# Response 1:

## Playing Notakto

## Question 1 – Playing Notakto

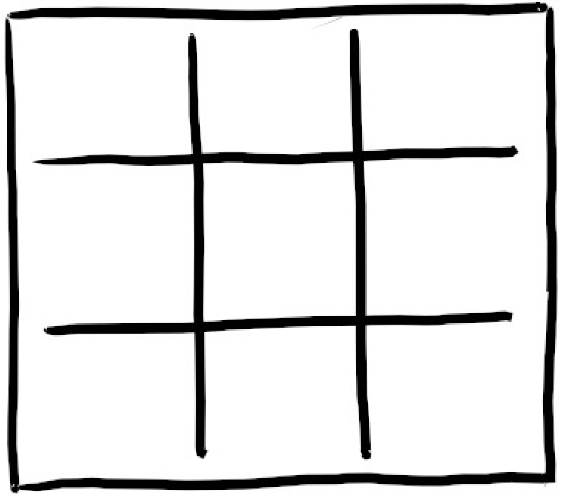
Consider the following two-player game on an $n \times n$ board, with $n \leq 6$. The two players take turns filling the board with the same symbol "X." The first player that completes a row, a column, or the main diagonal and anti-diagonal loses the game.

Expectations: For this question, the following is expected.

a) Implement a neural network architecture to play the game. To aid you in your task, you are allowed to research network architectures that have been used for other games and you might use common open-source libraries like tensor flow.
b) You should try different architectures and parameters, and eventually decide on specific choices.
c) You need to write a short (around 3 to 5 pages) report that addresses the choice of architecture, parameters, and how you asses the performance of your network.
d) The report should also include observations you find interesting during the training process. In particular, if the training process provides any insight that can be translated into strategies to play the game.

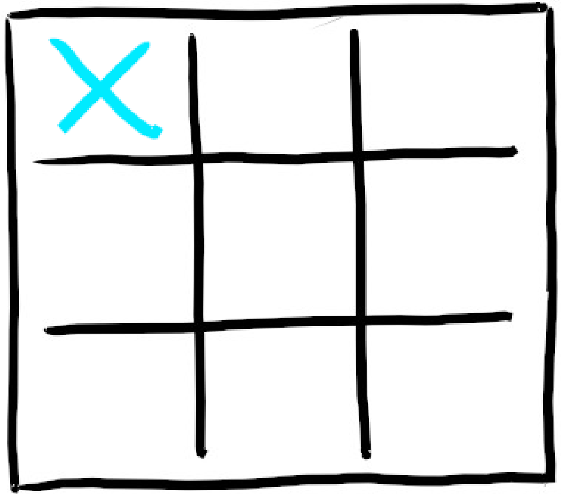
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



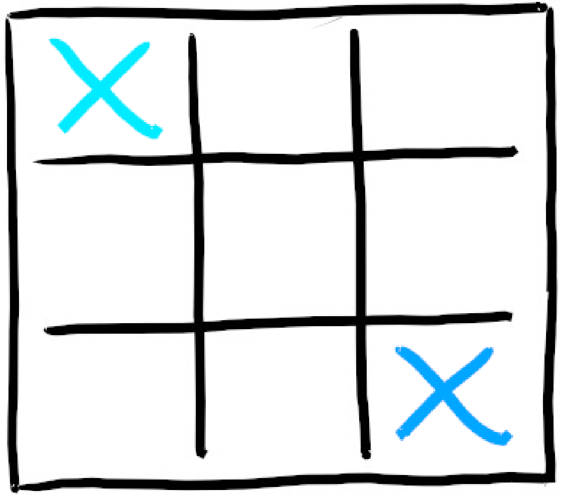
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



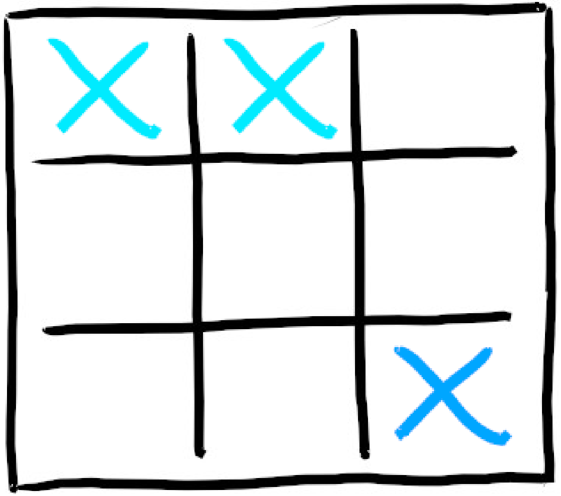
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



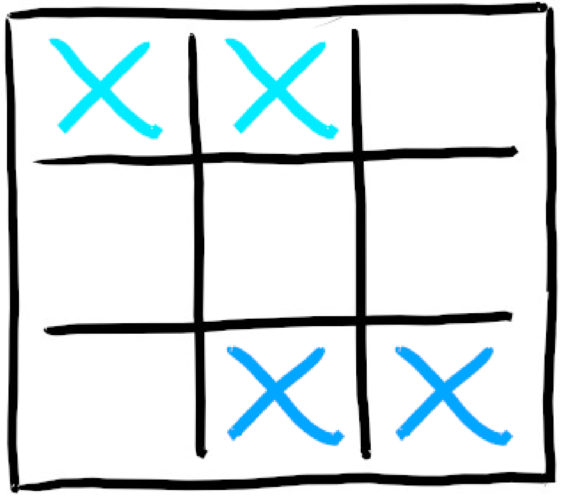
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



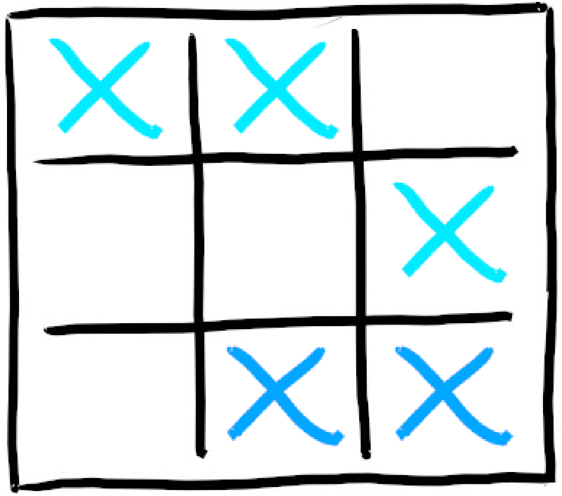
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



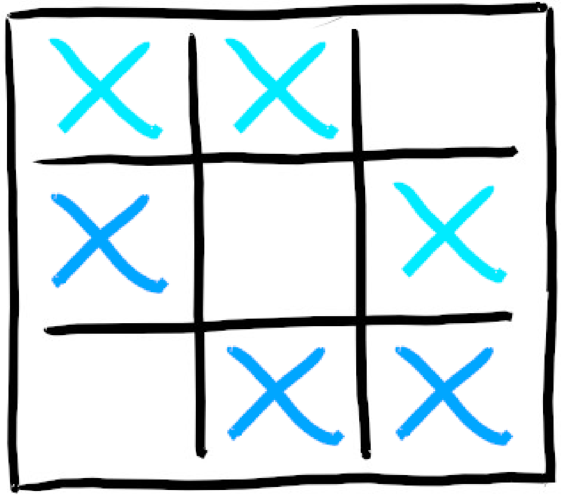
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



