



クラウド・テクノロジー活用部会内勉強会
TensorFlowを触ってみよう

手抜きバージョン

2016年03月14日

先端IT活用推進コンソーシアム
クラウド・テクノロジー活用部会
リーダー 荒本道隆

TensorFlowとは

- 『TensorFlow紹介文の適当和訳 ななめ読み用』参照

http://qiita.com/tomo_makes/items/af23c1ac0d94b764da55



TensorFlowとは

TensorFlowはデータフローグラフを使った数値計算のためのオープンソースライブラリです。ノードは数学的操作を、エッジはノード間でやりとりされる多次元データ配列（テンソル tensors）を示します。柔軟なアーキテクチャにより、単一APIで、数値計算をデスクトップでも、サーバでも、モバイル端末でも、CPUやGPUの数を問わず、その上で実行できます。TensorFlowはもともと Google Machine Intelligence research organizationのGoogle Brain Teamの研究者やエンジニアにより、機械学習と深層ニューラルネットワーク研究のため作られました。しかし、他の領域にも適用できるよう、十分に一般化されたものです。

データフローグラフとは

データフローグラフは、数値計算を、有向グラフのノードとエッジを使って表します。ノードは通常数学的操作を表しますが、データの取込み、結果の出力、値の書込みと永続化も表せます。エッジはノード間の入出力関係を表します。こうしたデータエッジが、動的なサイズを持つ、多次元データ配列（テンソル）を表します。TensorFlowの名前の由来は、このグラフ内を流れるテンソルのフローです。ノードは、入力エッジが運ぶテンソルが全て出揃うと、各演算デバイスに割り当てられ、非同期・並列で実行されます。

人気の投稿

- 1. Cordova(PhoneGap)準備、iOSエミュレータ起動まで - 新年だし、ハイブリッドアプリ開発環境を準備する
- 5. IonicでAndroid開発準備から.apkパブリッシュまで - 新年だし、ハイブリッドスマートフォンアプリ開発
- 2. Ionic Framework準備、iOSエミュレータまで - 新年だし、Mac OSXでハイブリッドアプリ開発環境を準備する
- 3. Ionic Creator Beta入門編 - 新年だし、ハイブリッドスマートフォンアプリ開発環境を準備する
- 4. Ionicでアイコン、スプラッシュスクリーン編 - 新年だし、ハイブリッドスマートフォンアプリ開発

TensorFlowとは

データフローグラフとは

TensorFlowの特徴

TensorFlowは誰が使えるか

なぜGoogleはTensorFlowをオープンソース化したか

以下参考

参考資料

- TensorFlow勉強会(1)
 - <http://connpass.com/event/23531/>
 - オススメ動画：怪しみながら使うTensorFlow（46分）
- 丸レク
 - <http://www.slideshare.net/maruyama097/neural-network-tensorflow>
- TensorFlowを算数で理解する
 - <http://qiita.com/icoxfog417/items/fb5c24e35a849f8e2c5d>
- 特にプログラマーでもデータサイエンティストでもないけど、Tensorflowを1ヶ月触ったので超分かりやすく解説
 - <http://qiita.com/tawago/items/c977c79b76c5979874e8>

『怪しみながら使うTensorFlow』ダイジェスト

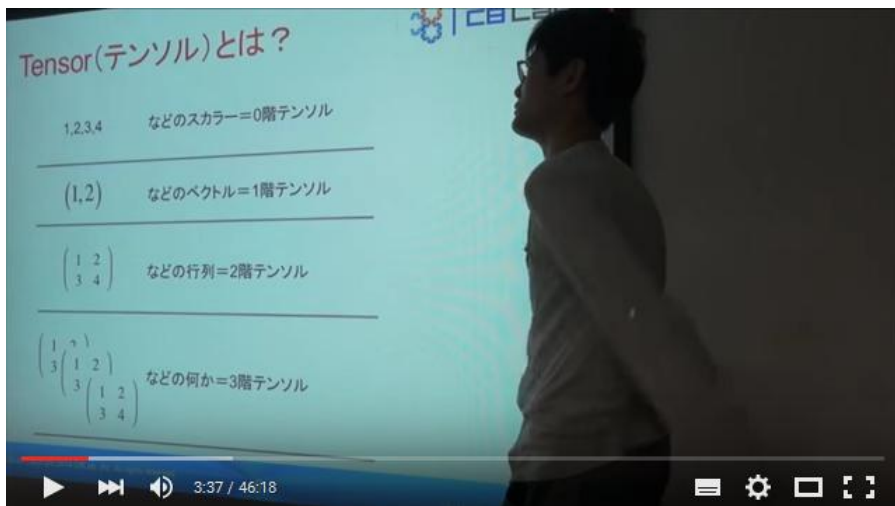
Tensor(テンソル)とは?

