CONNECT FOUR

CMPE 260 - PROJECT PRESENTATION

Abhishek Bais, Haley feng, princy joy, Shannon phu

Graduate students software engineering, SJSU

OUTLINE

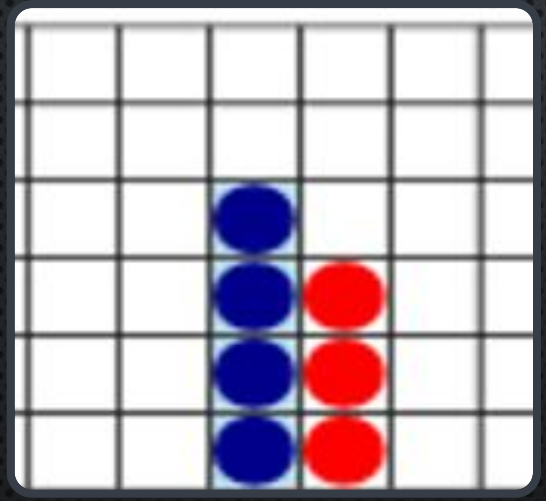
- The GAME
- Problem Statement
- Motivation
- Methodology
- Results
- Demo

THE GAME

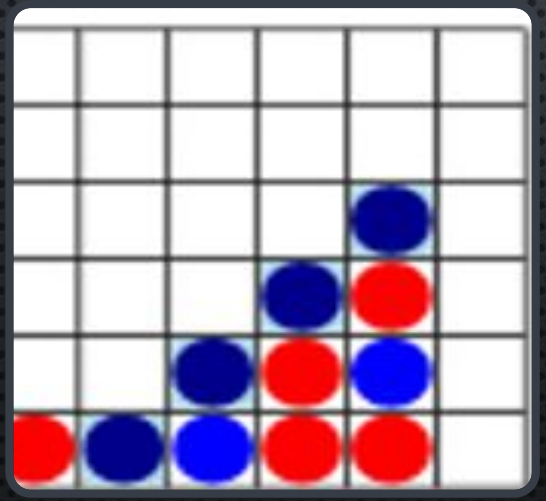
Connect Four is a popular two player game

Each player takes turns to drop a selected colored piece in a 6x7 grid

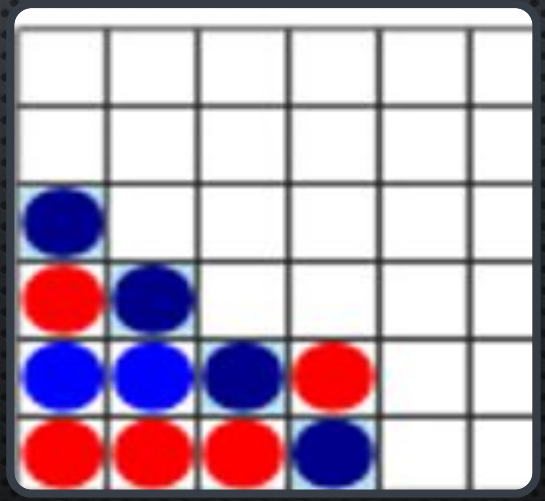
First player to form a 4-in-row connection wins