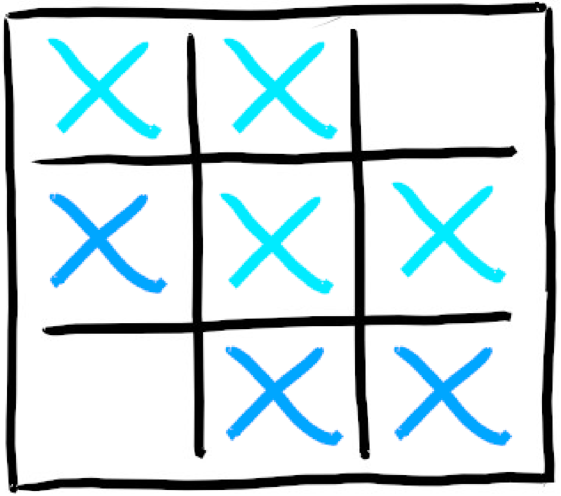
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



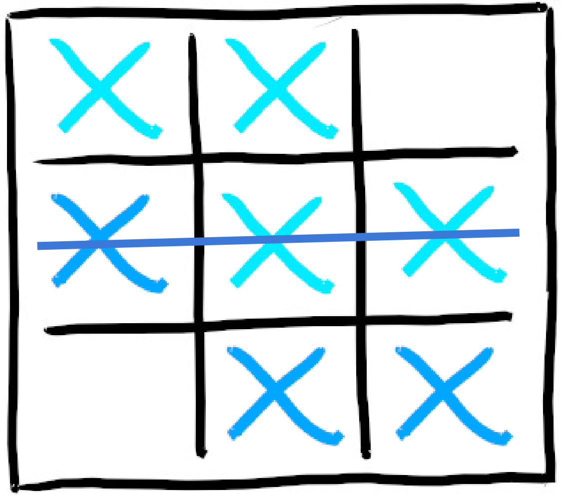
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



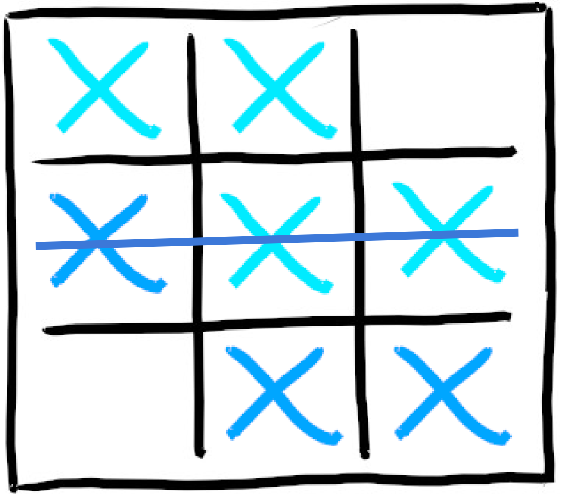
# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



# The Game of Notakto

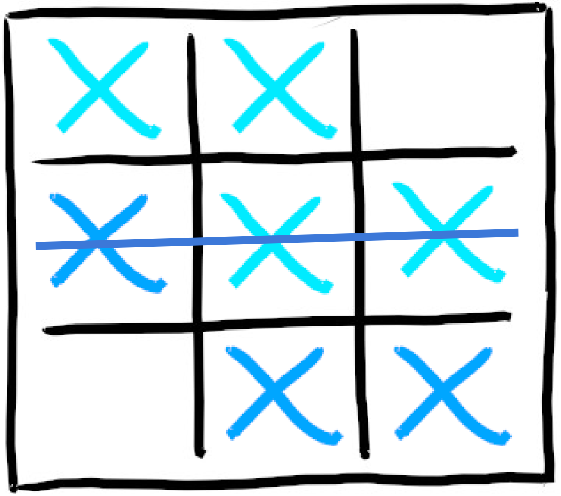
(Chow 2010)
(Plambeck and Whitehead 2013)

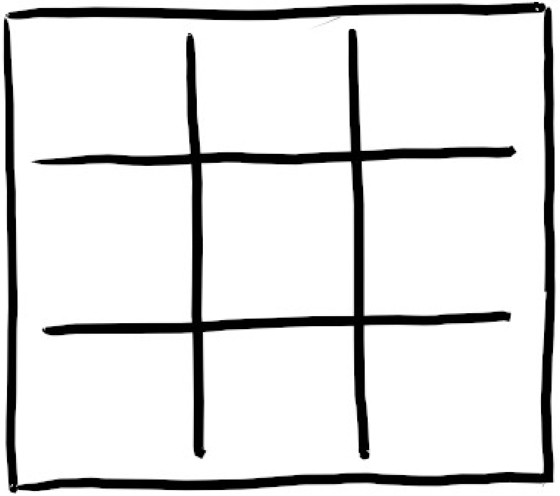


**Player 2 Wins!**

# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)

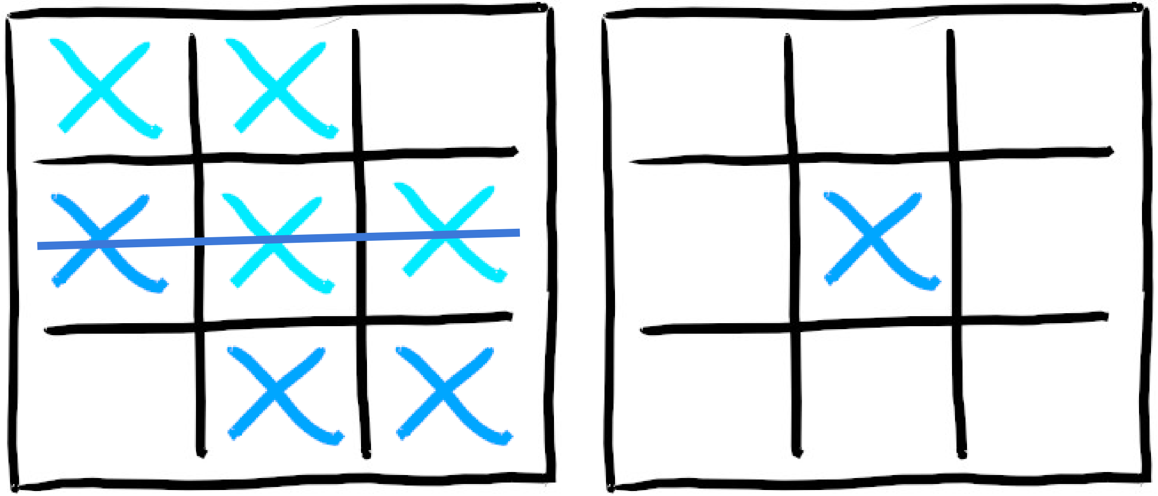




**Player 2 Wins!**

# The Game of Notakto

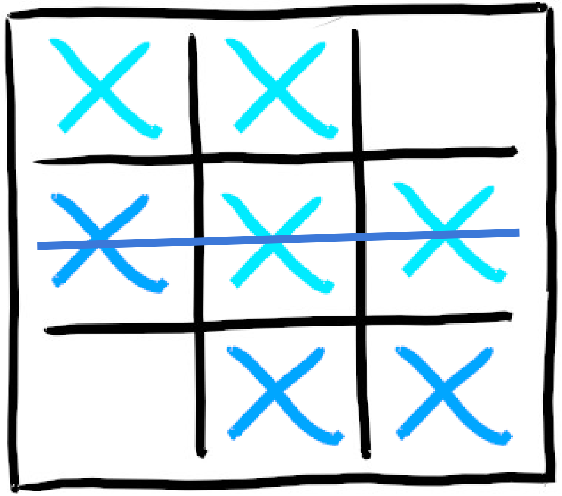
(Chow 2010)
(Plambeck and Whitehead 2013)



