# First Course: Neural Networks and Deep Learning

Be able to explain the major trends driving the rise of deep learning, and understand where and how it is applied today

**Learning target**:

Understand the major trends driving the rise of deep learning.

Be able to explain how deep learning is applied to supervised learning

Understand what the major categories of models are (such as CNNs and RNNs), and when they should be applied

Be able to recognize the basics of when deep learning will (or will not) work well

**Introduction to Deep Learning:**

What is a neural network?

Supervised learning with neural networks

Why is deep learning taking off?

Learn to set up a machine learning problem with a neural network mindset. Learn to use vectorization to speed up your models

**Learning target:**

Build a logistic regression model, structured a shallow neural network

Implement the main steps of an ML algorithm, including prediction, derivative computation, and gradient descent

Implement computationally efficient, highly vectorized, versions of models

Understand how to compute derivatives of logistic regression, using a backpropagation mindset

Become familiar with python and numpy

Work with iPython notebooks

Be able to implement vectorization across multiply training examples

**Logistic regression as a neural network:**

Binary classification

Logistic regression

Logistic regression cost function

Gradient descent

Derivative

More derivative examples

Computation graph

Derivatives with a computation graph

Logistic regression gradient descent

Gradient descent on m example

**Python and vectorization:**

Vectorization

More vectorization examples

Vectorizing logistic regression

Vectorizing logistic regression’s gradient output

Broadcasting in python

A note on python/numpy vectors

Quick tour of Jupyter/iPython notebooks

Explanation of logistic regression cost function

Learn to build a neural network with one hidden layer, using forward propagation and backpropagation

**Learning target:**

Understand hidden units and hidden layers

Be able to apply a variety of activation functions in a neural network

Build your first forward and backward propagation with a hidden layer

Apply random initialization of your neural network

Become fluent with deep learning notations and representations

Build and train a neural network with one hidden layer

**Shallow neural network:**

Neural network overview

Neural network representation

Computing a neural network’s output

Vectorizing across multiple examples

Explanation for vectorized implementation

Activation functions

Why do you need non-linear activation functions?

Derivatives of activation functions

Gradient descent for neural networks

Backpropagation intuition

Random initialization

Understand the key computations underlying deep learning, use them to build and train deep neural networks, and apply it to computer vision

**Learning target:**

See deep neural networks as successive blocks put one after each other

Build and train a deep L-layer neural network

Analyze matrix and vector dimension to check neural network implementations

Understand how to use a cache to pass information from forward propagation to backward propagation

Understand the role of hyperparameters in deep learning

**Deep neural network:**

Deep L-layer neural network

Forward propagation in a deep network

Getting your matrix dimension right

Why deep representations?

Build blocks of deep neural networks

Forward and backward propagation

Parameters vs hyperparameters

What does this have to do with the brain?

# Second Course: Improving Deep Neural Network: Hyperparameter tuning, Regularization and Optimization

Practical aspects of deep learning

**Learning target:**

Recall that different types of initializations lead to different results

Recognize the importance of initialization in complex neural networks

Recognize the difference between train/dev/test sets

Diagnose the bias and variance issues in your model

Learn when to use regularization methods such as dropout or L2 regularization

Understand experimental issues in deep learning such as vanishing or exploding gradients and learn how to deal with them

Use gradient checking to verify the correctness of your backpropagation implementation

**Setting up your machine learning application:**

Train/dev/test sets

Bias/variance

Basic recipe for machine learning

**Regularizing your neural network:**

Regularization

Why regularization reduce overfitting?

Dropout regularization

Understanding dropout

Other regularization methods

**Setting up your optimization problem:**

Normalizing inputs

Vanishing/exploding gradients

Weight initialization for deep networks

Numerical approximation of gradients

Gradient checking

Gradient checking implementation notes

Optimization algorithm

**Learning target:**

Remember different optimization methods such as (stochastic) gradient descent, momentum, RMSProp and Adam

Use random mini-batches to accelerate the convergence and improve the optimization

Know the benefit of learning rate decay and apply it to your optimization

**Optimization algorithm:**

Mini-batch gradient descent

Understanding mini-batch gradient descent

Understanding exponentially weighted averages

Bias correction in exponentially weighted averages

Gradient descent with momentum

RMSProp

Adam optimization algorithm

Learning rate decay

The problem of local optima

Hyperparameter tuning, batch normalization and programming frameworks

**Learning target:**

Master the process of hyperparameter tuning

**Hyperparameter tuning:**

Tuning process

Using an appropriate scale to pick hyperparameters

Hyperparameters tuning in practice: Pandas vs Caviar

**Batch normalization:**

Normalizing activations in a network

Fitting batch norm into a neural network

Why does batch norm work?

