Preface

The field of deep learning is currently going through exponential growth. Because there are currently tens of thousand of technical papers being published each year on deep learning, it would not be possible for a single textbook to do justice to this entire volume of work. Our objective will be to provide an introduction to the key concepts that underlie deep learning. We will cover fundamental deep network architectures, such as multilayer perceptrons, convolution networks, recurrent neural networks and transformers. We will also cover the deep learning software frameworks, such as TensorFlow and PyTorch, that are used to train and implement deep networks. Our goal is to help you learn to use the latest deep networks and implementation tools, but also to understand (as much as we can) how and why they work.

Notation and the Neural Network Design Textbook

In our experience, a key to being able to discuss such a wide-ranging topic is to have a clear and consistent notation. Researchers have come to deep learning from varied backgrounds, which leads to a variety of notation and terminology. Fortunately, we have a good foundation on which to base our notation – our introductory textbook on neural networks [Hagan et al., 2014]. We wrote the first edition of that book in 1994, and the second edition was published in 2014. We will see that many of the fundamental concepts behind deep networks were known in the 1990s and were covered in that book. Here we will use the same notation and terminology that we developed for that text, with appropriate extensions as needed. We will often refer to sections of that book when they are relevant to our deep learning presentation. (That text was not limited to shallow networks.) For convenience, we will use the identifier NND2 to refer to that textbook. It is available online at `https://hagan.okstate.edu/nnd.html`.

Chapter Organization and Github Site

There are two types of chapters in this text. The first type covers theoretical topics. Each of those chapters is divided into the following sections: Objectives, Theory and Examples, Summary of Results, Solved Problems, Epilogue, Further Reading and Exercises. The Theory and Examples section comprises the main body of each chapter. It includes the development of fundamental ideas as well as worked examples (indicated by the icon shown here in the margin). The Summary of Results section provides a convenient listing of important equations and concepts and facilitates the use of the book as an industrial reference. About a third of each chapter is devoted to the Solved Problems section, which provides detailed examples for all key concepts.

The second type of chapter covers deep learning software. Deep learning is not something that can be effectively studied from a solely theoretical viewpoint. The fact is that there are many aspects of deep learning that are not well understood at a theoretical level – we do not understand why they work as well as they do. For this reason, it is important to combine the theory with hands-on experimentation. We have chapters on Python, TensorFlow and PyTorch. Currently, Python is the language in which most deep learning work is being done, and TensorFlow and PyTorch are, by far, the most popular current frameworks for deep learning training and deployment. For each software chapter there is a matching jupyter notebook available on the textbook github site. In this way, you can follow along with the chapter and experiment with all of the coding examples. In addition, there are sets of laboratory jupyter notebooks associated with each of the coding chapters, in which you can dig deeper into the topics.

All of the chapters of this text, as well as all of the laboratories are available on the github site `https://github.com/NNDesignDeepLearning/NNDesignDeepLearning`.

The second way in which we use Python is through the Neural Network Design: Deep Learning Demonstrations, which can be accessed from the website `https://pypi.org/project/nndesigndemos`. These interactive demonstrations illustrate important concepts in each chapter. The icon shown in the margin identifies references to these demonstrations in the text.

Book Organization

Deep neural networks can have several different types of inputs, depending on the application area.

- For *tabular* inputs, each item presented to the network consists of a list of numerical or categorical variables. There is no assumed relationship between the variables, and they can be arranged in any order. The order in which the items are presented to the network does not change the network response to an item.
- For *image* inputs, each item consists of pixels that are arranged in two or three dimensional arrays. The arrangement of the pixels within the arrays is important, and pixels cannot be rearranged. As with tabular inputs, the order of item presentation does not change the network response.
- For *sequential* inputs, the items can be either tabular or image. However, unlike the previous two cases, the order in which the items are presented to the network is important. This can be time series data, text, DNA sequences, audio signals, video, etc. It is assumed that there is a relationship among the items being input to the network, and that the network will use this relationship.

In this text, we will consider all three types of inputs and the networks that are most relevant to each type. We begin with tabular inputs and the multilayer network, which are the focus of Chapters 2-4. (Images can be converted to tabular form, but the result is not the most efficient.) In Chapter 8 we introduce the convolution network, which was especially designed for image processing. In Chapters 11-12 we present dynamic networks, which are designed to operate on sequential inputs.

The chapters that focus on network architectures and training methods are interspersed with chapters that cover the software that we use to implement and train the networks. The most popular deep learning software frameworks are TensorFlow and PyTorch. These are covered in Chapters 6 and 10. They are both accessed using the Python programming language, which we introduce in Chapter 5.

Chapter 7 presents some additional notation that we need to move from multilayer networks to the more sophisticated convolution and dynamic networks. Chapter 9 discusses how we can analyze a trained network to understand how it works. This analysis can be applied to any type of network.