1. Record Nr. UNINA9910522921003321

Autore	Long Liangqu
Titolo	Beginning deep learning with TensorFlow : work with Keras, MNIST data sets, and advanced neural networks / / Liangqu Long, Xiangming Zeng
Pubbl/distr/stampa	New York, New York : , : Apress L. P., , [2022] ©2022
ISBN	1-5231-5104-8 1-4842-7915-8
Descrizione fisica	1 online resource (727 pages)
Disciplina	006.31
Soggetti	Machine learning
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Intro -- Table of Contents -- About the Authors -- About the Technical Reviewer -- Acknowledgments -- Chapter 1: Introduction to Artificial Intelligence -- 1.1 Artificial Intelligence in Action -- 1.1.1 Artificial Intelligence Explained -- 1.1.2 Machine Learning -- 1.1.3 Neural Networks and Deep Learning -- 1.2 The History of Neural Networks -- 1.2.1 Shallow Neural Networks -- 1.2.2 Deep Learning -- 1.3 Deep Learning Characteristics -- 1.3.1 Data Volume -- 1.3.2 Computing Power -- 1.3.3 Network Scale -- 1.3.4 General Intelligence -- 1.4 Deep Learning Applications -- 1.4.1 Computer Vision -- 1.4.2 Natural Language Processing -- 1.4.3 Reinforcement Learning -- 1.5 Deep Learning Framework -- 1.5.1 Major Frameworks -- 1.5.2 TensorFlow 2 and 1.x -- 1.5.3 Demo -- 1.6 Development Environment Installation -- 1.6.1 Anaconda Installation -- 1.6.2 CUDA Installation -- 1.6.3 TensorFlow Installation -- 1.6.4 Common Editor Installation -- 1.7 Summary -- 1.8 Reference -- Chapter 2: Regression -- 2.1 Neuron Model -- 2.2 Optimization Method -- 2.3 Linear Model in Action -- 2.4 Summary -- 2.5 References -- Chapter 3: Classification -- 3.1 Handwritten Digital Picture Dataset -- 3.2 Build a Model -- 3.3 Error Calculation -- 3.4 Do We Really Solve the Problem? -- 3.5 Nonlinear Model -- 3.6 Model Complexity -- 3.7 Optimization Method -- 3.8 Hands-On Handwritten Digital Image Recognition -- 3.8.1 Build the Network -- 3.8.2 Model Training -- 3.9 Summary -- 3.10

