PA

Part 1: Write the pattern file

a. input file

```
0: 1 1 1 1 1 1 0
1: 0 1 1 0 0 0 0
2: 1 1 0 1 1 0 1
3: 1 1 1 1 0 0 1
4: 0 1 1 0 0 1 1
5: 1 0 1 1 0 1 1
6: 1 0 1 1 1 1 1
7: 1 1 1 0 0 0 0
8: 1 1 1 1 1 1 1
9: 1 1 1 1 0 1 1
A: 1 1 1 0 1 1 1
b: 0 0 1 1 1 1 1
C: 1 0 0 1 1 1 0
d: 0 1 1 1 1 0 1
E: 1 0 0 1 1 1 1
F: 1 0 0 0 1 1 1
```

b. output file

```
0 (48): 0 1 1 0 0 0 0
1 (49): 0 1 1 0 0 0 1
2 (50): 0 1 1 0 0 1 0
3 (51): 0 1 1 0 0 1 1
4 (52): 0 1 1 0 1 0 0
5 (53): 0 1 1 0 1 0 1
6 (54): 0 1 1 0 1 1 0
7 (55): 0 1 1 0 1 1 1
8 (56): 0 1 1 1 0 0 0
9 (57): 0 1 1 1 0 0 1
A (65): 1 0 0 0 0 0 1
B (66): 1 0 0 0 0 1 0
C (67): 1 0 0 0 0 1 1
D (68): 1 0 0 0 1 0 0
E (69): 1 0 0 0 1 0 1
F (70): 1 0 0 0 1 1 0
H (72): 1 0 0 1 0 0 0
```

c pat file

ここまでの情報をもとにトレーニングに使用するテキストファイルを用意する。作成したファイルの内容は以下の通り。

```
Number of patterns = 17
Number of inputs = 7
Number of outputs = 7
[patterns]
1 1 1 1 1 1 0    0 1 1 0 0 0 0  # 0
0 1 1 0 0 0 0    0 1 1 0 0 0 1  # 1
1 1 0 1 1 0 1    0 1 1 0 0 1 0  # 2
1 1 1 1 0 0 1    0 1 1 0 0 1 1  # 3
0 1 1 0 0 1 1    0 1 1 0 1 0 0  # 4
1 0 1 1 0 1 1    0 1 1 0 1 0 1  # 5
1 0 1 1 1 1 1    0 1 1 0 1 1 0  # 6
1 1 1 0 0 0 0    0 1 1 0 1 1 1  # 7
1 1 1 1 1 1 1    0 1 1 1 0 0 0  # 8
1 1 1 1 0 1 1    0 1 1 1 0 0 1  # 9
1 1 1 0 1 1 1    1 0 0 0 0 0 1  # A
0 0 1 1 1 1 1    1 0 0 0 0 1 0  # B
1 0 0 1 1 1 0    1 0 0 0 0 1 1  # C
0 1 1 1 1 0 1    1 0 0 0 1 0 0  # D
1 0 0 1 1 1 1    1 0 0 0 1 0 1  # E
1 0 0 0 1 1 1    1 0 0 0 1 1 0  # F
1 0 0 1 0 0 0    1 0 0 1 0 0 0  # H
```

Part 2

作成した pat を利用して分析を行う。適切な分析となるようにパラメータの調整を行った結果を以下で説明する。

・ネットワーク設定

入力および出力ユニットは固定で7とある。Layer 1に関しては、パラメータの調整を行った結果、10が一番適切であったためこの設定とする。以下は設定のスクリーンショットである。

