人工知能(AI)法務、構築の試み(第２回)

(渡邉明彦 2020/07/19)

人工知能(AI)の、不思議な学習

【第１回の Recap】

前回(第１回)、ケーススタディの素材として、

「認知症診断の脳波解析ソフトウェアを日本で独占販売する契約書」のレビューを依頼された。」

「同ソフトウェアは、人口知能(AI)で脳波を解析し、認知症の診断を支援する。認知症患者など3000?以上の脳波を解析してデータベースを構築、解析アルゴリズムを開発した。90%以上の診断精度があるとされる。」

を考えよう。このような場合に、どこから手をつければよいのだろうか？

という仮説例を提示した。

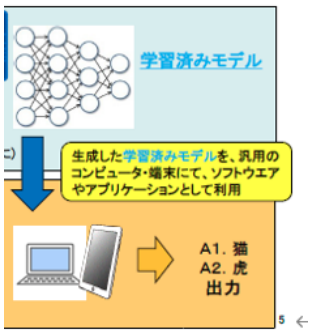
【検討の成果】は、Deep Learning の開発者の業績が知的な成果であることは疑い得ないが、例えば、MNIST のような手書き文字の判読ソフト自体は、機械的な処理しかおこなっておらず、その具体化されたソフトウェアなり、プログラムを、「知的な」作用をもつ実体と解釈してはならない。

- 「脳」で象徴されているような、複雑な操作は行われておらず、損失関数を逐次的に－機械的に！ー行っているのに過ぎない。

- 法的な分析のためには、「学習」、「教師あり」とか、「判定する」とかという、比喩的な表現にまどわされないために、あくまで、コード(スクリプト)レベルで、検討する必要がある。

【具体的な Takeaways】

第１回目の検討の、法務的な示唆は、次のようになるであろう。



左図の↓であるが「生成した学習済みのモデルを、汎用のコンピュータ・端末にて、ソフトウェアアプリケーションとして利用」させる、ソフトウエア・ライセンス契約ではなく、

例として m 行 X 列の重み W をもつニューラルネットワークでは、W は、

$$\boldsymbol{w} = \begin{pmatrix} w11 & w12 & \dots & w1m \\ w21 & w22 & \dots & w2n \\ & & & \\ wm1 & wm2 & \dots & wmn \end{pmatrix}$$

となる。AI 契約は、「最適化された重みの行列 W の利用権を提供・取得したり、ライセンスを受けることが本体となるデータ提供契約になるのではないか。

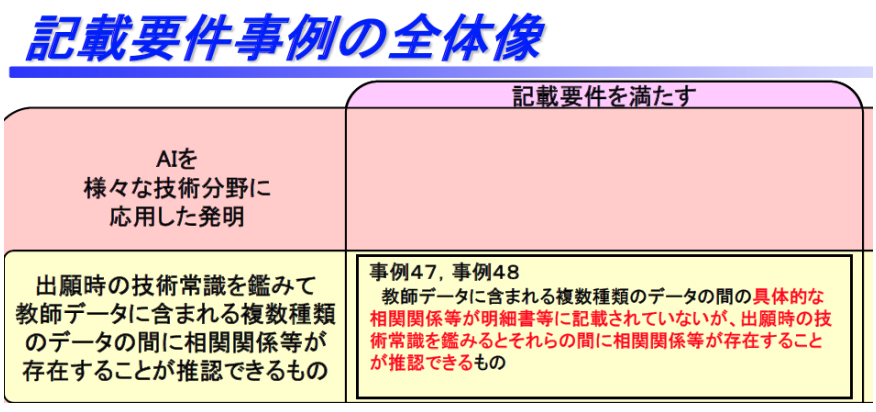
【第 2 回ーはじめに】

重ねて引用することになるが、特許庁調整課審査基準室『AI 関連技術に関する特許審査事例について』では、

- **IoT関連技術で収集された大量のデータの分析・学習は、AI関連技術の機械学習により実施されることが多い。**
- **機械学習には様々なものがあるが、近年では、コンピュータの飛躍的な計算性能向上等により、多層構造のニューラルネットワークを用いたディープラーニング（深層学習）が実施可能となり、大量のデータに基づいて高品質な学習済みモデルの生成が実現されてきている。**
- **生成した学習済みモデルは、未知のデータに対しても正解を出力することができる。**

