物性アプリオープン
フォーラム概要説明

2019/08/28 (水) 16:00-17:00

東京大学物性研究所 第5セミナー室(A615)

物性アプリオープンフォーラムとは？

目的

- 自分の開発した物性アプリを公開したい
- 既存のアプリに自らの研究のための新機能を組み込みたい
- 既存のアプリを自らの開発したアプリに組み込みたい
- 既存のアプリを利用して研究を行いたい

このような考えを持つ研究者がスムーズにそれを実行に移せるよう、
必要な技術・知識を習得する。

2015年4月より開始。
日程：毎月最終水曜日15:00〜16:00で定期的に開催(一部都合により変更)。
講義スタイル：途中質問OK、フランクな研究会。

物性アプリオープンフォーラムとは？

内容例

- ライセンスの種類(MIT/X,Apache,GNU GPL等)と特徴。
- バージョン管理ソフトとリポジトリサーバーの種類と特徴
- 国内外の物性アプリの開発方法・開発状況
- 既存物性アプリを実際に使用し、コンパイル/操作方法や得られる物理量、機能についてレビューする
- 同系統の物性アプリ間の機能等の比較
- 現在自分のアプリ開発で直面している問題について発表
  (理想としては解決策を研究会で検討・模索する)

Google Colaboratory 体験記

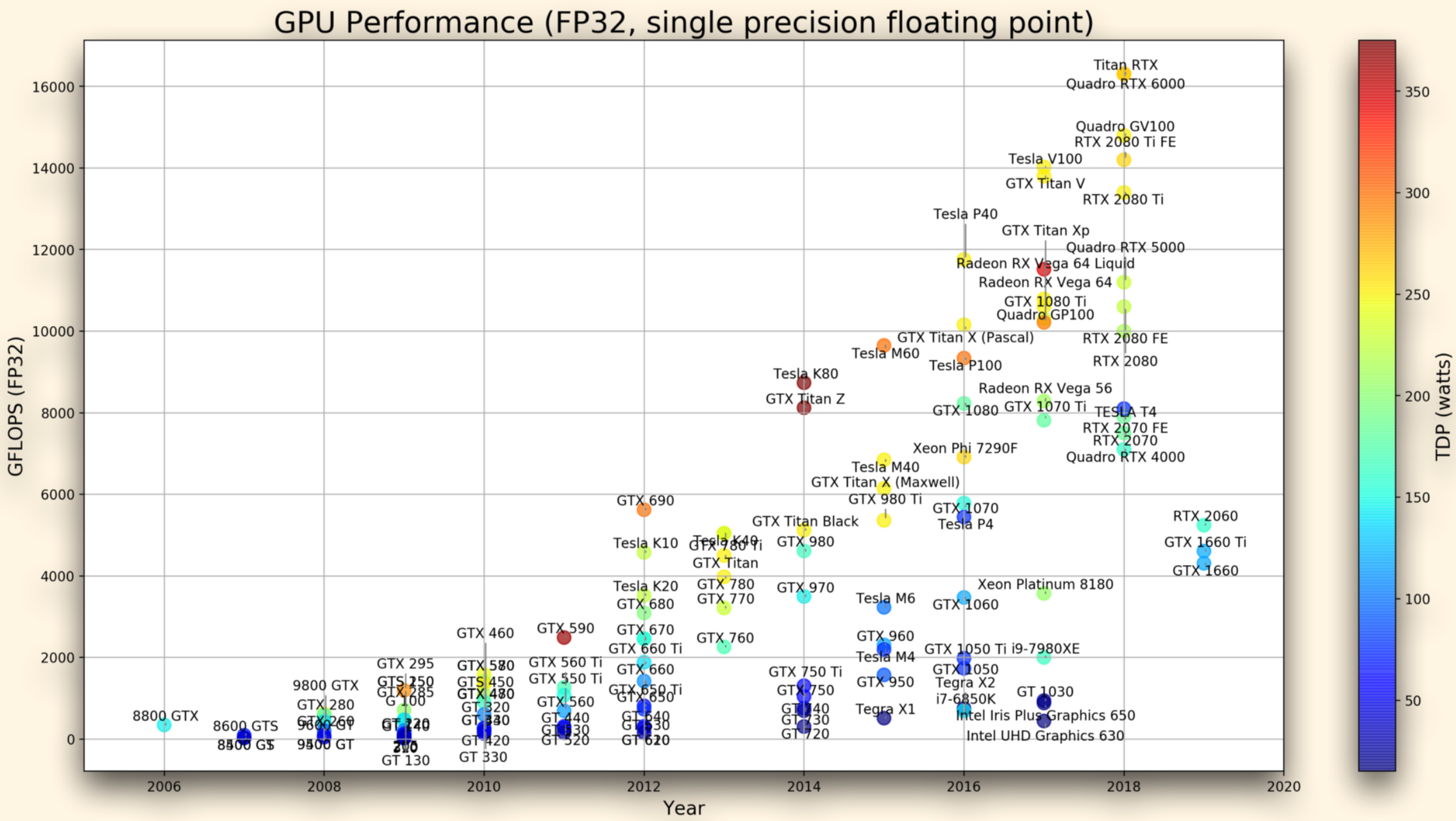
東京大学物性研究所　物質設計評価施設

大型計算機室 吉見一慶

2019/08/28 (水) 16:00-17:00

東京大学物性研究所 第5セミナー室(A615)