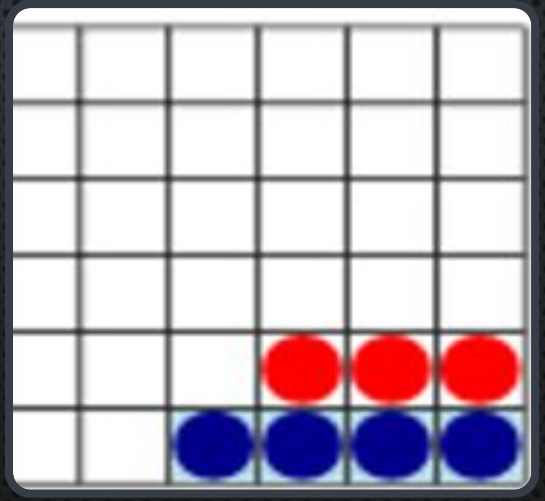
4-IN-ROW VERTICAL



4-IN-ROW DIAGONAL



4-IN-ROW DIAGONAL



4-IN-ROW HORIZONTAL

PROBLEM STATEMENT

CREATE A REAL-LIFE CONNECT FOUR GAMING EXPERIENCE FOR A HUMAN PLAYER AGAINST A COMPUTER AGENT

TRAIN REINFORCEMENT LEARNING (RL) GUIDED COMPUTER AGENTS VIA BATTLES AGAINST A COMPUTER AGENT THAT MAKES RANDOM MOVES AND AGAINST A COMPUTER AGENT THAT USES A NON-REINFORCEMENT LEARNING MINIMAX ALGORITHM TO PICK BEST COMPUTER AGENT IN TERMS OF WIN RATE AND EFFICIENCY

THE FOUR COMPUTER AGENTS CONTRASTED ARE

MINIMAX AGENT

USES A BACKTRACKING, RECURSIVE ALGORITHM USED IN GAME THEORY TO MAKE MOVES THAT RESULT IN MAXIMUM IMMEDIATE GAIN

MONTE CARLO AGENT

USES REINFORCEMENT LEARNING TO LEARN DIRECTLY FROM GAME EXPERIENCES WITHOUT USING ANY PRIOR MARKOV DECISION PROCESS KNOWLEDGE

Q LEARNING AGENT

USES A REINFORCEMENT LEARNING OFF-POLICY VALUE BASED SCHEME BASED ON THE BELLMAN'S EQUATION TO LEARN THE VALUE OF OPTIMAL POLICY REGARDLESS OF ACTION

SARSA LEARNING AGENT

USES A REINFORCEMENT LEARNING ON-POLICY VALUE BASED SCHEME TO LEARN THE VALUE OF THE OPTIMAL POLICY BASED ON ACTION DERIVED FROM CURRENT POLICY

MOTIVATION

MODEL AN INTERMEDIATE COMPLEXITY (6X7 BOARD GAME)

MOST EXISTING REINFORCEMENT LEARNING MODELS IN THE GAMING CONTEXT ARE FINE-TUNED TO TIC-TAC-TOE (A SIMPLISTIC 3X3 BOARD GAME WITH A SMALL STATE SPACE) OR ALPHA-GO (A PROGRAM THAT PLAYS GO, A 19X19 BOARD GAME AFTER STORING 30 MILLION POSITIONS)

PROVIDE INSIGHTS INTO HOW REINFORCEMENT LEARNING GUIDED COMPUTER AGENTS PERFORM IN BATTLES AGAINST A COMPUTER AGENT THAT MAKES RANDOM MOVES AND IN BATTLES AGAINST A COMPUTER AGENT THAT USE MINIMAX (A BACKTRACKING, RECURSIVE GAME THEORY ALGORITHM) IN TERMS OF WIN RATE AND EFFICIENCY