1,2,3,4 などのスカラー=0階テンソル

(1,2) などのベクトル=1階テンソル

$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ などの行列=2階テンソル

$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 3 & 1 & 2 \\ 3 & 4 \end{pmatrix}$ などの何か=3階テンソル



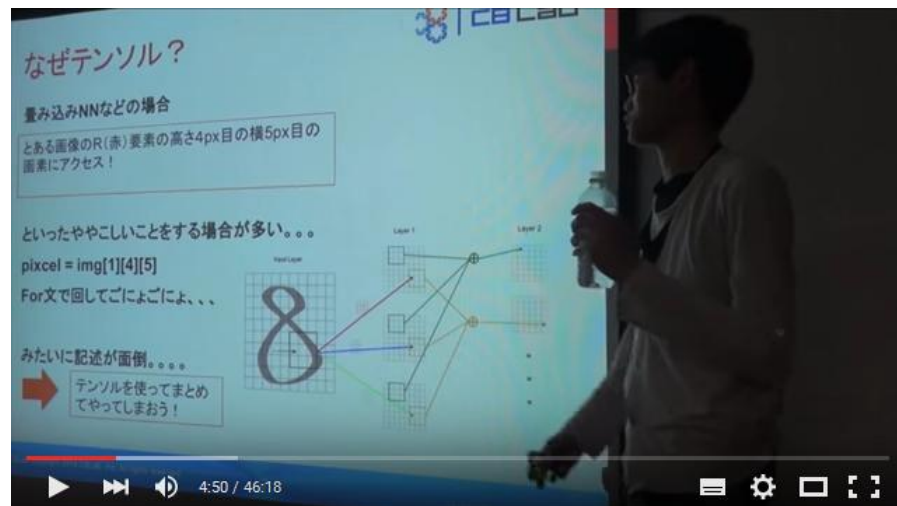
3:37 / 46:18

なぜテンソル?

畳み込みNNなどの場合
とある画像のR(赤)要素の高さ4px目の横5px目の
要素にアクセス!

といったややこしいことをする機会が多い。。。
pixel = img[1][4][5]
For文で回してごよごよによ、...

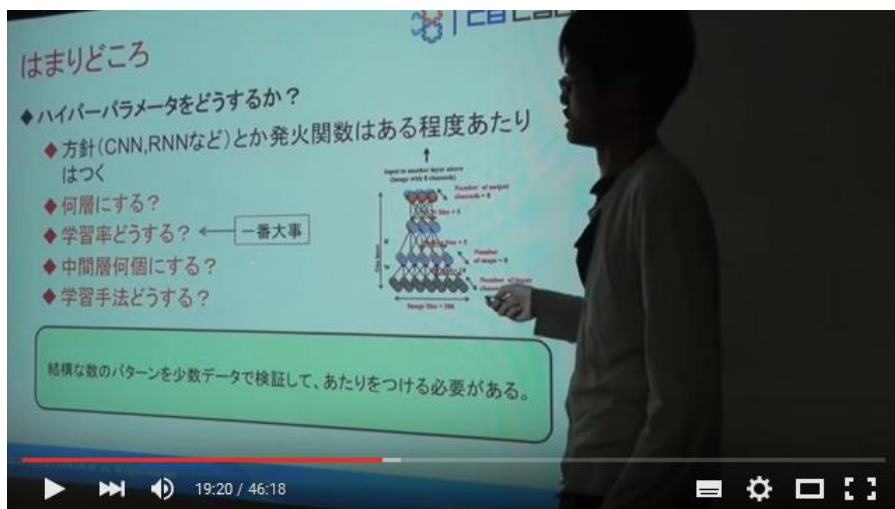
みたいに記述が面倒。。。
→ テンソルを使ってまとめてやってみよう!



4:50 / 46:18

はまりどころ

- ◆ ハイパーパラメータをどうするか?
 - ◆ 方針(CNN,RNNなど)とか発火関数はある程度あたりはつく
 - ◆ 何層にする?
 - ◆ 学習率どうする? ← 一番大事
 - ◆ 中間層何個にする?
 - ◆ 学習手法どうする?

