ae-9.ディープラーニングによる画像理解

（AI 演習）（全１５回）

https://www.kkaneko.jp/ai/ae/index.html

金子邦彦

CC BY NC SA

人工知能（AI）の学習のための指針

1. **実践重視**： AIツールを実際に使用し、機能に慣れる

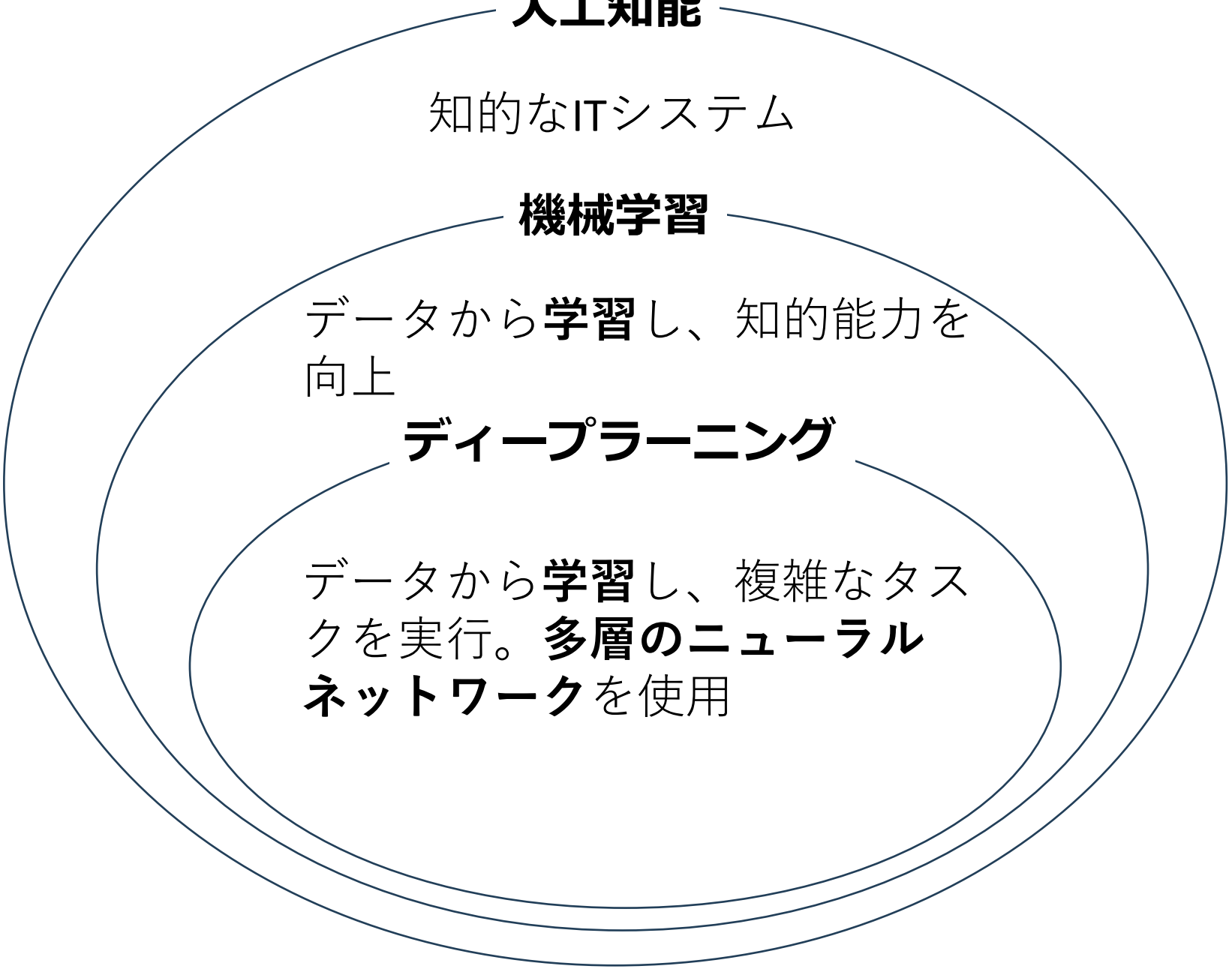
2. **エラーを恐れない**： 実行においては、エラーの発生の可能性がある．エラーを恐れず，むしろ学習の一部として捉えるポジティブさが大切．

3. **段階的学習**：基礎から応用へと段階的に学習を進め，AIの可能性を前向きに捉える

アウトライン

1. イントロダクション
2. 畳み込みニューラルネットワークによる画像分類の例
3. 画像分類と学習済みモデルの活用
4. 様々な画像理解のタスク
5. 物体検出
6. セグメンテーション
7. セグメンテーションのバリエーション

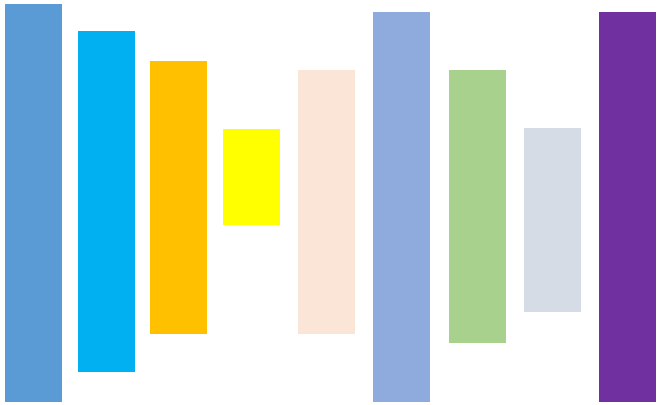
9-1. イントロダクション


人工知能
知的なITシステム
機械学習
データから**学習**し、知的能力を向上
ディープラーニング
データから**学習**し、複雑なタスクを実行。**多層のニューラルネットワーク**を使用

ディープニューラルネットワーク

ディープラーニングは**多層のニューラルネットワーク**を用いた機械学習

層の数が少ない（浅い）



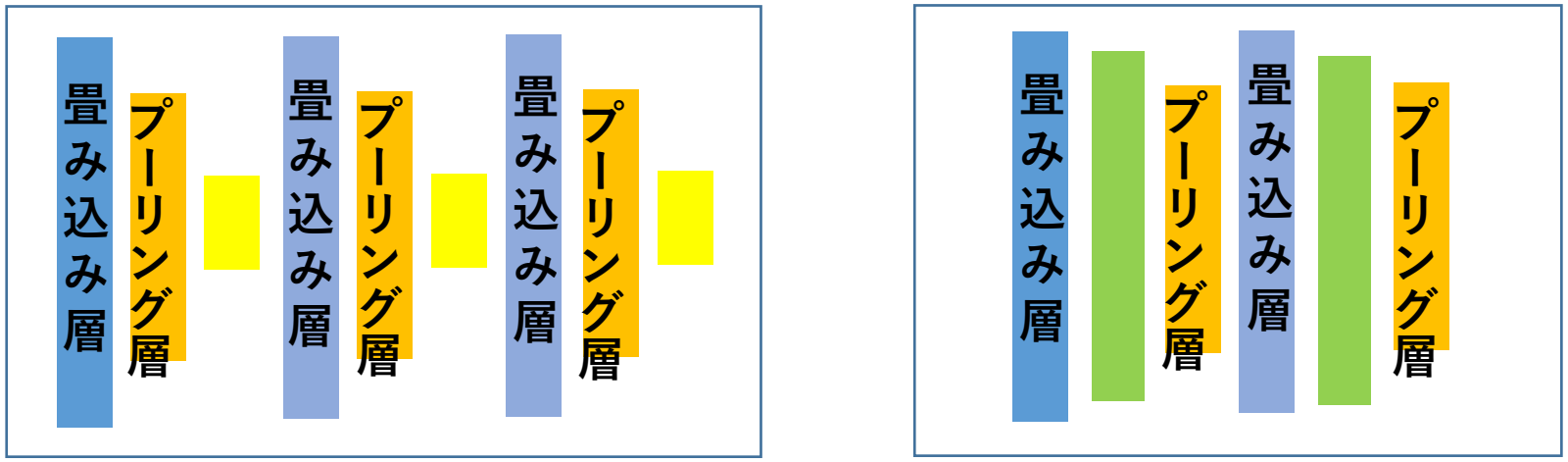
層の数が多い（深い）

畳み込みニューラルネットワーク（CNN）の基本構造

- **画像理解**や**画像の分析**に特化した**ディープラーニング**の一種.
- 主に**畳み込み層**、**プーリング層**、**全結合層**の３種類で構成
  - **畳み込み層**：画像の局所的な**特徴**（エッジ、テクスチャなど）をとらえる
  - **プーリング層**：**特徴マップ**の**サイズを縮小**し，**過学習を防止.**
  - **全結合層**：**特徴マップ**をもとに**画像**を処理する

畳み込みニューラルネットワーク（CNN）の例

- CNNの主要な層構成：
  - **プーリング層**
  - **畳み込み層**
  - **全結合層**
- これらの層を組み合わせることで，さまざまなバリエーションのCNNを構築できる.



