深度學習視覺辨識
(Deep Learning for Visual Recognition)

Time: 2018/01/25 (Thu) (9:00 -12:00, 13:00-16:00)
Place: 國立臺北護理健康大學 (台北市明德路365號) 親仁樓 B112
Host: 祝國忠 院長 (健康科技學院院長)

Min-Yuh Day
戴敏育
Assistant Professor
專任助理教授

Dept. of Information Management, Tamkang University
淡江大學 資訊管理學系

http://mail.tku.edu.tw/myday/
2018-01-25
戴敏育博士
(Min-Yuh Day, Ph.D.)
淡江大学资管系兼任助理教授
中央研究院资讯科学研究所访问学人
国立台湾大学资讯管理博士

Publications Co-Chairs, IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013-)
Program Co-Chair, IEEE International Workshop on Empirical Methods for Recognizing Inference in TExt (IEEE EM-RITE 2012-)
Workshop Chair, The IEEE International Conference on Information Reuse and Integration (IEEE IRI)
Outline

• AI, Machine Learning and Deep Learning

• Visual Recognition
  – Image Classification

• Deep Learning for Visual Recognition with TensorFlow and Keras
CIFAR-10 dataset

airplane

automobile

bird

cat

deer

dog

frog

horse

ship

truck

Source: https://www.cs.toronto.edu/~kriz/cifar.html
Deep Learning Foundations:
Neural Networks
Keras:
High-level API for TensorFlow
Artificial Intelligence

Machine Learning & Deep Learning

ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.

MACHINE LEARNING
Machine learning begins to flourish.

DEEP LEARNING
Deep learning breakthroughs drive AI boom.


Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Artificial Intelligence (AI) is many things

Ecosystem of AI

Source: https://www.i-scoop.eu/artificial-intelligence-cognitive-computing/
Artificial Intelligence (AI)
Intelligent Document Recognition algorithms

Source: https://www.i-scoop.eu/artificial-intelligence-cognitive-computing/
Deep Learning Evolution

Source: http://www.erogol.com/brief-history-machine-learning/
Machine Learning Models

Deep Learning

Association rules

Decision tree

Clustering

Bayesian

Kernel

Ensemble

Dimensionality reduction

Regression Analysis

Instance based

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
3 Machine Learning Algorithms

Machine Learning (ML) / Deep Learning (DL)

Supervised Learning
- Decision Tree Classifiers
- Linear Classifiers
- Rule-based Classifiers
- Probabilistic Classifiers
  - Support Vector Machine (SVM)
  - Neural Network (NN)
  - Deep Learning (DL)
  - Naïve Bayes (NB)
  - Bayesian Network (BN)
  - Maximum Entropy (ME)

Unsupervised Learning
- Reinforcement Learning

Deep Learning and Deep Neural Networks
"Deep learning."

Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics. Deep learning discovers intricate structure in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Deep convolutional nets have brought about breakthroughs in processing images, video, speech and audio, whereas recurrent nets have shone light on sequential data such as text and speech.

Machine-learning technology powers many aspects of modern society: from web searches to content filtering on social networks to recommendations on e-commerce websites, and it is increasingly present in consumer products such as cameras and smartphones. Machine-learning systems are used to identify objects in images, transcribe speech into text, match news items, posts or products with users’ interests, and select relevant results of search. Increasingly, these applications make use of a class of techniques called deep learning.

Conventional machine-learning techniques were limited in their ability to process natural data in their raw form. For decades, conventional methods attempted to encode the natural data into a feature space that was more amenable to machine learning, but often were limited by the methods used to construct such feature spaces. Deep learning aims to process the raw data directly, using multiple layers of non-linearity to discover abstract representations of the data useful for a large collection of tasks. Deep learning has been very successful in classifying images, where it has outperformed human-level performance on a number of visual object recognition benchmarks.

Deep learning can also be used to discover natural language understanding in a data-driven manner. Deep-learning-based methods can be used to detect human sentiment from text, and to perform automatic translation between languages. In addition, deep-learning-based methods are used in a variety of other applications such as medicine and computer vision.

Deep Learning

- A powerful class of machine learning model
- Modern reincarnation of artificial neural networks
- Collection of simple, trainable mathematical functions
- Compatible with many variants of machine learning

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
What is Deep Learning?

- Loosely based on (what little) we know about the brain

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Neural Networks (NN)
A mostly complete chart of
Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org

Source: http://www.asimovinstitute.org/neural-network-zoo/
Convolutional Neural Networks
(CNN or Deep Convolutional Neural Networks, DCNN)


Source: http://www.asimovinstitute.org/neural-network-zoo/
Recurrent Neural Networks (RNN)

Source: http://www.asimovinstitute.org/neural-network-zoo/
Long / Short Term Memory (LSTM)


Source: http://www.asimovinstitute.org/neural-network-zoo/
Gated Recurrent Units (GRU)


Source: http://www.asimovinstitute.org/neural-network-zoo/
Generative Adversarial Networks (GAN)

Source: http://www.asimovinstitute.org/neural-network-zoo/
Support Vector Machines (SVM)


Source: http://www.asimovinstitute.org/neural-network-zoo/
Neural networks (NN) 1960

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

X1  

X2  

Source: https://www.youtube.com/watch?v=bxeffV8XR&index=1&list=PLiaHhY2lB9dHaRr6b7XevZt2RzP0U
Multilayer Perceptrons (MLP) 1985

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Support Vector Machine (SVM) 1995

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Hinton presents the Deep Belief Network (DBN)

New interests in deep learning and RBM

State of the art MNIST 2005

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Deep Recurrent Neural Network (RNN) 2009

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Convolutional DBN 2010

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Max-Pooling CDBN 2011

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Deep Learning

Geoffrey Hinton
Yann LeCun
Yoshua Bengio
Andrew Y. Ng
From image to text

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

From image to text

Image: deep convolution neural network (CNN)
Text: recurrent neural network (RNN)

A group of **people** sitting on a boat in the water.

Recurrent Neural Network (RNN)

Course Description

Natural language processing (NLP) is one of the most important technologies of the information age. Understanding complex language utterances is also a crucial part of artificial intelligence. Applications of NLP are everywhere because people communicate most everything in language: web search, advertisement, emails, customer service, language translation, radiology reports, etc. There are a large variety of underlying tasks and machine learning models powering NLP applications. Recently, deep learning approaches have obtained very high performance across many different NLP tasks. These models can often be trained with a single end-to-end model and do not require traditional, task-specific feature engineering. In this spring quarter course students will learn to implement, train, debug, visualize and invent their own neural network models. The course provides a deep excursion into cutting-edge research in deep learning applied to NLP. The final project will involve training a complex recurrent neural network and applying it to a large scale NLP problem. On the model side we will cover word vector representations, http://cs224d.stanford.edu/
Recurrent Neural Networks (RNNs)
RNN

\[ \begin{align*}
& h_t \\
& X_t \\
& A \\
& \quad \\
& h_t
\end{align*} \]

= 

\[ \begin{align*}
& h_0 \\
& X_0 \\
& A \\
& A \\
& \quad \\
& \cdots \\
& X_t \quad \quad \\
& h_t
\end{align*} \]

Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
I grew up in France… I speak fluent French.

Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Long Short Term Memory (LSTM)

Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Gated Recurrent Unit (GRU)

\[ h_t \leftarrow h_{t-1} \]

\[ x_t \]

\[ r_t \]

\[ z_t \]

\[ \tilde{h}_t \]

\[ \sigma \]

\[ \text{tanh} \]

Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
LSTM vs GRU

LSTM

i, f and o are the input, forget and output gates, respectively. c and c̃ denote the memory cell and the new memory cell content.

