university of groningen

Machine Learning

Quantum tic-tac-toe

› Department of Artificial Intelligence

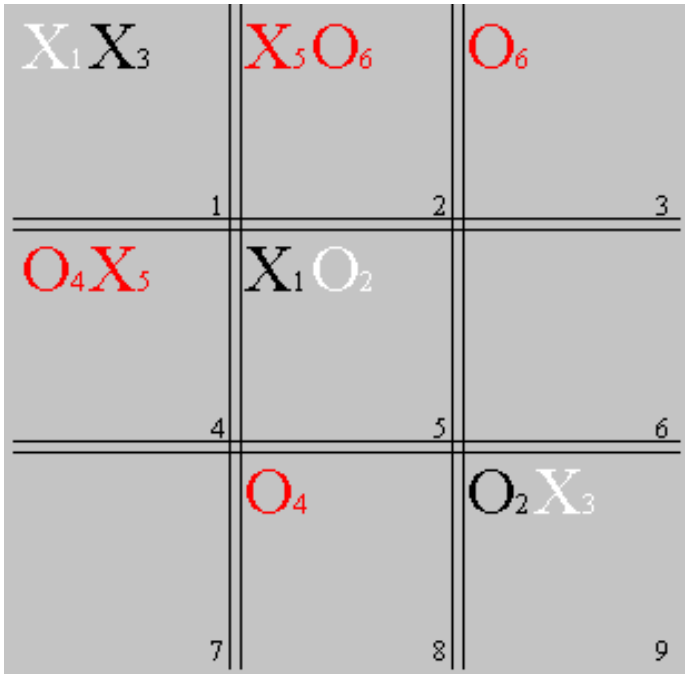
› Eva van Viegen
Peter Smit
Coen Jonker
Marc Volger

Overview

- Introduction
  - Problem description: Quantum-Tic-Tac-Toe
- Method
  - Agents
  - Feedforward agent learning
- Implementation
- Experimental set-up
- Preliminary Results
- Discussion

