TensorFlowで遊んでみよう！

IL×CM×CR 合同勉強会

class method

自己紹介

名前 : 平田 圭 (@masuwo3)

所属 : データ分析チーム

現在 : 2015年5月よりクラスメソッド入社

過去 : 特任助教、SE、IT講師、etc...

ML : 研究室でかじった程度

Python歴 : 一ヶ月くらい

class method

今日のアジェンダ

- TensorFlowについての紹介
- TensorFlowを使ってみる

class method

TensorFlowの概要

class method

TensorFlowとはなにか

- 巷で話題の機械学習ライブラリ
- Deep Learning以降
- オープンソース / Googleが主導
- Python 2.7
- CUDA 7.0

caffe v0.1
2013/10/20

.....

chainer 公開
2015/6/9

TensorFlow 公開
2015/11/20

class method

TensorFlowの特長 (公式から)

- **柔軟性**
  - 機械学習のモデルを柔軟に記述できる
  - Neural Networkに限定しない
- **ポータビリティ**
  - 環境に合わせて計算処理を行う
  - CPU/GPU, ラップトップ/サーバ など
- **研究成果と製品の連結**
  - 研究成果の検証を行いやすく
  - プロダクトに転用しやすく

class method

TensorFlowにおける計算処理

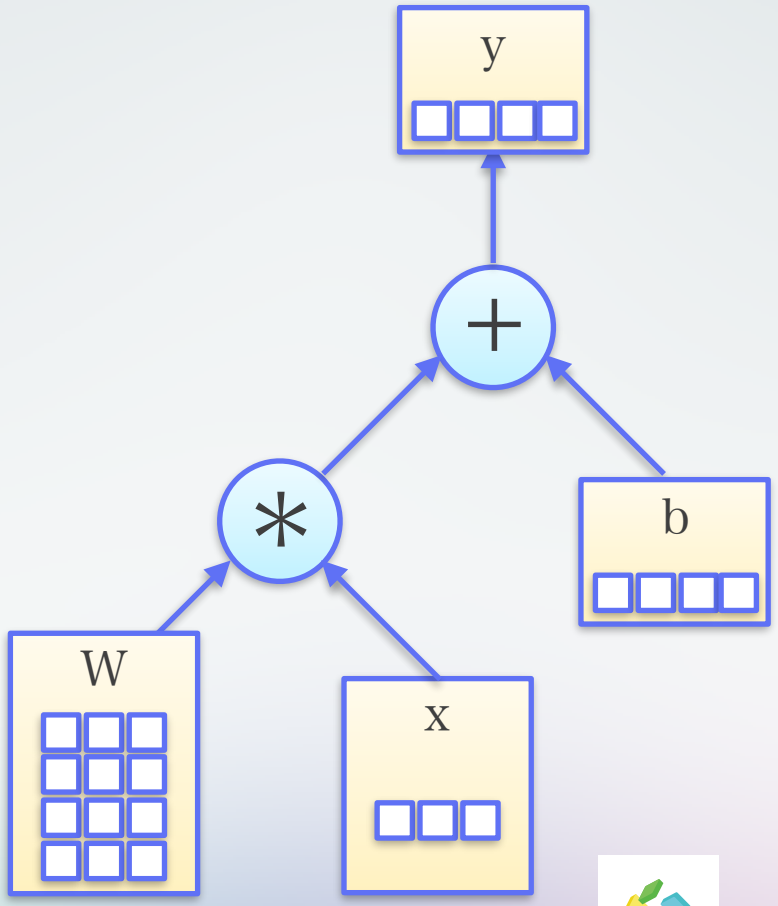
class method

TensorFlowにおける計算処理

TensorFlowは計算をグラフ構造で表現する

$$y = Wx + b$$

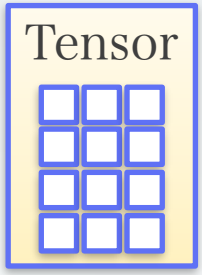
$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ w_{21} & \cdots & w_{2n} \\ \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$