演算性能の推移



ref.) https://blog.inten.to/hardware-for-deep-learning-part-3-gpu-8906c1644664

GPUを利用するには？

1. クラウド環境の利用
    1. Amazon Web Service (AWS)
    2. Microsoft Azure
    3. Google Compute Engine
    4. さくらインターネット
2. スパコンの利用
    1. 物性研スパコン システムB
    2. ABCIスパコン
    3. TSUBAME
3. 自分で購入

クラウド環境での利用

スペックの近いインスタンスでの従量課金の料金比較

(2018/11/30時点)

サービス名	シリーズ名／モデル	GPU	1000時間単位の金額 (105円/＄で換算)
AWS	p2インスタンス	1	$1542 (約 ¥16万3千円)
GCP	NVIDIA® Tesla® P100	1	$1600 (約17万円)
Azure	NV シリーズ	1	¥176,960
さくらインターネット(*)	Tesla P100	1	¥357,000

(*)さくらインターネットは10万円程度の月額利用もあり(ただし初期投資に80万近く必要)。

ref.) https://hoominkani.hatenablog.com/entry/2018/11/30/123830

スパコンでの利用について

1. 物性研スパコン システムB
    1. Tesla K40×2 /ノード
    2. ポイント申請すれば無料 (2pt./ノード)
2. ABCIスパコン
    1. Nvidia Tesla V100 × 4 /ノード
    2. 1ノード占有：18万(税抜)/1000時間
3. TSUBAME 3.0
    1. Nvidia Tesla P100 ×4/ノード
    2. 1ノード占有：10万(税抜)/1000時間 (学術利用)

自分で購入

Tesla V100の場合

1GPUあたり100万程度

+　ワークステーション導入

→ 4GPUだと400-500万程度

ref.) HPC sytemsのホームページより抜粋

研究で本格的に使用する場合には、クラウドに比べ、

コストが約1/4で済むABCIスパコンやTSUBAMEなどの

国の機関を利用するのがお得そう。

→もう少しお手軽に始められないか？

Google Colaboratoryとは？

- 機械学習の教育・研究用に使われることを目的にGoogleが無償で提供しているクラウドで実行される Jupyter ノートブック環境
- Googleアカウントさえあれば無料で利用可能
- Tesla K80 GPUが無料で使用可能
- ただし、12時間ルールと90分ルールがあり。

Google Colaboratoryとは？

- 仕様
  - **Ubuntu 18.04.2**
  - **2vCPU @2.2GHz**
  - **13GB RAM**
  - ストレージ：**GPUなし 40GB, GPUあり 360GB**
  - **GPU NVIDIA Tesla K80 12GB**
- ルール
  - アイドル状態が**90**分続くと停止
  - 連続使用は最大**12**時間
  - **Notebook**サイズは最大**20MB**

Google Colaboratoryの始め方 (1)

