

# Table of Contents

|                                                      |    |
|------------------------------------------------------|----|
| <b>Preface</b>                                       | 1  |
| <b>Section 1: Getting Started with Deep Learning</b> |    |
| <b>Chapter 1: Introduction to Deep Learning</b>      | 9  |
| What is deep learning?                               | 9  |
| Biological and artificial neurons                    | 11 |
| ANN and its layers                                   | 13 |
| Input layer                                          | 14 |
| Hidden layer                                         | 14 |
| Output layer                                         | 15 |
| Exploring activation functions                       | 15 |
| The sigmoid function                                 | 15 |
| The tanh function                                    | 16 |
| The Rectified Linear Unit function                   | 17 |
| The leaky ReLU function                              | 18 |
| The Exponential linear unit function                 | 20 |
| The Swish function                                   | 20 |
| The softmax function                                 | 21 |
| Forward propagation in ANN                           | 22 |
| How does ANN learn?                                  | 25 |
| Debugging gradient descent with gradient checking    | 32 |
| Putting it all together                              | 37 |
| Building a neural network from scratch               | 37 |
| Summary                                              | 40 |
| Questions                                            | 41 |
| Further reading                                      | 41 |
| <b>Chapter 2: Getting to Know TensorFlow</b>         | 43 |
| What is TensorFlow?                                  | 43 |
| Understanding computational graphs and sessions      | 44 |
| Sessions                                             | 46 |
| Variables, constants, and placeholders               | 47 |
| Variables                                            | 47 |
| Constants                                            | 49 |
| Placeholders and feed dictionaries                   | 49 |
| Introducing TensorBoard                              | 51 |
| Creating a name scope                                | 53 |
| Handwritten digit classification using TensorFlow    | 55 |
| Importing the required libraries                     | 55 |

|                                                            |     |
|------------------------------------------------------------|-----|
| Loading the dataset                                        | 55  |
| Defining the number of neurons in each layer               | 57  |
| Defining placeholders                                      | 57  |
| Forward propagation                                        | 58  |
| Computing loss and backpropagation                         | 59  |
| Computing accuracy                                         | 60  |
| Creating summary                                           | 61  |
| Training the model                                         | 61  |
| Visualizing graphs in TensorBoard                          | 63  |
| <b>Introducing eager execution</b>                         | 68  |
| <b>Math operations in TensorFlow</b>                       | 69  |
| <b>TensorFlow 2.0 and Keras</b>                            | 73  |
| Bonjour Keras                                              | 73  |
| Defining the model                                         | 73  |
| Defining a sequential model                                | 74  |
| Defining a functional model                                | 74  |
| Compiling the model                                        | 75  |
| Training the model                                         | 76  |
| Evaluating the model                                       | 76  |
| MNIST digit classification using TensorFlow 2.0            | 76  |
| <b>Should we use Keras or TensorFlow?</b>                  | 77  |
| <b>Summary</b>                                             | 78  |
| <b>Questions</b>                                           | 79  |
| <b>Further reading</b>                                     | 79  |
| <b>Section 2: Fundamental Deep Learning Algorithms</b>     |     |
| <b>Chapter 3: Gradient Descent and Its Variants</b>        | 83  |
| <b>Demystifying gradient descent</b>                       | 84  |
| Performing gradient descent in regression                  | 89  |
| Importing the libraries                                    | 90  |
| Preparing the dataset                                      | 90  |
| Defining the loss function                                 | 91  |
| Computing the gradients of the loss function               | 92  |
| Updating the model parameters                              | 94  |
| <b>Gradient descent versus stochastic gradient descent</b> | 96  |
| <b>Momentum-based gradient descent</b>                     | 99  |
| Gradient descent with momentum                             | 99  |
| Nesterov accelerated gradient                              | 101 |
| <b>Adaptive methods of gradient descent</b>                | 103 |
| Setting a learning rate adaptively using Adagrad           | 103 |
| Doing away with the learning rate using Adadelat           | 106 |
| Overcoming the limitations of Adagrad using RMSProp        | 110 |
| Adaptive moment estimation                                 | 111 |
| Adamax – Adam based on infinity-norm                       | 113 |
| Adaptive moment estimation with AMSGrad                    | 116 |