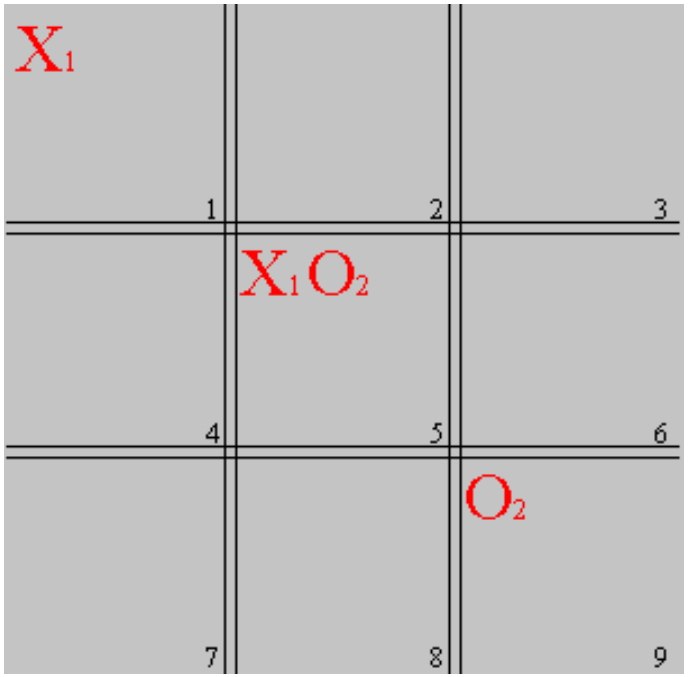
Introduction – Problem description

- Quantum tic-tac-toe



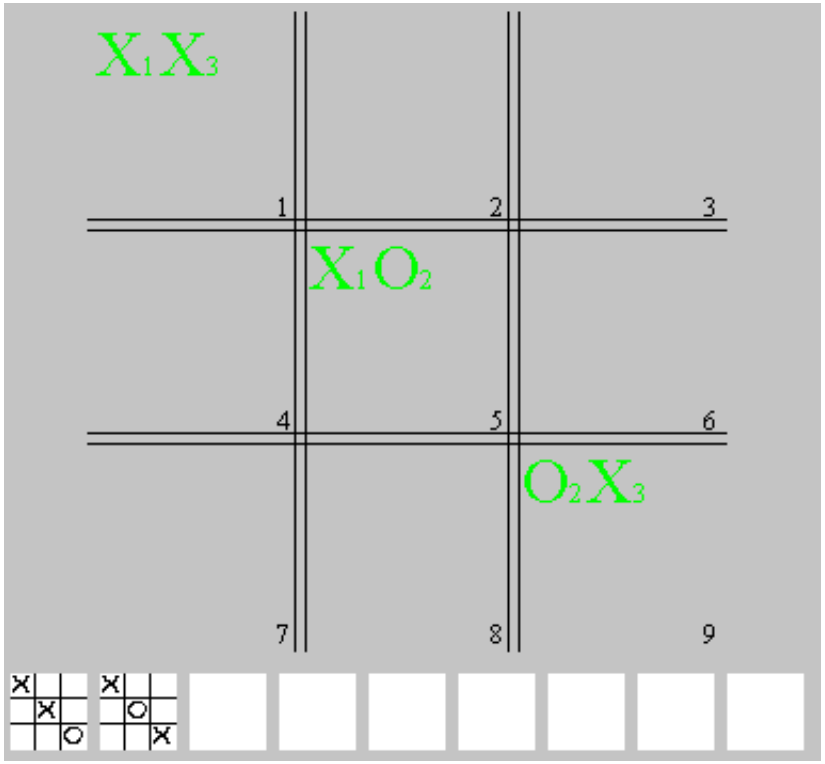
Introduction – Problem description

- Quantum tic-tac-toe
- Quantum move



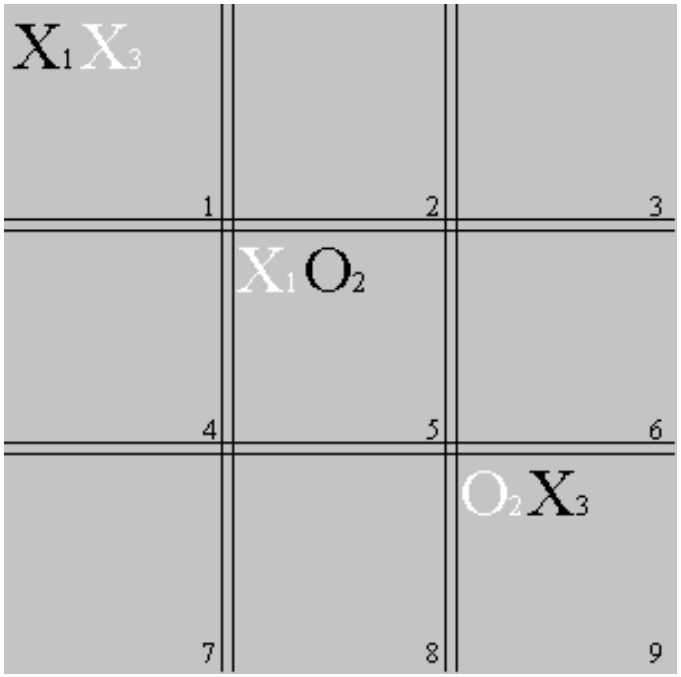
Introduction – Problem description

- Quantum tic-tac-toe
- Quantum move
- Cycle detection



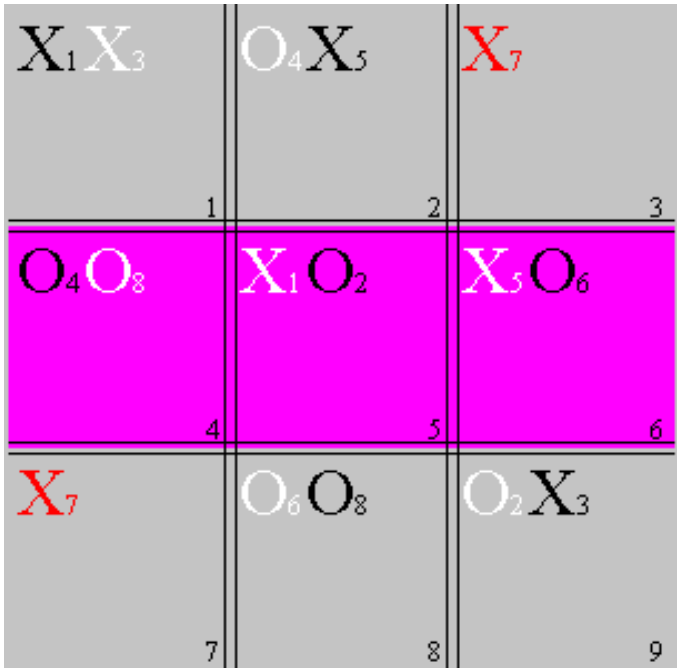
Introduction – Problem description

- Quantum tic-tac-toe
- Quantum move
- Cycle detection
- Collapse move



