イーグロース株式会社

画像処理向け深層学習の動向と医用画像への応用

今西　勁峰

今西　勁峰　Keiho Imanishi

- 所属

  -イーグロース株式会社　代表取締役
- 大学の専攻

  -データベース（学士）

  -医用画像処理、医用VRアプリケーション（修士・博士）
- よく使う言語

  -C++、OpenGL 、CUDA（医用画像処理）

  -PHP、SQL (Webアプリ)

  -Python（数値解析、深層学習）

事業概要

- 医療画像、医用アプリケーション研究開発

  -医用画像処理

  - 放射線治療支援向けビューア
  - 診断・手術支援用2D・3D医用画像処理

  -院内業務システム

  - 予約、検査支援、文書・画像管理用データベース

  主要製品へは深層学習技術を組み込むことで機能レベルの向上を図る

- 技術コンサルティング

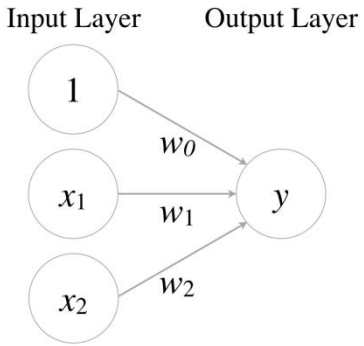
  -院内インフラ管理、システム・技術導入、スタッフ教育

画像処理向け人工知能

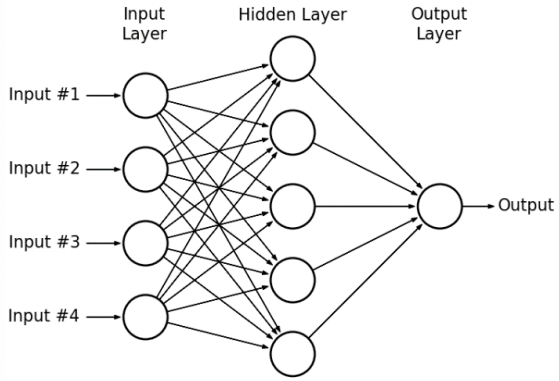
ニューラルネットワーク

- 歴史が古く、1950年前後に発表された形式ニューロンやパーセプトロンが基本
- 多層パーセプトロンの出現

  -線形分離から非線形分離へ



単純パーセプトロン
(線形分離可能問題)



多層パーセプトロン
（非線形分離可能問題）

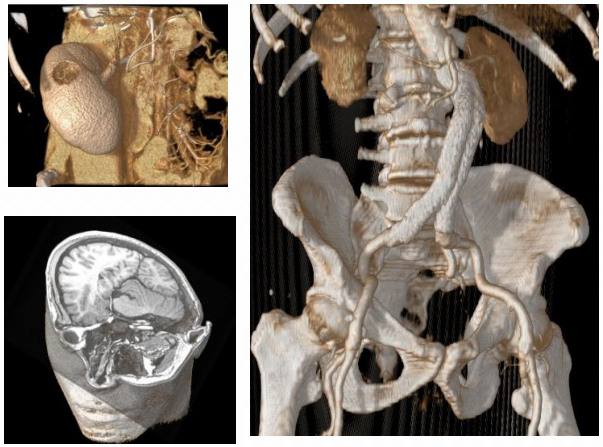
多層パーセプトロンは多くの複雑な問題でも解けるように思えたが、実際は計算量、チューニングの難しさからブームが廃れた

ディープラーニングの登場

- GPU(元は描画専用)の性能向上が後押し

-2003　GLSL言語策定

　ユーザが直接GPUへ命令し、多様なレンダリング結果を並列演算で高速表示。特に医療分野では数百万円のチップしか表現できなかった画像が数万円の汎用GPUで行えるようになった。



-2006　CUDAによる並列演算技術発表
(処理時間を数倍～数十倍短縮)

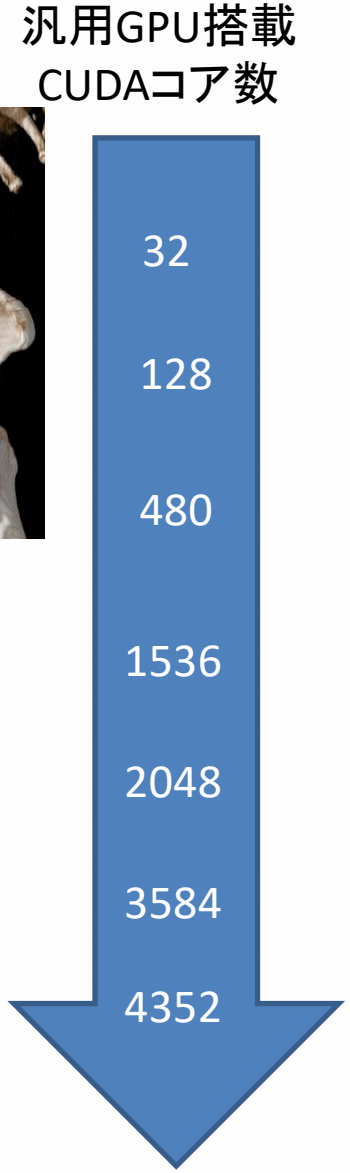
-2008～　数値解析分野へGPGPU技術が爆発的に普及

-2012　GPUを用いたCNNモデルALexNet(火付け役)が登場

…

様々な深層学習技術が次々発表される
識別、物体検出、領域分割、言語処理…

2019年現在GPUの並列処理コア数は4,000を超える



画像処理における「特徴抽出」手法

- 画素値に対する統計分布

  -画像の二値化法、コントラスト調整等に利用

- 主成分分析

  -次元削減や回転軸解析等に利用

- フィルター関数による特徴抽出

  -エッジ抽出

  -画像を「特徴量空間」に変換し、深層学習等に利用

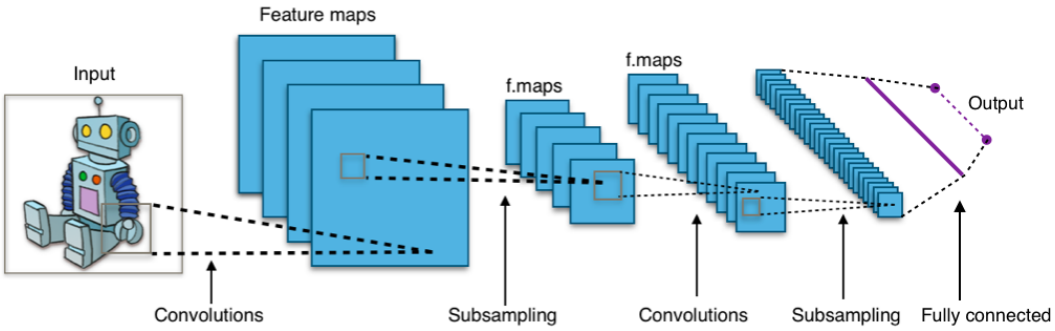
- スパースモデリング

  -画像を構成する「基底」の解析し、画像欠損の補間、ノイズ削減等に利用

畳み込みによる機械学習

- **Convolutional Neural Networks (CNN)**

  -画像から特徴量を抽出するフィルターを、ニューラルネットワークを利用して構築し、最適化する



[wikipedia]

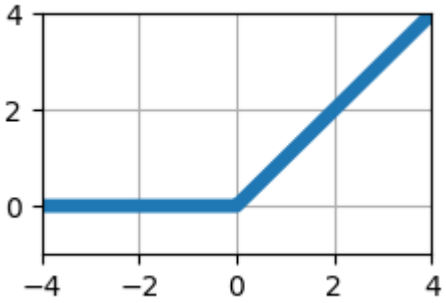
- **Deep Convolutional Neural Networks (DCNN)**

  -CNNを多段階かつ非線形関数によって階層化し、広域で複雑な形状に対応したネットワークを学習させる

GPU性能向上によってDCNNが現実的な解析手法となった

CNNがなぜうまく行くか

- ニューラルネットでは、入力データを高次元空間に配置し、分離できる境界線を探す(回帰)
- 弱い非線形関数でも、多層であれば複雑な形状が表現できる
  -上位層の出力に対し、下記のような関数でフィルタリング処理してから、下位層に伝える

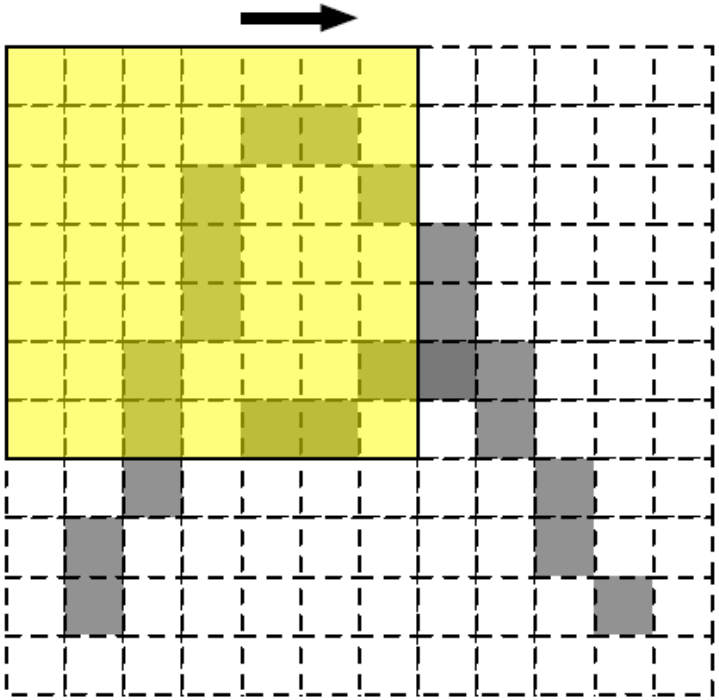


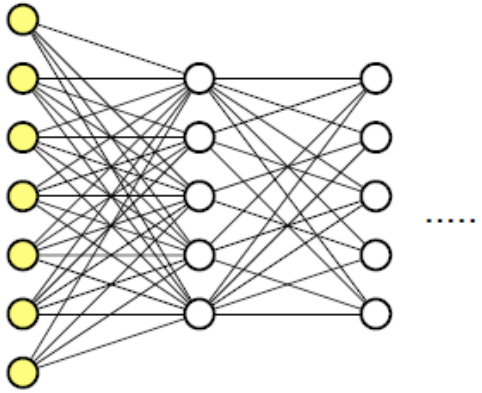
Relu

※非線形関数が多層に存在するDCNNになれば、
何を抽出しているのかは人間には理解できないと言われている

- 階層を重ねたフィルター処理によって画像内の任意の範囲を少ないリソースで特徴構成できる
- ニューラルネットで重み付けすることで特徴量空間の一部を拡大するような効果が生まれる

Convolutional Neural Networks



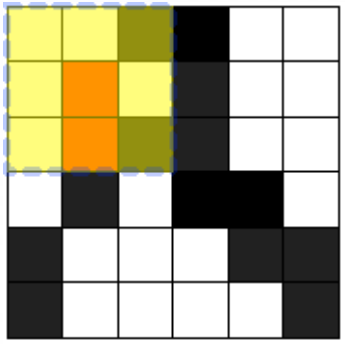


- 画像全体を入力層とするのではなく、領域を制限した「窓」を入力したネットワークを構築
- 「窓」を移動させて画像全体をスキャンすることにより、出力ノードの値に対応した特徴量空間に画像を変換
- 変換の過程が「折り畳み (convolution)」演算

畳み込み演算

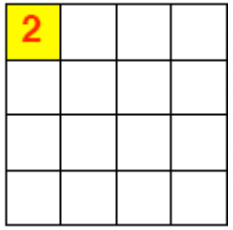
例：3x3ピクセルのフィルターを画像に適用させる

まず右上の3x3ピクセルにフィターの重みを掛けて足し合わせる。

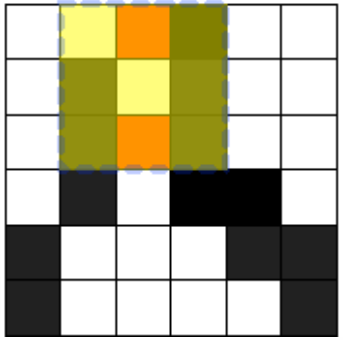


×

0 1 0
0 1 0
0 1 0

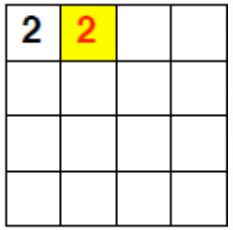


右隣の3x3に移動して同じ操作を行う。



×

0 1 0
0 1 0
0 1 0



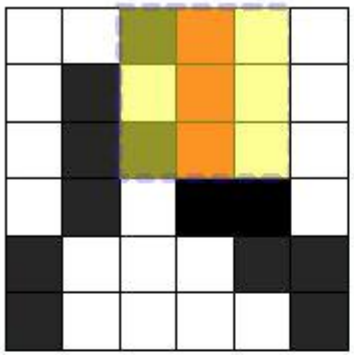
足し合わせた出力値を並べていく

3x3の中央の縦のピクセルを抽出する重みフィルター

畳み込み演算

例：3x3ピクセルのフィルターを画像に適用させる

右隣の3x3に移動して同じ操作を行う。

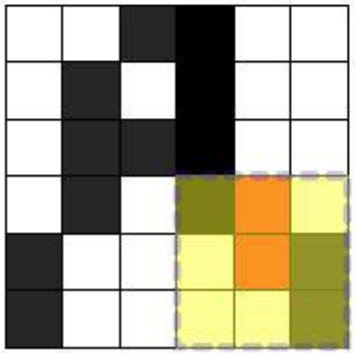


× 0 1 0
0 1 0
0 1 0

2	2	3	

.
.
.
.