GRU

r and z are the reset and update gates, and h and h̃ are the activation and the candidate activation.

LSTM Recurrent Neural Network

Source: https://github.com/Vict0rSch/deep_learning/tree/master/keras/recurrent
The Sequence to Sequence model (seq2seq)

Source: http://suriyadeepan.github.io/2016-12-31-practical-seq2seq/
Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

Source: https://www.youtube.com/watch?v=bxetV8XR5&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
The Neuron

\[ x_1, x_2, \ldots, x_n \rightarrow w_1, w_2, \ldots, w_n \rightarrow y \]
Neuron and Synapse

Source: https://en.wikipedia.org/wiki/Neuron
The Neuron

\[ y = F \left( \sum_{i} w_i x_i \right) \]

\[ F(x) = \max(0, x) \]

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
y = \text{max} \left( 0, -0.21 \cdot x_1 + 0.3 \cdot x_2 + 0.7 \cdot x_3 \right)
Neural Networks

Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2lBX9hdHaRr6b7XevZtgZRa1PoU
Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

Source: https://www.youtube.com/watch?v=bxet2-T8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Neural Networks

Input Layer
(X)

Hidden Layers
(H)

Output Layer
(Y)

Deep Neural Networks
Deep Learning

Source: https://www.youtube.com/watch?v=bx62T-V8XRz&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

Neuron

Synapse

X1

X2

Y
Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

Hours
Sleep

Hours
Study

Score

Source: https://www.youtube.com/watch?v=bx2T-V8XR&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Neural Networks

Input Layer \((X)\)  
Hidden Layer \((H)\)  
Output Layer \((Y)\)  

Source: https://www.youtube.com/watch?v=bxet-V8XR&index=1&list=PLiaHhY2BX9hdHaRr6b7XevZtgZRa1PoU
<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours</td>
<td>Hours</td>
</tr>
<tr>
<td>Sleep</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>-----------</td>
<td>---</td>
</tr>
<tr>
<td>Hours Sleep</td>
<td>Hours Study</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>Training</td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td>8</td>
</tr>
</tbody>
</table>
\[ Y = W X + b \]
Output

$Y = WX + b$

input

Weights

bias

Trained

Source: https://www.youtube.com/watch?v=G8eNWzOgqE
\[ W X + b = Y \]

Scores → Probabilities

Source: https://www.youtube.com/watch?v=G8eNWzxOggE
SoftMAX

\[ W X + b = Y \]

\[
\begin{bmatrix}
2.0 \\
1.0 \\
0.1
\end{bmatrix}
\]

\[
S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}
\]

Logits \quad Scores \quad Probabilities

Source: https://www.youtube.com/watch?v=G8eNWzxOgqE
\[ S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} = \frac{e^{2.0}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{2.0}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.7 \]

\[ S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} = \frac{e^{1.0}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{1.0}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.2 \]

\[ S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} = \frac{e^{0.1}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{0.1}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.1 \]
Training a Network

= Minimize the Cost Function

Source: https://www.youtube.com/watch?v=bxe2T-V8XR&s=index=1&list=PLiaHhY2IBX9hdHaRr6b7XevZtgZRa1PoU
Training a Network

= Minimize the **Cost** Function
Minimize the **Loss** Function
Error = Predict Y - Actual Y
Error : Cost : Loss

Source: https://www.youtube.com/watch?v=bxerT-V8XR8&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Error = Predict Y - Actual Y

Error : Cost : Loss

Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Error = Predict Y - Actual Y

Error : Cost : Loss

Source: https://www.youtube.com/watch?v=bxet8XR86Q&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
Activation Functions
Activation Functions

Sigmoid

\[ f(x) = \frac{1}{1 + e^{-x}} \]

[0, 1]

TanH

\[ f(x) = \tanh(x) \]

[-1, 1]

ReLU (Rectified Linear Unit)

\[ f(x) = \max(0, x) \]
Activation Functions

Sigmoid:
\[ f(x) = \frac{1}{1 + e^{-x}} \]

Tanh:
\[ \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \]

ReLU:
\[
\begin{align*}
  f(x) &= \begin{cases} 
    0 & \text{for } x < 0 \\
    x & \text{for } x \geq 0 
  \end{cases}
\end{align*}
\]
Loss Function
Binary Classification: 2 Class

Activation Function: Sigmoid

Loss Function: Binary Cross-Entropy
Multiple Classification: 10 Class

Activation Function: SoftMAX

Loss Function: Categorical Cross-Entropy
Dropout: a simple way to prevent neural networks from overfitting

(a) Standard Neural Net

(b) After applying dropout.

Learning Algorithm

While not done:

Pick a random training example “(input, label)”
Run neural network on “input”
Adjust weights on edges to make output closer to “label”

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
\[ y = \max(0, -0.21 \times x_1 + 0.3 \times x_2 + 0.7 \times x_3) \]
Next time:

\[
y = \max (0, -0.23 \times x_1 + 0.31 \times x_2 + 0.65 \times x_3)
\]

\[
y = \max (0, -0.21 \times x_1 + 0.3 \times x_2 + 0.7 \times x_3)
\]

Weights

- $x_1$ with weight $-0.23$
- $x_1$ with weight $-0.21$
- $x_2$ with weight $0.31$
- $x_2$ with weight $0.3$
- $x_3$ with weight $0.65$
- $x_3$ with weight $0.7$

Inputs

Outputs

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Optimizer:
Stochastic Gradient Descent (SGD)

\[ J(w) \]

Global cost minimum
Initial weight

Gradient
This shows a function of 2 variables: real neural nets are functions of hundreds of millions of variables!
Visual Recognition

Image Classification
Convolutional Neural Networks
(CNNs / ConvNets)

http://cs231n.github.io/convolutional-networks/
A regular 3-layer Neural Network

http://cs231n.github.io/convolutional-networks/
A ConvNet arranges its neurons in three dimensions (width, height, depth)

http://cs231n.github.io/convolutional-networks/
The activations of an example ConvNet architecture.

http://cs231n.github.io/convolutional-networks/
ConvNets

http://cs231n.github.io/convolutional-networks/
ConvNets

http://cs231n.github.io/convolutional-networks/
ConvNets

http://cs231n.github.io/convolutional-networks/
ConvNets
max pooling

max pool with 2x2 filters and stride 2

http://cs231n.github.io/convolutional-networks/
Convolutional Neural Networks (CNN) (LeNet)
Sparse Connectivity

Source: http://deeplearning.net/tutorial/lenet.html
Convolutional Neural Networks (CNN) (LeNet)

Shared Weights

Source: http://deeplearning.net/tutorial/lenet.html
Convolutional Neural Networks (CNN) (LeNet)

example of a convolutional layer

Source: http://deeplearning.net/tutorial/lenet.html
Convolutional Neural Networks (CNN) (LeNet)

Source: [http://deeplearning.net/tutorial/lenet.html](http://deeplearning.net/tutorial/lenet.html)
Neural Network and Deep Learning

28 × 28 = 784

“Activation”
Gradient Descent
how neural networks learn

Average cost of all training data...

\[
\begin{align*}
(0.18 - 0.00)^2 + \\
(0.29 - 0.00)^2 + \\
(0.58 - 0.00)^2 + \\
(0.77 - 0.00)^2 + \\
(0.20 - 0.00)^2 + \\
(0.36 - 0.00)^2 + \\
(0.93 - 0.00)^2 + \\
(1.00 - 0.00)^2 + \\
(0.95 - 1.00)^2 + \\
(0.35 - 0.00)^2
\end{align*}
\]

What’s the “cost of this difference?"

