Preface

Audience

This book was developed based on our teaching of undergraduate and graduate level courses in neural networks and deep learning over the past several years for electrical and computer engineering students. In this book we present the fundamentals of neural networks and their applications in various areas, such as computer vision, natural language processing, reinforcement learning, graphs, and so on. Thus, this book is intended for as a text for the core courses in machine learning taught to majors in electrical engineering, computer engineering, mechanical engineering, manufacturing engineering, as well as science (biology or chemistry) and business. It should also prove useful to engineers and researchers who wish to apply deep learning approaches into their projects.

The purpose of this book is to explain how the major deep learning algorithms and architectures are formalized and developed from mathematical equations. In this book, a balanced coverage of both theory and practice is provided. We cover a wide range of fundamental topics, which include, in an order of from-basic-to-advanced, linear regression, logistic regression, basic neural networks, convolution neural networks (CNNs), object detection, generative adversarial networks (GANs), recurrent neural networks (RNNs), word embeddings, attention mechanism, transformers, tabular reinforcement learning, deep Q-learning, policy gradient reinforcement learning (PG RL), graph neural networks (GNNs). For each topic, we give an intuitive introduction and present the development of the algorithms and architectures from the basic mathematical concepts, using least math skills as possible. To help the reader digest the theorical parts, we provide practical examples with Python programs in most chapters. At the end of each chapter, well-designed problems are provided to help the reader in mastering the subject.

Prerequisites

The prerequisites for studying the material in this book include two aspects: math and program language. In the math aspect, it is assumed that the reader has completed courses in calculus (especially vector calculus), linear algebra and probability theory. In addition, the knowledge of optimization is very helpful. As a review, Chapter 2 provides the basic concepts in these math topics, which may be required for understanding the subsequent chapters. In the program language aspect, we adopt Python language and PyTorch framework because of their abundant packages and open-source accessibility. Although we provide a tutorial on PyTorch in Chapter 7, we expect that many readers may have had an experience with Python.

An Outline for the Reader

The organization of this book is illustrated in Fig.1. **Chapter 1** gives a general introduction to neural networks and deep learning. **Chapter 2** provides a review of mathematical topics, which are required for understanding the subsequent chapters. These topics include linear algebra, matrix

calculus, probability theory and gradient-based optimization. The rest of the book can be divided into four parts: fundamentals of neural networks, computer vision, natural language processing and reinforcement learning.

Part I, *Fundamentals of Neural Networks*, lays a foundation of deep learning, with an emphasis on neural networks. It starts with the simplest model, linear regression, and ends with graph neural networks. In this part, we pay major attention to 1) the mathematical representation of general neural networks and their training (e.g., backpropagation), 2) practical issues in developing neural networks, such as overfitting, regularization, batch schedule, optimization methods, and validation, 3) two important types of neural networks: convolutional neural networks and graph neural networks.

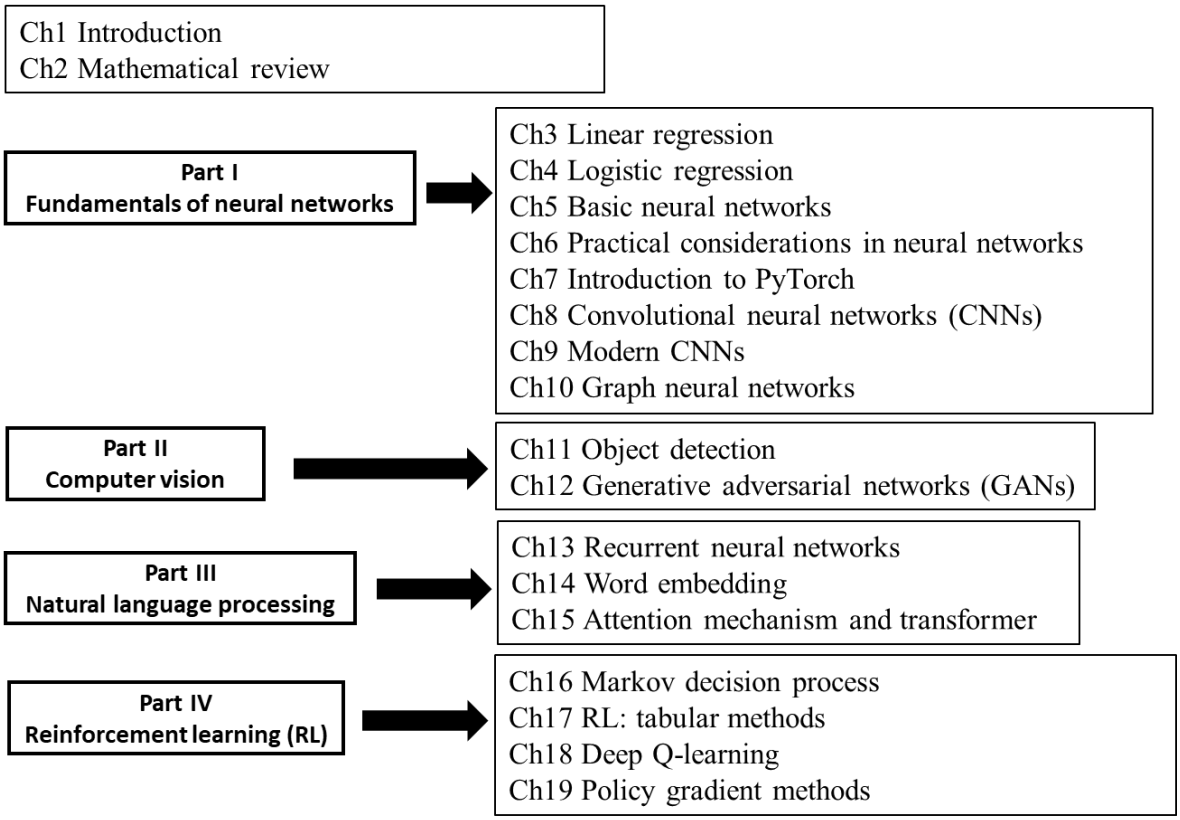

Fig.1 Outline of the book organization

Chapter 3 introduces the basic concepts of machine learning, such as dataset, loss function, training, and gradient descent, through a simple model called linear regression, and describes how these concepts are implemented in mathematical forms and Python programs. Although the practical application of linear regression is not significant, its importance in the theory of statistical learning is noteworthy. Many basic concepts in neural networks and deep learning are generalized from linear regression.