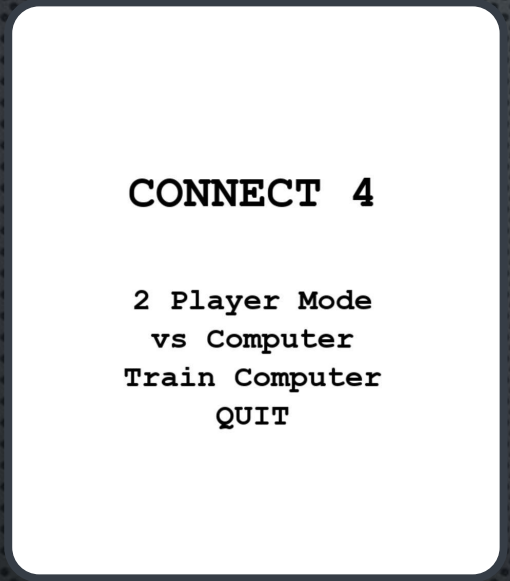
METHODOLOGY

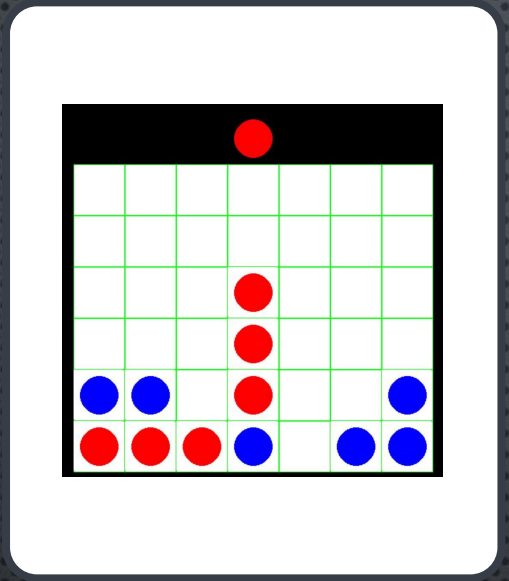
SETUP THE GAME TO BE PLAYED IN THREE MODES
Mode 1 - Single Player mode (Human vs trained rl guided computer agent)
Mode 2 - Two Player mode (Both human)
Mode 3 – train mode (2 rl guided computer agents battle against each other for n iterations)

IMPLEMENTED FOUR DIFFERENT ALGORITHMS TO MODEL FOUR COMPUTER AGENTS
Algorithm 1 - minimax
Algorithm 2 - q learning
Algorithm 3 - sarsa learning
Algorithm 4 - monte carlo

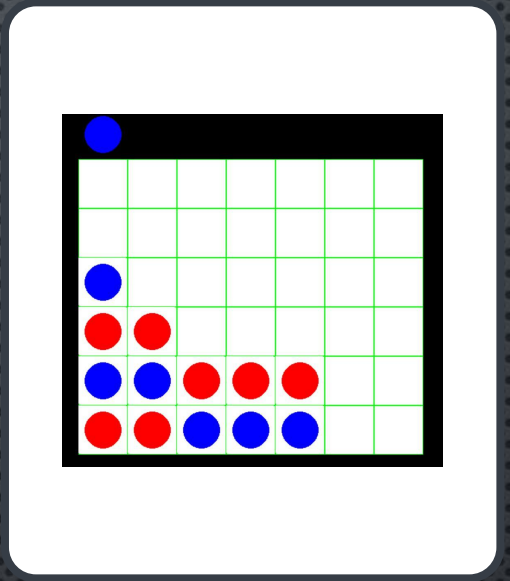
TRAINED AND HYPER-PARAMETER TUNED RL GUIDED COMPUTER AGENTS IN BATTLES VS RANDOM MOVE COMPUTER AGENT, MINIMAX