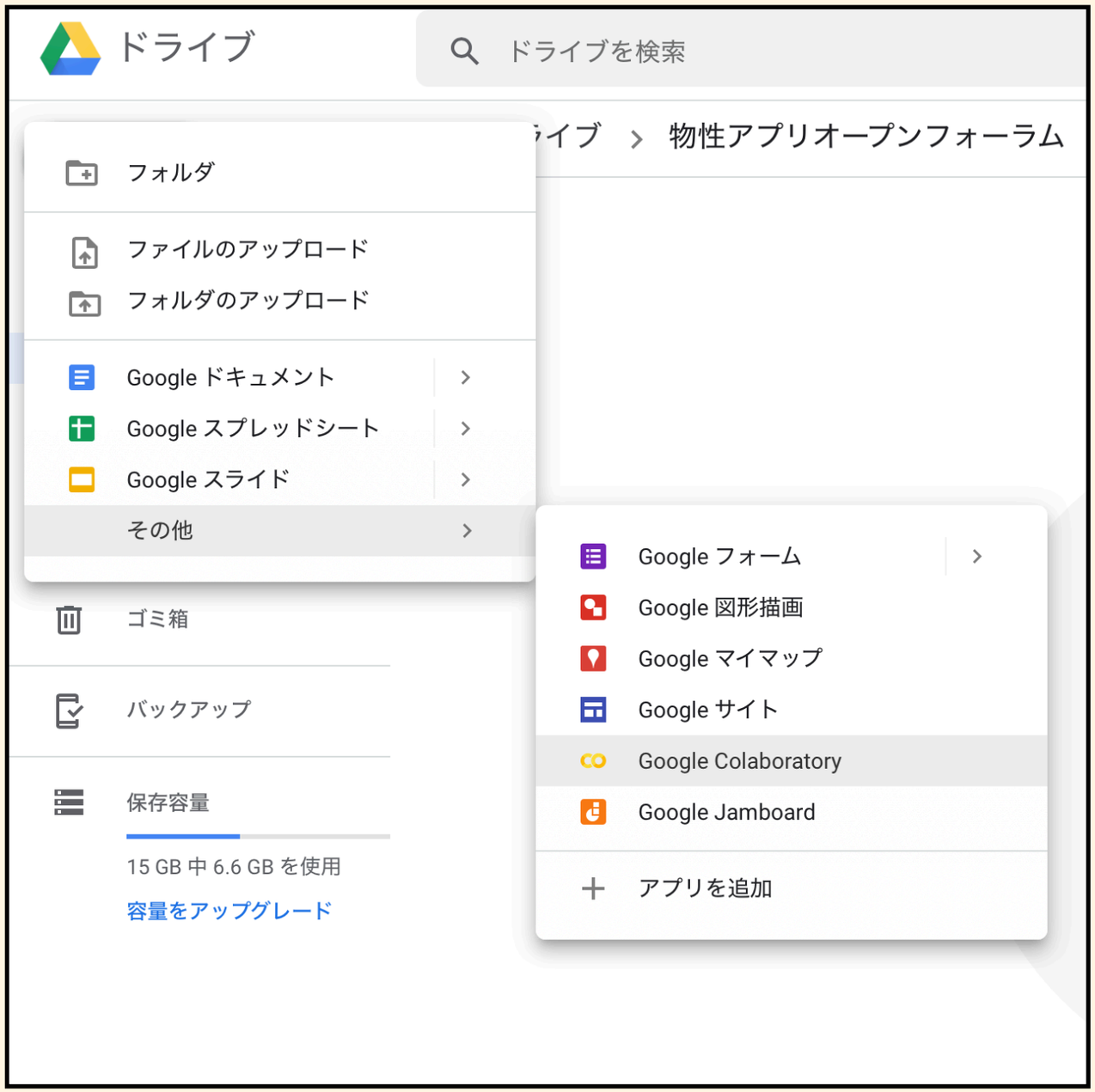
- Googleのアカウントにログイン後、Google Driveを選択

① フォルダのドロップダウンメニュー：「アプリで開く」→「アプリを追加」を選択。