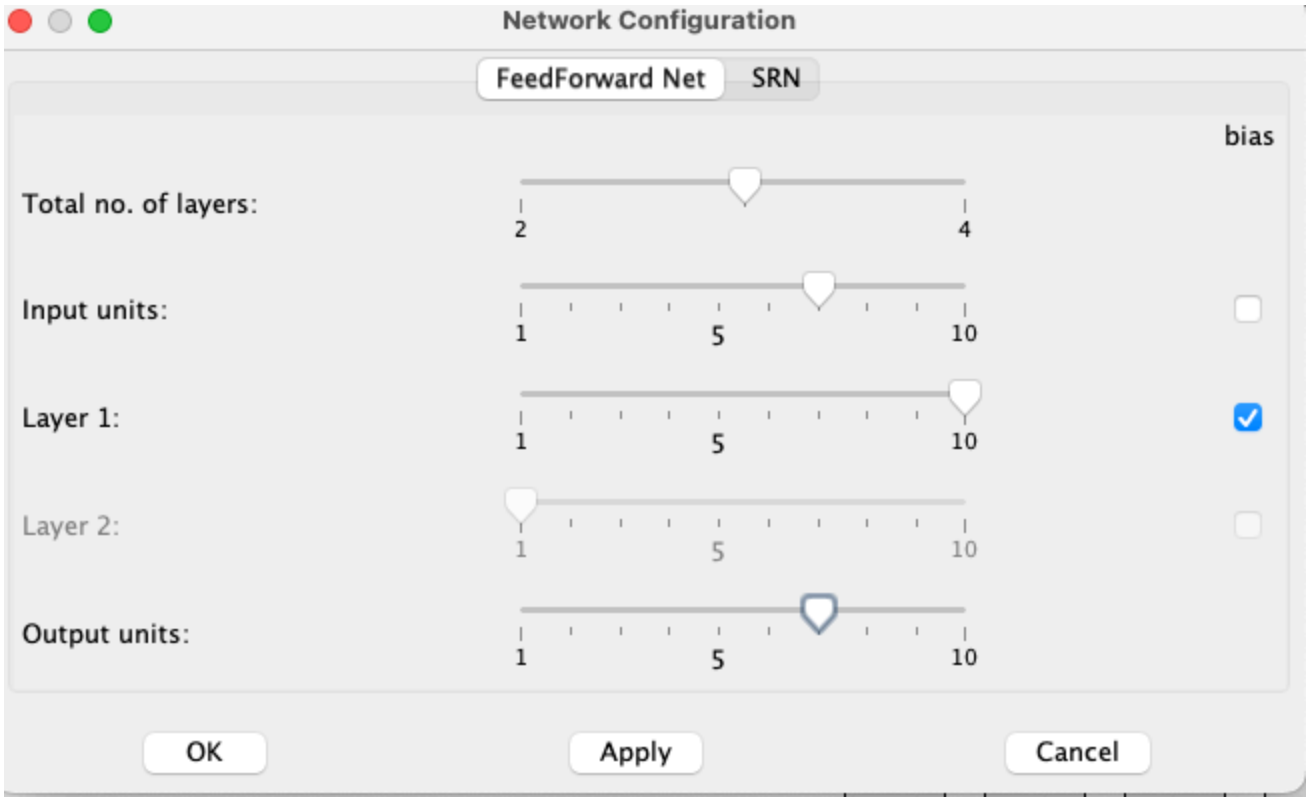

FIGURE1 Network Configuration

・コントロールパネル設定
コントロールパネルでは主に、「Laerning rate」と「Momentum」を設定することができる。デフォルト設定値はそれぞれ
Laerning rate: 0.3
Momentum: 0.8
であるが、今回の分析では、学習効率を上げるとError Progressが1のあたりに収束が発生してしまい、適切なモデルを構築することができないケースが存在した。以下は、適切ではないと考えるモデルのError progressの出力例:

