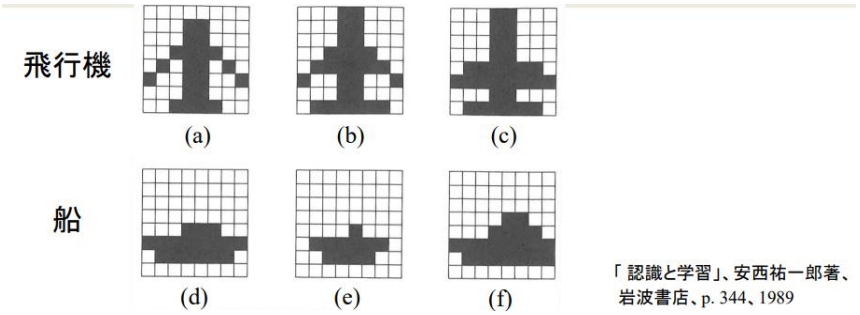
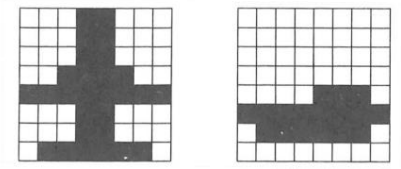
プログラム作品についてのご説明

＜制作について＞
・制作期間：三ヶ月
・制作人数：二人
（※在学中の大学での講義「ニューラルネットワーク」にて雛形ファイルが配布されたためです。）
・制作箇所：before_learning.cpp→（109～373 行目）＆（462～541 行目）
after_learning.cpp→一番上の関数である、readinout を除いた全て
（※ after_learning.cpp では before_learning.cpp の関数等をそのまま利用しています。）
（※ソースコードの方でも▽、△で囲って表しています。）

＜プログラムについて＞
・言語：C++
・目的：深層学習（ディープラーニング）※1 を構築する
・内容：下記の(a)～(f)の 6 種類の画像を学習させ、その上で、更に下の 2 種類の画像（左が飛行機、右が船を想定）を推定させる



「認識と学習」、安西祐一郎著、
岩波書店、p. 344、1989


「認識と学習」、安西祐一郎著、
岩波書店、p. 346、1989

※1 深層学習、またはディープラーニングとは、人間の神経細胞の仕組みを再現したニューラルネットワークを用いた機械学習の手法の 1 つであり、多層構造のニューラルネットワークを用いることが特徴です。現在では画像認識や音声認識、翻訳などさまざまな分野で大きな成果を生み出しています。
「深層学習とは？」.NTT コミュニケーションズ.
<https://www.ntt.com/bizon/glossary/j-s/deep-learning.html >（参照日 2023 年 3 月 6 日）.

＜zip ファイルの内容物について＞

・before_learning.cpp：最初にこちらを実行

・after_learning.cpp：次にこちらを実行

・inputfile.txt： before_learning.cpp を実行する際に使用

・outputfile.txt： 〃

・rondom_weight.txt： 〃

・complete_weight.txt： after_learning.cpp を実行する際に使用

・test.txt： 〃

（※もしお時間がありましたら、４ページ目の zip ファイルの内容物に関する説明もご覧ください。）

＜実行について＞

・「 ./before_learning -E 50000 -I inputfile.txt -W random_weight.txt -O outputfile.txt complete_weight.txt」

（※最初の方の数字の 50000 は学習回数の上限です。ただし過学習※2 を避けるため、十分に学習が完了したとみなすと学習を打ち切る仕様となっております。今回のケースですと、おおよそ 15～20 回で学習が完了致します。）

・「./after_learning test.txt complete_weight.txt」

※2「過学習」とは、「過剰適合」や「オーバフィッティング」とも呼ばれる現象で、学習データにだけ適応した学習ばかりが過剰に進んでしまい、AI の、未知のデータに対して推定する性能が下がってしまうことを指します。

「過学習」.三菱電機.

< https://www.mitsubishielectric.co.jp/corporate/special/hello-ai/words/kagakusyu.html >

（参照日 2023 年 3 月 6 日）.

