A meta analysis of tournaments and an evaluation of performance in the Iterated Prisoner's Dilemma.

Nikoleta E. Glynatsi, Dr Vincent A. Knight

1 Abstract

The Iterated Prisoner's Dilemma has been used for decades as a powerful model of behavioural interactions. From the celebrated performance of Tit for Tat in the early computer tournaments, to the introduction of Pavlov and the use of sophisticated strategies such as neural networks, the literature has been exploring the performance of strategies in the iterated games for years. With the use of an open source package we now have access to most of these strategies. This paper conducts a meta analysis of a large number of tournaments and evaluates the performance of over 200 strategies from the literature.

2 Background

The Iterated Prisoner's Dilemma is a two players repeated game. At each turn a player chooses between cooperation (C) or defection (D) whilst the prior outcomes matters. The payoff at each given turn are defined by the matrix,

$$\begin{pmatrix} R & S \\ T & P \end{pmatrix}$$

where $T > R > P > S$ and $2R > T + S$.

In the early 80's **literature review on strategies**.

Type of tournaments, **literature review on types**, for example a standard tournament is a round robin tournament where the number of turns and the tournament is also repeated. A noisy tournament is similar to a standard tournament. A noisy tournament in a round robin tournament where noise is introduced. Noise is the probability that a players action is flipped.

A probabilistic ending tournament. Similar to a standard tournament however in a probabilistic ending tournament the number of turns is not specified. There is a probability (ranges between 0 and 1) that the match will end in the next round.

Over time which strategies were defined as the best

This manuscript accesses the performance of strategies. This is achieved by analysing a large data set containing results of the iterated prisoner's dilemma tournaments, which is archived in.

The structure of this manuscript is as follows. In Section 3 the data collection is covered. In Section 4, the best performed strategies for each type of tournament and overall are presented. Section 5, explores the traits which contribute to good performance and finally in Section 6 we conclude and summarise the results.

3 Data generating process

For the purposes of this manuscript a large data set containing results on iterate prisoner's dilemma tournaments has been generated. In this section the method and the outputs of the data generating process are covered.

This manuscript uses the open source package Axelrod-Python [1] to simulate tournaments of the IPD, more specifically, version 3.0.0. Axelrod-Python allows for different types of IPD computer tournaments to be simulated whilst containing a list of over 200 strategies. Most of these are strategies described in the literature, with a few exceptions being strategies that have been contributed specifically to the package. Though Axelrod-Python features several tournament types, this work considers only standard tournaments, noisy tournaments, probabilistic ending tournaments and noisy probabilistic ending ones.

Standard tournaments, are tournaments with N strategies and an n number of turns. Similarly, **noisy tournaments** are of N strategies and of n number of turns but at each turn there is a noise probability of p that a player's action will be flipped. **Probabilistic ending tournaments**, are of size N and after each turn a match between strategies ends with a given probability e. Finally, **noisy probabilistic ending** tournaments have both a noise probability p and an ending probability e. For smoothing the simulated results a tournament is repeated for r number of times.

For each run of the generating data process a random size N, a random list of strategies are selected and a random number of repetitions r, and for these, a standard, noise, probabilistic ending and a noisy probabilistic ending tournament are performed. More specifically, the process of data generation is given in Algorithm 1.

foreach *seed* $\in [0, 12285]$ **do**

$N \leftarrow$ randomly select integer from $[N_{min}, N_{max}]$;

players $\leftarrow$ randomly select players from list of size $[N_{min}, N_{max}]$;

$r \leftarrow$ randomly select integer from $[r_{min}, r_{max}]$;

$n \leftarrow$ randomly select integer from $[n_{min}, n_{max}]$;

$p \leftarrow$ randomly select float from $[p_{min}, p_{max}]$;