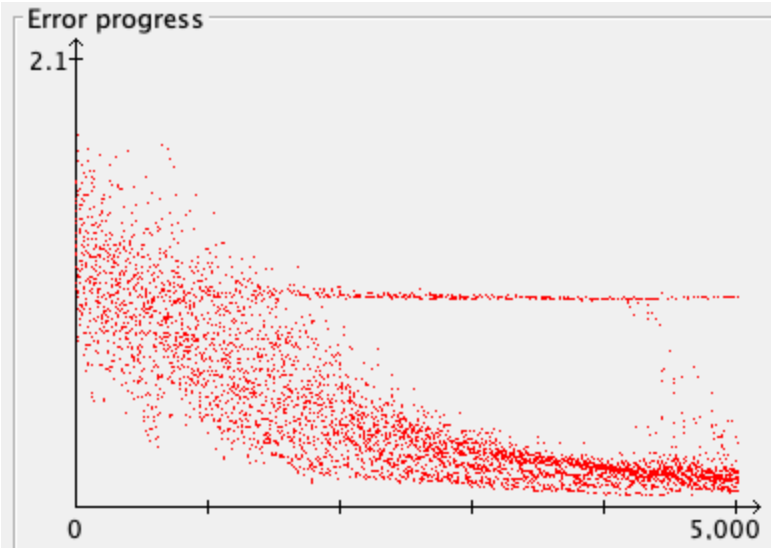

FIGURE 2 Bad pattern

そのため、学習効率を下げすぎることなく、適切ではないモデルの構築可能性を上げるため、今回は以下の設定を付与した。

Laerning rate: 0.2

Momentum: 0.8

実際のスクリーンショットは以下である。

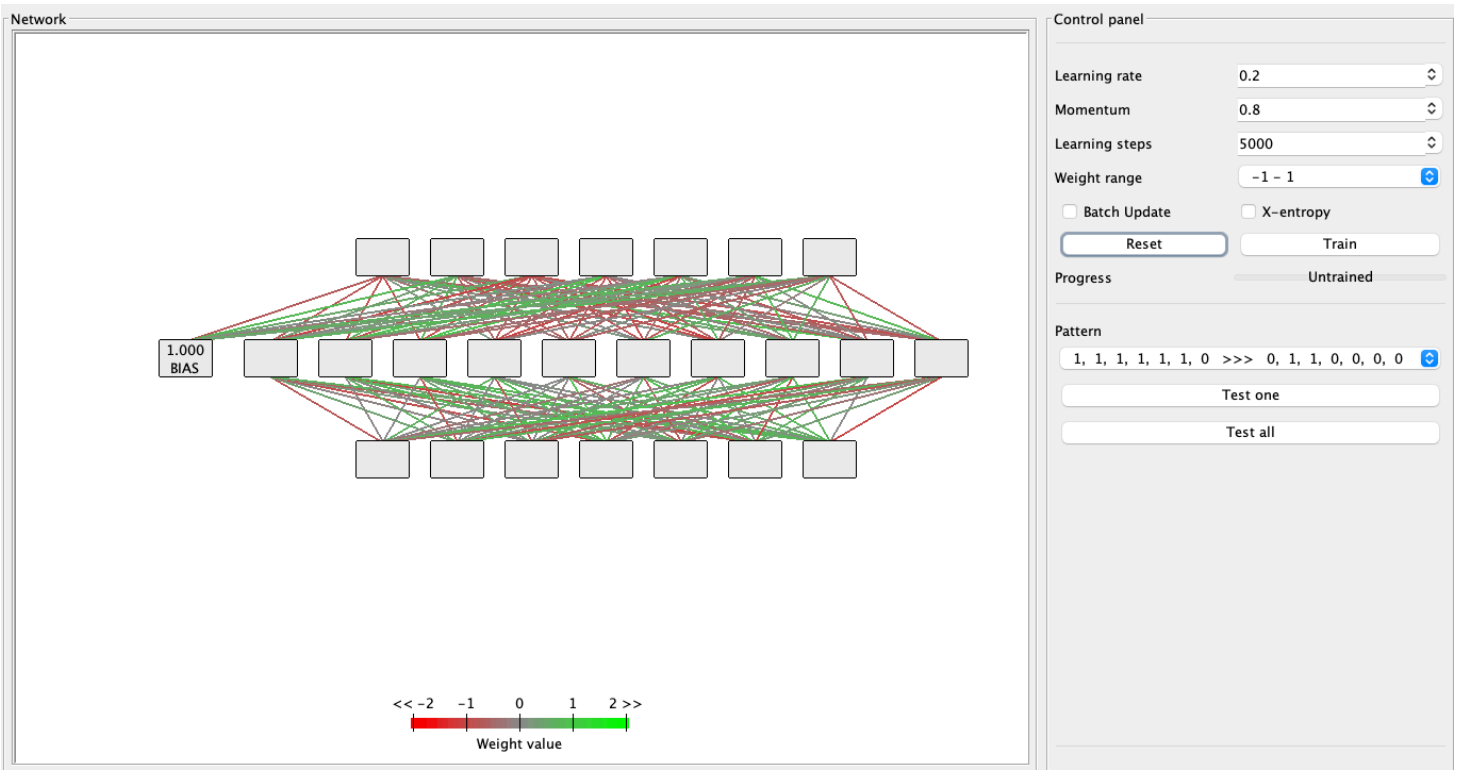

