclic graph (DAG) $\rightsquigarrow$ here focus on dense neural networks.
- **Step 1**
  - Insert an input $\boldsymbol{x}$ 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...)

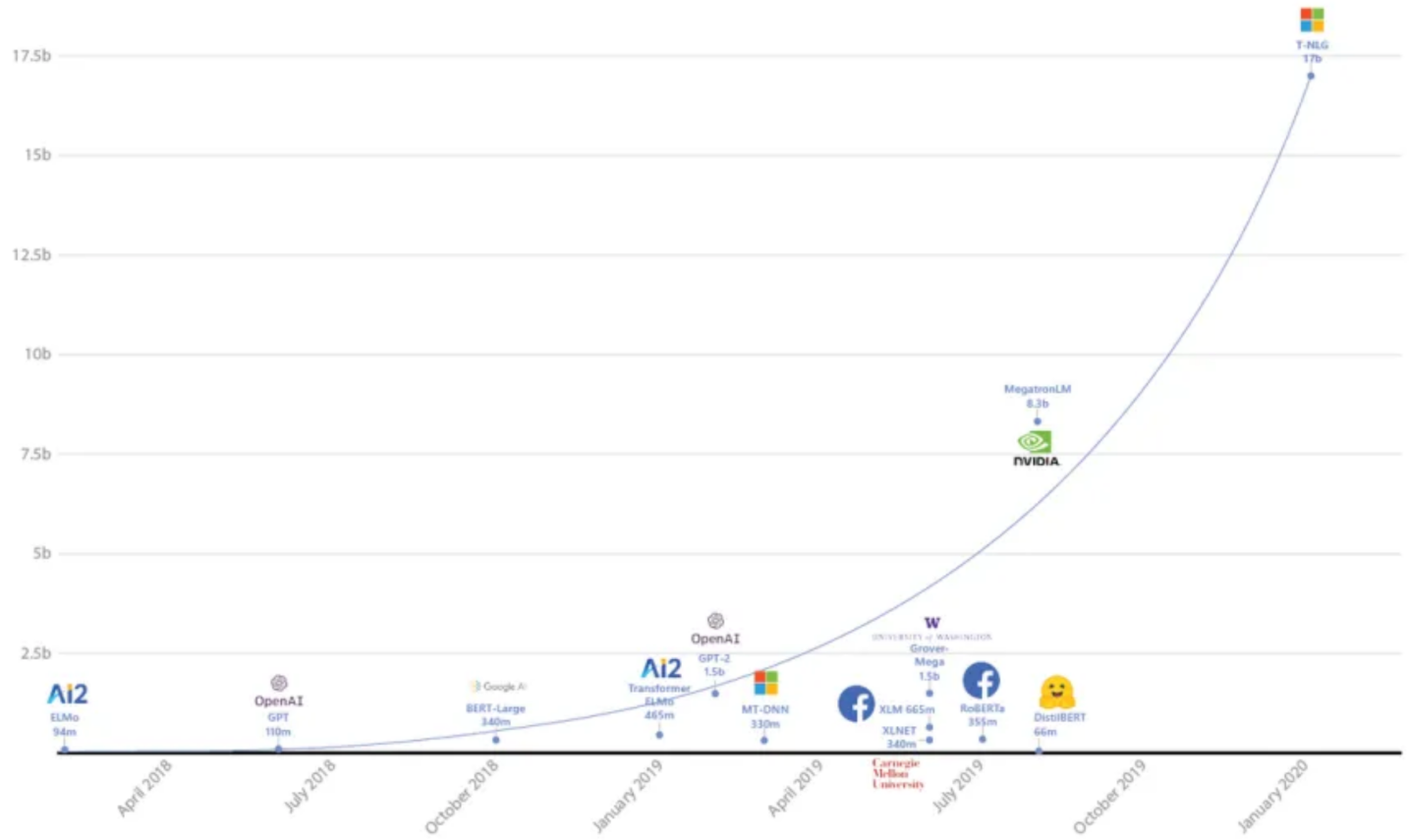
How Deep?

