Lecture Note of EE: 541
A Computational Introduction to Deep Learning

Dr. Brandon Franzke

Scribe: Yi FAN

Contents

Part I

Basic theory

Preface

Welcome to EE 541: A computational Introduction to deep learning. We hope everything goes smoothly for you this semester.

This handout is organized into two main themes: Theory Backgrounds and Applications. The primary purpose of compiling this lecture handout is to help you grasp the core content of this course more conveniently and efficiently. The first theme answers the questions 'What' and 'Why', providing an extended and readable version of the PPTs, helping alleviate confusion or feeling of loss during seemingly overwhelming materials. The second theme delves deeper into the question of "How," offering examples, hints, and guidance for homework or other meaningful tasks.

Each theme consists of 8-9 chapters, aligned with the structure of our slides. Concision and readability come first! Our goal is to help students who read these materials gain clarity of what they are learning, why it matters, and how to apply it effectively.

Given my personal level of expertise, this lecture note may unavoidably have some shortcomings. I will ensure to update it weekly and actively seek your feedback for improvement.

1 Introduction Machine Learning

1.1 What is Machine Learning

Learning is any process by which a system improves performance from experience

- Herbert Simon

Machine learning, a key application of artificial intelligence (AI), enables systems to automatically learn and improve through experience. There are three key words inside this area: Performance P, Task T, and Experience E. A Well-defined learning can be concluded as improving Performance at given Task with Experience, {P,T,E}.

First and foremost, Learning is not a one-time event but a continuous process where systems iteratively improve its performance over time. Secondly, The ultimate goal of learning is generalization at given task. Be it accuracy in former predictions or low loss during training, the process cannot truly be considered fruitful if fails to generalize a satisfying outcome at this specific task. Eventually, learning is grounded in experience, i.e., data or observations. In machine learning, "experience" is represented by training datasets that allow the system to generalize and adapt. Pay close attention to your dataset, You can never spend too much time on it!

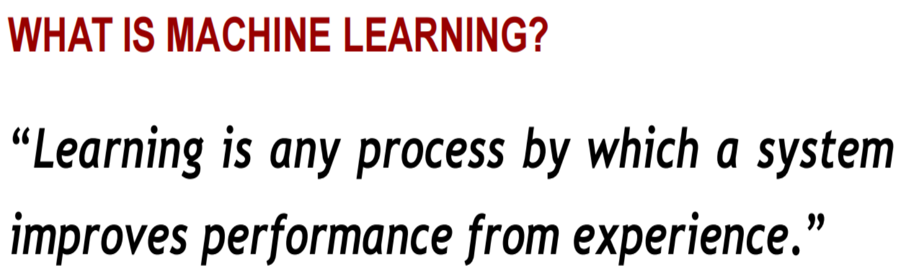

Figure 1: Definition of Machine Leaning

1.2 All models are wrong, But some are useful

- George E. P. Box

1.2.1 The Nature of Models as Simplifications

Machine learning models—be it linear regression, neural networks, or deep learning models—are simplifications or approximations of the real world. These models are built to capture patterns from data but cannot fully represent reality with all its complexity. For example: Data may contain noise or biases. A model's assumptions (e.g., linear relationships or specific distributions) often do not perfectly align with reality. Despite these imperfections, models remain effective tools for solving specific problems and making predictions within their intended scope.

1.2.2 The Usefulness of Models Lies in Their Context

The success of machine learning models depends heavily on the context of their application. For example: A simple linear regression model may fail to capture complex nonlinear patterns but could still be sufficient for certain straightforward tasks due to its