FIGURE 3 Parameter

これらの設定を利用して実際に分析を行った結果を以下で説明する。

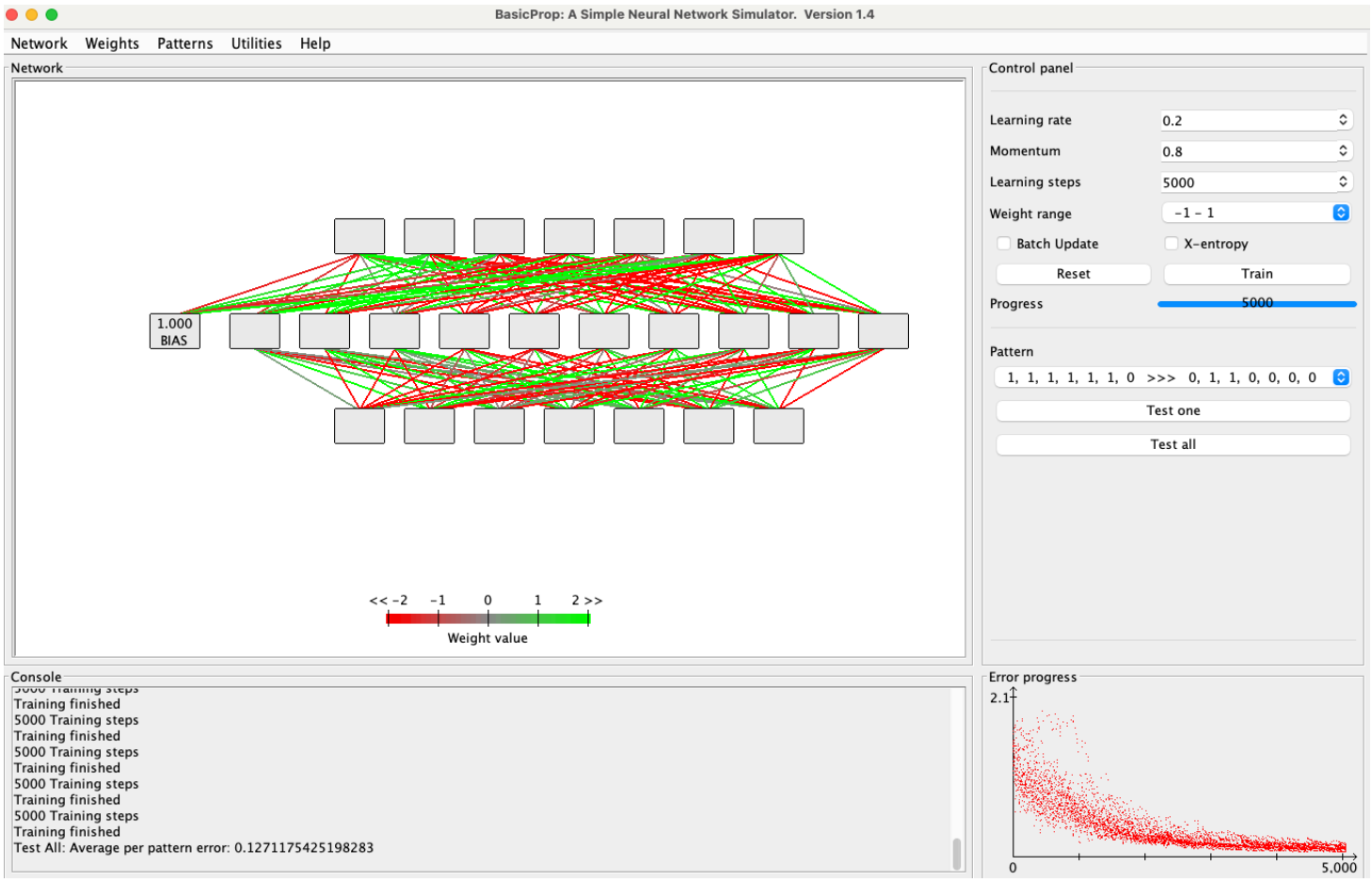

FIGURE 4 5000

5000回トレーニングを行った時点で収束が始まっていることがError progressのグラフから確認できる。また、この時点でのエラー率は0.127...であり、もう少し学習が必要だということがわかる。