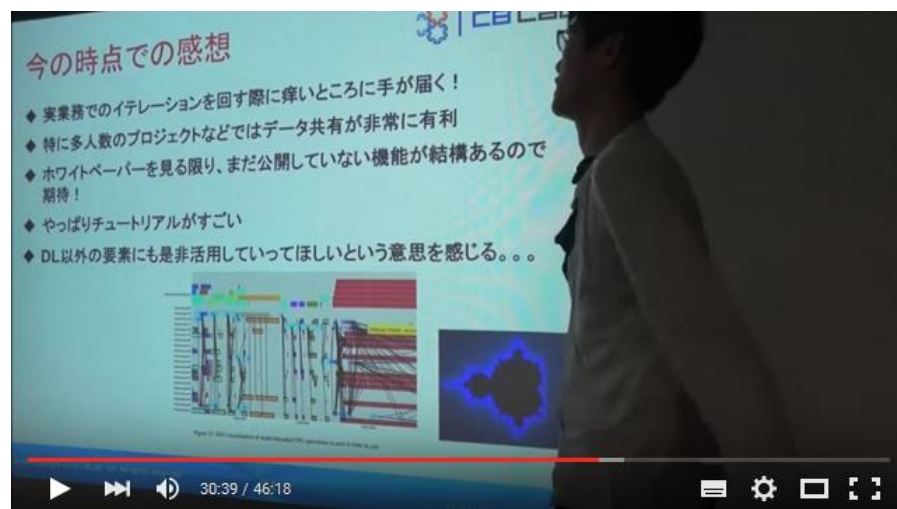


結構な数のパターンを少数データで検証して、あたりをつける必要がある。

19:20 / 46:18

今の時点での感想

- ◆ 実業務でのイテレーションを回す際に痒いところに手が届く!
- ◆ 特に多数のプロジェクトなどではデータ共有が非常に有利
- ◆ ホワイトペーパーを見る限り、まだ公開していない機能が結構あるので期待!
- ◆ やっぱりチュートリアルがすごい
- ◆ DL以外の要素にも是非活用して行ってほしいという意思を感じる。。。



30:39 / 46:18

TensorFlowのインストール方法

- 手順や詳細な説明は、公式サイトが充実
 - <https://www.tensorflow.org/>
 - 対応OS: Ubuntu (Linux), Mac
 - Windowsは、必要なツール(bazel)がうまく動かないらしい
 - GPU が見えるのはLinux のみ
 - 言語: Python2.7, Python3.3+



TensorFlowを触ってみよう

- 全体の流れ

- 私がAmazonEC2上に環境を構築
 - 興味のある人は、是非、各自でやってみてください
 - 見ているだけでも十分な内容ですが、自分専用の環境は楽しいので
- GPUは準備が間に合わなかったなので、使いません
- 今日はsshでログインできるので、無理に構築しなくてもOK

- 注意事項

- 実行時に /tmpやカレントの下に、固定名でフォルダができる
 - 個別に環境を構築していない人は、実行がバッティングする危険アリ
 - ソースをザッと見て、パスを修正するのが吉
- 実行がバッティングしたっぽい時は、一声かけてください

環境構築手順 - 1

The screenshot shows the AWS Management Console interface for selecting an Amazon Machine Image (AMI). The breadcrumb trail indicates the current step is "1. AMIの選択". The main heading is "ステップ 1: Amazon マシンイメージ (AMI)".

Three AMI options are listed:

- SUSE Linux Enterprise Server 12 SP1 (HVM), SSD Volume Type - ami-f8220896**
SUSE Linux
無料利用枠の対象
SUSE Linux Enterprise Server 12 Service Pack 1 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.
ルートデバイスタイプ: ebs 仮想化タイプ: hvm
- Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-a21529cc** (highlighted with a red box)
Ubuntu
無料利用枠の対象
Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
ルートデバイスタイプ: ebs 仮想化タイプ: hvm
- Microsoft Windows Server 2012 R2 Base - ami-14b8bc7a**
Windows
無料利用枠の対象
Microsoft Windows 2012 R2 Standard edition with 64-bit architecture. [English]
ルートデバイスタイプ: ebs 仮想化タイプ: hvm

Each AMI entry includes a "選択" (Select) button and the architecture "64ビット".

At the bottom, there is a blue banner with the text: "データベースインスタンスを起動中ですか。Amazon RDSをお試しください。" and a "非表示" (Hide) link.

The footer of the console includes "フィードバック", "日本語", "© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.", "プライバシーポリシー", and "利用規約".

環境構築手順-2

EC2 Management Console

https://ap-northeast-1.console.aws.amazon.com/ec2/v2/home?r

AWS サービス 編集 MICHITAKA ARAMOTO 東京 サポート

1. AMIの選択 2. インスタンスタイプの選択 3. インスタンスの設定 4. ストレージの追加 5. インスタンスのタグ付け 6. セキュリティグループの設定 7. 確認

ステップ 2: インスタンスタイプの選択

T2 インスタンスは VPC のみです。T2 インスタンスは VPC で作成されます。T2 と VPC については、[詳細はこちら](#)。

	ファミリー	タイプ	vCPU	メモリ (GiB)	インスタンス ストレージ (GB)	EBS 最適化利用	ネットワークパフォーマンス
<input type="checkbox"/>	汎用	t2.nano	1	0.5	EBS のみ	-	低から中
<input type="checkbox"/>	汎用	t2.micro 無料利用枠の対象	1	1	EBS のみ	-	低から中
<input type="checkbox"/>	汎用	t2.small	1	2	EBS のみ	-	低から中
<input checked="" type="checkbox"/>	汎用	t2.medium	2	4	EBS のみ	-	低から中
<input type="checkbox"/>	汎用	t2.large	2	8	EBS のみ	-	低から中
<input type="checkbox"/>		m4.large	2	8	EBS のみ	はい	中
<input type="checkbox"/>			4	16	EBS のみ	はい	高い