|                                                                 |     |
|-----------------------------------------------------------------|-----|
| Nadam – adding NAG to ADAM                                      | 117 |
| <b>Summary</b>                                                  | 120 |
| <b>Questions</b>                                                | 121 |
| <b>Further reading</b>                                          | 122 |
| <b>Chapter 4: Generating Song Lyrics Using RNN</b>              | 123 |
| <b>Introducing RNNs</b>                                         | 124 |
| The difference between feedforward networks and RNNs            | 126 |
| Forward propagation in RNNs                                     | 127 |
| Backpropagating through time                                    | 130 |
| Gradients with respect to the hidden to output weight, $V$      | 132 |
| Gradients with respect to hidden to hidden layer weights, $W$   | 136 |
| Gradients with respect to input to the hidden layer weight, $U$ | 142 |
| Vanishing and exploding gradients problem                       | 143 |
| Gradient clipping                                               | 146 |
| <b>Generating song lyrics using RNNs</b>                        | 147 |
| Implementing in TensorFlow                                      | 150 |
| Data preparation                                                | 150 |
| Defining the network parameters                                 | 153 |
| Defining placeholders                                           | 154 |
| Defining forward propagation                                    | 154 |
| Defining BPTT                                                   | 155 |
| Start generating songs                                          | 156 |
| <b>Different types of RNN architectures</b>                     | 161 |
| One-to-one architecture                                         | 161 |
| One-to-many architecture                                        | 162 |
| Many-to-one architecture                                        | 163 |
| Many-to-many architecture                                       | 164 |
| <b>Summary</b>                                                  | 164 |
| <b>Questions</b>                                                | 165 |
| <b>Further reading</b>                                          | 165 |
| <b>Chapter 5: Improvements to the RNN</b>                       | 167 |
| <b>LSTM to the rescue</b>                                       | 168 |
| Understanding the LSTM cell                                     | 169 |
| Forget gate                                                     | 170 |
| Input gate                                                      | 171 |
| Output gate                                                     | 172 |
| Updating the cell state                                         | 173 |
| Updating hidden state                                           | 176 |
| Forward propagation in LSTM                                     | 177 |
| Backpropagation in LSTM                                         | 178 |
| Gradients with respect to gates                                 | 179 |
| Gradients with respect to weights                               | 182 |
| Gradients with respect to $V$                                   | 182 |
| Gradients with respect to $W$                                   | 183 |
| Gradients with respect to $U$                                   | 186 |
| Predicting Bitcoin prices using LSTM model                      | 187 |

|                                                       |     |
|-------------------------------------------------------|-----|
| Data preparation                                      | 188 |
| Defining the parameters                               | 191 |
| Define the LSTM cell                                  | 192 |
| Defining forward propagation                          | 192 |
| Defining backpropagation                              | 193 |
| Training the LSTM model                               | 194 |
| Making predictions using the LSTM model               | 195 |
| <b>Gated recurrent units</b>                          | 196 |
| Understanding the GRU cell                            | 197 |
| Update gate                                           | 198 |
| Reset gate                                            | 198 |
| Updating hidden state                                 | 199 |
| Forward propagation in a GRU cell                     | 201 |
| Backpropagation in a GRU cell                         | 201 |
| Gradient with respect to gates                        | 202 |
| Gradients with respect to weights                     | 203 |
| Gradients with respect to V                           | 203 |
| Gradients with respect to W                           | 204 |
| Gradients with respect to U                           | 206 |
| Implementing a GRU cell in TensorFlow                 | 206 |
| Defining the weights                                  | 206 |
| Defining forward propagation                          | 207 |
| <b>Bidirectional RNN</b>                              | 207 |
| <b>Going deep with deep RNN</b>                       | 210 |
| <b>Language translation using the seq2seq model</b>   | 211 |
| Encoder                                               | 212 |
| Decoder                                               | 214 |
| Attention is all we need                              | 217 |
| <b>Summary</b>                                        | 220 |
| <b>Questions</b>                                      | 221 |
| <b>Further reading</b>                                | 221 |
| <b>Chapter 6: Demystifying Convolutional Networks</b> | 223 |
| <b>What are CNNs?</b>                                 | 224 |
| Convolutional layers                                  | 225 |
| Strides                                               | 229 |
| Padding                                               | 230 |
| Pooling layers                                        | 232 |
| Fully connected layers                                | 233 |
| <b>The architecture of CNNs</b>                       | 234 |
| <b>The math behind CNNs</b>                           | 235 |
| Forward propagation                                   | 235 |
| Backward propagation                                  | 237 |
| <b>Implementing a CNN in TensorFlow</b>               | 244 |
| Defining helper functions                             | 244 |
| Defining the convolutional network                    | 245 |
| Computing loss                                        | 247 |

