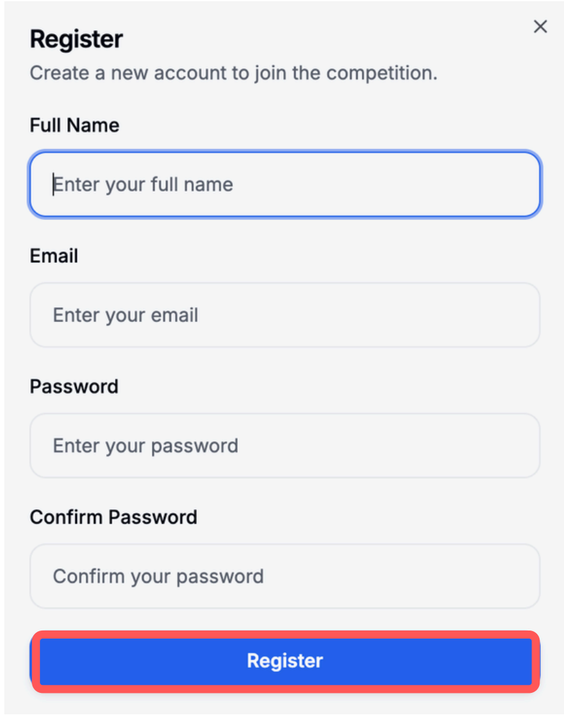
TUTORIAL FOR QUAIC

Quantum AI Competition

1.アカウント作成
2.AIと対戦
3.モデルの作成
4.モデルの学習
5.モデルの提出

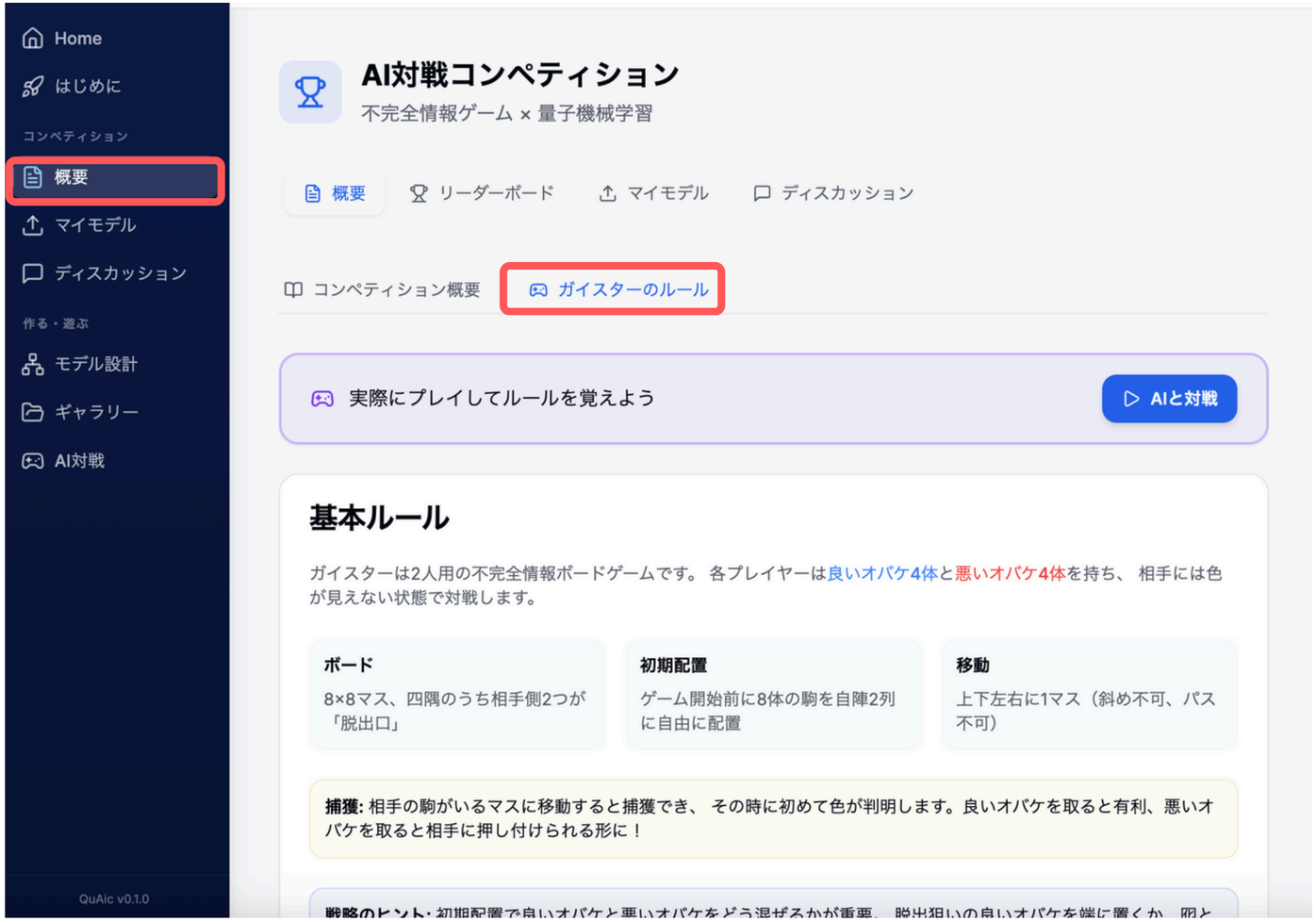
アカウント作成

右上の [Register] を押す