"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

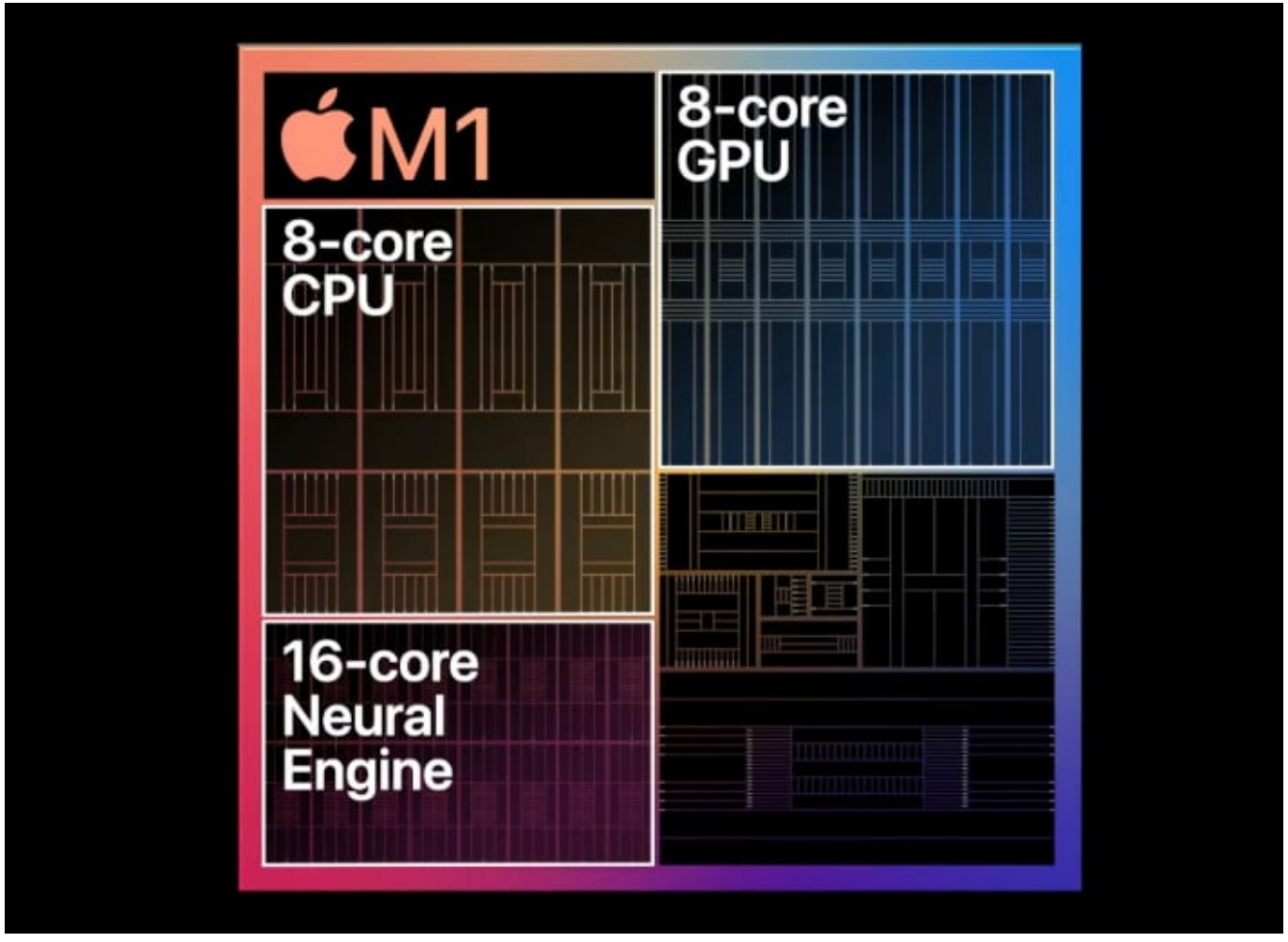
Bengio, Hinton and LeCun Ushered in Major Breakthroughs in Artificial Intelligence