画像をスキャンするように全体にフィルターを適応する

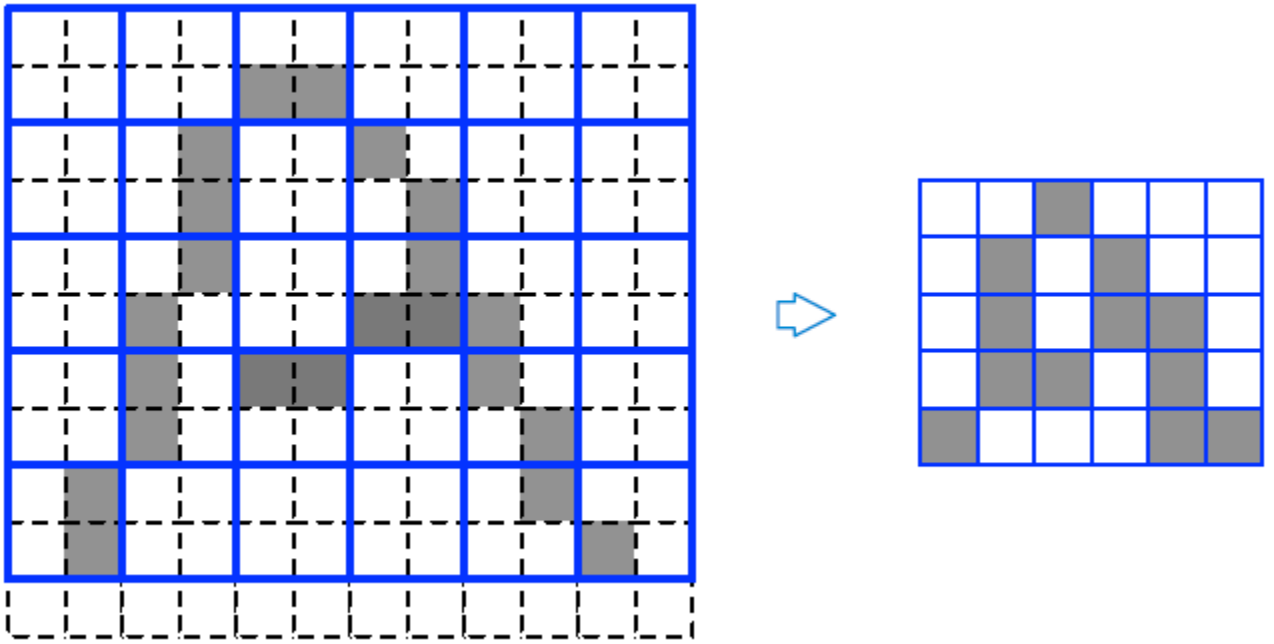


× 0 1 0
0 1 0
0 1 0

2	2	3	0
3	1	3	1
2	1	2	2
1	0	1	2

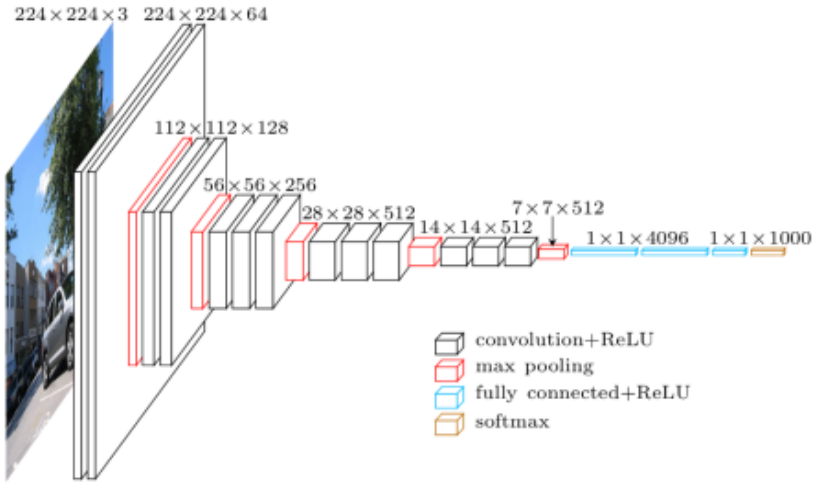
画像の特徴（この場合、縦のライン）に比例する出力が得られる。

Max-Pooling



- 単純にConvolutionを繰り返すと、変換した特徴空間の数に従って情報量が倍増していってしまう。
  → 情報量を抑える画像をダウンサンプリングする
  → 画像の細かいノイズに対して学習がロバストになる
- 画素を2x2、3x3などのエリアに分割し、最大値を選ぶmax-pooling、平均値を取るaverage-poolingなど

CNNによる教師あり学習



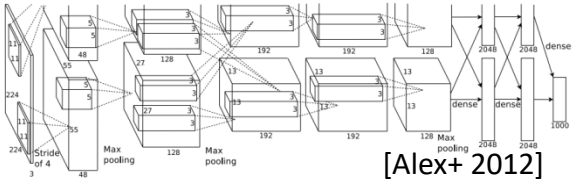
1. サンプルを入力し、結果を算出
2. 結果を評価し、教師ラベル(正解)との誤差を計算
3. 重みを調節し、1に戻る

Step2で得られる誤差がある程度収束するまでを一定回数(Epoch)繰り返す

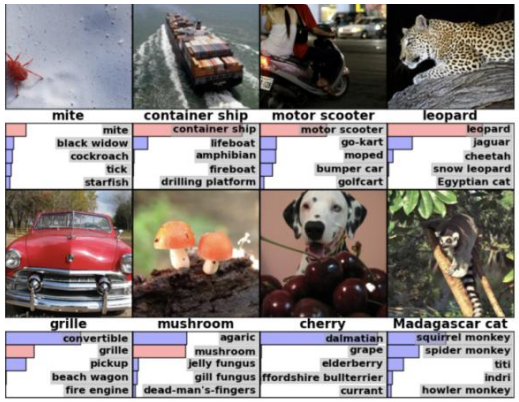
様々な目的関数を設けることで複雑な課題に対応可能

応用例

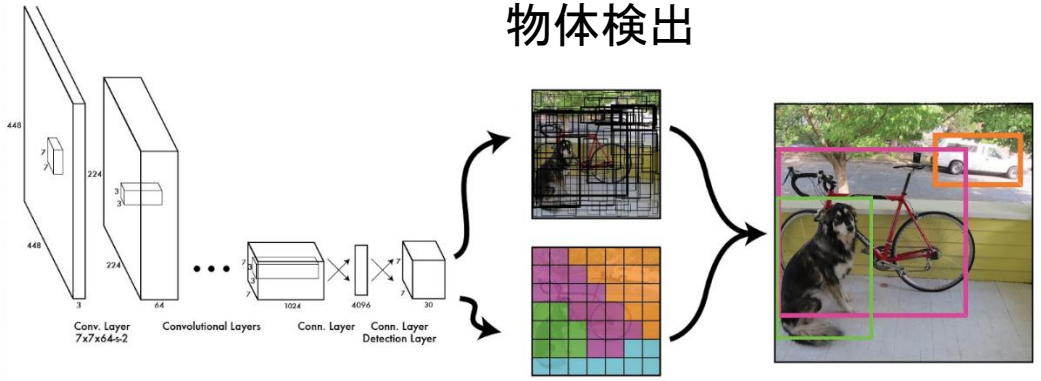
識別



[Alex+ 2012]



物体検出



[Joseph 2016]

領域分割

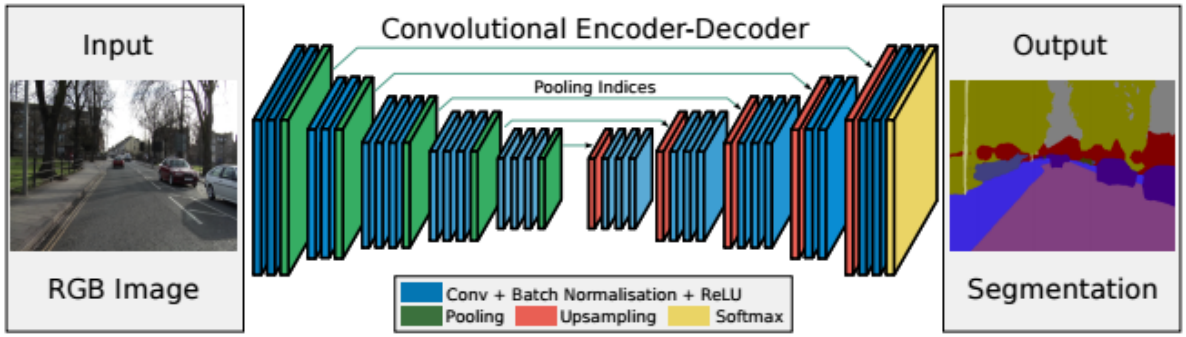
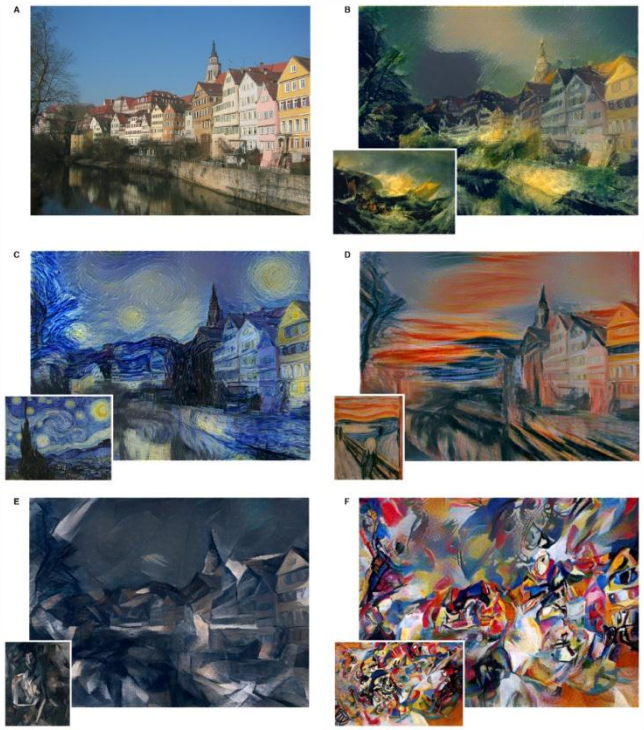


Fig. 2. An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final decoder output feature maps are fed to a soft-max classifier for pixel-wise classification.

[Vijay+ 2016]

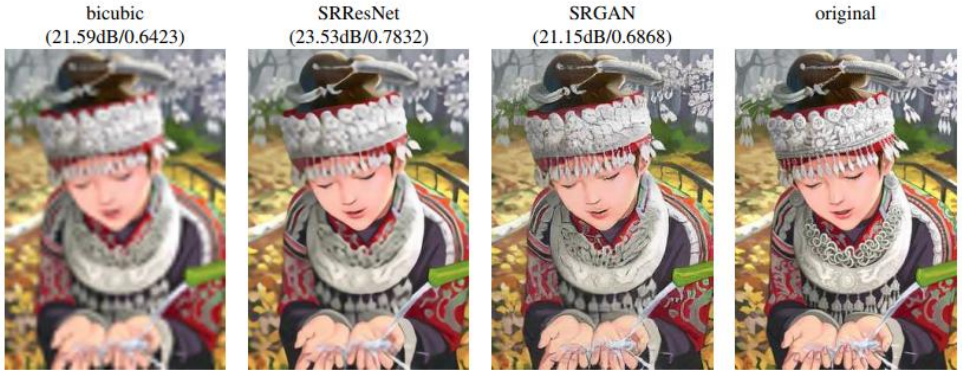
応用例

スタイル変換



[Leon+ 2015]

超解像



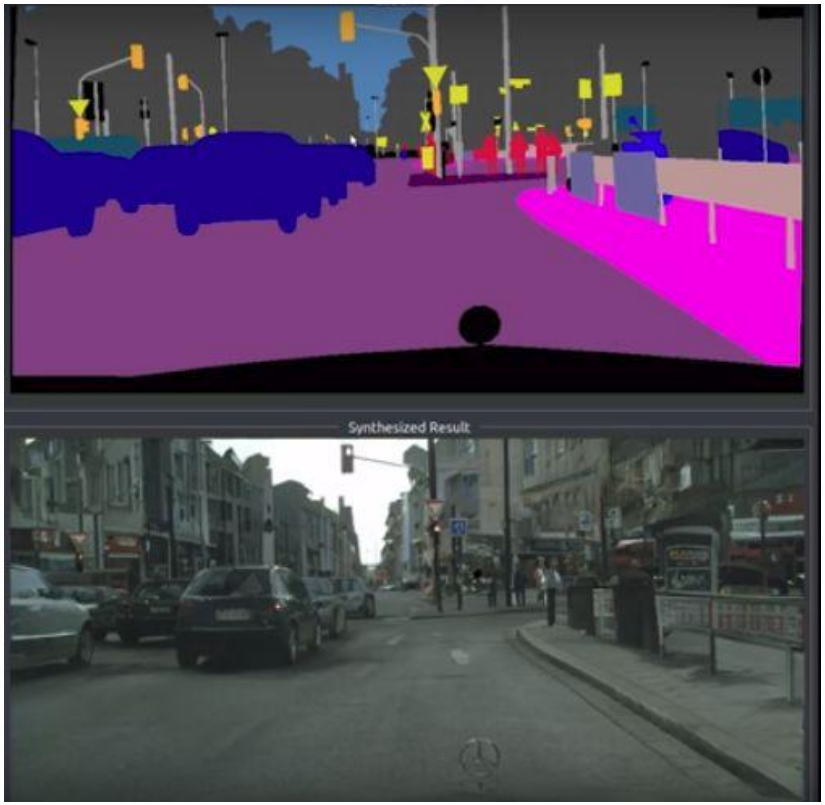
[Ledig+ 2017]

色付け

[飯塚+ 2016]

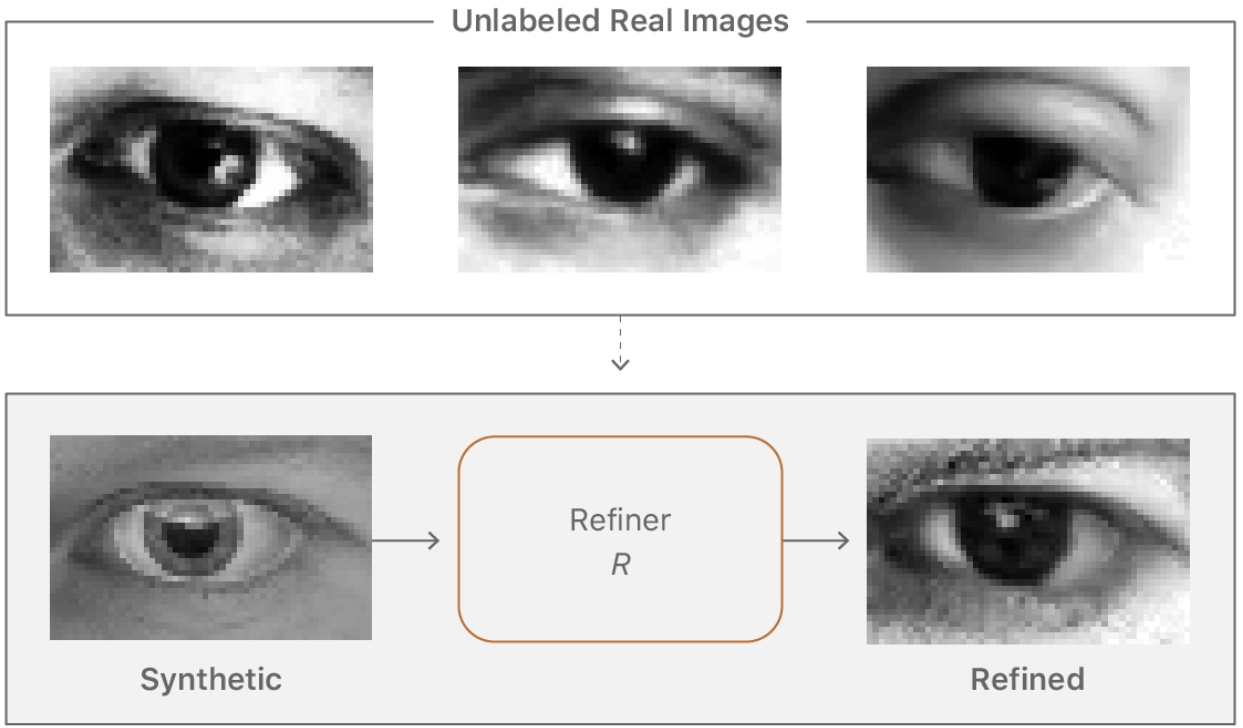
応用例(Conditional GAN)