必要事項を入力して [Register] を押す

ガイスターのルールの確認



[概要] →[ガイスターのルール]
よりガイスターのルールを
確認することができる

AIと対戦：モデル選択

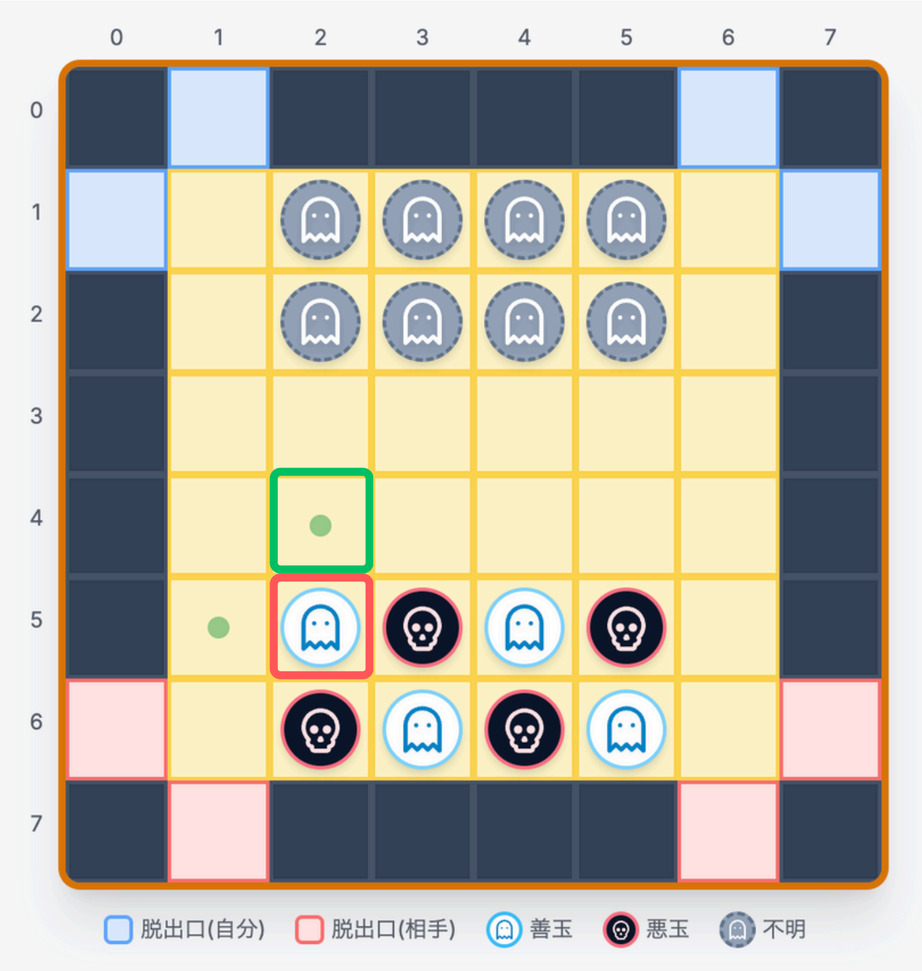
1 サイドバーより [AI対戦] を選択
2 色推定モデルと、
　対戦相手のAIを選択
3 [対戦開始] ボタンを押す