Deep Networks: Alchemy

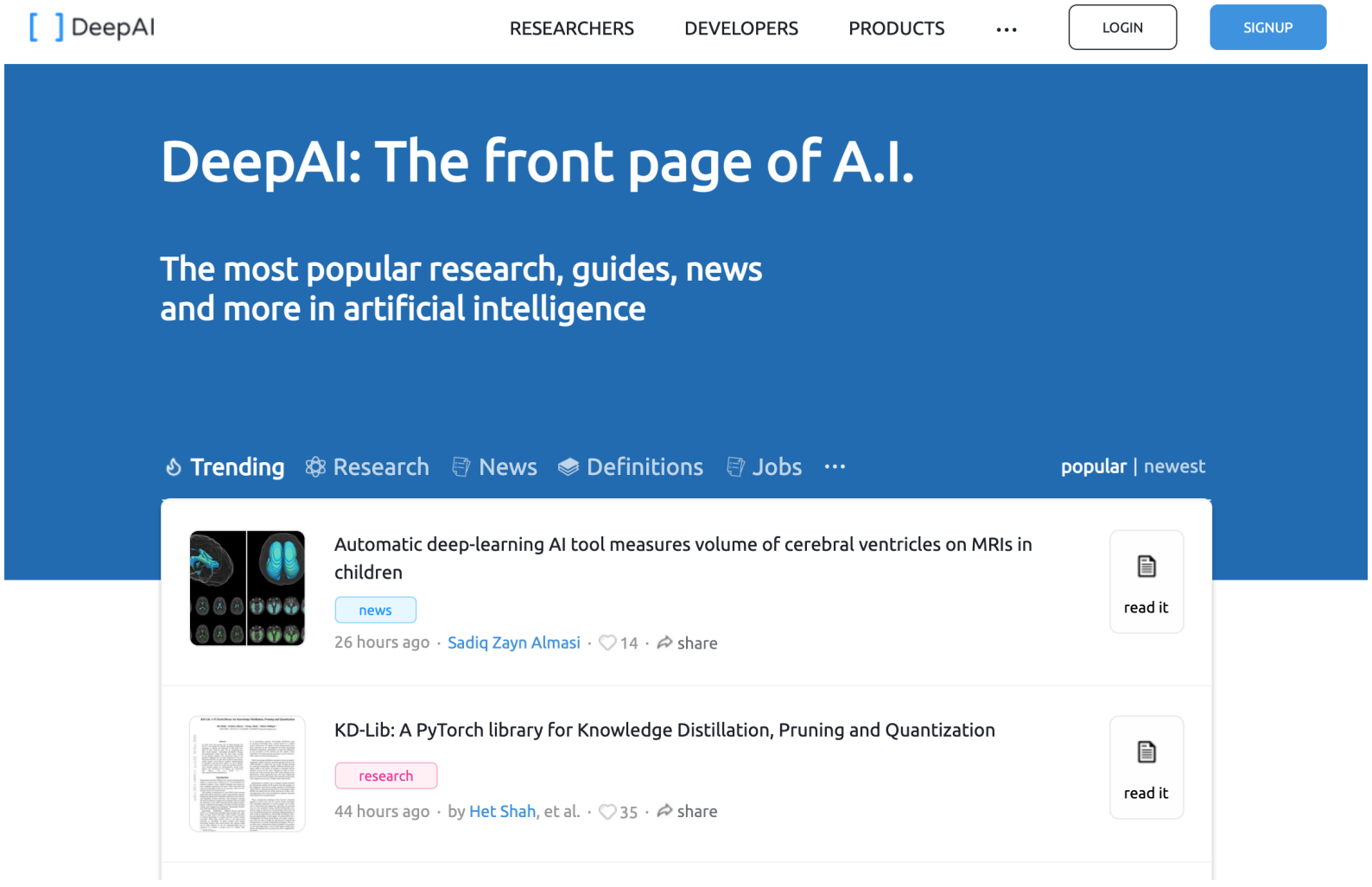
LeCun vs Rahimi: Has Machine Learning Become Alchemy?

Synced Dec 13, 2017 · 4 min read ★

Deep Networks: Revolution



Deep Networks: Revolution (?)



Deep Networks: Revolution (?)

