PythonとKerasによるディープラーニング
3章　入門：ニューラルネットワーク
3.2 Kerasの紹介

16T4063F 結城洸太

Kerasの紹介

- Pythonのディープラーニングのフレームワーク
- ディープラーニングモデルを定義して訓練するための手段を提供
- MITライセンスで配布されており、営利目的のプロジェクトでも無償で利用可能

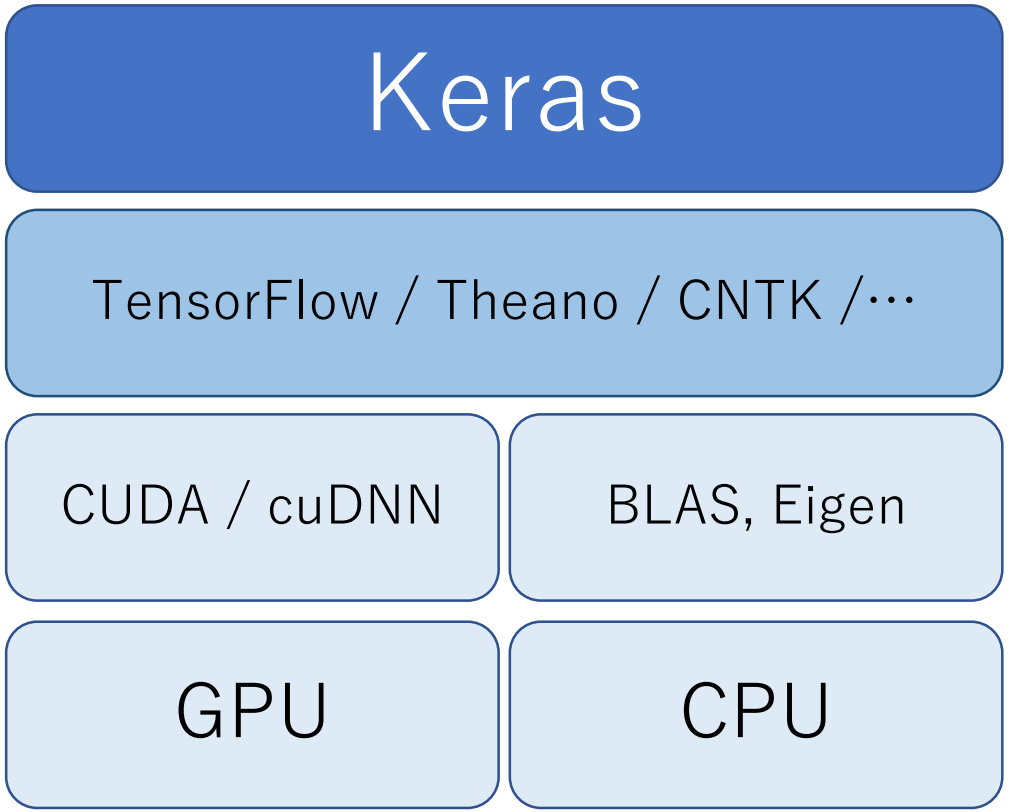
Kerasの特徴

- CPUでもGPUでも同じコードでシームレスに実行できる。
- ディープラーニングモデルのプロトタイプを簡単にすばやく作成できる。
- 畳み込みネットワーク、リカレントネットワーク、およびそれらの組み合わせを組み込みでサポートしている。
- 複数入力/複数出力モデル、層の共有、モデルの共有など、任意のネットワークアーキテクチャをサポートしている。

3.2.1 Keras、TensorFlow、Theano、CNTK

- Kerasのバックエンドエンジンとして、テンソルライブラリを利用する。
- テンソルライブラリを１つ選択して結び付けるのではなく、モジュール方式で処理する。

→数種類のバックエンドエンジンをシームレスに接続できる。(コード側での変更なしにすべてで実行できる)



3.2.2 速習：Kerasを使った開発

Kerasの標準的なワークフロー

1. 訓練データ(入力テンソルと目的テンソル)を定義する。
2. 入力値を目的値にマッピングする複数の層からなるネットワーク(モデル)を定義する。
3. 損失関数、オプティマイザ、監視する指標を選択することで、学習プロセスを設定する。
4. モデルのfitメソッドを呼び出すことで、訓練データを繰り返し学習する。

コード例(2層、損失関数１つのモデル)

```
from keras import models
from keras import layers

model = models.Sequential()
model.add(layers.Dense(32, activation='relu', input_shape(784,)))
model.add(layers.Dense(10, activation='softmax'))(x)

from keras import optimizers

model.compile(optimizer=optimizers.RMSprop(lr=0.001),
                    loss='mse',
                    metrics=['accuracy'])

model.fit(input_tensor, target_tensor, batch_size=128, epochs=10)
```