Source: 3Blue1Brown (2017), Gradient descent, how neural networks learn | Chapter 2, deep learning, https://www.youtube.com/watch?v=IHzwWFHWa-w
Backpropagation

Source: 3Blue1Brown (2017), What is backpropagation really doing? | Chapter 3, deep learning, https://www.youtube.com/watch?v=Ilg3gGewQ5U
Learning Algorithm

While not done:

Pick a random training example “(input, label)”
Run neural network on “input”
Adjust weights on edges to make output closer to “label”

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Deep Learning with TensorFlow
# Deep Learning Libraries: Tensorflow and Keras

## Deep learning libraries: GitHub activity from February 11 to April 12, 2017

<table>
<thead>
<tr>
<th>new contributors from 2017-02-11 to 2017-04-12</th>
<th>new forks from 2017-02-11 to 2017-04-12</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1: 131 tensorflow/tensorflow</td>
<td>#1: 4192 tensorflow/tensorflow</td>
</tr>
<tr>
<td>#2: 63  fchollet/keras</td>
<td>#2: 991  fchollet/keras</td>
</tr>
<tr>
<td>#3: 51  pytorch/pytorch</td>
<td>#3: 810  BVLC/caffe</td>
</tr>
<tr>
<td>#4: 49  dmnc/mxnet</td>
<td>#4: 517  deeplearning4j/deeplearning4j</td>
</tr>
<tr>
<td>#5: 18  Theano/Theano</td>
<td>#5: 414  dmnc/mxnet</td>
</tr>
<tr>
<td>#6: 11  BVLC/caffe</td>
<td>#6: 307  pytorch/pytorch</td>
</tr>
<tr>
<td>#7: 11  Microsoft/CNTK</td>
<td>#7: 244  Microsoft/CNTK</td>
</tr>
<tr>
<td>#8: 9   tflearn/tflearn</td>
<td>#8: 211  tflearn/tflearn</td>
</tr>
<tr>
<td>#9: 9   pfnet/chainer</td>
<td>#9: 134  torch/torch7</td>
</tr>
<tr>
<td>#10: 8  torch/torch7</td>
<td>#10: 131 Theano/Theano</td>
</tr>
<tr>
<td>#11: 5  deeplearning4j/deeplearning4j</td>
<td>#11: 116 baidu/paddle</td>
</tr>
<tr>
<td>#12: 4  NVIDIA/DIGITS</td>
<td>#12: 88  NVIDIA/DIGITS</td>
</tr>
<tr>
<td>#13: 3  baidu/paddle</td>
<td>#13: 55  pfnet/chainer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>new issues from 2017-02-11 to 2017-04-12</th>
<th>aggregate activity from 2017-02-11 to 2017-04-12</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1: 1175 tensorflow/tensorflow</td>
<td>#1: 36.64 tensorflow/tensorflow</td>
</tr>
<tr>
<td>#2: 568  fchollet/keras</td>
<td>#2: 12.52  fchollet/keras</td>
</tr>
<tr>
<td>#3: 499  dmnc/mxnet</td>
<td>#3: 8.53  dmnc/mxnet</td>
</tr>
<tr>
<td>#4: 286  pytorch/pytorch</td>
<td>#4: 6.09  BVLC/caffe</td>
</tr>
<tr>
<td>#5: 257  Microsoft/CNTK</td>
<td>#5: 5.92  pytorch/pytorch</td>
</tr>
<tr>
<td>#6: 239  deeplearning4j/deeplearning4j</td>
<td>#6: 5.12  deeplearning4j/deeplearning4j</td>
</tr>
<tr>
<td>#7: 219  baidu/paddle</td>
<td>#7: 4.12  Microsoft/CNTK</td>
</tr>
<tr>
<td>#8: 173  Theano/Theano</td>
<td>#8: 2.93  Theano/Theano</td>
</tr>
<tr>
<td>#9: 171  BVLC/caffe</td>
<td>#9: 2.86  baidu/paddle</td>
</tr>
<tr>
<td>#10: 112 NVIDIA/DIGITS</td>
<td>#10: 2.17  tflearn/tflearn</td>
</tr>
<tr>
<td>#11: 84  tflearn/tflearn</td>
<td>#11: 1.68  NVIDIA/DIGITS</td>
</tr>
<tr>
<td>#12: 57  pfnet/chainer</td>
<td>#12: 1.38  torch/torch7</td>
</tr>
<tr>
<td>#13: 47  torch/torch7</td>
<td>#13: 1.12  pfnet/chainer</td>
</tr>
</tbody>
</table>

Source: https://twitter.com/fchollet/status/852194634470223873
Deep Learning Software

• Keras
  – Deep Learning library for TensorFlow, CNTK

• Tensorflow
  – TensorFlow™ is an open source software library for numerical computation using data flow graphs.

• CNTK
  – Computational Network Toolkit by Microsoft Research

• PyTorch
  – Tensors and Dynamic neural networks in Python with strong GPU acceleration

Source: http://deeplearning.net/software_links/
Keras: The Python Deep Learning library

You have just found Keras.

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

Read the documentation at Keras.io.

Keras is compatible with: Python 2.7-3.6.

http://keras.io/
TensorFlow

An open-source software library for Machine Intelligence

Eager Execution
We're announcing eager execution, an imperative, define-by-run interface to TensorFlow. Check out the README to get started today.

TensorFlow 1.4 has arrived!
We're excited to announce the release of TensorFlow 1.4! Check out the release notes for all the latest.

Announcing TensorFlow Lite
Learn more about TensorFlow's lightweight solution for mobile and embedded devices.

https://www.tensorflow.org/
PyTorch

Tensors and Dynamic neural networks in Python with strong GPU acceleration.

PyTorch is a deep learning framework that puts Python first.

We are in an early-release Beta. Expect some adventures.

Learn More

http://pytorch.org/
• Keras is a **high-level neural networks API**
• Written in Python and capable of running on top of TensorFlow, CNTK, or Theano.
• It was developed with a focus on enabling fast experimentation.
• Being able to go from idea to result with the least possible delay is key to doing good research.

Source: https://keras.io/
Install Keras

• Step 1. Install backend engines: TensorFlow
  – Installing TensorFlow on Ubuntu
  – Installing TensorFlow on macOS
  – Installing TensorFlow on Windows

• Step 2. Install Keras
  – sudo pip install keras
  – pip install keras

https://keras.io/#installation
Installing TensorFlow

We've built and tested TensorFlow on the following 64-bit laptop/desktop operating systems:

- MacOS X 10.11 (El Capitan) or later.
- Ubuntu 14.04 or later
- Windows 7 or later.

Although you might be able to install TensorFlow on other laptop or desktop systems, we only support (and only fix issues in) the preceding configurations.

The following guides explain how to install a version of TensorFlow that enables you to write applications in Python:

- Installing TensorFlow on Ubuntu
- Installing TensorFlow on macOS
- Installing TensorFlow on Windows
- Installing TensorFlow from Sources

Many aspects of the Python TensorFlow API changed from version 0.n to 1.0. The following guide explains how to migrate older TensorFlow applications to Version 1.0:

https://www.tensorflow.org/install/
Keras Installation

**Installation**

Keras uses the following dependencies:

- numpy, scipy
- yaml
- HDF5 and h5py (optional, required if you use model saving/loading functions)
- Optional but recommended if you use CNNs: cuDNN.