＜結果について＞

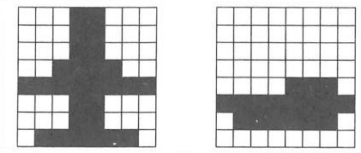
・before_learning.cpp：

```
PS C:\Users\kyei1\OneDrive\ドキュメント\デスクトップ\プログラミング作品> ./before_learning -E 50000 -I inputfile.txt -W random_weight.txt -O outputfile.txt complete_weight.txt
*****weight
*****inputdata
0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,1,1,0,0,0,1,0,1,1,0,1,0,1,0,0,1,1,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1,1,0,0
0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,1,1,0,0,0,1,1,1,1,1,1,0,1,0,0,1,1,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,0
0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,1,1,1,1,1,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0
*****outputdata
1,0
1,0
1,0
0,1
0,1
0,1
*****learn_finish
numepoch:17
data:1
3 1 0.929049
3 2 0.0695932
data:2
3 1 0.950658
3 2 0.0487916
data:3
3 1 0.898523
3 2 0.100743
data:4
3 1 0.0567738
3 2 0.946683
data:5
3 1 0.0791615
3 2 0.922911
data:6
3 1 0.0697011
3 2 0.929052
```

・after_learning.cpp：

```
PS C:\Users\kyei1\OneDrive\ドキュメント\デスクトップ\プログラミング作品> g++ after_learning.cpp -o after_learning
PS C:\Users\kyei1\OneDrive\ドキュメント\デスクトップ\プログラミング作品> ./after_learning test.txt complete_weight.txt
****test[0]_result
airplane:94.7964%
ship:5.94568%

****test[1]_result
airplane:7.27266%
ship:93.2164%
```



「認識と学習」、安西祐一郎著、
岩波書店、p. 346、1989

（※左：test[0]、右：test[1]）

・before_learning.cpp：教師データから最適な重みを学習し、学習後、最適な重みデータが記録されたファイルを生成します。ファイルの名前は何でも良いので、ここでは便宜上 complete_weight.txt とさせていただきます。

・after_learning.cpp：complete_weight.txt をもとに test.txt のデータを、行ごとに推定します。

・inputfile.txt：ニューラルネットワーク※3 の入力層における教師データで、＜プログラムについて＞で提示した画像(a)～(f)の 64 画素を、白を 0、黒を 1 として改行せずに一行で記述しています。

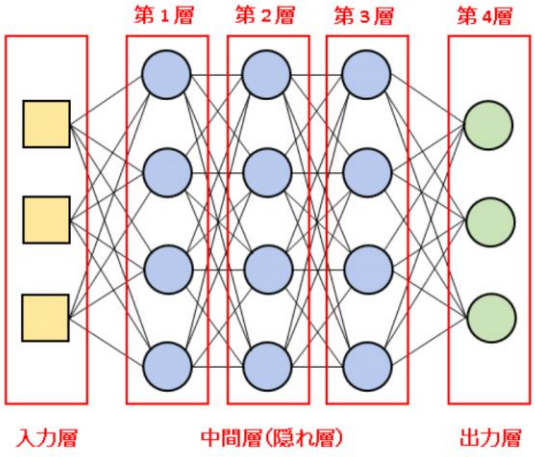
・outputfile.txt：ニューラルネットワークの出力層における教師データで、inputfile.txt の n 行目の答えを、(1,0)が飛行機、(0,1)が船として n 行目に記述しています。

・rondom_weight.txt：±0.3 の範囲で乱数生成された重みデータの初期値で、64×32×2 のニューラルネットワークの各辺の重み、計 2,112 個の乱数を階層ごとに記述しています。

・complete_weight.txt： before_learning.cpp を実行した際に生成されるファイルです。なのでこちらのファイルは zip ファイルに入れる必要は無いのですが、念のため入れさせて頂きます。

・test.txt：推定したいデータが入力されたもので、構造自体は inputfile.txt と同じです。

※3 ニューラルネットワーク



「ニューラルネットワーク(Artifical Neural Network)」.Python 数値計算入門.
< http://python.atelierkobato.com/neural/ > （参照日 2023 年 3 月 6 日）.

（※今回のケースですと、入力層のノード数が 64、第一層のノード数が 32、出力層（第二層）のノード数が 2 のニューラルネットワークです。）