② 「Google Colaboratory」と検索して接続。

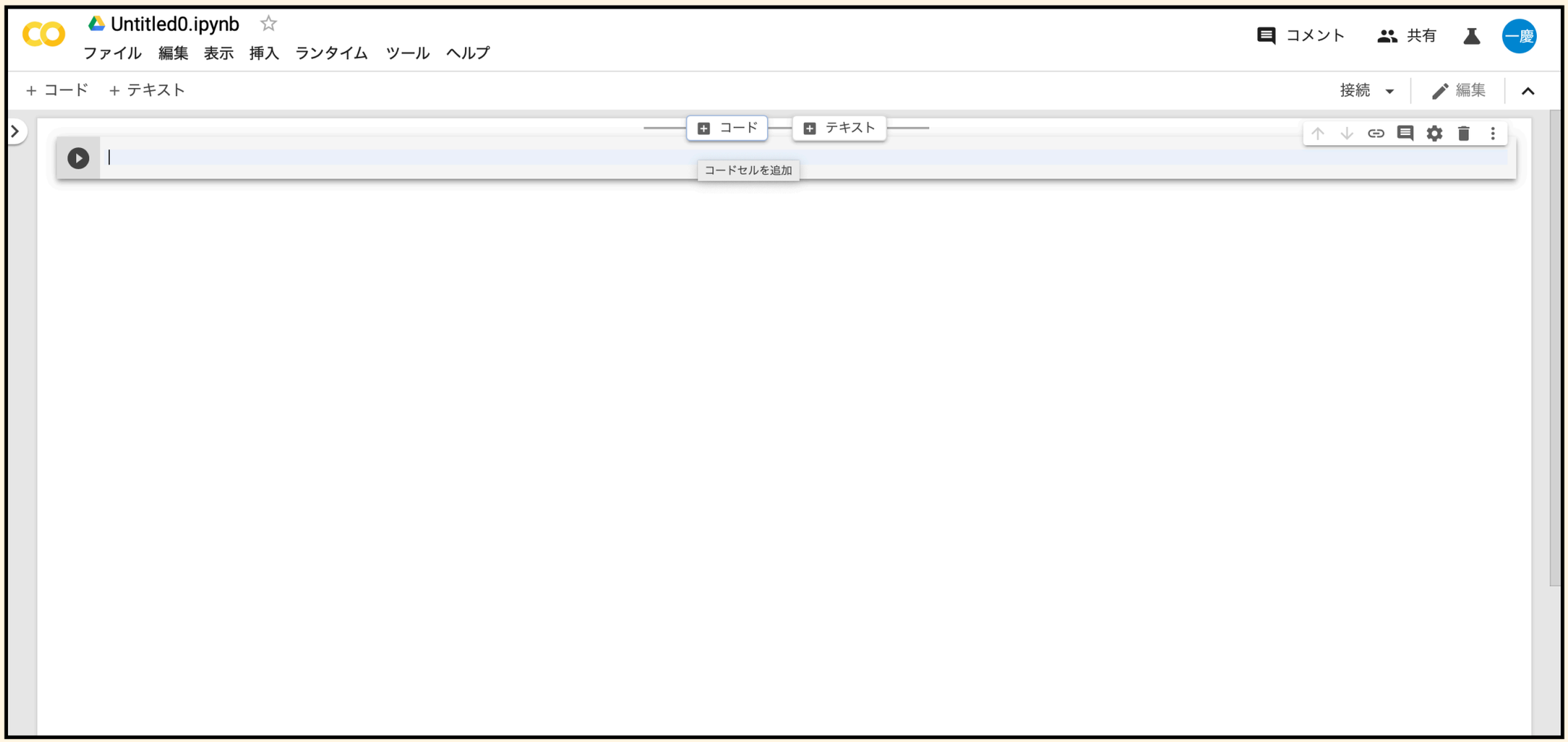
Google Colaboratoryの始め方 (2)

