

List of Listings

Chapter 2. TensorFlow essentials

[Listing 2.1. Computing the inner product of two vectors without using a library](#)

[Listing 2.2. Computing the inner product using NumPy](#)

[Listing 2.3. Different ways to represent tensors](#)

[Listing 2.4. Creating tensors](#)

[Listing 2.5. Using the negation operator](#)

[Listing 2.6. Using a session](#)

[Listing 2.7. Using the interactive session mode](#)

[Listing 2.8. Logging a session](#)

[Listing 2.9. Using a variable](#)

[Listing 2.10. Saving variables](#)

[Listing 2.11. Loading variables](#)

[Listing 2.12. Defining the average update operator](#)

[Listing 2.13. Running iterations of the exponential average algorithm](#)

[Listing 2.14. Filling in missing code to complete the exponential average algorithm](#)

[Listing 2.15. Annotating with a summary op](#)

[Listing 2.16. Writing summaries to view in TensorBoard](#)

Chapter 3. Linear regression and beyond

[Listing 3.1. Visualizing raw input](#)

[Listing 3.2. Solving linear regression](#)

[Listing 3.3. cost functionUsing a polynomial model](#)

[Listing 3.4. Splitting the dataset into testing and training sets](#)

[Listing 3.5. Evaluating regularization parameters](#)

[Listing 3.6. Parsing raw CSV datasets](#)

Chapter 4. A gentle introduction to classification

[Listing 4.1. Using linear regression for classification](#)

[Listing 4.2. Executing the graph](#)

[Listing 4.3. Measuring accuracy](#)

[Listing 4.4. Linear regression failing miserably for classification](#)

[Listing 4.5. Using one-dimensional logistic regression](#)

[Listing 4.6. Setting up data for two-dimensional logistic regression](#)

[Listing 4.7. Using TensorFlow for multidimensional logistic regression](#)

[Listing 4.8. Visualizing multiclass data](#)

[Listing 4.9. Setting up training and test data for multiclass classification](#)

[Listing 4.10. Using softmax regression](#)

[Listing 4.11. Executing the graph](#)

Chapter 5. Automatically clustering data

[Listing 5.1. Traversing a directory for data](#)

[Listing 5.2. Representing audio in Python](#)

[Listing 5.3. Obtaining a dataset for k-means](#)

[Listing 5.4. Implementing k-means](#)

[Listing 5.5. Organizing data for segmentation](#)

[Listing 5.6. Segmenting an audio clip](#)

[Listing 5.7. Setting up the SOM algorithm](#)

[Listing 5.8. Defining how to update the values of neighbors](#)

[Listing 5.9. Getting the node location of the closest match](#)

[Listing 5.10. Generating a matrix of points](#)

[Listing 5.11. Running the SOM algorithm](#)

[Listing 5.12. Testing the implementation and visualizing the results](#)

Chapter 6. Hidden Markov models

[Listing 6.1. Defining the HMM class](#)

[Listing 6.2. Creating a helper function to access emission probability of an observation](#)

[Listing 6.3. Initializing the cache](#)

[Listing 6.4. Updating the cache](#)

[Listing 6.5. Defining the forward algorithm given an HMM](#)

[Listing 6.6. Defining the HMM and calling the forward algorithm](#)

[Listing 6.7. Adding the Viterbi cache as a member variable](#)

[Listing 6.8. Defining an op to update the forward cache](#)

[Listing 6.9. Defining an op to update the back pointers](#)

[Listing 6.10. Defining the Viterbi decoding algorithm](#)

[Listing 6.11. Running the Viterbi decode](#)

Chapter 7. A peek into autoencoders

[Listing 7.1. Python class schema](#)

[Listing 7.2. Using name scopes](#)

[Listing 7.3. Autoencoder class](#)

[Listing 7.4. Training the autoencoder](#)

[Listing 7.5. Testing the model on data](#)

[Listing 7.6. Using your Autoencoder class](#)

[Listing 7.7. Batch helper function](#)

[Listing 7.8. Batch learning](#)

[Listing 7.9. Loading images](#)

[Listing 7.10. Reading from the extracted CIFAR-10 dataset](#)

[Listing 7.11. Reading all CIFAR-10 files to memory](#)

[Listing 7.12. Converting CIFAR-10 image to grayscale](#)

[Listing 7.13. Setting up the autoencoder](#)

Chapter 8. Reinforcement learning

[Listing 8.1. Importing relevant libraries](#)