ドメイン変換による画像合成
(ラベル情報を与えて、本物っぽい画像を擬似的に生成)[Wang+ 2017]



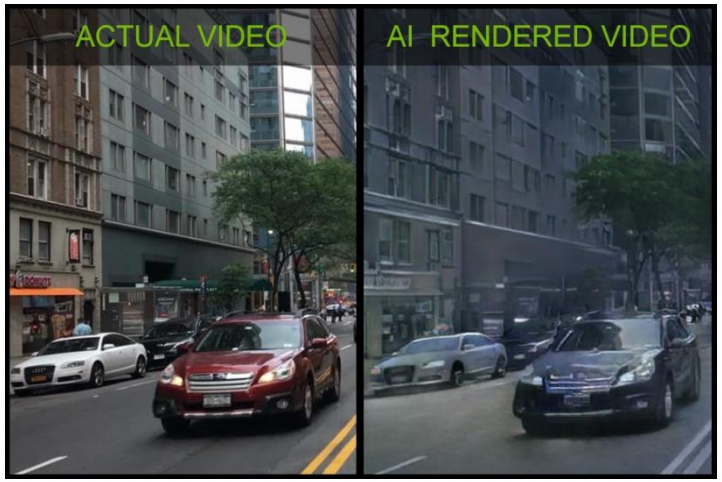
応用例(SimGAN)

元の画像とのL1ノルム距離を損失関数の正則項として与えるなどの技術を採用し、
合成画像の「写実性の向上」を目的としたGAN [Apple 2017]



応用例

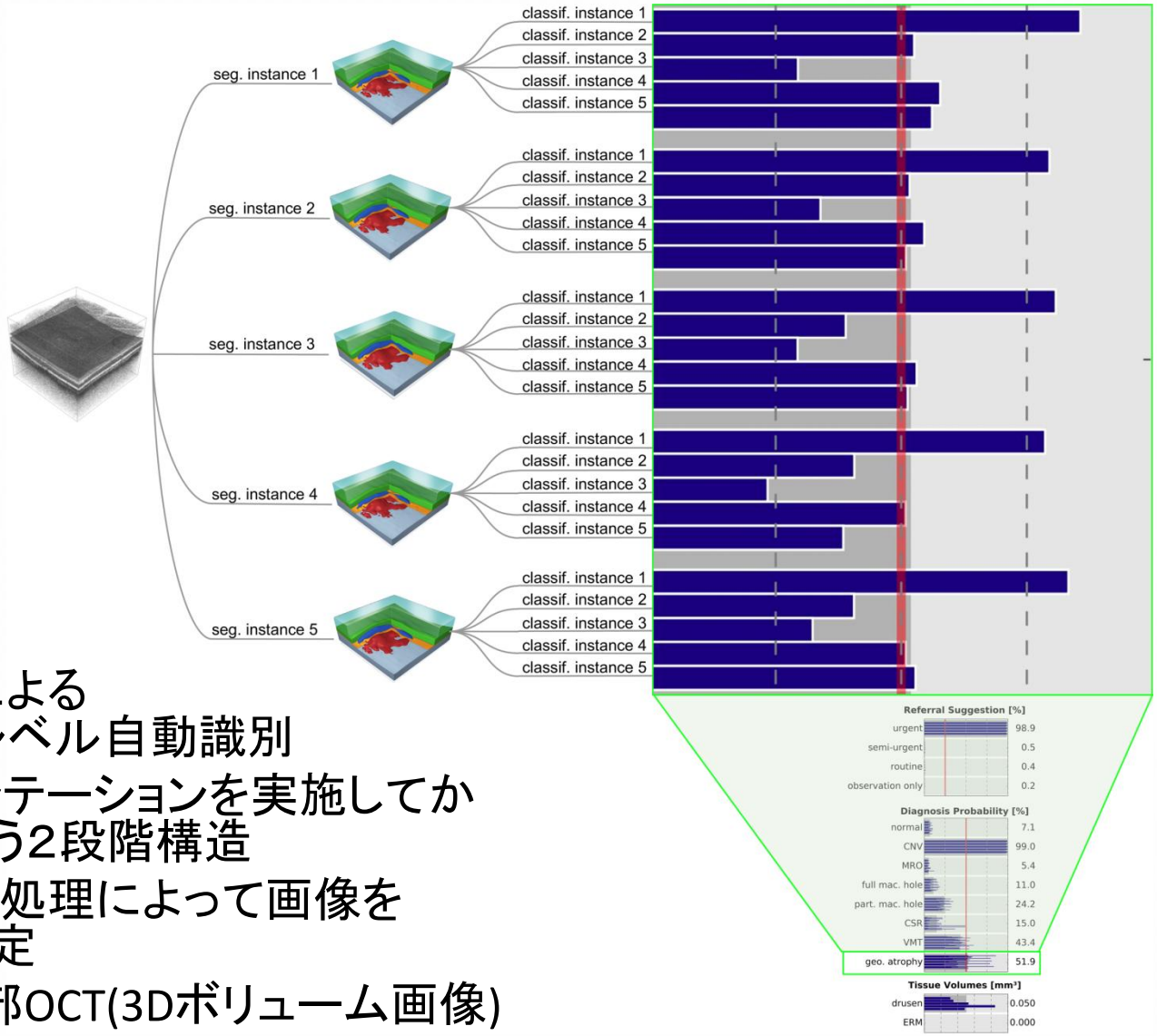
実映像から3DCGの仮想環境を自動生成するGAN [NVidia 2018]



セグメンテーション
ネットワークの処理
を経由

Pix2pixのGAN
によるVR映像生成

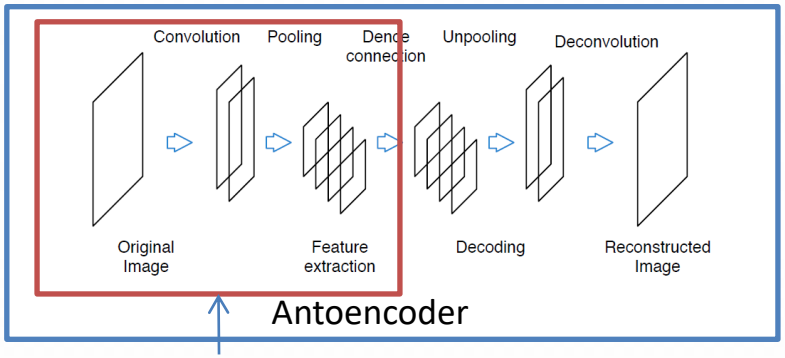
医用画像への適用例(分類モデル)



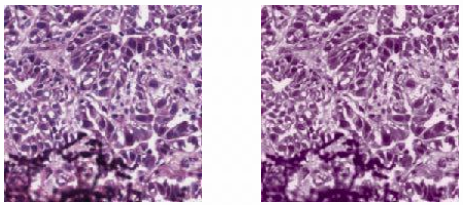
- 深層学習モデルによる
  黄斑病変の診断レベル自動識別
- 3D-Unetでセグメンテーションを実施してから
  らの識別判定という2段階構造
- セグメンテーション処理によって画像を
  抽象化してから判定
- 対象データ:網膜部OCT(3Dボリューム画像)

[Fauw+, 2018]

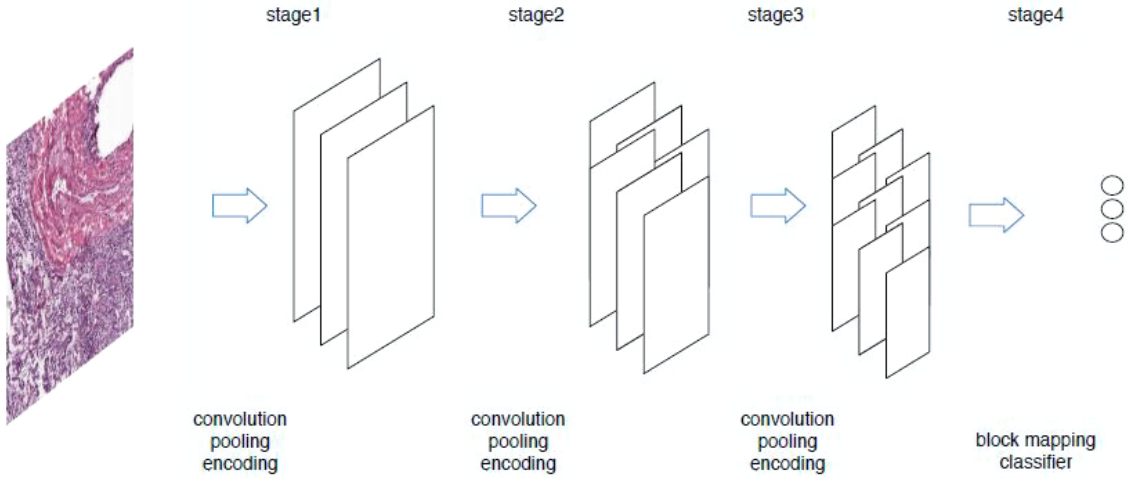
医用画像への適用例(分類モデル)

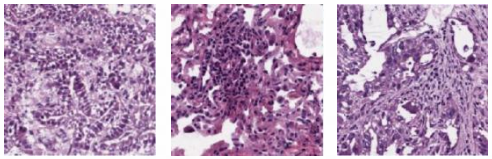


画像の符号化モデル



Input　　Reconstraucted



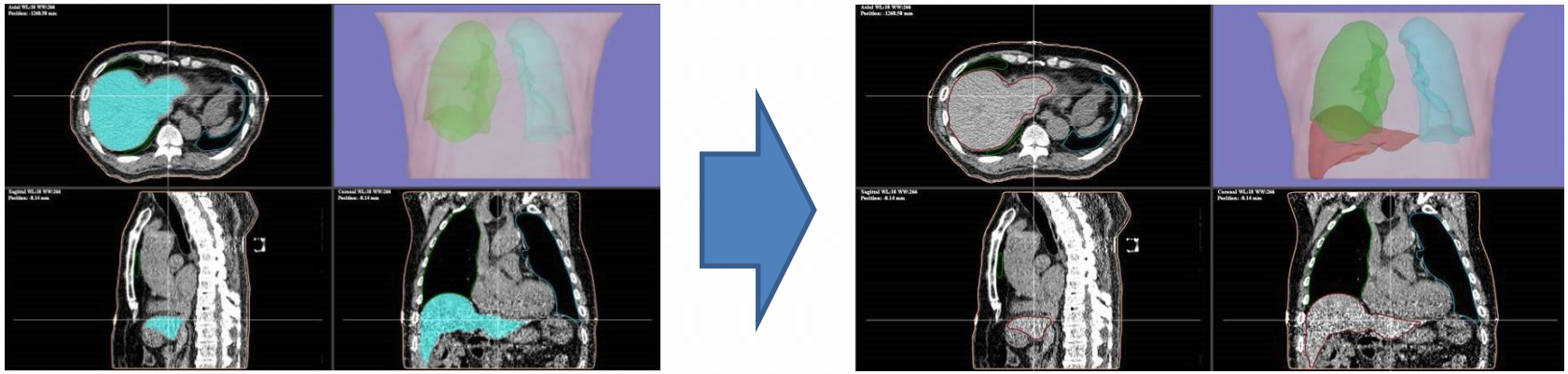


サブタイプ分類(左からTRU, PP, PI)

[Ono, 2018]

- 高解像(2048x2048)病理画像から肺腺癌の遺伝子発現サブタイプ分類
- Autoencoderによる事前学習で精度良く符号化可能なモデルを生成
- 2048pxでは98.9%の正答率で分類可能

医用画像への適用例（セグメンテーション）



[e-Growth, 2018]

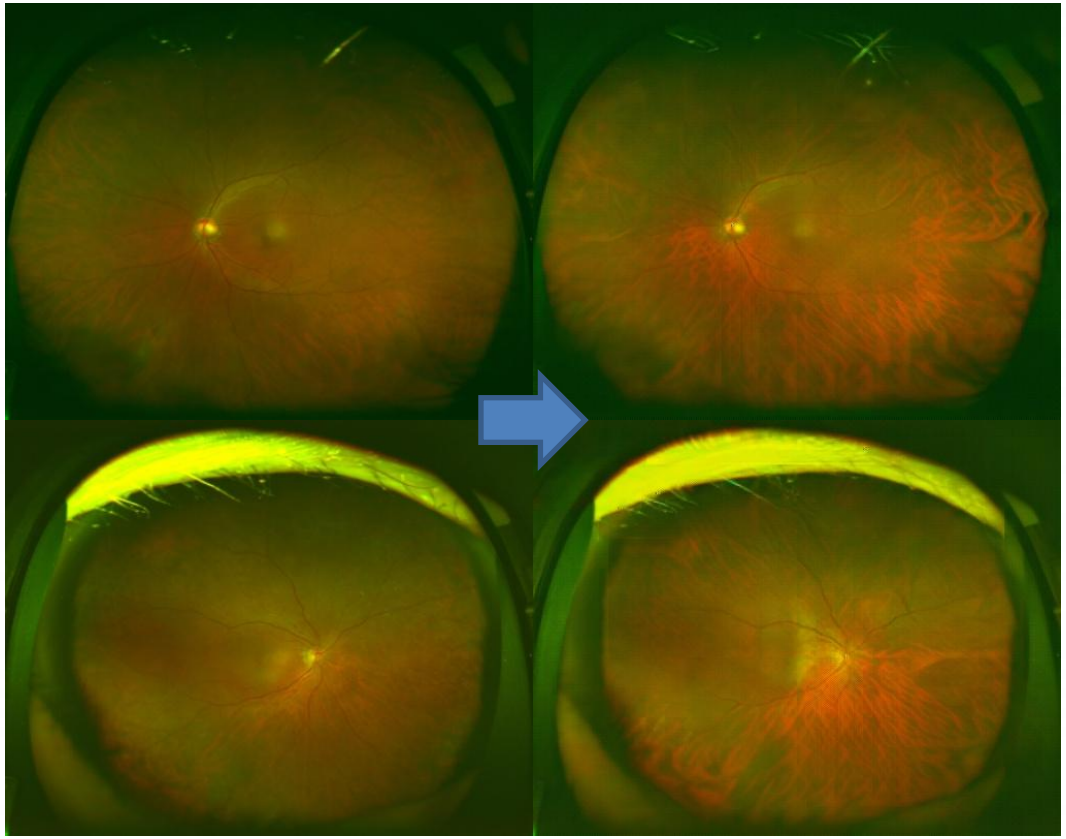
- CTから関心領域の3D領域情報自動抽出
- 放射線治療計画画像の関心領域輪郭データを利用
- 精度評価(3D領域におけるDice係数)