Chapter 4 reveals a transition from regression to classification, i.e., from linear regression to logistic regression, by introducing a non-linear activation function (e.g. sigmoid function) to the output of linear regression. In fact, a neuron (or node) in a neural network is a logistic regression unit. The metrics for classification performance are described.

Chapter 5 provides a thoroughly treatment of general neural networks with a few layers, including mathematical representation (i.e., forward propagation), backward propagation, and parameter updating based on gradient descent. Various activation functions are introduced. The presentation focuses on classification tasks, typically with softmax activation at the output layer and cross-entropy as the loss function.

Chapter 6 treats some common issues in designing and training neural networks. These issues include overfitting, dataset for validation, weight initialization, input pre-processing (e.g. normalization), gradient exploding/vanishing, and convergence of optimization. In this chapter, we provide the mechanisms of these issues and the corresponding treatments.

To implement deep neural networks, a practical and efficient way is to utilize frameworks (e.g., Tensorflow, PyTorch, etc.). **Chapter 7** gives an overview of PyTorch framework and a tutorial on how to implement and train a multi-layer neural network utilizing the facilities of PyTorch framework.

Chapter 8 is devoted entirely to the principle of convolutional neural networks (CNNs). While we present a detailed mathematical definition of convolution layers and pooling layers, we pay a special attention to the intuitive motivation of CNNs and the difference/similarity between CNNs and general neural networks. As an example, LeNet-5 is implemented on the CIFAR-10 dataset through PyTorch.

Chapter 9 introduces the popular modern CNN architectures in deep learning, which are typically used for computer vision tasks. These architectures include AlexNet, VGG network, GoogLeNet, and ResNet. In addition, serval benchmark datasets (e.g., MNIST, CIFAR10, ImageNet, COCO Dataset) are introduced. We also show how to instantiate a pretrained model and then fine-tune it.

Chapter 10 covers the basic concepts and architectures of graph neural networks. The analysis of graph Laplacian matrix and message-passing framework are presented in detail as two important components in the foundation of graph neural networks. We also give an overview of the applications and corresponding architectures of graph neural networks.

Part II, *Computer Vision*, consists of only two chapters, which cover object detection and generative adversarial networks respectively, and lays a foundation of computer vision related applications.

Chapter 11 treats an important computer vision task – object detection. In this chapter, we present the principle of YOLO-based object detection and implementations of pretrained YOLO2 and YOLO3 algorithms. In addition, we describe how to train a YOLO architecture using customized datasets.

Chapter 12 presents the principle, algorithm, and architecture of generative adversarial networks (GANs) for generating fake images. To make it concrete, we give a tutorial based on deep convolutional GANs (DCGAN) to generate fake face images and fake handwritten digits.

Part III, *Natural Language Processing*, including chapters 13-15, provides a foundation of deep learning for sequential (or time-series) data processing. The typical applications include text understanding, speech recognition, and language translation.

Chapter 13 presents various architectures of recurrent neural networks (RNNs) for sequential data processing, backpropagation through time for RNN training, and language models for RNNs. The

architectures presented in this chapter include basic RNNs, Gated recurrent unit (GRU), long short-term memory (LSTM), deep RNNs, and bidirectional RNNs.

Chapter 14 describes the word embedding algorithms used in a powerful word embedding tool – word2vec. These algorithms include continuous bag-of-word model (CBOW) and skip-gram model. Two techniques, hierarchical softmax and negative sampling, are introduced to improve the training efficiency.

Chapter 15 presents a sequence deep learning model, called *transformer*, that adopts the attention mechanism. We also introduce a unified framework of natural language processing – BERT (Bidirectional Encoder Representations from Transformer).

Part IV, *Reinforcement Learning*, covers the fundamentals of reinforcement learning through chapters 16-19.

Chapter 16 describes the basic settings of reinforcement learning, the definition and properties of Markov Decision Process (MDP), value function and policy iteration. It addresses how to search for an optimal policy in an MDP problem by dynamic programming when the environment model is given.

Chapter 17 treats tabular methods in reinforcement learning, which is only practical for the problems with small state-action space. These methods include Monte-Carlo (MC) method, Temporal Difference (TD) method, SARSA algorithm, and Q-learning. This chapter provides a theoretical foundation for chapters 18 and 19 covering deep reinforcement learning with neural networks.

Chapter 18 treats value function approximation in reinforcement learning. State-value function or state-action function is approximately represented by a parameterized function, which can be implemented as neural networks. In this chapter, deep Q-learning is described in detail.

Chapter 19 introduces the policy gradient methods in which the policy functions, instead of value functions, are directly learned by neural networks. The policy gradient theorem is the theoretical foundation for various policy gradient methods and thus addressed in depth. The covered methods include REINFORCE algorithm, Q actor-critic algorithm and advantage actor-critic (A2C) algorithm.