COMPARED AND CONTRASTED RL GUIDED AGENTS ON WIN RATIO, EFFICIENCY

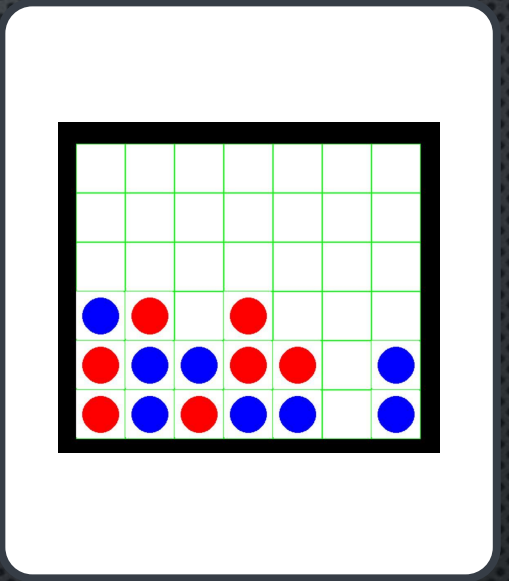




2 PLAYER MODE



Vs COMPUTER AGENT



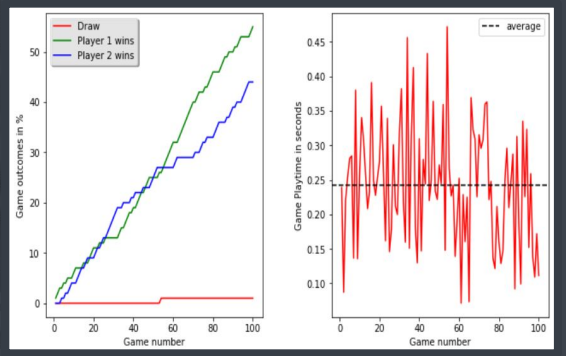
TRAIN RL GUIDED COMPUTER AGENTS

METHODOLOGY – SETUP THE GAME

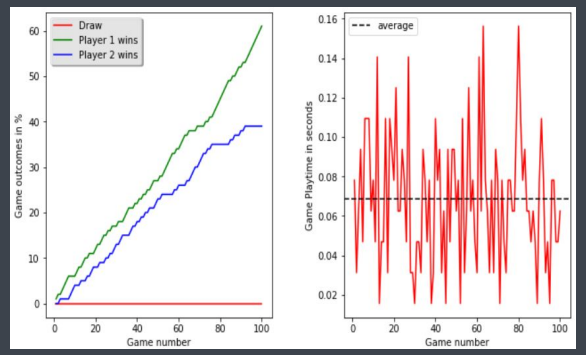
ALGORITHM	HYPER PARAMETERS
Q LEARNING	epsilon - 0.2 alpha - [learning rate, tuning exp 0.05, 0.25, 0.3, 0.75, 0.9] gamma - [discount, tuning exp 0.25, 0.50, 0.75, 0.9, 0.98]
SARSA LEARNING	epsilon - 0.2 alpha - [learning rate, tuning exp 0.05, 0.25, 0.3, 0.75] gamma - [discount, tuning exp 0.25, 0.50, 0.75, 0.9, 0.98]
MONTE CARLO	exploration coefficient - [tuning exp 0.8, 1, 1.4, 1.8]
MINIMAX	depth - 0.5

QLEARNER, SARSA LEARNER first tune learning rate ‘alpha’, then tune discount ‘gamma’ on top of it

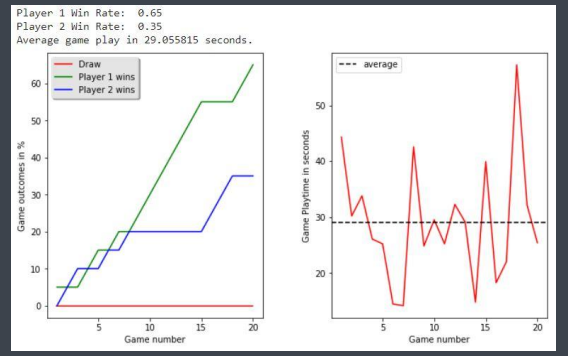
METHODOLOGY – HYPER PARAMETER TUNING FOR RL GUIDED COMPUTER AGENTS



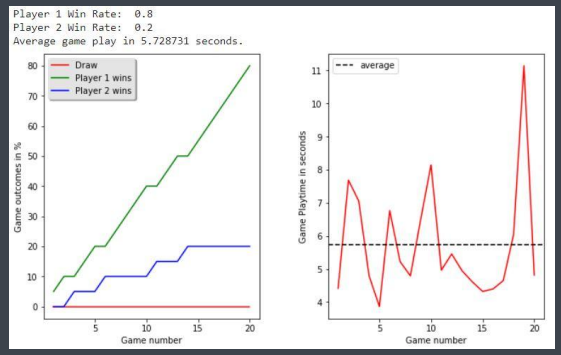
QLEARNER VS RANDOM MOVE AGENT
Win Rate 55%



SARSA LEARNER VS RANDOM MOVE AGENT
Win Rate 61%



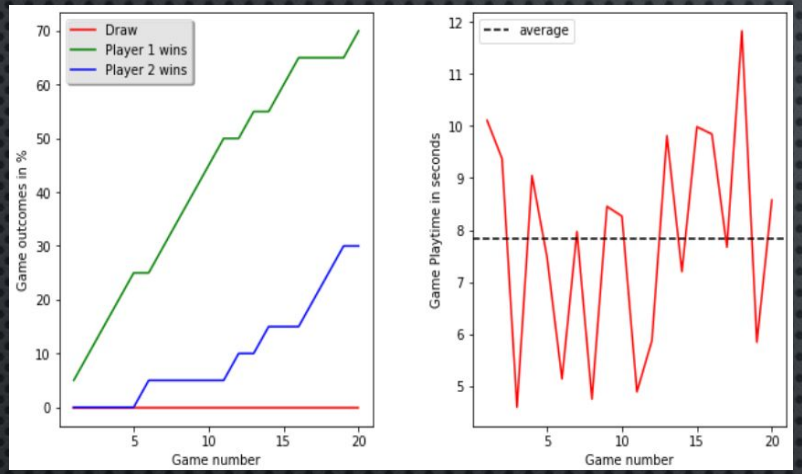
MONTE CARLO VS RANDOM MOVE AGENT
Win Rate 65%