  -体表: 99%

  -肺野:96%

  -肝臓:93%

医用画像への適用例（GAN）



Normal　　Tigroid

[e-Growth, ツカザキ眼科, 2018]

- GANによる眼底画像への特徴(病変部等)付加
- 事前に実画像による訓練済みモデルを97.7%で騙すことに成功

放射線治療計画画像 DICOM-RTデータの 深層学習活用

放射線治療計画画像の活用

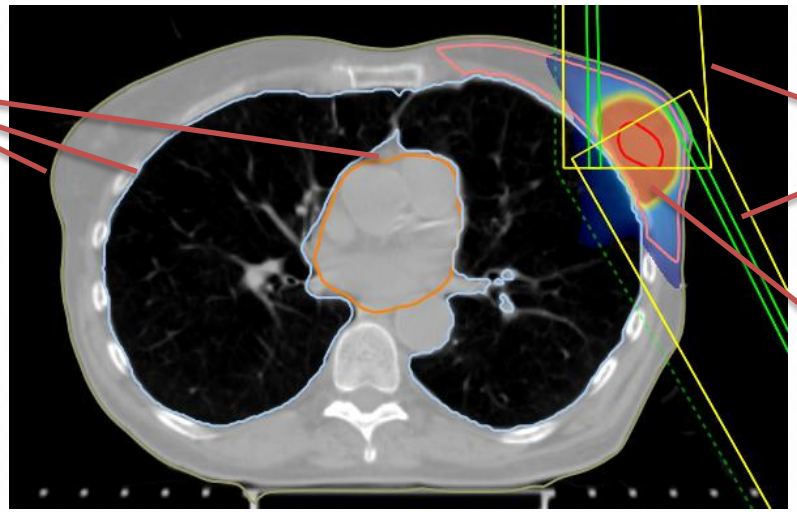
- DICOM-RT

  -DICOMの拡張フォーマット

  - CT/MRI
  - RT Structure Set(臓器や腫瘍等関心領域(ROI)の2Dコンター)
  - RT Plan(Beam方向、強度やMLC形状等の照射情報)
  - RT Dose(線量分布)

  -専門医によってマニュアルで関心領域が定義されてきた貴重な資源

  -放射線治療を行っている施設であればどこでも多く保持している



画像と教師データがセットとなって保持されており、しかも大量にある → 深層学習向き！

データ利用にはいくつか難点がある

- 教師用3Dデータとして扱うには

  -CT/MRIの3Dデータ

  - DICOMタグを読み取って3D再構成することが必要

  -関心領域の3Dデータ

  - RT Structure Setは2Dのコンター(線)情報しか持たない

    -RT Structure SetのDICOMタグを読み取って2Dコンター→2D領域→3D領域への変換ノウハウが必要

  -線量分布の3Dデータ

  - 位置合わせ

    -Doseの原点はCT/MRIの原点とは同じでない

  - 3D補間処理

    -Doseボリュームデータは CTの解像度とは同じでない(もっと疎)

Growth RTVの紹介

- イーグロース社製DICOM/DICOM-RT三次元医用画像解析ツール

  -DICOM-RT解析エンジン・Viewer
  - RTデータ解析エンジンは京大病院全電子カルテ端末での導入実績あり

  -深層学習による臓器抽出機能
  - 現在は肺、肝臓、腎臓など腹部臓器が中心

  -プログラマブルなPython連携プラットフォーム搭載
  - 2019.02よりベータ版配布開始
  - Python実行環境標準搭載(ユーザ側でPython環境のカスタマイズは自由)
  - 数行のPythonコードでDICOM-RTの様々な3Dデータアクセス、書き出し可
  - 双方向のデータ交換を実現

    -例: DICOM-RTデータからROIの取り出し⇔新たなROIをDICOM-RTデータへ追加定義

  -現時点でサポートOSはWindows 7/8/8.1/10(64bit)のみ

DICOM / DICOM-RT形式の取扱を知らなくても、
本プラットフォームを介して3D画像データを簡便に取得加工して研究開発に利用可能

コード例

CTの3Dデータ取得例

```
grtv = pyGRTV()                                                    # インスタンス生成
grtv.readRTDir(“c:/testdata/”)                                     # DICOM-RT読み込み
ret, img_short = grtv.readGRTVRaw(grtv.GRTV_DATA_TYPE_IMAGE)       # 3Dデータ取得
grtv.writeRaw(“c:/testdata_ct.raw”, img_short , dataType=np.int16) # ファイル保存
```

関心領域の3Dデータ取得例

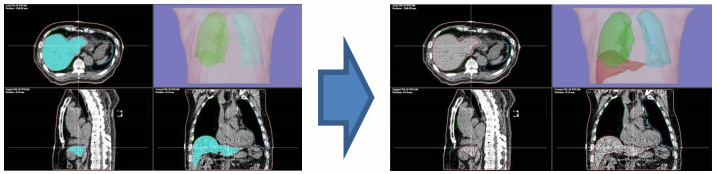
```
grtv = pyGRTV()                                                    # インスタンス生成
grtv.readRTDir(“c:/testdata/”)                                     # DICOM-RT読み込み
ret, roi_uint8 = grtv.readGRTVRaw(grtv.GRTV_DATA_TYPE_ROI, dataID=0)  # 0番目のROIを取得
grtv.writeRaw(“c:/testdata_roi_0.raw“, roi_uint8, dataType=np.uint8)  # ファイル保存
```

関心領域の3Dメッシュ(PLY形式 or STL形式)の一括書き出し例

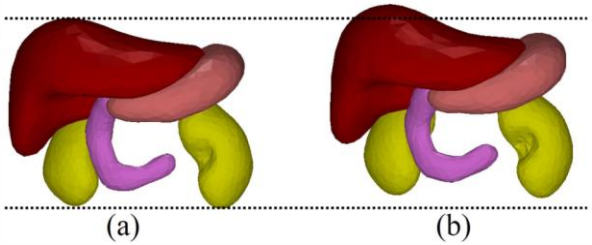
```
grtv = pyGRTV()                                                    # インスタンス生成
grtv.readRTDir(“c:/testdata/”)                                     # DICOM-RT読み込み
roiList = []
ret = grtv.getROIList(roiList)                                     # 定義済みROIリスト取得

#ROIごとにPLY形式で書き出し
for roi in roiList:
    meshPath = "c:/mesh/%s.ply" % (roi["roi_name"])
    ret = grtv.exportROIMesh(dataID=int(roi["roi_index"]), meshPath=meshPath, meshType=grtv.GRTV_MESH_TYPE_PLY)
```

活用例

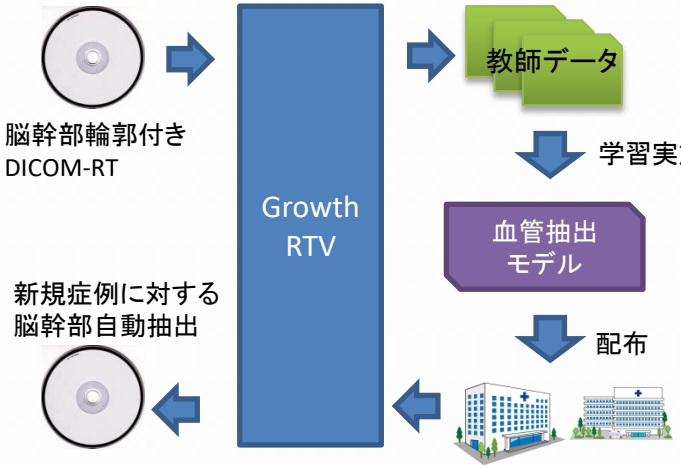
- 臓器自動領域抽出、腫瘍自動認識向け深層学習用データの大量生成
  - 活用製品例
    「Growth RTV」の臓器抽出機能(関連知財出願済)



- ROIの位置合わせ研究用データの大量生成
  - 活用対象研究例(スパースモデリング)
    呼吸性変位を表現可能な複数臓器の統計的変位モデル構築の試み, 京都大学, 2018



- 「Growth RTV」の臓器抽出機能の拡張(プラグイン機能)
  - 活用例
    Growth RTVでは対応できていない臓器または腫瘍に対し
    1. ユーザ側にDICOM-RTデータさえあれば、本基盤を通して深層学習用データを生成し学習
    2. 「Growth RTV」の導入施設へ学習済みモデルの提供
    3. 提供されたモデルで臓器領域を予測し、「Growth RTV」経由で新規ROIとしてDICOM-RTへ反映



研究(大学・企業)と検証(病院)実施の分業化を促進し、
研究協力対象の拡大や競争の活発化、研究実施ハードル軽減を実現

おまけ

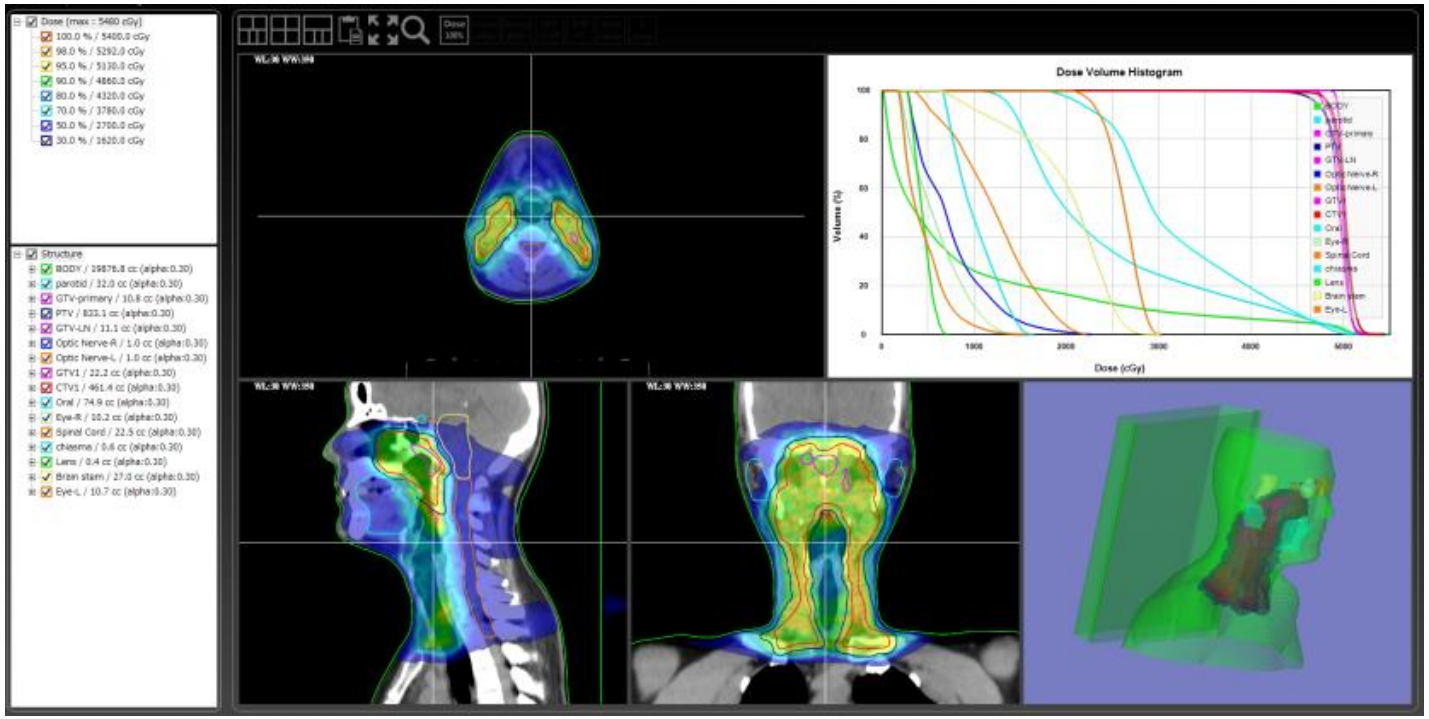
- Growth RTV Edu

  -研究・教育・医療機関の関係者向けの無償版DICOM-RT Viewer

  -有償版「Growth RTV」と同等の表示機能

  - Image, Structure(2D, 3D), Plan(CPごと), Dose, DVH
  - Beam's Eye View, DRR, 体表照射形状

  -DICOM Q/R、臓器抽出、Python連携機能は非搭載

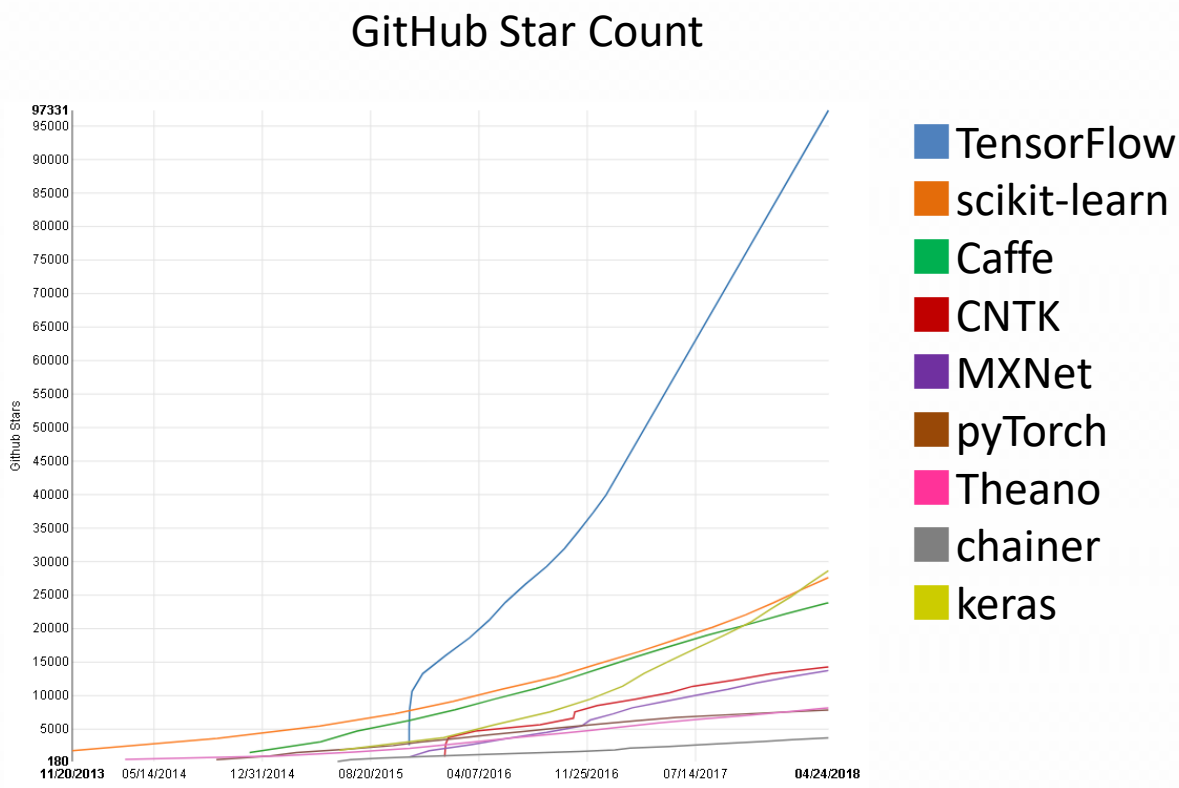


実習に向けての導入説明

ディープラーニング実行環境

フレームワーク

- Google, Amazon, Microsoft, Facebookなどを始め、ディープラーニング向けフレームワークが乱発



2015年にGoogleから発表されたTensorFlowが圧倒的な人気

オススメな環境（非熟練者向け）

- 自分で構築

  -言語：Python 3.xx

  -フレームワーク: TensorFlow, Keras

  -OS: WindowsまたはLinux(Ubuntu)