$e \leftarrow$ randomly select float from $[e_{min}, e_{max}]$;

result standard $\leftarrow$ Axelrod.tournament(players, n, r);

result noisy $\leftarrow$ Axelrod.tournament(players, n, p, r);

result probabilistic ending $\leftarrow$ Axelrod.tournament(players, e, r);

result noisy probabilistic ending $\leftarrow$ Axelrod.tournament(players, p, e, r);

end

return *result standard, result noisy, result probabilistic ending, result noisy probabilistic ending*;

At each given run of Algorithm 1 the run parameters are randomly selected. The parameters of each run and their respective min and max values are given by Table 1.

parameter	parameter explanation	min value	max value
N	number of strategies	3	195
r	number of repetitions	10	100
n	number of turns	1	200
p	probability of flipping action at each turn	0	1
e	probability of match ending in the next turn	0	1

Table 1: Data generation parameters' values

The source code for the data generating process has been packaged and is available here. A total of 12,285 runs of Algorithm 1 have been performed. For each run the results for 4 different tournaments were collected, thus a total of 49,140 $(12,285 \times 4)$ tournaments results have been retrieved. Each tournament outputs a result summary in the form of Table 2. The result summary has a length N because each row contains information for each strategy participated in the tournament.

						Rates							
Rank	Name	Median score	Cooperation rating	Win	Initial C	CC	CD	DC	DD	CC to C	CD to C	DC to C	DD to C
0	EvolvedLookerUp2 2 2	2.97	0.705	28.0	1.0	0.639	0.066	0.189	0.106	0.836	0.481	0.568	0.8
1	Evolved FSM 16 Noise 05	2.875	0.697	21.0	1.0	0.676	0.020	0.135	0.168	0.985	0.571	0.392	0.07
2	PSO Gambler 1 1 1	2.874	0.684	23.0	1.0	0.651	0.034	0.152	0.164	1.000	0.283	0.000	0.136
3	PSO Gambler Mem1	2.861	0.706	23.0	1.0	0.663	0.042	0.145	0.150	1.000	0.510	0.000	0.122
4	Winner12	2.835	0.682	20.0	1.0	0.651	0.031	0.141	0.177	1.000	0.441	0.000	0.462
...	...	...	...	...	...	...	...	...	...	...	...	...	...

Table 2: Output result.

Several other measures are included in tha data set. These include the tournament parameters given in Table 1 and the measures described in Table 3. A few of this measures are strategies' classifiers as given in Axelrod-Python, such as stochastic and memory depth. SSeeror is a measure defined in literature and the cooperation rating measures are calculated from the outputs of the tournaments.

The data collection and the output from the generated tournaments have been covered in this section. Several measures that will be used in the analysis of the following sections have been presented. These are also summarised in Table in the Appendix. In the following Section, the top performances at each tournament are overall are presented.

measure	measure explanation	value
stochastic	strategy classifier from [1]. Whether a strategy is stochastic	True or False
memory depth	strategy classifier from [1]. the number of turns a strategy takes into account to make a decision	0 to inf
makes use of game	strategy classifier from [1]. Whether a strategy makes used of the game information	
SSeeror	how extortionate a strategy behaves as stated in	
memory usage		
cooperation rating max		
cooperation rating min		
cooperation rating median		
cooperation rating mean		
cooperation rating comp to max		
cooperation rating comp to min		
cooperation rating comp to median		
cooperation rating comp to mean		

Table 3: Manually calculated measures

4 Top Performances

In this section the top performed strategies for each tournament type, and overall are presented. The strategies have been ranked based on their median normalised rank, which is denoted as $\bar{R}$.

4.0.1 Standard Tournaments

A total of 12,285 results on standard tournaments have been collected. On average a standard tournament had 122 strategies, a lenght of 100 turns and were repeated for 55 repetitions. The tournament top 15 performances in such tournaments are given by Table 4 and their distributions are given by Figure 1.

