Lecture 13. Deep Learning, Karnaugh Mapping, and Unsupervised Classification

Muhammad A. Alam
alam@purdue.edu
Copyright 2018

This material is copyrighted by M. Alam under the following Creative Commons license:

Conditions for using these materials is described at

http://creativecommons.org/licenses/by-nc-sa/2.5/
Course Outline

\[ \bar{y} = f(\bar{x}) \quad \bar{x} = x_1, x_2, \ldots, x_n \quad \bar{y} = y_1, y_2, \ldots, y_m \]

Lecture 1: Introduction
Lecture 2: Collecting and plotting \( x_1, x_2, \ldots, x_n \)
Lecture 3: Physical and empirical \( f, F, df/dx, \ldots \)
Lecture 4: Model selection between \( f_1, f_2, \ldots \)
Lecture 5: Model Selection: Cross-validation and Bootstrapping method
Lecture 6: Scaling theory with known \( f \), \( f(\bar{x}) = f(\bar{X}) \)
Lecture 7: Scaling theory with unknown \( f \), \( \bar{x} \rightarrow X \)
Lecture 8: Design of experiments to determine \( \bar{y}_{\text{max}} = f(\bar{x}) \)
Lecture 9: DOE and ANOVA
Lecture 11: Principle component analysis for classifying \{y\}.
Lecture 12-13: Machine learning … Statistical approach learn \( f \)
Lecture 14: Interpretable ML: Physics-based machine learning \( f = f_{\text{physics}} + \Delta f \)
Lecture 15: Conclusions
Outline

1. Introduction

2. A two input, single and multiple perceptron problem

3. Backpropagation and coefficient fitting

4. Machine learning and Karnaugh mapping

5. Other forms of Machine Learning (Unsupervised, optical, quantum)

6. Conclusions
Reliability of Solar Farms ...
.... represented by two input ANN

\[ w_1 \text{RH} + w_2 \text{T} = c_1 \]

\[ \sigma(w_1, w_2, c) = \frac{1}{1 + \exp(-(w_1 \text{T} + w_2 \text{RH} - c_1)/\sigma)} \]
Region defined by two lines
Region defined by multiple lines

![Diagram](image-url)
Deep network

Muhammad A. Alam, Purdue University
2D Image Recognition by Deep Network

- The network must fire if the input is in the coloured area

More complex decision boundaries

- Network to fire if the input is in the yellow area
  - “OR” two polygons

Complex decision boundaries

- Can compose arbitrarily complex decision boundaries
Outline

1. Introduction
2. A two input, single and multiple perceptron problem
3. Backpropagation and coefficient fitting
4. Machine learning and Karnaugh mapping
5. Other forms of Machine Learning (Unsupervised, optical, quantum)
6. Conclusions
Backpropagation algorithm

Input pair: 0.05, 0.10   Output pair 0.01, 0.99

\[ \text{net}_{h1} = 0.15 \times 0.05 + 0.2 \times 0.1 + 0.35 \times 1 = 0.3775 \]
\[ \text{out}_{h1} = \frac{1}{1 + e^{-\text{net}_{h1}}} = \frac{1}{1 + e^{-0.3775}} = 0.593269992 \]
\[ \text{net}_{o1} = 0.4 \times 0.593269992 + 0.45 \times 0.596884378 + 0.6 \times 1 = 1.105905967 \]
\[ \text{out}_{o1} = \frac{1}{1 + e^{-\text{net}_{o1}}} = \frac{1}{1 + e^{-1.105905967}} = 0.75136507 \]
\[ E_{o1} = \frac{1}{2}(\text{target}_{o1} - \text{out}_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083 \]
\[ E_{o2} = 0.023560026 \]
\[ E_{o2} = 0.772928465 \]
\[ E_{total} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371106 \]

... continued: Backpropagation algorithm

\[ \text{out}_{h1} = \frac{1}{1 + e^{-\text{net}_{h1}}} \]

\[ \frac{\partial \text{out}_{h1}}{\partial \text{net}_{h1}} = \text{out}_{h1}(1 - \text{out}_{h1}) = 0.59326999(1 - 0.59326999) = 0.241300709 \]

\[ nct_{i1} = w_i * i_1 + w_3 * i_2 + b_1 * 1 \]

\[ \frac{\partial nct_{i1}}{\partial w_i} = i_1 = 0.05 \]

\[ \frac{\partial E_{total}}{\partial w_i} = \frac{\partial E_{total}}{\partial \text{out}_{h1}} \frac{\partial \text{out}_{h1}}{\partial nct_i} \frac{\partial nct_i}{\partial w_i} \]

