

DEEP LEARNING

JOHN D. KELLEHER

The MIT Press Essential Knowledge Series

A complete list of the titles in this series appears at the back of this book.

The MIT Press | Cambridge, Massachusetts | London, England

CONTENTS

Series Foreword	vii
Preface	ix
Acknowledgments	xi

© 2019 The Massachusetts Institute of Technology

All rights reserved. No part of this book may be reproduced in any form by any electronic or mechanical means (including photocopying, recording, or information storage and retrieval) without permission in writing from the publisher.

This book was set in Chaparral Pro by Toppan Best-set Premedia Limited.
Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Names: Kelleher, John D., 1974- author.

Title: Deep learning / John D. Kelleher.

Description: Cambridge, MA : The MIT Press, [2019] | Series:

The MIT press essential knowledge series | Includes bibliographical references and index.

Identifiers: LCCN 2018059550 | ISBN 9780262537551 (pbk. : alk. paper)

Subjects: LCSH: Machine learning. | Artificial intelligence.

Classification: LCC Q325.5 .K454 2019 | DDC 006.3/1—dc23 LC record available at <https://lcn.loc.gov/2018059550>

1098765

1	Introduction to Deep Learning	1
2	Conceptual Foundations	39
3	Neural Networks: The Building Blocks of Deep Learning	65
4	A Brief History of Deep Learning	101
5	Convolutional and Recurrent Neural Networks	159
6	Learning Functions	185
7	The Future of Deep Learning	231

Glossary	251
Notes	257
References	261
Further Readings	267
Index	269

the network using large datasets. By contrast, hyperparameters are the parameters of a model (in these cases, the parameters of a neural network architecture) and/or training algorithm that cannot be directly estimated from the data but instead must be specified by the person creating the model, either through the use of heuristic rules, intuition, or trial and error. Often, much of the effort that goes into the creation of a deep learning network involves experimental work to answer the questions in relation to hyperparameters, and this process is known as hyperparameter tuning. The next chapter will review the history and evolution of deep learning, and the challenges posed by many of these questions are themes running through the review. Subsequent chapters in the book will explore how answering these questions in different ways can create networks with very different characteristics, each suited to different types of tasks. For example, recurrent neural networks are best suited to processing sequential/time-series data, whereas convolutional neural networks were originally developed to process images. Both of these network types are, however, built using the same fundamental processing unit, the artificial neuron; the differences in the behavior and abilities of these networks stems from how these neurons are arranged and composed.

A BRIEF HISTORY OF DEEP LEARNING

The history of deep learning can be described as three major periods of excitement and innovation, interspersed with periods of disillusionment. Figure 4.1 shows a timeline of this history, which highlights these periods of major research: on threshold logic units (early 1940s to the mid 1960s), connectionism (early 1980s to mid-1990s), and deep learning (mid 2000s to the present). Figure 4.1 distinguishes some of the primary characteristics of the networks developed in each of these three periods. The changes in these network characteristics highlight some of the major themes within the evolution of deep learning, including: the shift from binary to continuous values; the move from threshold activation functions, to logistic and tanh activation, and then onto ReLU activation; and the progressive deepening of the networks, from single layer,

we will describe it in chapter 5. The fact that LSTM can propagate activations over long periods enables them to process sequences that include long-distance dependencies (interactions between elements in a sequence that are separated by two or more positions). For example, the dependency between the subject and the verb in an English sentence: *The **dog/dogs** in that house **is/are** aggressive*. This has made LSTM networks suitable for language processing, and for a number of years they have been the default neural network architecture for many natural language processing models, including machine translation. For example, the sequence-to-sequence (seq2seq) machine translation architecture introduced in 2014 connects two LSTM networks in sequence (Sutskever et al. 2014). The first LSTM network, the encoder, processes the input sequence one input at a time, and generates a distributed representation of that input. The first LSTM network is called an encoder because it encodes the sequence of words into a distributed representation. The second LSTM network, the decoder, is initialized with the distributed representation of the input and is trained to generate the output sequence one element at a time using a feedback loop that feeds the most recent output element generated by the network back in as the input for the next time step. Today, this seq2seq architecture is the basis for most modern machine translation systems, and is explained in more detail in chapter 5.

By the late 1990s, most of the conceptual requirements for deep learning were in place, including both the algorithms to train networks with multiple layers, and the network architectures that are still very popular today (CNNs and RNNs). However, the problem of the vanishing gradients still stifled the creation of deep networks. Also, from a commercial perspective, the 1990s (similar to the 1960s) experienced a wave of hype based on neural networks and unrealized promises. At the same time, a number of breakthroughs in other forms of machine learning models, such as the development of support vector machines (SVMs), redirected the focus of the machine learning research community away from neural networks: at the time SVMs were achieving similar accuracy to neural network models but were easier to train. Together these factors led to a decline in neural network research that lasted up until the emergence of deep learning.

The Era of Deep Learning

The first recorded use of the term deep learning is credited to Rina Dechter (1986), although in Dechter's paper the term was not used in relation to neural networks; and the first use of the term in relation to neural networks is credited to Aizenberg et al. (2000).⁶ In the mid-2000s, interest in neural networks started to grow, and it was around

this time that the term deep learning came to prominence to describe deep neural networks. The term deep learning is used to emphasize the fact that the networks being trained are much deeper than previous networks.

One of the early successes of this new era of neural network research was when Geoffrey Hinton and his colleagues demonstrated that it was possible to train a deep neural network using a process known as greedy layer-wise pretraining. Greedy layer-wise pretraining begins by training a single layer of neurons that receives input directly from the raw input. There are a number of different ways that this single layer of neurons can be trained, but one popular way is to use an autoencoder. An autoencoder is a neural network with three layers: an input layer, a hidden (encoding) layer, and an output (decoding) layer. The network is trained to reconstruct the inputs it receives in the output layer; in other words, the network is trained to output the exact same values that it received as input. A very important feature in these networks is that they are designed so that it is not possible for the network to simply copy the inputs to the outputs. For example, an autoencoder may have fewer neurons in the hidden layer than in the input and output layer. Because the autoencoder is trying to reconstruct the input at the output layer, the fact that the information from the input must pass through this bottleneck in the hidden layer forces the autoencoder to learn an encoding of the input data in the

hidden layer that captures only the most important features in the input, and disregards redundant or superfluous information.⁷

Layer-Wise Pretraining Using Autoencoders

In layer-wise pretraining, the initial autoencoder learns an encoding for the raw inputs to the network. Once this encoding has been learned, the units in the hidden encoding layer are fixed, and the output (decoding) layer is thrown away. Then a second autoencoder is trained—but this autoencoder is trained to reconstruct the representation of the data generated by passing it through the encoding layer of the initial autoencoder. In effect, this second autoencoder is stacked on top of the encoding layer of the first autoencoder. This stacking of encoding layers is considered to be a greedy process because each encoding layer is optimized independently of the later layers; in other words, each autoencoder focuses on finding the best solution for its immediate task (learning a useful encoding for the data it must reconstruct) rather than trying to find a solution to the overall problem for the network.

Once a sufficient number⁸ of encoding layers have been trained, a tuning phase can be applied. In the tuning phase, a final network layer is trained to predict the target output for the network. Unlike the pretraining of the earlier layers of the network, the target output for the final layer is different from the input vector and is specified