**When using the TensorFlow backend:**

- TensorFlow
  - See installation instructions.

**When using the Theano backend:**

- Theano
  - See installation instructions.

To install Keras, `cd` to the Keras folder and run the install command:

```
sudo python setup.py install
```

You can also install Keras from PyPI:

```
sudo pip install keras
```

[https://keras.io/#installation](https://keras.io/#installation)
Google TensorFlow

About TensorFlow

TensorFlow™ is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API.

https://www.tensorflow.org/
TensorFlow Playground

Tinker With a **Neural Network** Right Here in Your Browser.
Don’t Worry, You Can’t Break It. We Promise.

DATA
Which dataset do you want to use?
Which properties do you want to feed in?

Ratio of training to test data: 50%
Noise: 0
Batch size: 10

INPUT

OUTPUT
Test loss 0.000
Training loss 0.000

4 neurons
2 neurons
2 neurons

The outputs are mixed with varying *weights*, shown by the thickness of the lines.
This is the output from one *neuron*. Hover to see it larger.

http://playground.tensorflow.org/
TensorFlow is an Open Source Software Library for Machine Intelligence

https://www.tensorflow.org/
numerical computation using data flow graphs
Tensor

- 3
  - # a rank 0 tensor; this is a **scalar** with shape []
- [1., 2., 3.]
  - # a rank 1 tensor; this is a **vector** with shape [3]
- [[[1., 2., 3.], [4., 5., 6.]]
  - # a rank 2 tensor; a **matrix** with shape [2, 3]
- [[[1., 2., 3.]], [[7., 8., 9.]]]
  - # a rank 3 **tensor** with shape [2, 1, 3]
Nodes:
mathematical operations

edges:
multidimensional data arrays (tensors)
communicated between nodes
Computation is a Dataflow Graph

Graph of Nodes, also called Operations or ops.

- bias
- weights
- examples
- labels

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Computation is a Dataflow Graph

Edges are N-dimensional arrays: Tensors

bias
weights
inputs
targets

MatMul
Add
Relu
Xent

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Logistic Regression as Dataflow Graph

Nodes
Operations
ops

Edges are N-dimensional arrays: Tensors

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Computation is a Dataflow Graph

‘Biases’ is a variable

Some ops compute gradients

-= updates biases

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016
Data Flow Graph

Source: https://www.tensorflow.org/
Data Flow Graph
Python
Download Anaconda

Download Anaconda Now

Get Superpowers with Anaconda

Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages for data science.

Which version should I download and install?
With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.

https://www.continuum.io/downloads
Download Anaconda Python 3.6

Anaconda 4.3.1

For macOS

**macOS 10.12.2 users:** To prevent permissions problems, we recommend that you upgrade to macOS 10.12.3 or later before installing Anaconda.

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

**Changelog**

**Graphical Installer**

1. Download the graphical installer
2. Double-click the downloaded .pkg file and follow the instructions

**Command Line Installer**

1. Download the command-line installer
2. Optional: Verify data integrity with MD5 or SHA-256 [More info]
3. In your terminal window type one of the below and follow the instructions:
   - **Python 3.6 version**
   - **Python 2.7 version**

[https://www.continuum.io/downloads](https://www.continuum.io/downloads)
OS X Anaconda Python 3.6 Installation

Command Line Installer
Download the command-line installer

In your terminal window type one of the below and follow the instructions:

Python 3.6 version
bash Anaconda3-4.3.1-MacOSX-x86_64.sh

Python 2.7 version
bash Anaconda2-4.3.1-MacOSX-x86_64.sh

https://www.continuum.io/downloads
OS X Anaconda 3 - 4.3.1
Python 3.6 Installation
Anaconda3-4.3.1-MacOSX-x86_64.pkg
Install Anaconda 3

Welcome to the Anaconda3 Installer

You will be guided through the steps necessary to install this software.

- Introduction
  - Read Me
  - License
  - Destination Select
  - Installation Type
  - Installation
  - Summary

Go Back Continue
Install Anaconda 3

Important Information

Anaconda is a modern open source analytics platform powered by Python. See https://www.continuum.io/downloads/.

By default, this installer modifies your bash profile to put Anaconda in your PATH. To disable this, choose "Customize" at the "Installation Type" phase, and disable the "Modify PATH" option. If you do not do this, you will need to add ~/anaconda/bin to your PATH manually to run the commands, or run all anaconda commands explicitly from that path.

To install to a different location, select "Change Install Location..." at the "Installation Type" phase, the choose "Install on a specific disk...", choose the disk you wish to install on, and click "Choose Folder...". The "Install for me only" option will install anaconda to the default location, ~/anaconda.

The packages included in this installation are:
- alabaster 0.7.9
Install Anaconda 3

Software License Agreement

Copyright 2016, Continuum Analytics, Inc.

All rights reserved under the 3-clause BSD License:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
Install Anaconda 3

To continue installing the software you must agree to the terms of the software license agreement.

Click Agree to continue or click Disagree to cancel the installation and quit the Installer.

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by the copyright holders.
Install Anaconda 3

Standard Install on “Macintosh HD”

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.
Install Anaconda 3

Select a Destination

How do you want to install this software?

- Install for all users of this computer

- Install for me only

- Install on a specific disk...

Installing this software requires 1.4 GB of space.

You have chosen to install this software in your home folder. Only the current user will be able to use this software.
Install Anaconda 3

Standard Install on “Macintosh HD”

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.
Install Anaconda 3

Registering updated applications...

Install time remaining: About a minute
Install Anaconda 3

Anaconda is the leading open data science platform powered by Python.

Share your notebooks and packages on Anaconda Cloud! [Sign up for free]

178 python packages included.