③ 「新規」をクリックして、「その他」-「Colaboratory」を選択。



Google Colaboratoryの始め方 (3)

④ 以下の画面が立ち上がって利用可能に！



コード：pythonのコードを記入 (Shift + Enterで実行可能)

テキスト：メモ用(コード共有の際などに利用)。

Google Colaboratoryの始め方 (4)

コード：pythonのコードを記入 (Shift + Enterで実行可能)

テキスト：メモ用(コード共有の際などに利用)。

Google Colaboratoryの始め方 (5)

1. シェルコマンドの実行は!{コマンド}で実行

ex.) pip installでパッケージをインストール

!{pip install *package_name*}

2. カレントディレクトリの変更 (マジックコマンドを実行)

ex. ）%cd {*directory_name*}

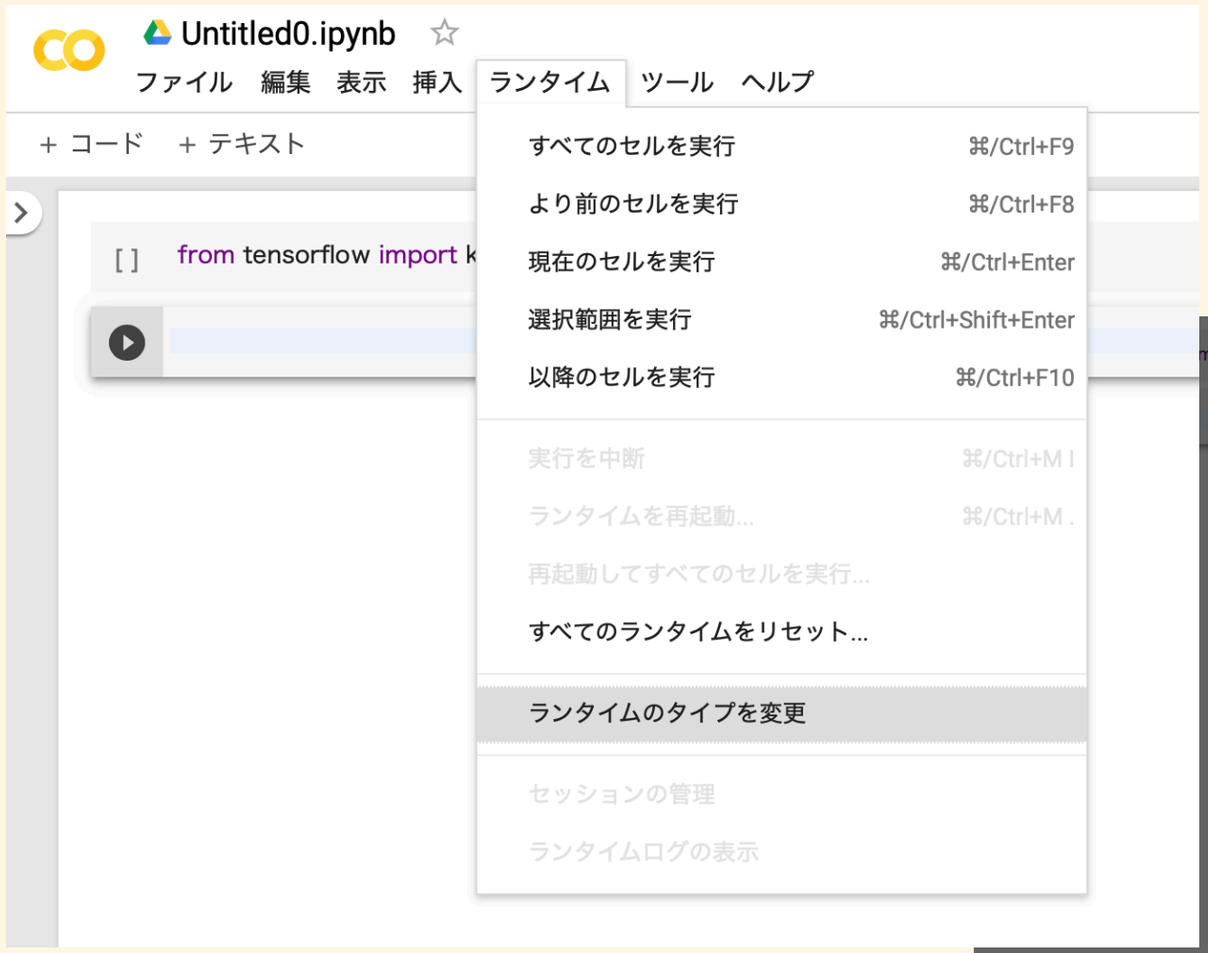
3. シェルコマンドのみ実行するセルの作成

ex.)
%%bash
pip install *package_name*
cd *directory_name*

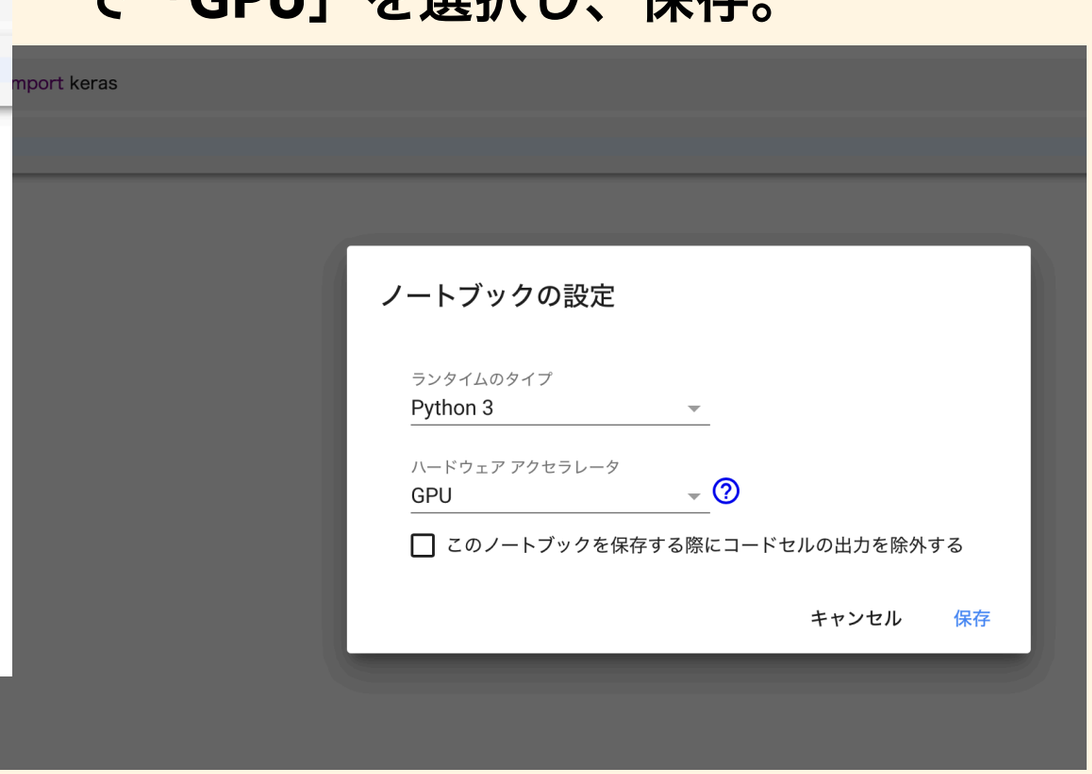
Google Colaboratoryの始め方 (6)

GPUの有効化

1. 「ランタイム」- 「ランタイムのタイプを変更」を選択



2. 「ハードウェアアクセラレータ」で「GPU」を選択し、保存。



3. GPUがアサインされていることを確認

```
[2]  import tensorflow  as tf
     tf.test.gpu_device_name()

     '/device:GPU:0'
```

Google Colaboratoryの始め方 (7)