AIと対戦：駒配置

善玉を置きたい場所をクリック

配置が完了した様子

AIと対戦：対戦画面



1 動かしたい駒をクリック
2 駒の可能な移動先が緑の点で表示されるので、動かしたい先をクリック
3 AIが駒を動かす

どちらかの勝利条件が満たされるまで、1~3を繰り返す

モデル作成：HNN Composerの起動

[モデル設計] → [HNN Composer] を選択

モデル作成：HNN Composerでのモデル構築

HNN Composerが表示されるので、
そこでモデルを構築
※HNN Composerの詳細に関しては
　p12以降を参照

モデル構築が終わったら
[Train in Google Colab] を選択

モデルの学習

config.jsonのダウンロードが開始

Colaboratory の画面が開けたら
[すべてのセルを実行] をクリック

モデルの学習

```
[3]
for filename in uploaded.keys():
    if filename.endswith('.json'):
        config_filename = filename
        break

(variable) config_filename: Any | None

if config_filename:
    # 設定を解析して表示
    with open(config_filename, 'r') as f:
        config = json.load(f)

    print(f"\n設定を読み込みました: {config_filename}")

    # Qugeisterのローダーを使用
    from qugeister.models.hnn_config_loader import load_hnn_config
    hnn_config = load_hnn_config(config_filename)

    print(f"\n=== HNN設定 ===")
    print(f"  量子ビット数: {hnn_config.quantum.n_qubits}")
    print(f"  量子レイヤー数: {hnn_config.quantum.n_layers}")
    print(f"  埋め込み: {hnn_config.quantum.embedding}")
    print(f"  Ansatz: {hnn_config.quantum.ansatz}")
else:
    print("ERROR: JSONファイルがアップロードされていません。config.jsonをアップロードしてください")

config.jsonファイルをアップロードしてください:
ファイル選択  選択されていません  Cancel upload
```

ファイルがアップロードできれば、
11セル目で学習が開始
→学習が終わると .pthファイルが
ダウンロードされる

3セル目で [ファイル選択] の
ボタンが出現
→ダウンロードされたconfig.json
ファイルをアップロード

```
[11]
            torch.save(model.state_dict(), 'best_model.pth')
        else:
            patience_counter += 1

        if (epoch + 1) % 10 == 0:
            print(f"Epoch {epoch+1}/{EPOCHS}")
            print(f"  Train Loss: {train_loss:.4f}, Acc: {train_acc:.4f}")
            print(f"  Val Loss: {val_loss:.4f}, Acc: {val_acc:.4f}")

        # Early Stopping
        if patience_counter >= PATIENCE:
            print(f"\nEarly stopping at epoch {epoch+1}")
            break

print(f"\n学習完了! ベスト検証精度: {best_val_acc:.4f}")

学習を開始します...
注意: 量子層のbackprop微分はCPUで実行されます

Training:  75%  15/20 [08:52<02:56, 35.39s/it]
Epoch 10/20
  Train Loss: 0.5971, Acc: 0.6646
  Val Loss: 0.5982, Acc: 0.6623
```