  -GPU：NVIDIA社GeForceブランドチップの8GB以上がおすすめ
  (GTX 1070(8GB): 5万円, GTX1080Ti(11GB): 10万円)

  -既存PCにGPUを差し替えるだけでも動く

- 市販品のディープラーニング向けPCを購入

  -50万円～

自分で構築をオススメする。
価格面のみでなく、フレームワークのバージョンアップデートが頻繁で、
新しい技術を使いたい際は自分でトラブルシューティングする必要がある

いざ、インストール（例としてWindows編）

どこかのWebページをググって、その通りやってみる。

1. CUDAドライバーをNVIDIA社からダウンロードしてインストール
2. cuDNNをNVIDIA社からダウンロードして、参考しているWebページの言うとおりに設置
3. Anaconda(Python環境)をダウンロードしてインストール
4. Webページを参考にしながら、TensorFlowやKeras等をインストール

```
> conda create --name tensorflow python=3.5
> activate tensorflow
> conda install jupyter
> conda install scipy
> pip install tensorflow
> pip install tensorflow-gpu
> pip install keras
```

5. 簡単な学習用サンプルプログラムを走らせる
6. 結果は？

実は、結構の確率で動きません。。。

初心者が挫折するポイント

- 閲覧数の多いWeb記事ほど最新でない可能性が多く、最新製品のバージョンと合わない可能性が高い
- 深層学習環境は複数企業の製品で構成されており、製品同士のバージョン依存性が高い

  -CUDAドライバー、cuDNN： NVidia

  -TensorFlow、Keras: Google

  -pyTorch: Facebook

  -Anaconda: Anaconda

- 企業戦略的に古いバージョンのドライバーのダウンロードページは見つけにくいことが多い

熟練者でも環境づくりは試行錯誤で見つけてマニュアル化し、
必要なファイル群は手元に保管しておく

非熟練者でも5分で立ち上がる
ディープラーニング環境

Google Colaboratory

- サービス概要抜粋

  -機械学習の教育や研究に利用できる研究ツールです。特別な設定なしで、Jupyter Notebook 環境をご利用いただけます。

  -ほとんどの主要なブラウザで動作し、PC バージョンの Chrome と Firefox では完全に動作するよう検証済みです。

  -Colaboratory は無料でご利用いただける研究プロジェクトです。

とにかく、必要なものはPCブラウザとGoogleアカウントのみ。
しかも、無料な上に、GPUリソース(10GB以上)も利用可能

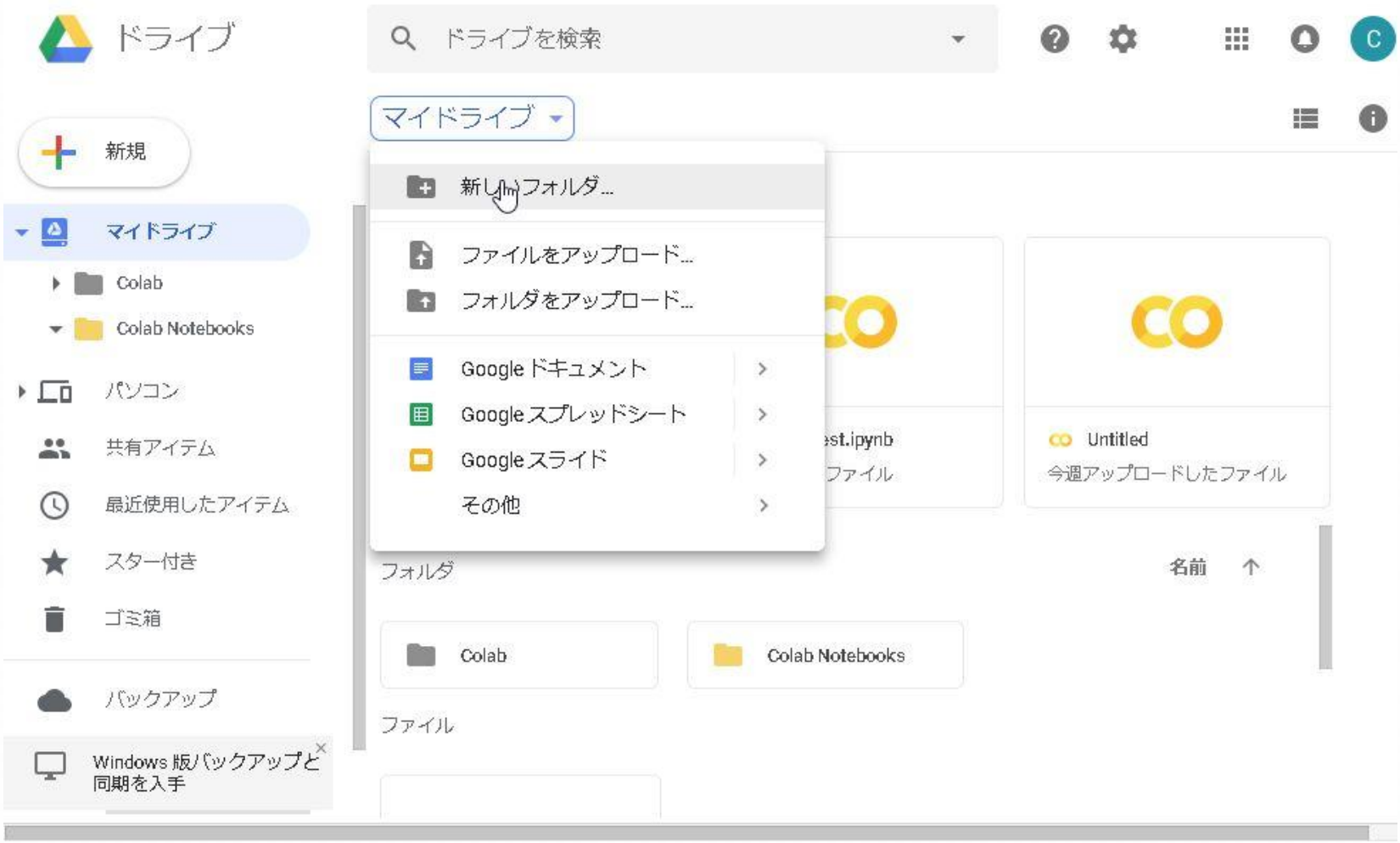
簡単な利用手順

1. Googleアカウント作成(なければ)
2. Google DriveからColaboratoryの利用設定
3. Colaboratory用ファイル作成
4. GPUの利用設定

わずか数ステップで環境が出来上がり。
しかも、OpenCV(画像処理)やNumpy(数値演算)など
深層学習に必要なPythonモジュールは最初から組み込まれている