Introduction – Problem description

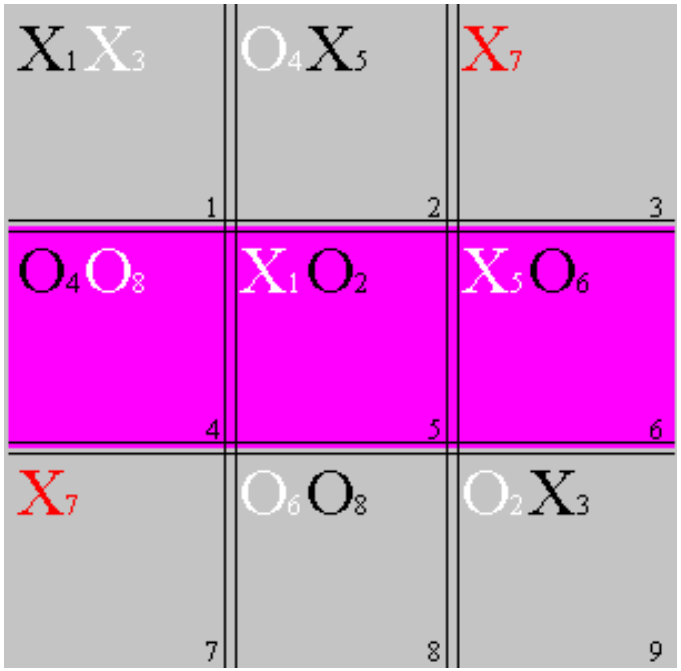
- Quantum tic-tac-toe
- Quantum move
- Cycle detection
- Collapse move
- End game



Introduction – Problem description

- Quantum tic-tac-toe
- Quantum move
- Cycle detection
- Collapse move
- End game

- Notice that X has an a priori chance of ±60% of winning



Method – Agents

- Two agents playing X or O in the game
  - Random agent
  - Smarter random agent (Winning moves)
  - Feedforward agent (Feedforward neural network)