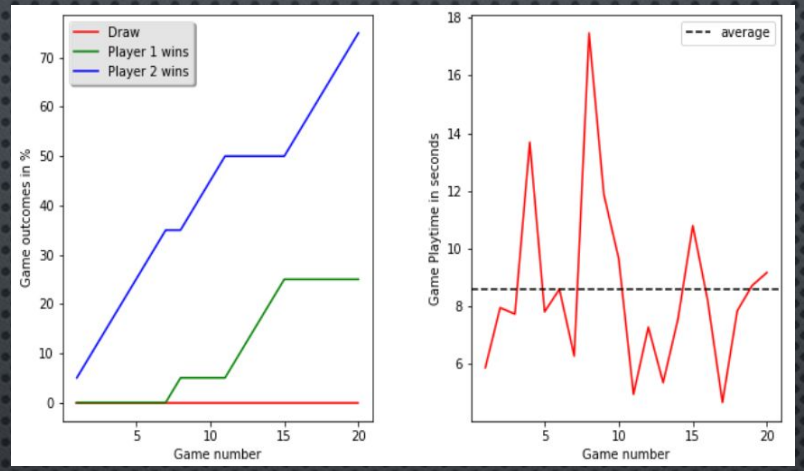
MINIMAX VS RANDOM MOVE AGENT
Win Rate 80%

MINIMAX has the highest Win Ratio = 80% in 20 battles vs RANDOM MOVE Agent

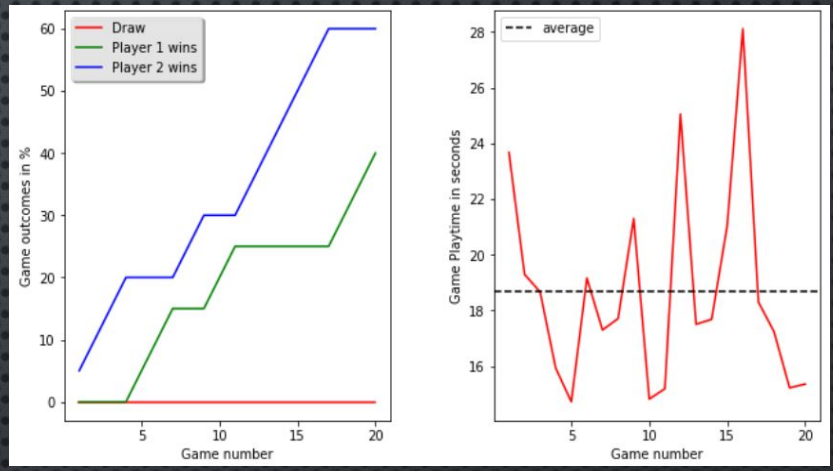
RESULTS – COMPUTER AGENTS VS RANDOM MOVE COMPUTER AGENT WIN RATE



MINIMAX (Player 1) vs
Q LEARNER (Player 2)
QLEARNER Win Rate 30%



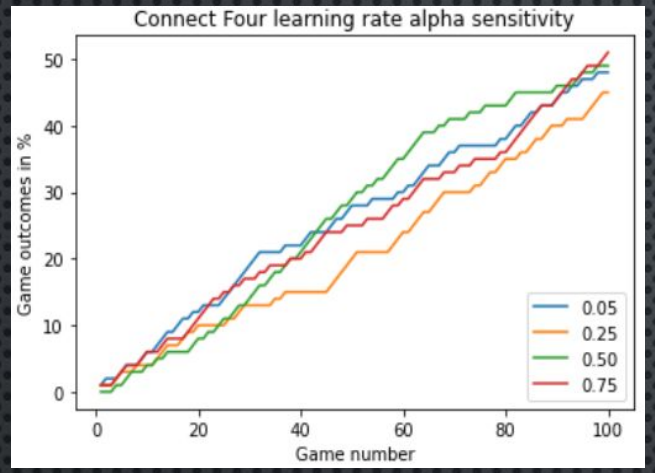
MINIMAX (Player 1) vs
SARSA LEARNER (Player 2)
SARSA LEARNER Win Rate 75%