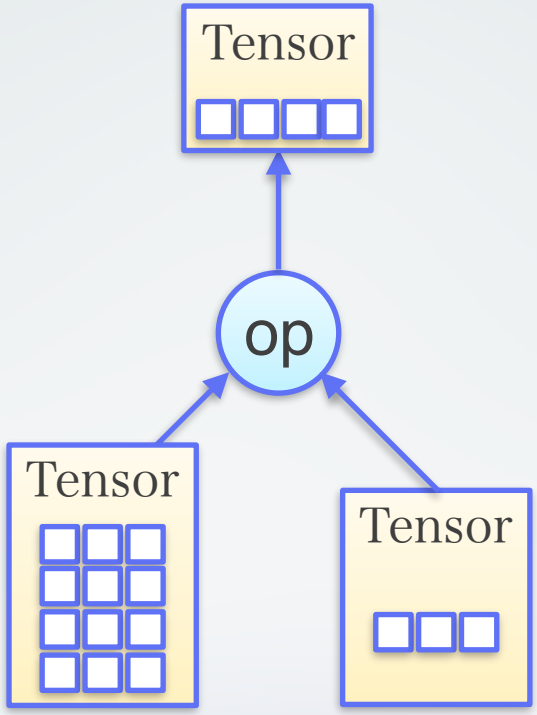
class method

TensorとOperation

データはTensor



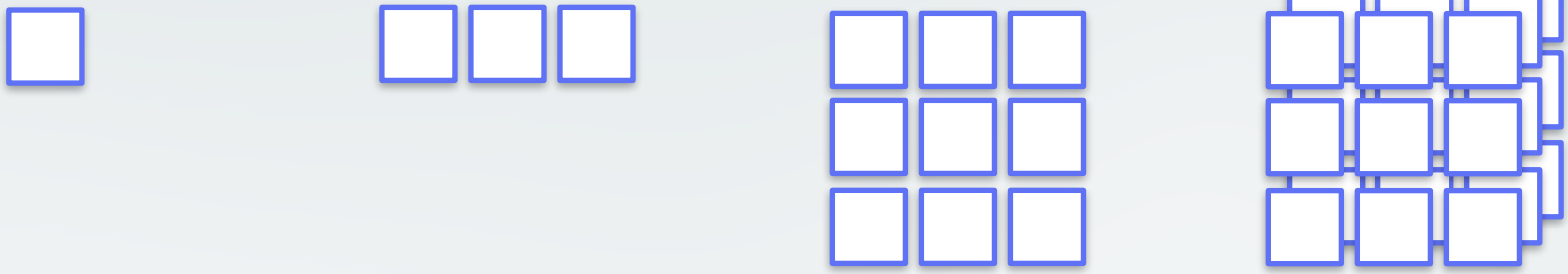
計算処理はOperation



class method

Tensorについて

階数0のテンソル (スカラ)	階数1のテンソル (ベクトル)	階数2のテンソル (行列)	階数3のテンソル



a	a[]	a[][]	a[][][]
1	[1,2,3,…]	[[1,2,3,…], [1,2,3,…], [1,2,3,…], …]	[[[1,2,3,…],…], [[1,2,3,…],…], [[1,2,3,…],…], …]

class method

Sessionについて

計算処理は2フェイズ(Define & Run)