- Agents use an internal gamemodel to play

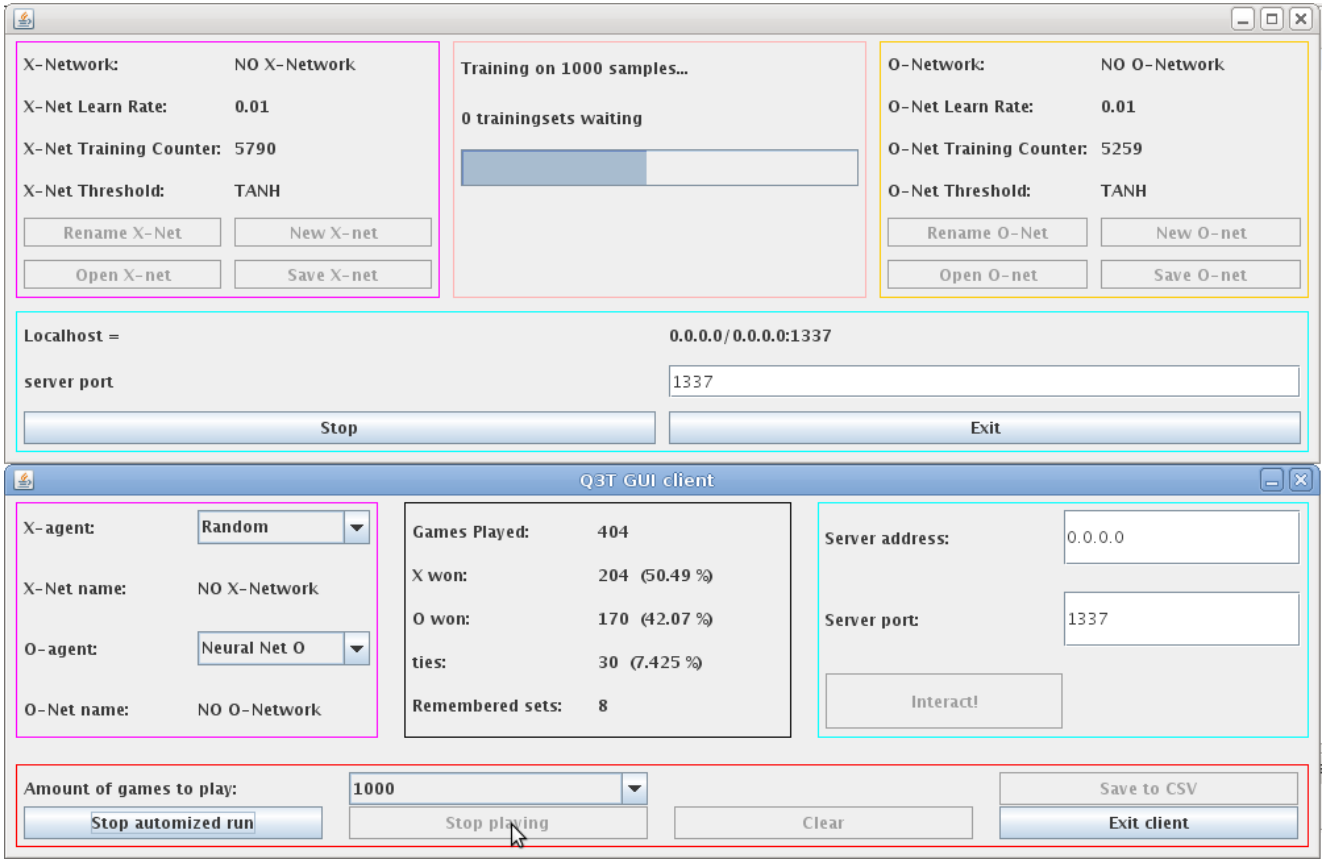
Method – Feedforward agent learning

- Clients play batch: 500, 1000, 5000, or 10000 games
- Server recieves batches and trains network on them
- Parameters of feedforward agent:
  - Threshold functions (tanh, sigmoid, no threshold)
  - Amount of levels of hidden neurons
  - Amount of hidden neurons per level
  - Learning rate
  - Initialization of the weights, always random
  - Epsilon, not adjustable, is 0.1

Implementation

- Software & Tools
  - Java Development Kit 1.6 (Netbeans 7.0)
  - Subversion 1.6.6
  - Java doc
  - jUnit tests (version 4)
- Server & Client
  - Client plays games
  - Server trains networks