と「学習」の意義が強調されており、

『記載要件事例の全体像』では、「教師データ」が次のように位置づけられている。



また、記載要件を満たす事例 47 には、【課題を解決するための手段】中に、「教師データを用いて、教師あり機械学習によりニューラルネットワークを学習させる。」との記載がある。（この事例は、「回帰問題」で、MNIST のような「分類問題」ではないが、ここでは、その点には立ち入らな

い。）

では、この「学習」、「教師」、「教師あり機械学習」とは、具体的に、何なんだろうか？また、「(ソース)コードレベル」で検討してみたい。

これからも、さかんにお世話になるであろう書籍・資料を引用する際の conventions を取り決めておきたい。
斉藤： 斉藤 康毅●著『ゼロから作る Deep Learning』(オライリー・ジャパン 2016 年)
Müller: Andreas C. Müller 他●著(中田訳)『Python で始める機械学習』(オライリー・ジャパン 2017 年)

【教師は、どこに？】

「教師」の存在を、斉藤によって確かめてみよう。

MINIST は、「機械学習用の画像データとしては最も基本的かつよく使われる、分類コンテスト用のデータセット」です。

```
import sys, os
sys.path.append(os.pardir)
from dataset.mnist import load_mnist

(x_train, t_train), (x_test, t_test) = load_mnist(flatten=True, normalize=False, one_hot_label=False)

print("x_train:", x_train.shape)
print("t_train:", t_train.shape)
print("x_test:", x_test.shape)
print("t_test:", t_test.shape)
```

```
x_train: (60000, 784)
t_train: (60000,)
x_test: (10000, 784)
t_test: (10000,)
```

このデータセットをダウンロードすると、x_train という訓練画像が 60, 000 枚、x_test というテスト画像が 10, 000 枚提供されます。

訓練画像の 1 枚目は 

で、「5」でしょうか(笑)？

```
import sys, os
sys.path.append(os.pardir)
import numpy as np
from dataset.mnist import load_mnist
from PIL import Image

def img_show(img):
    pil_img = Image.fromarray(np.uint8(img))
    pil_img.show()

(x_train, t_train), (x_test, t_test) = load_mnist(flatten=True, normalize=False, one_hot_label=False)

img = x_train[0]
label = t_train[0]
print(label)

print(img.shape)
img = img.reshape(28, 28)
print(img.shape)

img_show(img)
```

```
5
(784,)
(28, 28)
```

この正解は、「ラベル」とよばれ、同じく 60, 000 ある t_train におさめられており、表示すると「5」で、「5」が正解のようです。

教師あり学習は、画像データと教師ラベルが対になって与えられたデータセットを利用されることから、「教師」は、t_train のようなデータの集まりということになります。(訓練画像と教師ラベルを、テストデータと対比する意味で、「訓練データ」とか「教師データ」とか呼ぶ場合もあるようです。

ともかくも、「教師」は、label = t_train にありました。ただし、「いた」のではないようです(笑)。

```
img = x_train[0]
label = t_train[0]
print(label)

print(img.shape)
img = img.reshape(28, 28)
print(img.shape)

img_show(img)
```

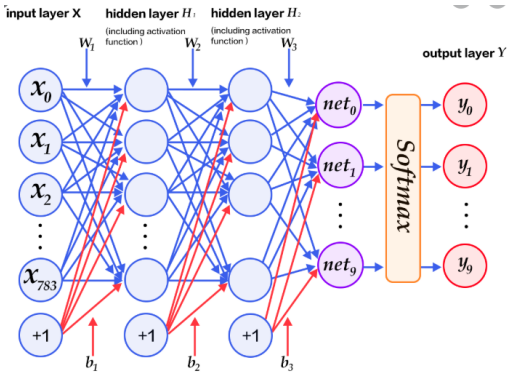
```
5
```

【「学習」とは、何をやっているのだろうか？】

今回の冒頭にあげた重みの行列 W、

$$\boldsymbol{w} = \begin{pmatrix} w11 & w12 & \dots & w1m \\ w21 & w22 & \dots & w2n \\ & & & \\ wm1 & wm2 & \dots & wmn \end{pmatrix}$$