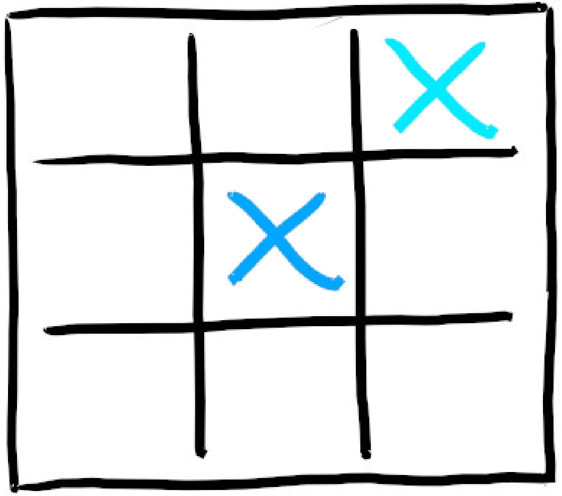
**Player 2 Wins!**

# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)

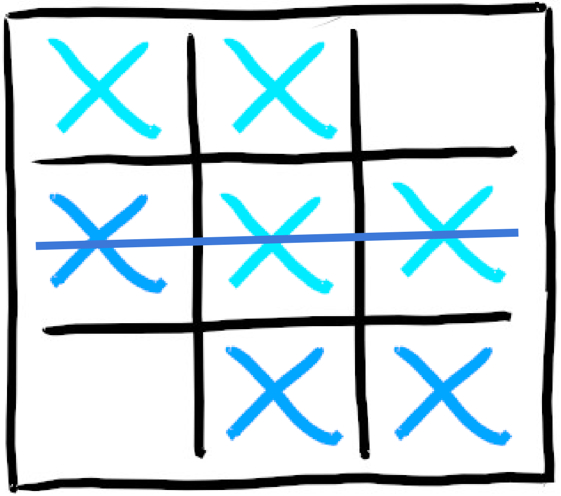


**Player 2 Wins!**

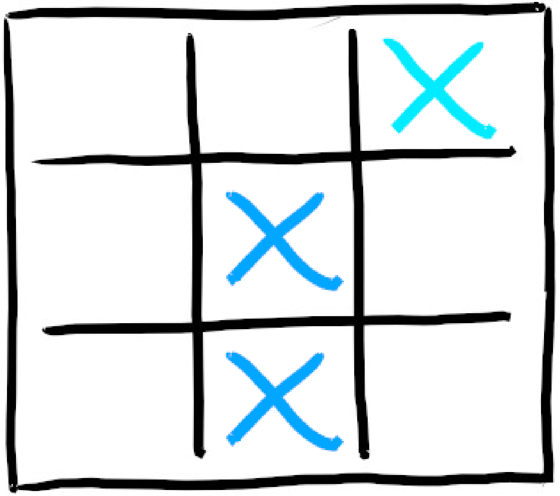


# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)

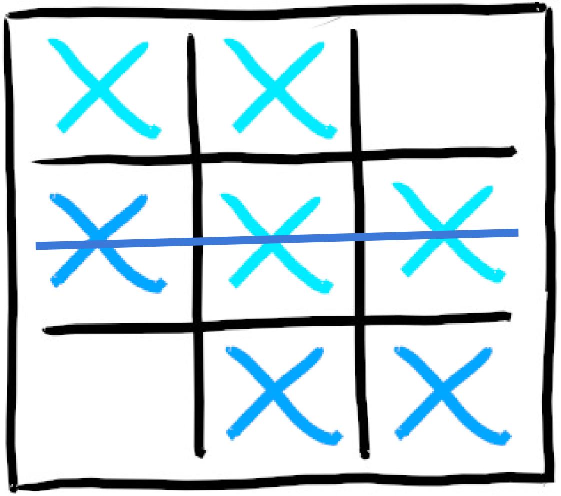


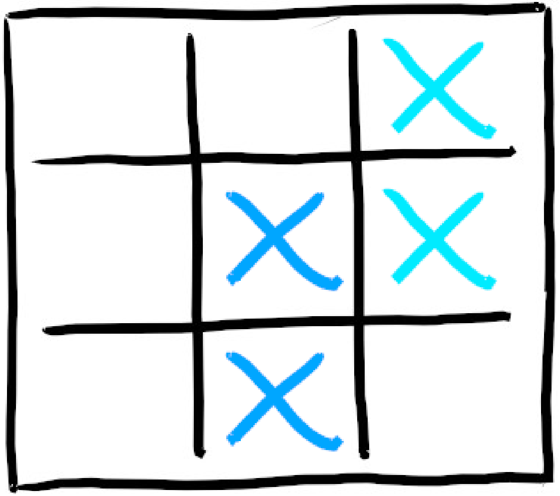
**Player 2 Wins!**



# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)