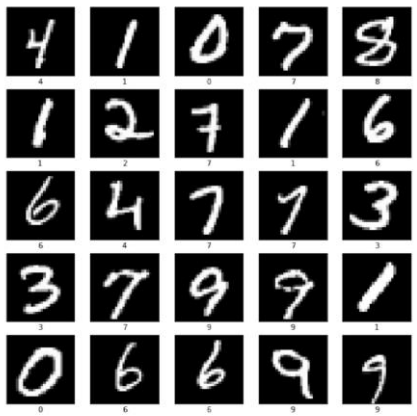
さまざまなバリエーション

9-2. 畳み込みニューラルネットワークによる画像分類の例

手書き数字の MNIST データセットの分類

Pythonによる手書き数字認識の実装

学習に使用する訓練データ
（抜粋）



4 1 0 7 8
1 2 7 1 6
6 4 7 7 3
3 7 9 9 1
0 6 6 9 9

画像 60000枚
（うち一部）

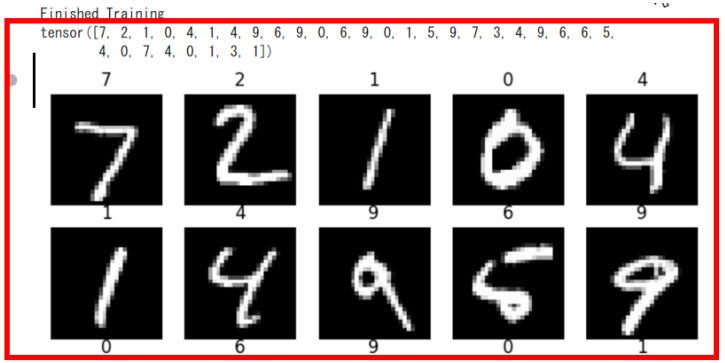
正解 60000個
（うち一部）

プログラム

データを用いて学習を行う
学習ののち，画像分類を行う

```
[4] !pip install -U scikit-learn matplotlib

    import torch
    import torch.nn as nn
    import torch.optim as optim
    from sklearn import datasets
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    import matplotlib.pyplot as plt

    # データの取得と前処理
    iris = datasets.load_iris()
    X = iris.data
    y = iris.target

    # データの標準化
    scaler = StandardScaler()
    X = scaler.fit_transform(X)

    # 訓練データとテストデータの分割
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    X_train = torch.tensor(X_train, dtype=torch.float32)
    y_train = torch.tensor(y_train, dtype=torch.long)
    X_test = torch.tensor(X_test, dtype=torch.float32)
    y_test = torch.tensor(y_test, dtype=torch.long)

    # ニューラルネットワークの定義
    class Net(nn.Module):
        def __init__(self):
            super(Net, self).__init__()
            self.fc1 = nn.Linear(4, 10)  # 入力は4次元（Irisの特徴量）
            self.fc2 = nn.Linear(10, 3)  # 出力は3クラス

        def forward(self, x):
            x = torch.relu(self.fc1(x))
            x = self.fc2(x)
            return x

    net = Net()
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(net.parameters(), lr=0.01)
```

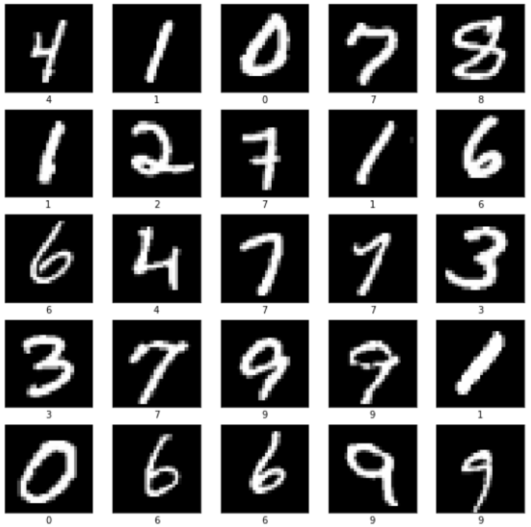
学習の結果、文字認識の能力を獲得



手書き数字の MNIST データセットの分類

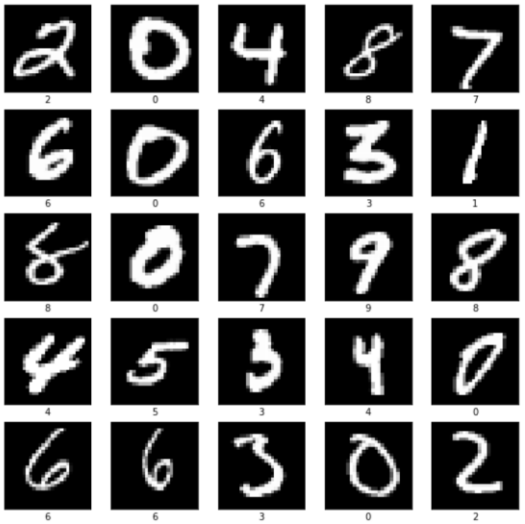
手書き文字の画像を 0 ～ 9 に分類するタスク

- **訓練データ：60,000枚の画像と正解**
- **検証データ：10,000枚の画像と正解**
- **訓練データ**を使用して、**モデルの学習**を行う
- **検証データ**を使用して、**学習されたモデルの検証**を行う



60000枚の画像と正解

訓練データ



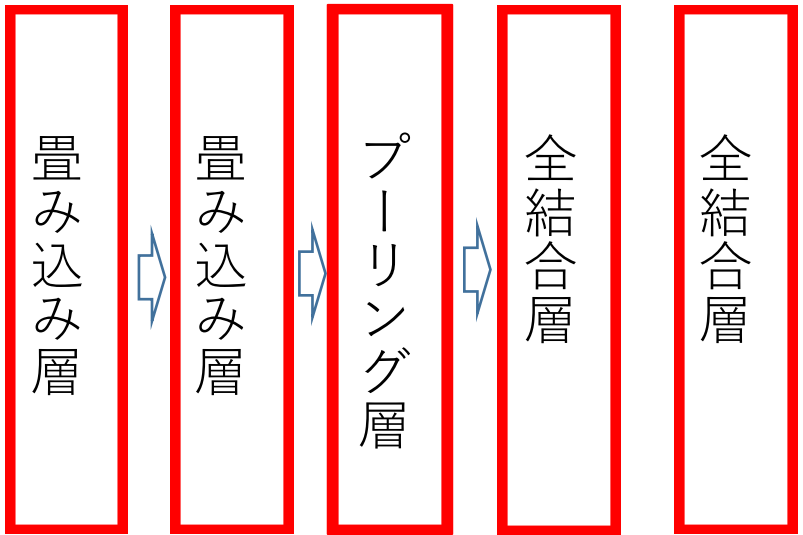
10000枚の画像と正解

検証データ

Pythonによる手書き数字認識のプログラム

```
def __init__(self):
    super(Net, self).__init__()
    self.conv1 = nn.Conv2d(1, 32, 3)  # 1層目の畳み込み層
    self.conv2 = nn.Conv2d(32, 64, 3) # 2層目の畳み込み層
    self.pool = nn.MaxPool2d(2, 2)    # プーリング層
    self.fc1 = nn.Linear(64 * 12 * 12, 128) # 全結合層1
    self.fc2 = nn.Linear(128, 10)     # 全結合層2（出力層）
```

下の畳み込みニューラルネットワークを組み立てるプログラム

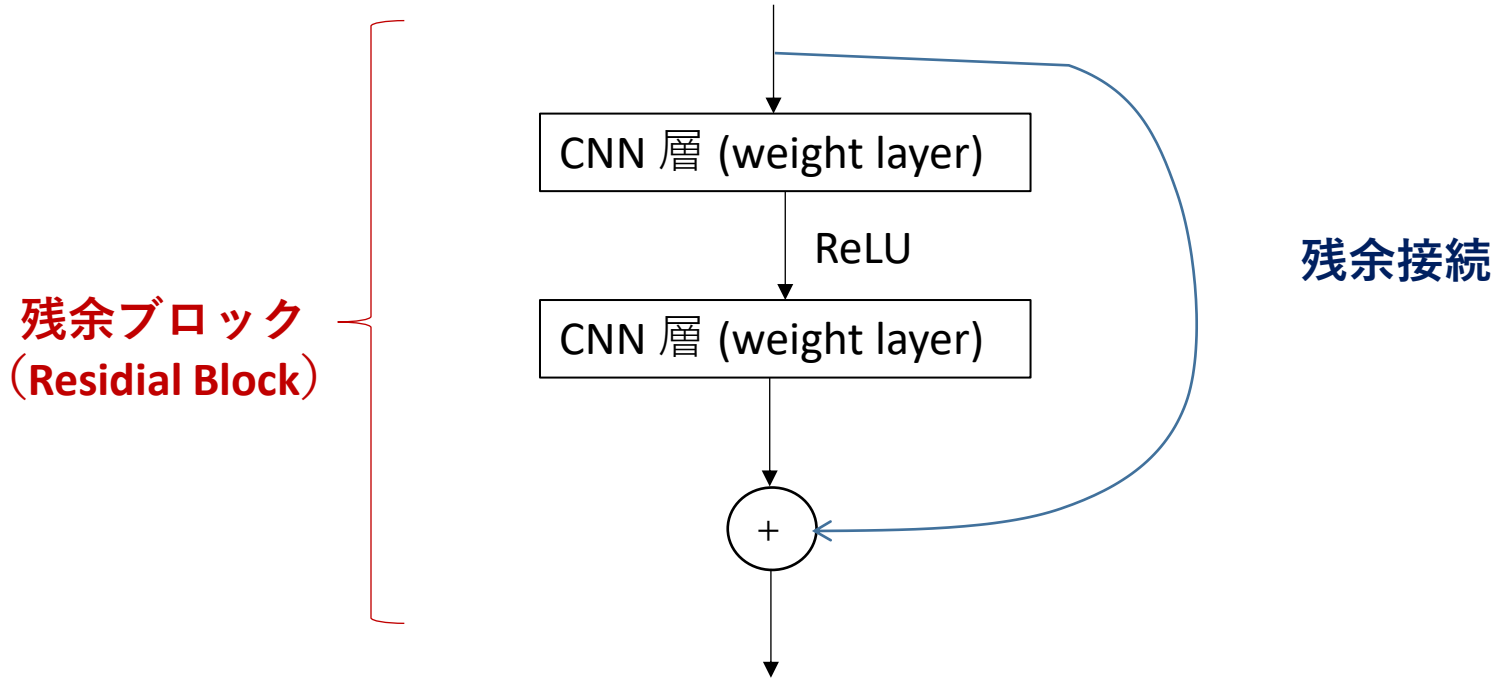


演習１を参照

ResNet (2015 年) の基本アイデア

残余接続の導入.

- 層をスキップして，後続の層へ直接データを送る経路を追加.
- **残余ブロック（Residual Block）** を構成
- 勾配消失問題（過学習の原因となる）の緩和を実現



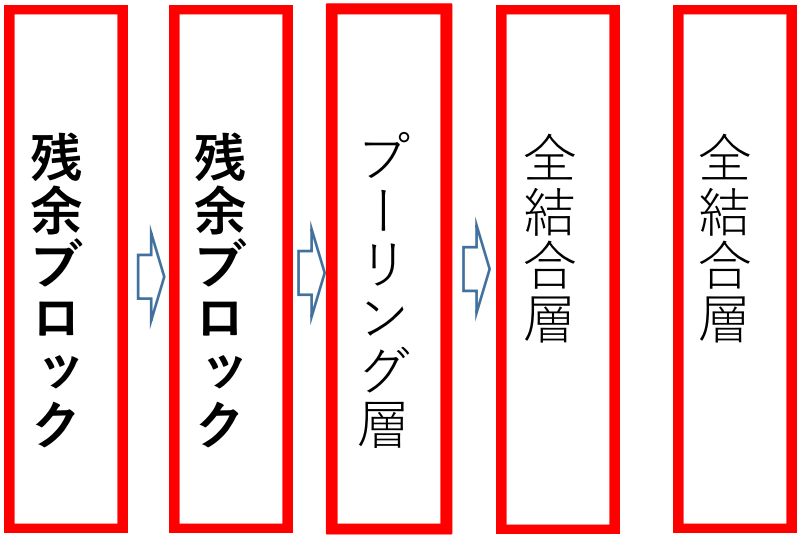
Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun,

Deep Residual Learning for Image Recognition, IEEE Conference on Computer Vision and Pattern Recognition, 2016.