モデル提出

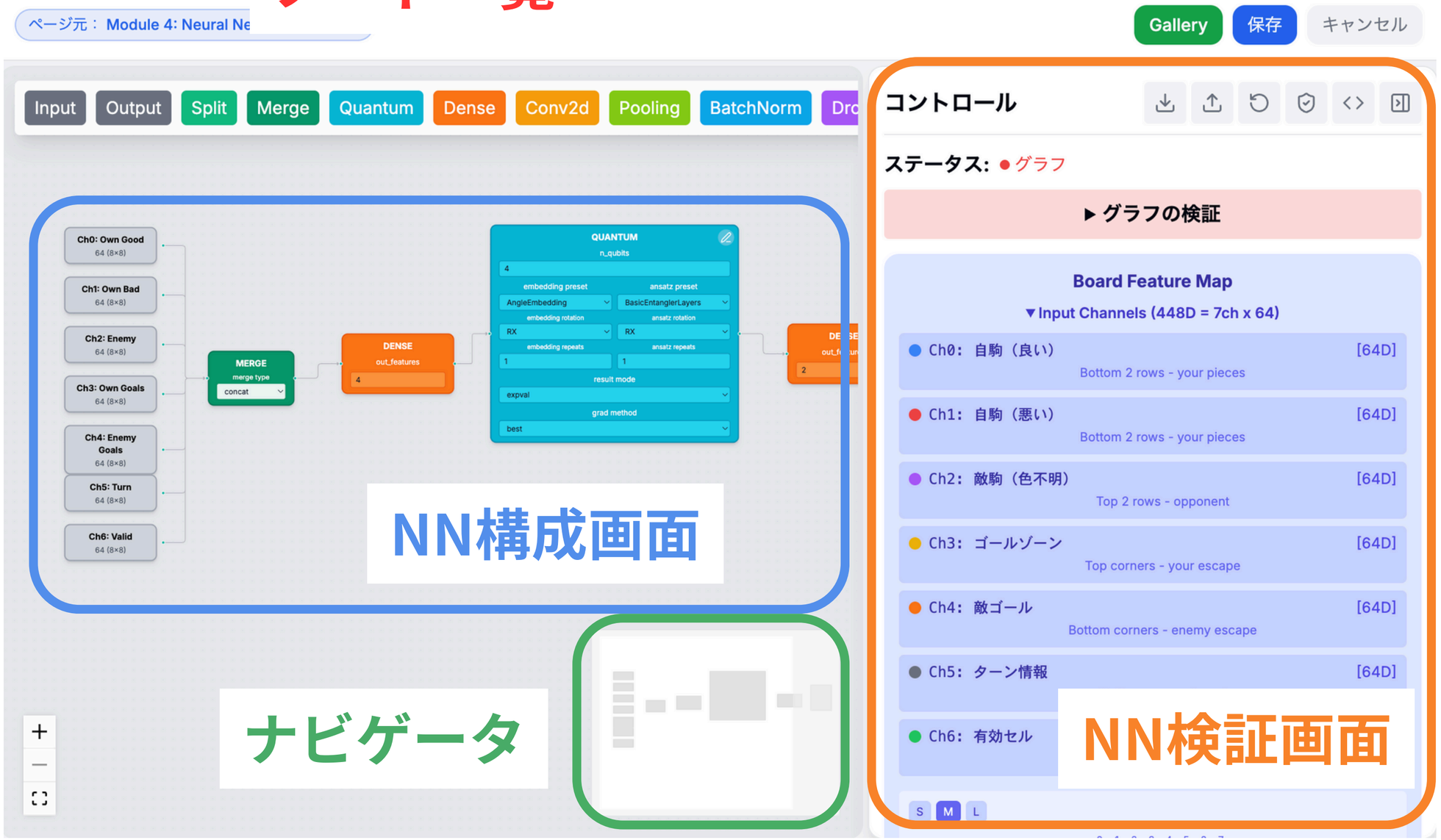
[マイモデル] → [モデルを提出]
よりモデルをアップロード

提出履歴を見て、提出したモデルが
[検証済み] となっていればOK

補助資料

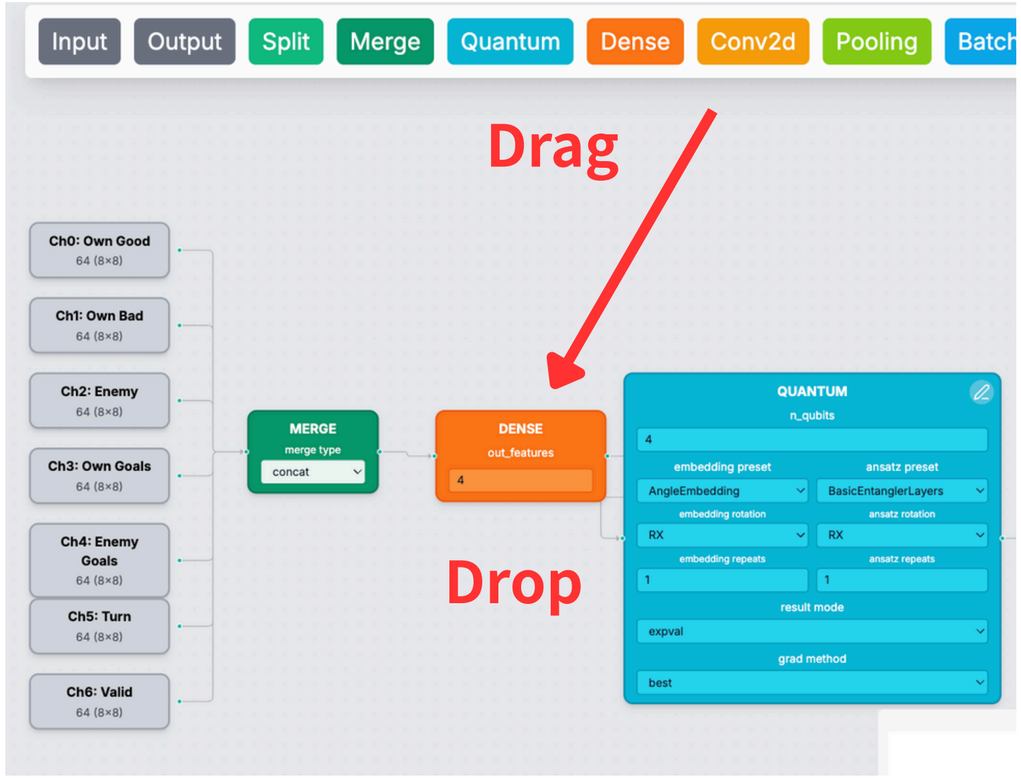
HNN COMPOSER
取扱説明書

ノード一覧

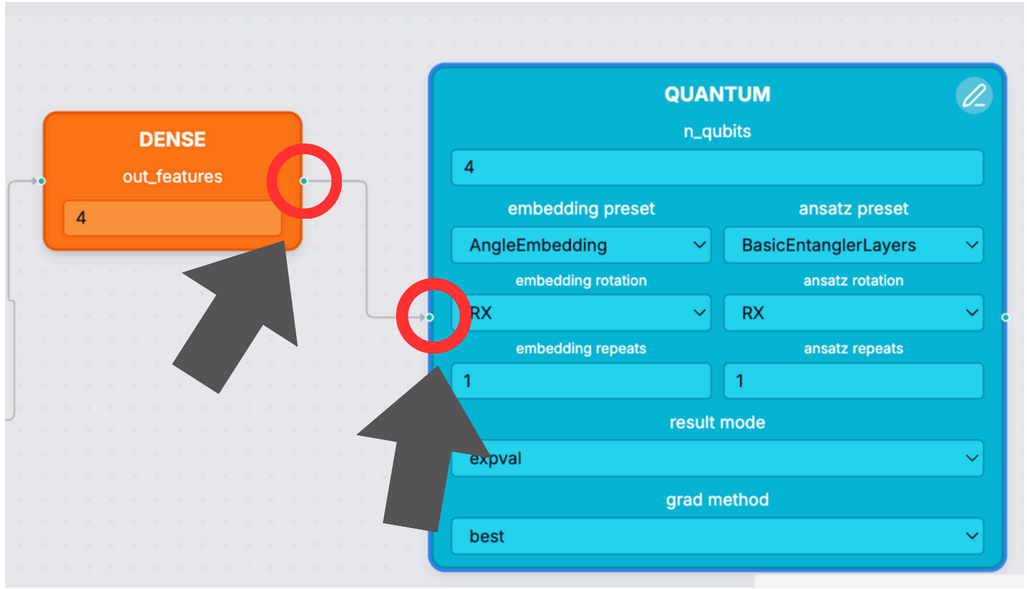

ページ元： Module 4: Neural Ne
Gallery
保存
キャンセル
Input
Output
Split
Merge
Quantum
Dense
Conv2d
Pooling
BatchNorm
Ch0: Own Good
64 (8×8)
Ch1: Own Bad
64 (8×8)
Ch2: Enemy
64 (8×8)
Ch3: Own Goals
64 (8×8)
Ch4: Enemy Goals
64 (8×8)
Ch5: Turn
64 (8×8)
Ch6: Valid
64 (8×8)
MERGE
merge type
concat
DENSE
out_features
4
QUANTUM
n_qubits
4
embedding preset
AngleEmbedding
ansatz preset
BasicEntanglerLayers
embedding rotation
RX
ansatz rotation
RX
embedding repeats
1
ansatz repeats
1
result mode
expval
grad method
best
NN構成画面
ナビゲータ
コントロール
ステータス: ● グラフ
▶ グラフの検証
Board Feature Map
▼ Input Channels (448D = 7ch x 64)
Ch0: 自駒（良い） [64D]
Bottom 2 rows - your pieces
Ch1: 自駒（悪い） [64D]
Bottom 2 rows - your pieces
Ch2: 敵駒（色不明） [64D]
Top 2 rows - opponent
Ch3: ゴールゾーン [64D]
Top corners - your escape
Ch4: 敵ゴール [64D]
Bottom corners - enemy escape
Ch5: ターン情報 [64D]
Ch6: 有効セル
S
M
L
NN検証画面

NN構成画面の操作方法

- 画面上部にあるノード一覧からドラッグアンドドロップしてノードを配置
- ノードをクリックしてからデリートキーを押すと削除できる
- ノードの横にある緑の丸同士をドラッグして繋ぐことで、ノードを結合