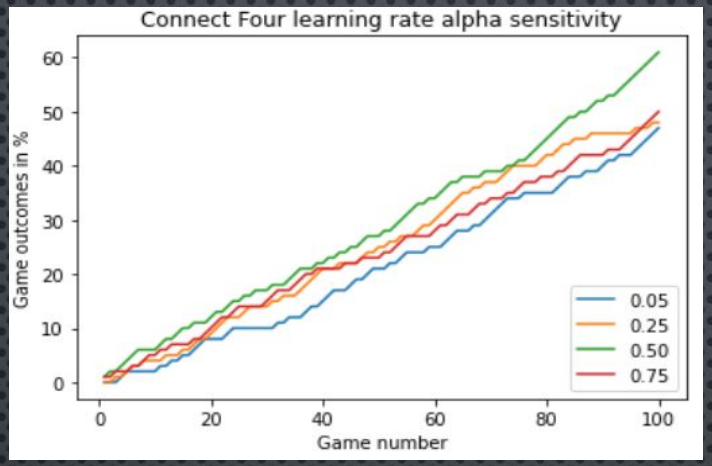


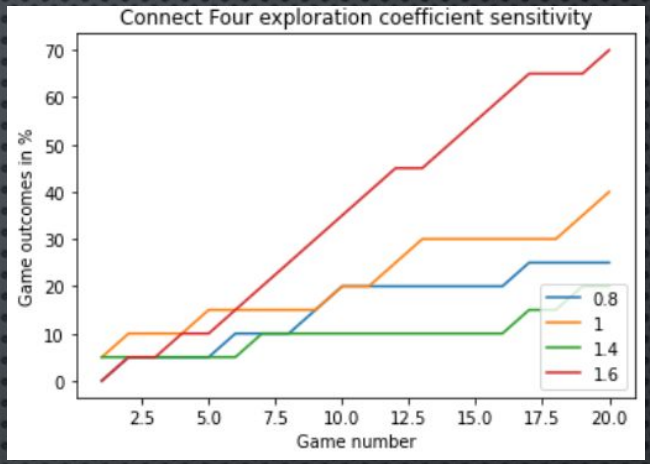
MINIMAX (Player 1) vs
MONTE CARLO (Player 2)
MONTE CARLO Win Rate 60%

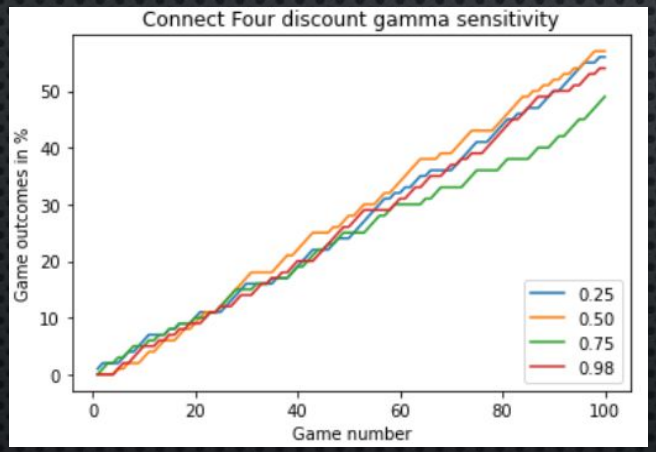
SARSA LEARNER has the highest Win Ratio = 75% in 20 battles vs MiniMax Agent

