Experimental Design for Machine Learning on Multimedia Data

Lecture 3

Dr. Gerald Friedland,
fractor@eecs.berkeley.edu

Website: http://www.icsi.berkeley.edu/~fractor/spring2019/
Discuss Homework

- Homework 1: Any questions
- Homework 2: Let’s do it.
- Homework 3: online today and also in paper here

Please start forming teams.

Rishi is now official and has office hours to discuss homework & project.
Reminder: The New Scientific Method

\[ E = mc^2 \]
Reminder: Thought Framework Machine Learning

- Intelligence: *The ability to adapt* (Binet and Simon, 1904)

- Machine learning *adapts a finite state machine* $M$ *to an unknown function based on observations.*

- Input: $n$ rows of observations (instances) in a table with header:

$$ (x_1, x_2, \ldots, x_m, f(\vec{x})) $$

where $f(\vec{x})$ is a column with labels we call target function.

- Output: State machine $M$ that maps a point

$$ (x_1, x_2, \ldots, x_m) \implies f(\vec{x}) $$
Thought Framework: Machine Learning

Assume

$$x_i \in \mathbb{R}, f(\vec{x}) \in \{0,1\}$$

(binary classifier)

Question:

How many state transitions does $M$ need to model the training data?
Refresh: Memory Arithmetic

• Information is reduction of uncertainty:  
  \[ H = -\log_2 P = -\log_2 \frac{1}{\text{#states}} = \log_2 \text{#states} \]  measured in bits.

• Information: \( \log_2 \text{#states} \) (positive bits)  
  Uncertainty: \( \log_2 P = \log_2 \frac{1}{\text{#states}} \) (negative bits)

• If states are not equiprobable, Shannon Entropy provides tighter bound.

Important for homework!
Thought Framework: Machine Learning

Assume

\[ x_i \in \mathbb{R}, f(\vec{x}) \in \{0, 1\} \]

(binary classifier)

Question:

How many state transitions does \( M \) need to model the training data?

Maximally: \#rows (lookup table)
Minimally: ?
Thought Framework: Machine Learning

- **Intellectual Capacity**: The number of unique target functions a machine learner is able to represent (as a function of the number of model parameters).

- **Memory Equivalent Capacity (MEC)**: A machine learner’s intellectual capacity is memory-equivalent to $N$ bits when the machine learner is able to represent all $2^N$ binary labeling functions of $N$ uniformly random inputs.

- At MEC or higher, $M$ is able to memorize all possible state transitions from the input to the output.
Project Question 2

- What is the estimated memory-equivalent capacity of the data you have for the machine learner you are using?
- Is there bias? If so: Normalize it.
- Test at least 3 different machine learners.
Memorization is worst-case generalization

- Using more parameters than needed for memorization is a waste of resources (CPU, memory, I/O, engineer tuning time).
- Using as many parameters as needed for memorization will most likely not generalize to a held-out data set. This, the machine learner overfits.
- Reducing parameters below memorization capacity will, in the best case, make the machine learner forget what’s not relevant: generalization.
Machine Learning as Engineering Discipline

• **Supervised Machine Learners have a memory capacity in bits** that is **computable** and **measurable**.
  
  • Artificial Neural Networks with gating functions (Sigmoid, ReLU, etc.) have
    
    • a capacity upper limit that can be determined analytically using 4 principles
    
    • an effective capacity that can be measured on actual implementations.

• Predicting and measuring capacity allows for task-independent optimization of a concrete network architecture, learning algorithm, convergence tricks, etc...

• Capacity requirement can be approximately predicted given the input data and ground truth.

• Generalization can fail as a result of input redundancies. Occam’s Razor helps to minimize the risk.
Repeat: The Perceptron