Define

- 計算のグラフモデルを構築するフェイズ
- この時点では計算結果は確定しない

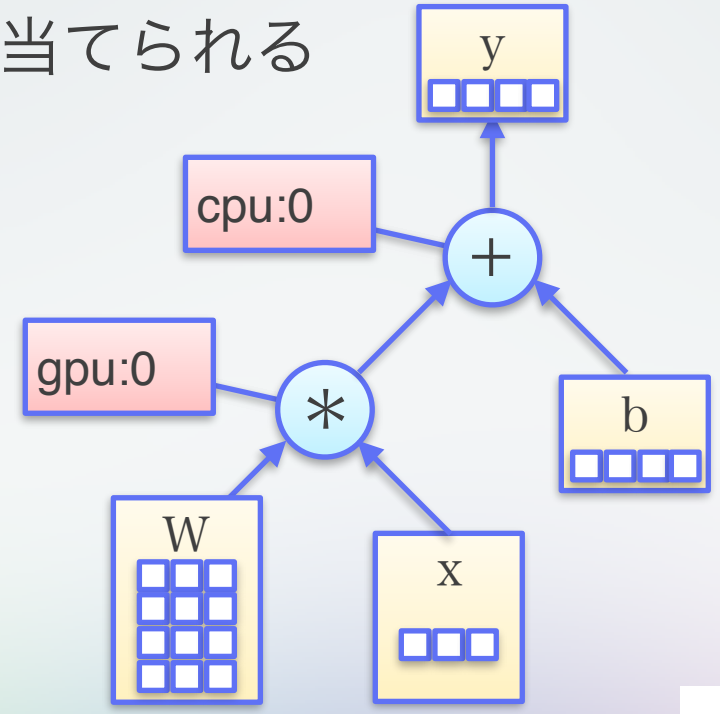
Run

- グラフモデルから計算結果を確定するフェイズ
- **Session**にモデルを投入し、計算結果を得る

class method

Sessionについて

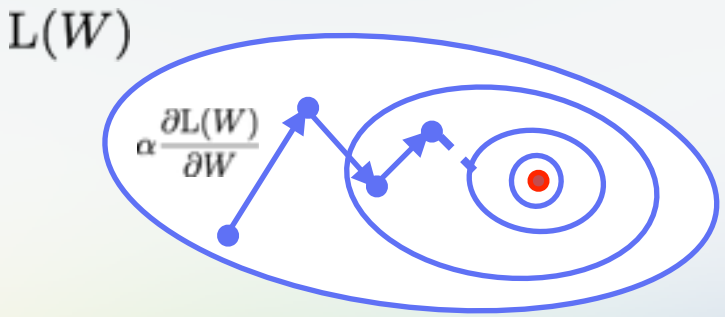
- **Session**
  - バックエンドのC++モジュールとのコネクション
  - 実際の演算はこのC++モジュール上で行われる
  - 計算リソースが自動的に割り当てられる



class method

Optimizerについて

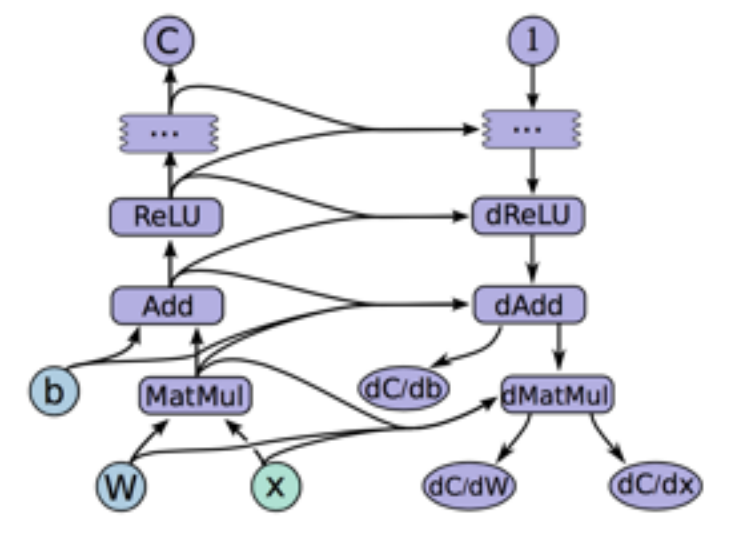
- 損失関数の最小化
  - 機械学習の「学習」は、損失関数の最小化に置き換えられる
  - 最小点の探索には、勾配計算が必要
  - TensorFlowでは**Optimizer**で学習を一括して行う



