PythonとKerasによるディープラーニング
7.2 Kerasのコールバックと TensorBoardを使ったディープラーニングモデルの調査と監視

17T4028N 菊田尚樹

コールバックによるモデルの制御

訓練中にモデルの内部で起きていることをより詳しく確認し、制御したい→Kerasのコールバックを用いる

コールバック

fit呼び出しを通じてモデルに渡され、訓練中に様々なタイミングでモデルから呼び出されるオブジェクト。モデルの状態やその性能に関する利用可能なデータのすべてにアクセスし、訓練の中止、モデルの保存、異なる重みの読み込み、モデルの状態の変更といった措置をとれる。

コールバックによるモデルの制御

◇コールバックの使用例

・モデルのチェックポイント化
訓練中の様々な地点でモデルの現在の重みを保存

・訓練の中止
訓練データでの損失値がそれ以上改善しなくなったところで訓練を中止する

・特定のパラメータの動的な調整
訓練中にオプティマイザの学習率などのパラメータの値を動的に調整

・訓練と検証の指標を記録
ログに記録する又は学習された表現を可視化する

コールバックによるモデルの制御

Keras.callbackモジュールの例

- keras.callbacks.ModelCheckpoint ←
- keras.callbacks.EarlyStopping ←
- keras.callbacks.ReduceLROnPlateau ←
- keras.callbacks.LearningRateScheduler
- keras.callbacks.CSVLogger

今回は3つを見ていく

ModelCheckpoint とEarlyStopping

EarlyStoppingコールバック：訓練の中止

過学習が起きたら中止といったことができるので効率的

ModelCheckpointコールバック：モデルの保存

1つのエポックが終了した時点で最も性能が良いモデルの保存ができる

成果指標が一定のエポック数にわたって改善されなかった場合はEarlyStoppingとModelCheckpointを組み合わせて使うことで効率よく実験できる。

ModelCheckpoint とEarlyStopping

指標が正解率の場合の例→

① コールバックのリストを作る

② fitのパラメータにリストを渡す

monitor : 監視する指標

patience :

コールバックの起動条件

今回は2回連続で指標が改善しなければ中止される

```
import keras
callbacks_list = [
  keras.callbacks.EarlyStopping(
            monitor = 'val_acc' ,
            patience = 1 ,
  ),
  keras.callbacks.ModelCheckpoint(
            filepath = 'my_model.h5' ,
            monitor = 'val_loss' ,
            save_best_only = True
  )
]
model.compile(optimizer = 'rmsprop' ,
        loss = 'binary_crossentropy' ,
        metrics = ['acc']
 )
model.fit(x , y ,
        epochs = 10 ,
        batch_size = 32 ,
         callbacks = callbacks_list ,
          validation_data = (x_val , y_val)
)
```

ReduceLROnPlateauコールバック

学習率の調整を行う。検証データでの損失率が改善しなくなった場合に用いる。

```
callbacks_list = [
  keras.callbacks. ReduceLROnPlateau(
            monitor = 'val_loss' ,
            factor = 0.1 ,    #学習率を10で割る
            patience = 10 ,
  )
]

model.fit(x , y ,
       epochs = 10 ,
       batch_size = 32 ,
       callbacks = callbacks_list ,
       validation_data = (x_val , y_val)
)
```

カスタムコールバック

既存のコールバックではカバーされていない動作を訓練中に実行したい場合はコールバックを自作することで実現する。

keras.callbacks.Callbacksクラスのサブクラスとして作成する。

関数は呼び出しのタイミングごとに以下の6つがある

on_epoch_begin	on_epoch_end
on_batch_begin	on_batch_end
on_train_begin	on_train_end

```
class <名前> (keras.callbacks.Callbacks) :
  def on_train_begin( self , logs = {} ) :
        <処理>
  def on_batch_end( self , batch , logs = {} ) :
        <処理>
```