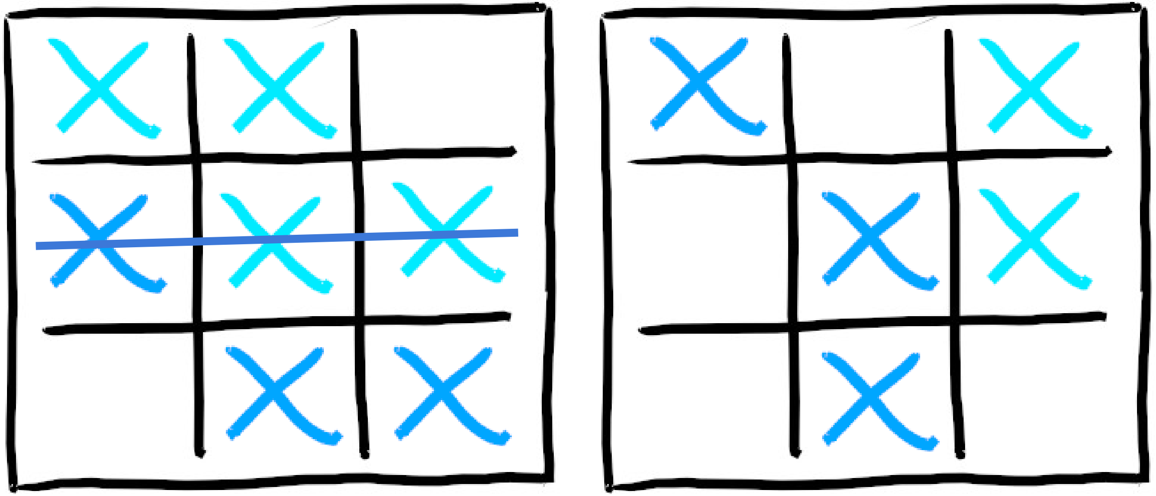




**Player 2 Wins!**

# The Game of Notakto

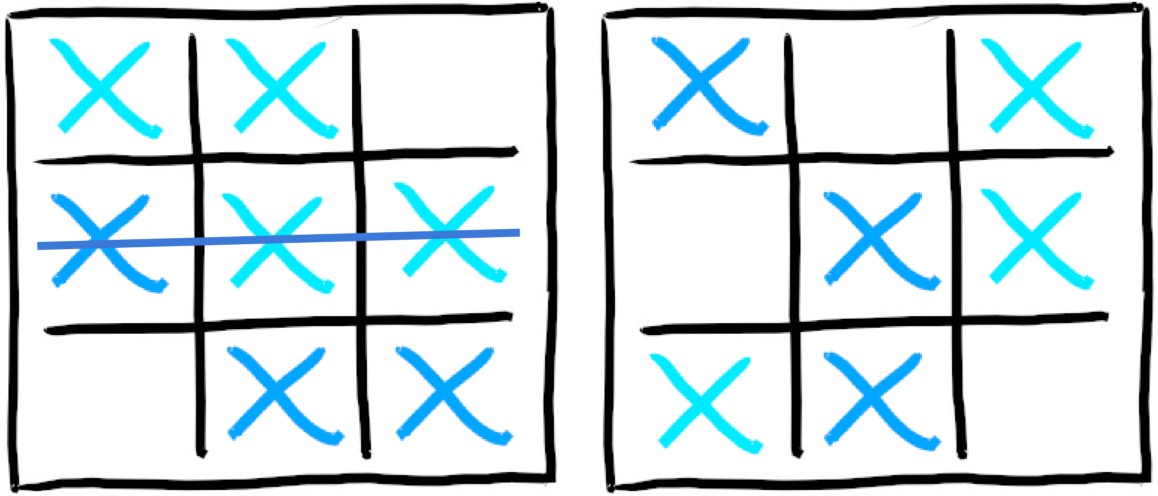
(Chow 2010)
(Plambeck and Whitehead 2013)



**Player 2 Wins!**

# The Game of Notakto

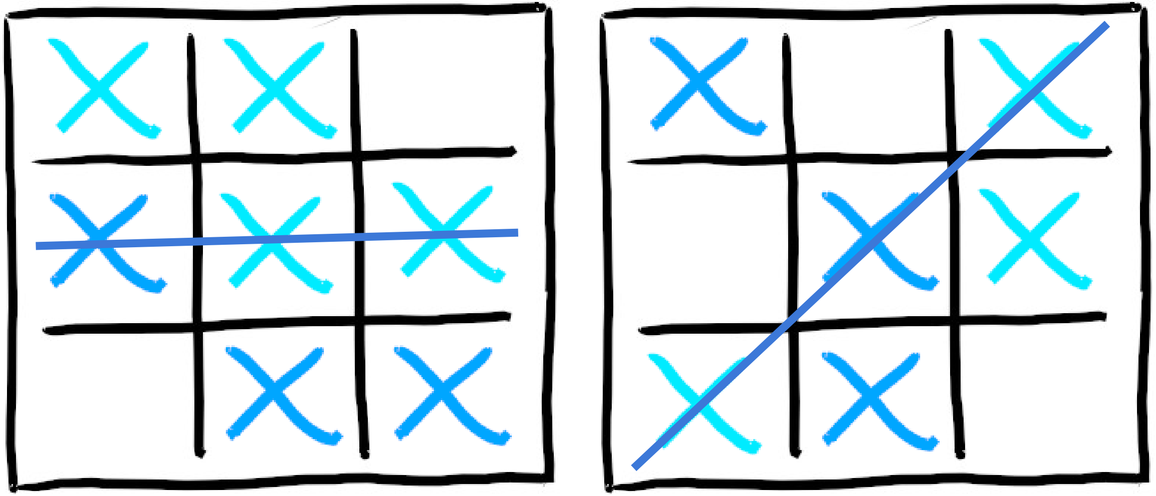
(Chow 2010)
(Plambeck and Whitehead 2013)



**Player 2 Wins!**

# The Game of Notakto

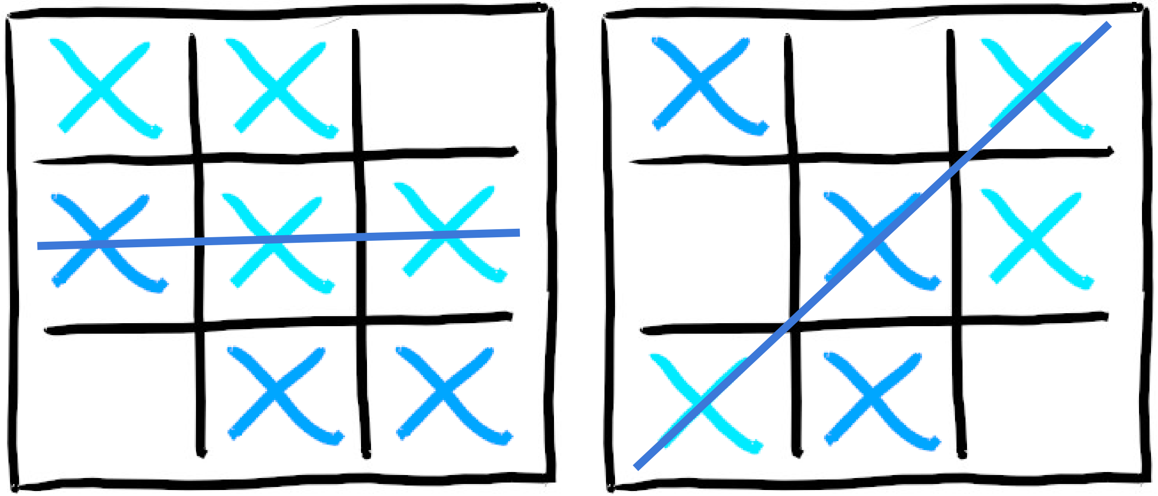
(Chow 2010)
(Plambeck and Whitehead 2013)