TECH | ARTIFICIAL INTELLIGENCE

TL;DR

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

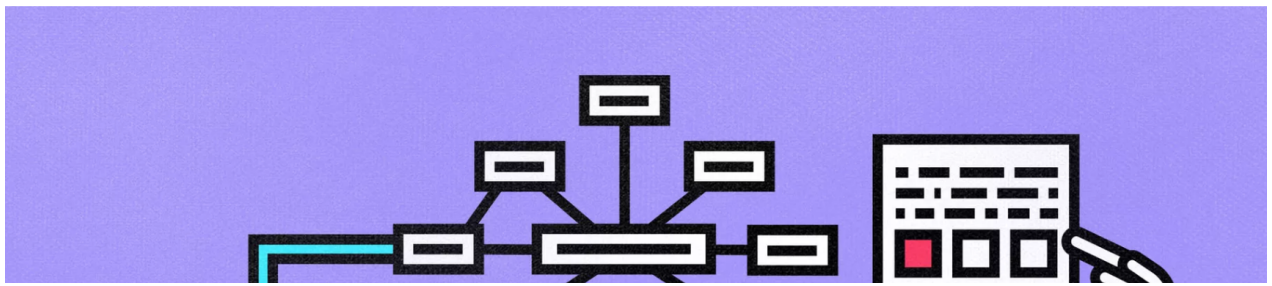
2

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

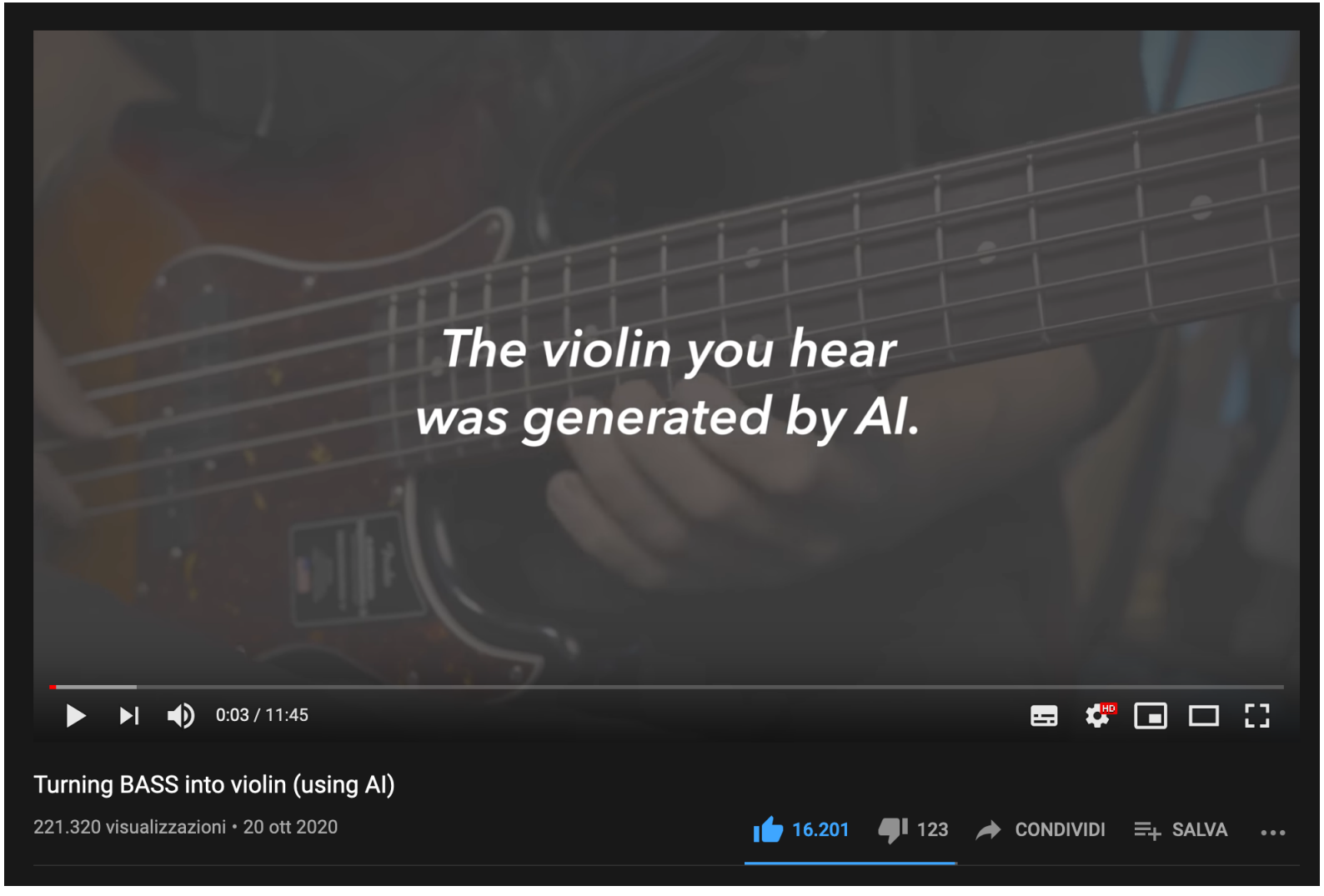
By James Vincent | May 13, 2019, 10:52am EDT