設定方法

Google Driveにログイン後に、Colaboratory用フォルダを作成



設定方法

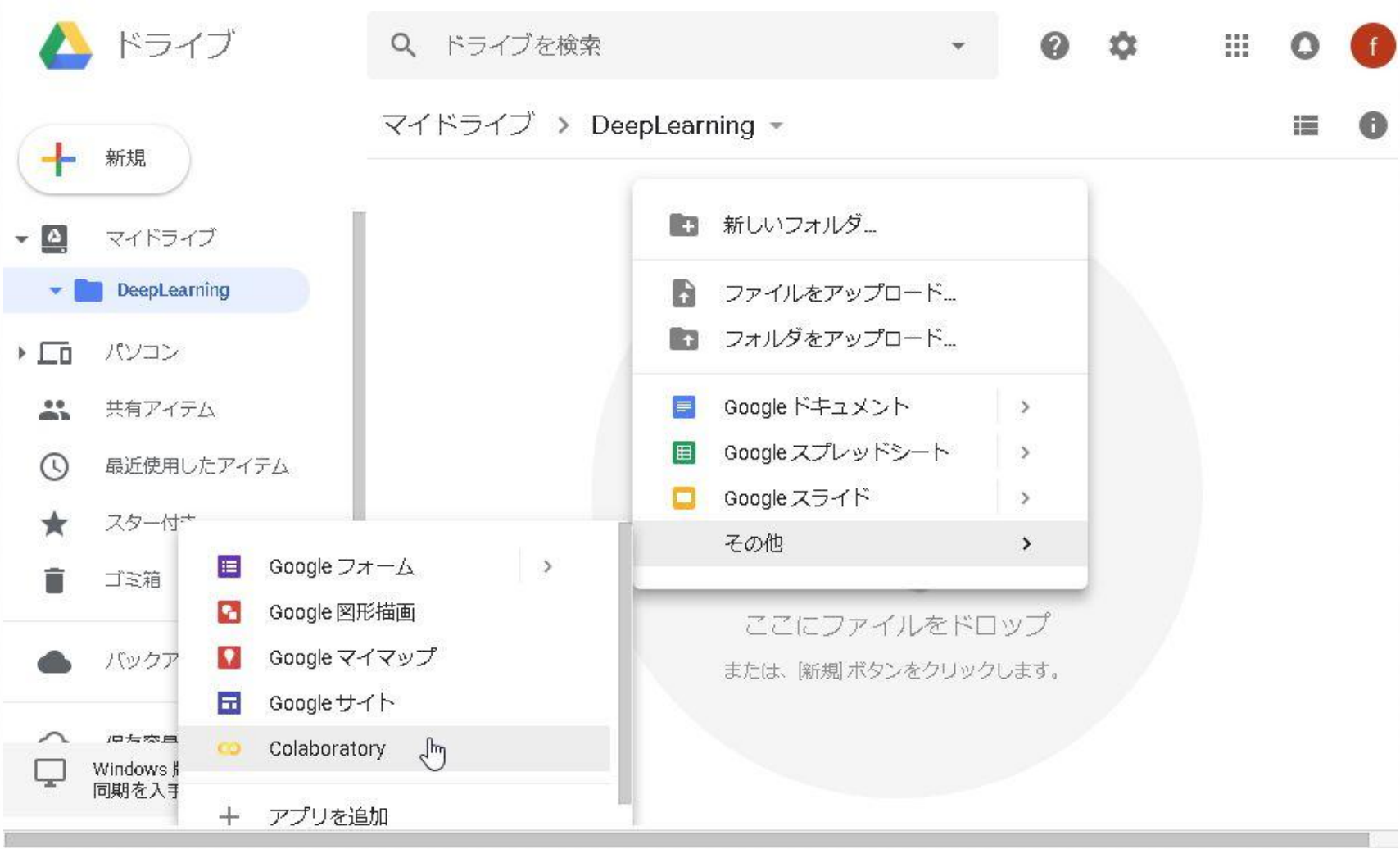
作成したフォルダを選択して、メニューから「アプリを追加」を選択

設定方法

検索窓に「Colaboratory」と入力し、表示された「接続」ボタンをクリックすれば、ColaboratoryとGoogle Driveの連携は完了

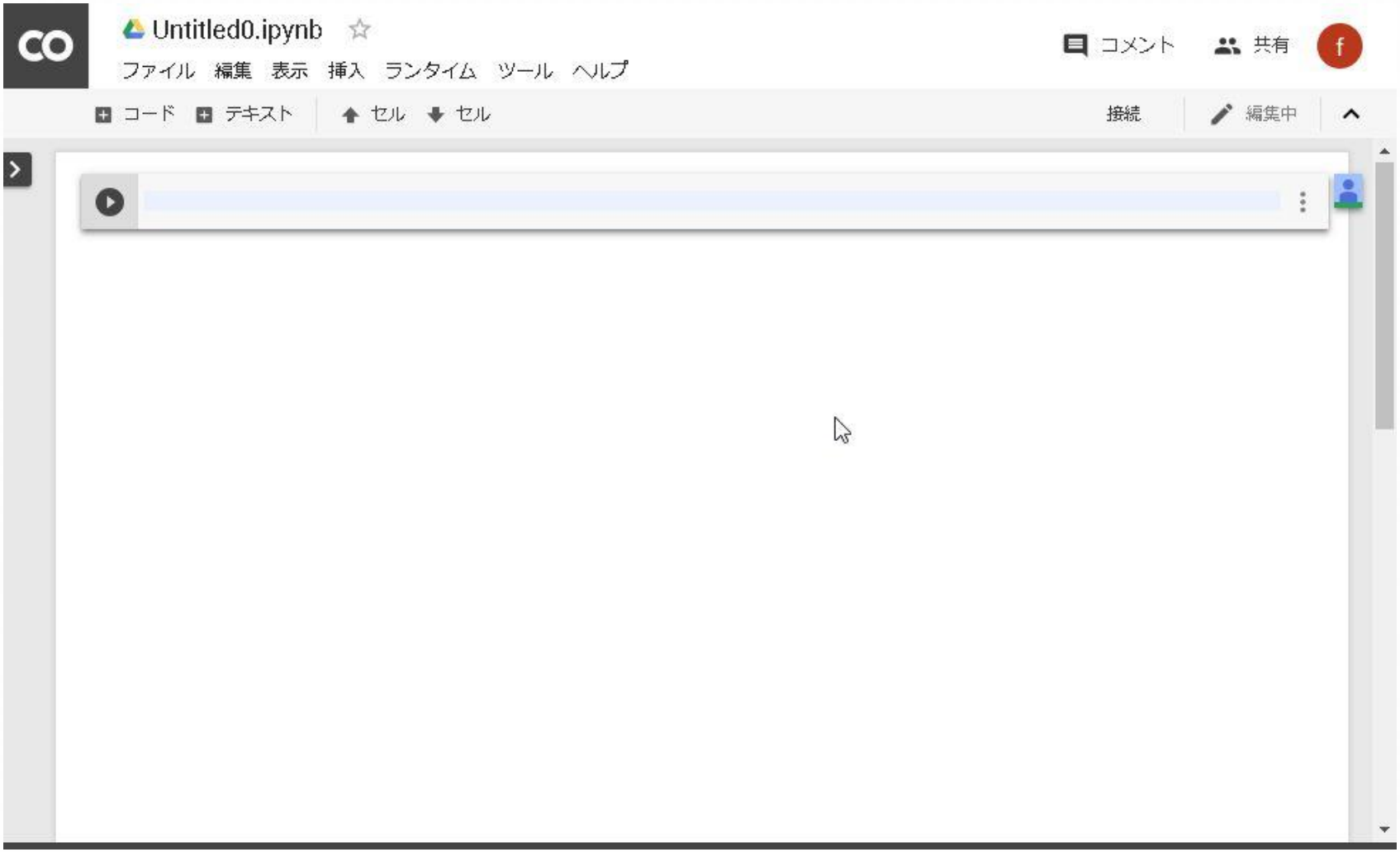
設定方法

フォルダ内の画面で右クリックから「その他」-「Colaboratory」を選択する



設定方法

Colaboratory用ファイルが新規に追加され、コード作成用画面が自動表示される

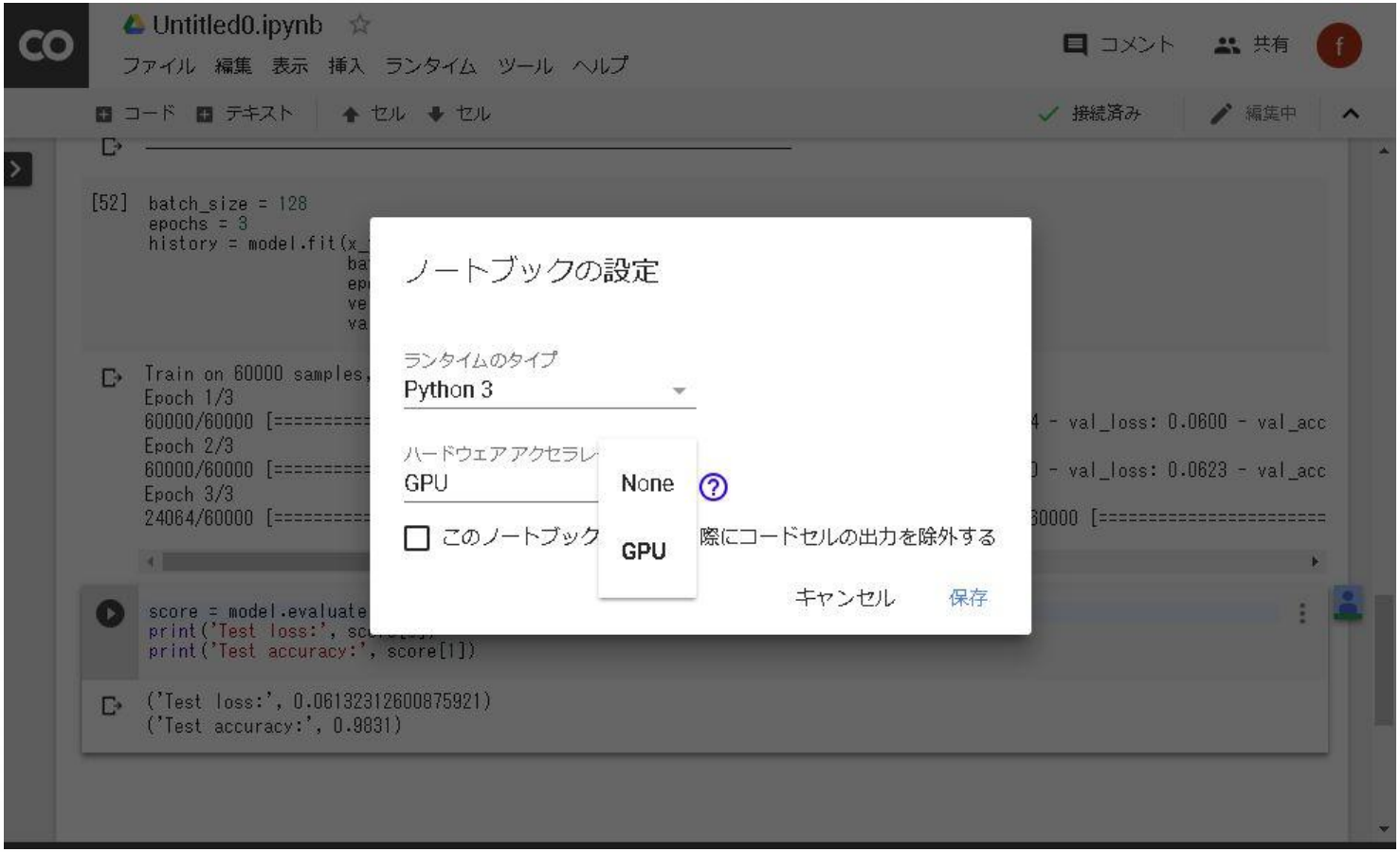


GPU有効設定

- CorabolatoryではデフォルトでGPU利用が有効になっていないので、設定でGPUをONに切り替え必要がある
- これを忘れても学習自体は動くが、時間が掛かる
- メニューの[編集]-[ノートブックの設定]を選択

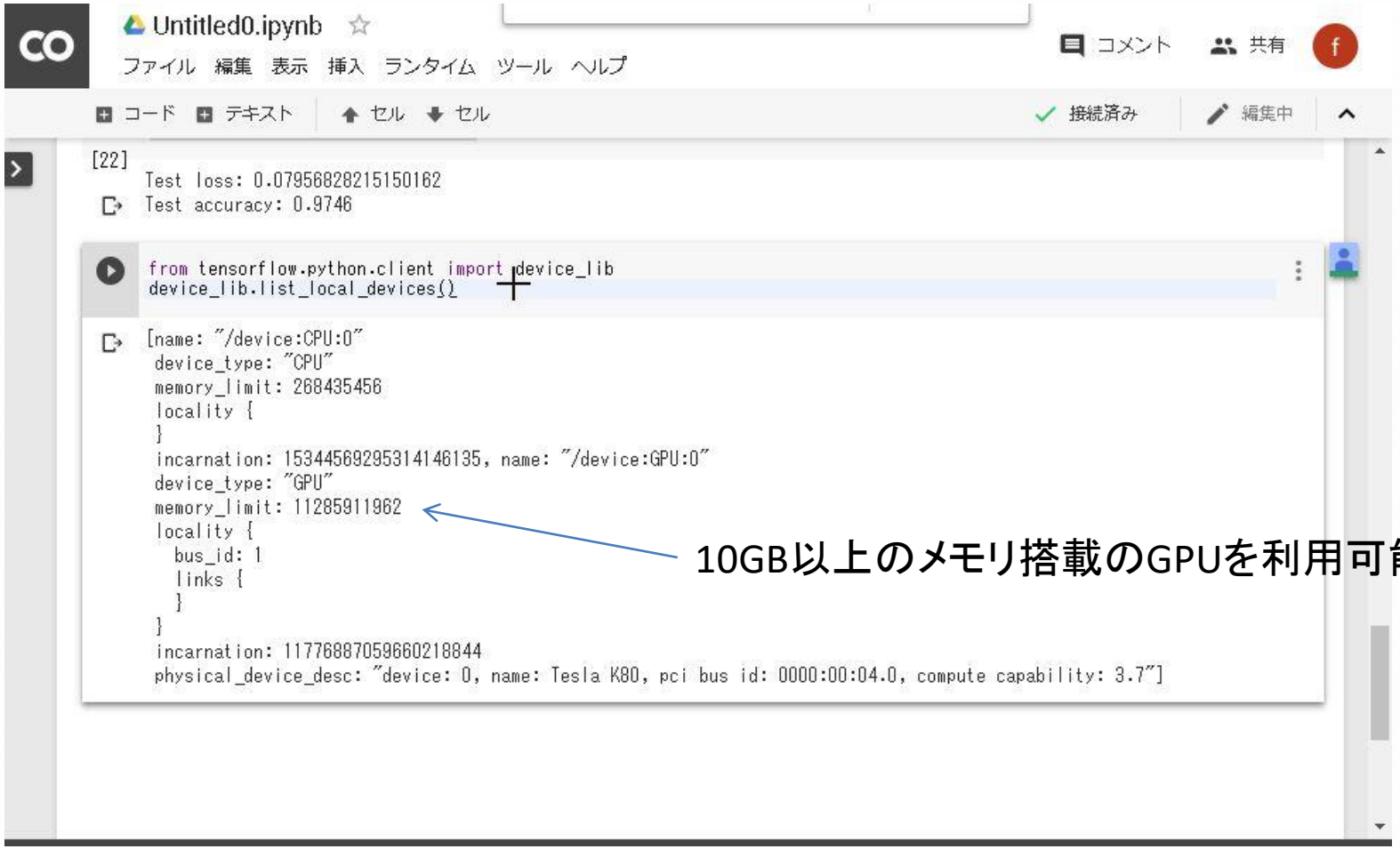
GPU有効設定

- ランタイムのタイプをPython3
- ハードウェアアクセラレータは[None]から[GPU]へ変更



GPU有効設定（これで準備完了）

- 設定変更後はすぐに有効になっていない可能性があるので、
- 一旦ノートを閉じて再度開く(あるいは[ランタイム]-[すべてのランタイムをリセット])
- GPUのスペックを確認



