

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

INTRODUCTION TO DEEP LEARNING

Deep learning is the subfield of artificial intelligence that focuses on creating large neural network models that are capable of making accurate *data-driven decisions*. Deep learning is particularly suited to contexts where the data is complex and where there are large datasets available. Today most online companies and high-end consumer technologies use deep learning. Among other things, Facebook uses deep learning to analyze text in online conversations. Google, Baidu, and Microsoft all use deep learning for image search, and also for machine translation. All modern smart phones have deep learning systems running on them; for example, deep learning is now the standard technology for speech recognition, and also for face detection on digital cameras. In the healthcare sector, deep learning is used to process medical images (X-rays, CT, and MRI scans) and diagnose health conditions. Deep learning is also at the core of self-driving cars, where it is used for

belonging to more than four thousand identities (Taigman et al. 2014).

The Key Ingredients of Machine Learning

The above example of deciding which arithmetic operation best explains the relationship between inputs and outputs in a set of data illustrates the three key ingredients in machine learning:

1. Data (a set of historical examples).
2. A set of functions that the algorithm will search through to find the best match with the data.
3. Some measure of fitness that can be used to evaluate how well each candidate function matches the data.

All three of these ingredients must be correct if a machine learning project is to succeed; below we describe each of these ingredients in more detail.

We have already introduced the concept of a dataset as a two-dimensional table (or $n \times m$ matrix),⁵ where each row contains the information for one example, and each column contains the information for one of the features in the domain. For example, table 1.2 illustrates how the sample inputs and outputs of the first unknown arithmetic

function problem in the chapter can be represented as a dataset. This dataset contains four examples (also known as instances), and each example is represented using two input features and one output (or target) feature. Designing and selecting the features to represent the examples is a very important step in any machine learning project.

As is so often the case in computer science, and machine learning, there is a tradeoff in feature selection. If we choose to include only a minimal number of features in the dataset, then it is likely that a very informative feature will be excluded from the data, and the function returned by the machine learning algorithm will not work well. Conversely, if we choose to include as many features as possible in the domain, then it is likely that irrelevant or redundant features will be included, and this will also likely result in the function not working well. One reason for this is that the more redundant or irrelevant features that are included, the greater the probability for the machine learning algorithm to extract patterns that are based on spurious correlations between these features. In these cases, the algorithm gets confused between the real patterns in the data and the spurious patterns that only appear in the data due to the particular sample of examples that have been included in the dataset.

Finding the correct set of features to include in a dataset involves engaging with experts who understand

the domain, using statistical analysis of the distribution of individual features and also the correlations between pairs of features, and a trial-and-error process of building models and checking the performance of the models when particular features are included or excluded. This process of dataset design is a labor-intensive task that often takes up a significant portion of the time and effort expended on a machine learning project. It is, however, a critical task if the project is to succeed. Indeed, identifying which features are informative for a given task is frequently where the real value of machine learning projects emerge.

The second ingredient in a machine learning project is the set of candidate functions that the algorithm will consider as the potential explanation of the patterns in the data. In the unknown arithmetic function scenario previously given, the set of considered functions was explicitly specified and restricted to four: *addition*, *subtraction*, *multiplication*, or *division*. More generally, the set of functions is implicitly defined through the inductive bias of the

Table 1.2. A simple tabular dataset

Input 1	Input 2	Target
5	5	25
2	6	12
4	4	16
2	2	04

machine learning algorithm and the function representation (or model) that is being used. For example, a neural network model is a very flexible function representation.

The third and final ingredient to machine learning is the measure of fitness. The measure of fitness is a function that takes the outputs from a candidate function, generated when the machine learning algorithm applies the candidate function to the data, and compares these outputs with the data, in some way. The result of this comparison is a value that describes the fitness of the candidate function relative to the data. A fitness function that would work for our unknown arithmetic function scenario is to count in how many of the examples a candidate function returns a value that exactly matches the target specified in the data. Multiplication would score four out of four on this fitness measure, addition would score one out of four, and division and subtraction would both score zero out of four. There are a large variety of fitness functions that can be used in machine learning, and the selection of the correct fitness function is crucial to the success of a machine learning project. The design of new fitness functions is a rich area of research in machine learning. Varying how the dataset is represented, and how the candidate functions and the fitness function are defined, results in three different categories of machine learning: supervised, unsupervised, and reinforcement learning.