Batch norm at test time

**Multi-class classification:**

Softmax regression

Training a softmax classifier

**Introduction to programming framework:**

Deep learning frameworks

Tensorflow

# Third Course: Structuring Machine Learning Project

ML strategy (1)

**Learning target:**

Understand why machine learning strategy is important

Apply satisficing and optimizing metrics to set up your goal for ML project

Choose a correct train/dev/test split of your dataset

Understand how to define human-level performance

Use human-level performance to define your key priorities of ML projects

Take the correct strategic decision based on observations of performances and dataset

**Introduction to ML strategy:**

Why ML strategy

Orthogonalization

**Setting up your goal:**

Single number evaluation metric

Satisficing and optimizing metric

Train/dev/test distributions

Size of the dev and test sets

When to change dev/test sets and metrics

**Comparing to human-level performance:**

Why human-level performance?

Avoidable bias

Understanding human-level performance

Surpassing human-level performance

Improving your model performance

ML strategy (2)

**Learning target:**

Understand what multi-task learning and transfer learning are

Recognize bias, variance and data-mismatch by looking at the performance of your algorithm on train/dev/test sets

**Error analysis:**

Carrying out error analysis

Cleaning up incorrectly labeled data

Build your first system quickly, then iterate

**Mismatched training and dev/test set:**

Training and testing on different distributions

Bias and variance with mismatched data distributions

Addressing data mismatch

**Learning from multiple tasks:**

Transfer learning

Multi-task learning

**End-to-end deep learning:**

What is end-to-end deep learning

Whether to use end-to-end deep learning

# Forth Course: Convolutional Neural Networks

Learn to implement the foundation layers of CNNs (pooling, convolutions) and to stack them properly in a deep network to solve multi-class image classification problems

**Learning target:**

Understand the convolution operation

Understand the pooling operation

Remember the vocabulary used in convolutional neural network (padding, stride, filter, …)

Build a convolutional neural network for image multi-class classification

**Convolutional neural networks:**

Computer vision

Edge detection example

More edge detection

Padding

Strided convolutions

Convolutions over volume

One layer of a convolutional network

Simple convolutional network example

Pooling layers

CNN example

Why convolutions?

Learn about practical tricks and methods used in deep CNNs straight from the research papers

**Learning target:**

Understand multiple foundational papers of convolutional neural networks

Analyze the dimensionality reduction of a volume in a very deep network

Understand and implement a residual network

Build a deep neural network using Keras

Implement a skip-connection in your network

Clone a repository from github and use transfer learning

**Case studies:**

Why look at case studies?

Classic networks

ResNets

Why ResNets work

Networks in networks and 1x1 convolutions

Inception network motivation

Inception network

**Practical advices for using ConvNets:**

Using open-source implementation

Transfer learning

Data augmentation

State of computer vision

Learn how to apply your knowledge of CNNs to one of toughest but hottest field of computer vision: Object detection

**Learning target:**

Understand the challenges of object localization, object detection and landmark finding

Understand and implement intersection over union

Understand how we label a dataset for an object detection application

Remember the vocabulary of object detection (landmark, anchor, bounding box, grid, …)

**Detection algorithm:**

Object localization

Landmark detection

Object detection

Convolutional implementation of sliding windows

Bounding box predictions

Intersection over union

Non-max suppression

Anchor boxes

YOLO algorithm

Region proposals

Discover how CNNs can be applied to multiple fields, including art generation and face recognition. Implement your own algorithm to generate art and recognize face.

**Face recognition:**

What is face recognition?

One shot learning

Siamese network

Triplet loss

Face verification and binary classification

**Neural style transfer:**

What is neural style transfer?

What are deep ConvNets learning?

Cost function

Content cost function

Style cost function

1D and 3D generalizations

# Fifth Course: Sequence Models

Learn about recurrent neural networks. This type of model has been proven to perform extremely well on temporal data. It has several variants including LSTMs, GRUs and Bidirectional RNNs, which you are going to learn about in this section.

**Recurrent neural networks:**

Why sequence models

Notation

Recurrent neural network model

Backpropagation through time

Different types of RNNs

Language model and sequence generation

Sampling novel sequences

Vanishing gradients with RNNs

Gated Recurrent Unit(GRU)

Long Short Term Memory(LSTM)

Bidirectional RNN

Deep RNNs

Natural language processing with deep learning is an important combination. Using word vector representation and embedding layers you can train recurrent neural networks with outstanding performances in a wide variety of industries. Examples of applications are sentiment analysis, named entity recognition and machine translation.

**Introduction to word embeddings:**

Word representation

Using word embeddings

Properties of embeddings

Embedding matrix

**Learning word embeddings: Word2vec & GloVe:**

Learning word embeddings

Word2Vec

Negative sampling

GloVe word vectors

**Applications using word embeddings:**

Sentiment classification

Debiasing word embeddings

Sequence models can be augmented using an attention mechanism. This algorithm will help your model understand where it should focus its attention given a sequence of inputs. This week, you will also learn about speech recognition and how to deal with audio data.

**Various sequence to sequence architectures:**

Basic models

Picking the most likely sentence

Beam search

Refinements to beam search

Error analysis in beam search

Blue score

Attention model intuition

Attention model

**Speech recognition – audio data:**

Speech recognition

Trigger word detection