Listen to this article

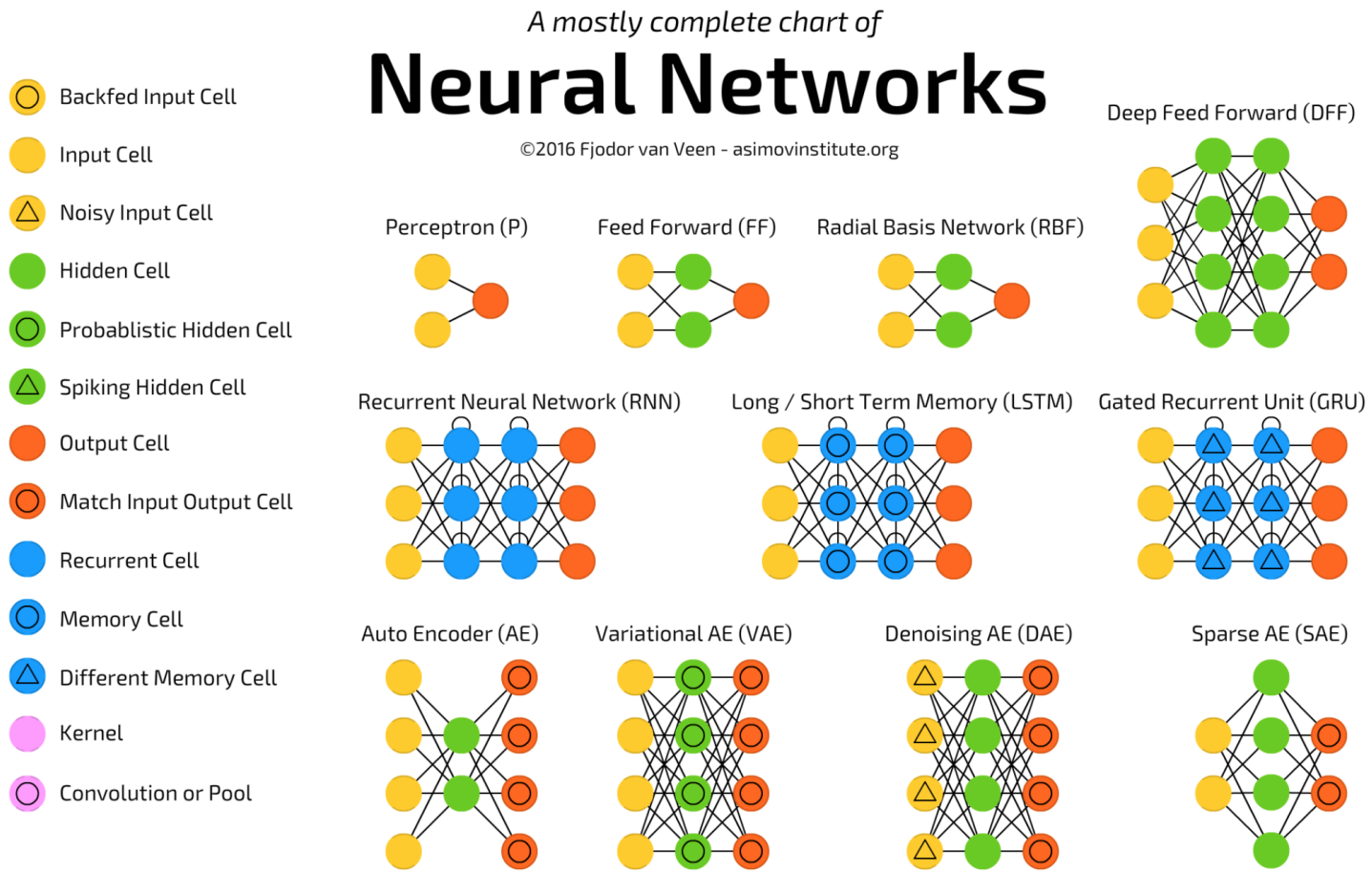
SHARE



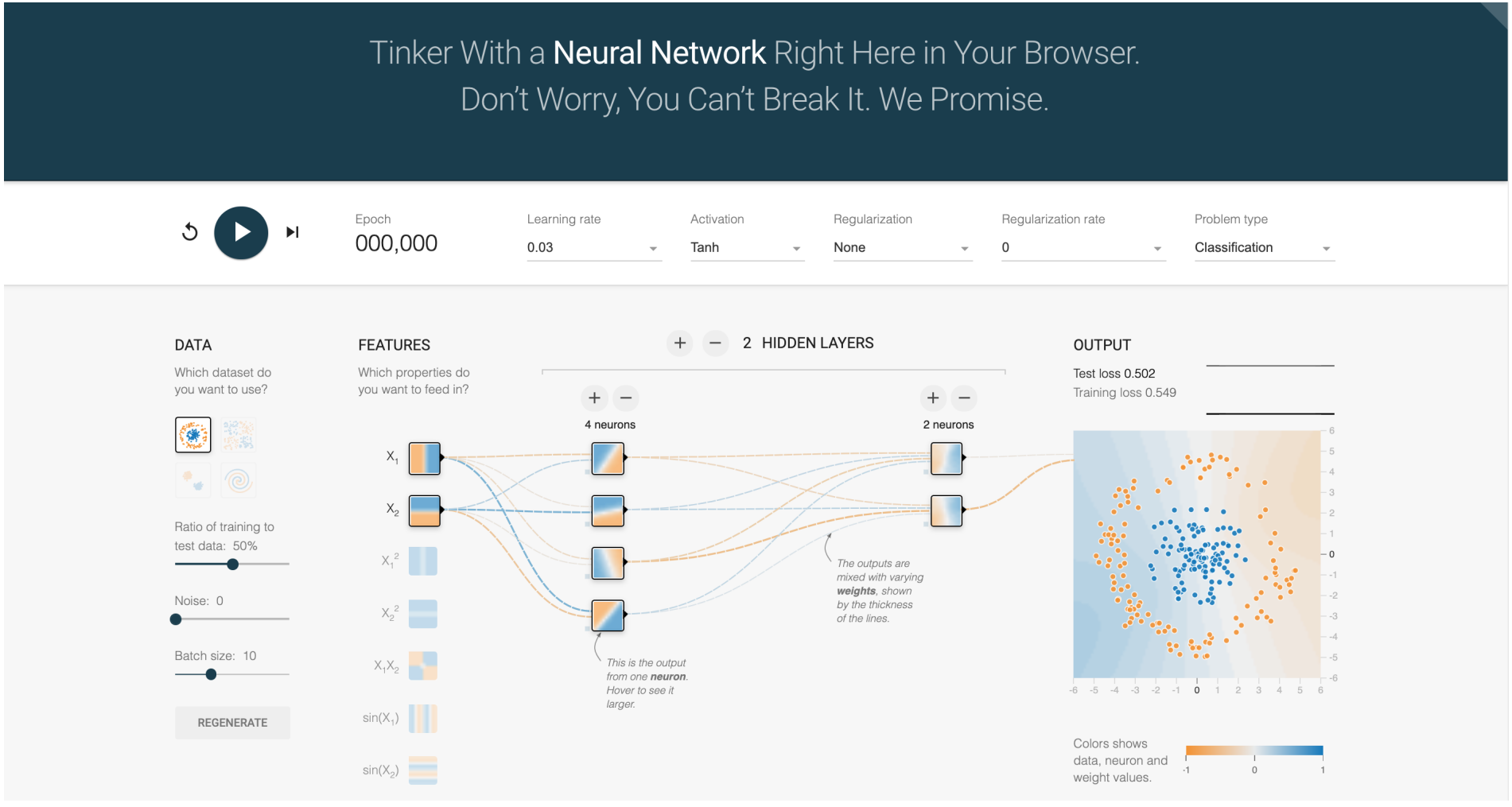
Deep Networks: Revolution (??)