Pythonによる手書き数字認識のプログラム（ResNet を導入した改善版）

```
def __init__(self):
    super(Net, self).__init__()
    self.conv1 = ResidualBlock(1, 32)  # 32チャンネルへ
    self.conv2 = ResidualBlock(32, 64)  # 64チャンネルへ
    self.pool = nn.MaxPool2d(2, 2)    # プーリング層
    self.fc1 = nn.Linear(64 * 12 * 12, 128) # 全結合層1
    self.fc2 = nn.Linear(128, 10)     # 全結合層2（出力層）
```

下の畳み込みニューラルネットワークを組み立てるプログラム



演習２を参照

演習１．手書き数字の認識. CNN の効果

① Google Colaboratory のページを開く

https://colab.research.google.com/drive/18IPPkY96Oc6jkYD2su4cFgWcoYAskLo_?usp=sharing

② プログラムや説明や実行結果が掲載されていることを確認。

各自でよく読む。

③「**１．手書き数字の認識（教師あり学習）（畳み込みニューラルネットワークを使用）**」を確認

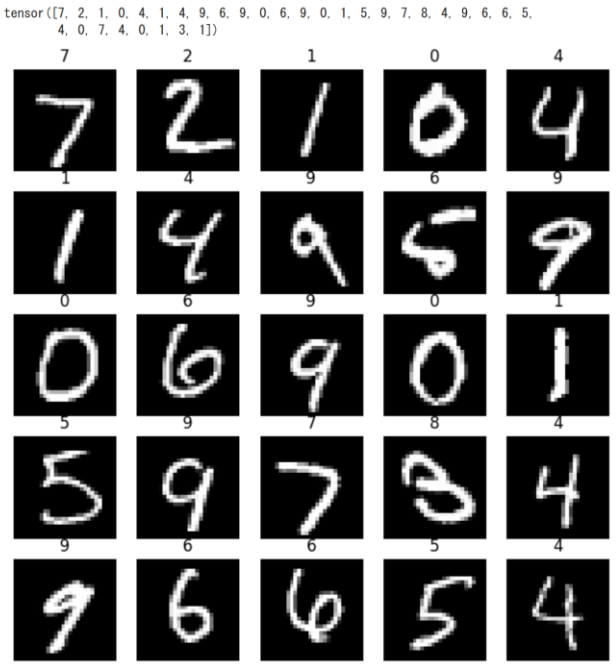
１．手書き数字の認識（教師あり学習）（畳み込みニューラルネットワークを使用）

【概要】

データセット：MNIST（0-9の手書き数字）。
畳み込みニューラルネットワークの使用。
　１層目：畳み込み、カーネルのサイズは３×３
　２層目：畳み込み、カーネルのサイズは３×３
　３層目：プーリング層
　４層目：全結合層、ニューロン数は１２８
　５層目：全結合層、ニューロン数は１０
学習エポック数：10、学習が進むにつれて損失が減少。
学習曲線のプロットにより、過学習がないことを確認

【プログラムの説明】

手書き数字(0, 1, 2, 3, 4, 5, 6, 7, 8, 9の手書き数字)のMNISTデータセットの**画像データ**を使用して**10クラス**分類を行います。



演習２．手書き数字の認識. ResNet による改善

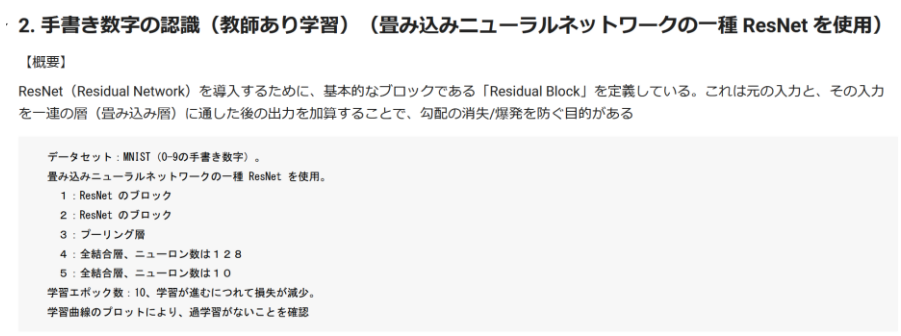
① Google Colaboratory のページを開く

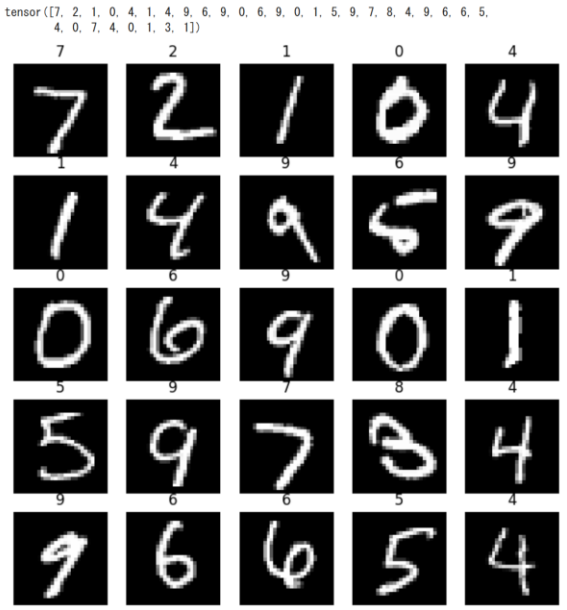
https://colab.research.google.com/drive/18IPPkY96Oc6jkYD2su4cFgWcoYAskLo_?usp=sharing

② プログラムや説明や実行結果が掲載されていることを確認。

各自でよく読む。

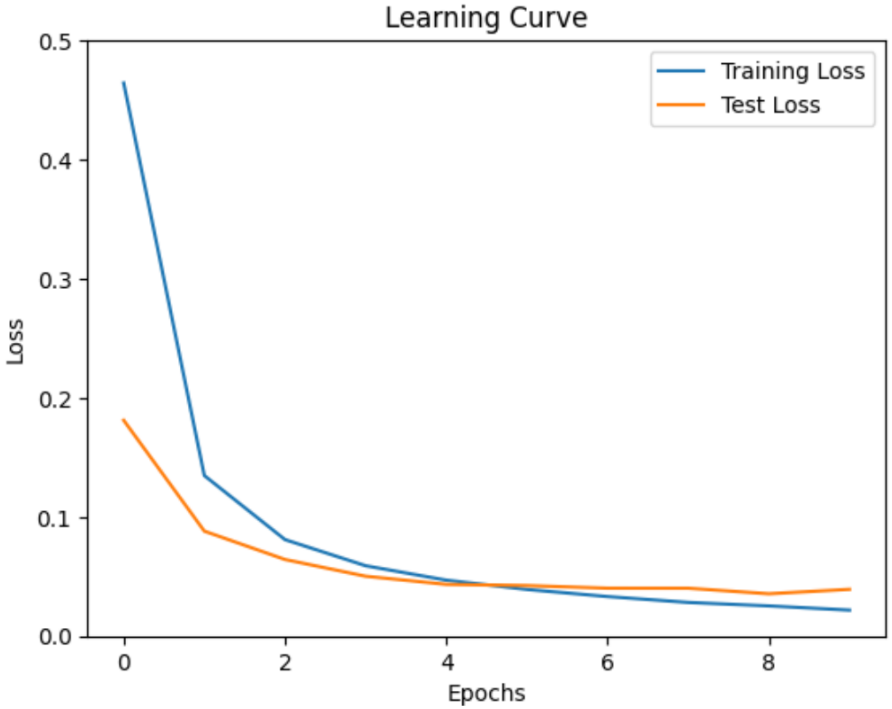
③「**2．手書き数字の認識（教師あり学習）（畳み込みニューラルネットワークの一種 ResNet を使用）**」を確認。





学習の繰り返しと学習曲線

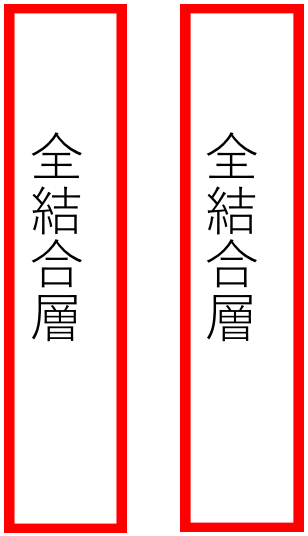
同じ訓練データを用いた学習を１０回繰り返し.
→　学習の繰り返しにより損失（誤差）は減少



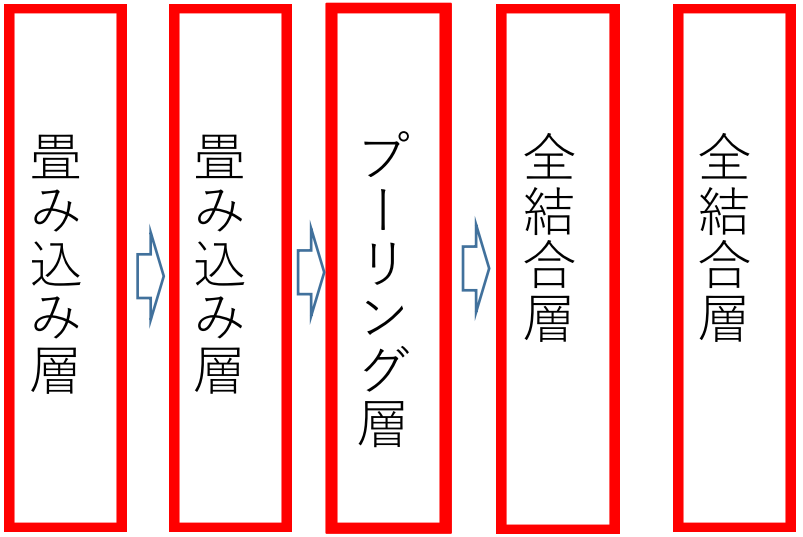
青色：学習の<u>繰り返し</u>ごとの訓練データでの**損失（誤差）**の変化
オレンジ色：学習の<u>繰り返し</u>ごとの検証データでの**損失（誤差）**の変化