Supervised, Unsupervised, and Reinforcement Learning

Supervised machine learning is the most common type of machine learning. In supervised machine learning, each example in the dataset is labeled with the expected output (or target) value. For example, if we were using the dataset in table 1.1 to learn a function that maps from the inputs of annual income and debt to a credit solvency score, the credit solvency feature in the dataset would be the target feature. In order to use supervised machine learning, our dataset must list the value of the target feature for every example in the dataset. These target feature values can sometimes be very difficult, and expensive, to collect. In some cases, we must pay human experts to label each example in a dataset with the correct target value. However, the benefit of having these target values in the dataset is that the machine learning algorithm can use these values to help the learning process. It does this by comparing the outputs a function produces with the target outputs specified in the dataset, and using the difference (or error) to evaluate the fitness of the candidate function, and use the fitness evaluation to guide the search for the best function. It is because of this feedback from the target labels in the dataset to the algorithm that this type of machine learning is considered supervised. This is the type of machine learning that was demonstrated by the example of

choosing between different arithmetic functions to explain the behavior of an unknown function.

Unsupervised machine learning is generally used for clustering data. For example, this type of data analysis is useful for customer segmentation, where a company wishes to segment its customer base into coherent groups so that it can target marketing campaigns and/or product designs to each group. In unsupervised machine learning, there are no target values in the dataset. Consequently, the algorithm cannot directly evaluate the fitness of a candidate function against the target values in the dataset. Instead, the machine learning algorithm tries to identify functions that map similar examples into clusters, such that the examples in a cluster are more similar to the other examples in the same cluster than they are to examples in other clusters. Note that the clusters are not prespecified, or at most they are initially very underspecified. For example, the data scientist might provide the algorithm with a target number of clusters, based on some intuition about the domain, without providing explicit information on relative sizes of the clusters or regarding the characteristics of examples that belong in each cluster. Unsupervised machine learning algorithms often begin by guessing an initial clustering of the examples and then iteratively adjusting the clusters (by dropping instances from one cluster and adding them to another) so as to improve the fitness of the cluster set. The fitness functions used in

unsupervised machine learning generally reward candidate functions that result in higher similarity within individual clusters and, also, high diversity between clusters.

Reinforcement learning is most relevant for online control tasks, such as robot control and game playing. In these scenarios, an agent needs to learn a policy for how it should act in an environment in order to be rewarded. In reinforcement learning, the goal of the agent is to learn a mapping from its current observation of the environment and its own internal state (its memory) to what action it should take: for instance, *should the robot move forward or backward or should the computer program move the pawn or take the queen*. The output of this policy (function) is the action that the agent should take next, given the current context. In these types of scenarios, it is difficult to create historic datasets, and so reinforcement learning is often carried out *in situ*: an agent is released into an environment where it experiments with different policies (starting with a potentially random policy) and over time updates its policy in response to the rewards it receives from the environment. If an action results in a positive reward, the mapping from the relevant observations and state to that action is reinforced in the policy, whereas if an action results in a negative reward, the mapping is weakened. Unlike in supervised and unsupervised machine learning, in reinforcement learning, the fact that learning is done *in situ* means that the training and

inference stages are interleaved and ongoing. The agent infers what action it should do next and uses the feedback from the environment to learn how to update its policy. A distinctive aspect of reinforcement learning is that the target output of the learned function (the agent's actions) is decoupled from the reward mechanism. The reward may be dependent on multiple actions and there may be no reward feedback, either positive or negative, available directly after an action has been performed. For example, in a chess scenario, the reward may be +1 if the agent wins the game and -1 if the agent loses. However, this reward feedback will not be available until the last move of the game has been completed. So, one of the challenges in reinforcement learning is designing training mechanisms that can distribute the reward appropriately back through a sequence of actions so that the policy can be updated appropriately. Google's DeepMind Technologies generated a lot of interest by demonstrating how reinforcement learning could be used to train a deep learning model to learn control policies for seven different Atari computer games (Mnih et al. 2013). The input to the system was the raw pixel values from the screen, and the control policies specified what joystick action the agent should take at each point in the game. Computer game environments are particularly suited to reinforcement learning as the agent can be allowed to play many thousands of games against the computer game system in order to learn a successful

policy, without incurring the cost of creating and labeling a large dataset of example situations with correct joystick actions. The DeepMind system got so good at the games that it outperformed all previous computer systems on six of the seven games, and outperformed human experts on three of the games.

Deep learning can be applied to all three machine learning scenarios: supervised, unsupervised, and reinforcement. Supervised machine learning is, however, the most common type of machine learning. Consequently, the majority of this book will focus on deep learning in a supervised learning context. However, most of the deep learning concerns and principles introduced in the supervised learning context also apply to unsupervised and reinforcement learning.

Why Is Deep Learning So Successful?

In any data-driven process the primary determinant of success is knowing what to measure and how to measure it. This is why the processes of feature selection and feature design are so important to machine learning. As discussed above, these tasks can require domain expertise, statistical analysis of the data, and iterations of experiments building models with different feature sets. Consequently, dataset design and preparation can consume a significant

In any data-driven
process the primary
determinant of success
is knowing what to
measure and how to
measure it.