Reference -- Chapter 4: Basic TensorFlow -- 4.1 Data Types -- 4.1.1 Numeric -- 4.1.2 String -- 4.1.3 Boolean -- 4.2 Numerical Precision -- 4.3 Tensors to Be Optimized -- 4.4 Create Tensors -- 4.4.1 Create Tensors from Arrays and Lists -- 4.4.2 Create All-0 or All-1 Tensors -- 4.4.3 Create a Customized Numeric Tensor -- 4.4.4 Create a Tensor from a Known Distribution -- 4.4.5 Create a Sequence.
4.5 Typical Applications of Tensors -- 4.5.1 Scalar -- 4.5.2 Vector -- 4.5.3 Matrix -- 4.5.4 Three-Dimensional Tensor -- 4.5.5 Four-Dimensional Tensor -- 4.6 Indexing and Slicing -- 4.6.1 Indexing -- 4.6.2 Slicing -- 4.6.3 Slicing Summary -- 4.7 Dimensional Transformation -- 4.7.1 Reshape -- 4.7.2 Add and Delete Dimensions -- 4.7.3 Swap Dimensions -- 4.7.4 Copy Data -- 4.8 Broadcasting -- 4.9 Mathematical Operations -- 4.9.1 Addition, Subtraction, Multiplication and Division -- 4.9.2 Power Operations -- 4.9.3 Exponential and Logarithmic Operations -- 4.9.4 Matrix Multiplication -- 4.10 Hands-On Forward Propagation -- Chapter 5: Advanced TensorFlow -- 5.1 Merge and Split -- 5.1.1 Merge -- 5.1.2 Split -- 5.2 Common Statistics -- 5.2.1 Norm -- 5.2.2 Max, Min, Mean, and Sum -- 5.3 Tensor Comparison -- 5.4 Fill and Copy -- 5.4.1 Fill -- 5.4.2 Copy -- 5.5 Data Limiting -- 5.6 Advanced Operations -- 5.6.1 tf.gather -- 5.6.2 tf.gather_nd -- 5.6.3 tf.boolean_mask -- 5.6.4 tf.where -- 5.6.5 tf.scatter_nd -- 5.6.6 tf.meshgrid -- 5.7 Load Classic Datasets -- 5.7.1 Shuffling -- 5.7.2 Batch Training -- 5.7.3 Preprocessing -- 5.7.4 Epoch Training -- 5.8 Hands-On MNIST Dataset -- Chapter 6: Neural Networks -- 6.1 Perceptron -- 6.2 Fully Connected Layer -- 6.2.1 Tensor Mode Implementation -- 6.2.2 Layer Implementation -- 6.3 Neural Network -- 6.3.1 Tensor Mode Implementation -- 6.3.2 Layer Mode Implementation -- 6.3.3 Optimization -- 6.4 Activation function -- 6.4.1 Sigmoid -- 6.4.2 ReLU -- 6.4.3 LeakyReLU -- 6.4.4 Tanh -- 6.5 Design of Output Layer -- 6.5.1 Common Real Number Space -- 6.5.2 [0, 1] Interval -- 6.5.3 [0,1] Interval with Sum 1 -- 6.5.4 (-1, 1) Interval -- 6.6 Error Calculation -- 6.6.1 Mean Square Error Function -- 6.6.2 Cross-Entropy Error Function -- 6.7 Types of Neural Networks -- 6.7.1 Convolutional Neural Network -- 6.7.2 Recurrent Neural Network.
6.7.3 Attention Mechanism Network -- 6.7.4 Graph Convolutional Neural Network -- 6.8 Hands-On of Automobile Fuel Consumption Prediction -- 6.8.1 Dataset -- 6.8.2 Create a Network -- 6.8.3 Training and Testing -- 6.9 References -- Chapter 7: Backward Propagation Algorithm -- 7.1 Derivatives and Gradients -- 7.2 Common Properties of Derivatives -- 7.2.1 Common Derivatives -- 7.2.2 Common Property of Derivatives -- 7.2.3 Hands-On Derivative Finding -- 7.3 Derivative of Activation Function -- 7.3.1 Derivative of Sigmoid Function -- 7.3.2 Derivative of ReLU Function -- 7.3.3 Derivative of LeakyReLU Function -- 7.3.4 Derivative of Tanh Function -- 7.4 Gradient of Loss Function -- 7.4.1 Gradient of Mean Square Error Function -- 7.4.2 Gradient of Cross-Entropy Function -- 7.5 Gradient of Fully Connected Layer -- 7.5.1 Gradient of a Single Neuron -- 7.5.2 Gradient of Fully Connected Layer -- 7.6 Chain Rule -- 7.7 Back Propagation Algorithm -- 7.8 Hands-On Optimization of Himmelblau -- 7.9 Hands-On Back Propagation Algorithm -- 7.9.1 Dataset -- 7.9.2 Network Layer -- 7.9.3 Network model -- 7.9.4 Network Training -- 7.9.5 Network Performance -- 7.10 References -- Chapter 8: Keras Advanced API -- 8.1 Common Functional Modules -- 8.1.1 Common Network Layer Classes -- 8.1.2 Network Container -- 8.2 Model Configuration, Training, and Testing -- 8.2.1 Model Configuration -- 8.2.2 Model Training -- 8.2.3 Model Testing -- 8.3 Model Saving and Loading -- 8.3.1 Tensor Method -- 8.3.2 Network Method -- 8.3.3 SavedModel method -- 8.4 Custom Network -- 8.4.1 Custom Network Layer --