class method

Optimizerについて

- **Optimiser**
  - 勾配計算から探索までを一括で行う
    - 勾配の計算は、グラフモデルを自動的に変換
    - 学習手法に合わせて内部の重みを更新していく
  - グラフで表現された損失関数から、自動的に学習を行う



class method

サンプルコードの解説

```
import tensorflow as tf
import numpy as np

# Make 100 phony data points in NumPy.
x_data = np.float32(np.random.rand(2, 100)) # Random input
y_data = np.dot([0.100, 0.200], x_data) + 0.300

# Construct a linear model.
b = tf.Variable(tf.zeros([1]))
W = tf.Variable(tf.random_uniform([1, 2], -1.0, 1.0))
y = tf.matmul(W, x_data) + b

# Minimize the squared errors.
loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)

# For initializing the variables.
init = tf.initialize_all_variables()

# Launch the graph
sess = tf.Session()
sess.run(init)

# Fit the plane.
for step in xrange(0, 201):
    sess.run(train)
    if step % 20 == 0:
        print step, sess.run(W), sess.run(b)
```

Define

- データ・モデルの定義
- 学習モデルの構築
- 損失関数、Optimizer

Run

- 計算処理の実施
- Optimizerを用いた学習

class method

サンプルコードの解説

Define

```
# Make 100 phony data points in NumPy.
x_data = np.float32(np.random.rand(2, 100)) # Random input
y_data = np.dot([0.100, 0.200], x_data) + 0.300
```

訓練用の入出力データを作成

- $x = (x_1, x_2)^{\mathrm{T}}$
  $\bar{y} = (0.1, 0.2)(x_1, x_2)^{\mathrm{T}} + 0.3$

```
# Construct a linear model.
b = tf.Variable(tf.zeros([1]))
W = tf.Variable(tf.random_uniform([1, 2], -1.0, 1.0))
y = tf.matmul(W, x_data) + b
```

訓練モデルの構築

- モデル内部の重みを宣言

$$y = wx + b$$

```
# Minimize the squared errors.
loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)
```

損失関数、optimizerを設定

- $loss(w) = \frac{(y - \bar{y})^2}{2}$
- SGD(確率的勾配降下法)を指定

class method

サンプルコードの解説

Run

```
# For initializing the variables.
init = tf.initialize_all_variables()

# Launch the graph
sess = tf.Session()
sess.run(init)

# Fit the plane.
for step in xrange(0, 201):
    sess.run(train)
    if step % 20 == 0:
        print step, sess.run(W), sess.run(b)
```

内部の重みを初期化

学習を繰り返し実行

- 20回ごとに結果を表示

class method

サンプルコードの解説

- 実行結果

```
0 [[ 0.33891663 -0.30841002]] [ 0.76058954]
20 [[ 0.10073041  0.04387058]] [ 0.37181979]
40 [[ 0.09185985  0.15895699]] [ 0.32304254]
60 [[ 0.09554119  0.18877636]] [ 0.30741674]
80 [[ 0.09815928  0.19681442]] [ 0.30239245]
100 [[ 0.09931748  0.19906586]] [ 0.30077288]
120 [[ 0.09976013  0.19971865]] [ 0.3002499]
140 [[ 0.09991822  0.19991349]] [ 0.30008087]
160 [[ 0.0999726   0.19997297]] [ 0.30002618]
180 [[ 0.09999093  0.19999145]] [ 0.30000848]
200 [[ 0.099997    0.19999729]] [ 0.30000275]
```