|                                                 |     |
|-------------------------------------------------|-----|
| Starting the training                           | 248 |
| Visualizing extracted features                  | 248 |
| <b>CNN architectures</b>                        | 250 |
| LeNet architecture                              | 251 |
| Understanding AlexNet                           | 251 |
| Architecture of VGGNet                          | 252 |
| GoogleNet                                       | 254 |
| Inception v1                                    | 255 |
| Inception v2 and v3                             | 259 |
| <b>Capsule networks</b>                         | 261 |
| Understanding Capsule networks                  | 264 |
| Computing prediction vectors                    | 265 |
| Coupling coefficients                           | 266 |
| Squashing function                              | 267 |
| Dynamic routing algorithm                       | 267 |
| Architecture of the Capsule network             | 268 |
| The loss function                               | 269 |
| Margin loss                                     | 269 |
| Reconstruction loss                             | 270 |
| <b>Building Capsule networks in TensorFlow</b>  | 271 |
| Defining the squash function                    | 271 |
| Defining a dynamic routing algorithm            | 272 |
| Computing primary and digit capsules            | 273 |
| Masking the digit capsule                       | 275 |
| Defining the decoder                            | 275 |
| Computing the accuracy of the model             | 275 |
| Calculating loss                                | 276 |
| Margin loss                                     | 276 |
| Reconstruction loss                             | 277 |
| Total loss                                      | 277 |
| Training the Capsule network                    | 277 |
| <b>Summary</b>                                  | 278 |
| <b>Questions</b>                                | 279 |
| <b>Further reading</b>                          | 279 |
| <b>Chapter 7: Learning Text Representations</b> | 281 |
| <b>Understanding the word2vec model</b>         | 282 |
| Understanding the CBOW model                    | 283 |
| CBOW with a single context word                 | 286 |
| Forward propagation                             | 288 |
| Backward propagation                            | 291 |
| CBOW with multiple context words                | 294 |
| Understanding skip-gram model                   | 297 |
| Forward propagation in skip-gram                | 298 |
| Backward propagation                            | 302 |
| Various training strategies                     | 303 |
| Hierarchical softmax                            | 303 |

|                                                                 |     |
|-----------------------------------------------------------------|-----|
| Negative sampling                                               | 305 |
| Subsampling frequent words                                      | 306 |
| <b>Building the word2vec model using gensim</b>                 | 306 |
| Loading the dataset                                             | 307 |
| Preprocessing and preparing the dataset                         | 308 |
| Building the model                                              | 310 |
| Evaluating the embeddings                                       | 311 |
| <b>Visualizing word embeddings in TensorBoard</b>               | 312 |
| <b>Doc2vec</b>                                                  | 315 |
| Paragraph Vector – Distributed Memory model                     | 315 |
| Paragraph Vector – Distributed Bag of Words model               | 316 |
| Finding similar documents using doc2vec                         | 317 |
| <b>Understanding skip-thoughts algorithm</b>                    | 320 |
| <b>Quick-thoughts for sentence embeddings</b>                   | 321 |
| <b>Summary</b>                                                  | 322 |
| <b>Questions</b>                                                | 323 |
| <b>Further reading</b>                                          | 323 |
| <b>Section 3: Advanced Deep Learning Algorithms</b>             |     |
| <b>Chapter 8: Generating Images Using GANs</b>                  | 327 |
| <b>Differences between discriminative and generative models</b> | 328 |
| <b>Say hello to GANs!</b>                                       | 329 |
| Breaking down the generator                                     | 331 |
| Breaking down the discriminator                                 | 332 |
| How do they learn though?                                       | 333 |
| Architecture of a GAN                                           | 335 |
| Demystifying the loss function                                  | 335 |
| Discriminator loss                                              | 336 |
| First term                                                      | 336 |
| Second term                                                     | 337 |
| Final term                                                      | 338 |
| Generator loss                                                  | 338 |
| Total loss                                                      | 338 |
| Heuristic loss                                                  | 339 |
| Generating images using GANs in TensorFlow                      | 340 |
| Reading the dataset                                             | 341 |
| Defining the generator                                          | 341 |
| Defining the discriminator                                      | 342 |
| Defining the input placeholders                                 | 342 |
| Starting the GAN!                                               | 342 |
| Computing the loss function                                     | 342 |
| Discriminator loss                                              | 343 |
| Generator loss                                                  | 344 |
| Optimizing the loss                                             | 345 |
| Starting the training                                           | 345 |
| Generating handwritten digits                                   | 345 |