ファイル読み込みと保存

1. ローカルからのアップロード・ダウンロード
    1. ファイルブラウザ

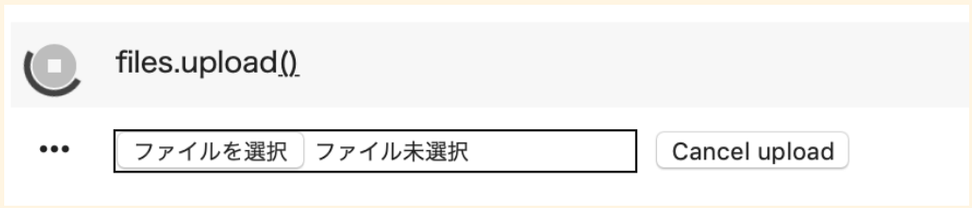
        左ペインの「ファイル」を選択して、「アップロード」を実行。
    2. コマンド

(1) google.colabからfilesをインポート

```
from google.colab import files
```

(2) アップロード

files.upload()をタイプして、対象ファイルを選択。



(3)ダウンロード

files.download (“file_name”)でダウンロード。

Google Colaboratoryの始め方 (8)

2. ランタイムとGoogleドライブの接続

1. 右のコマンドを実行

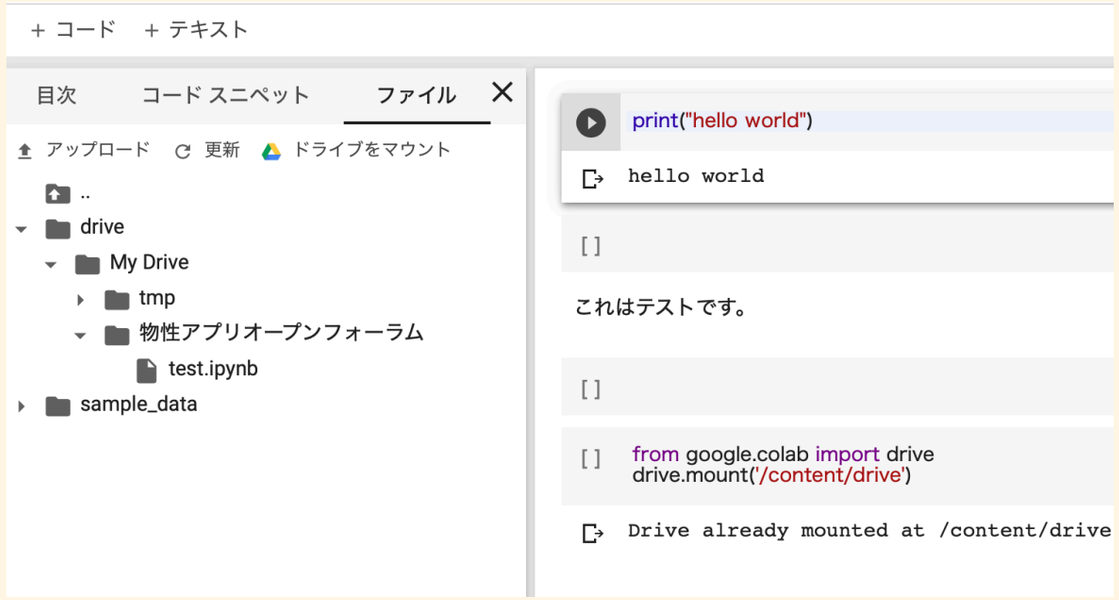
```
[5]  from google.colab import drive
     drive.mount('/content/drive')

     Go to this URL in a browser: https://accounts.google.com/

     Enter your authorization code:
     ··········
     Mounted at /content/drive
```

2. Webで認証+コード貼り付け

→ **/content/drive/My\ Drive/ にGoogle Driveのディレクトリが接続。**



左ペインの「ファイル」から確認可能。

Google Colaboratoryを試す (1)

「初心者のためのTensorFlow 2.0」に従って遊んでみる。

https://www.tensorflow.org/beta/tutorials/quickstart/beginner

1-9の数字の識別を行うプログラム

1. ライブラリのインポート

```
from __future__ import absolute_import, division, print_function, unicode_literals

!pip install -q tensorflow-gpu==2.0.0-beta1
import tensorflow as tf
```

```
     |████████████████████████████████| 348.9MB 40kB/s
     |████████████████████████████████| 501kB 63.2MB/s
     |████████████████████████████████| 3.1MB 53.8MB/s
```

Google Colaboratoryを試す (2)