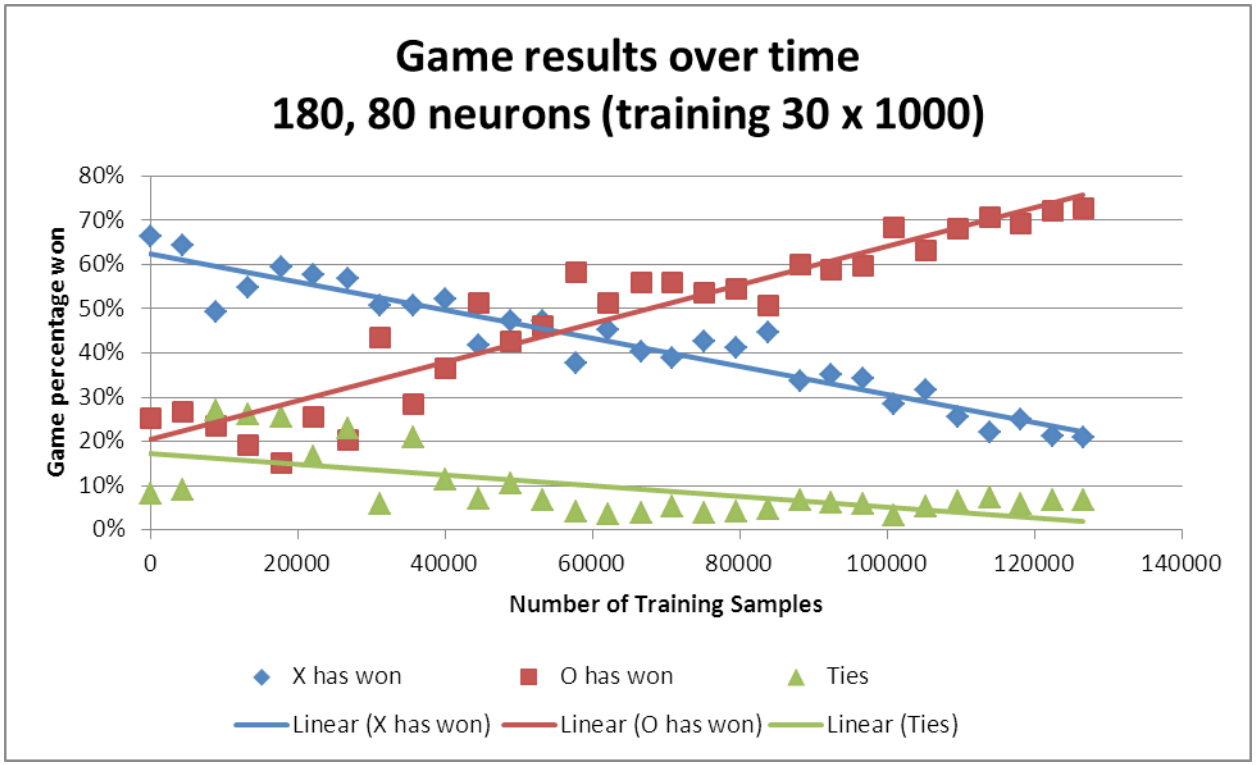
Implementation (2)