畳み込みニューラルネットワークの効果

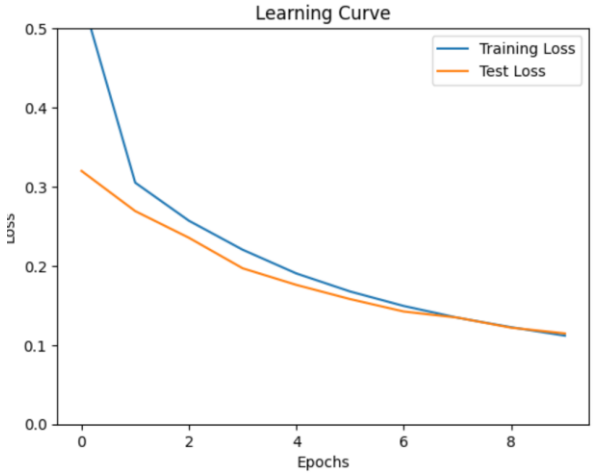
単純なニューラルネットワークの例



畳み込みニューラルネットワークの例

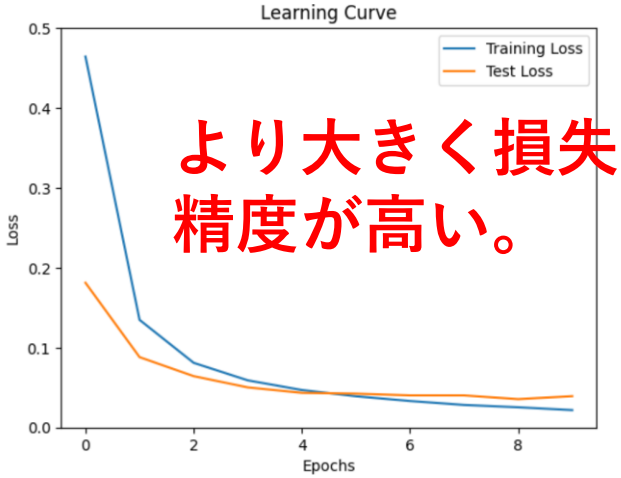


手書き文字 (0から9) MNIST の分類での損失



学習の繰り返しにより、損失は減少

手書き文字 (0から9) MNIST の分類での損失



より大きく損失は減少。
精度が高い。

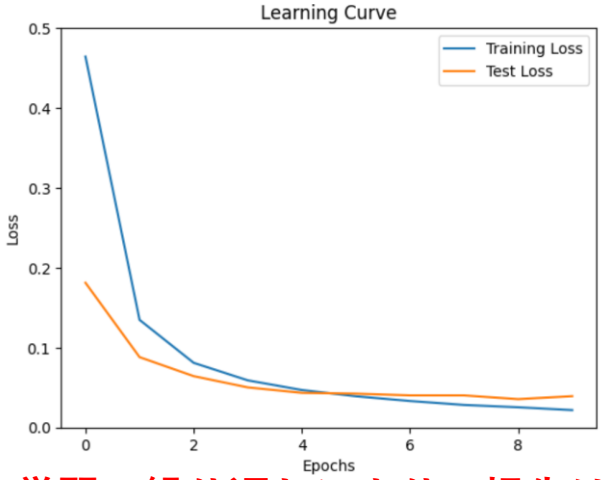
学習の繰り返しにより、損失は減少

ResNet による改善の効果

畳み込みニューラルネットワークの例

畳み込み層 ⇨ 畳み込み層 ⇨ プーリング層 ⇨ 全結合層 ⇨ 全結合層

手書き文字 (0から9) MNIST の分類での損失

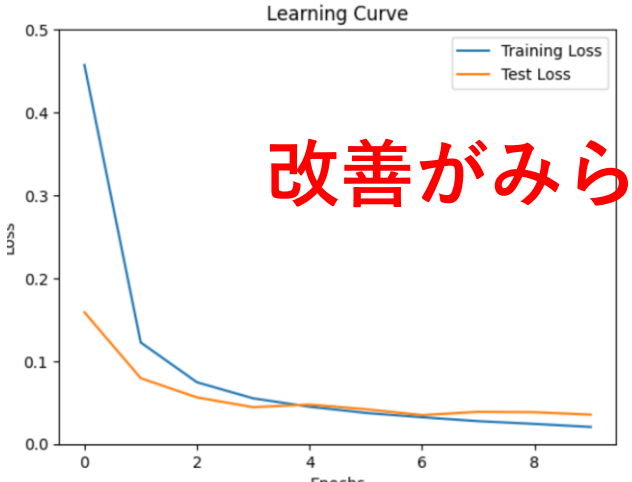


学習の繰り返しにより、損失は減少

ResNet の残余ブロックの導入

残余ブロック ⇨ **残余ブロック** ⇨ プーリング層 ⇨ 全結合層 ⇨ 全結合層

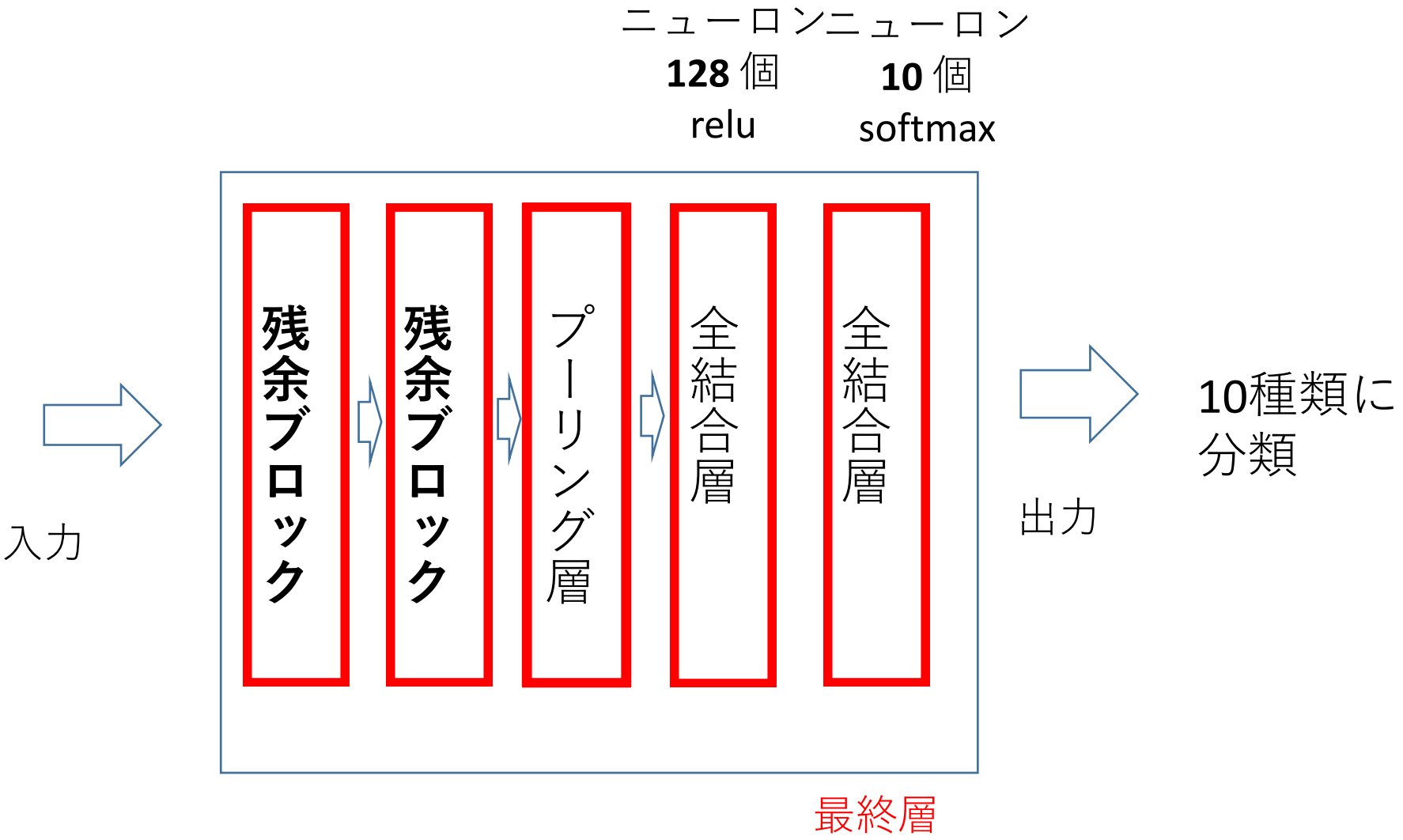
手書き文字 (0から9) MNIST の分類での損失



改善がみられる

学習の繰り返しにより、損失は減少

演習２の畳み込みニューラルネットワーク



ここまでのまとめ

手書き文字 MNIST データセット

- 0～9の手書き数字、28×28ピクセル、60,000枚の訓練データ、10,000枚の検証データ

手書き文字 MNIST データセットの分類

- 数字0～9に分類、訓練データでモデルを学習、検証データでモデルを検証

ResNet (2015年)

- 残余接続と残余ブロックを導入
- 勾配消失の緩和

9-3. 画像分類と学習済みモデルの活用

ImageNet-1K データセットの概要

- **画像の総数**：カラー画像 **約120万枚**
- **クラス数**：**1000 種類**に分類済み
- **画像アノテーション済み：**特定のクラスのオブジェクトの有無
- 位置情報： 45万枚については，オブジェクトの位置と大きさのデータも付与

```
Class ID: 0, Class Name: tench
Class ID: 1, Class Name: goldfish
Class ID: 2, Class Name: great white shark
Class ID: 3, Class Name: tiger shark
Class ID: 4, Class Name: hammerhead
Class ID: 5, Class Name: electric ray
Class ID: 6, Class Name: stingray
Class ID: 7, Class Name: cock
Class ID: 8, Class Name: hen
Class ID: 9, Class Name: ostrich
```

1000種類のうち 0〜9番のクラス名

文献: ImageNet Large Scale Visual Recognition Challenge
Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, Li Fei-Fei

COCO データセットの概要

- **用途：画像理解**
- **データの特徴**
  - **80のカテゴリ**
  - 数十万枚の画像
  - **複雑なシーン**
  - **小さなオブジェクト**を含む**現実的なシーン**

学習済みモデルの活用メリット

既存の大規模データを用いたAI学習

- コスト削減：
  - **学習の手間を削減**
- 高い性能：
  - **大規模データセットでの学習により優れた性能を持つ**
  - 様々なタスクで高い精度を実現
- 学習済みモデルの調整可能（転移学習，ファインチューニング）