オンライン実習

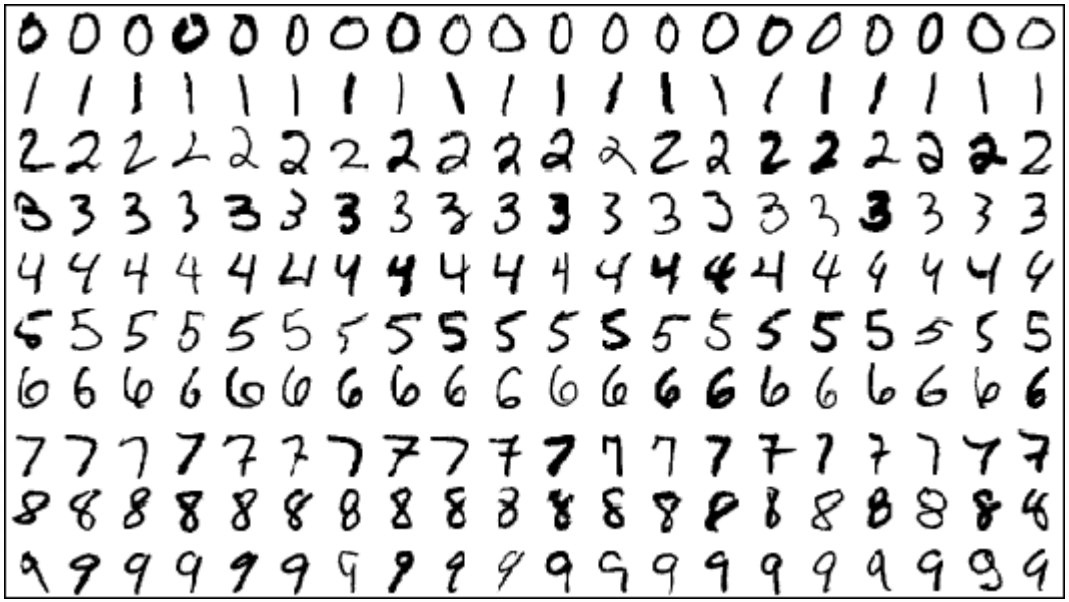
「グーグルコラボ」で試すDEEP LEARNING

コードデモ

- MNISTのデータセットで学習を行い、簡単な数値判定性能を見てみよう

  -MNISTとは
  手書きの数字「0～9」に正解ラベルが与えられているデータセット

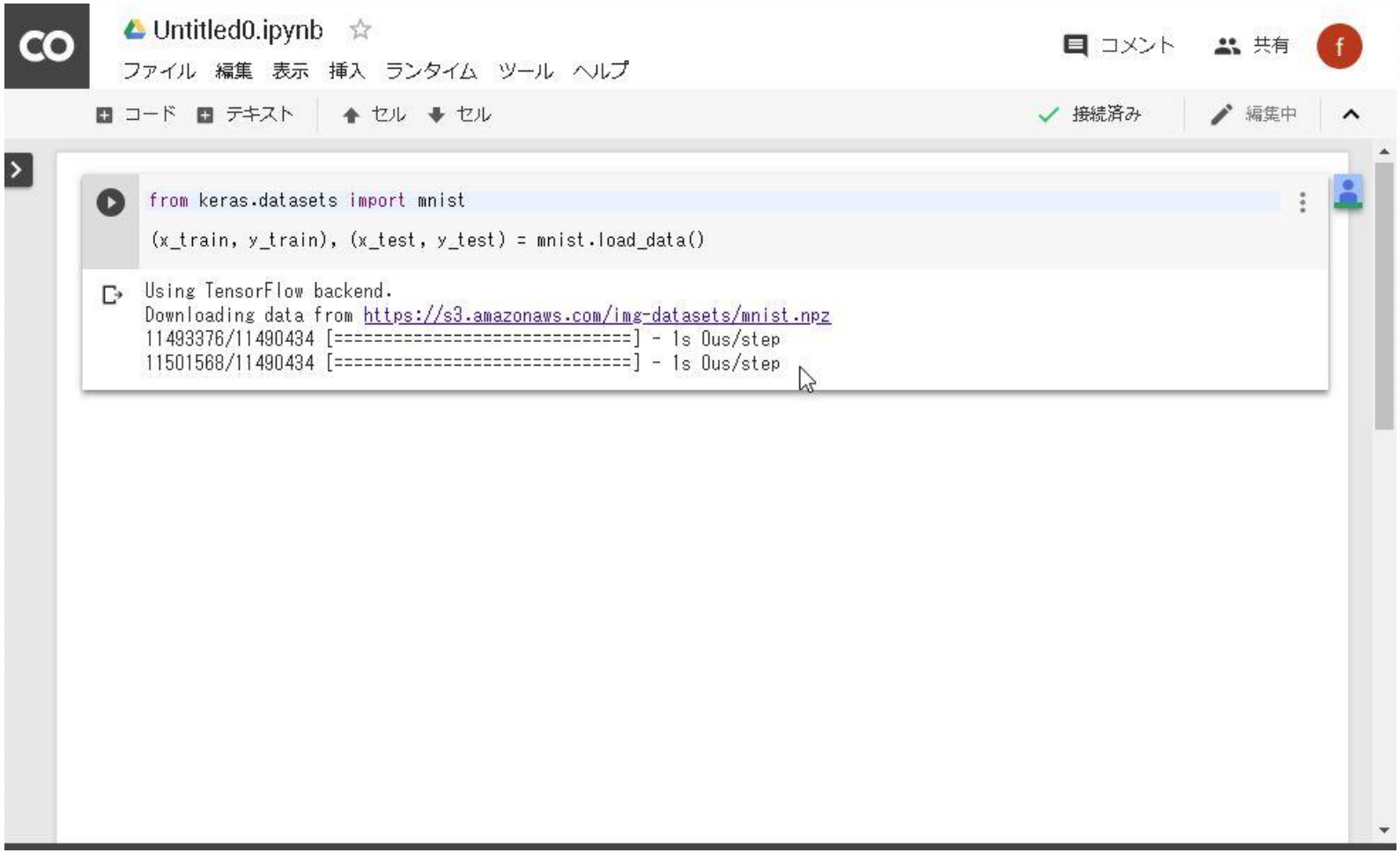
  -各画像はグレースケール画像で28x28ピクセル



コードデモ

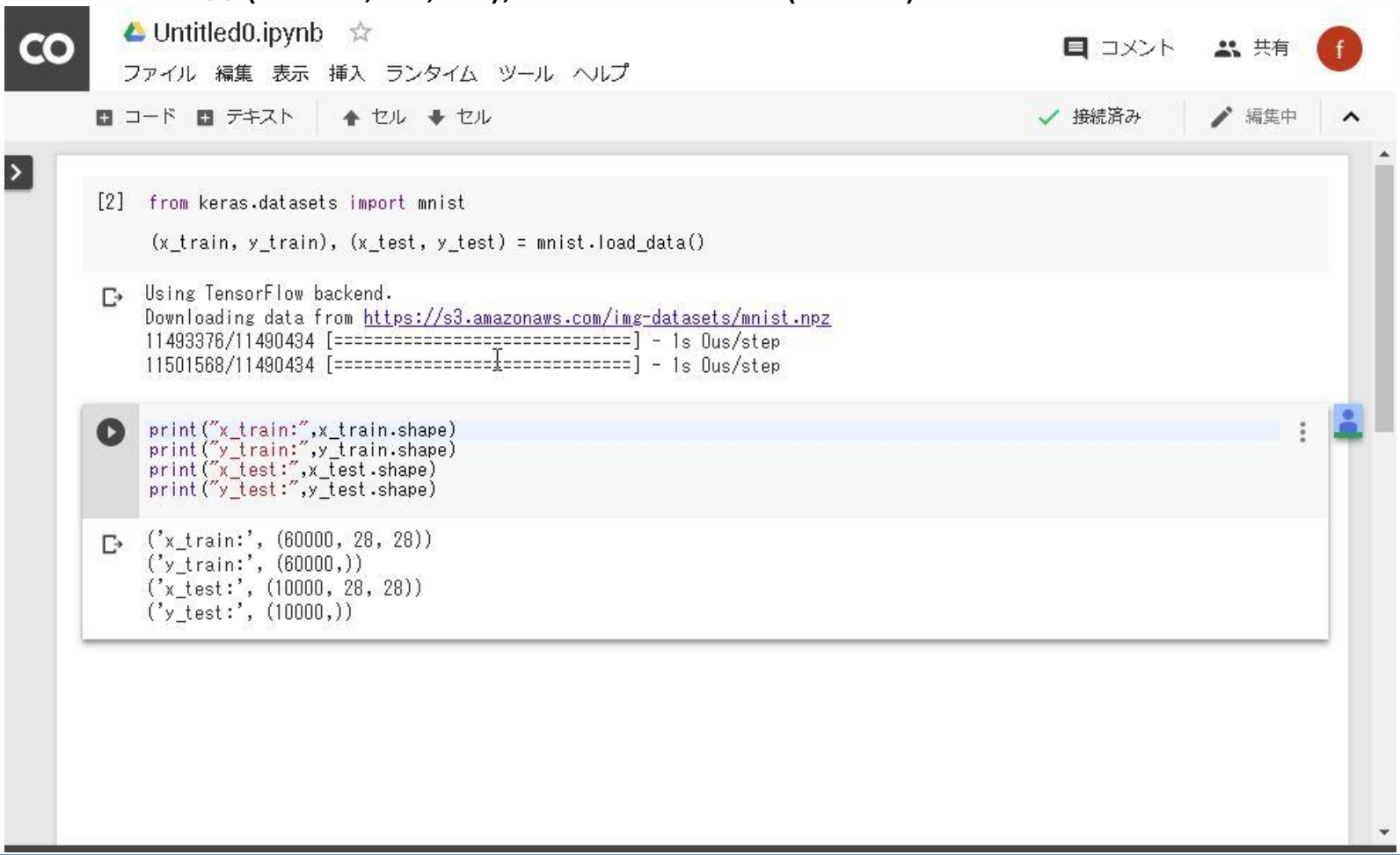
- データ読み込み

  -MNISTデータ用クラスがKerasで作成されていて、読み込みの関数を呼ぶと自動的にWeb上からMNISTのデータをダウンロードして展開する



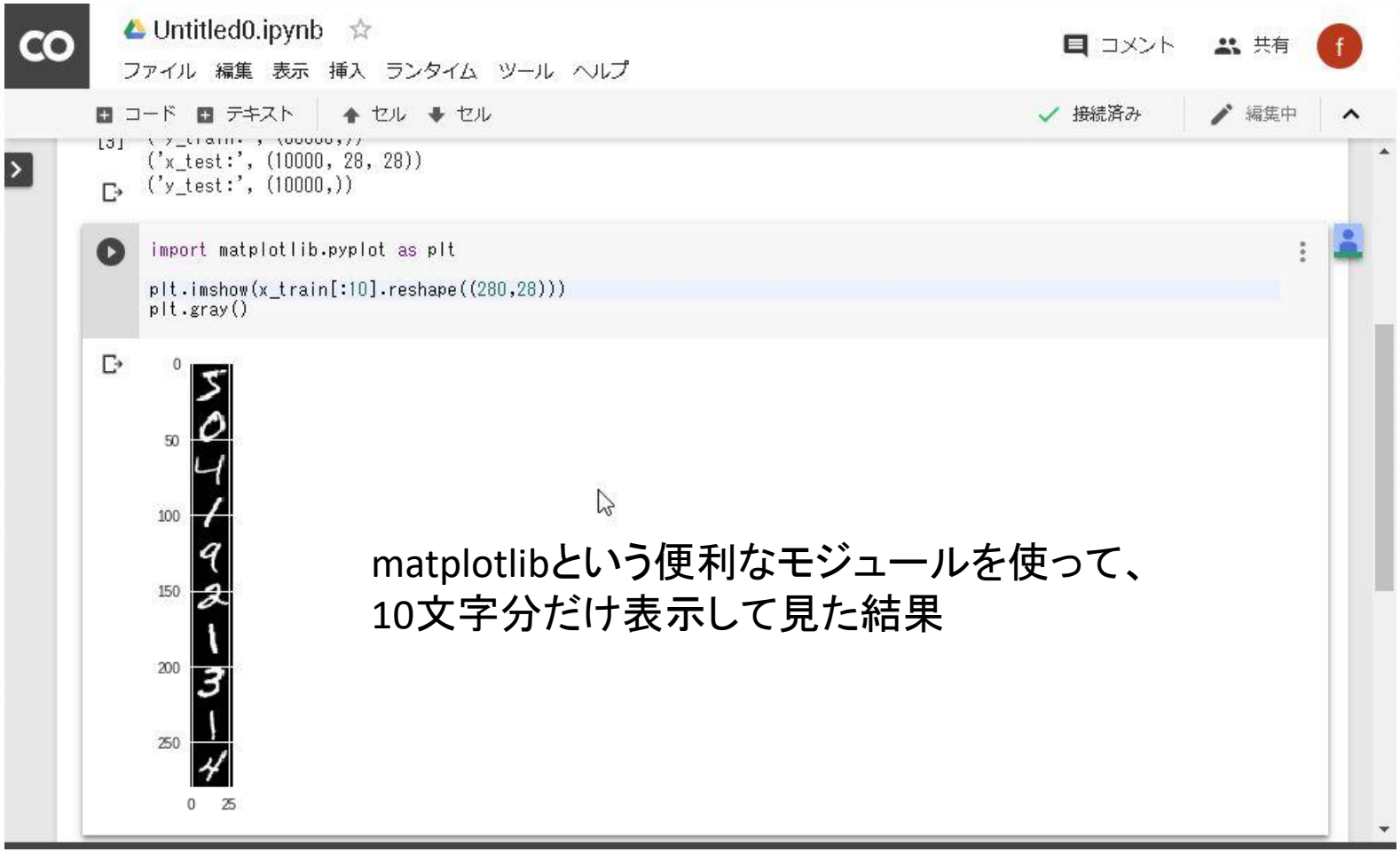
コードデモ

- データ確認
  学習用画像(60000, 28, 28), 学習用ラベル(60000),
  テスト用画像(10000, 28, 28), テスト用ラベル(10000) の4配列