2. MNISTデータセットのダウンロード

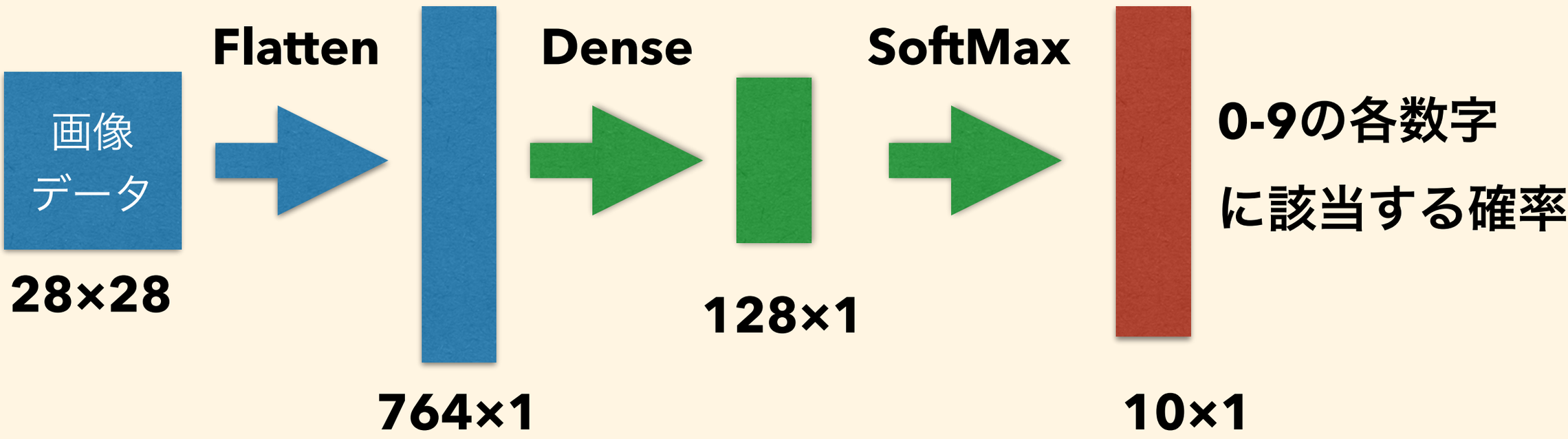
ピクセル値(0~255) を整数から0~1のfloat32に変更。

```
[9]  mnist = tf.keras.datasets.mnist

     (x_train, y_train), (x_test, y_test) = mnist.load_data()
     x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.n
11493376/11490434 [==============================] - 0s 0us/step
```

3. モデルの作成



Google Colaboratoryを試す (3)

3. モデルの作成

```
[10]  model = tf.keras.models.Sequential([
        tf.keras.layers.Flatten(input_shape=(28, 28)),
        tf.keras.layers.Dense(128, activation='relu'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(10, activation='softmax')
      ])

      model.compile(optimizer='adam',
      loss='sparse_categorical_crossentropy',
      metrics=['accuracy'])
```

4. 計算の実行

```
[11]  model.fit(x_train, y_train, epochs=5)

      model.evaluate(x_test, y_test)
```

Google Colaboratoryを試す (4)

```
[11] model.fit(x_train, y_train, epochs=5)

     model.evaluate(x_test, y_test)
```