**Player 2 Wins!**

# The Game of Notakto

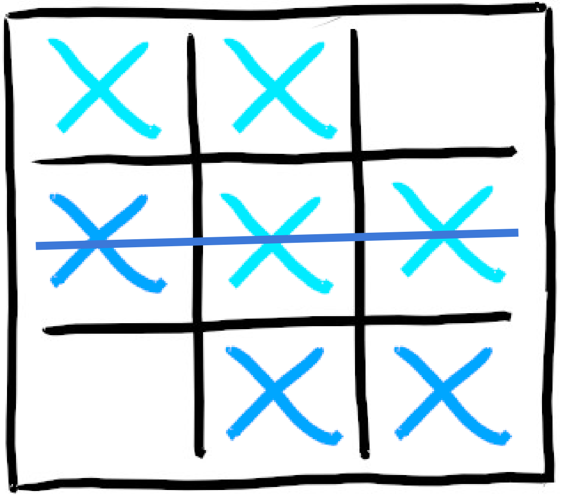
(Chow 2010)
(Plambeck and Whitehead 2013)



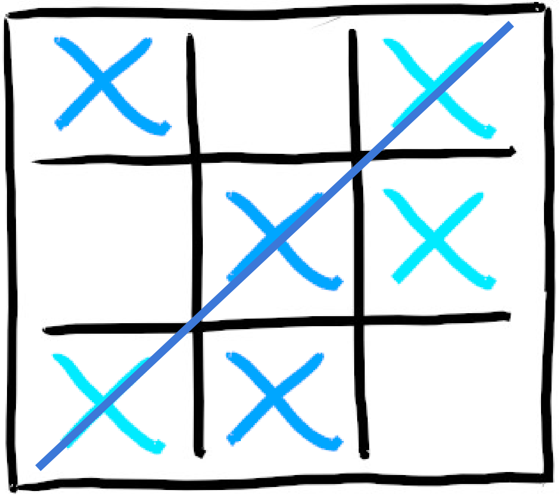
Player 2 Wins!    Player 1 Wins!

# The Game of Notakto

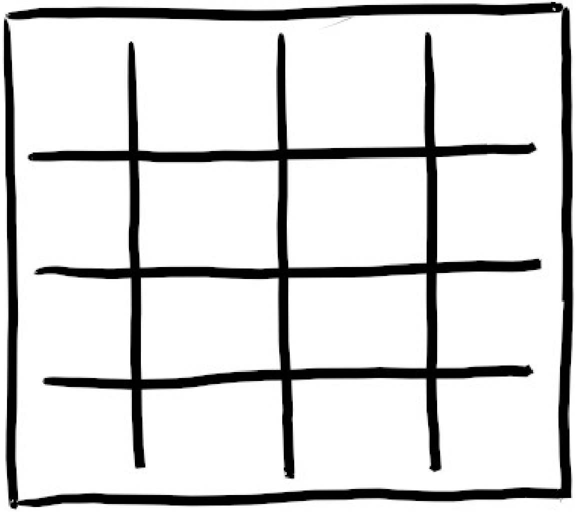
(Chow 2010)
(Plambeck and Whitehead 2013)



**Player 2 Wins!**



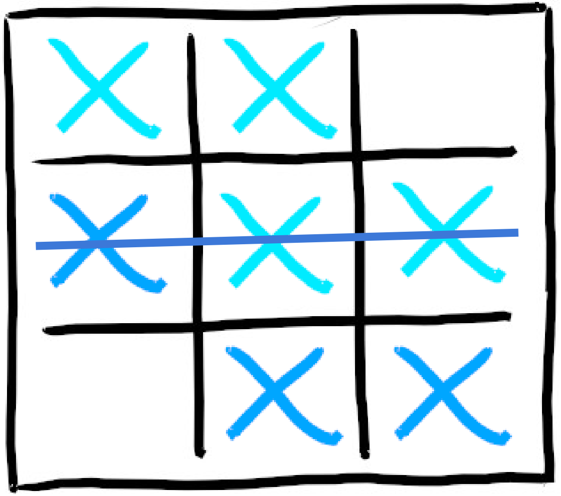
**Player 1 Wins!**



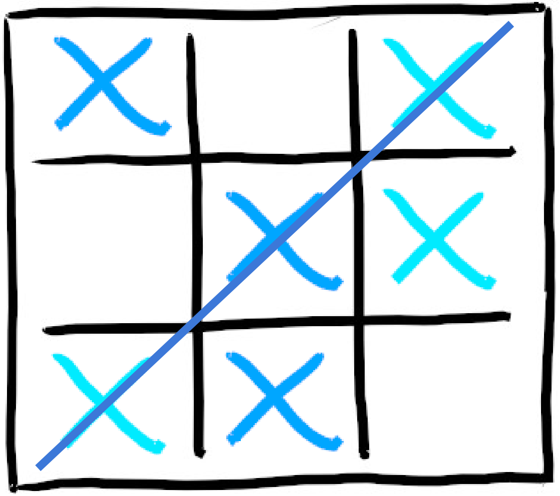
???

# The Game of Notakto

(Chow 2010)
(Plambeck and Whitehead 2013)



**Player 2 Wins!**



**Player 1 Wins!**

| | | | |
|---|---|---|---|
| a | b | c | d |
| e | f | g | h |
| b | a | d | c |
| f | e | h | g |

# How to Win at Notakto

| n | Winner under perfect play | Reference |
|---|---|---|
| 1 | 2nd Player | |
| 2 | 1st Player | |
| 3 | 1st Player | |
| 4 | 2nd Player | (Chow 2010) |
| 5 | 1st Player | (Chow 2010) |
| 6 | 1st Player | (Chen et al. 2021) |
| $n = 4k$ | 2nd Player | (Chen et al. 2021) |

# AlphaGo Zero to the Rescue?

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe
- Why not Notakto?

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe
- Why not Notakto?

(Chen et al. 2021)

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe
- Why not Notakto?

(Chen et al. 2021)

- AlphaGo Zero fails to learn competent play for small board sizes $n > 5$

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe
- Why not Notakto?

(Chen et al. 2021)

- AlphaGo Zero fails to learn competent play for small board sizes $n > 5$
- Rare nature of competent Notakto boards

# AlphaGo Zero to the Rescue?

**AlphaGo Zero** (Silver et al. 2017)

- *Generic* ANN-Based framework for board games
- Chess, Go, …
- Converges on a winning strategy for Tic-Tac-Toe
- Why not Notakto?

(Chen et al. 2021)

- AlphaGo Zero fails to learn competent play for small board sizes n > 5
- Rare nature of competent Notakto boards
- MCTS and sampling method also encode prior knowledge (Marcus 2018)

“Twin Systems”
(Keane and Kenny 2019)

Symbolic & ANN
Heavy Interaction

Logic ⟷ ANN
(Leitgeb 2005)

Symbolic w ANN
Component

Logic ---> ANN
(Garcez, Lamb, and Gabbay 2008)

**Weakly-Coupled**

**Strongly-Coupled**

# The Spectrum of Neuro-Symbolic Proposals

# Training Boards

(Lai 2015)

# Training Boards

(Lai 2015)

**Generating the Boards**

- Distribution
  - Boards should reflect competent play

- Variety
  - e.g. Boards that are unfairly stacked against player

- Volume
  - Enough boards for ANN to generalize

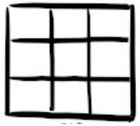
# Training Boards

(Lai 2015)

## Generating the Boards

- Distribution
  - Boards should reflect competent play
- Variety
  - e.g. Boards that are unfairly stacked against player
- Volume
  - Enough boards for ANN to generalize