Supported packages: 453

Source: https://docs.continuum.io/anaconda/pkg-docs
Install Anaconda 3

<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>headdict</td>
<td>1.0.0</td>
</tr>
<tr>
<td>icu</td>
<td>54.1</td>
</tr>
<tr>
<td>idna</td>
<td>2.2</td>
</tr>
<tr>
<td>imagesize</td>
<td>0.71</td>
</tr>
<tr>
<td>ipython</td>
<td>4.5.2</td>
</tr>
<tr>
<td>ipython_genutils</td>
<td>5.1.0</td>
</tr>
<tr>
<td>ipywidgets</td>
<td>0.1.0</td>
</tr>
<tr>
<td>isort</td>
<td>0.24</td>
</tr>
<tr>
<td>itsdangerous</td>
<td>0.3</td>
</tr>
<tr>
<td>jbig</td>
<td>2.1</td>
</tr>
<tr>
<td>jdkal</td>
<td>1.1</td>
</tr>
<tr>
<td>jedi</td>
<td>0.0.0</td>
</tr>
<tr>
<td>jinja2</td>
<td>2.9.4</td>
</tr>
<tr>
<td>jpeg</td>
<td>9.1</td>
</tr>
<tr>
<td>jsonschema</td>
<td>2.5.1</td>
</tr>
<tr>
<td>jupyter</td>
<td>1.0.0</td>
</tr>
<tr>
<td>jupyter-client</td>
<td>0.4.8</td>
</tr>
<tr>
<td>jupyter_console</td>
<td>5.0.0</td>
</tr>
<tr>
<td>jupyter_core</td>
<td>4.2.1</td>
</tr>
<tr>
<td>lazy-object-proxy</td>
<td>2.2.2</td>
</tr>
<tr>
<td>libiconv</td>
<td>1.14</td>
</tr>
<tr>
<td>libpng</td>
<td>1.6.27</td>
</tr>
<tr>
<td>libtiff</td>
<td>4.6.3</td>
</tr>
<tr>
<td>libxml2</td>
<td>2.0.4</td>
</tr>
<tr>
<td>libxslt</td>
<td>1.12</td>
</tr>
<tr>
<td>llvmlite</td>
<td>0.15.0</td>
</tr>
<tr>
<td>loki</td>
<td>1.28</td>
</tr>
<tr>
<td>lxml</td>
<td>3.7.2</td>
</tr>
<tr>
<td>lxml</td>
<td>0.23</td>
</tr>
<tr>
<td>matplotlib</td>
<td>2.0.0</td>
</tr>
<tr>
<td>mistune</td>
<td>0.7.3</td>
</tr>
<tr>
<td>mkl</td>
<td>2017.0.1</td>
</tr>
<tr>
<td>mkl-service</td>
<td>1.1.2</td>
</tr>
<tr>
<td>mpmath</td>
<td>0.19</td>
</tr>
<tr>
<td>multipledispatch</td>
<td>0.4.9</td>
</tr>
<tr>
<td>nbconvert</td>
<td>4.2.8</td>
</tr>
<tr>
<td>nbformat</td>
<td>4.2.8</td>
</tr>
<tr>
<td>networkx</td>
<td>1.11</td>
</tr>
<tr>
<td>nltk</td>
<td>3.2.2</td>
</tr>
<tr>
<td>notebook</td>
<td>1.37</td>
</tr>
<tr>
<td>numba</td>
<td>4.3.1</td>
</tr>
<tr>
<td>numpy</td>
<td>0.30.1</td>
</tr>
<tr>
<td>numpyexpr</td>
<td>2.6.1</td>
</tr>
<tr>
<td>numexpr</td>
<td>1.113</td>
</tr>
<tr>
<td>numpycall</td>
<td>0.6.8</td>
</tr>
<tr>
<td>numpydoc</td>
<td>0.5.0</td>
</tr>
<tr>
<td>odo</td>
<td>2.4.1</td>
</tr>
<tr>
<td>openpyxl</td>
<td>1.0.2k</td>
</tr>
<tr>
<td>openssl</td>
<td>0.19.2</td>
</tr>
<tr>
<td>pandas</td>
<td>150</td>
</tr>
<tr>
<td>partt</td>
<td>101</td>
</tr>
<tr>
<td>path.py</td>
<td>102</td>
</tr>
<tr>
<td>pathlib2</td>
<td>102</td>
</tr>
<tr>
<td>patsy</td>
<td>104</td>
</tr>
<tr>
<td>pep8</td>
<td>106</td>
</tr>
<tr>
<td>pexexpect</td>
<td>107</td>
</tr>
<tr>
<td>pickleshare</td>
<td>108</td>
</tr>
<tr>
<td>pillow</td>
<td>109</td>
</tr>
<tr>
<td>pip</td>
<td>110</td>
</tr>
<tr>
<td>plyn</td>
<td>111</td>
</tr>
<tr>
<td>prompt_toolkit</td>
<td>112</td>
</tr>
<tr>
<td>psutil</td>
<td>113</td>
</tr>
<tr>
<td>py</td>
<td>114</td>
</tr>
<tr>
<td>pyproject</td>
<td>115</td>
</tr>
<tr>
<td>pyrobject</td>
<td>116</td>
</tr>
<tr>
<td>pytables</td>
<td>117</td>
</tr>
<tr>
<td>pytest</td>
<td>118</td>
</tr>
<tr>
<td>python</td>
<td>119</td>
</tr>
<tr>
<td>python-dateutil</td>
<td>120</td>
</tr>
<tr>
<td>python.app</td>
<td>121</td>
</tr>
<tr>
<td>pytz</td>
<td>122</td>
</tr>
<tr>
<td>pyyaml</td>
<td>123</td>
</tr>
<tr>
<td>pyzmq</td>
<td>124</td>
</tr>
<tr>
<td>qt</td>
<td>125</td>
</tr>
<tr>
<td>qtaawesome</td>
<td>126</td>
</tr>
<tr>
<td>qtcconsole</td>
<td>127</td>
</tr>
<tr>
<td>qtpy</td>
<td>128</td>
</tr>
<tr>
<td>readline</td>
<td>129</td>
</tr>
<tr>
<td>redis</td>
<td>130</td>
</tr>
<tr>
<td>redis-py</td>
<td>131</td>
</tr>
<tr>
<td>requests</td>
<td>132</td>
</tr>
<tr>
<td>rope</td>
<td>133</td>
</tr>
<tr>
<td>ruamel_yaml</td>
<td>134</td>
</tr>
<tr>
<td>scikit-image</td>
<td>135</td>
</tr>
<tr>
<td>scikit-learn</td>
<td>136</td>
</tr>
<tr>
<td>scipy</td>
<td>137</td>
</tr>
<tr>
<td>seahorn</td>
<td>138</td>
</tr>
<tr>
<td>setuptools</td>
<td>139</td>
</tr>
<tr>
<td>simplegeneric</td>
<td>140</td>
</tr>
<tr>
<td>singledispatch</td>
<td>141</td>
</tr>
</tbody>
</table>

178 python packages included.
Anaconda-Navigator

Launchpad
Anaconda-Navigator

Thanks for installing Anaconda!

Anaconda Navigator helps you easily start important Python applications and manage the packages in your local Anaconda installation. It also connects you to online resources for learning and engaging with the Python, SciPy, and PyData community.

To help us improve Anaconda Navigator, fix bugs, and make it even easier for everyone to use Python, we gather anonymized usage information, just like most web browsers and mobile apps.

To opt out of this, please uncheck below (You can always change this setting in the Preferences menu).

- Yes, I’d like to help improve Anaconda.
- Ok, don’t show again

- jupyter notebook
- anaconda-fusion
- glueviz
- rstudio

- Launch
- Install

Web-based, interactive computing environment. Edit and run notebooks docs while describing the data.

Integration between Excel® and Anaconda via Notebooks. Run data science functions, interact with results and create advanced visualizations in a code-free app inside Excel.

Multidimensional data visualization across files. Explore relationships within and among related datasets.

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Jupyter Notebook

**jupyter notebook**

4.3.1

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch

**qtconsole**

4.2.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch

**spyder**

3.1.2

Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features.

Launch

**anaconda-fusion**

1.0.2

Integration between Excel® and Anaconda via Notebooks. Run data science functions, interact with results and create advanced visualizations in a code-free app inside Excel.