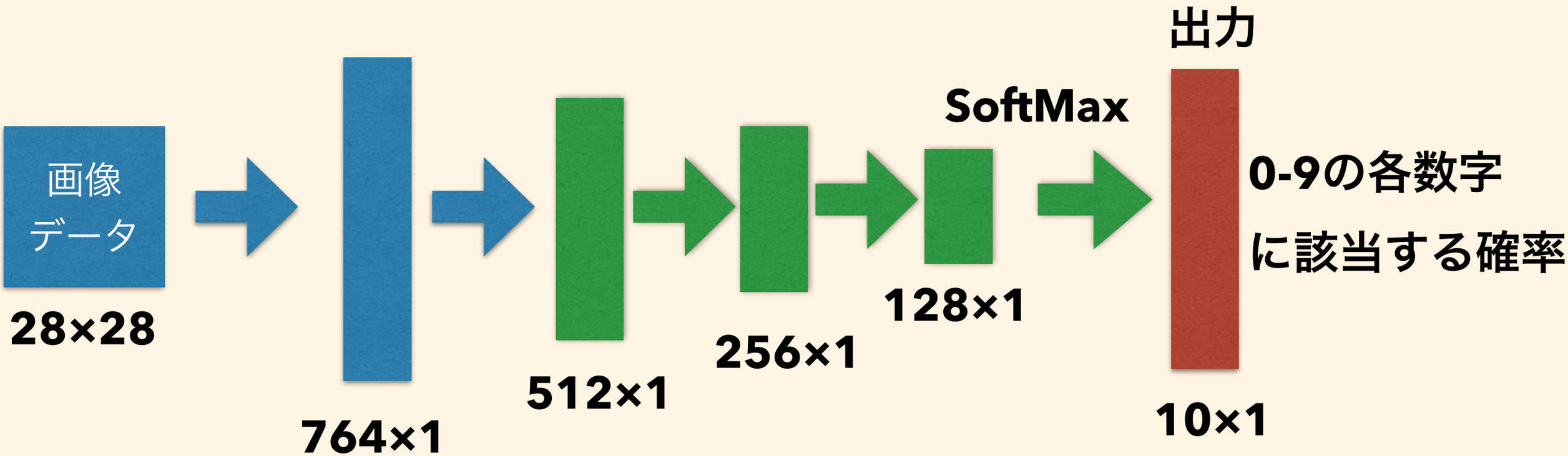
```
WARNING: Logging before flag parsing goes to stderr.
W0825 00:54:06.464202 139970857625472 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/math_
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 60000 samples
Epoch 1/5
60000/60000 [==============================] - 6s 101us/sample - loss: 0.3011 - accuracy: 0.9133
Epoch 2/5
60000/60000 [==============================] - 5s 84us/sample - loss: 0.1434 - accuracy: 0.9573
Epoch 3/5
60000/60000 [==============================] - 5s 83us/sample - loss: 0.1085 - accuracy: 0.9667
Epoch 4/5
60000/60000 [==============================] - 5s 85us/sample - loss: 0.0891 - accuracy: 0.9724
Epoch 5/5
60000/60000 [==============================] - 5s 84us/sample - loss: 0.0779 - accuracy: 0.9756
10000/10000 [==============================] - 0s 49us/sample - loss: 0.0767 - accuracy: 0.9767
[0.0766623124226462, 0.9767]
```

1エポック約6[s]、精度は最終的に97.67%になった。

(ちなみに、CPUで実行してもこの程度だと実行時間に変化なし)

Google Colaboratoryを試す (4)

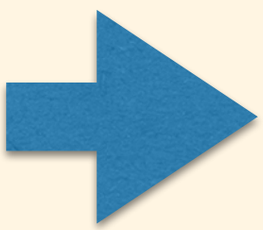
中間層を3層にして解析し、計算時間を計測。



CPU: 1エポック約13[s]、GPU: 1エポック約8[s]

Google Colaboratoryのノートブック

1. 立ち上げられるランタイムの数
   Google アカウント一つにつき、GPUありとなしで1つずつ。
2. 複数のノートブックを立ち上げても、
   ランタイム条件(GPUあり/なし)が同じ設定ならば、
   同じインスタンスにつながる。



計算させるノートブックと同じ条件で
ノートブックを立ち上げることで、
計算状況の確認(メモリやディスク残量など)が可能。

おまけ～東大松尾研データサイエンス講座

東大松尾研データサイエンス講座

https://weblab.t.u-tokyo.ac.jp/gciデータサイエンティスト育成講座・演習コンテ/

GCIデータサイエンティスト育成講座

東京大学で開催中のグローバル消費インテリジェンス寄附講座におけるデータ解析パートのコンテンツです。

全15Chapterを通し、データのセットアップから解析、可視化やモデルのアウトプットまでを一気通貫で学習できます。

DOWNLOAD

1. JupyterNoteBookの使い方とPythonの基礎

2. Numpy・Scipy・Pandas・Matplotlibの基礎

3. 実際のデータを使った記述統計学と回帰分析

など全15章のチュートリアルがjupyter notebook形式で配布(CC license)。

1章あたり4時間程度のコンテンツ(らしいです)。

おまけ～グローバル消費インテリジェンス寄付講座とは？

狙い：東京大学に世界最先端のプラットフォームを創設し、人材育成および学問分野の確立を加速化する。

1. 教育プログラム(社会人向けもあり)

全15回のチュートリアル講座
前項であげた内容をレクチャー。
終了後は認定証を贈呈。

2. 企業との共同研究

データの集積・分析に長けており、
高い関心を持っている各業界の
リーディングカンパニーと共同して
研究を行い、消費活動の理解を促進。

参画企業一覧

CCC dwango FIELDS IGPI 株式会社 経営共創基盤 KPI Solutions LAWSON
Panasonic RECRUIT wellness intage

協力機関

unidam

運営団体

GROW 経済産業省

まとめ

- 物性アプリオープンフォーラム
    - 毎月の予定で開催予定→ぜひ参加を！
- GPU導入に関して
    - 本格的に研究する場合はスパコン利用がよさそう。
    - pythonを使用していて気軽に試したいのであれば、Google Colabがおすすめ。
- Google Colabの使用
    - Googleアカウントがあれば簡単に使用可能。
    - ただし、最大12時間の制約があるので注意。

参考文献 図解速習 DEEP LEARNING , 初心者のためのTensorFlow 2.0