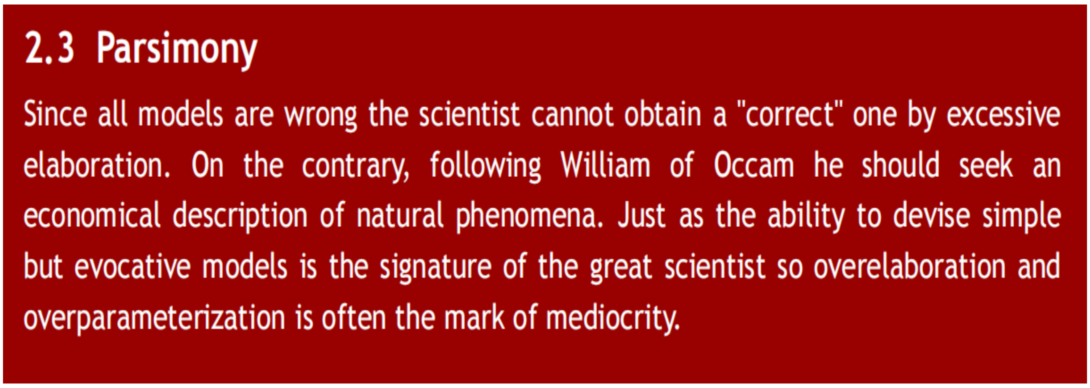

Figure 2: Accuracy- generalization trade-off

interpretability. A deep learning model may provide high accuracy but could perform poorly when there is insufficient data or computational resources. Only when applied to the right scenarios, 'some models are useful'.

1.2.3 Trade-off Between accuracy and Generalization

Generalization refers to a model's ability to perform well on unseen data. This is the ultimate goal of most machine learning tasks since models are deployed in environments different from their training sets. **Accuracy** is the proportion of correct predictions made by a model on a given dataset, typically the training or test data, and is often used as a primary metric for evaluating how well a model fits known data.

The conflict between accuracy and generalization arises because achieving perfect accuracy on the known training data often harms the model's ability to perform well on unseen data. This is directly tied to the bias-variance trade-off. As George E. P. Box put forwards: 'we should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity.' Generally, A 'useful' model strikes a balance: It simplifies reality just enough to generalize well. It avoids overfitting by not chasing extreme accuracy on training data, thereby maintaining robustness across datasets.

In conclusion, a model is always a simplification or abstraction of reality, which means it can never capture every detail perfectly. However, despite their imperfections, models can still provide valuable insights and guidance in understanding systems, making decisions, or predicting outcomes—provided they are applied within their intended scope.

1.3 Data Representation and Hypothesis Class

Let us dive into an interesting example to learn how to represent our data. As shown in Figure 3. We labeled three pictures with the labels 'good' or 'fail'. Your job is to decide what class does the following picture belong to. Isn't it somewhat unreasonable that you feel completely at a loss? Perhaps we can take a different perspective and explore this image in a binary manner, like a computer.

First, let's process this image in a top-left-to-bottom-right order, assigning white as 0 and black as 1. This 7x7 image will then be flattened into a 49-bit binary sequence. Our input now becomes x which has 49 dimensions $x_1, x_2, ..., x_{49}$. This seems easier