Install

**glueviz**

0.9.1

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install

**rstudio**

1.0.136

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Install
Jupyter Notebook
New Python 3
print("hello, world")
from platform import python_version
print("Python Version:", python_version())
Create Python Environments with Anaconda

• Python 3.6
• Python 3.5
  – Python 3.5.3
  – Python 3.5.2
• Python 2.7

https://conda.io/docs/py2or3.html
Anaconda Create New Python 3.5 Environment (py35)

Source: http://conda.pydata.org/docs/py2or3.html
Verify that conda is installed, check current conda version

• `conda --version`

• Update conda to the current version
  – `conda update conda`

http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version
Check current conda version
Check current python version
Check conda environments

• conda --version
• python --version
• conda info --envs

http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version
Terminal

terminal
```bash
$ conda list
# packages in environment at /Users/imyday/anaconda:

<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>_license</td>
<td>1.1</td>
<td>py36_1</td>
</tr>
<tr>
<td>alabaster</td>
<td>0.7.9</td>
<td>py36_0</td>
</tr>
<tr>
<td>anaconda</td>
<td>4.3.1</td>
<td>np111py36_0</td>
</tr>
<tr>
<td>anaconda-client</td>
<td>1.6.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>anaconda-navigator</td>
<td>1.5.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>anaconda-project</td>
<td>0.4.1</td>
<td>py36_0</td>
</tr>
<tr>
<td>appnope</td>
<td>0.1.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>appscript</td>
<td>1.0.1</td>
<td>py36_0</td>
</tr>
<tr>
<td>astroid</td>
<td>1.4.9</td>
<td>py36_0</td>
</tr>
<tr>
<td>astropy</td>
<td>1.3</td>
<td>np111py36_0</td>
</tr>
<tr>
<td>babel</td>
<td>2.3.4</td>
<td>py36_0</td>
</tr>
<tr>
<td>backports</td>
<td>1.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>beautifulsoup4</td>
<td>4.5.3</td>
<td>py36_0</td>
</tr>
<tr>
<td>bitarray</td>
<td>0.8.1</td>
<td>py36_0</td>
</tr>
<tr>
<td>blaze</td>
<td>0.10.1</td>
<td>py36_0</td>
</tr>
<tr>
<td>bokeh</td>
<td>0.12.4</td>
<td>py36_0</td>
</tr>
<tr>
<td>boto</td>
<td>2.45.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>bottleneck</td>
<td>1.2.0</td>
<td>np111py36_0</td>
</tr>
<tr>
<td>cffi</td>
<td>1.9.1</td>
<td>py36_0</td>
</tr>
<tr>
<td>chardet</td>
<td>2.3.0</td>
<td>py36_0</td>
</tr>
<tr>
<td>chest</td>
<td>0.2.3</td>
<td>py36_0</td>
</tr>
</tbody>
</table>
```

[http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version](http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version)
python --version

```
[iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
```
conda --version

```
[imyday-MacBook-Pro:~ imyday]$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
[imyday-MacBook-Pro:~ imyday]$ conda --version
conda 4.3.14
[imyday-MacBook-Pro:~ imyday]$ conda info --envs
# conda environments:
#
py27    /Users/imyday/anaconda/envs/py27
py35    /Users/imyday/anaconda/envs/py35
root    * /Users/imyday/anaconda

[imyday-MacBook-Pro:~ imyday]$ source activate py35
(py35) [imyday-MacBook-Pro:~ imyday]$ python --version
Python 3.5.3 :: Continuum Analytics, Inc.
(py35) [imyday-MacBook-Pro:~ imyday]$ conda --version
conda 4.3.14
(py35) [imyday-MacBook-Pro:~ imyday]$ source deactivate py35
[imyday-MacBook-Pro:~ imyday]$ conda info --envs
# conda environments:
#
py27    /Users/imyday/anaconda/envs/py27
py35    /Users/imyday/anaconda/envs/py35
root    * /Users/imyday/anaconda
```
conda create -n py352 python=3.5.2 anaconda

Create a Python 3.5.2 environment
conda create -n py352 python=3.5.2 anaconda

# To activate this environment, use:
# > source activate py352
#
# To deactivate this environment, use:
# > source deactivate py352

Source: http://conda.pydata.org/docs/py2or3.html
conda info --envs

# conda environments:
#
py27       /Users/imyday/anaconda/envs/py27
py35       /Users/imyday/anaconda/envs/py35
py352      /Users/imyday/anaconda/envs/py352
root       * /Users/imyday/anaconda

python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)

source activate py352
(py352) python --version
Python 3.5.2 :: Anaconda 4.3.1 (x86_64)
TensorFlow
conda info --envs
conda --version
python --version
conda list
conda create -n tensorflow python=3.5
source activate tensorflow
activate tensorflow
sudo pip install tensorflow
pip install tensorflow
sudo pip install keras
pip install keras
pip install tensorflow
jupyter notebook

Source: https://github.com/martin-gorner/tensorflow-mnist-tutorial/blob/master/INSTALL.txt
Source: http://deeplearning.net/software/theano/install.html#anaconda
pip install tensorflow

bash-3.2$ pip install tensorflow
Collecting tensorflow
  Downloading tensorflow-1.1.0-cp36-cp36m-macosx_10_11_x86_64.whl (31.3MB)
    100% [===============================================] 31.3MB 23kB/s
Requirement already satisfied: wheel>=0.26 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: six>=1.10.0 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Collecting protobuf>=3.2.0 (from tensorflow)
  Downloading protobuf-3.2.0-py2.py3-none-any.whl (360kB)
    100% [===============================================] 368kB 453kB/s
Requirement already satisfied: werkzeug>=0.11.10 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: numpy>=1.11.0 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: setuptools in ./anaconda/lib/python3.6/site-packages/setuptools-27.2.0-py3.6.egg (from protobuf>=3.2.0->tensorflow)
Installing collected packages: protobuf, tensorflow
Successfully installed protobuf-3.2.0 tensorflow-1.1.0
bash-3.2$
TensorFlow Playground

Tinker With a **Neural Network** Right Here in Your Browser. Don’t Worry, You Can’t Break It. We Promise.

http://playground.tensorflow.org/
TensorBoard

Main Graph

Auxiliary nodes

https://www.tensorflow.org/tensorboard/index.html#graphs
Getting Started With TensorFlow

This guide gets you started programming in TensorFlow. Before using this guide, install TensorFlow. To get the most out of this guide, you should know the following:

- How to program in Python.
- At least a little bit about arrays.
- Ideally, something about machine learning. However, if you know little or nothing about machine learning, then this is still the first guide you should read.

TensorFlow provides multiple APIs. The lowest level API—TensorFlow Core—provides you with complete programming control. We recommend TensorFlow Core for machine learning researchers and others who require fine levels of control over their models. The higher level APIs are built on top of TensorFlow Core. These higher level APIs are typically easier to learn and use than TensorFlow Core. In addition, the higher level APIs make repetitive tasks easier and more consistent between different users. A high-level API like tf.estimator helps you manage data sets, estimators, training and inference.

This guide begins with a tutorial on TensorFlow Core. Later, we demonstrate how to implement the same model in tf.estimator. Knowing TensorFlow Core principles will give you a great mental model of how things are working internally when you use the more compact higher level API.
Try your first TensorFlow

$ python

```python
>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!')
>>> sess = tf.Session()
>>> sess.run(hello)
'Hello, TensorFlow!'
>>> a = tf.constant(10)
>>> b = tf.constant(32)
>>> sess.run(a+b)
42
```
Hello TensorFlow

```python
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
sess.run(hello)

b'Hello, TensorFlow!'
```

https://github.com/tensorflow/tensorflow
```
import tensorflow as tf
sess = tf.Session()
a = tf.constant(10)
b = tf.constant(32)
sess.run(a+b)
```