## Evaluating the Boards



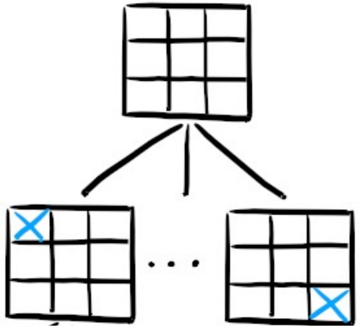
# Training Boards

(Lai 2015)

### Generating the Boards

- Distribution
  - Boards should reflect competent play
- Variety
  - e.g. Boards that are unfairly stacked against player
- Volume
  - Enough boards for ANN to generalize

### Evaluating the Boards



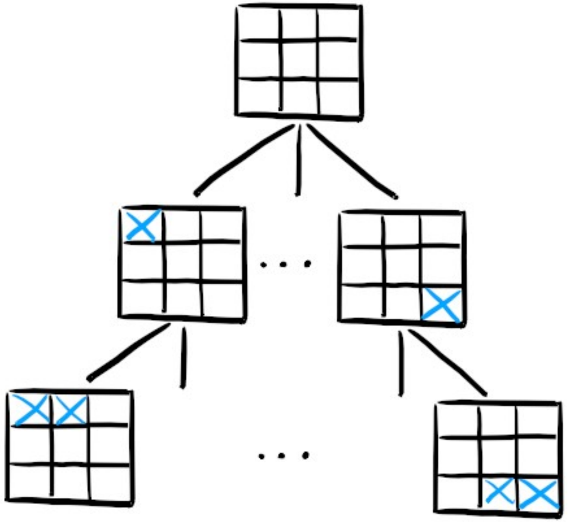
# Training Boards

(Lai 2015)

## Generating the Boards

- Distribution
  - Boards should reflect competent play
- Variety
  - e.g. Boards that are unfairly stacked against player
- Volume
  - Enough boards for ANN to generalize