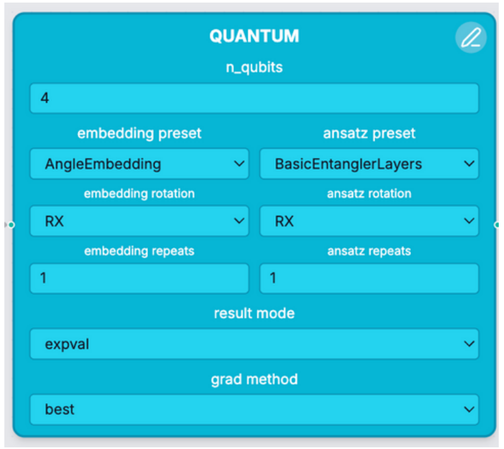


結合を切りたいときは、繋がっている矢印をクリックで選択して、デリートボタンを押す

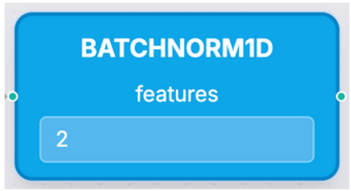


ノードの種類

SPLIT
データを分割する層

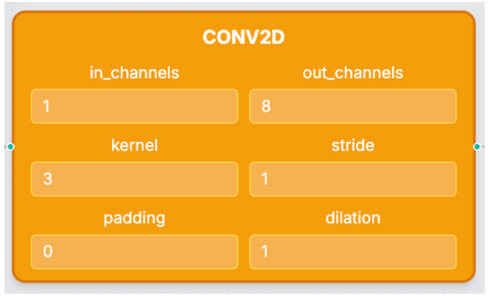


QUANTUM
量子層

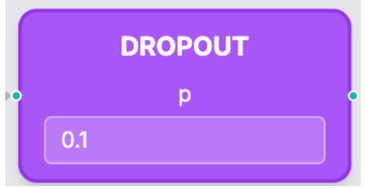


BATCHNORM1D
バッチ正規化層

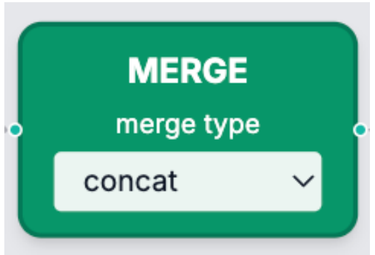
DENSE
全結合層



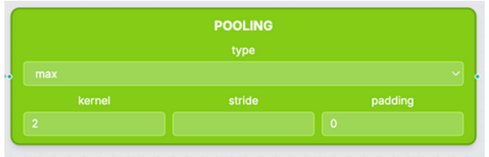