\[ \frac{\partial E_{total}}{\partial \text{net}_{h1}} = \frac{\partial E_{total}}{\partial \text{out}_{h1}} \frac{\partial \text{out}_{h1}}{\partial nct_{i1}} + \frac{\partial E_{total}}{\partial \text{net}_{h2}} \frac{\partial \text{net}_{h2}}{\partial nct_{i1}} \]

\[ \frac{\partial E_{total}}{\partial \text{net}_{h1}} = \frac{\partial E_{total}}{\partial \text{out}_{h1}} \frac{\partial \text{out}_{h1}}{\partial nct_{i1}} + \frac{\partial E_{total}}{\partial \text{net}_{h2}} \frac{\partial \text{net}_{h2}}{\partial nct_{i1}} \]

\[ w_i^{+} = w_i - \eta * \frac{\partial E_{total}}{\partial w_i} = 0.15 - 0.5 * 0.000438568 = 0.149780716 \]

\[ w_i^{+} = 0.19956143 \]

\[ w_j^{+} = 0.24975114 \]

\[ w_j^{+} = 0.29950229 \]

Old error: 0.298371109

New error: 0.291027924
Backpropagation algorithm

\[ E_{\text{total}} = \frac{1}{2}(\text{target}_{o1} - \text{out}_{o1})^2 + \frac{1}{2}(\text{target}_{o2} - \text{out}_{o2})^2 \]

\[ \frac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} = 2 \cdot \frac{1}{2}(\text{target}_{o1} - \text{out}_{o1})^{2-1} \cdot -1 + 0 \]

\[ \frac{\partial E_{\text{total}}}{\partial \text{out}_{o2}} = -(\text{target}_{o2} - \text{out}_{o2}) = -(0.01 - 0.75136507) = 0.74136507 \]

\[ \frac{\partial \text{out}_{o1}}{\partial \text{net}_{o1}} = \text{out}_{o1}(1 - \text{out}_{o1}) = 0.75136507(1 - 0.75136507) = 0.186815602 \]

\[ \text{net}_{o1} = w_5 \cdot \text{out}_{h1} + w_6 \cdot \text{out}_{h2} + b_2 \cdot 1 \]

\[ \frac{\partial \text{net}_{o1}}{\partial w_5} = 1 \cdot \text{out}_{h1} \cdot w_5^{(1-1)} + 0 + 0 = \text{out}_{h1} = 0.593269992 \]

\[ \frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} \cdot \frac{\partial \text{out}_{o1}}{\partial \text{net}_{o1}} \cdot \frac{\partial \text{net}_{o1}}{\partial w_5} \]

\[ \frac{\partial E_{\text{total}}}{\partial w_5} = 0.74136507 \cdot 0.186815602 \cdot 0.593269992 = 0.082167041 \]

\[ w_5^+ = w_5 - \eta \cdot \frac{\partial E_{\text{total}}}{\partial w_5} = 0.4 - 0.5 \cdot 0.082167041 = 0.35891648 \]

\[ w_6^+ = 0.511301270 \]

\[ w_k^+ = 0.561370121 \]
... continued: Updated Coefficients & Error

<table>
<thead>
<tr>
<th>Epoch</th>
<th>Epoch</th>
<th>Epoch</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1500</td>
<td>0.1498</td>
<td></td>
</tr>
<tr>
<td>0.2000</td>
<td>0.1996</td>
<td></td>
</tr>
<tr>
<td>0.2500</td>
<td>0.2498</td>
<td></td>
</tr>
<tr>
<td>0.3000</td>
<td>0.2995</td>
<td></td>
</tr>
<tr>
<td>0.4000</td>
<td>0.3589</td>
<td></td>
</tr>
<tr>
<td>0.4500</td>
<td>0.4087</td>
<td></td>
</tr>
<tr>
<td>0.5000</td>
<td>0.5113</td>
<td></td>
</tr>
<tr>
<td>0.5500</td>
<td>0.5614</td>
<td></td>
</tr>
<tr>
<td>i1,i2</td>
<td>o1,o2</td>
<td>o1,o2</td>
</tr>
<tr>
<td>0.05</td>
<td>0.7514</td>
<td>xxxxx</td>
</tr>
<tr>
<td>0.10</td>
<td>0.77293</td>
<td>xxxxx</td>
</tr>
<tr>
<td>Err.</td>
<td>0.2983</td>
<td>0.2910</td>
</tr>
</tbody>
</table>

After 10k iteration, gets within 1% of the final result (0.01, 0.99)
Outline

1. Introduction

2. A two input, single and multiple perceptron problem

3. Backpropagation and coefficient fitting

4. Machine learning and Karnaugh mapping

5. Other forms of Machine Learning (Unsupervised, optical, quantum)

6. Conclusions
Digital Synthesis has similar form

\[
y = \overline{A} \cdot B \cdot C + A \overline{B} \cdot C + A \cdot B \cdot C + A \cdot B \overline{C}
\]

\[
y = AC + BC + AB
\]

