Chapter 1: Getting Started with Deep Learning
(a) Gradient Descent - GD

(b) Stochastic Gradient Descent - SGD

Input Layer

Hidden Layer

Output Layer
Chapter 2: First Look at TensorFlow

<table>
<thead>
<tr>
<th>Select Target Platform</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Click on the green buttons that describe your target platform. Only supported platforms will be shown.</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows</td>
</tr>
<tr>
<td>Architecture</td>
<td>x86_64</td>
</tr>
<tr>
<td>Distribution</td>
<td>Fedora</td>
</tr>
<tr>
<td>Version</td>
<td>16.04</td>
</tr>
<tr>
<td>Installer Type</td>
<td>runfile (local)</td>
</tr>
</tbody>
</table>
NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

I Agree To the Terms of the cuDNN Software License Agreement
Please check your framework documentation to determine the recommended version of cuDNN.
If you are using cuDNN with a Pascal (GTX 1080, GTX 1070), version 5 or later is required.

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 8.0

- cuDNN User Guide
- cuDNN Install Guide
- cuDNN v5.1 Library for Linux
- cuDNN v5.1 Library for Power8
- cuDNN v5.1 Library for Windows 7
- cuDNN v5.1 Library for Windows 10
- cuDNN v5.1 Library for OSX
- cuDNN v5.1 Release Notes
- cuDNN v5.1 Runtime Library for Ubuntu14.04 (Deb)
- cuDNN v5.1 Developer Library for Ubuntu14.04 (Deb)
- cuDNN v5.1 Code Samples and User Guide (Deb)
- cuDNN v5.1 Runtime Library for Ubuntu16.04 Power8 (Deb)
- cuDNN v5.1 Developer Library for Ubuntu16.04 Power8 (Deb)
- cuDNN v5.1 Code Samples and User Guide Power8 (Deb)

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 7.5
weight \rightarrow f \rightarrow output

output = f (input, weight)
Renamed function `tf.initialize_all_variables` to `tf.global_variables_initializer`
Old: sess.run(tf.initialize_all_variables())
New: sess.run(tf.global_variables_initializer())

Renamed function `tf.train.SummaryWriter` to `tf.summary.FileWriter`
Old: writer = tf.train.SummaryWriter(logs_path, 
New: writer = tf.summary.FileWriter(logs_path, 

Added keyword 'logits' to reordered function `tf.nn.softmax_cross_entropy_with_logits`
Added keyword 'labels' to reordered function `tf.nn.softmax_cross_entropy_with_logits`
Old: cross_entropy = tf.nn.softmax_cross_entropy_with_logits(Ylogits, Y_)
New: cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=Ylogits, labels=Y_)

Renamed function `tf.scalar_summary` to `tf.summary.scalar`
Old: tf.scalar_summary("cost", cross_entropy)
New: tf.summary.scalar("cost", cross_entropy)

Renamed function `tf.scalar_summary` to `tf.summary.scalar`
Old: tf.scalar_summary("accuracy", accuracy)
New: tf.summary.scalar("accuracy", accuracy)

Renamed function `tf.merge_all_summaries` to `tf.summary.merge_all`
Old: summary_op = tf.merge_all_summaries()
New: summary_op = tf.summary.merge_all()
Chapter 3: Using TensorFlow on a Feed-Forward Neural Network
\[ \text{net}_i = \sum_j w_{ij} x_j \]  \hspace{1cm} (a)

\[ \text{net}_i = \sum_j w_{ij} x_j + b_i \]  \hspace{1cm} (b)

\[ \text{out}_i = \frac{1}{1 + e^{-\text{net}_i}} \]
\[ \text{out}_i = \frac{e^{net_i}}{\sum_{j=1}^{N} e^{net_j}} \]

\[ 0 \leq \text{out}_i \leq 1 \quad \text{con} \quad \sum_i \text{out}_i = 1 \]
Chapter 4: TensorFlow on a Convolutional Neural Network
Chapter 5: Optimizing TensorFlow
Autoencoders
Chapter 6: Recurrent Neural Networks
\[ S_t = f(U \cdot X_t + W \cdot S_{t-1}) \]

\[ O_t = V \cdot S_t \]
Chapter 7: GPU Computing
cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

I Agree To the Terms of the cuDNN Software License Agreement

Please check your framework documentation to determine the recommended version of cuDNN. If you are using cuDNN with a Pascal (GTX 1080, GTX 1070), version 5 or later is required.