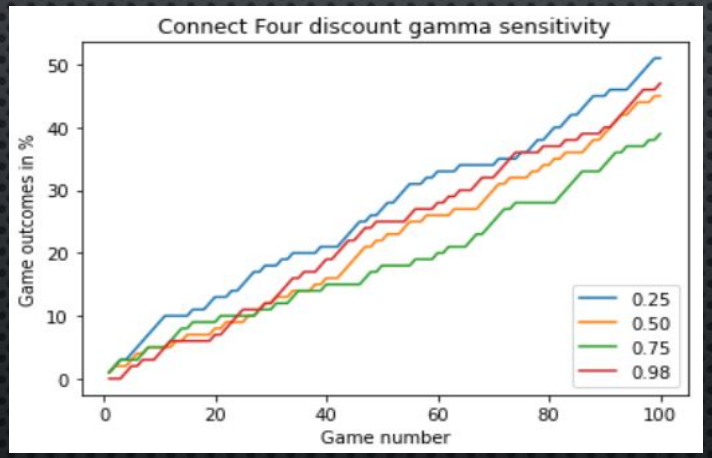
RESULTS – RL GUIDED COMPUTER AGENTS VS MINIMAX AGENT WIN RATE











Q LEARNER
(lr 0.50, discount 0.50 best)

SARSA LEARNER
(lr 0.50, discount 0.25 best)

MONTE CARLO
(exploration coeff 1.6 best)

RESULTS – RL GUIDED COMPUTER AGENTS HYPER PARAMETER TUNING

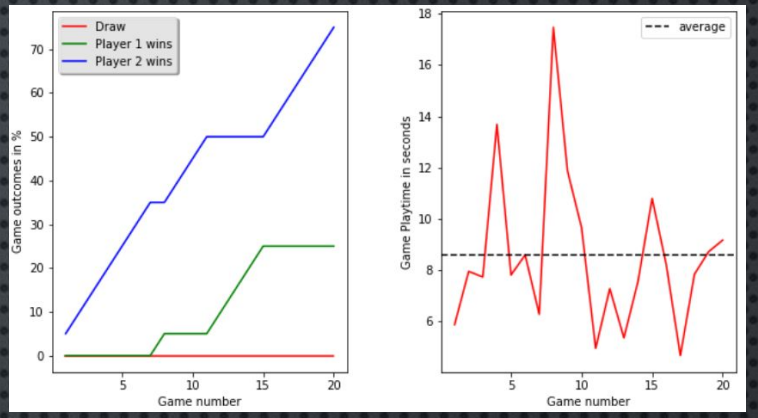
WIN RATE COMPARISON

BEST OVERALL
WORST OVERALL
BEST IN COLUMN

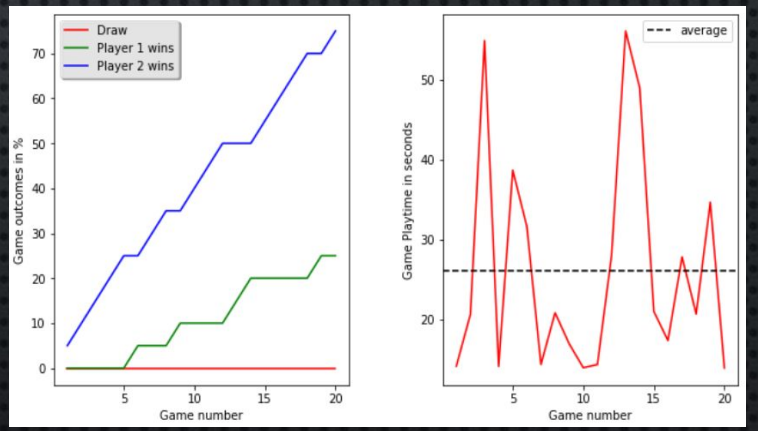
	VS BASELINE AGENTS		VS RL GUIDED COMPUTER AGENT		
	Random Move	Minimax *(80%-win rate vs Random Move)*	Q LEARNER	SARSA LEARNER	MONTE CARLO
Q LEARNER	0.55	0.30	NA	**0.44**	0.35
SARSA LEARNER	0.61	**0.75**	0.55	NA	**0.75**
MONTE CARLO	**0.65**	**0.60**	**0.65**	**0.25**	NA