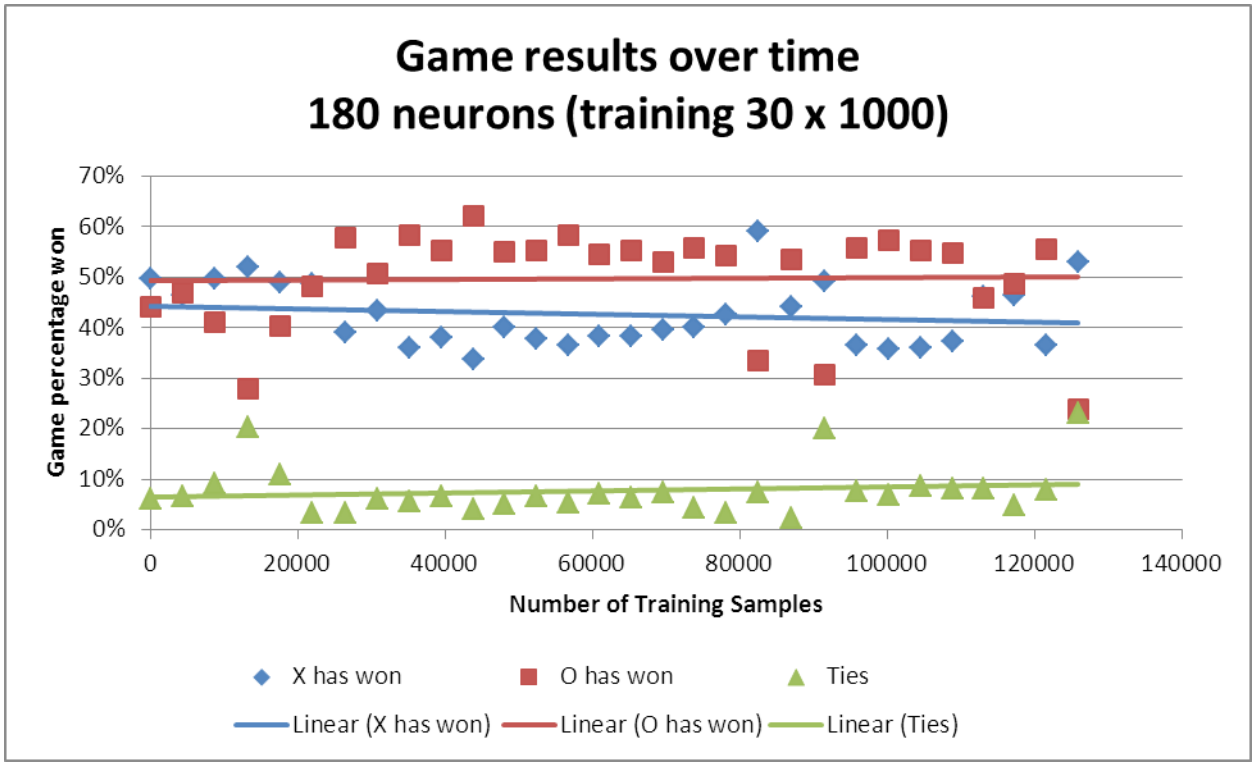
Experimental set-up

- X is played by random agent
- O is played by feedforward agent
  - Epsilon = 0.1
  - Threshold function is ‘tanh’
  - Learning rate = 0.01
  - Hidden neurons is varied (first, second layer):
    - 180, 80
    - 180
    - 80, 180