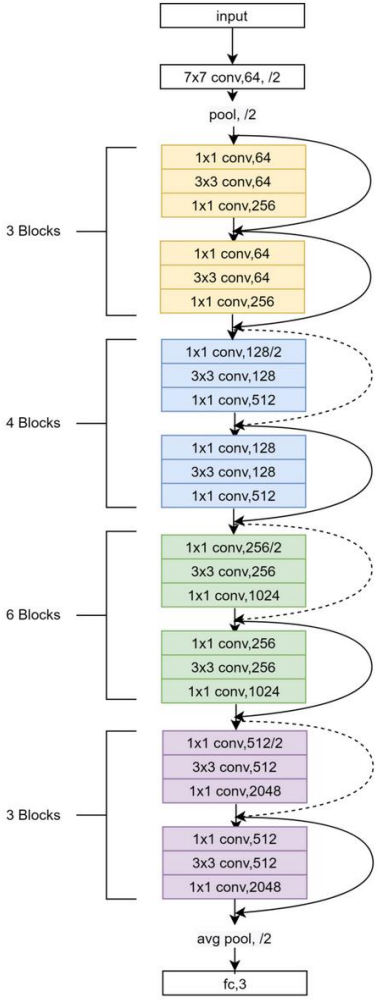
演習 3

学習済みモデルを活用した画像分類

演習の目的

次の理解

- 1000種類への画像分類
- ImageNet-1K で学習済みの ResNet-50 (ResNet の一種で，層の数は 50）の利用



ResNet-50 の構造

ResearchGate の次のページから引用
https://www.researchgate.net/figure/An-illustration-of-ResNet-50-layers-architecture_fig1_350421671

① Google Colaboratory のページを開く
https://colab.research.google.com/drive/18IPPkY96Oc6jkYD2su4cFgWcoYAskLo_?usp=sharing

② プログラムや説明や実行結果が掲載されていることを確認。

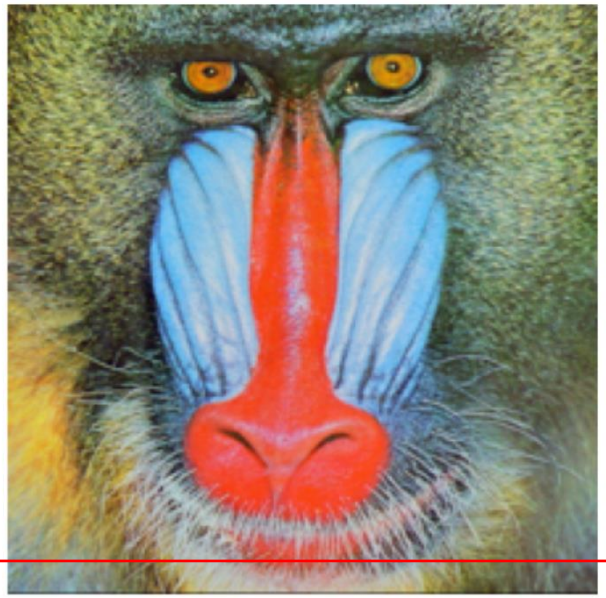
各自でよく読む。

③「**3. カラー画像分類（教師あり学習）（ImageNet-1Kで学習済みのResNet-50モデルを使用）**」を確認。

3. カラー画像分類（教師あり学習）（ImageNet-1Kで学習済みのResNet-50モデルを使用）

【概要】

使用モデル：ResNet-50（ImageNet-1K データセットで学習済み）
特徴：ResNet-50は畳み込みニューラルネットワークの一種。ResNet の残余ブロックの利用を特徴とする。



Predicted for baboon.jpg: [('baboon', 14.075982093811035), ('guenon',

分類結果が表示される

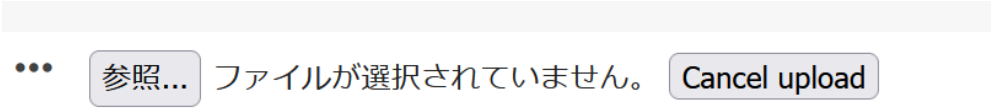
発展学習

自分で画像ファイルを用意して、**画像分類を実行**。そのことで、精度や機能についてより深く理解

① 画像ファイルを準備

② いまの Google Colaboratory のページの「**4. 自分で準備した画像の画像分類**」のコードセルを実行

③ 画面の指示に従い、画像ファイルを選択することで、アップロード



参照をクリックし，画像ファイルを選ぶ

④ 結果を確認

うまく行かないと思ったときでも別の画像で試す

9-4. さまざまな画像理解のタスク

画像理解

- **さまざまな応用**：スマホ，デジカメ，自動車，ロボット
- **さまざまな種類**：画像分類，物体検出，セグメンテーション，顔画像処理，姿勢推定など

元画像

画像の画素単位で分類を行うセグメンテーションの結果

画像理解の主な種類

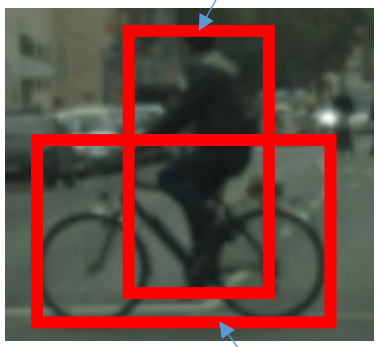
① **画像分類**

「何があるか」を理解

person
bicycle

② **物体検出**

場所と大きさも理解



③ **セグメンテーション**

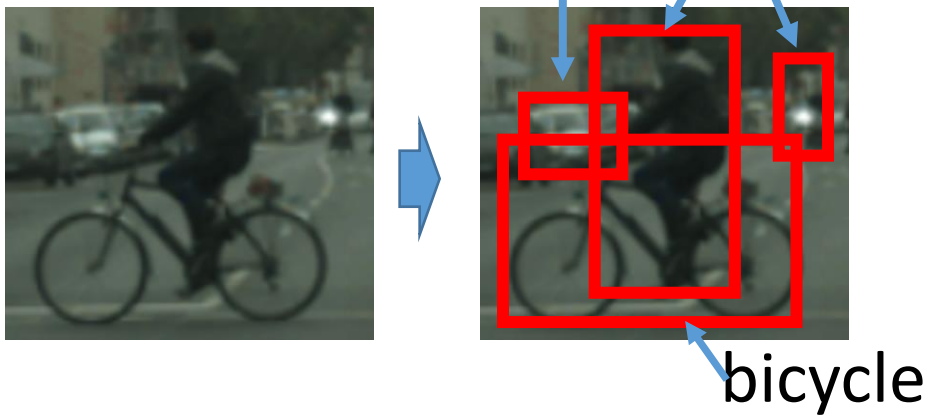
画素単位で理解

① 画像分類

```
Score 0.9827020168304443, Label lab_coat
Score 0.003087261691689491, Label syringe
Score 0.002431107917800545, Label beaker
Score 0.001660922775045037, Label stethoscope
Score 0.0003795088559854775, Label plate
```

画像分類の結果は，ラベルと**確率**
※ 5 つの候補 (top 5) が表示されている

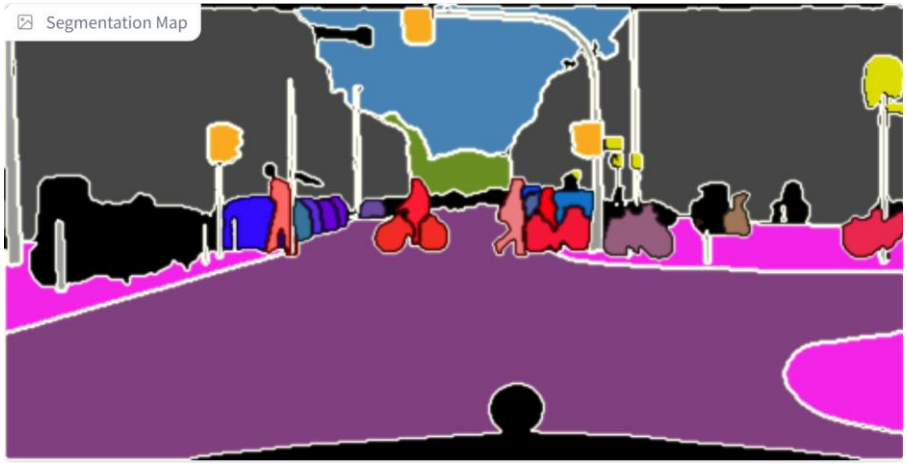
② 物体検出



バウンディングボックス，
ラベルを得る

バウンディングボックスは，
物体を囲む最小のボックス（四角形）

③ セグメンテーション



物体の形を画素単位で抜き出し

ラベルを得ることもできる

画像理解の応用例：医療分野

画像内の差異の抽出（傷，汚れ，病変など）

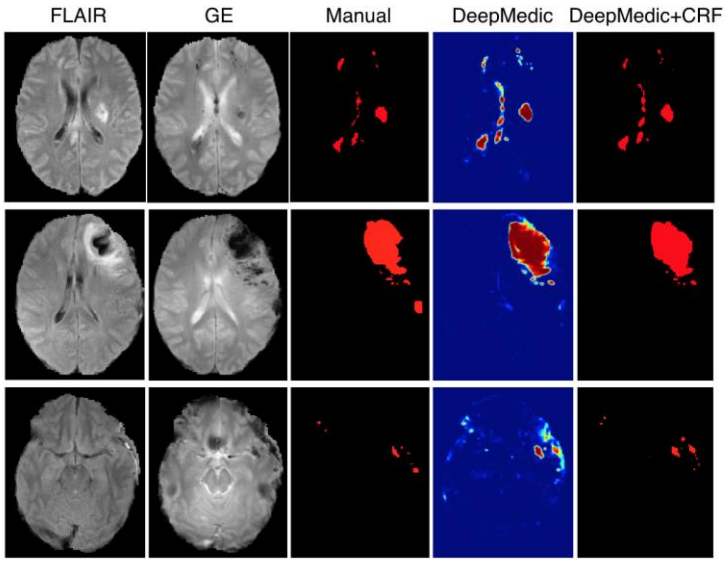


Figure 11: Three examples from the application of our system on the TBI database. It is capable of precise segmentation of both small and large lesions. Second row depicts one of the common mistakes observed. A contusion near the edge of the brain is under-segmented, possibly mistaken for background. Bottom row shows one of the worst cases, representative of the challenges in segmenting TBI. Post-surgical sub-dural debris is mistakenly captured by the brain mask. The network partly segments the abnormality, which is not a celebral lesion of interest.