CONV2D
畳み込み層



DROPOUT
ドロップアウト層



MERGE
多数の出力を一出力に結合する層

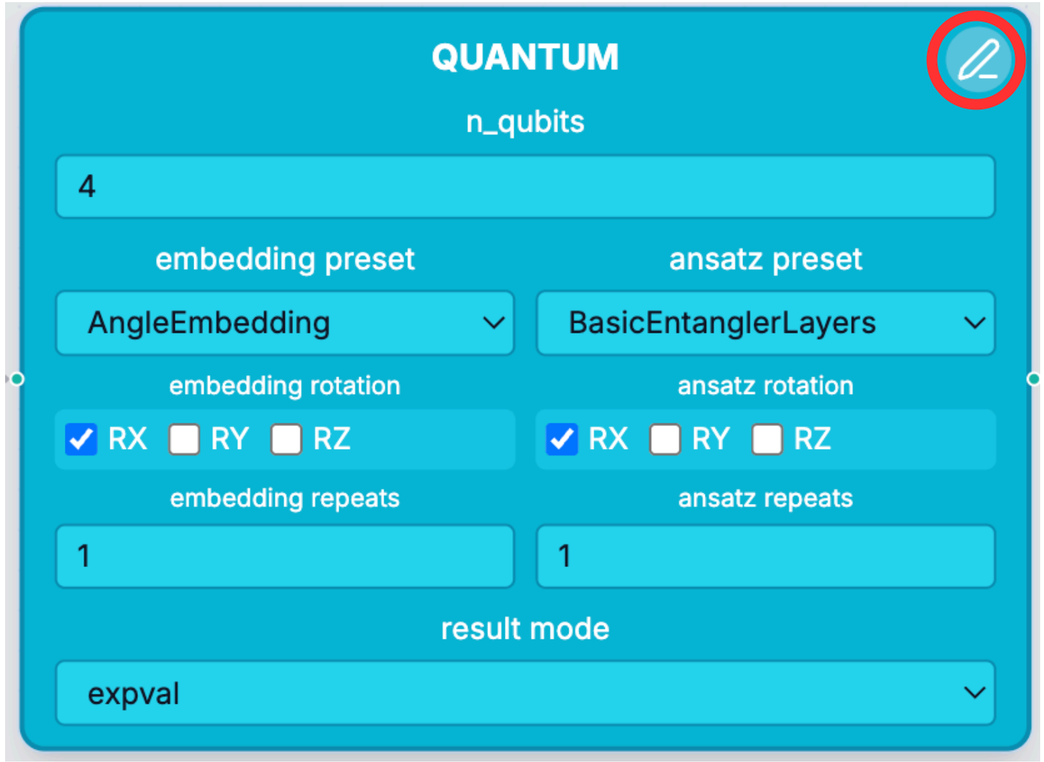


POOLING
プーリング層

QUANTUMノードの詳細

量子回路の詳細設計

n_qubits: qubit数



embedding preset
どのように情報を量子状態に埋め込むか

embedding rotation
回転角を何のゲートで実装するか

embedding repeats
埋め込みの層を何層にするか

ansatz preset
学習を行う回路の構造

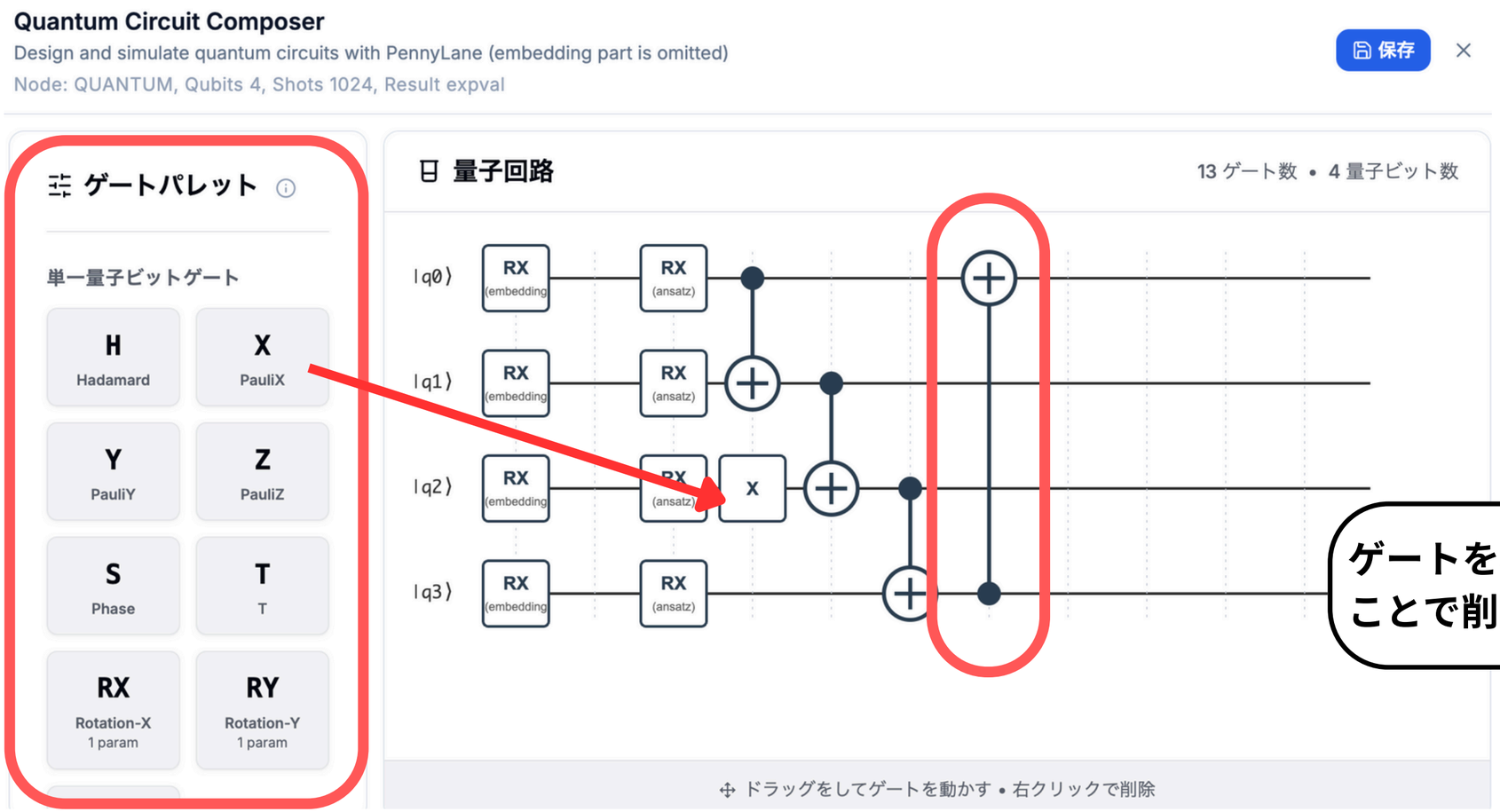
ansatz rotation
学習回路に用いる回転角

ansatz repeats