class method

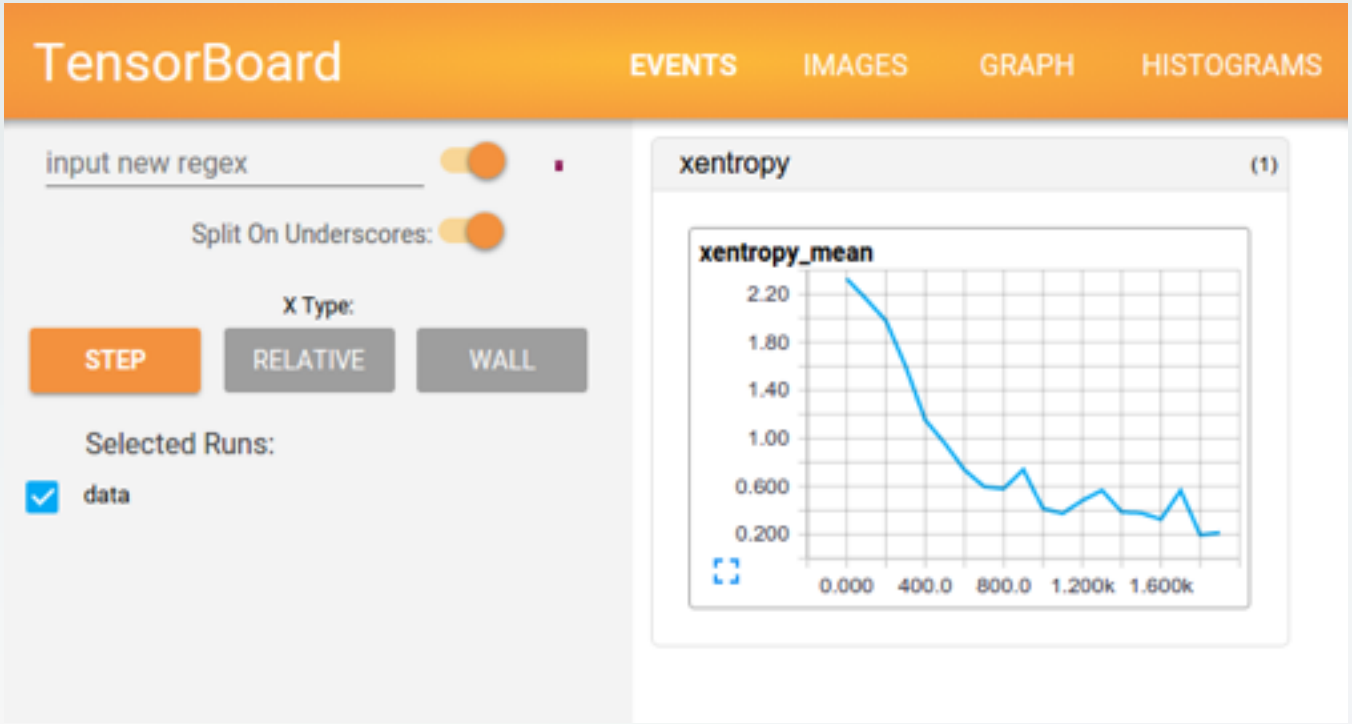
Chainerとの違い

- MLライブラリ
  - 大きな違いはない印象
  - APIの好みの問題？
  - (Python歴1ヶ月の感想です)
- 分散処理
  - どちらもシングルマシン / マルチGPU
  - TensorFlowは将来的にマルチマシン対応?