The top 10 out of the 15 strategies, are variations of Evolved FSM, Evolved HMM, EvolvedLookerUp and PSO Gambler, are strategies introduced in [3]. These are all strategies that have be trained using reinforcement learnings techniques to adjust their game play. Moreover, they have been trained using the strategies within the library [1], thus their performance is not a surprise. DoubleCrosser, and Fool Me Once, do not exist within the literature but have been especially written to contribute to [1]. Fool me once will forgive one defection and then defects for ever when DoubleCrosser will forgive the first 3 defections and then defect forever. Moreover DoubleCrosser is a strategy which makes use of the lenght of the game. Finally, Winner 12 [4] and DBS are both from the the literature. DBS [2] is strategy specifically trained for noisy enviroments, however, it ranks highly only in standard ones.

Name	$\bar{R}$ in standard tournaments
Evolved HMM 5	0.006579
Evolved FSM 16	0.009901
EvolvedLookerUp2_2_2	0.010638
Evolved FSM 16 Noise 05	0.016393
PSO Gambler 2_2_2	0.021390
Evolved ANN	0.028736
Evolved ANN 5	0.033898
PSO Gambler 1_1_1	0.037234
Evolved FSM 4	0.048387
PSO Gambler Mem1	0.050000
Winner12	0.059459
Fool Me Once	0.061224
DBS	0.070866
DoubleCrosser	0.071895
BackStabber	0.075000

Table 4: Standard top performances

In the top strategies there appears to be a mixture of 13 complex strategies, and 2 simple strategies that are quick to turn to defection forever.

The distributions of the normalised ranks of the top 15 strategies is given in Figure1. Overall there is no much variation in the distributions, there mainly tailed closer to 0. Thus all these strategoties performed well at each given tournament that the have been involved. In the later session that other tournaments are explored we will see that this is true only to standard tournaments. With stochastic uncearta such as noise and probabilistic ending there is more variation in the performances involved.