Online calculator: http://www.32x8.com/

Muhammad A. Alam, Purdue University
Any logic circuit can be synthesized by AND, OR, and NOT gates …
A perceptron implements AND, OR, NOT gates

\[ L - N + k \text{ here } L = \text{(positive input)}, N = \text{total number} = 2, k = 1 \text{ is the threshold for binary logic} \]

\[ L - \delta \]

\[ N - L + k \]

\[ A \rightarrow \text{AND} \rightarrow -0.5 \]

\[ A \rightarrow \text{OR} \rightarrow -0.5 \]

\[ A \rightarrow \text{NOT} \rightarrow -0.5 \]

Muhammad A. Alam, Purdue University
**XOR cannot be represented by one layer**

\[
Z = \overline{A} \cdot B + A \cdot \overline{B} \\
= (A + B) \cdot (\overline{A} + \overline{B}) \\
= (A + B) \cdot (A \cdot B)
\]
“XOR” implementation by ANN

\[ Z = \overline{A} \cdot B + A \cdot \overline{B} \]
\[ = (A + B) \cdot (\overline{A} + \overline{B}) \]

3 perceptrons, two layers, 6 weights and 3 thresholds, 9 parameters.

Muhammad A. Alam, Purdue University
Another example ....single layer in disjunctive normal form can express any truth table

<table>
<thead>
<tr>
<th>CD</th>
<th>AB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>00</td>
</tr>
<tr>
<td>00</td>
<td>1</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ Z = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot D + A \cdot \overline{(B)} \cdot C \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot D \]

\[ + A \cdot B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot \overline{D} \]

\[ \text{Exponential in } N \]

\[ \text{Requires } (O N \cdot 2^{N-1}) \text{ weights superexponential in } N \]
Depth vs. width trade-off (Karnaugh map reduction)

Depth vs. width can be traded off.

$$Z = \overline{A} \overline{B} + \overline{A} \overline{C} \overline{D} + A \overline{B} \overline{D}$$

Deep network requires $3(N-1)$ perceptrons with $9(N-1)$ parameters.

Linear in $N$, which can be arranged in $2 \log_2 N$ layers.

Alternatively, Shannon limit for $n$ input is at least $2^n/n$
Outline

1. Introduction
2. A two input, single and multiple perceptron problem
3. Backpropagation and coefficient fitting
4. Machine learning and Karnaugh mapping
5. Other forms of Machine Learning (Unsupervised, optical, quantum)
6. Conclusions
Unsupervised k-mean clustering

Supervised

Unsupervised

Two clusters
Pick centroids manually

Calculate ALL centroid-to-point distances.
Shorter distance wins the cluster

Move centroid to new average
Repeat until points do not change cluster
Supervised vs. unsupervised clustering

2-mean cluster

3-mean cluster

If you like this book, you may like this book (because the folks that belong to your cluster do)
Deep Learning with Coherent Nanophotonic Circuits

Deep Learning with Coherent Nanophotonic Circuits

Yuxun Song1, Nicholas C. Hartoe2, Xiao Sun1, Shige Zhu1, Hugo Larcher3, Erik Englund4, and Marin Soljačić2

27

Muhammad A. Alam, Purdue University
Deep Learning with Coherent Nanophotonic Circuits

Yichen Shen1, Nicholas C. Harris1, Scott Skirlo3, Mihika Prabhu1, Tom Baehr-Jones2, Michael Hochberg2, Xin Sun3, Shijie Zhou4, Hugo Larochelle2, Dirk Englund1, and Marin Soljačić1

FIG. 2. Illustration of Optical Interference Unit a. Optical micrograph of an experimentally fabricated 22-mode on-chip optical interference unit; the physical region where the optical neural network program exists is highlighted in grey. The system acts as an optical field-programmable gate array—a test bed for optical experiments. b. Schematic illustration of the optical neural network program demonstrated here which realizes both matrix multiplication and amplification fully optically. c. Schematic illustration of a single phase shifter in the Mach-Zehnder Interferometer (MZI) and the transmission curve for tuning the internal phase shifter of the MZI.
Conclusions

1. Multi-input, multiple layer network can be used to represent complex functional forms.

2. Since the approach does not rely on physics, it can handle complex interpolation problem. The out-of-domain predictions are difficult and error prone.

3. The mapping onto the digital logic synthesis (i.e. Karnaugh mapping) answers some key questions regarding the depth and width of the network. It also suggests how the neural network synthesizes logic step-by-step.