class method

Chainerとの違い

- TensorBoard

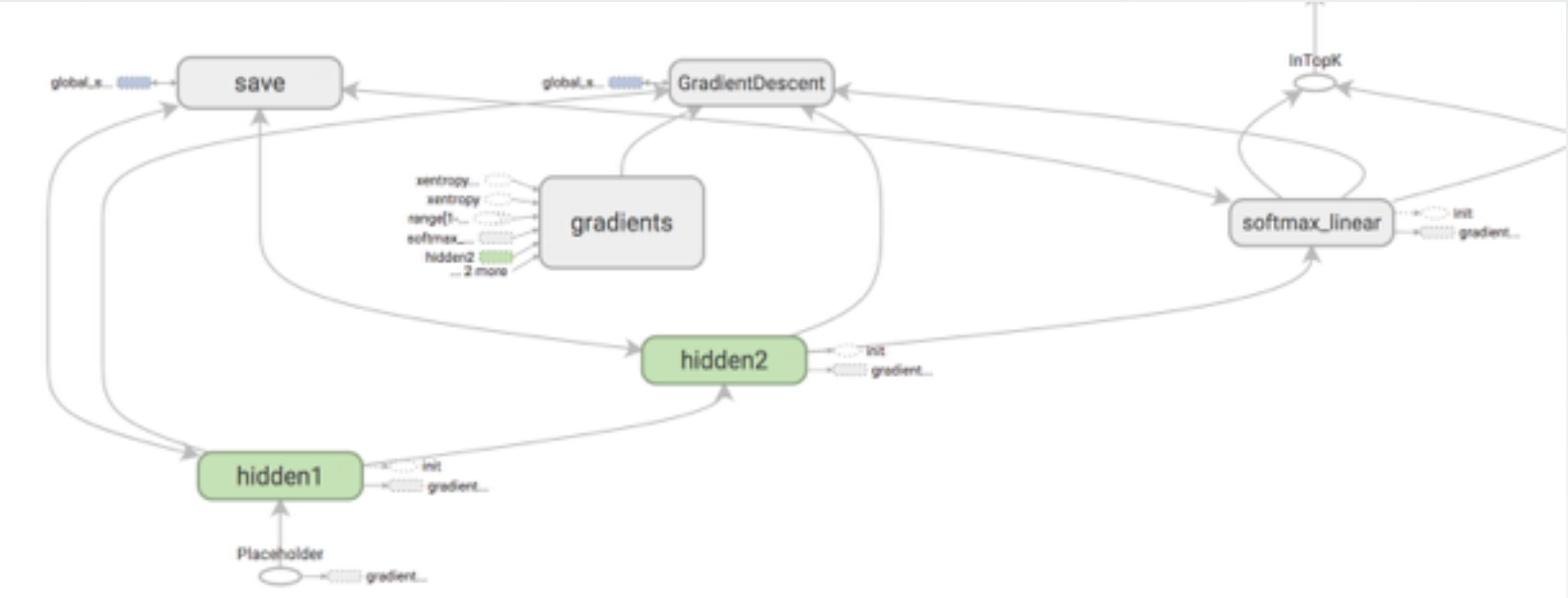


学習過程やモデルの構造を可視化

class method

Chainerとの違い

- TensorBoard



学習過程やモデルの構造を可視化

class method

TensorFlowを使ってみる

class method

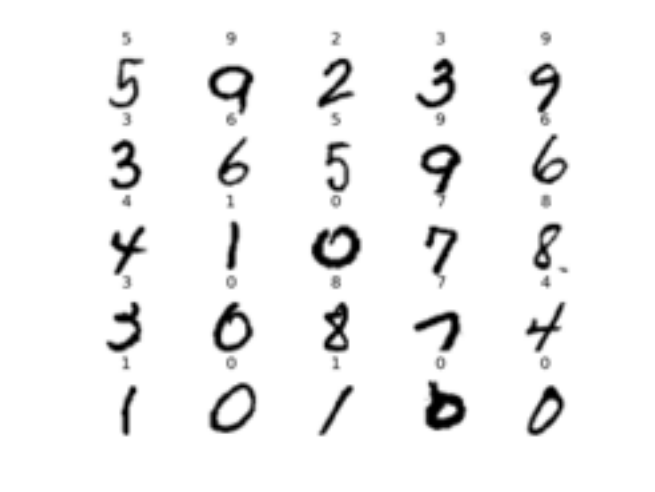
TensorFlowのチュートリアル

- とても充実
  - MNIST for Beginners
  - **Deep MNIST for Expert**
  - Convolutional Neural Networks
  - Vector Representations of Words
  - Sequence-to-Sequence Models
  - Mandelbrot Set (?)
  - etc…