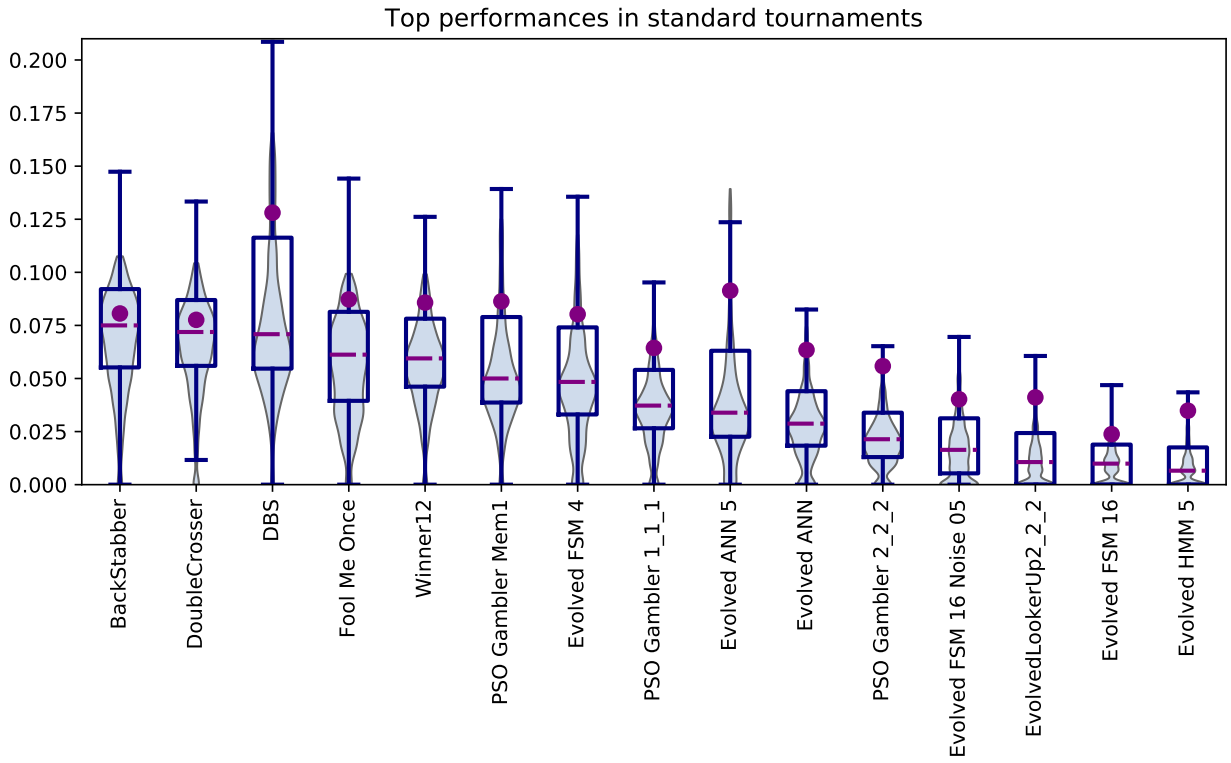


Figure 1: P.D.

4.0.2 Noisy Tournaments

The noisy tournament top 14 performances, based on $\bar{R}$ are given by Table 5. The distributions of their R is also given by Figure 1.

4.0.3 Probabilistic ending Tournaments

4.0.4 Noisy & Probabilistic ending Tournaments

4.0.5 Overall Tournaments

5 Evaluation of performance

6 Conclusion and Discussion

References

[1] The Axelrod project developers . Axelrod: 3.0.0, April 2016.

Name	$\bar{R}$ in noise tournaments
Grumpy	0.139535
e	0.190476
Cooperator	0.195652
Tit For 2 Tats	0.205196
Cycle Hunter	0.222222
Risky QLearner	0.224242
Retaliate 3	0.230769
Retaliate 2	0.237624
Retaliate	0.243094
Hard Tit For 2 Tats	0.246575
Limited Retaliate 3	0.250000
ShortMem	0.252720
Limited Retaliate	0.256983
Limited Retaliate 2	0.260274
ϕ	0.262008

