Deep Learning Nanodegree Syllabus

Build Deep Learning Networks Today

Congratulations on considering the Deep Learning Nanodegree program!

Before You Start

Educational Objectives: Become an expert in neural networks, and learn to implement them in Keras and TensorFlow. Build convolutional networks for image recognition, recurrent networks for sequence generation, generative adversarial networks for image generation, and more.

Prerequisite Knowledge: Make sure to set aside adequate time on your calendar for focused work. In order to succeed in this program, we recommend having intermediate experience with Python, including numpy and pandas, and basic knowledge of machine learning. You'll also need to be familiar with algebra, calculus (multivariable derivatives) and linear algebra (matrix multiplication).

If you'd like to refresh your skills for this program, we suggest the AI with Python Nanodegree program.

Contact Info

While going through the program, if you have questions about anything, you can reach us at deeplearning-support@udacity.com.

Nanodegree Program Info

The Deep Learning Nanodegree program offers you a solid introduction to the world of artificial intelligence. In this program, you'll master fundamentals that will enable you to go further in the field, launch or advance a career, and join the next generation of deep learning talent that will help define a beneficial new AI-powered future for our world. You will study cutting-edge topics such as Neural Networks, Convolutional Networks, Recurrent Neural Networks, Generative Adversarial Networks, and Deep Reinforcement Learning, and build projects in Keras and NumPy, in addition to TensorFlow. You'll learn from authorities such as Sebastian Thrun, Ian Goodfellow, and Andrew Trask, and participate in our Experts-in-Residence program, where you'll gain exclusive insights from working professionals in the field. For anyone interested in this transformational technology, this program is an ideal point-of-entry.

The program is comprised of 5 courses and 5 projects. Each project you build will be an opportunity to prove your skills and demonstrate what you've learned in your lessons.

This is a term-based program that requires students to keep pace with their peers. The program is delivered in 1 term spread over 4 months. On average, students will need to spend about 12-15 hours per week in order to complete all required coursework, including lecture and project time.

Length of Program: 4 months
Frequency of Classes: Term-based
Number of Reviewed Projects: 5
Instructional Tools Available: Video lectures, Personalized project reviews, Text instructions, Quizzes, In-classroom mentorship

Projects

Building a project is one of the best ways both to test the skills you've acquired and to demonstrate your newfound abilities to future employers. Throughout this Nanodegree program, you'll have the opportunity to prove your skills by building the following projects:

- Your First Neural Network
- Dog-breed Classifier
- Generate TV Scripts
- Generate Faces
- Teach a Quadcopter How to Fly

In the sections below, you'll find a detailed description of each project along with the course material that presents the skills required to complete the project.

Project 1: Your First Neural Network

Learn neural networks basics, and build your first network with Python and Numpy. Use modern deep learning frameworks (Keras, TensorFlow) to build multi-layer neural networks, and analyze real data. In this project, you will build and train neural networks from scratch to predict the number of bikeshare users on a given day.

Supporting Lesson Content: Neural Networks

Lesson	Learning Outcomes
INTRODUCTION TO NEURAL NETWORKS	➔ In this lesson, you will learn solid foundations on deep learning and neural networks. You'll also implement gradient descent and backpropagation in python right here in the classroom.
IMPLEMENTING GRADIENT DESCENTS	➔ Mat will introduce you to a different error function and guide you through implementing gradient descent using numpy matrix multiplication.
TRAINING NEURAL NETWORKS	➔ Now that you know what neural networks are, in this lesson you will learn several techniques to improve their training.
SENTIMENT ANALYSIS	➔ In this lesson, Andrew Trask, the author of Grokking Deep Learning, will walk you through using neural networks for sentiment analysis.
KERAS	➔ In this section, you'll get a hands-on introduction to Keras. You'll learn to apply it to analyze movie reviews.
TENSORFLOW	➔ In this section you'll get a hands-on introduction to TensorFlow, Google's deep learning framework, and you'll be able to apply it on an image dataset.

Project 2: Dog Breed Classifier

In this project, you will learn how to build a pipeline that can be used within a web or mobile app to process real-world, user-supplied images. Given an image of a dog, your algorithm will identify an estimate of the canine's breed. If supplied an image of a human, the code will identify the resembling dog breed. Along with exploring state-of-the-art CNN models for classification, you will make important design decisions about the user experience for your app.