Physical interpretation: Energy threshold

## Repeat: Activation Functions (too many)

<table>
<thead>
<tr>
<th>Function</th>
<th>Formula</th>
<th>Derivative</th>
<th>Domain</th>
<th>Differentiability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity</td>
<td>$f(x) = x$</td>
<td>$f'(x) = 1$</td>
<td>$(-\infty, \infty)$</td>
<td>$C^\infty$</td>
</tr>
<tr>
<td>Binary step</td>
<td>$f(x) = \begin{cases} 0 &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(x) = \begin{cases} 0 &amp; \text{for } x \neq 0 \ ? &amp; \text{for } x = 0 \end{cases}$</td>
<td>${0,1}$</td>
<td>$C^{-1}$</td>
</tr>
<tr>
<td>Logistic (a.k.a. Soft step)</td>
<td>$f(x) = \frac{1}{1+e^{-x}}$</td>
<td>$f'(x) = f(x)(1-f(x))$</td>
<td>$(0,1)$</td>
<td>$C^\infty$</td>
</tr>
<tr>
<td>TanH</td>
<td>$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$</td>
<td>$f'(x) = 1 - f(x)^2$</td>
<td>$(-1,1)$</td>
<td>$C^\infty$</td>
</tr>
<tr>
<td>ArcTan</td>
<td>$f(x) = \tan^{-1}(x)$</td>
<td>$f'(x) = \frac{1}{x^2 + 1}$</td>
<td>$\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$</td>
<td>$C^\infty$</td>
</tr>
<tr>
<td>Softsign [7][8]</td>
<td>$f(x) = \frac{x}{1 +</td>
<td>x</td>
<td>}$</td>
<td>$f'(x) = \frac{1}{(1 +</td>
</tr>
<tr>
<td>Rectified linear unit (ReLU) [9]</td>
<td>$f(x) = \begin{cases} 0 &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(x) = \begin{cases} 0 &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$[0,\infty)$</td>
<td>$C^0$</td>
</tr>
<tr>
<td>Leaky rectified linear unit (Leaky ReLU) [10]</td>
<td>$f(x) = \begin{cases} 0.01x &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(x) = \begin{cases} 0.01 &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$(-\infty, \infty)$</td>
<td>$C^0$</td>
</tr>
<tr>
<td>Parameteric rectified linear unit (PReLU) [11]</td>
<td>$f(\alpha,x) = \begin{cases} \alpha x &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(\alpha,x) = \begin{cases} \alpha &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$(-\infty, \infty)$</td>
<td>$C^0$</td>
</tr>
<tr>
<td>Randomized leaky rectified linear unit (RRelu) [12]</td>
<td>$f(\alpha,x) = \begin{cases} \alpha x &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(\alpha,x) = \begin{cases} \alpha &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$(-\infty, \infty)$</td>
<td>$C^0$</td>
</tr>
<tr>
<td>Exponential linear unit (ELU) [13]</td>
<td>$f(\alpha,x) = \begin{cases} \alpha(e^x - 1) &amp; \text{for } x &lt; 0 \ x &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$f'(\alpha,x) = \begin{cases} f(\alpha,x) + \alpha &amp; \text{for } x &lt; 0 \ 1 &amp; \text{for } x \geq 0 \end{cases}$</td>
<td>$(-\alpha, \infty)$ when $\alpha = 1$, otherwise $C^0$</td>
<td>$C^1$ when $\alpha = 1$, otherwise $C^0$</td>
</tr>
</tbody>
</table>

Activation functions approximate the sharp decision boundary.

How many functions can be modeled using a Perceptron?

\[ 0.9x_1 + 2x_2 \geq 1 \]
\[ 0.9x_1 + 2x_2 < 1 \]

Source: R. Rojas, Intro to Neural Networks
**Definition 3.1** (VC Dimension [47]). The VC dimension $D_{VC}$ of a hypothesis space $f$ is the maximum integer $D = D_{VC}$ such that some dataset of cardinality $D$ can be shattered by $f$. Shattered by $f$ means that any arbitrary labeling can be represented by a hypothesis in $f$. If there is no maximum, it holds $D_{VC} = \infty$. 

**Vapnik-Chervonenkis Dimension**
How many points can we label in general?

Formula by Schlaefli (1852):

\[ T(n, k) = T(n - 1, k) + T(n - 1, k - 1), \quad (3) \]

where \( T(n, 1) = T(1, k) = 2 \) or iteratively:

\[ T(n, k) = 2 \sum_{l=0}^{k-1} \binom{n - 1}{l} \quad (4) \]

<table>
<thead>
<tr>
<th>( n \setminus k )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>8</td>
<td>14</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>10</td>
<td>22</td>
<td>30</td>
<td><strong>32</strong></td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>12</td>
<td>32</td>
<td>52</td>
<td>62</td>
<td><strong>64</strong></td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>14</td>
<td>44</td>
<td>84</td>
<td>114</td>
<td>126</td>
<td><strong>128</strong></td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>16</td>
<td>58</td>
<td>128</td>
<td>198</td>
<td>240</td>
<td>254</td>
<td><strong>256</strong></td>
</tr>
</tbody>
</table>

Table 1: Some values of the \( T(n, k) \) function indicating the number of distinct threshold functions on \( n \) points in general position in \( k \) dimensions as defined by [22].

\[ T(n, k) = 2^n \text{ for } k \geq n. \]
Example: Boolean Functions

- $2^v$ functions of $v$ boolean variables
- $2^v$ labelings of $2^v$ points.
- For $v=2$, all but 2 functions work: XOR, NXOR

Source: R. Rojas, Intro to Neural Networks
Machine Learning as an Encoder/Decoder

Main trick: Let the Machine Learner learn uniform random points!

Source: D. MacKay: Information Theory, Inference and Learning
Critical Points: Perceptron (Cover, MacKay)

N=K: VC Dimension (for points in random position)
N=2K: Cover/MacKay Capacity

Source: D. MacKay: Information Theory, Inference and Learning
Generalizing from Perceptron to Perceptron Networks

FIGURE 4.1 Architectural graph of a multilayer perceptron with two hidden layers.

Careful: Other Architectures

Example Solutions to XOR

Source: R. Rojas, Intro to Neural Networks
Best Case Scenario?

Just measure in bits!

The *memory* capacity of any *binary classifier* cannot be better than the number of relevant bits in the model (pigeon hole principle, no universal lossless compression).

This is: $n$ bits in the model can *maximally* model $n$ bits of data.
Next Lecture

- Capacity for Neural Networks explained: See also cheat sheet.
- Practical applications
- Demo