Table 5: Noisy top performances

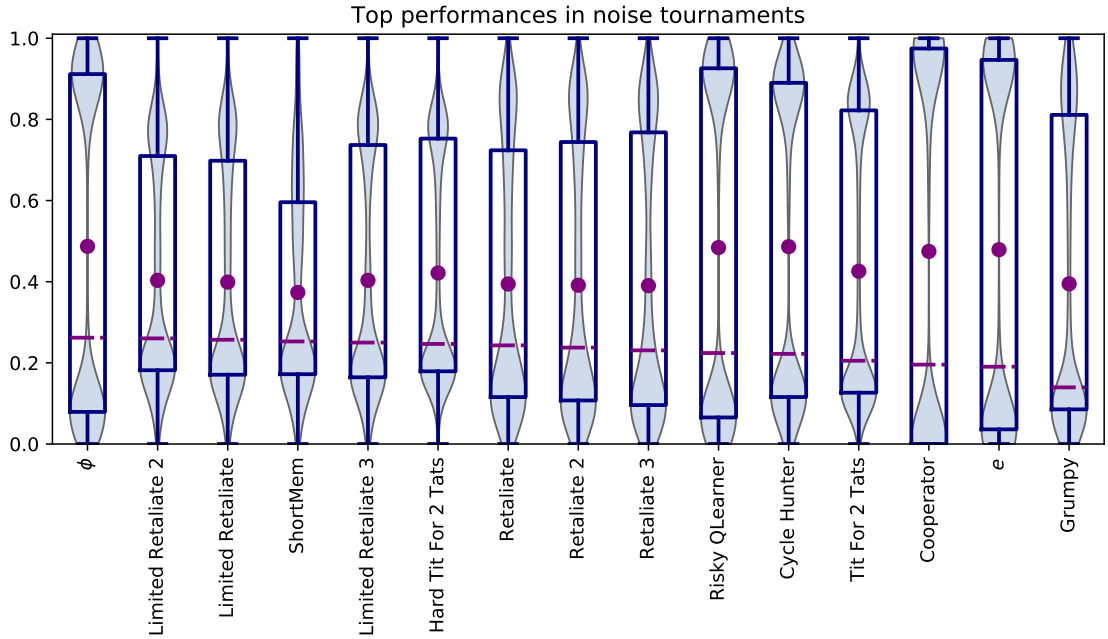
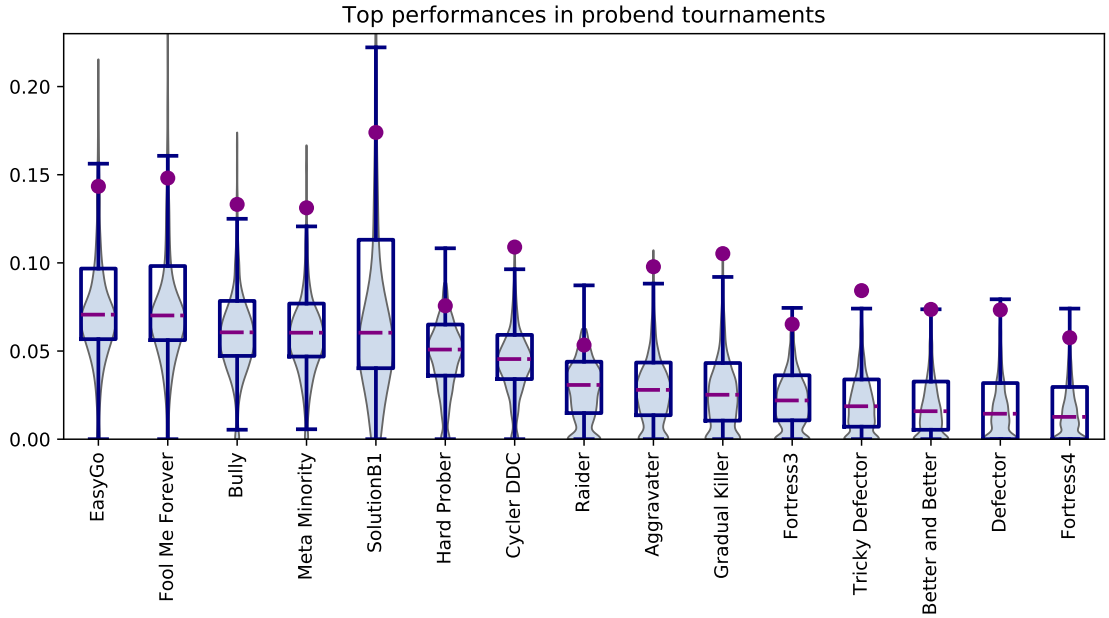
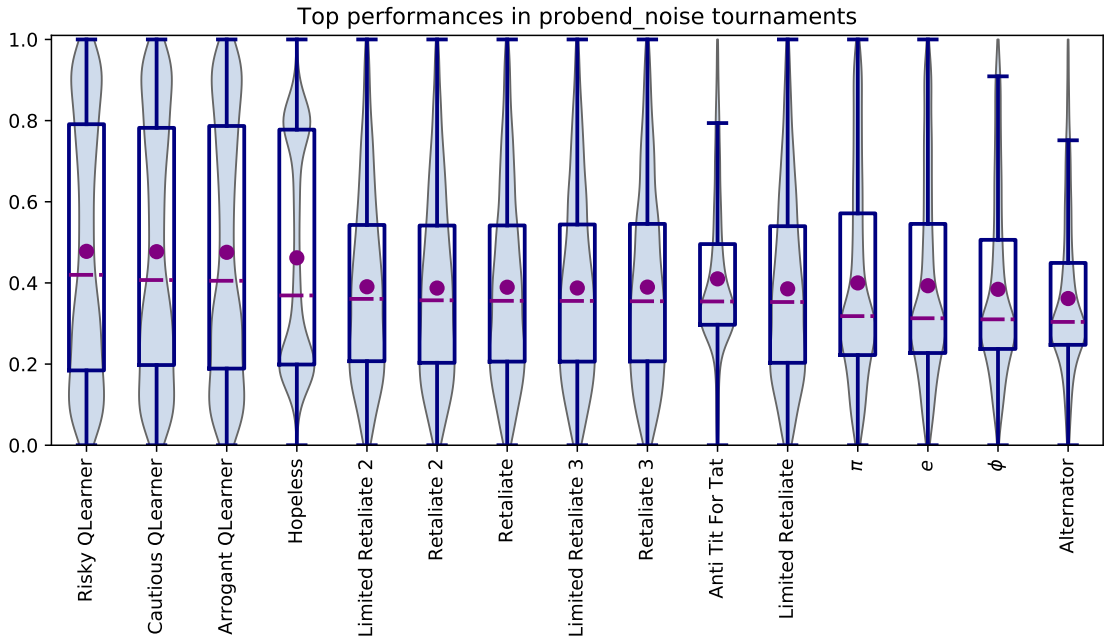