|                                                   |     |
|---------------------------------------------------|-----|
| <b>DCGAN – Adding convolution to a GAN</b>        | 348 |
| Deconvolutional generator                         | 348 |
| Convolutional discriminator                       | 349 |
| Implementing DCGAN to generate CIFAR images       | 350 |
| Exploring the dataset                             | 351 |
| Defining the discriminator                        | 352 |
| Defining the generator                            | 353 |
| Defining the inputs                               | 355 |
| Starting the DCGAN                                | 355 |
| Computing the loss function                       | 355 |
| Discriminator loss                                | 356 |
| Generator loss                                    | 356 |
| Optimizing the loss                               | 356 |
| Train the DCGAN                                   | 357 |
| <b>Least squares GAN</b>                          | 359 |
| Loss function                                     | 361 |
| LSGAN in TensorFlow                               | 362 |
| Discriminator loss                                | 362 |
| Generator loss                                    | 362 |
| <b>GANs with Wasserstein distance</b>             | 363 |
| Are we minimizing JS divergence in GANs?          | 363 |
| What is the Wasserstein distance?                 | 366 |
| Demystifying the k-Lipschitz function             | 367 |
| The loss function of WGAN                         | 368 |
| WGAN in TensorFlow                                | 369 |
| <b>Summary</b>                                    | 370 |
| <b>Questions</b>                                  | 371 |
| <b>Further reading</b>                            | 371 |
| <b>Chapter 9: Learning More about GANs</b>        | 373 |
| <b>Conditional GANs</b>                           | 374 |
| Loss function of CGAN                             | 376 |
| Generating specific handwritten digits using CGAN | 376 |
| Defining the generator                            | 377 |
| Defining discriminator                            | 378 |
| Start the GAN!                                    | 378 |
| Computing the loss function                       | 379 |
| Discriminator loss                                | 379 |
| Generator loss                                    | 380 |
| Optimizing the loss                               | 380 |
| Start training the CGAN                           | 380 |
| Generate the handwritten digit, 7                 | 381 |
| <b>Understanding InfoGAN</b>                      | 382 |
| Mutual information                                | 383 |
| Architecture of the InfoGAN                       | 385 |
| Constructing an InfoGAN in TensorFlow             | 386 |
| Defining generator                                | 387 |
| Defining the discriminator                        | 388 |