42

https://github.com/tensorflow/tensorflow
```python
import tensorflow as tf

# Model parameters
W = tf.Variable([.3], dtype=tf.float32)
b = tf.Variable([-3], dtype=tf.float32)
# Model input and output
x = tf.placeholder(tf.float32)
linear_model = W*x + b
y = tf.placeholder(tf.float32)

# loss
loss = tf.reduce_sum(tf.square(linear_model - y))  # sum of the squares
# optimizer
optimizer = tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)

# training data
x_train = [1, 2, 3, 4]
y_train = [0, -1, -2, -3]
# training loop
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)  # reset values to wrong
for i in range(1000):
    sess.run(train, {x: x_train, y: y_train})

# evaluate training accuracy
curr_W, curr_b, curr_loss = sess.run([W, b, loss], {x: x_train, y: y_train})
print("W: \$s b: \$s loss: \$s"%(curr_W, curr_b, curr_loss))

W: [-0.9999969] b: [ 0.99999082] loss: 5.69997e-11
```

Source: [https://www.tensorflow.org/get_started/get_started](https://www.tensorflow.org/get_started/get_started)
```python
import numpy as np
import tensorflow as tf

feature_columns = [tf.feature_column.numeric_column("x", shape=[1])]

estimator = tf.estimator.LinearRegressor(feature_columns=feature_columns)

x_train = np.array([1., 2., 3., 4.])
y_train = np.array([0., -1., -2., -3.])
x_eval = np.array([2., 5., 8., 1.])
y_eval = np.array([-1.01, -4.1, -7, 0.])

input_fn = tf.estimator.inputs.numpy_input_fn(
    {'x': x_train}, y_train, batch_size=4, num_epochs=None, shuffle=True)
train_input_fn = tf.estimator.inputs.numpy_input_fn(
    {'x': x_train}, y_train, batch_size=4, num_epochs=1000, shuffle=False)
eval_input_fn = tf.estimator.inputs.numpy_input_fn(
    {'x': x_eval}, y_eval, batch_size=4, num_epochs=1000, shuffle=False)

estimator.train(input_fn=input_fn, steps=1000)

train_metrics = estimator.evaluate(input_fn=train_input_fn)
eval_metrics = estimator.evaluate(input_fn=eval_input_fn)
print("train metrics: \%r" % train_metrics)
print("eval metrics: \%r" % eval_metrics)
```

train metrics: {'average_loss': 2.7210228e-07, 'loss': 1.0884091e-06, 'global_step': 1000}
eval metrics: {'average_loss': 0.0025725411, 'loss': 0.010290165, 'global_step': 1000}
TensorFlow and Deep Learning
1. Overview

> TensorFlow and deep learning_ without a PhD

In this codelab, you will learn how to build and train a neural network that recognises handwritten digits. Along the way, as you enhance your neural network to achieve 99% accuracy, you will also discover the tools of the trade that deep learning professionals use to train their models efficiently.

This codelab uses the MNIST dataset, a collection of 60,000 labeled digits that has kept generations of PhDs busy for almost two decades. You will solve the problem with less than 100 lines of Python / TensorFlow code.

What you'll learn

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0
Sample code for "Tensorflow and deep learning, without a PhD" presentation and code lab.

```
mlengine
.gitignore
CONTRIBUTING.md
INSTALL.txt
LICENSE
README.md
mnist_1.0_softmax.py
mnist_2.0_five_layers_sigmoid.py
mnist_2.1_five_layers_relu_lrdcay...
```

mlengine: added example using the Tensorflow high level layers API
.gitignore: small bug fix in batch norm
CONTRIBUTING.md: initial commit 2
INSTALL.txt: Update INSTALL.txt
LICENSE: Initial commit
README.md: better image URL
mnist_1.0_softmax.py: global_variables_initializer used everywhere instead of inirialize_al...
mnist_2.0_five_layers_sigmoid.py: Fix spacing in the network structure comment
mnist_2.1_five_layers_relu_lrdcay...: Fix spacing in the network structure comment

Latest commit ed331aa 25 days ago

https://github.com/martin-gorner/tensorflow-mnist-tutorial/
TensorFlow and Deep Learning

• What is a neural network and how to train it
• How to build a basic 1-layer neural network using TensorFlow
• How to add more layers
• Training tips and tricks: overfitting, dropout, learning rate decay ...
• How to troubleshoot deep neural networks
• How to build convolutional networks

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0
git clone https://github.com/martin-gorner/tensorflow-mnist-tutorial.git

cd tensorflow-mnist-tutorial

python3 mnist_1.0_softmax.py

python mnist_1.0_softmax.py

pythonw mnist_1.0_softmax.py
MNIST dataset:
60,000 labeled digits

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0
Train a Neural Network

Training digits updates to weights and biases => better recognition (loop)

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Training digits

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Training digits
Test digits

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Cross entropy loss

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Accuracy

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Biases

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Weights and Biases

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#2
Softmax
Cross-entropy
Mini-batch
Very Simple Model: Softmax Classification

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Very Simple Model: Softmax Classification

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Very Simple Model: Softmax Classification

\[
\text{softmax}(L_n) = \frac{e^{L_n}}{\| e^L \|}
\]

weighted sum of all pixels + bias

neuron outputs

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
In Matrix notation, 100 images at a time

$X$: 100 images, one per line, flattened

$W$: 10 columns, 784 lines

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
X: 100 images, one per line, flattened
X: 100 images, one per line, flattened
X: 100 images, one per line, flattened

<table>
<thead>
<tr>
<th>W₀,0</th>
<th>W₀,1</th>
<th>W₀,2</th>
<th>W₀,3</th>
<th>...</th>
<th>W₀,9</th>
</tr>
</thead>
<tbody>
<tr>
<td>W₁,0</td>
<td>W₁,1</td>
<td>W₁,2</td>
<td>W₁,3</td>
<td>...</td>
<td>W₁,9</td>
</tr>
<tr>
<td>W₂,0</td>
<td>W₂,1</td>
<td>W₂,2</td>
<td>W₂,3</td>
<td>...</td>
<td>W₂,9</td>
</tr>
<tr>
<td>W₃,0</td>
<td>W₃,1</td>
<td>W₃,2</td>
<td>W₃,3</td>
<td>...</td>
<td>W₃,9</td>
</tr>
<tr>
<td>W₄,0</td>
<td>W₄,1</td>
<td>W₄,2</td>
<td>W₄,3</td>
<td>...</td>
<td>W₄,9</td>
</tr>
<tr>
<td>W₅,0</td>
<td>W₅,1</td>
<td>W₅,2</td>
<td>W₅,3</td>
<td>...</td>
<td>W₅,9</td>
</tr>
<tr>
<td>W₆,0</td>
<td>W₆,1</td>
<td>W₆,2</td>
<td>W₆,3</td>
<td>...</td>
<td>W₆,9</td>
</tr>
<tr>
<td>W₇,0</td>
<td>W₇,1</td>
<td>W₇,2</td>
<td>W₇,3</td>
<td>...</td>
<td>W₇,9</td>
</tr>
<tr>
<td>W₈,0</td>
<td>W₈,1</td>
<td>W₈,2</td>
<td>W₈,3</td>
<td>...</td>
<td>W₈,9</td>
</tr>
</tbody>
</table>

...  

<table>
<thead>
<tr>
<th>L₀,0</th>
<th>L₀,1</th>
<th>L₀,2</th>
<th>L₀,3</th>
<th>...</th>
<th>L₀,9</th>
</tr>
</thead>
</table>

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
X: 100 images, one per line, flattened

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
What are "weights" and "biases"?

How is the "cross-entropy" computed?