8.4.2 Customized Network -- 8.5 Model Zoo -- 8.5.1 Load Model -- 8.6 Metrics -- 8.6.1 Create a Metrics Container -- 8.6.2 Write Data -- 8.6.3 Read Statistical Data -- 8.6.4 Clear the Container -- 8.6.5 Hands-On Accuracy Metric -- 8.7 Visualization -- 8.7.1 Model Side -- 8.7.2 Browser Side.
8.8 Summary -- Chapter 9: Overfitting -- 9.1 Model Capacity -- 9.2 Overfitting and Underfitting -- 9.2.1 Underfitting -- 9.2.2 Overfitting -- 9.3 Dataset Division -- 9.3.1 Validation Set and Hyperparameters -- 9.3.2 Early Stopping -- 9.4 Model Design -- 9.5 Regularization -- 9.5.1 L0 Regularization -- 9.5.2 L1 Regularization -- 9.5.3 L2 Regularization -- 9.5.4 Regularization Effect -- 9.6 Dropout -- 9.7 Data Augmentation -- 9.7.1 Rotation -- 9.7.2 Flip -- 9.7.3 Cropping -- 9.7.4 Generate Data -- 9.7.5 Other Methods -- 9.8 Hands-On Overfitting -- 9.8.1 Build the Dataset -- 9.8.2 Influence of the Number of Network Layers -- 9.8.3 Impact of Dropout -- 9.8.4 Impact of Regularization -- 9.9 References -- Chapter 10: Convolutional Neural Networks -- 10.1 Problems with Fully Connected N -- 10.1.1 Local Correlation -- 10.1.2 Weight Sharing -- 10.1.3 Convolution Operation -- 10.2 Convolutional Neural Network -- 10.2.1 Single-Channel Input and Single Convolution Kernel -- 10.2.2 Multi-channel Input and Single Convolution Kernel -- 10.2.3 Multi-channel Input and Multi-convolution Kernel -- 10.2.4 Stride Size -- 10.2.5 Padding -- 10.3 Convolutional Layer Implementation -- 10.3.1 Custom Weights -- 10.3.2 Convolutional Layer Classes -- 10.4 Hands-On LeNet-5 -- 10.5 Representation Learning -- 10.6 Gradient Propagation -- 10.7 Pooling Layer -- 10.8 BatchNorm Layer -- 10.8.1 Forward Propagation -- 10.8.2 Backward Propagation -- 10.8.3 Implementation of BatchNormalization layer -- 10.9 Classical Convolutional Network -- 10.9.1 AlexNet -- 10.9.2 VGG Series -- 10.9.3 GoogLeNet -- 10.10 Hands-On CIFAR10 and VGG13 -- 10.11 Convolutional Layer Variants -- 10.11.1 Dilated/Atrous Convolution -- 10.11.2 Transposed Convolution -- o + 2p k = n * s -- o + 2p k n * s -- Matrix Transposition -- Transposed Convolution Implementation -- 10.11.3 Separate Convolution.
10.12 Deep Residual Network -- 10.12.1 ResNet Principle -- 10.12.2 ResBlock Implementation -- 10.13 DenseNet -- 10.14 Hands-On CIFAR10 and ResNet18 -- 10.15 References -- Chapter 11: Recurrent Neural Network -- 11.1 Sequence Representation Method -- 11.1.1 Embedding Layer -- 11.1.2 Pre-trained Word Vectors -- 11.2 Recurrent Neural Network -- 11.2.1 Is a Fully Connected Layer Feasible? -- 11.2.2 Shared Weight -- 11.2.3 Global Semantics -- 11.2.4 Recurrent Neural Network -- 11.3 Gradient Propagation -- 11.4 How to Use RNN Layers -- 11.4.1 SimpleRNNCell -- 11.4.2 Multilayer SimpleRNNCell Network -- 11.4.3 SimpleRNN Layer -- 11.5 Hands-On RNN Sentiment Classification -- 11.5.1 Dataset -- 11.5.2 Network Model -- 11.5.3 Training and Testing -- 11.6 Gradient Vanishing and Gradient Exploding -- 11.6.1 Gradient Clipping -- 11.6.2 Gradient Vanishing -- 11.7 RNN Short-Term Memory -- 11.8 LSTM Principle -- 11.8.1 Forget Gate -- 11.8.2 Input Gate -- 11.8.3 Update Memory -- 11.8.4 Output Gate -- 11.8.5 Summary -- 11.9 How to Use the LSTM Layer -- 11.9.1 LSTMCell -- 11.9.2 LSTM layer -- 11.10 GRU Introduction -- 11.10.1 Reset Door -- 11.10.2 Update Gate -- 11.10.3 How to Use GRU -- 11.11 Hands-On LSTM/GRU Sentiment Classification -- 11.11.1 LSTM Model -- 11.11.2 GRU model -- 11.12 Pre-trained Word Vectors -- 11.13 Pre-trained Word Vectors -- 11.14 References -- Chapter 12: Autoencoder -- 12.1 Principle of Autoencoder -- 12.2 Hands-On Fashion MNIST Image Reconstruction -- 12.2.1 Fashion MNIST Dataset -- 12.2.2 Encoder -- 12.2.3 Decoder -- 12.2.4 Autoencoder -- 12.2.5

Network Training -- 12.2.6 Image Reconstruction -- 12.3 Autoencoder Variants -- 12.3.1 Dropout Autoencoder -- 12.3.2 Adversarial Autoencoder -- 12.4 Variational Autoencoder -- 12.4.1 Principle of VAE -- 12.4.2 Reparameterization Trick -- 12.5 Hands-On VAE Image Reconstruction -- 12.5.1 VAE model.
12.5.2 Reparameterization Trick.

Sommario/riassunto

Incorporate deep learning into your development projects through hands-on coding and the latest versions of deep learning software, such as TensorFlow 2 and Keras. The materials used in this book are based on years of successful online education experience and feedback from thousands of online learners. You'll start with an introduction to AI, where you'll learn the history of neural networks and what sets deep learning apart from other varieties of machine learning. Discovery the variety of deep learning frameworks and set-up a deep learning development environment. Next, you'll jump into simple classification programs for hand-writing analysis. Once you've tackled the basics of deep learning, you move on to TensorFlow 2 specifically. Find out what exactly a Tensor is and how to work with MNIST datasets. Finally, you'll get into the heavy lifting of programming neural networks and working with a wide variety of neural network types such as GANs and RNNs. Deep Learning is a new area of Machine Learning research widely used in popular applications, such as voice assistant and self-driving cars. Work through the hands-on material in this book and become a TensorFlow programmer! You will: Develop using deep learning algorithms Build deep learning models using TensorFlow 2 Create classification systems and other, practical deep learning applications.