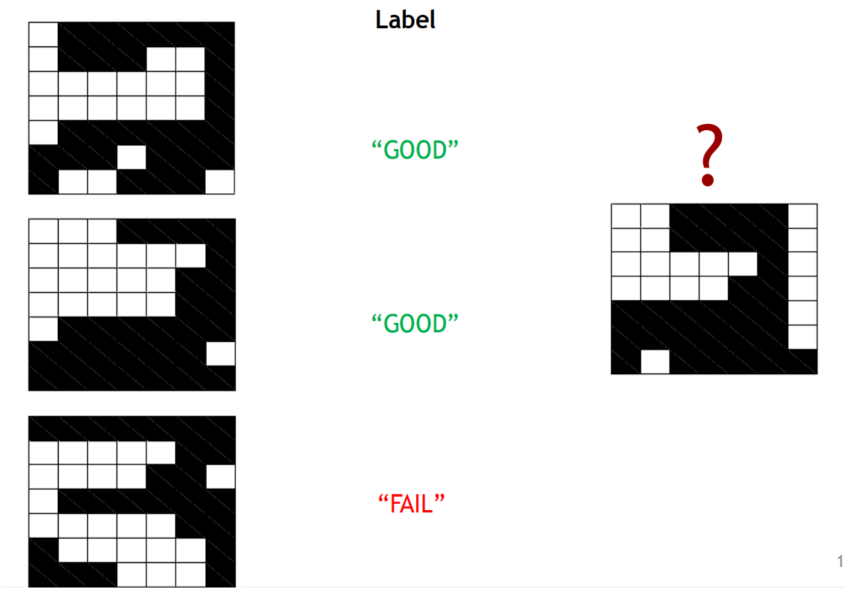

Figure 3: The data domain

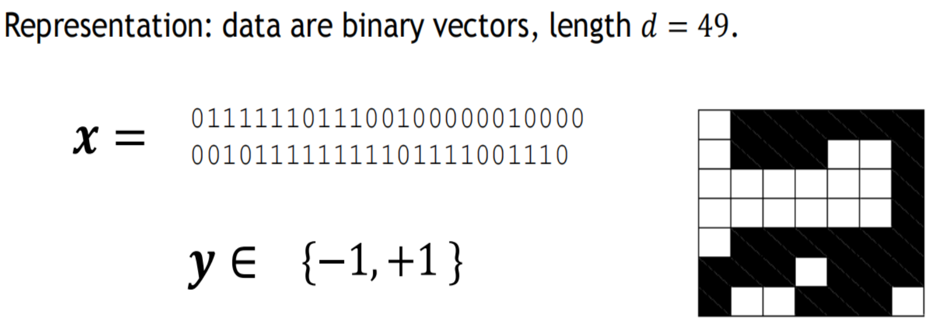

Figure 4: The binary representation

and more obvious. Then, let's mark good as $+1$ and fail as -1. Likely, out outcome has a new form y now, where y have two distinct values $+1$ or -1. Our job can be rewritten as finding a linear representation of y as a function of x. This task can be simply accomplished by machine learning methodology, hypothesizing mapping data to label using linear classifier. The math formula is shown below.

$$\hat{y} = \text{sign}(\boldsymbol{\theta} \cdot \boldsymbol{x}) = \text{sign}\left(\theta_1 x_1 + \cdots + \theta_{49} x_{49}\right) \tag{1}$$

Usually we use two types of learning methods. The closed-form, also known as explicit one, starts from calculating θ using many samples and then choose exact θ to minimize a cost function. The sequential(implicit) method includes guessing θ_1, evaluating model and updating/adjusting parameters if wrong $\theta_{i+1} = \theta_i + y x_i$.

1.4 Three types of Machine Learning

(1)Supervised Learning, pared data-points. Inference mode, predict new data.

(2)Unsupervised Learning, requires input but not desired output.

(3)Reinforcement Learning, e.g, game playing, no explicit model, no specific x,y pairs.