4. There are variety of machine learning tools: Supervised vs. unsupervised, random forest methods, optical methodologies. All these address specific issues, such as speed of classification, energy cost of training, etc.
References

The example involving passing probability vs. hours studied is taken from Real Statistics with Excel: Logistic Regression http://www.real-statistics.com/logistic-regression

Logistic Regression by Excel in Youtube: https://www.youtube.com/watch?v=EKRjDurXau0

Has a step by step analysis: https://www.youtube.com/watch?v=jiQ14p8KPIk4

The logistic calculator is here: http://astatsa.com/Logit_Probit/

The corresponding Wikipedia page gives detailed information https://en.wikipedia.org/wiki/Logistic_regression

Random Forest Model: https://www.youtube.com/watch?v=gmmV4drPT54

Random Forest Tutorial: https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377b35560d7d

Support Vector Machine p. 67 Machine Learning for Absolute Beginners by Oliver Theobald

An excellent presentation by Xavier Amatraian on NETFlix Recommendation Systems presented at 2012 ACM Meeting https://www.slideshare.net/xamat/netflix-recommendations-beyond-the-5-stars

Also see https://www.slideshare.net/xamat/kdd-2014-tutorial-the-recommender-problem-revisited


TensorFlow: REF: https://www.coursera.org/lecture/deep-learning-business/6-1-introduction-to-tensorflow-playground-ArB

Muhammad A. Alam, Purdue University
<table>
<thead>
<tr>
<th>Machine Learning References</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Intuitive explanation:</strong></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=nz-FrbAa8dY">https://www.youtube.com/watch?v=nz-FrbAa8dY</a></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=eX2sY2La4Ew">https://www.youtube.com/watch?v=eX2sY2La4Ew</a></td>
</tr>
<tr>
<td><strong>Excel:</strong></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=jQI4pkKP9k4">https://www.youtube.com/watch?v=jQI4pkKP9k4</a></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=gNhogkJ_q7U">https://www.youtube.com/watch?v=gNhogkJ_q7U</a></td>
</tr>
<tr>
<td><strong>Simple Visual Explanation</strong></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=yIYKR4sgzI8">https://www.youtube.com/watch?v=yIYKR4sgzI8</a></td>
</tr>
<tr>
<td><strong>Derivation of the parameters</strong></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=YMJtsYlp4kg">https://www.youtube.com/watch?v=YMJtsYlp4kg</a></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=YMJtsYlp4kg">https://www.youtube.com/watch?v=YMJtsYlp4kg</a></td>
</tr>
<tr>
<td><strong>Step by step:</strong></td>
</tr>
<tr>
<td><a href="https://www.youtube.com/watch?v=HQ7P-Ft7Cuc">https://www.youtube.com/watch?v=HQ7P-Ft7Cuc</a></td>
</tr>
</tbody>
</table>

---

**Artificial Neural Network and Digital Logic Synthesis**

- [http://toritris.weebly.com/](http://toritris.weebly.com/)
- [https://www.youtube.com/watch?v=RALqlk7T4xc](https://www.youtube.com/watch?v=RALqlk7T4xc)
- [http://www-inst.eecs.berkeley.edu/~ee40/fa03/lecture/lecture29.pdf](http://www-inst.eecs.berkeley.edu/~ee40/fa03/lecture/lecture29.pdf)
- [https://www.youtube.com/watch?v=FOf00W8WSBg](https://www.youtube.com/watch?v=FOf00W8WSBg)
- [https://www.youtube.com/watch?v=UdpV-ksatkQ](https://www.youtube.com/watch?v=UdpV-ksatkQ) (must make the group in powers of 2)

How many hidden layers:
- [https://cse.buffalo.edu/~hungngo/classes/2010/711/lectures/008_1.pdf](https://cse.buffalo.edu/~hungngo/classes/2010/711/lectures/008_1.pdf)

What size net gives valid generalization?
E. B. Baum and David Haussler
Machine Learning References

[1] Wide Residual Networks
Sergey Zagoruyko, Nikos Komodakis
(Submitted on 23 May 2016 (v1), last revised 14 Jun 2017 (this version, v4))
URL: https://arxiv.org/abs/1605.07146


Discrete Weights:

http://www.pnas.org/content/113/48/E7655.short
1. Any function can be represented by a single layer neurons. If so, why does one use multiple layer “deep” network?

2. Explain why we use tanh, sigmoid, or LiRu other saturated function in machine learning.

3. What are the support vectors in a support vector machine?

4. What are ID3, decision tree, and random forest model? Explain the applicability of the model.

5. What is the difference between supervised vs. non-supervised learning? How does the random clustering model work?