RL GUIDED COMPUTER AGENT VS BASELINE AGENTS
SARSA LEARNER HAS HIGHEST WIN RATE OF 75% VS MINIMAX AGENT

RL GUIDED COMPUTER AGENT VS RL GUIDED COMPUTER AGENTS
SARSA LEARNER HAS THE HIGHEST WIN RATE OF 75% VS MONTE CARLO



MINIMAX (Player 1) vs
SARSA LEARNER (Player 2)
SARSA LEARNER Win Rate 75%



MONTE CARLO (Player 1) vs
SARSA LEARNER (Player 2)
SARSA LEARNER Win Rate 75%

RESULTS – WIN RATE

SPEED COMPARISON

FASTEST / SLOWEST	VS BASELINE AGENTS		VS RL GUIDED COMPUTER AGENT		
	Random Move	Minimax *(80%-win rate vs Random Move)*	Q LEARNER	SARSA LEARNER	MONTE CARLO
Q LEARNER	0.2422	**7.835**	NA	**0.0634**	24.138
SARSA LEARNER	**0.0681**	8.575	0.0634	NA	**26.117**
MONTE CARLO	**29.055**	**18.661**	**24.138**	**26.117**	NA

RL GUIDED COMPUTER AGENT VS BASELINE AGENTS

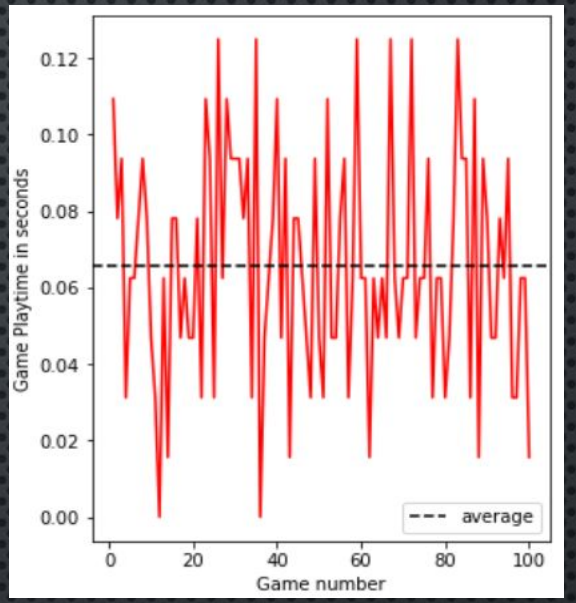
SARSA LEARNER IS FASTEST, AVG PLAY TIME OF 0.06s VS RANDOM MOVE AGENT

MONTE CARLO IS SLOWEST, AVG PLAY TIME OF 29.06s VS RANDOM MOVE AGENT

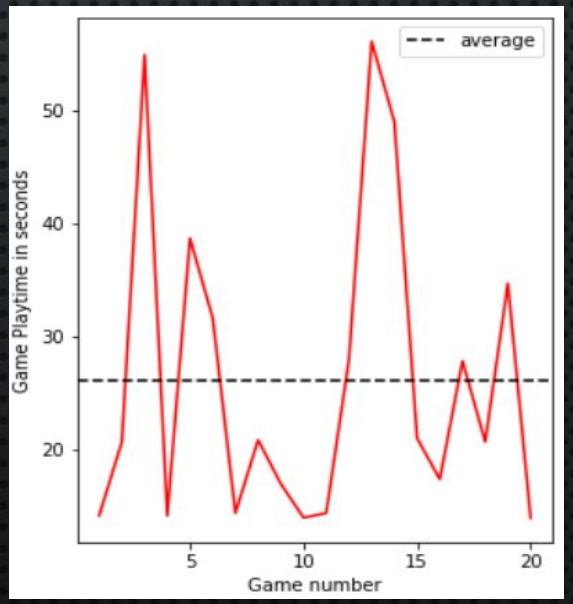
RL GUIDED COMPUTER AGENT VS RL GUIDED COMPUTER AGENTS

SARSA LEARNER IS FASTEST, AVG PLAY TIME OF 0.0634s VS Q LEARNER AGENT

MONTE CARLO IS SLOWEST, AVG PLAY TIME OF 26.117s VS SARSA LEARNER AGENT



SARSA



MONTE CARLO

RESULTS – EFFICIENCY

GITHUB REPO