How exactly does the training algorithm work?
\[ Y = f(X) \]

\[ Y = \text{softmax}(X \cdot W + b) \]

Predictions  
\[ Y[100, 10] \]

Images  
\[ X[100, 784] \]

Weights  
\[ W[784, 10] \]

Biases  
\[ b[10] \]

applied line by line

matrix multiply

broadcast on all lines

tensor shapes in [ ]

\[ Y = \text{tf.nn.softmax(tf.matmul(X, W) + b)} \]
TensorFlow (Python) Softmax

Predictions:
\[ Y[100, 10] \]

Tensor shapes:
\[ X[100, 784], \ W[784, 10], \ b[10] \]

\[ Y = \text{tf.nn.softmax}(\text{tf.matmul}(X, W) + b) \]

- Matrix multiply
- Broadcast on all lines

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Cross Entropy

Cross entropy: \[-\sum Y'_i \cdot \log(Y_i)\]

actual probabilities, "one-hot" encoded

computed probabilities

this is a "6"

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Minimizing Cross Entropy (Minimizing Loss)

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Training Loop

Training digits and labels

=> loss function

=> gradient (partial derivatives)

=> steepest descent

=> update weights and biases

=> repeat with next mini-batch of training images and labels

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
"mini-batches":
100 images and labels
import tensorflow as tf
```python
import tensorflow as tf
X = tf.placeholder(tf.float32, [None, 28, 28, 1])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))

init = tf.initialize_all_variables()
```

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#5
# model
Y = tf.nn.softmax(tf.matmul(tf.reshape(X, [-1, 784]), W) + b)

# placeholder for correct labels
Y_ = tf.placeholder(tf.float32, [None, 10])

# loss function
cross_entropy = -tf.reduce_sum(Y_ * tf.log(Y))

# % of correct answers found in batch
is_correct = tf.equal(tf.argmax(Y, 1), tf.argmax(Y_, 1))
accuracy = tf.reduce_mean(tf.cast(is_correct, tf.float32))
sess = tf.Session()
sess.run(init)

for i in range(1000):
    # load batch of images and correct answers
    batch_X, batch_Y = mnist.train.next_batch(100)
    train_data={X: batch_X, Y_: batch_Y}

    # train
    sess.run(train_step, feed_dict=train_data)
# success ?
a,c = sess.run([accuracy, cross_entropy],
             feed_dict=train_data)

# success on test data ?
test_data={X: mnist.test.images, Y_: mnist.test.labels}
a,c = sess.run([accuracy, cross_entropy], feed=test_data)
mnist_1.0_softmax.py

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#5
Deep Learning

Go deep!
5 fully-connected layers

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Sigmoid

\[
\frac{1}{1 + e^{-x}}
\]

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
5 fully-connected layers
TensorFlow MNIST Tutorial
ReLU
TensorFlow MNIST Tutorial
ReLU

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial

![Accuracy Chart](https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/)

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Learning Rate

Slow down...

Learning rate decay
LR = 0.003

TensorFlow MNIST Tutorial

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial

Accuracy

Cross entropy loss

Training digits

Test digits

Weights

Biases

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Dropout
Overfitting

Accuracy

Cross entropy loss

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Dropout

Training
pkeep = 0.75

Test
pkeep = 1.0

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial

**Accuracy**

- **Training accuracy**
- **Test accuracy**

**Cross entropy loss**

- **Training loss**
- **Test loss**

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial

Accuracy

Cross entropy loss

5 layers Sigmoid

97.9%

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Overfitting

Overfitting ?!

Too many neurons

BAD Network

Not enough DATA

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Convolutional Layer

$W_{[4, 4, 3]}$
Convolutional Layer

\[ W_{1}[4, 4, 3] \]

\[ W_{2}[4, 4, 3] \]

\[ W[4, 4, 3, 2] \]

filter size

input channels

output channels

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Convolutional Max-Pool

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
All Convolutional

Hacker's tip

ALL Convolutional

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Bigger Convolutional Neural Network

28x28x1

28x28x4

14x14x8

7x7x12

200

10

+ biases on all layers

convolutional layer, 4 channels
W1[5, 5, 1, 4] stride 1

convolutional layer, 8 channels
W2[4, 4, 4, 8] stride 2

convolutional layer, 12 channels
W3[4, 4, 8, 12] stride 2

fully connected layer
W4[7x7x12, 200]

softmax readout layer
W5[200, 10]
Bigger Convolutional Neural Network

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
Bigger Convolutional Neural Network + Dropout

- convolutional layer, 6 channels
  $W_1[6, 6, 1, 6]$ stride 1

- convolutional layer, 12 channels
  $W_2[5, 5, 6, 12]$ stride 2

- convolutional layer, 24 channels
  $W_3[4, 4, 12, 24]$ stride 2

- fully connected layer
  $W_4[7 \times 7 \times 24, 200]$,

- softmax readout layer
  $W_5[200, 10]$

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial
TensorFlow MNIST Tutorial

Accuracy

Cross entropy loss

larger convolutional network

Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
TensorFlow MNIST Tutorial

Source: https://github.com/martin-gorner/tensorflow-mnist-tutorial/
TensorFlow MNIST Tutorial

python3 mnist_1.0_softmax.py
python mnist_1.0_softmax.py
pythonw mnist_1.0_softmax.py

python3 mnist_2.0_five_layers_sigmoid.py
python3 mnist_2.2_five_layers_relu_lrdecay_dropout.py
python3 mnist_3.0_convolutional.py
python3 mnist_3.1_convolutional_bigger_dropout.py
python3 mnist_4.0_batchnorm_five_layers_sigmoid.py
python3 mnist_4.1_batchnorm_five_layers_relu.py
python3 mnist_4.2_batchnorm_convolutional.py

Source: https://github.com/martin-gorner/tensorflow-mnist-tutorial/
Source: https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/
References

- Martin Gorner (2017), TensorFlow and Deep Learning without a PhD, Part 2 (Google Cloud Next '17), https://www.youtube.com/watch?v=fTUwdXUFFl8
- Deep Learning Basics: Neural Networks Demystified, https://www.youtube.com/playlist?list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
- Deep Learning SIMPLIFIED, https://www.youtube.com/playlist?list=PLjJh1vlSEYgvGod9wWiydumYL8hOxiNu
- 3Blue1Brown (2017), But what *is* a Neural Network? | Chapter 1, deep learning, https://www.youtube.com/watch?v=aircAruvnKk
- 3Blue1Brown (2017), Gradient descent, how neural networks learn | Chapter 2, deep learning, https://www.youtube.com/watch?v=IHZwWFHwa-w
- 3Blue1Brown (2017), What is backpropagation really doing? | Chapter 3, deep learning, https://www.youtube.com/watch?v=Ilg3gGewQ5U
- TensorFlow: https://www.tensorflow.org/
- Keras: http://keras.io/
- Natural Language Processing with Deep Learning (Winter 2017), https://www.youtube.com/playlist?list=PL3FW7Lu3i5Jsnh1rnUwq_TcylNr7EkRe6
- Udacity, Deep Learning, https://www.youtube.com/playlist?list=PLAwxTw4SYaPn_OWPFT9ulXLuQrlmzHfOV
- https://github.com/leriomaggio/deep-learning-keras-tensorflow
深度學習視覺辨識
(Deep Learning for Visual Recognition)

Time: 2018/01/25 (Thu) (9:00 -12:00, 13:00-16:00)
Place: 國立臺北護理健康大學 (台北市明德路365號) 親仁樓 B112
Host: 祝國忠 院長 (健康科技學院院長)

Min-Yuh Day
戴敏育
Assistant Professor
專任助理教授

Dept. of Information Management, Tamkang University
淡江大學 資訊管理學系

http://mail.tku.edu.tw/myday/
2018-01-25