Supporting Lesson Content: Convolutional Neural Networks

Lesson Title	Learning Outcomes
CLOUD COMPUTING	➔ Take advantage of Amazon's GPUs to train your neural network faster. In this lesson, you'll setup an instance on AWS and train a neural network on a GPU.
CONVOLUTIONAL NEURAL NETWORK	➔ Alexis explains the theory behind Convolutional Neural Networks and how they help us dramatically improve performance in image classification.
CNNs IN TENSORFLOW	➔ In this lesson, you'll walk through an example Convolutional Neural Network (CNN) in TensorFlow. You'll study the line-by-line breakdown of the code and can download the code and run it yourself.
WEIGHT INITIALIZATION	➔ In this lesson, you'll learn how to find good initial weights for a neural network. Having good initial weights can place the neural network close to the optimal solution. This allows the neural network to come to the best solution quicker.
AUTOENCODERS	➔ Autoencoders are neural networks used for data compression, image denoising, and dimensionality reduction. Here, you'll build autoencoders using TensorFlow.
TRANSFER LEARNING IN TENSORFLOW	➔ In practice, most people don't train their own networks on massive datasets. In this lesson, you'll learn how to use a pretrained network on a new problem with transfer learning.
DEEP LEARNING FOR CANCER DETECTION	➔ In this lesson, Sebastian Thrun teaches us about his groundbreaking work detecting skin cancer with convolutional neural networks.

Project 3: Generate TV Scripts

In this project, you will build your own recurrent networks and long short-term memory networks with Keras and TensorFlow. You'll perform sentiment analysis and generate new text, and use recurrent networks to generate new text from TV scripts.

Supporting Lesson Content: Recurrent Neural Networks

Lesson	Learning Outcomes
RECURRENT NEURAL NETWORKS	➔ Ortal will introduce Recurrent Neural Networks (RNNs), which are machine learning models that are able to recognize and act on sequences of inputs.
LONG SHORT-TERM MEMORY NETWORK	➔ Luis explains Long Short-Term Memory Networks (LSTM), and similar architectures which have the benefits of preserving long term memory.
IMPLEMENTATION OF RNN AND LSTM	➔ Overview of what students will learn in this lesson, displayed when students start the lesson.
HYPERPARAMETERS	➔ In this lesson, we'll look at a number of different hyperparameters that are important for our deep learning work. We'll discuss starting values and intuitions for tuning each hyperparameter.
EMBEDDINGS AND WORD2VEC	➔ In this lesson, you'll learn about embeddings in neural networks by implementing the word2vec model.
SENTIMENT PREDICTION RNN	➔ In this lesson, you'll learn to implement a recurrent neural network for predicting sentiment. This is intended to give you more experience building RNNs.

Project 4: Generate Faces

Learn to understand and implement the DCGAN model to simulate realistic images, with Ian Goodfellow, the inventor of GANS (generative adversarial networks). Then, apply what you've learned to build a pair of Multi-Layer Neural Networks and make them compete against each other in order to generate realistic faces.

Supporting Lesson Content: Generative Adversarial Networks

Lesson	Learning Outcomes
GENERATIVE ADVERSARIAL NETWORK	➔ Ian Goodfellow, the inventor of GANs, introduces you to these exciting models. You'll also implement your own GAN on the MNIST dataset.
DEEP CONVOLUTIONAL GANs	➔ In this lesson you'll implement a Deep Convolution GAN to generate complex color images of house numbers.
GENERATE FACES	➔ Compete two neural networks against each other to generate realistic faces.
SEMI-SUPERVISED LEARNING	➔ Ian Goodfellow leads you through a semi-supervised GAN model, a classifier that can learn from mostly unlabeled data.

Project 5: Train a Quadcopter to Fly

In this project, you will design an agent that can fly a quadcopter, and then train it using a reinforcement learning algorithm of your choice. You will apply the techniques you have learnt in this module to find out what works best, but you will also have the freedom to come up with innovative ideas and test them on your own.

The project is divided into 4 sections which cover different aspects of getting the quadcopter to fly such as taking off, hovering, landing and so on.

Supporting Lesson Content: Reinforcement Learning

Lesson Title	Learning Outcomes
WELCOME TO RL	➔ The basics of reinforcement learning and OpenAI Gym.
THE RL FRAMEWORK: THE PROBLEM	➔ Learn how to define Markov Decision Processes to solve real-world problems.
THE RL FRAMEWORK: THE SOLUTION	➔ Learn about policies and value functions. ➔ Derive the Bellman Equations.
DYNAMIC PROGRAMMING	➔ Write your own implementations of iterative policy evaluation, policy improvement, policy Iteration, and value Iteration.
MONTE CARLO METHODS	➔ Implement classic Monte Carlo prediction and control methods. ➔ Learn about greedy and epsilon-greedy policies. ➔ Explore solutions to the Exploration-Exploitation Dilemma.
TEMPORAL-DIFFERENCE METHODS	➔ Learn the difference between the Sarsa, Q-Learning, and Expected Sarsa algorithms.
RL IN CONTINUOUS SPACES	➔ Learn how to adapt traditional algorithms to work with continuous spaces.
DEEP Q-LEARNING	➔ Extend value-based reinforcement learning methods to complex problems using deep neural networks.
POLICY GRADIENTS	➔ Policy-based methods try to directly optimize for the optimal policy. Learn how they work, and why they are important, especially for domains with continuous action spaces.
ACTOR-CRITIC METHODS	➔ Learn how to combine value-based and policy-based methods, bringing together the best of both worlds, to solve challenging reinforcement learning problems.