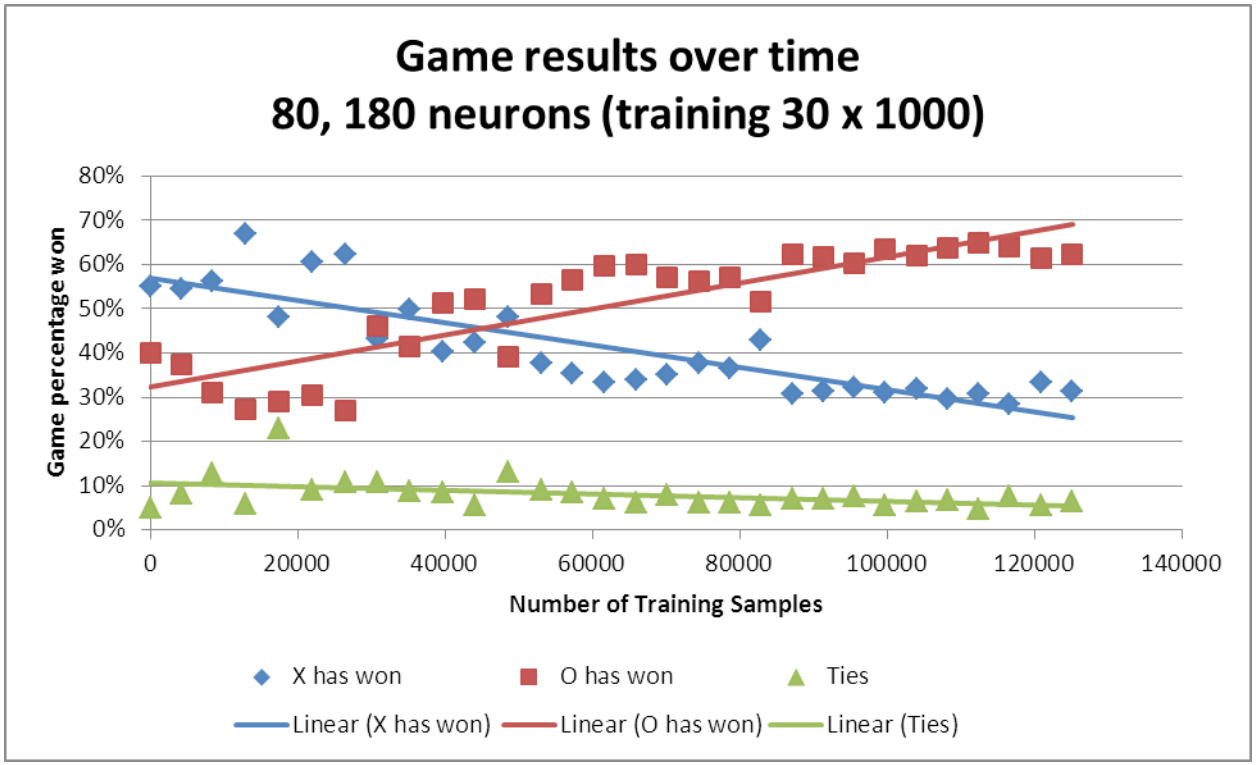
Preliminary Results



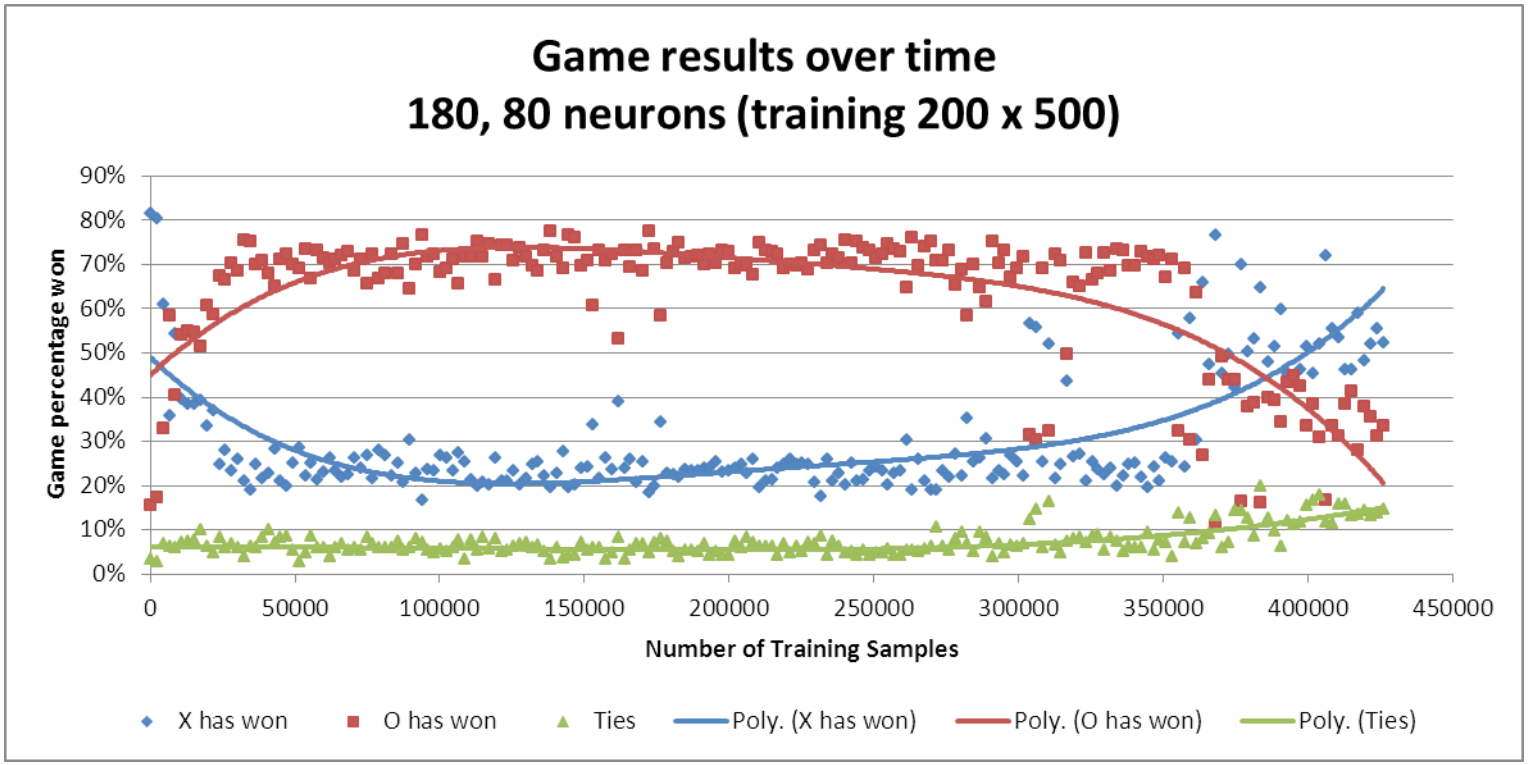
Preliminary Results (2)



Preliminary Results (3)



Preliminary Results (4)



Discussion

- Algorithm shows a nice learning curve
- But how to measure whether the results are good?
  - Random player is extremely bad
  - There are no human expert players (we know of)
- Ideas about smarter random agent with win/block moves
- Important: In what batch size is the server updated

Discussion (2)

- Looks like batch learning, but network actually trains on each game played
- Entire sweep?

- Upgrading to 4x4 tic-tac-toe

Thank you for your attention

http://www.codecup.nl
http://www.paradigmpuzzles.com/QT3Play.htm

Any questions?