[Listing 8.2. Helper function to get prices](#)

[Listing 8.3. Helper function to plot the stock prices](#)

[Listing 8.4. Get data and visualize it](#)

[Listing 8.5. Defining a superclass for all decision policies](#)

[Listing 8.6. Implementing a random decision policy](#)

[Listing 8.7. Using a given policy to make decisions, and returning the performance](#)

[Listing 8.8. Running multiple simulations to calculate an average performance](#)

[Listing 8.9. Defining the decision policy](#)

[Listing 8.10. Implementing a more intelligent decision policy](#)

Chapter 9. Convolutional neural networks

[Listing 9.1. Loading images from a CIFAR-10 file in Python](#)

[Listing 9.2. Cleaning data](#)

[Listing 9.3. Preprocessing all CIFAR-10 files](#)

[Listing 9.4. Using the cifar_tools helper function](#)

[Listing 9.5. Visualizing images from the dataset](#)

[Listing 9.6. Generating and visualizing random filters](#)

[Listing 9.7. Using a session to initialize weights](#)

[Listing 9.8. Showing convolution results](#)

[Listing 9.9. Visualizing convolutions](#)

[Listing 9.10. Running the maxpool function to subsample convolved images](#)

[Listing 9.11. Setting up CNN weights](#)

[Listing 9.12. Creating a convolution layer](#)

[Listing 9.13. Creating a max-pool layer](#)

[Listing 9.14. The full CNN model](#)

[Listing 9.15. Defining ops to measure the cost and accuracy](#)

[Listing 9.16. Training the neural network by using the CIFAR-10 dataset](#)

Chapter 10. Recurrent neural networks

[Listing 10.1. Importing relevant libraries](#)

[Listing 10.2. Defining a class and its constructor](#)

[Listing 10.3. Defining the RNN model](#)

[Listing 10.4. Training the model on a dataset](#)

[Listing 10.5. Testing the learned model](#)

[Listing 10.6. Training and testing on dummy data](#)

[Listing 10.7. Loading data](#)

[Listing 10.8. Modifying the test function to pass in the session](#)

[Listing 10.9. Generate training data](#)

Chapter 11. Sequence-to-sequence models for chatbots

[Listing 11.1. Setting up constants and placeholders](#)

[Listing 11.2. Making a simple RNN cell](#)

[Listing 11.3. Stacking two RNN cells](#)

[Listing 11.4. Using MultiRNNCell to stack multiple cells](#)

[Listing 11.5. Defining a lookup table of scalars](#)

[Listing 11.6. Defining a lookup table of 4D vectors](#)

[Listing 11.7. Defining a lookup table of tensors](#)

[Listing 11.8. Looking up the embeddings](#)

[Listing 11.9. Extracting character vocab](#)

[Listing 11.10. Defining hyperparameters](#)

[Listing 11.11. Listing placeholders](#)

[Listing 11.12. Helper functions to build RNN cells](#)

[Listing 11.13. Encoder embedding and cell](#)

[Listing 11.14. Preparing input sequences to the decoder](#)

[Listing 11.15. Decoder embedding and cell](#)

[Listing 11.16. Decoder output \(training\)](#)

[Listing 11.17. Decoder output \(inference\)](#)

[Listing 11.18. Cost function](#)

[Listing 11.19. Calling an optimizer](#)

[Listing 11.20. Training the model](#)

Chapter 12. Utility landscape

[Listing 12.1. Importing relevant libraries](#)

[Listing 12.2. Generating dummy training data](#)

[Listing 12.3. Placeholders](#)

[Listing 12.4. Hidden layer](#)

[Listing 12.5. Output layer](#)

[Listing 12.6. Loss and optimizer](#)

[Listing 12.7. Preparing a session](#)

[Listing 12.8. Training the network](#)

[Listing 12.9. Preparing test data](#)

[Listing 12.10. Visualize results](#)

[Listing 12.11. Importing libraries](#)

[Listing 12.12. Preparing the training data](#)

[Listing 12.13. Placeholder](#)

[Listing 12.14. Preparing the session](#)

[Listing 12.15. Loading the VGG16 model](#)

[Listing 12.16. Preparing data for ranking](#)

[Listing 12.17. Training the ranking network](#)

[Listing 12.18. Preparing image sequences from video](#)

[Listing 12.19. Computing the utility of images](#)

[Listing 12.20. Visualizing utility scores](#)

 [PREV](#)
[List of Tables](#)