複数人がログインするので、ちょっと大きめ

キャンセル 戻る 確認と作成 次の手順: インスタンスの詳細の設定

フィードバック 日本語 © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. プライバシーポリシー 利用規約

環境構築手順-3

EC2 Management Console

https://ap-northeast-1.console.aws.amazon.com/ec2/v2/home?r

AWS サービス 編集 MICHITAKA ARAMOTO 東京 サポート

1. AMIの選択 2. インスタンスタイプの選択 3. インスタンスの設定 4. ストレージの追加 5. インスタンスのタグ付け 6. セキュリティグループの設定 7. 確認

ステップ 7: インスタンス作成の確認

▼ インスタンスタイプ [インスタンスタイプの編集](#)

インスタンスタイプ	ECU	vCPU	メモリ (GiB)	インスタンス ストレージ (GB)	EBS 最適化利用	ネットワークパフォーマンス
t2.medium	可変	2	4	EBSのみ	-	Low to Moderate

▼ セキュリティグループ [セキュリティグループの編集](#)

セキュリティグループ名 launch-wizard-4
説明 launch-wizard-4 created 2016-02-17T20:36:24.278+09:00

タイプ ⓘ	プロトコル ⓘ	ポート範囲 ⓘ
SSH	TCP	22

▶ インスタンスの詳細 [インスタンスの詳細の編集](#)

▶ ストレージ [ストレージの編集](#)

▶ タグ [タグの編集](#)

キャンセル 戻る 作成

フィードバック 日本語 © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. プライバシーポリシー 利用規約

Webサーバを起動するので、
ポート6006を開ける

環境構築手順－4

EC2 Management Console x +

https://ap-northeast-1.console.aws.amazon.com/ec2/v2/home?r 検索

AWS サービス 編集 MICHITAKA ARAMOTO 東京 サポート

1. AMIの選択 2. インスタンスタイプの選択 3. インスタンスの設定 4. ストレージの追加 5. インスタンスのタグ付け 6. セキュリティグループの設定 7. 確認

ステップ 6: セキュリティグループの設定

セキュリティグループは、インスタンスのトラフィックを制御するファイアウォールのルールセットです。このページで、特定のトラフィックに対してインスタンスへの到達を許可するルールを追加できます。たとえば、ウェブサーバーをセットアップして、インターネットトラフィックにインスタンスへの到達を許可する場合、HTTP および HTTPS ポートに無制限のアクセス権限を与えます。新しいセキュリティグループを作成するか、次の既存のセキュリティグループから選択することができます。Amazon EC2 セキュリティグループに関する [詳細はこちら](#)。

セキュリティグループの割り当て: 新しいセキュリティグループを作成する
 既存のセキュリティグループを選択する

セキュリティグループ名:

説明:

タイプ	プロトコル	ポート範囲	送信元
SSH	TCP	22	任意の場所 0.0.0.0/0
カスタム TCP ルール	TCP	6006	任意の場所 0.0.0.0/0

ルールの追加

警告

送信元が 0.0.0.0/0 のルールを指定すると、すべての IP アドレスからインスタンスにアクセスすることが許可されます。セキュリティグループのルールを設定して、既知の IP アドレスからのみアクセスできるようにすることをお勧めします。

キャンセル 戻る **確認と作成**

フィードバック 日本語 © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. プライバシーポリシー 利用規約

環境構築手順－5

EC2 Management Console

https://ap-northeast-1.console.aws.amazon.com/ec2/v2/home?r

AWS サービス 編集

1. AMIの選択 2. インスタンスタイプの選択 3. インスタンスの設定 4. ストレージの追加 5. インスタンスのタグ

ステップ 7: インスタンス作成の確認

インスタンスタイプ: t2.medium

セキュリティグループ: セキュリティグループの説明

タイプ: SSH

カスタム TCP ルール: カスタム TCP ルールの編集

インスタンスのタグ: インスタンスの詳細の編集

ストレージ: ストレージの編集

タグ: タグの編集

キャンセル インスタンスの作成

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. プライバシーポリシー 利用規約

既存のキーペアを選択するか、新しいキーペアを作成します。

キーペアは、AWS が保存するパブリックキーとユーザーが保存するプライベートキーファイルで構成されます。組み合わせて使用することで、インスタンスに安全に接続できます。Windows AMI の場合、プライベートキーファイルは、インスタンスへのログインに使用されるパスワードを取得するために必要です。Linux AMI の場合、プライベートキーファイルを使用してインスタンスに SSH で安全に接続できます。