学習回路の層を何層にするか

result mode
結果が確率分布か期待値か

回路設計画面

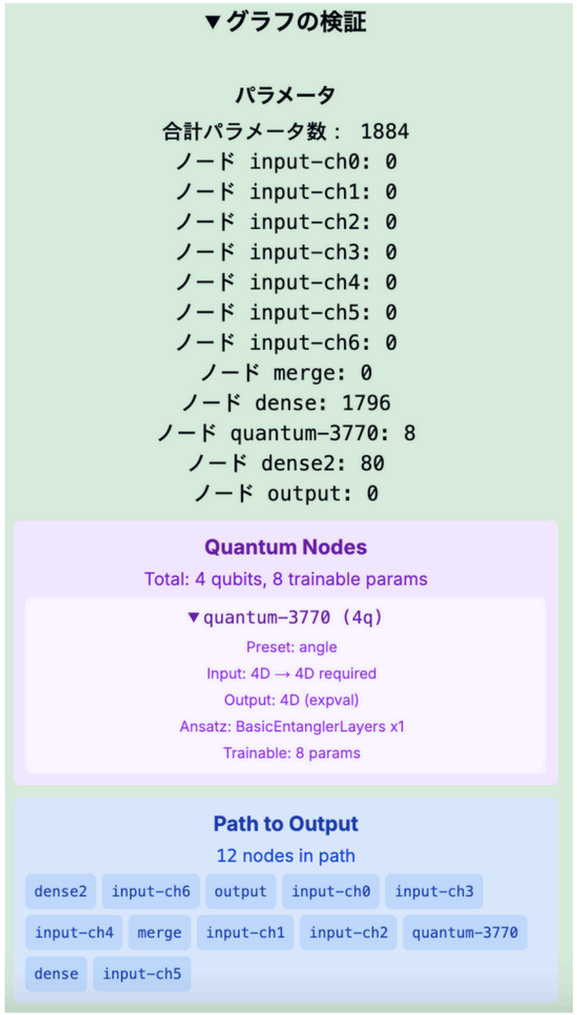
二量子ビットゲートは
ドラッグで
コントロール・
ターゲットビットを移動



ドラッグアンド
ドロップで
ゲートを配置

ゲートを右クリックする
ことで削除可能

NN検証画面

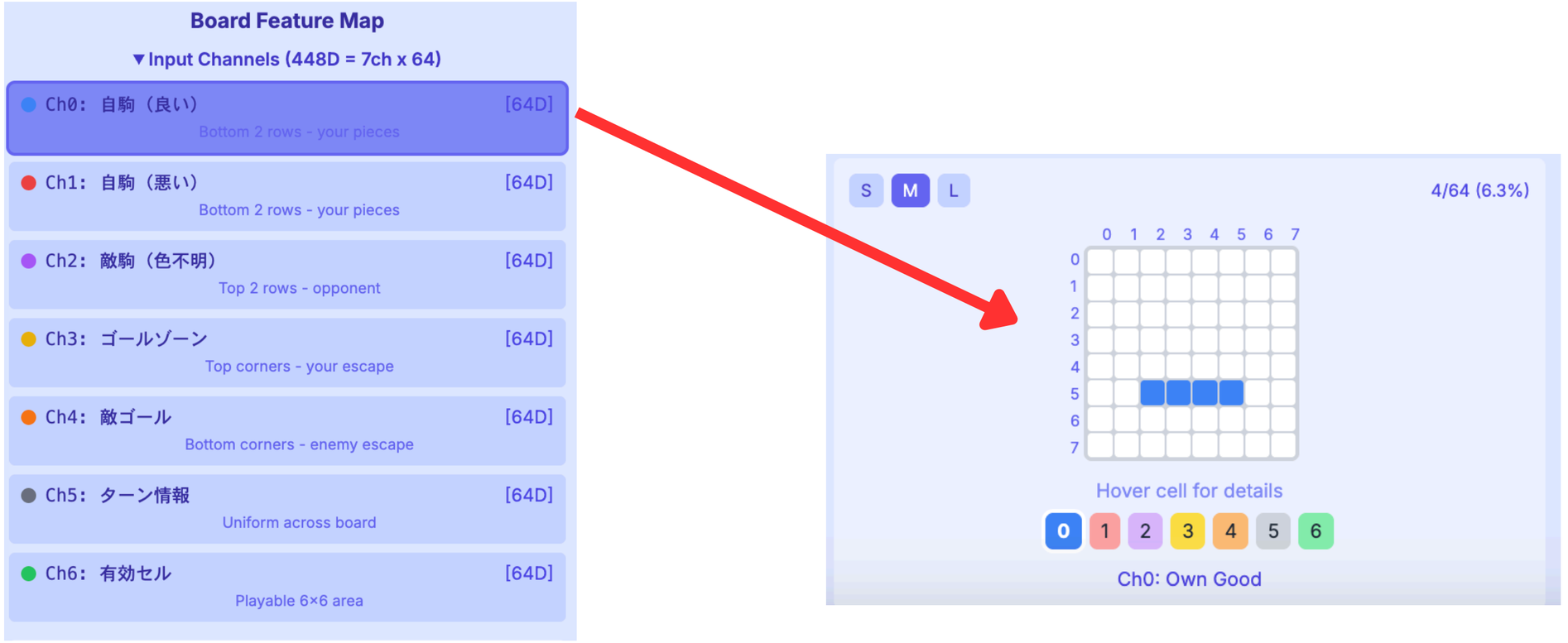


モデルグラフから層ごとの学習パラメータ数を算出し、合計と内訳を表示

0はパラメータを持たない処理ノード（入力・結合など）

層の追加ミス・次元の取り違えといった誤りを早期に発見可能

Board Feature Map



各チャンネルの情報が盤面のどこと対応しているかを図示