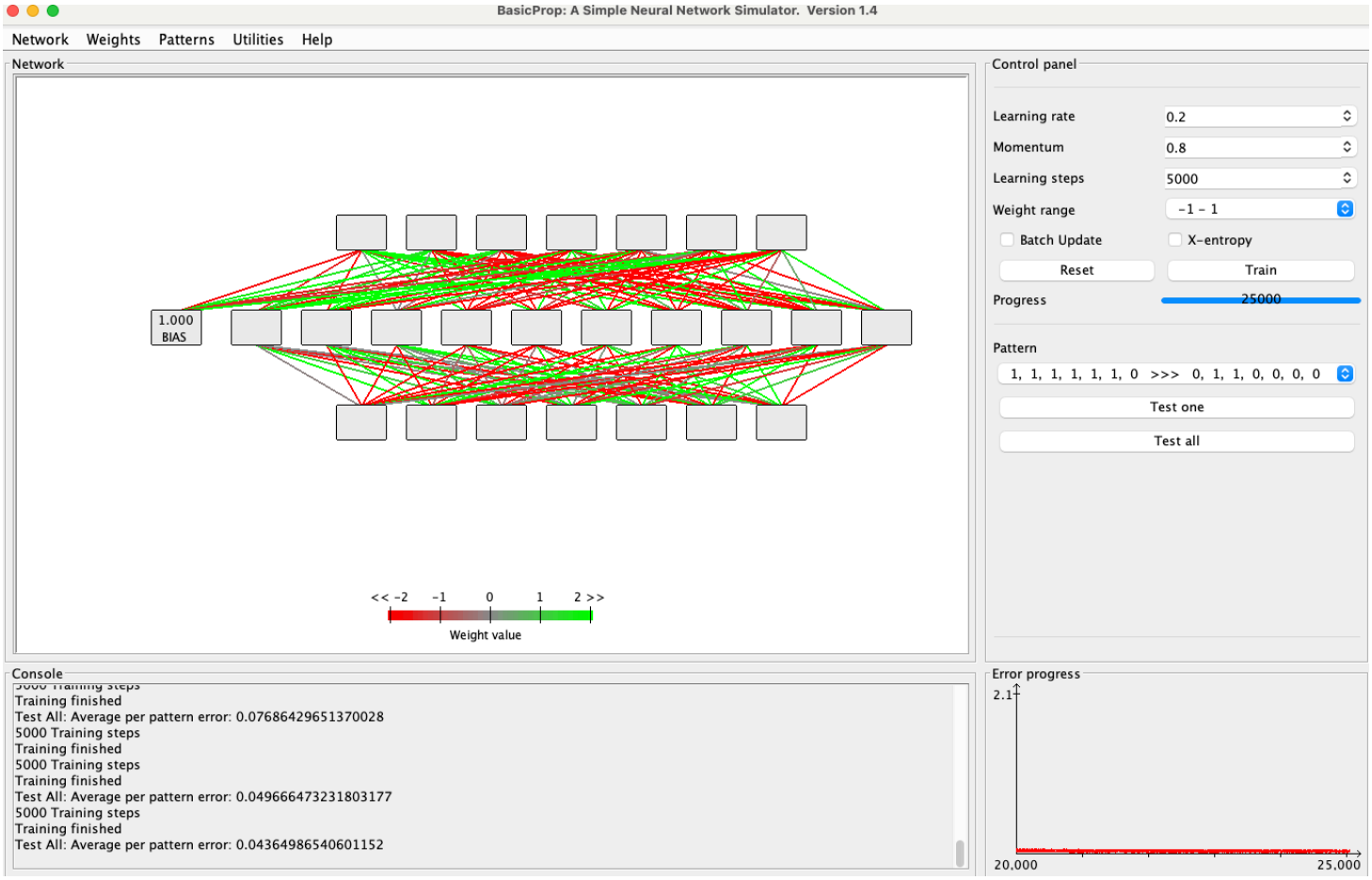

FIGURE 5 25000

5000回ずつトレーニングを行い、合計20000回トレーニングを行った時点でエラー率が0.05を切り、25000回トレーニングを行った時点でエラー率が0.04364と良い値を得ることができた。