注: 選択したキーペアは、このインスタンスに対して権限がある一連のキーに追加されます。パブリック AMI から既存のキーペアを削除するの詳細情報をご覧ください。

新しいキーペアの作成

キーペア名
aitc_cloud_20160217

キーペアのダウンロード

続行するには、事前にプライベートキーファイル(*.pem ファイル)をダウンロードする必要があります。それを、安全でアクセス可能な場所に保存します。一度作成されると、ファイルを再度ダウンロードすることはできなくなります。

EC2にログインするためのSSHの鍵をダウンロードする

環境構築手順-6

EC2 Management Console

https://ap-northeast-1.console.aws.amazon.com/ec2/v2/home?i

AWS サービス 編集 MICHITAKA ARAMOTO 東京 サポート

EC2 ダッシュボード

インスタンスの作成 接続 アクション

タグや属性によるフィルタ、またはキーワードによる検索

Name	インスタンス ID	インスタンスタイプ	アベイラビリティゾーン	インスタンスの状態	ステータスチェック
	i-e5f8fe40				

接続

Windows パスワードの取得
同様のものを作成

インスタンスの状態
インスタンスの設定
イメージ
ネットワーキング
ClassicLink
CloudWatch のモニタリング

接続方法を確認

インスタンス: i-e5f8fe40 パブリック IP: 52.192.141.1

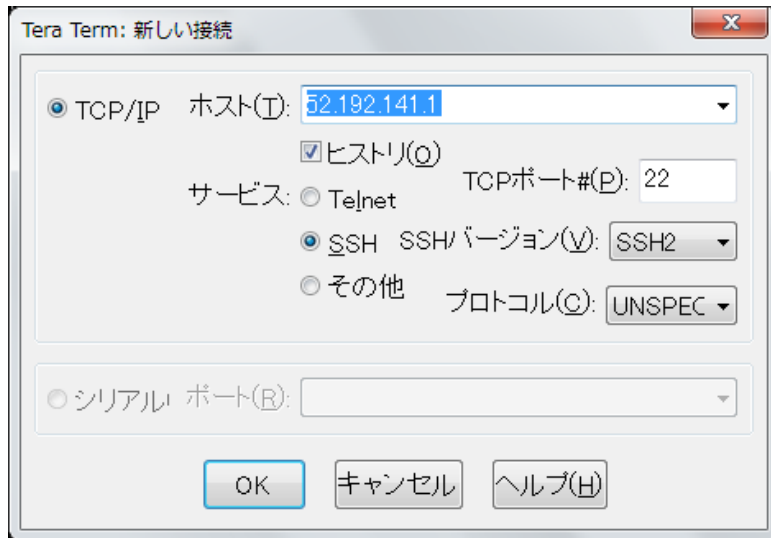
説明 ステータスチェック モニタリング タグ

インスタンス ID	i-e5f8fe40	パブリック DNS	-
インスタンスの状態	running	パブリック IP	52.192.141.1
インスタンスタイプ	t2.medium	Elastic IP	-

フィードバック 日本語 © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. プライバシーポリシー 利用規約