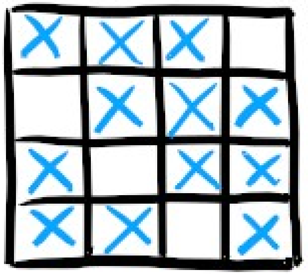
## Evaluating the Boards



# Feature Representation

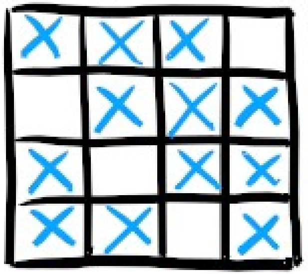
# Feature Representation

**Turn Flag**

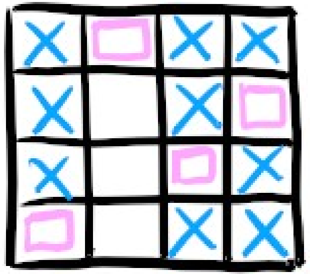


# Feature Representation

**Turn Flag**

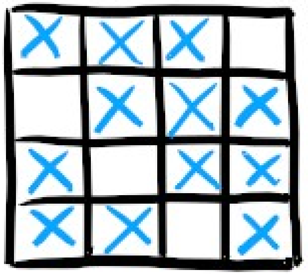


**Dead Squares**

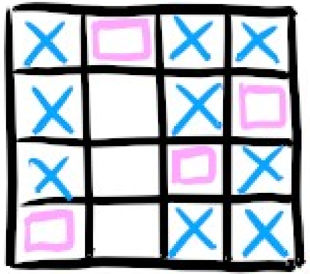


# Feature Representation

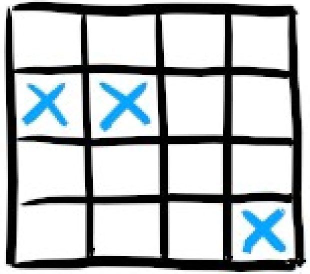
**Turn Flag**



**Dead Squares**

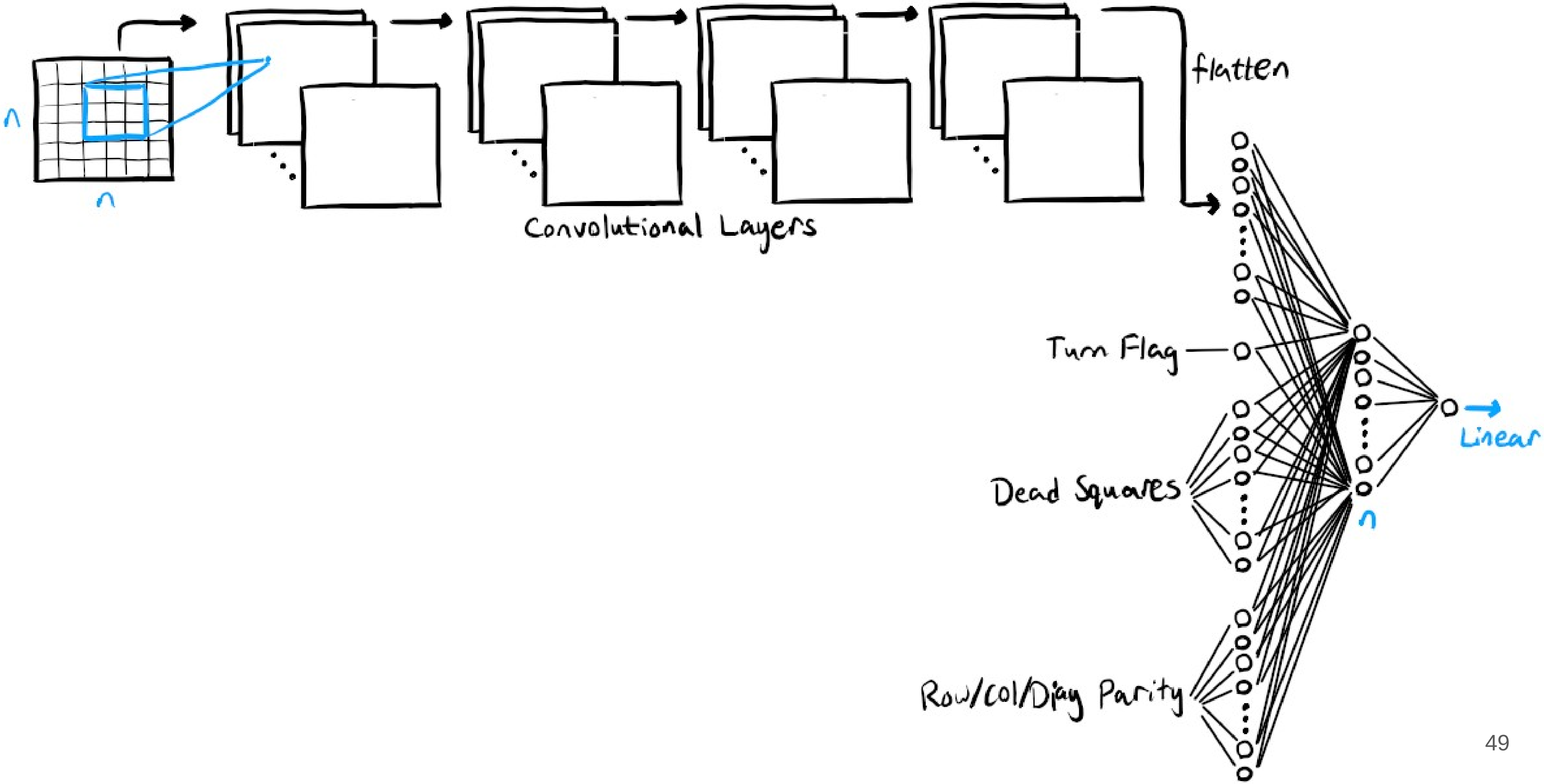


**Row Parity**



# Neural Network Architecture

# Neural Network Architecture



| n | Opponent | Featured Net | AlphaZero Net |
|---|---|---|---|
| $n = 3$ | Itself | 100% / 0% | 92% / 8% |
| | Featured Net | | 100% / 1% |
| | AlphaZero Net | 99% / 0% | |
| | Random | 94% / 82% | 90% / 84% |
| | Greedy | 79% / 26% | 88% / 20% |
| $n = 4$ | Itself | 8% / 92% | 92% / 8% |
| | Featured Net | | 18% / 82% |
| | AlphaZero Net | 18% / 82% | |
| | Random | 93% / 94% | 91% / 96% |
| | Greedy | 37% / 55% | 49% / 54% |
| $n = 5$ | Itself | 59% / 41% | 41% / 59% |
| | Featured Net | | 47% / 49% |
| | AlphaZero Net | 51% / 53% | |
| | Random | 99% / 99% | 99% / 96% |
| | Greedy | 54% / 48% | 60% / 39% |
| $n = 6$ | Itself | 23% / 77% | 69% / 31% |
| | Featured Net | | 33% / 53% |
| | AlphaZero Net | 47% / 67% | |
| | Random | 100% / 99% | 99% / 98% |
| | Greedy | 52% / 47% | 55% / 59% |

| n | Opponent | Featured Net | AlphaZero Net |
|---|---|---|---|
| $n = 3$ | Itself | 100% / 0% | 92% / 8% |
| | Featured Net | | 100% / 1% |
| | AlphaZero Net | 99% / 0% | |
| | Random | 94% / 82% | 90% / 84% |
| | Greedy | 79% / 26% | 88% / 20% |
| $n = 4$ | Itself | 8% / 92% | 92% / 8% |
| | Featured Net | | 18% / 82% |
| | AlphaZero Net | 18% / 82% | |
| | Random | 93% / 94% | 91% / 96% |
| | Greedy | 37% / 55% | 49% / 54% |
| $n = 5$ | Itself | 59% / 41% | 41% / 59% |
| | Featured Net | | 47% / 49% |
| | AlphaZero Net | 51% / 53% | |
| | Random | 99% / 99% | 99% / 96% |
| | Greedy | 54% / 48% | 60% / 39% |
| $n = 6$ | Itself | 23% / 77% | 69% / 31% |
| | Featured Net | | 33% / 53% |
| | AlphaZero Net | 47% / 67% | |
| | Random | 100% / 99% | 99% / 98% |
| | Greedy | 52% / 47% | 55% / 59% |

| n | Opponent | Featured Net | AlphaZero Net |
|---|---|---|---|
| $n = 3$ | Itself | 100% / 0% | 92% / 8% |
| | Featured Net | | 100% / 1% |
| | AlphaZero Net | 99% / 0% | |
| | Random | $\boxed{94\%}$ / 82% | $\boxed{90\%}$ / 84% |
| | Greedy | $\boxed{79\%}$ / 26% | $\boxed{88\%}$ / 20% |
| $n = 4$ | Itself | 8% / 92% | 92% / 8% |
| | Featured Net | | 18% / 82% |
| | AlphaZero Net | 18% / 82% | |
| | Random | 93% / $\boxed{94\%}$ | 91% / $\boxed{96\%}$ |
| | Greedy | 37% / $\boxed{55\%}$ | 49% / $\boxed{54\%}$ |
| $n = 5$ | Itself | 59% / 41% | 41% / 59% |
| | Featured Net | | 47% / 49% |
| | AlphaZero Net | 51% / 53% | |
| | Random | $\boxed{99\%}$ / 99% | $\boxed{99\%}$ / 96% |
| | Greedy | $\boxed{54\%}$ / 48% | $\boxed{60\%}$ / 39% |
| $n = 6$ | Itself | 23% / 77% | 69% / 31% |
| | Featured Net | | 33% / 53% |
| | AlphaZero Net | 47% / 67% | |
| | Random | $\boxed{100\%}$ / 99% | $\boxed{99\%}$ / 98% |
| | Greedy | $\boxed{52\%}$ / 47% | $\boxed{55\%}$ / 59% |

| n | Opponent | Featured Net | AlphaZero Net |
|---|---|---|---|
| $n=3$ | Itself | 100% / 0% | 92% / 8% |
| | Featured Net | | 100% / 1% |
| | AlphaZero Net | 99% / 0% | |
| | Random | 94% / 82% | 90% / 84% |
| | Greedy | 79% / 26% | 88% / 20% |
| $n=4$ | Itself | 8% / 92% | 92% / 8% |
| | Featured Net | | 18% / 82% |
| | AlphaZero Net | 18% / 82% | |
| | Random | 93% / 94% | 91% / 96% |
| | Greedy | 37% / 55% | 49% / 54% |
| $n=5$ | Itself | 59% / 41% | 41% / 59% |
| | Featured Net | | 47% / 49% |
| | AlphaZero Net | 51% / 53% | |
| | Random | 99% / 99% | 99% / 96% |
| | Greedy | 54% / 48% | 60% / 39% |
| $n=6$ | Itself | 23% / 77% | 69% / 31% |
| | Featured Net | | 33% / 53% |
| | AlphaZero Net | 47% / 67% | |
| | Random | 100% / 99% | 99% / 98% |
| | Greedy | 52% / 47% | 55% / 59% |

