Tile representation:

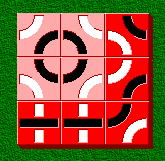
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 3-bit decode for tile | 9-bit decode for white path | 9-bit decode for black path |  | 3-bit decode for tile | 9-bit decode for white path | 9-bit decode for black path |
|  | 001 | 010  110  000 | 000  011  010 |  | 110 | 000  011  010 | 010  110  000 |
|  | 011 | 010  011  000 | 000  110  010 |  | 100 | 000  110  010 | 010  011  000 |
|  | 010 | 010  010  010 | 000  111  000 |  | 101 | 000  111  000 | 010  010  010 |
|  | 000 | 000  000  000 | 000  000  000 |  |  |  |  |

\*3-bit decode method is our old method. It is a quite compact way of representation. It is now used in Jun Bing’s board rule program.

\*9-bit decode method is the new method for extracting the white path pattern and black path pattern respectively in order to generate the white feature map and black feature map. The feature map will be the input of value network (CNN).

=====================================================================================

Training sample 1

 Golden output: 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000 | 000 | 000 | 000 | 000 |
| 000 | 001 | 011 | 100 | 000 |
| 000 | 100 | 110 | 001 | 000 |
| 000 | 010 | 010 | 110 | 000 |
| 000 | 000 | 000 | 000 | 000 |

45 degree board edge representation for white paths (size: 2n\*2n) (seems that the pattern is not obvious)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 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 |

3x3 per tile board representation for white paths (size: (2n+1)\*(2n+1)) (For a 22\*22 board size, 45\*45)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 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 |

Note: the place of 1 can be replaced by the score of the path. The score of the path is determined by the displacement of the two ends of the path and the difference of their orientation.

Training sample 2