class method

MNIST

- 手書き数字のデータセット
- サイズ : 28x28
- 70,000枚 (訓練 60k, テスト 10k)
- 画像識別のHello World



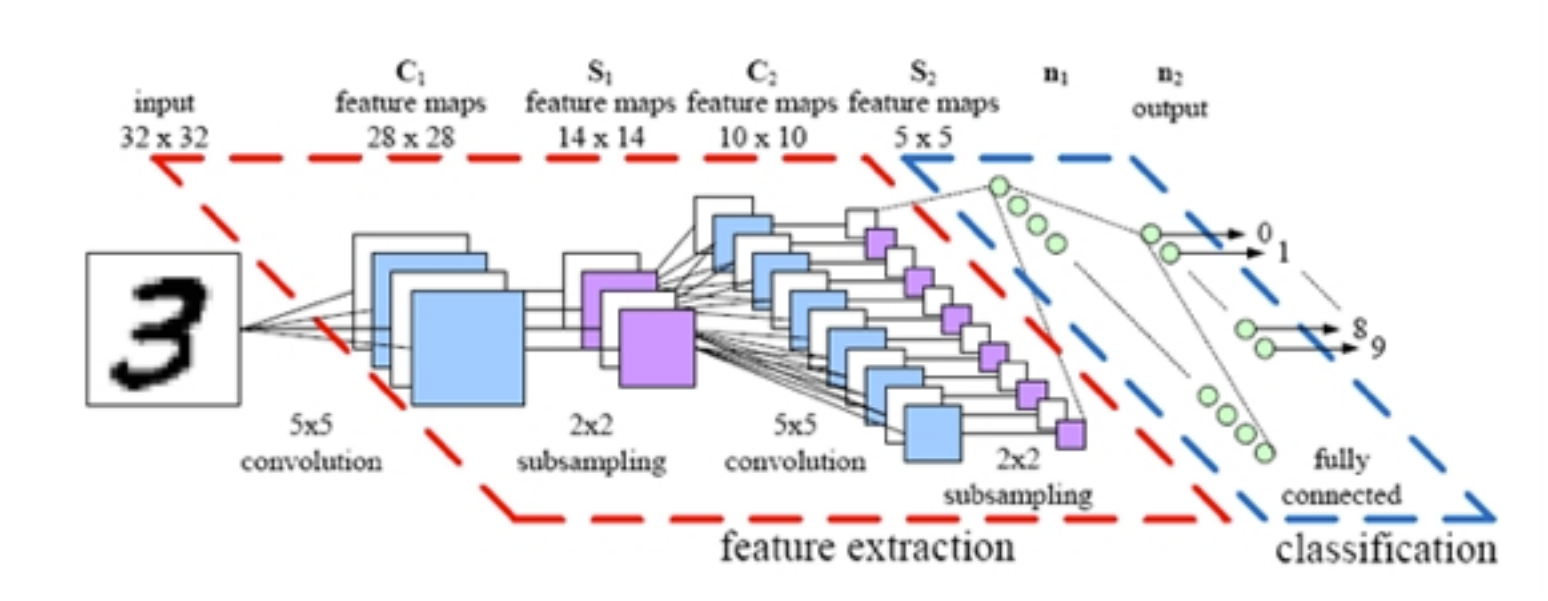
class method

Convolutional Neural Network

- 畳み込みNN (CNN)
  - Deep Learningの代表的手法
  - 主に画像識別に用いられる
  - 畳み込み層とプーリング層の組み合わせ
  - 人間の視覚野の仕組みを参考

class method

Convolutional Neural Network



詳細は割愛

class
method

TensorFlowでCNN

- 畳み込み層

```
tf.nn.conv2d(input, filter, strides, padding,
use_cudnn_on_gpu=None, name=None)
```

Computes a 2-D convolution given 4-D `input` and `filter` tensors.