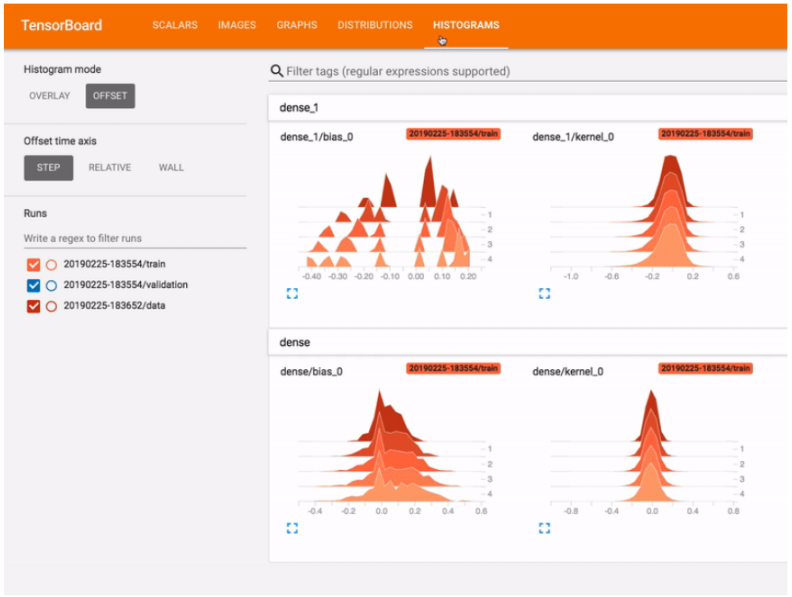
TensorBoard

TensorFlowに含まれるフレーム。ブラウザでモデル内部を可視化する。

KerasではTensorBoardコールバックによって使用する。

◇主な目的

- 訓練中に指標を視覚的に監視
- モデルのアーキテクチャの可視化
- 活性化と勾配のヒストグラムの可視化
- 埋め込みを3次元で調査



TensorBoard

Kerasでの利用方法

事前にTensorBoardが作成するログファイルを格納するディレクトリを用意しておく。　　$ mkdir my_log_dir

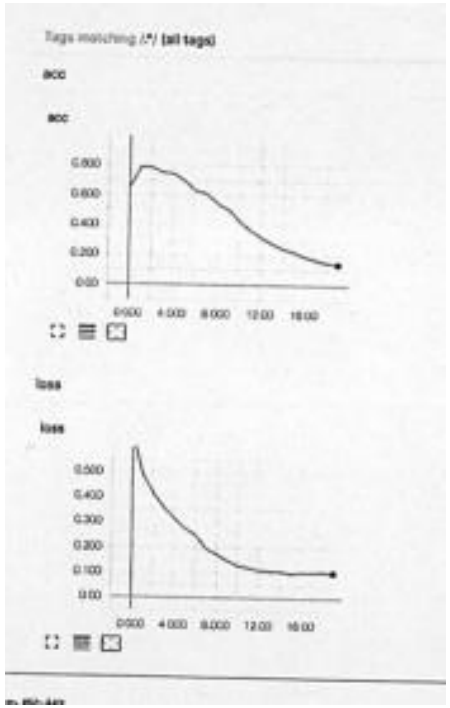
```
callbacks = [
  keras.callbacks. TensorBoard(
            log_dir = 'my_log_dir' ,
            histogram_freq = 1 ,
            embeddings_freq = 1
  )
]

history = model.fit(x_train , y_train , epochs=20,
                    batch_size = 128,
                    validation_split = 0.2 ,
                     callbacks = callbacks)
```

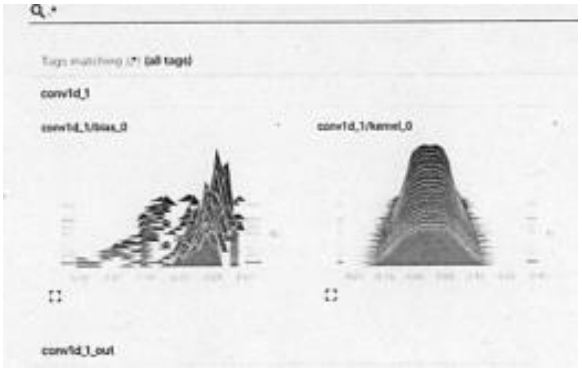
TensorBoard

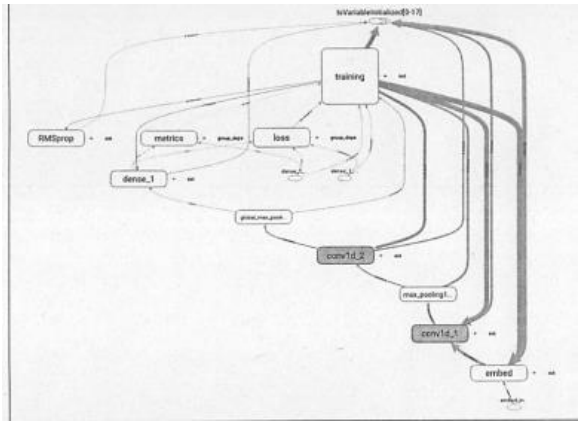
Kerasでの利用方法 続き

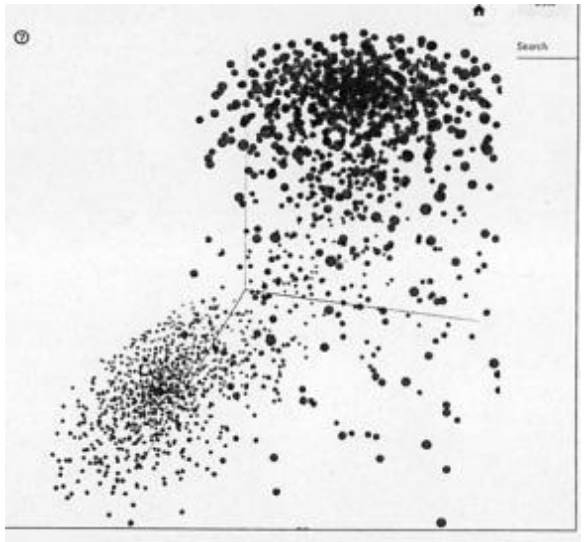
コマンドラインからTensorBoardサーバを立ち上げ、コールバックが書き込んでいるログを読み取らせる。

$ tensorboard --logdir = my_log_dir









まとめ

・Kerasのコールバックは訓練中のモデルを監視し、状態に応じて自動的にアクションを実行するのに用いる。

・TensorBoardはモデルの状態をブラウザで可視化するツール。KerasではTensorBoardコールバックで実行する。