Figure 2: R distributions for best performed strategies in noisy tournaments.

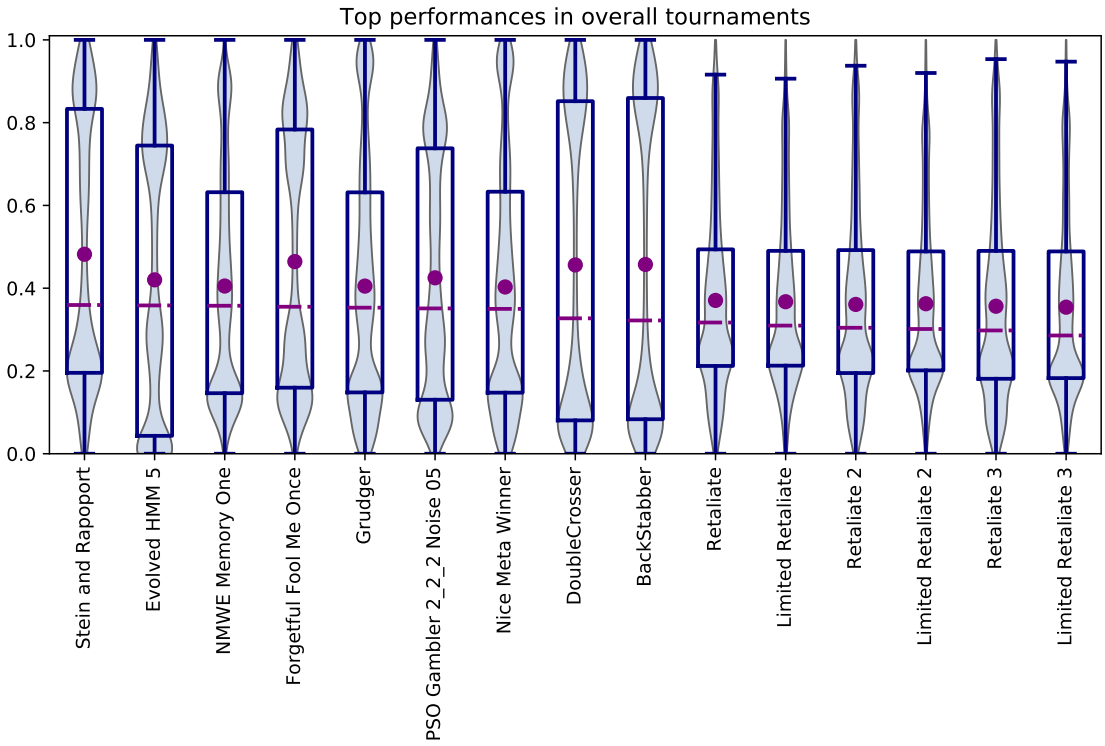
Name	$\bar{R}$ in probend tournaments
Fortress4	0.012658
Defector	0.014441
Better and Better	0.015873
Tricky Defector	0.018692
Fortress3	0.021978
Gradual Killer	0.025210
Aggravater	0.027972
Raider	0.030769
Cycler DDC	0.045455
Hard Prober	0.050847
SolutionB1	0.060403
Meta Minority	0.060403
Bully	0.060606
Fool Me Forever	0.070175
EasyGo	0.070652