Tools: Neural Network Zoo

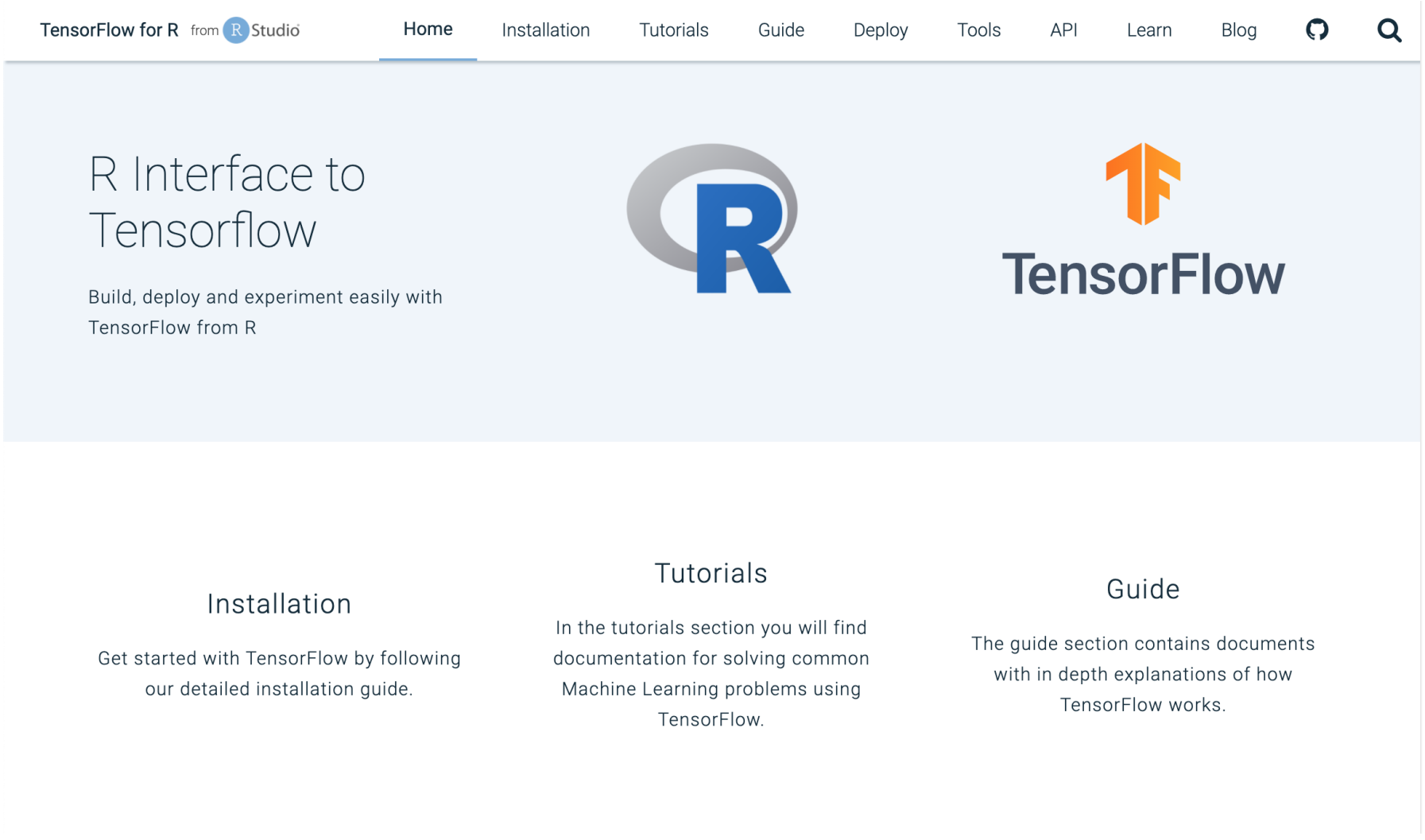


Tools: Neural Network Playground



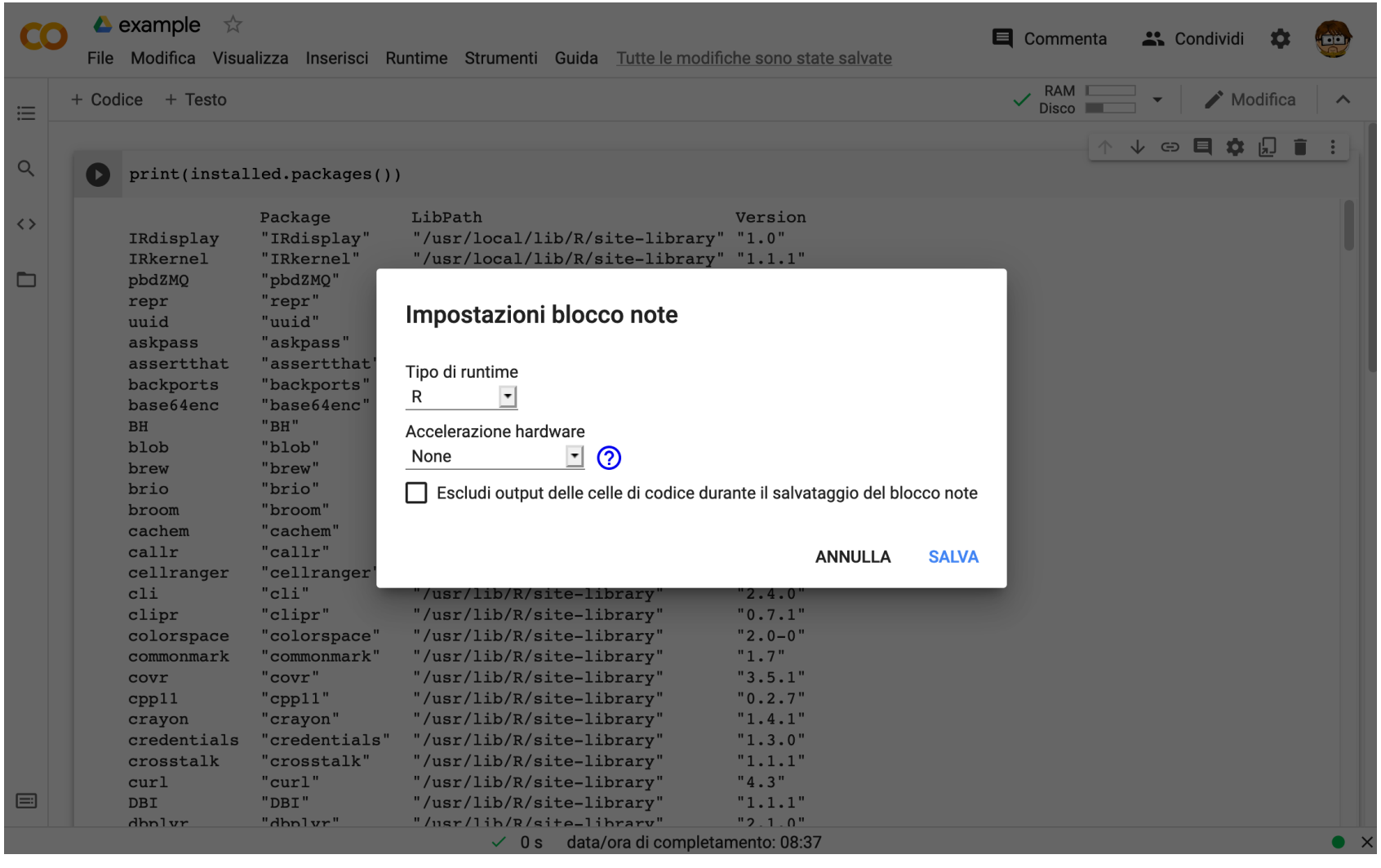