また、このトレーニング結果のweightファイルを(25000)として提出する。

加えて、このモデルに対するパターンごとの平均エラーの値を以下に記載する:

Pattern: " 1, 1, 1, 1, 1, 1, 0 >>> 0, 1, 1, 0, 0, 0, 0 "

Result: " 0.02, 0.98, 0.98, 0.04, 0, 0.03, 0.02 "

Pattern: " 0, 1, 1, 0, 0, 0, 0 >>> 0, 1, 1, 0, 0, 0, 1 "

Result: " 0.01, 0.99, 0.99, 0.01, 0.02, 0.01, 0.97 "

Pattern: " 1, 1, 0, 1, 1, 0, 1 >>> 0, 1, 1, 0, 0, 1, 0 "

Result: " 0.01, 0.99, 0.99, 0.02, 0.01, 0.99, 0.01 "

Pattern: " 1, 1, 1, 1, 0, 0, 1 >>> 0, 1, 1, 0, 0, 1, 1 "

Result: " 0, 1, 1, 0.02, 0.03, 0.97, 0.98 "

Pattern: " 0, 1, 1, 0, 0, 1, 1 >>> 0, 1, 1, 0, 1, 0, 0 "

Result: " 0.02, 0.98, 0.98, 0.01, 0.97, 0, 0.03 "

Pattern: " 1, 0, 1, 1, 0, 1, 1 >>> 0, 1, 1, 0, 1, 0, 1 "

Result: " 0.01, 1, 1, 0.02, 0.99, 0.01, 0.98 "

Pattern: " 1, 0, 1, 1, 1, 1, 1 >>> 0, 1, 1, 0, 1, 1, 0 "

Result: " 0.02, 0.98, 0.98, 0, 0.99, 0.99, 0 "
Pattern: " 1, 1, 1, 0, 0, 0, 0 >>> 0, 1, 1, 0, 1, 1, 1 "
Result: " 0.01, 0.99, 0.99, 0, 0.96, 0.99, 1 "
Pattern: " 1, 1, 1, 1, 1, 1, 1 >>> 0, 1, 1, 1, 0, 0, 0 "
Result: " 0, 1, 1, 0.96, 0, 0, 0.03 "
Pattern: " 1, 1, 1, 1, 0, 1, 1 >>> 0, 1, 1, 1, 0, 0, 1 "
Result: " 0, 1, 1, 0.97, 0, 0.01, 0.97 "
Pattern: " 1, 1, 1, 0, 1, 1, 1 >>> 1, 0, 0, 0, 0, 0, 1 "
Result: " 0.98, 0.02, 0.02, 0.01, 0.03, 0.02, 0.97 "
Pattern: " 0, 0, 1, 1, 1, 1, 1 >>> 1, 0, 0, 0, 0, 1, 0 "
Result: " 0.99, 0.01, 0.01, 0, 0.02, 0.97, 0 "
Pattern: " 1, 0, 0, 1, 1, 1, 0 >>> 1, 0, 0, 0, 0, 1, 1 "
Result: " 1, 0, 0, 0, 0, 0.98, 0.98 "
Pattern: " 0, 1, 1, 1, 1, 0, 1 >>> 1, 0, 0, 0, 1, 0, 0 "
Result: " 0.98, 0.02, 0.02, 0, 0.98, 0.02, 0 "
Pattern: " 1, 0, 0, 0, 1, 1, 1 >>> 1, 0, 0, 0, 1, 1, 0 "
Result: " 1, 0, 0, 0, 0.99, 0.99, 0.03 "
Pattern: " 1, 0, 0, 0, 1, 1, 1 >>> 1, 0, 0, 0, 1, 1, 0 "
Result: " 1, 0, 0, 0, 0.99, 0.99, 0.03 "
Pattern: " 1, 0, 0, 1, 0, 0, 0 >>> 1, 0, 0, 1, 0, 0, 0 "
Result: " 0.99, 0.01, 0.01, 0.98, 0, 0.01, 0.03 "