は、次のようなネットワークのデザインのよって決まってくるので、そのようなデザインが(暫定的にも)完了し、W の枠組みが決まるとしよう。



次の作業は、W を具体的に決定していくことだが、先ず適切な「重みの初期値」(斉藤 177 ページ)が設定される。斉藤では、重みとバイアスのパラメーターがおさめられている pickle ファイルを用いて、途中からの解説になっているが、

1. 「重みの初期値」から始まり、次の段階を予測する predict 関数により、各画像データ1枚ずつを分類して、各ラベルの確率を求める。

例: y = [0.1, 0.05, 0.6, 0.0, 0.05, 0.1, 0.0, 0.1, 0.0, 0.0]

のような行ベクトル(Numpy 配列)が得られる。

2. t_train には、正解が与えられており、正解ラベルを 1 とし、その他は 0 とする「one-hot 表現」の行ベクトルと比較する。

t = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

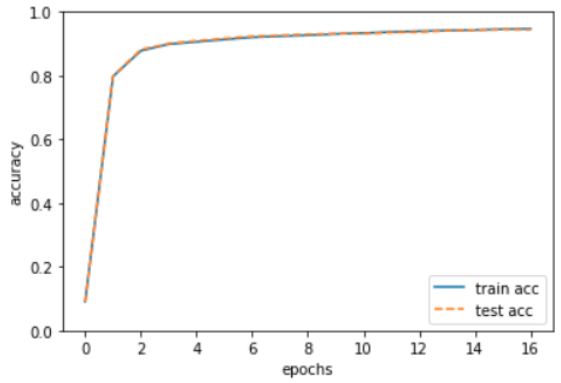
この場合「2」が正解で、この段階での「2」と判定される確率は「0. 6」になる。

3. 正解とのズレは、認識精度として把握され、この齟齬を少なくしていける W を、逐次、求め

ていく。
累積的な認識精度（Accuracy）は

```python
def init_network():
    with open("sample_weight.pkl", 'rb') as f:
        network = pickle.load(f)
    return network

def predict(network, x):
    W1, W2, W3 = network['W1'], network['W2'], network['W3']
    b1, b2, b3 = network['b1'], network['b2'], network['b3']

    a1 = np.dot(x, W1) + b1
    z1 = sigmoid(a1)
    a2 = np.dot(z1, W2) + b2
    z2 = sigmoid(a2)
    a3 = np.dot(z2, W3) + b3
    y = softmax(a3)

    return y

x, t = get_data()
network = init_network()

batch_size = 100
accuracy_cnt = 0

for i in range(0, len(x), batch_size):
    x_batch = x[i:i+batch_size]
    y_batch = predict(network, x_batch)
    p = np.argmax(y_batch, axis=1)
    accuracy_cnt += np.sum(p == t[i:i+batch_size])
```

で求められ、次第に認識精度が高まっていく様子は、下図のようになる。（なお、テストデータとの検証の問題ー過学習の問題ーは、今回は検討していない。下図の test acc である。）



【今回の検討の成果】

■ 「教師あり学習」という場合の「教師」とは、データセットの正解をおさめたラベルでる。

■ 「学習」とは、各画像について得られた正解ラベルの確率と、one-hot 表現された正解ラベ

ルとのズレを少なくしていくプロセスを、擬人的に表現している。

■ コード（スクリプト）レベルで、「学習」とか、「教師あり」の意味するところを確認できた。

【第 2 回目の結語】

「学習」、「教師あり学習」の意味するところと確認し、認識精度を 95%のように数量化できることを知った。しかし、次回以降に検討するように、この認識精度は、AI システムの性能、あるいは性能保証に結びつけられないであろう。また、正解ラベルの付いたデータセットを用意する作業の評価と、認識精度が可能な限り高められた AI システムを、現実のデータでテストする実証試験の位置づけの方が、法的には重要になると思われる。

（第 2 回目終わり）