環境構築手順ー7

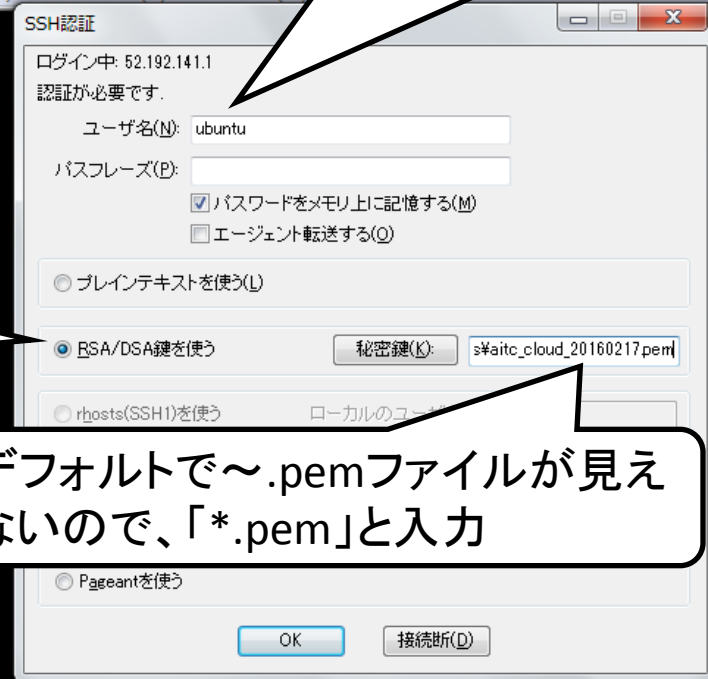
TeraTermでログインする場合



ユーザー名: ubuntu

このラジオボタンを選択

デフォルトで~.pemファイルが見えないので、「*.pem」と入力



環境構築手順－8

- 後は公式サイトの手順通り+α
 - https://www.tensorflow.org/versions/r0.7/get_started/os_setup.html

```
# Ubuntu/Linux 64-bit
$ sudo apt-get update
$ sudo apt-get install python-pip python-dev git

# 必要なライブラリを入れる
$ sudo pip install six numpy wheel ipython

# Ubuntu/Linux 64-bit, CPU only:
$ sudo pip install --upgrade https://storage.googleapis.com/tensorflow/linux/c
pu/tensorflow-0.5.0-cp27-none-linux_x86_64.whl

# ソースコードのダウンロード
$ git clone -b v0.6.0 --recurse-submodules https://github.com/tensorflow/tens
orflow.git
```

インストール完了! GPUを有効にするのは、もうちょっと大変

今日だけマルチユーザー化

- sshの秘密鍵を配布
- ログイン後に、自分のユーザーを作成

```
$ sudo adduser aramoto  
# パスワード以外は、空でOK  
  
# 作成したユーザーに変身: 公開鍵を置くとか、好きにしてください  
$ su - aramoto
```

- 自分のところにソースコードを取ってくる
 - → 気のむくままに改造OK

```
# ソースコードのダウンロード  
$ git clone -b v0.6.0 --recurse-submodules https://github.com/tensorflow/tensorflow.git
```

TensorFlowを理解しよう

- 参考URL

- http://www.slideshare.net/slideshow/embed_code/key/tkPJMfS5uuYI0N
- sample1.py と sample2.py 参照



デモを動かしてみよう

実行手順 - 1

- MNISTを試す

```
$ cd ~/tensorflow/tensorflow/examples/tutorials/mnist/  
$ python fully_connected_feed.py
```

<http://nextdeveloper.hatenablog.com/entry/2015/11/10/204609>



MNIST実行時エラー対策

- こんなエラーが発生するので

```
$ python fully_connected_feed.py
Traceback (most recent call last):
  File "fully_connected_feed.py", line 30, in <module>
    from tensorflow.examples.tutorials.mnist import input_data
ImportError: No module named tensorflow.examples.tutorials.mnist
```

- ソースコードを修正してください

```
30 # from tensorflow.examples.tutorials.mnist import input_data
31 # from tensorflow.examples.tutorials.mnist import mnist
32 import input_data
33 import mnist
```

ウワサのTensorBoardを試してみる

- 詳細なログが出ているので

```
$ ls -l ./data | awk '{print $5 "¥t" $9}'  
25341 events.out.tfevents.1455713350.ip-172-30-1-210  
1648877 t10k-images-idx3-ubyte.gz  
4542 t10k-labels-idx1-ubyte.gz  
9912422 train-images-idx3-ubyte.gz  
28881 train-labels-idx1-ubyte.gz
```

- TensorBoardを起動し、確認します

```
$ tensorboard --logdir `pwd`/data --port 6006
```

- ポート番号は重複しないように調整
- --logdir はフルパスで指定しないとダメ
- **chrome以外はメチャメチャ重い**
- GRAPH表示をキレイにするコツ
 - 「ズーム:25%」にしてリロード
 - GRAPH表示後にズームを元に戻す([Ctrl]+[0])

実行手順－2

- imagenetを試す
 - 画像に何が映っているかを判定

```
$ cd ~/tensorflow/tensorflow/models/image/imagenet
$ python classify_image.py
>> Downloading inception-2015-12-05.tgz 100.0%()
('Successfully downloaded', 'inception-2015-12-05.tgz', 88931400, 'bytes.')
I tensorflow/core/common_runtime/local_device.cc:25] Local device intra op
parallelism threads: 2
I tensorflow/core/common_runtime/local_session.cc:45] Local session inter
op parallelism threads: 2
giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca (s
core = 0.89233)
indri, indris, Indri indri, Indri brevicaudatus (score = 0.00859)
lesser panda, red panda, panda, bear cat, cat bear, Ailurus fulgens (score =
0.00264)
:
```

- **注意: 実行すると、データを/tmp/imagenet にダウンロード**

imagenetで画像判定

- ダウンロードした画像に何が映っているか判定

```
$ cd ~/tensorflow/tensorflow/models/image/imagenet  
$ wget http://画像ファイルのURL  
python classify_image.py --image_file 画像ファイル名
```