また、このモデルと同じ25000回というトレーニング回数でより低いエラー率を得ることができた設定について、以下にスクリーンショットを記載する。

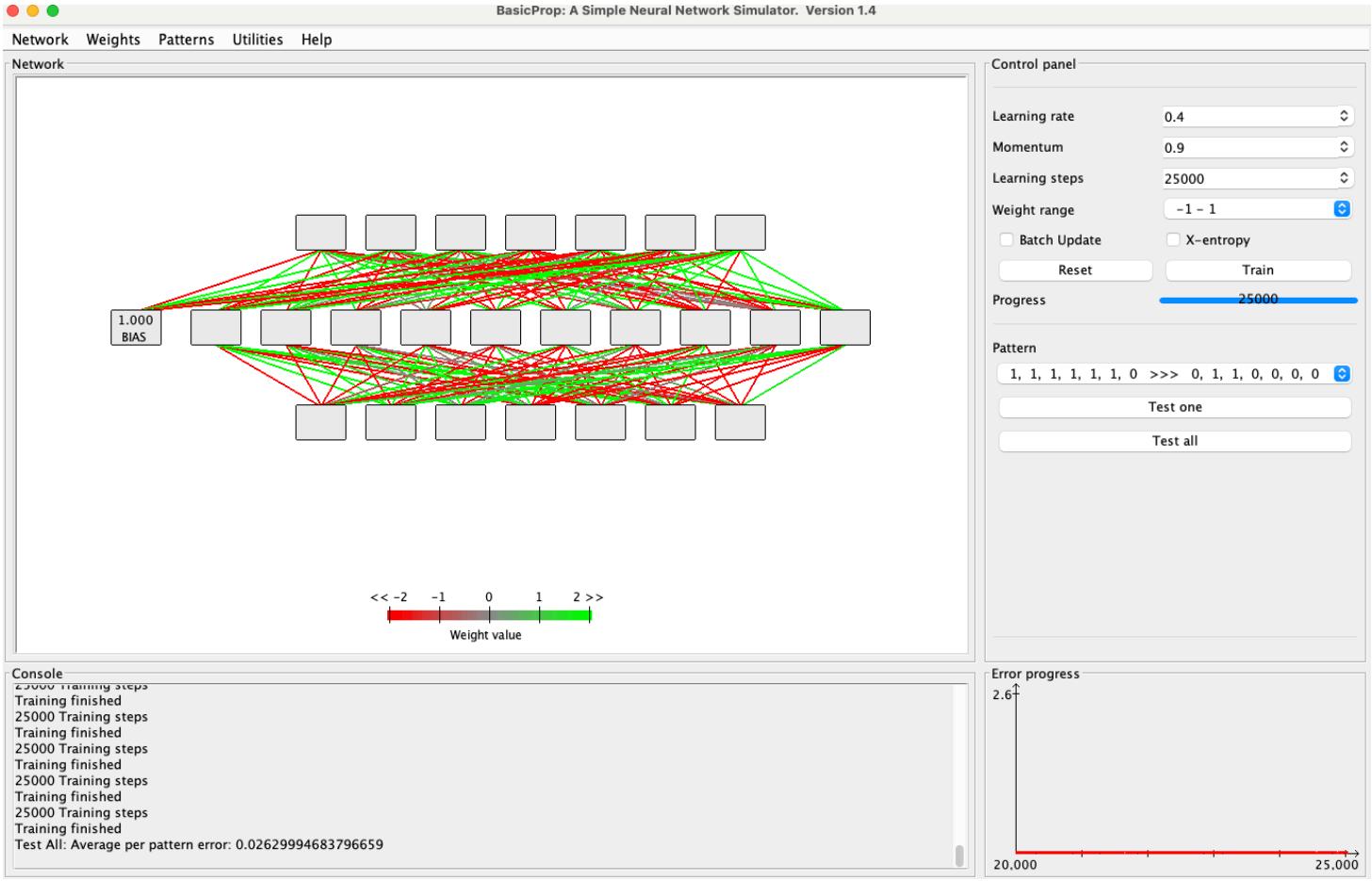

FIGURE 6 Optional case

このケースでは、より良い収束率を得るために以下の設定を行った:

Laerning rate: 0.4

Momentum: 0.9

その結果平均エラー率として0.026299を得ることができた。