脳内の病変の抽出

Efficient Multi-Scale 3D CNN with Fully Connected CRF for Accurate Brain Lesion Segmentation, Konstantinos Kamnitsas, Christian Ledig, Virginia F.J. Newcombe, Joanna P. Simpson, Andrew D. Kane, David K. Menon, Daniel Rueckert, Ben Glocker, arXiv: 1603.05959, 2016.

画像理解のまとめ

画像理解の主な種類

- **画像分類**

「何があるか」を理解。結果は「ラベル」として識別

各ラベルに対する「確率」も提供

- **物体検出**

物体の種類、場所、大きさを理解。場所と大きさについて

の結果は、物体を囲むバウンディングボックス。

セグメンテーション

画素単位で理解。

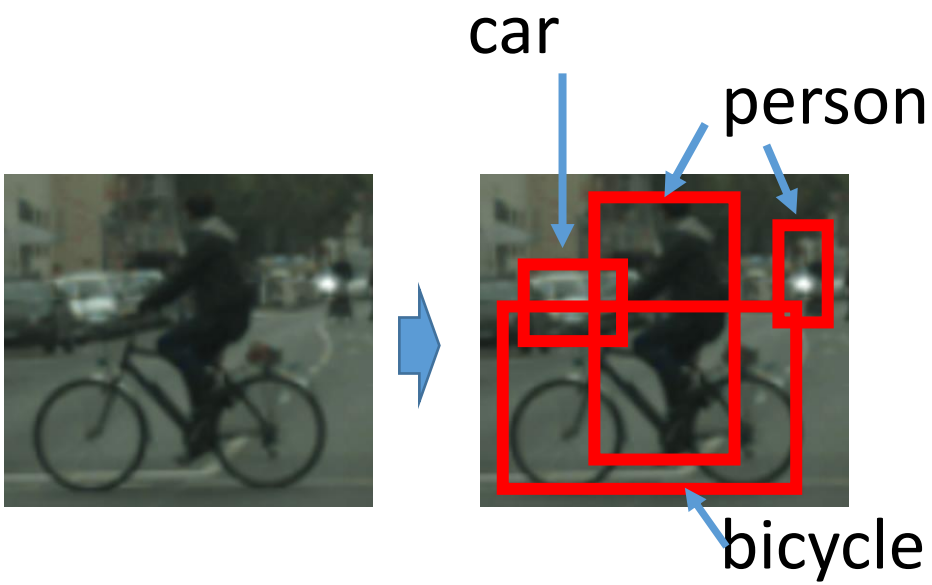
物体の「形」を詳細に抽出。

9-5. 物体検出

物体検出の例

Tree: 65%
Chair: 85%
Chair: 61%
Chair: 69%
Table: 75%
Chair: 47%
Chair: 57%
Table: 84%
Chair: 38%
0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000
1200

物体検出の特質



バウンディングボックス，ラベルを得る

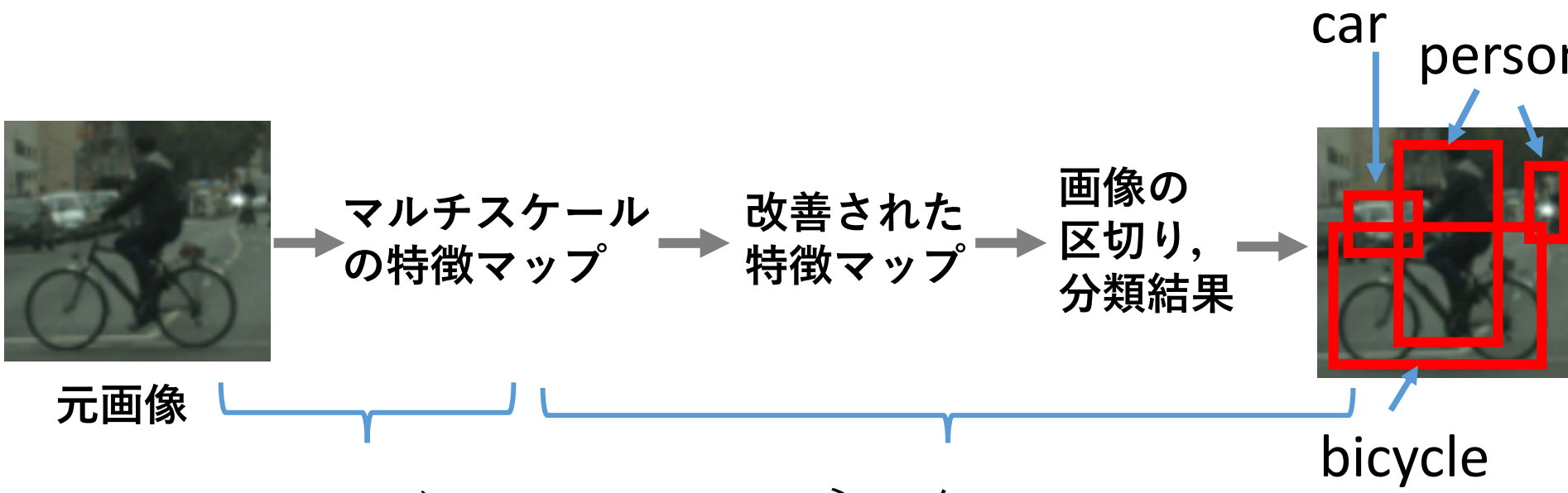
- **スケールの多様性**：画像内には**大小さまざまな物体**が存在

⇒　**バックボーン**，**ネック**の技術

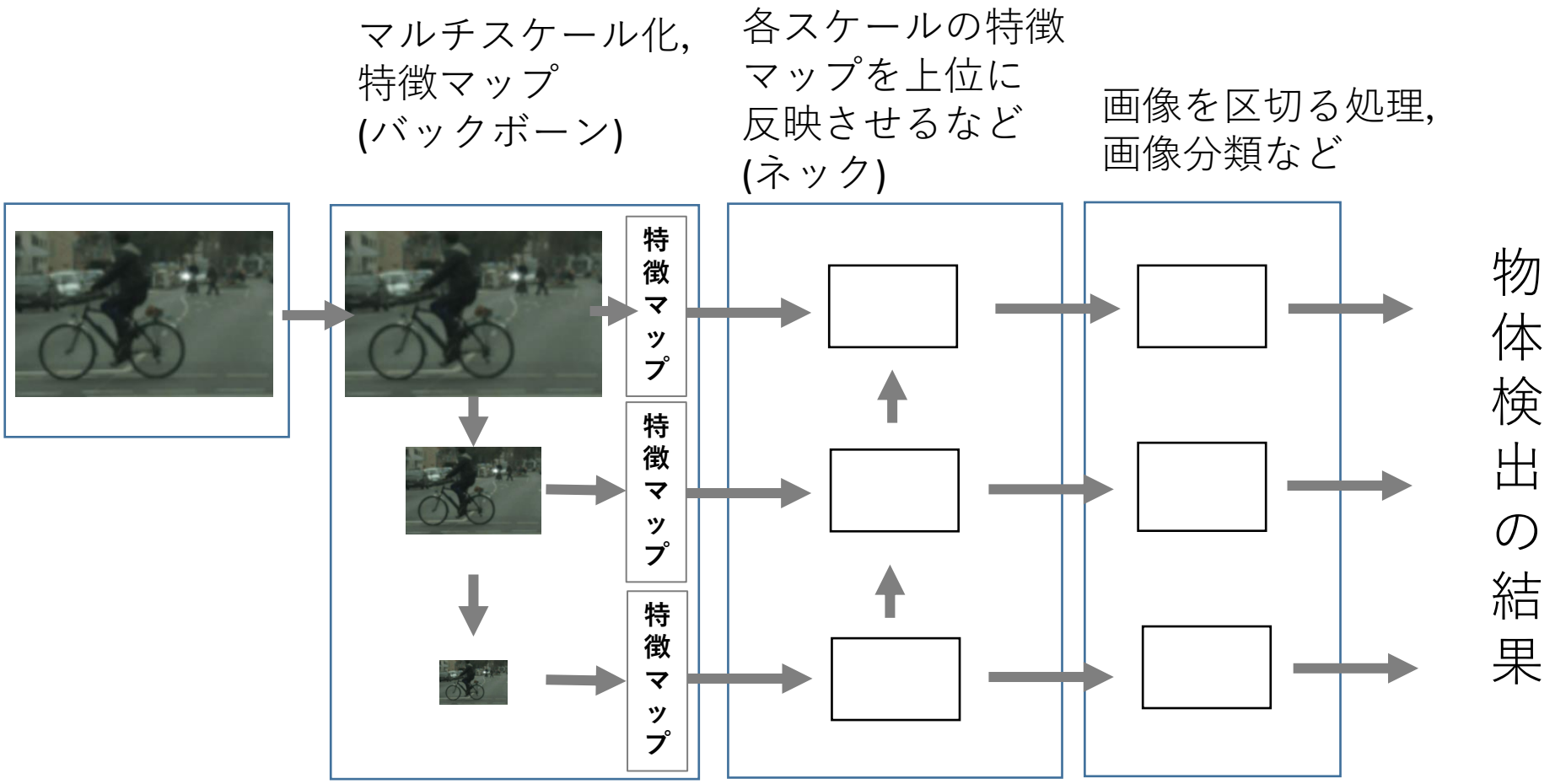
特徴マップ：画像の中の特定のパターンがどこにあるかなど示したマップ

バックボーンとネック

- **バックボーン**：　画像全体から**多様なサイズと形状の特徴**（マルチスケールの**特徴マップ**）を**抽出．基盤となる特徴抽出器としての役割**
- **ネック**：　バックボーンで得られた**特徴マップ**を**改善，統合．さまざまなサイズの物体を効率的に検出**