Tools: TensorFlow

Tools: TensorFlow for R

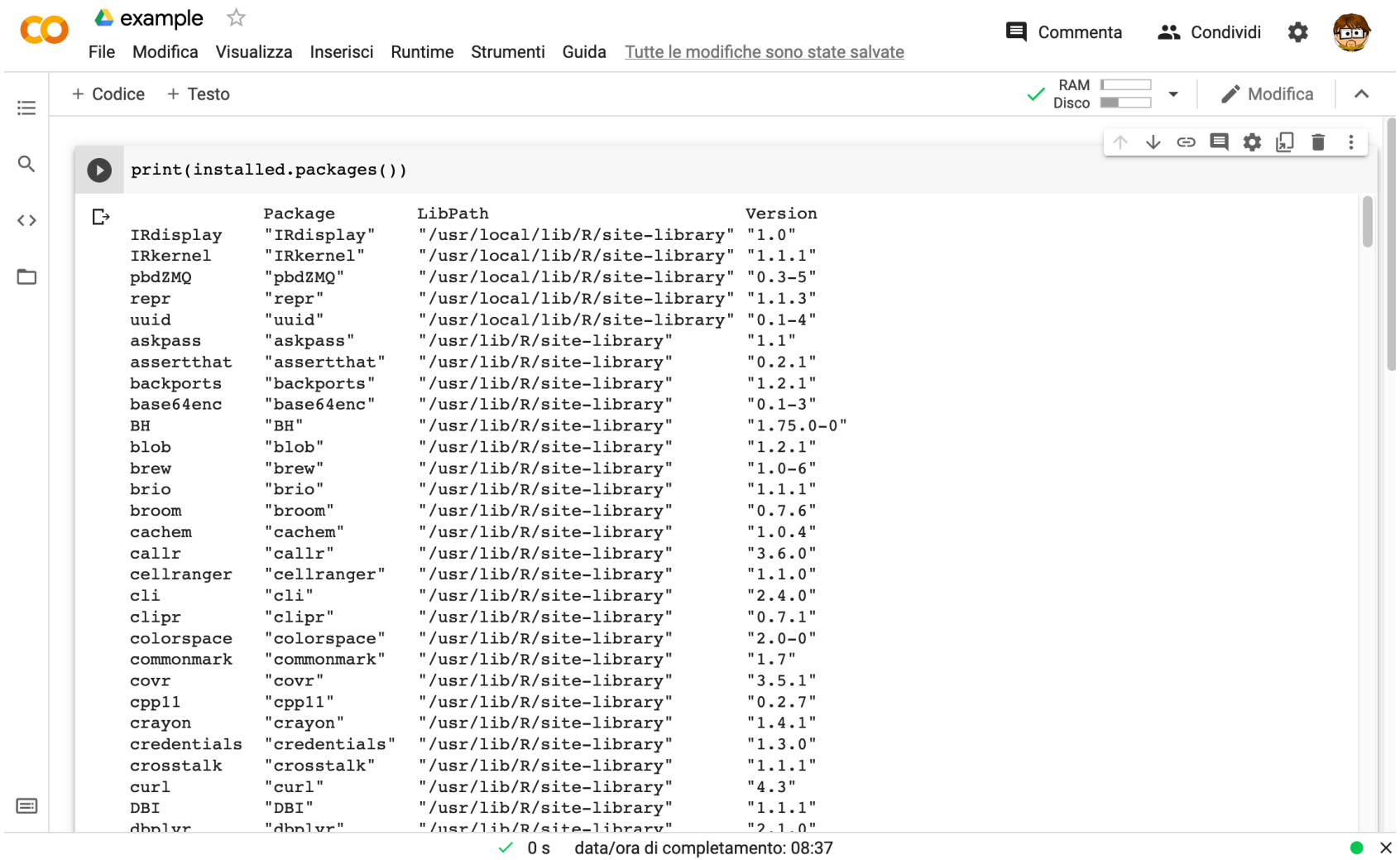


Tools: Colab

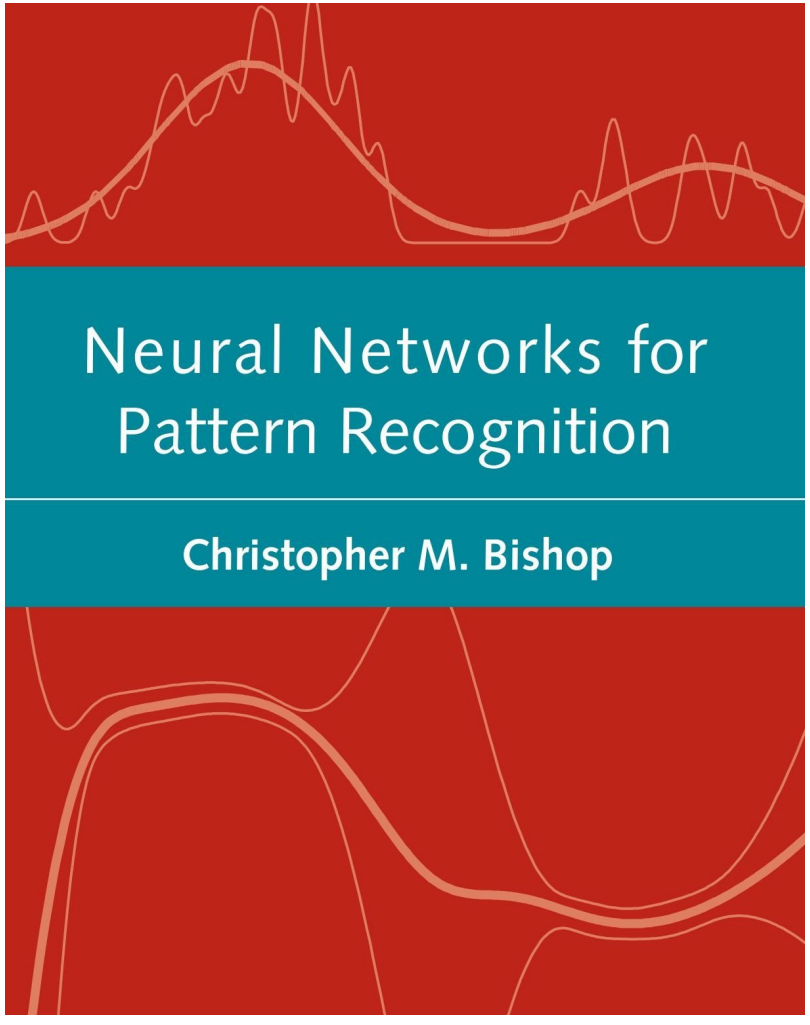
Tools: Colab for R



Tools: Colab for R



Resources: A Classic



Resources: Deep in R and beyond

Resources: Deep Dive