- プーリング層

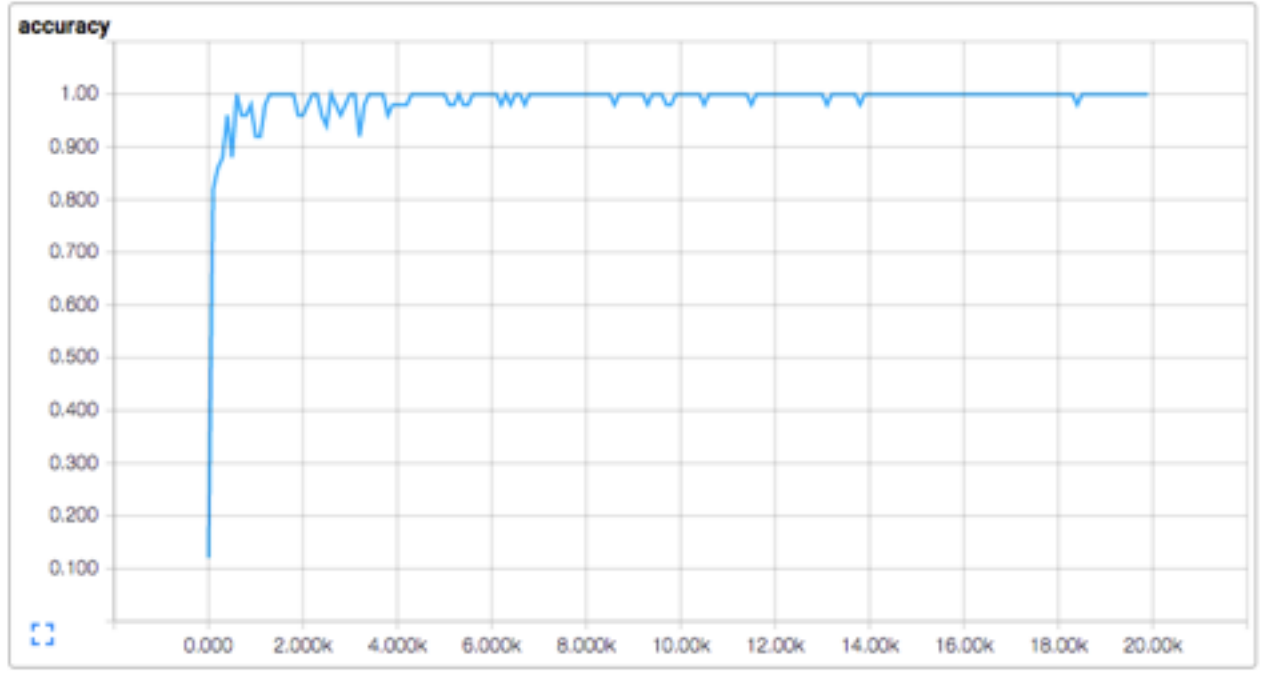
```
tf.nn.max_pool(value, ksize, strides, padding,
name=None)
```

Performs the max pooling on the input.

APIを組み合わせることでCNNを構築できる

class method

結果



テストデータでの正答率 : 98.71%

class method

まとめ

- TensorFlow
  - 行列計算 + ML用API
    - 最新の学習手法を簡単に検証できる
  - グラフモデルで計算を表現
    - 稼働環境のリソースを割り当て
    - 勾配計算を自動的におこなう
  - モデルや学習状況の可視化

class method

おまけ

やりたかったこと

- DQN (Deep-Q-Network)
  - Deep Learning + 強化学習
  - モニタ画像からゲームの戦略を学習
  - 同じモデルで様々なゲームを攻略
  - ゲームによってはプロ以上の成果

http://www.nature.com/nature/journal/v518/n7540/full/nature14236.html

class method