バックボーンとネック



全体で多層のニューラルネットワーク

YOLOの特徴と構造

主な特徴

- 物体検出のための高性能モデル
- 畳み込みニューラルネットワークを基盤とする
- 高速な動作と高い検出精度を両立

基本構造

- バックボーン：特徴抽出
- ネック：特徴マップの改善と統合

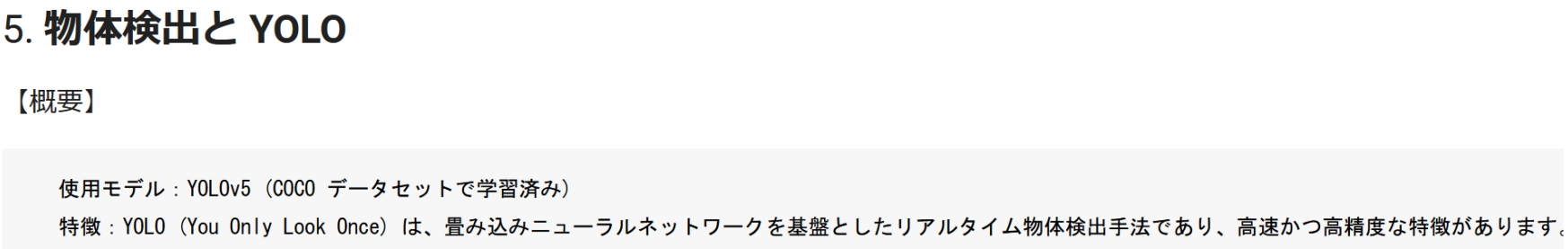
演習４．YOLO による物体検出

① Google Colaboratory のページを開く
https://colab.research.google.com/drive/18IPPkY96Oc6jkYD2su4cFgWcoYAskLo_?usp=sharing

② プログラムや説明や実行結果が掲載されていることを確認。

各自でよく読む。

③「**5. 物体検出と YOLO**」を確認。



発展学習

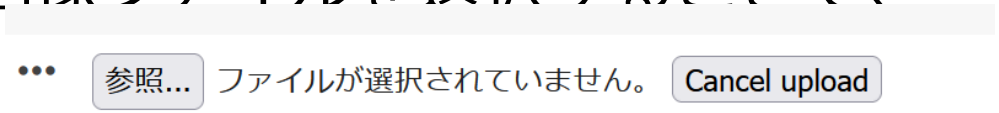
余裕のある人は、**自分で画像ファイルを用意**して、**画像分類を実行**。そのことで、精度や機能についてより深く理解

④ 前のページの①，②，③を行う

⑤ 画像ファイルを準備

⑥ いまの Google Colaboratory の「**5. 物体検出と YOLO**」のコードセルを実行

⑦ 画面の指示に従い、画像ファイルを選択することで、アップロード



参照をクリックし，画像ファイルを選ぶ

⑧ 結果を確認

うまく行かないと思ったとき
でも別の画像で試す

9-6. セグメンテーション

セグメンテーションの例

セグメンテーションは、**画像の画素の種類を識別**

セグメンテーションの例

セグメンテーション

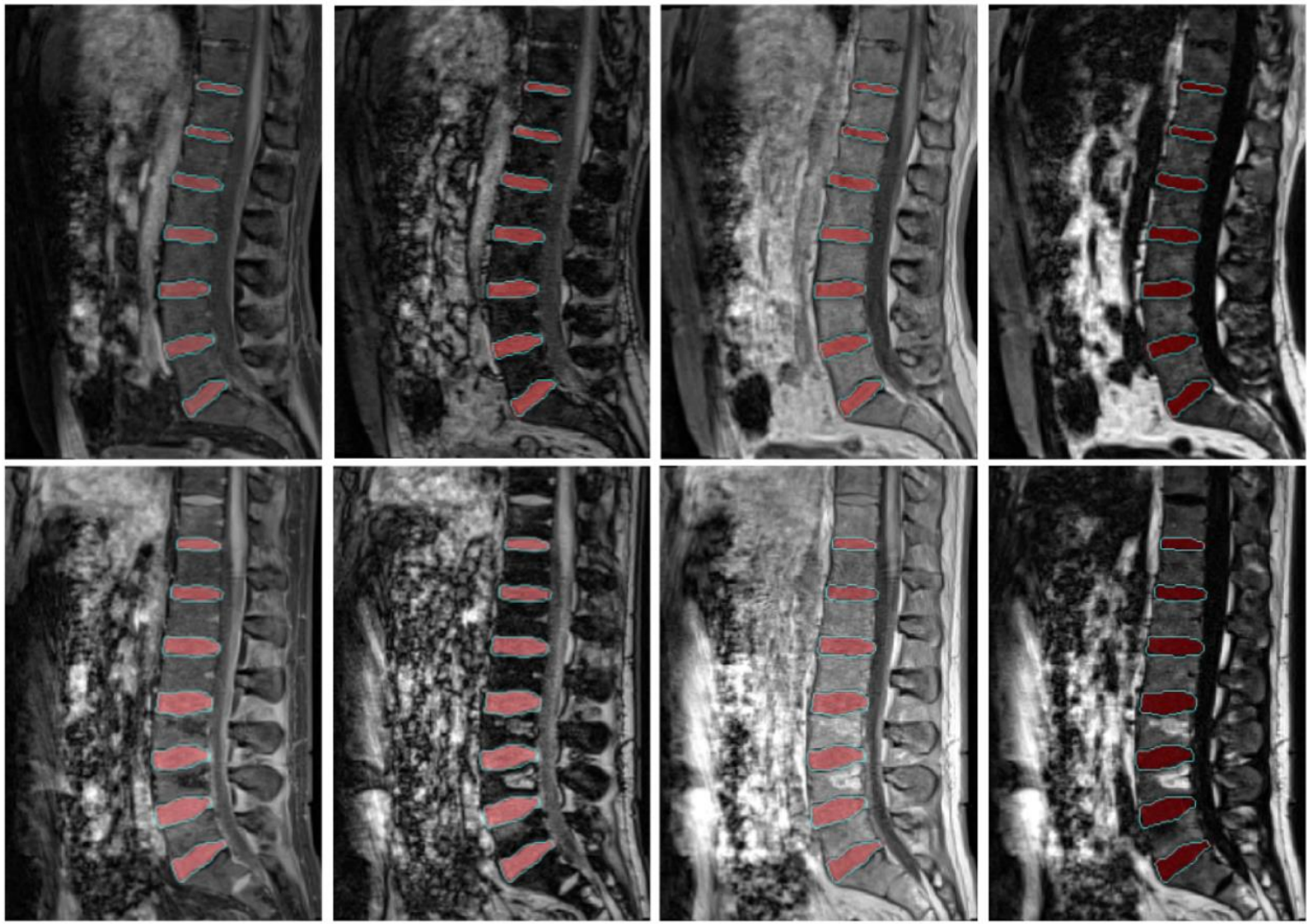
セグメンテーションの目的

- 画像を、**意味のある領域ごと**に分割
- 物体の「形」を詳細に解析

セグメンテーションの利点

- **物体**が**画像のどこにあるか**を正確に知ることができる
- **物体の形**や**大きさ**を**数値化**するための基礎

医療画像のセグメンテーション



https://github.com/josedolz/IVD-Net より転載

自動運転でのセグメンテーション

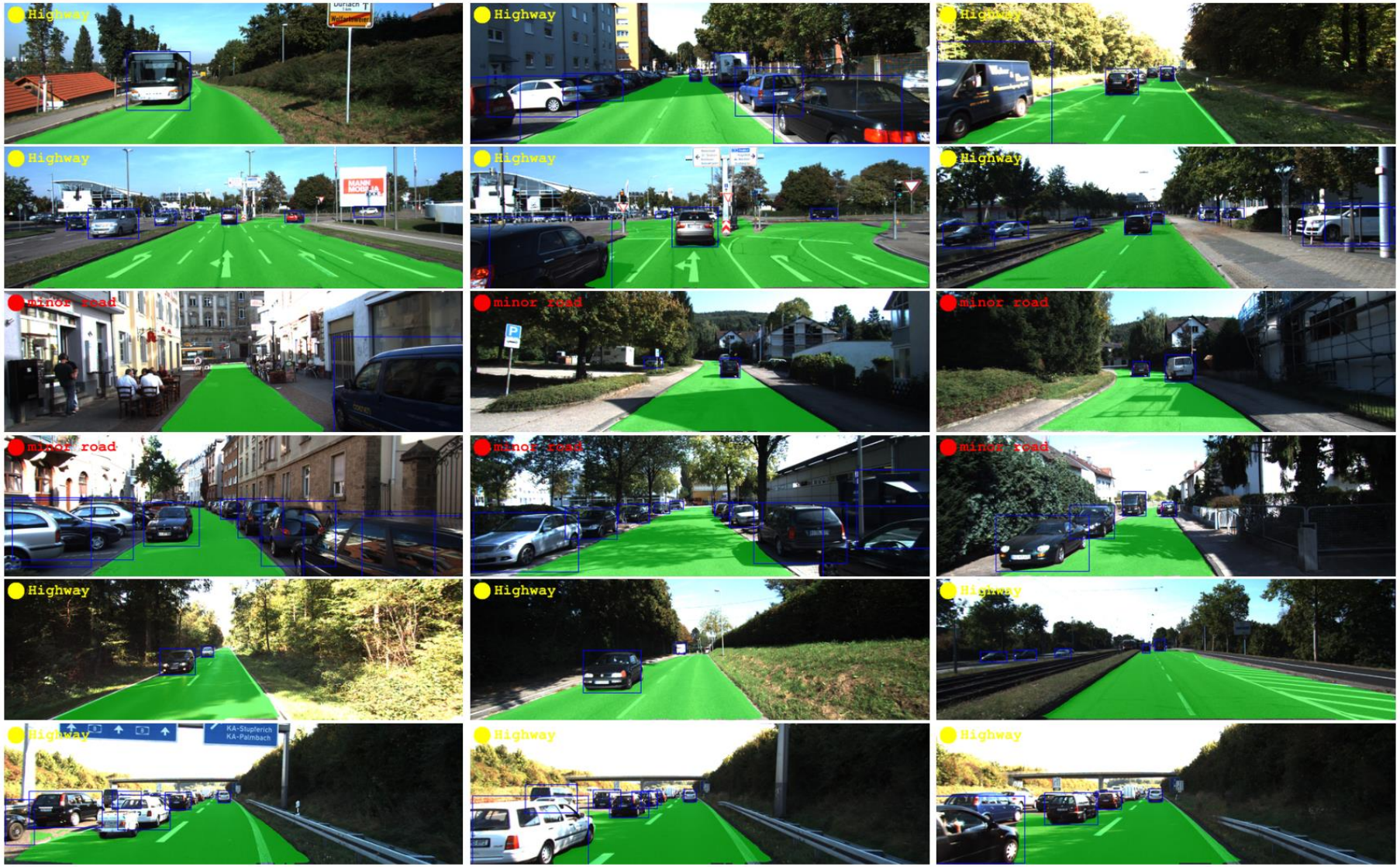
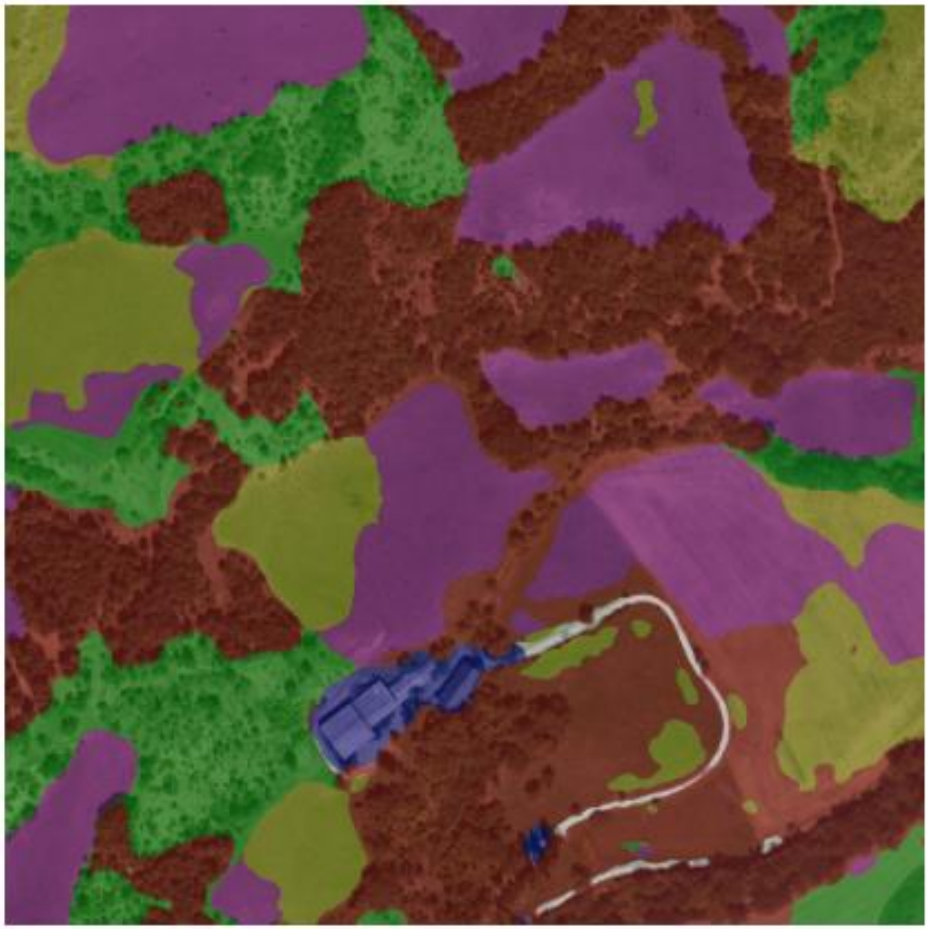