- どんなモノを事前に学習してあるか？
 - /tmp/imagenet の下のテキストファイル参照

実行手順－3

- cifar10を試す

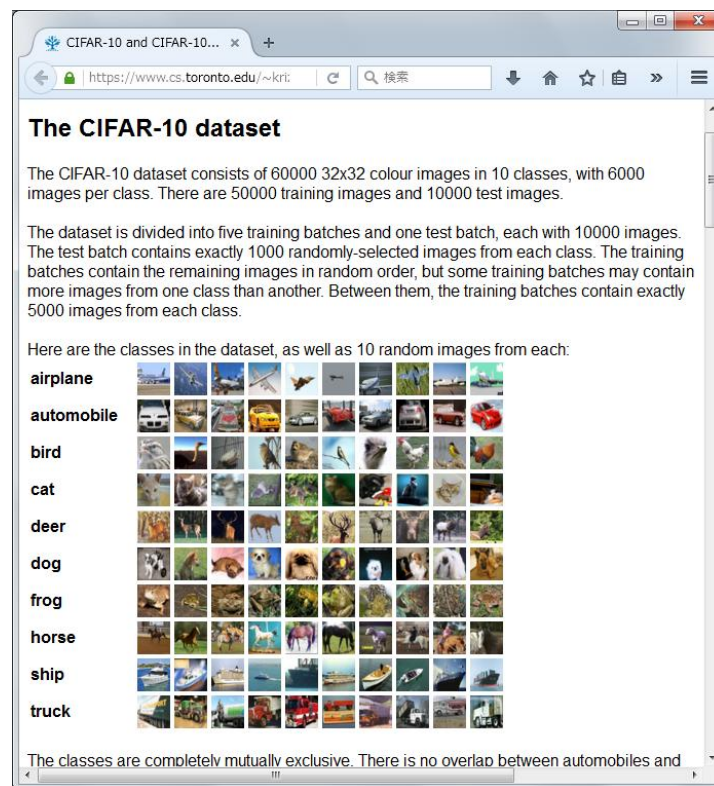
- 画像が、事前に学習した10種類のうちのどれに近い？

```
$ cd ~/tensorflow/tensorflow/models/image/cifar10  
$ python cifar10_train.py
```

- 注意

- データを /tmp/cifar10_data にダウンロード
- 学習結果を /tmp/cifar10_train に作成していく
 - ディレクトリ名を変更してください
- Tesla (GPU) を使って5時間
 - ソースコードの先頭参照