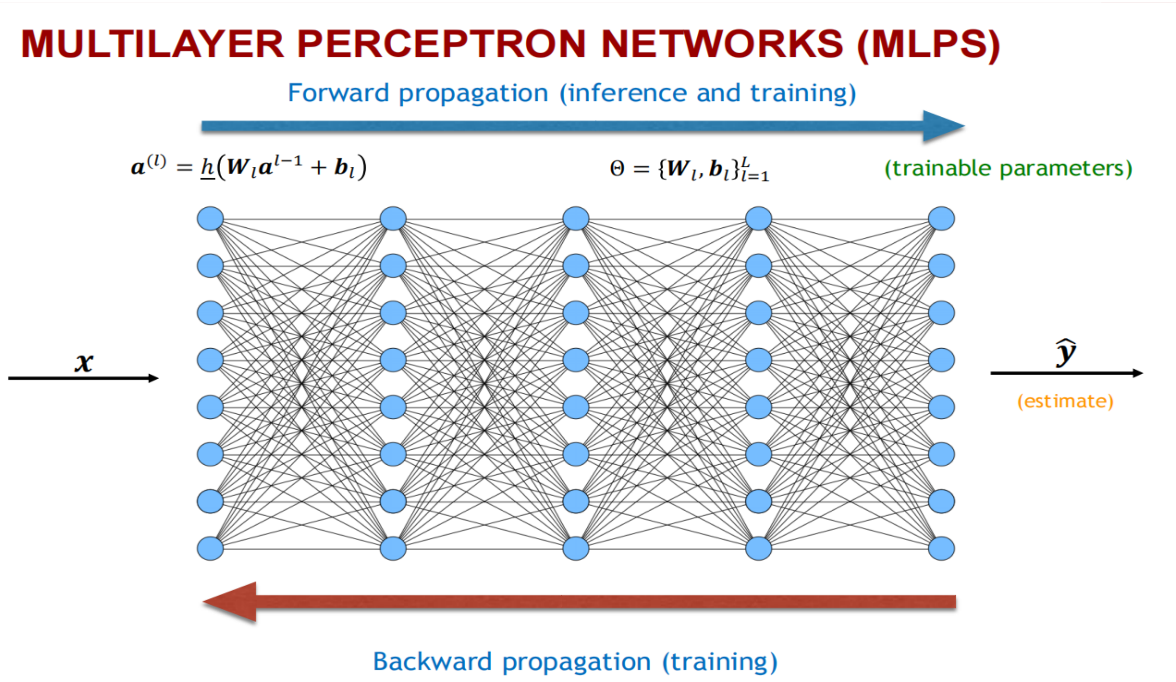

Figure 5: MLP Networks

1.5 Multilayer Perceptron Networks

MLP (Multi-Layer Perceptron) is a type of feedforward neural network composed of multiple layers of neurons, designed to solve supervised learning tasks, including classification and regression. An MLP consists of an input layer, one or more hidden layers, and an output layer. It achieves complex pattern learning through non-linear activation functions and weight optimization. The schematic diagram of MLP is shown in Figure 5.

Mathematically, an MLP can be expressed as:

$$\boldsymbol{a}^{(l)} = \underline{h}\left(\boldsymbol{W}_l\boldsymbol{a}^{l-1} + \boldsymbol{b}_l\right) \tag{2}$$

Where trainable parameters are:

$$\Theta = \{\boldsymbol{W}_l, \boldsymbol{b}_l\}_{l=1}^{L} \tag{3}$$

Here $a^{(l-1)}$ are inputs of layer l, W_l and b_l are weights and bias of layer l. $\underline{h}$ is the activation function, usually Sigmoid, tanh, Relu and so on. Finally, we have the output a_l of layer l. Here picture 6 showcases in detail how MLP process the input data at certain neuron.

(1)Linear Transformation: Each layer in an MLP performs a linear transformation of the input (via weights and biases). In general, MLP includes:

(2)Non-Linear Feature Extraction: Through activation functions (e.g., Sigmoid, ReLU), MLP enables the modeling of complex, non-linear relationships.

(3)Multi-Layer Structure: Unlike traditional models with shallow architectures, MLP introduces hidden layers to model, which captures the intricate patterns in the data.

(4)What's more, MLP uses backpropagation for weight optimization, which remains central to modern deep learning and will be discussed in detail in the following class. If we increase the number of hidden layers reasonably and go deeper, we will finally touch the area of **Deep Learning**.