Figure 6: Visualization of the MultiNet output.

https://arxiv.org/pdf/1612.07695v2.pdf より転載

衛星画像のセグメンテーション



■ Dense forest | ■ Sparse forest | ■ Moor | ■ Herbaceous formation | ■ Building | □ Road

(■ No information)

https://arxiv.org/pdf/2110.05812.pdf より転載

演習５．ドキュメント画像のセグメンテーション

① デモページを開く

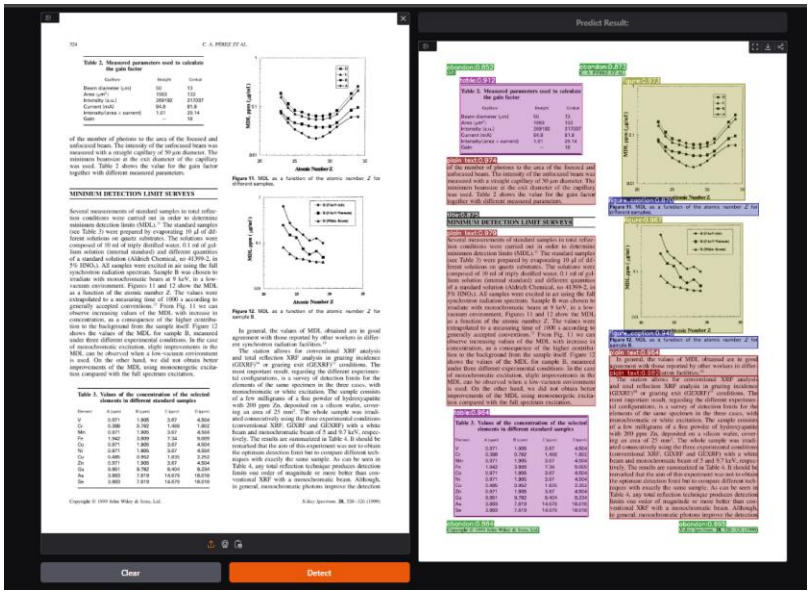
https://huggingface.co/spaces/opendatalab/DocLayout-YOLO

　オンラインサービスであり、混雑時などは動かない場合がある。授業中で動かなかった場合には、後日試してほしい

② 下の「**Examples**」で画像を選ぶ（画像をアップロードすることも可能）

③ 「**Detect**」 をクリック。結果を確認。

テキスト，図，キャプション，式，表を読み取り（色分け表示）



ここまでのまとめ

セグメンテーション

- セグメンテーションは、**画像の画素単位で、種類を識別**

セグメンテーションの目的

- 画像を、**意味のある領域ごと**に分割
- 物体の「形」を詳細に解析

セグメンテーションの利点

- **物体**が**画像のどこにあるか**を正確に知ることができる
- **物体の形**や**大きさ**を**数値化**するための基礎

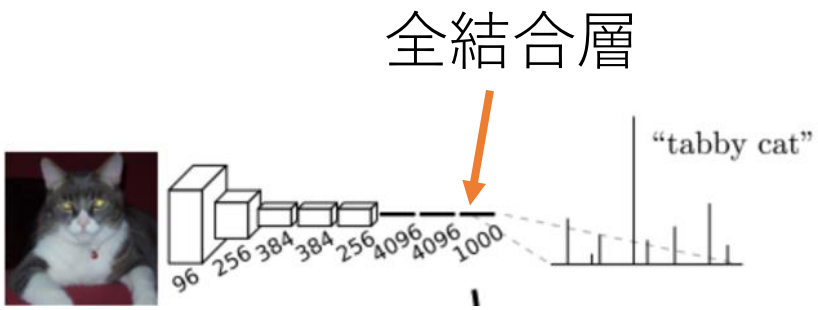
セグメンテーションを実現するFCN

- 畳み込みニューラルネットワーク（CNN）の**改良**
- **画素ごとに結果を出力**する

通常のニューラルネットワーク	**画像全体**に対する結果を出力（画像分類など）
FCN	**画素ごと**に結果を出力（セグメンテーションなど）

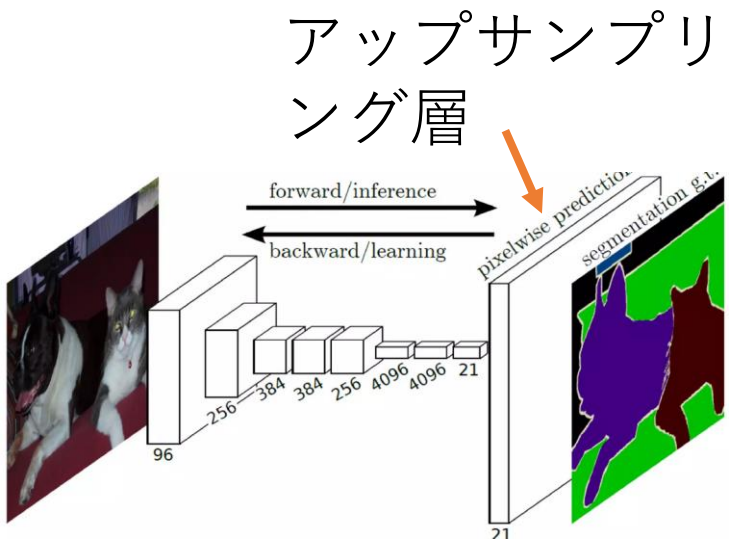
セグメンテーションを実現するFCN

- **畳み込み層**、**プーリング層**、**アップサンプリング層**で構成
- アップサンプリング層を持つことが特徴である。**アップサンプリング層**では、**特徴マップ**を、**元の入力画像の大きさまで拡大**



元画像　　結果

畳み込みニューラルネットワークによる**画像分類**



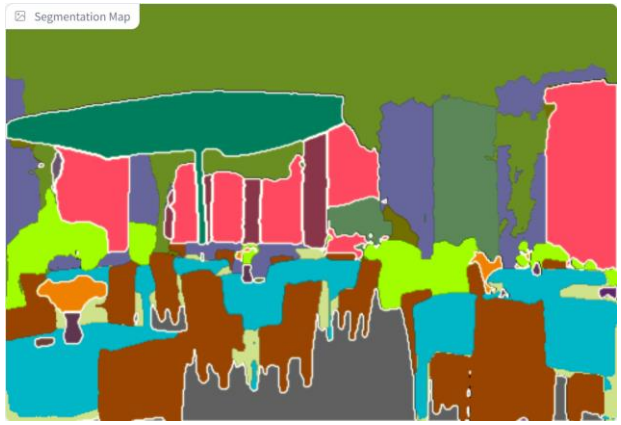
元画像　　結果

FCN による**セグメンテーション**

Jonathan Long, Evan Shelhamer, Trevor Darrell,
Fully Convolutional Networks for Semantic Segmentation, arXiv:1411.4038, 2015.

9-7. セグメンテーションのバリエーション

セグメンテーションの種類



セマンティック・セグメンテーション

画像内のすべての画素にラベルを付ける.
「個々の物体を検出する」という概念はない

インスタンス・セグメンテーション

物体を検出する．同じ種類の複数の物体がある場合は、別々のものとして認識．各物体に対してセグメンテーションを実施。

「物体として識別できない部分は結果がない」ということもある

パノプティック・セグメンテーション

セマンティック・セグメンテーションと
インスタンス・セグメンテーションの同時実行

セグメンテーションのバリエーション

- セマンティック・セグメンテーション

  「空」、「道路」などの広い領域の解析

- インスタンス・セグメンテーション

  物体の形状の分析、数の分析、変化の分析

- パノプティック・セグメンテーション

  複雑なシーンにおいて、セマンティックとインスタンスの両方の情報を同時に取得

インスタンス・セグメンテーションの例

画像を，インスタンス（物体）に分けセグメンテーションを実行

パノプティック・セグメンテーションの例

物体ではないエリアも対象である
（インスタンス・セグメンテーションでは、このようにはならない）

物体と物体の切れ目が判別できている
（セマンティック・セグメンテーションでは、このようにはならない）