|                                                             |     |
|-------------------------------------------------------------|-----|
| Define the input placeholders                               | 390 |
| Start the GAN                                               | 390 |
| Computing loss function                                     | 391 |
| Discriminator loss                                          | 391 |
| Generator loss                                              | 391 |
| Mutual information                                          | 392 |
| Optimizing the loss                                         | 393 |
| Beginning training                                          | 393 |
| Generating handwritten digits                               | 394 |
| <b>Translating images using a CycleGAN</b>                  | 395 |
| Role of generators                                          | 398 |
| Role of discriminators                                      | 399 |
| Loss function                                               | 400 |
| Cycle consistency loss                                      | 401 |
| Converting photos to paintings using a CycleGAN             | 403 |
| <b>StackGAN</b>                                             | 407 |
| The architecture of StackGANs                               | 408 |
| Conditioning augmentation                                   | 409 |
| Stage I                                                     | 409 |
| Generator                                                   | 409 |
| Discriminator                                               | 410 |
| Stage II                                                    | 410 |
| Generator                                                   | 411 |
| Discriminator                                               | 411 |
| <b>Summary</b>                                              | 411 |
| <b>Questions</b>                                            | 412 |
| <b>Further reading</b>                                      | 412 |
| <b>Chapter 10: Reconstructing Inputs Using Autoencoders</b> | 413 |
| <b>What is an autoencoder?</b>                              | 414 |
| Understanding the architecture of autoencoders              | 417 |
| Reconstructing the MNIST images using an autoencoder        | 418 |
| Preparing the dataset                                       | 419 |
| Defining the encoder                                        | 419 |
| Defining the decoder                                        | 420 |
| Building the model                                          | 420 |
| Reconstructing images                                       | 421 |
| Plotting reconstructed images                               | 421 |
| <b>Autoencoders with convolutions</b>                       | 422 |
| Building a convolutional autoencoder                        | 423 |
| Defining the encoder                                        | 424 |
| Defining the decoder                                        | 425 |
| Building the model                                          | 425 |
| Reconstructing the images                                   | 426 |
| <b>Exploring denoising autoencoders</b>                     | 427 |
| Denoising images using DAE                                  | 428 |
| <b>Understanding sparse autoencoders</b>                    | 430 |
| Building the sparse autoencoder                             | 433 |

|                                                              |     |
|--------------------------------------------------------------|-----|
| Defining the sparse regularizer                              | 433 |
| <b>Learning to use contractive autoencoders</b>              | 434 |
| Implementing the contractive autoencoder                     | 434 |
| Defining the contractive loss                                | 435 |
| <b>Dissecting variational autoencoders</b>                   | 436 |
| Variational inference                                        | 438 |
| The loss function                                            | 439 |
| Reparameterization trick                                     | 441 |
| Generating images using VAE                                  | 442 |
| Preparing the dataset                                        | 443 |
| Defining the encoder                                         | 443 |
| Defining the sampling operation                              | 444 |
| Defining the decoder                                         | 444 |
| Building the model                                           | 444 |
| Defining the generator                                       | 445 |
| Plotting generated images                                    | 445 |
| <b>Summary</b>                                               | 446 |
| <b>Questions</b>                                             | 447 |
| <b>Further reading</b>                                       | 447 |
| <b>Chapter 11: Exploring Few-Shot Learning Algorithms</b>    | 449 |
| <b>What is few-shot learning?</b>                            | 450 |
| <b>Siamese networks</b>                                      | 451 |
| <b>Architecture of siamese networks</b>                      | 452 |
| <b>Prototypical networks</b>                                 | 454 |
| <b>Relation networks</b>                                     | 460 |
| <b>Matching networks</b>                                     | 462 |
| Support set embedding function                               | 465 |
| Query set embedding function                                 | 465 |
| The architecture of matching networks                        | 466 |
| <b>Summary</b>                                               | 467 |
| <b>Questions</b>                                             | 468 |
| <b>Further reading</b>                                       | 468 |
| <b>Appendix A: Assessments</b>                               | 469 |
| <b>Chapter 1 - Introduction to Deep Learning</b>             | 469 |
| <b>Chapter 2 - Getting to Know TensorFlow</b>                | 470 |
| <b>Chapter 3 - Gradient Descent and Its Variants</b>         | 471 |
| <b>Chapter 4 - Generating Song Lyrics Using an RNN</b>       | 472 |
| <b>Chapter 5 - Improvements to the RNN</b>                   | 473 |
| <b>Chapter 6 - Demystifying Convolutional Networks</b>       | 474 |
| <b>Chapter 7 - Learning Text Representations</b>             | 475 |
| <b>Chapter 8 - Generating Images Using GANs</b>              | 476 |
| <b>Chapter 9 - Learning More about GANs</b>                  | 477 |
| <b>Chapter 10 - Reconstructing Inputs Using Autoencoders</b> | 478 |

|                                                     |     |
|-----------------------------------------------------|-----|
| Chapter 11 - Exploring Few-Shot Learning Algorithms | 479 |
| Other Books You May Enjoy                           | 481 |
| Index                                               | 485 |