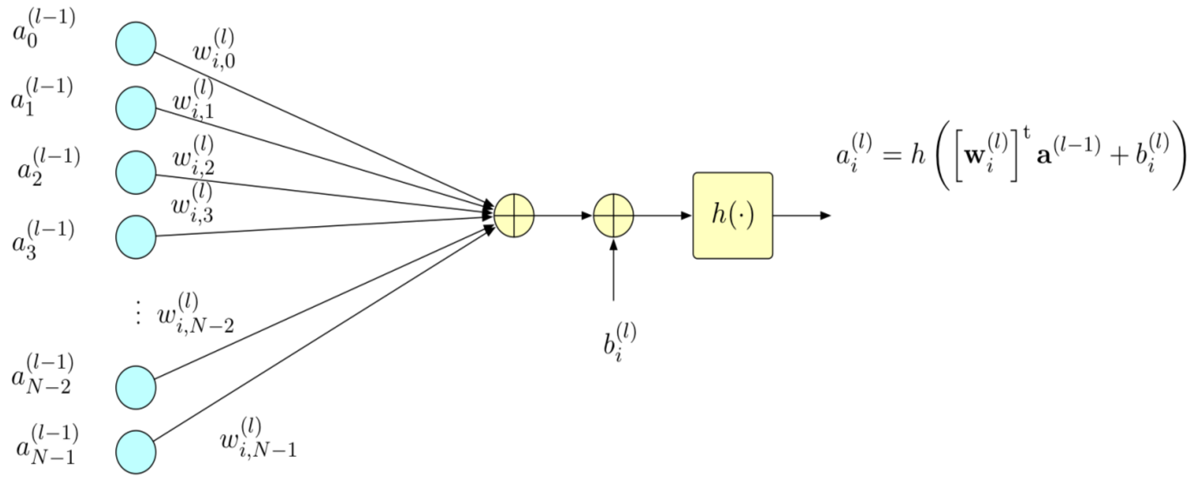

Figure 6: Hidden layer of MLP

1.6 Visualize your MLP: Tensorflow Playground

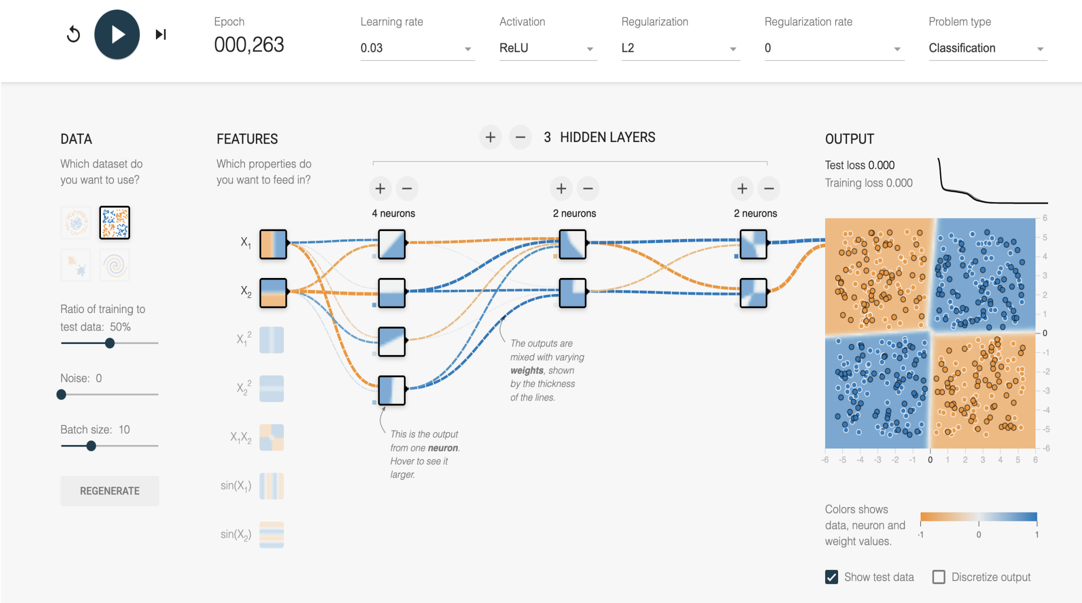

Figure 7: Tensorflow Playground

2 Python Fundamentals

Python

2.1 Data types and control structures

Data types and control structures

2.2 Data Structures

Data Structures

2.3 Objects and classes

Objects and classes

2.4 Python Libraries: NumPy and Matplotlib

Python Libraries: NumPy and Matplotlib

2.5 Reference and Recommended Reading

Reference and Recommended Reading