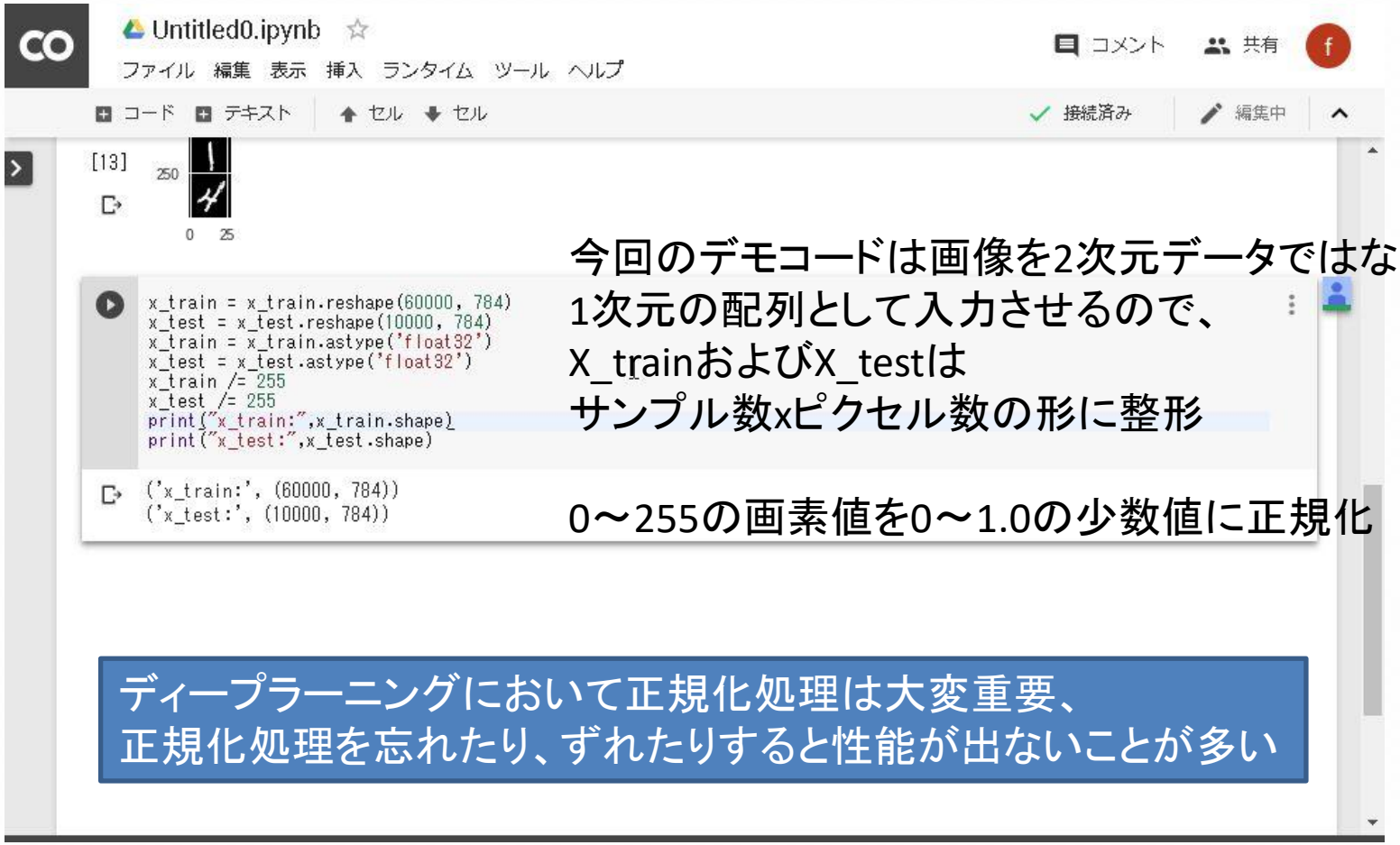
コードデモ

- データ確認
  文字を表示してみる。



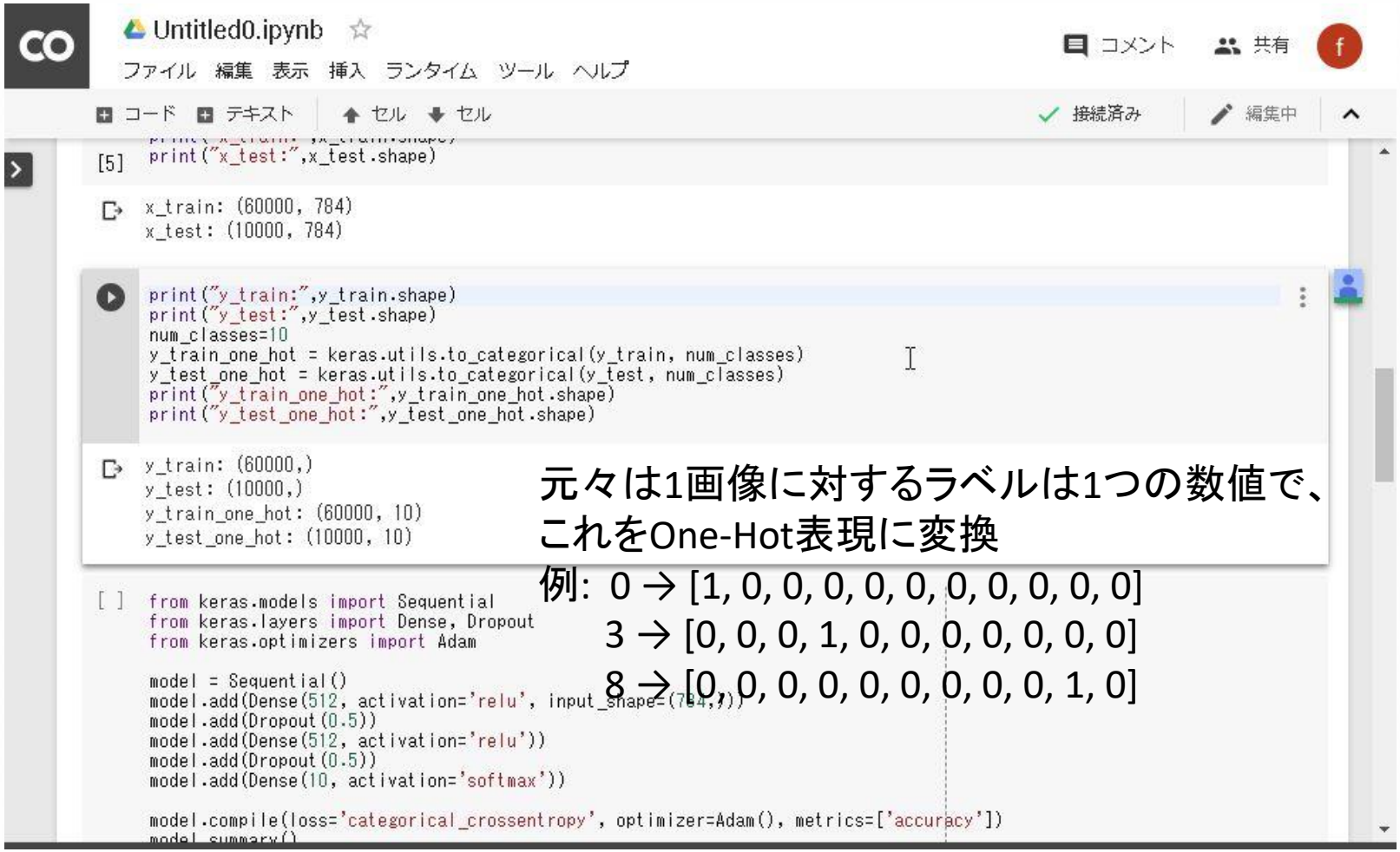
コードデモ

- 学習用にデータを整形＆正規化
  ネットワークの入力の形に合わせる必要がある



コードデモ

- 学習用にデータを整形＆正規化
  出力に応じて正解ラベルを変換

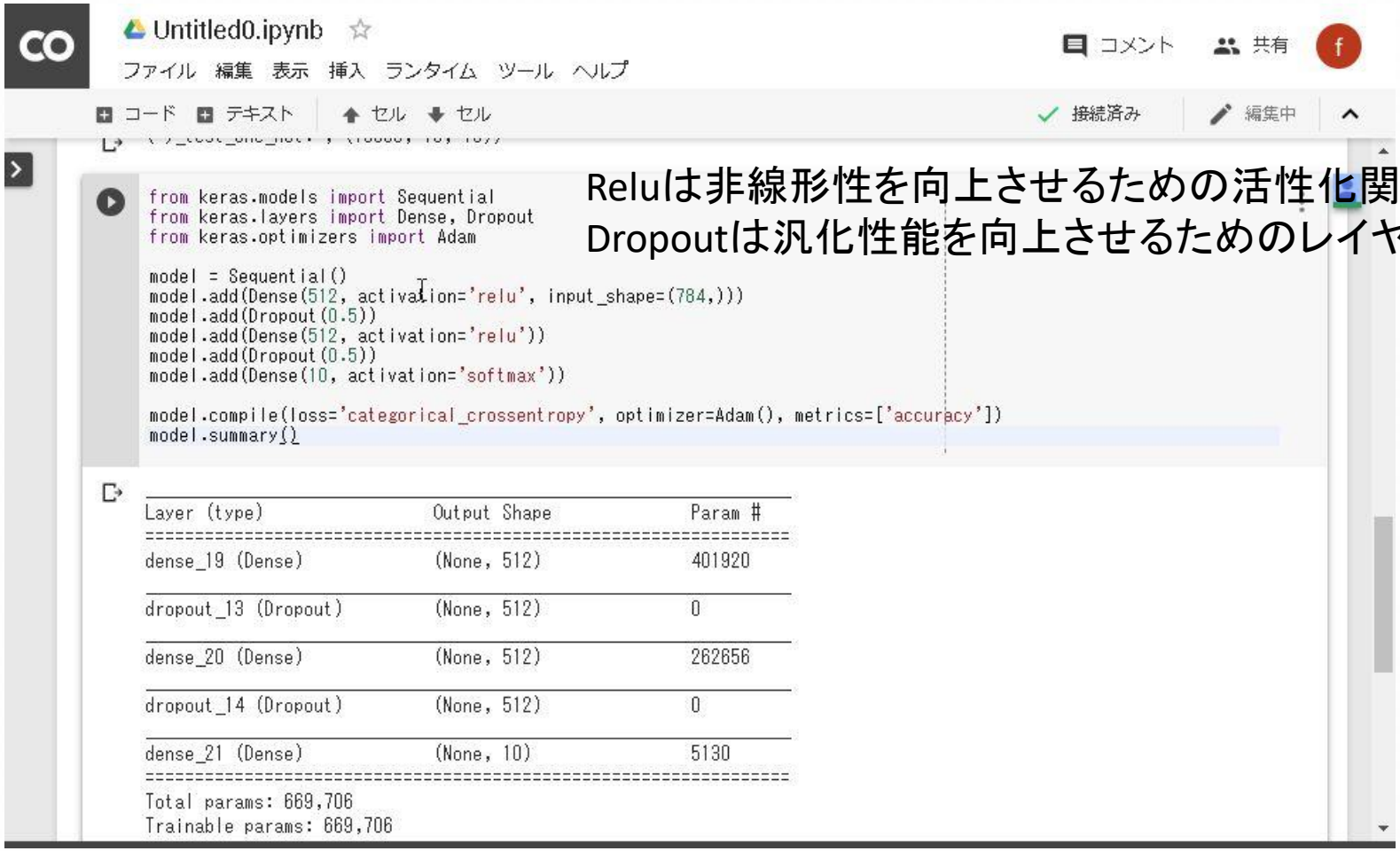


元々は1画像に対するラベルは1つの数値で、
これをOne-Hot表現に変換

例: 0 → [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
3 → [0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
8 → [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]

コードデモ

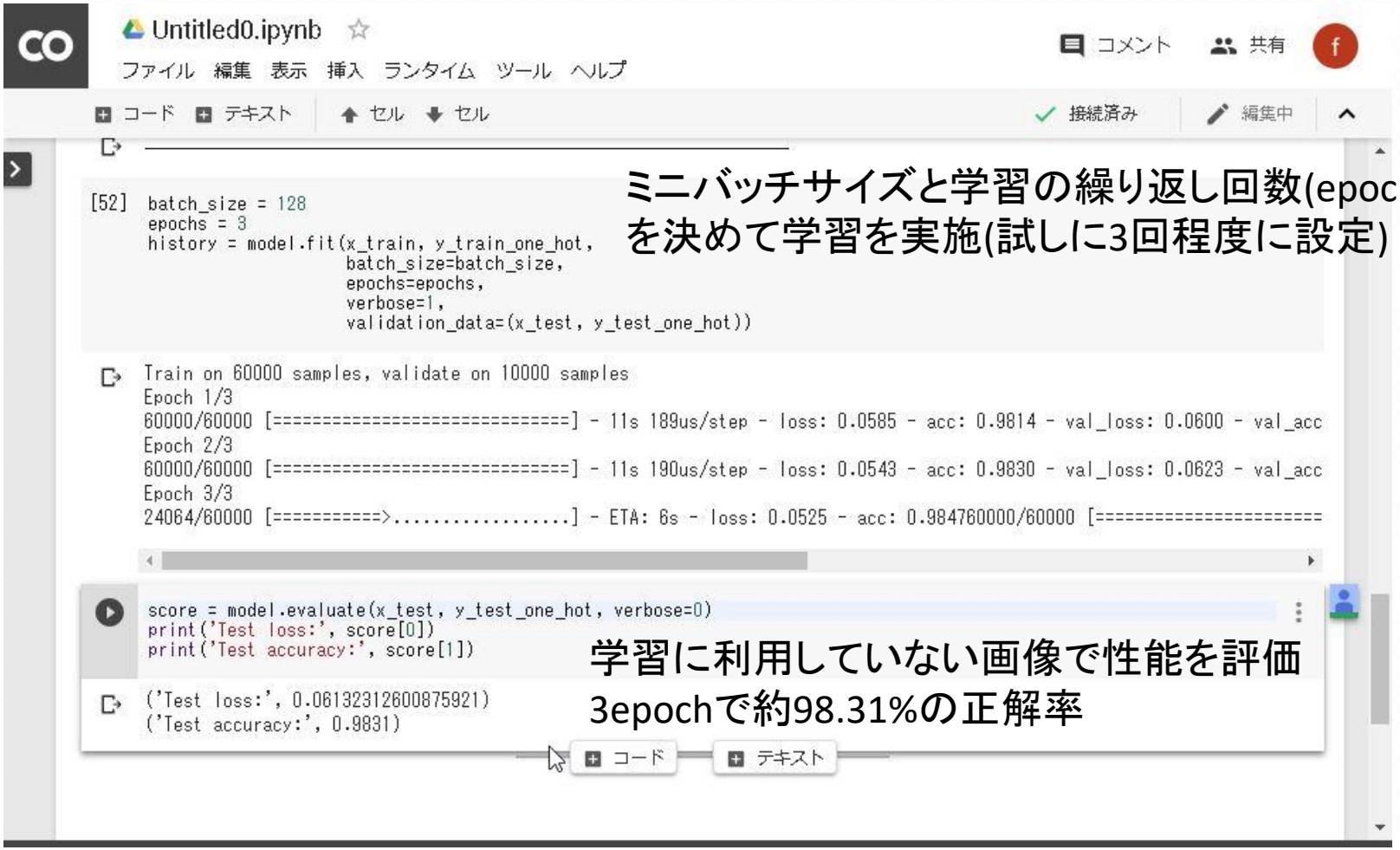
- 学習用モデルを生成
  今回は2層程度のモデルを作成



Reluは非線形性を向上させるための活性化関数で、
Dropoutは汎化性能を向上させるためのレイヤー

コードデモ

- 学習の実施と検証



実習資料ダウンロード先

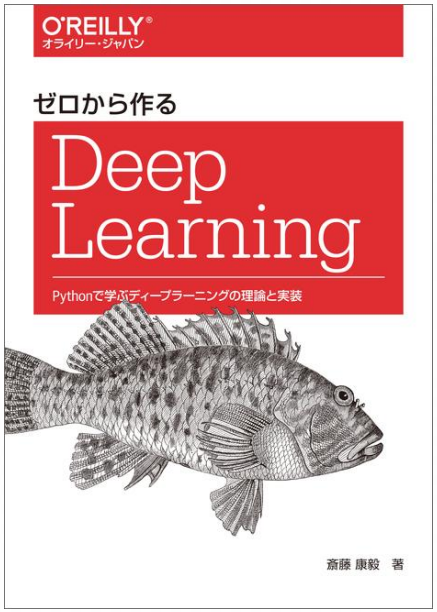
- 下記URLより実習用ソースコードやデータを入手可能です

  https://www.egrowth.co.jp/dl/seminor201902/
- Googleで「イーグロース　京都」から当社HPにアクセスいただき、「ニュース」メニューからでも上記URLへのリンク記事を用意しています。

その他参考資料

深層学習向け書籍

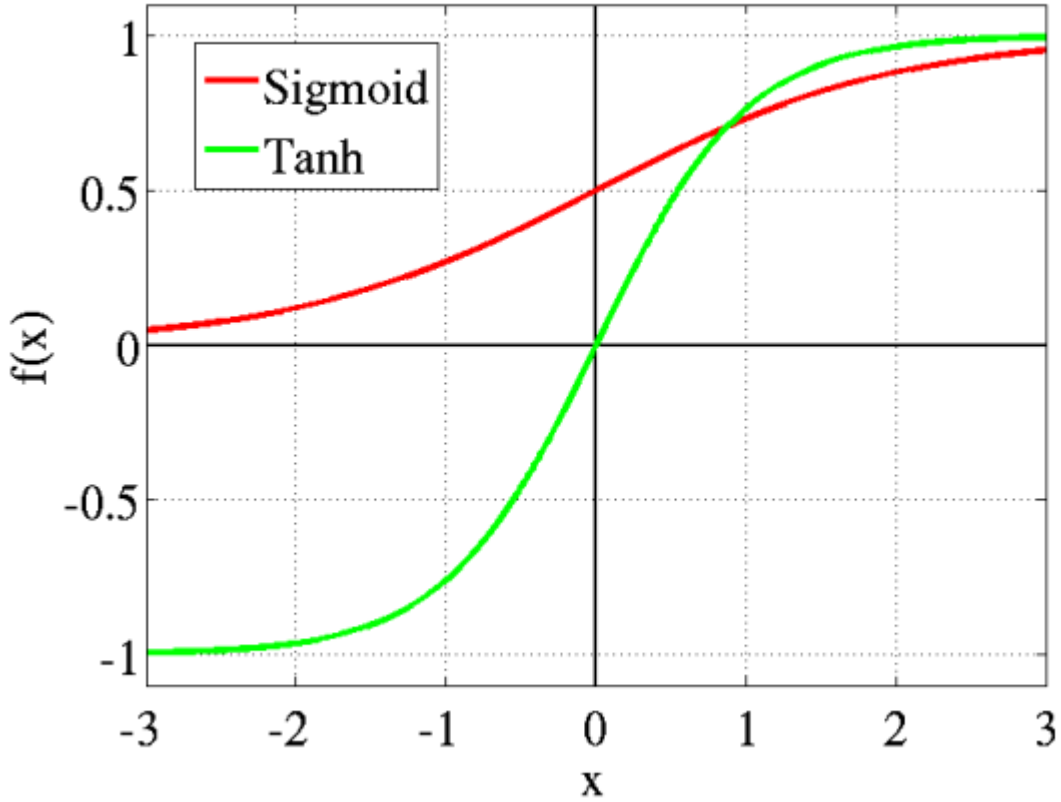
- ゼロから作る「Deep Learning」
  -フレームワークではなく、自らPythonで各層の構築を試みることが可能
  -GPU環境は利用しない
  -CNNの仕組み、重みの更新アルゴリズムについて理解できるので、ディープラーニングの理解を深めるのにおすすめ



斎藤 康毅　著
2016年09月 発行
オライリー・ジャパン

付録

- 活性化関数と正規化



[Towards Data Science]

Sigmoidではモデルの出力を0.0～1.0、tanhでは出力を-1.0～1.0の間に押し込むので、
活性化関数に合わせて教師ラベルを正規化する必要がある

付録

- ダイス(Dice)係数
  -2つの集合の類似度を表す係数の1つ
  -これ以外にJaccard係数、simpson係数などがある