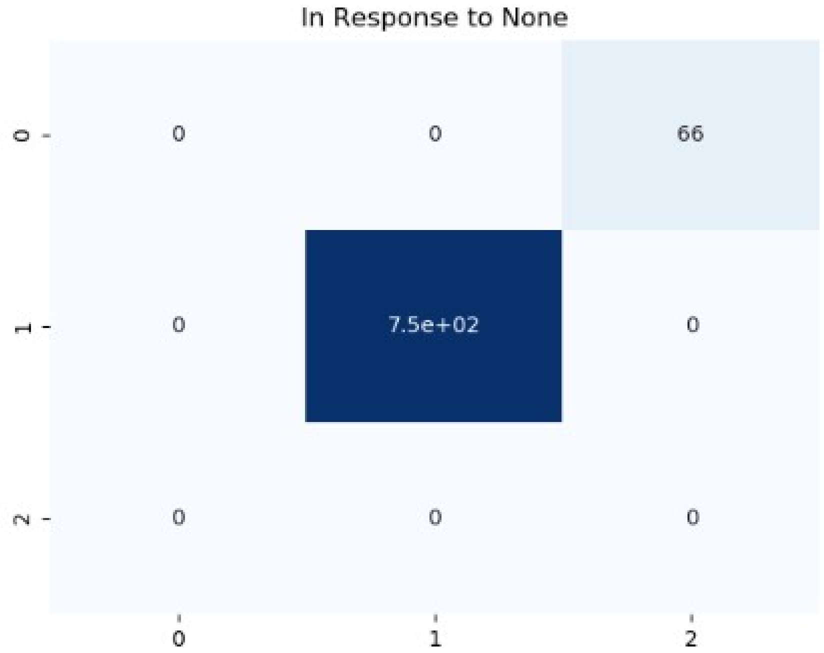
| n | Opponent | Featured Net | AlphaZero Net |
|---|---|---|---|
| $n = 3$ | Itself | 100% / 0% | 92% / 8% |
| | Featured Net | | 100% / 1% |
| | AlphaZero Net | 99% / 0% | |
| | Random | 94% / 82% | 90% / 84% |
| | Greedy | 79% / 26% | 88% / 20% |
| $n = 4$ | Itself | 8% / 92% | 92% / 8% |
| | Featured Net | | 18% / 82% |
| | AlphaZero Net | 18% / 82% | |
| | Random | 93% / 94% | 91% / 96% |
| | Greedy | 37% / 55% | 49% / 54% |
| $n = 5$ | Itself | 59% / 41% | 41% / 59% |
| | Featured Net | | 47% / 49% |
| | AlphaZero Net | 51% / 53% | |
| | Random | 99% / 99% | 99% / 96% |
| | Greedy | 54% / 48% | 60% / 39% |
| $n = 6$ | Itself | 23% / 77% | 69% / 31% |
| | Featured Net | | 33% / 53% |
| | AlphaZero Net | 47% / 67% | |
| | Random | 100% / 99% | 99% / 98% |
| | Greedy | 52% / 47% | 55% / 59% |

# Mapping the Net’s Strategy

- We try to extract the ANN’s strategy from a heat map
- 1000 games against Greedy
- Collect net’s *responses* to moves
- Only keep responses that resulted in a **win**

# Mapping the Net's Strategy

- We try to extract the ANN's strategy from a heat map
- 1000 games against Greedy
- Collect net's *responses* to moves
- Only keep responses that resulted in a **win**



| | | | |
|---|---|---|---|
| a | b | c | d |
| e | f | g | h |
| b | a | d | c |
| f | e | h | g |

## Using an LSTM Instead
(Near-Future Work)

**Note:**

- Winning strategies depend on
  *what opponent played last*
- Contrast with Chess, Go, Tic-Tac-Toe, ...

**Hypothesis:**

- Can’t evaluate a single static board
- We need sequential information

*So I should use, e.g. a ConvLSTM to learn Notakto.*

## Using an LSTM Instead
(Near-Future Work)

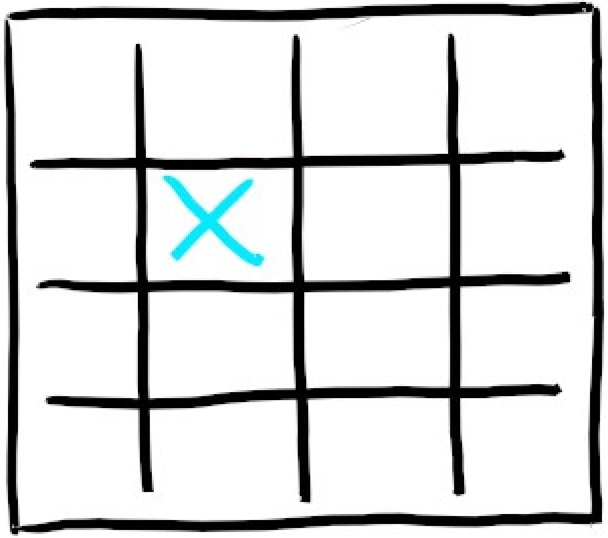
**Note:**

- Winning strategies depend on *what opponent played last*
- Contrast with Chess, Go, Tic-Tac-Toe, ...

**Hypothesis:**

- Can’t evaluate a single static board
- We need sequential information

*So I should use, e.g. a ConvLSTM to learn Notakto.*



## Using an LSTM Instead
(Near-Future Work)

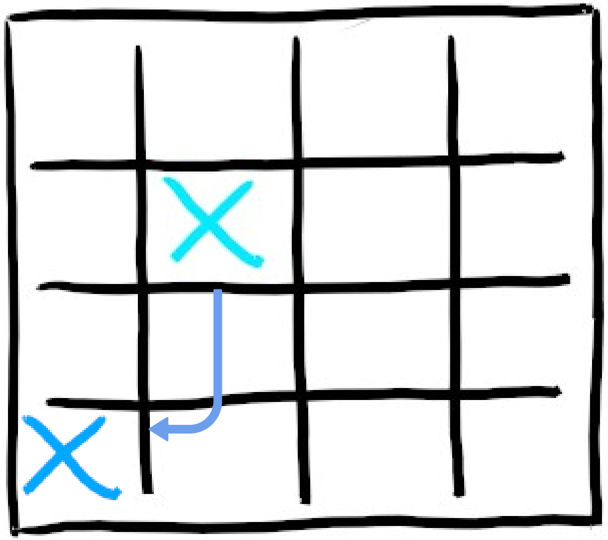
**Note:**

- Winning strategies depend on *what opponent played last*
- Contrast with Chess, Go, Tic-Tac-Toe, ...

**Hypothesis:**

- Can't evaluate a single static board
- We need sequential information

*So I should use, e.g. a ConvLSTM to learn Notakto.*



## Reconsidering Features
(Near-Future Work)

(Lai 2015)

## Reconsidering Features
(Near-Future Work)

(Lai 2015)

“For neural networks to work well, the feature representation needs to be relatively smooth in how the input space is mapped to the output space. Positions that are close together in the feature space should, as much as possible, have similar evaluations.”

# Q&A

**References:**


Chen, Z.; Wang, C.; Laturia, P.; Crandall, D.; and Blanco, S. 2021. **How to play Notakto: Can reinforcement learning achieve optimal play on combinatorial games?**

Chow, T. 2010. **Neutral tic tac toe.** MathOverflow. URL:https://mathoverflow.net/q/24693 (version: 2021-03-15).

Lai, M. 2015. **Giraffe: Using deep reinforcement learning to play chess**. arXiv preprint arXiv:1509.01549.

Marcus, G. 2018. **Innateness, alphazero, and artificial intelligence**. arXiv preprint arXiv:1801.05667.

Nair, S. 2018. **Alpha zero general (any game, any framework!).** https://github.com/suragnair/alpha-zerogeneral

Plambeck, T. E., and Whitehead, G. 2013. **The secrets of notakto: Winning at x-only tic-tac-toe**. arXiv preprint arXiv:1301.1672.

Silver, D. et al. 2017. **Mastering the game of go without human knowledg**e. *Nature* 550(7676):354–359.

## Github:

https://github.com/ais-climber/notakto-player/tree/main/v1_feedforward_net