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 8.0

cuDNN User Guide
cuDNN Install Guide
cuDNN v5.1 Library for Linux
cuDNN v5.1 Library for Power8
cuDNN v5.1 Library for Windows 7
cuDNN v5.1 Library for Windows 10
cuDNN v5.1 Library for OSX
cuDNN v5.1 Release Notes
cuDNN v5.1 Runtime Library for Ubuntu14.04 (Deb)
cuDNN v5.1 Developer Library for Ubuntu14.04 (Deb)
cuDNN v5.1 Code Samples and User Guide (Deb)
cuDNN v5.1 Runtime Library for Ubuntu16.04 Power8 (Deb)
cuDNN v5.1 Developer Library for Ubuntu16.04 Power8 (Deb)
cuDNN v5.1 Code Samples and User Guide Power8 (Deb)

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 7.5

Download cuDNN v5 (May 27, 2016), for CUDA 8.0

Download cuDNN v5 (May 12, 2016), for CUDA 7.5
Chapter 8: Advanced TensorFlow Programming

Keras: Deep Learning library for Theano and TensorFlow

You have just found Keras.

Keras is a high-level neural networks library, written in Python and capable of running on top of either TensorFlow 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 total modularity, minimalism, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Supports arbitrary connectivity schemes (including multi-input and multi-output training).
- Runs seamlessly on CPU and GPU.

Read the documentation at Keras.io.

Keras is compatible with Python 2.7-3.5.
Chapter 9: Advanced Multimedia Programming with TensorFlow
TF graphs go in,

```
0x00000000  movq   (%rdx), %rax
0x00000003  vmovaps (%rax), %xmm0
0x00000007  vmulps %xmm0, %xmm0, %xmm0
0x0000000b  vmovaps %xmm0, (%rdi)
...
```

Optimized & specialized assembly comes out.
TF-Level Block Diagram

TensorFlow

Existing TensorFlow Core

TF CPU Ops | TF GPU Ops | TF TPU Ops | XLA

XLA:CPU | XLA:GPU | XLA:TPU

XLA HLO

Target-independent Optimizations & Analyses

XLA HLO

Target-dependent Optimizations & Analyses

Target-specific Code Generation

XLA Backend
Chapter 10: Reinforcement Learning
Making new env: FrozenLake-v0
Score over time: 0.3585
Final Q-Table Values

\[
\begin{bmatrix}
4.9034838e-03 & 1.2373352e-02 & 5.04857351e-01 & 1.18572787e-02 \\
6.14009765e-04 & 1.34354386e-03 & 1.39327124e-03 & 5.88345699e-01 \\
2.42003179e-03 & 2.53712381e-03 & 1.27103632e-03 & 3.36417875e-01 \\
1.60332674e-03 & 6.60331077e-04 & 6.50987843e-04 & 1.96388199e-01 \\
6.38172447e-01 & 1.23434831e-03 & 1.35672865e-03 & 8.99709408e-05 \\
0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\
1.78445198e-01 & 1.27421388e-04 & 2.70432817e-05 & 7.55201005e-12 \\
0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\
5.85462465e-05 & 1.52400799e-03 & 6.22678642e-05 & 3.00741087e-01 \\
3.15488045e-03 & 6.66874039e-02 & 0.00000000e+00 & 4.21513681e-04 \\
0.00000000e+00 & 0.60000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\
0.00000000e+00 & 0.60000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\
1.20525081e-04 & 0.00000000e+00 & 9.20956992e-01 & 0.00000000e+00 \\
0.00000000e+00 & 0.00000000e+00 & 9.91561828e-01 & 0.00000000e+00 \\
0.00000000e+00 & 0.60000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\
\end{bmatrix}
\]

\[
\text{loss} = \sum (Q \text{-target} - Q)^2
\]