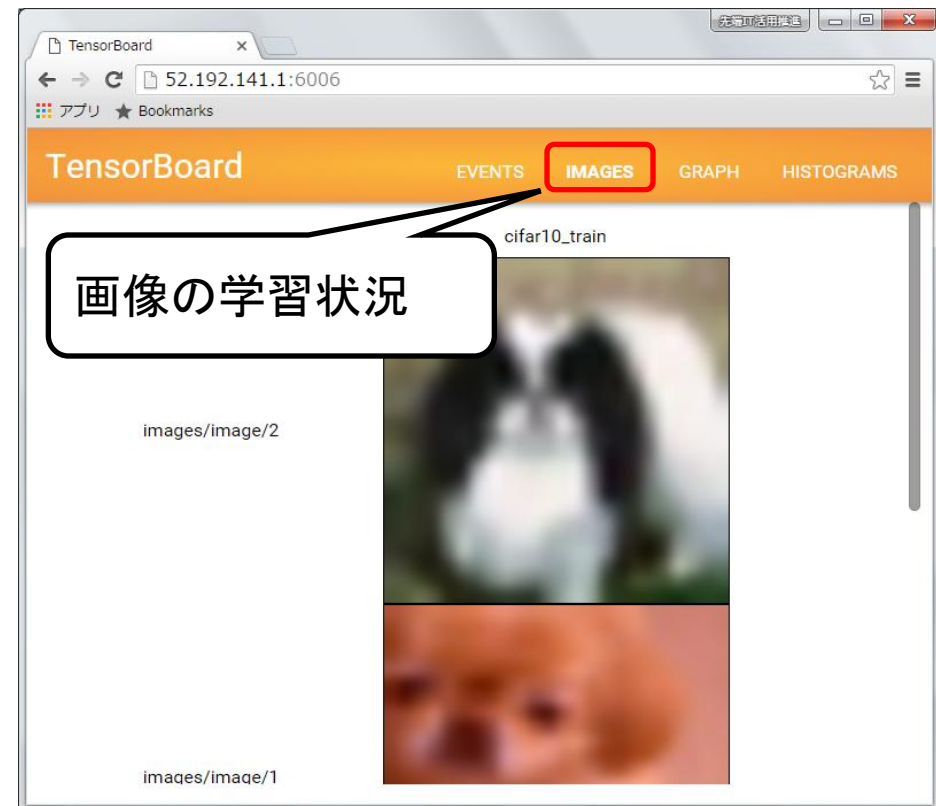
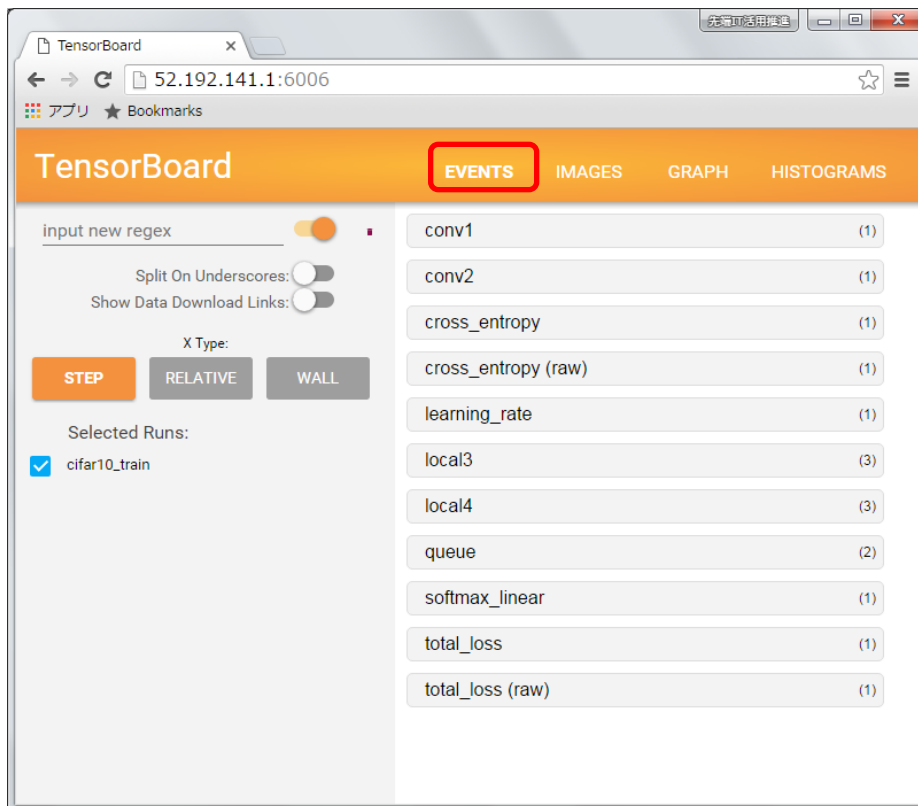
<https://www.cs.toronto.edu/~kriz/cifar.html>



cifar10もTensorBoardで見してみる

- TensorBoardを起動し、確認します

```
$ tensorboard --logdir /tmp/cifar10_train --port 6006
```



フリータイム

- 片っ端から実行してみる
 - 面白いデモを見つけたら、教えて
 - 動かないものは、動かす方法を共有して
- ソースコードを眺めてみる
- TensorBoard で見る
 - ログが出ない場合、以下のようなコードを追加

```
166 # Instantiate a SummaryWriter to output summaries and the Graph.  
167 summary_writer = tf.train.SummaryWriter(FLAGS.train_dir,  
168                                     graph_def=sess.graph_def)
```

~/tensorflow/tensorflow/examples/tutorials/mnist/fully_connected_feed.py より

参考情報

サンプルコード

githubに[サンプル](#)が色々あります。tensorflow/tensorflow/models の下に3つのフォルダで分かれていますね。

imageフォルダには

- * mnist : [手書き数字を分類](#)
- * cifar10 : [画像を分類](#)
- * imagenet : [cnnの1モデル](#)
- * alexnet : [cnnの1モデル](#)

※ cnn (Convolutional Neural Networks : [畳み込みニューラルネットワーク](#))

embeddingフォルダには

- * word2vec : [各単語の関係をベクトルで定量化して表現](#)

rnnフォルダには

- * rnn : [過去の文脈から次にどの単語がどのくらいの確率で出てくるか](#)
- * seq2seq : [機械翻訳](#)

<http://developers.gnavi.co.jp/entry/2016/01/28/TensorFlow%E3%81%A7%E5%A7%8B%E3%82%81%E3%82%8B%E6%B7%B1%E5%B1%A4%E5%AD%A6%E7%BF%92%E3%81%9D%E3%82%82%E3%81%9D%E3%82%82%E4%BD%95%E3%81%8C%E5%87%BA%E6%9D%A5%E3%82%8B%E3%81%AE%EF%BC%9F>