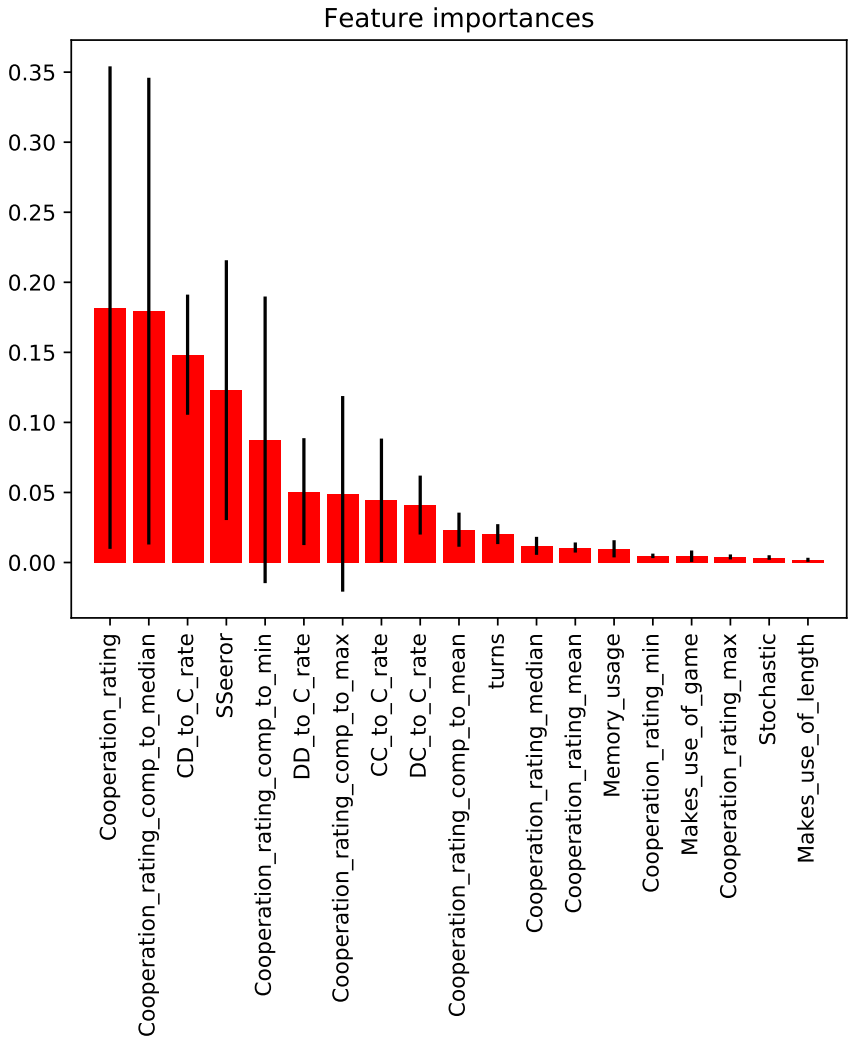


Name	$\bar{R}$ in probend_noise tournaments
Alternator	0.303899
ϕ	0.310253
e	0.312925
π	0.318182
Limited Retaliate	0.352941
Anti Tit For Tat	0.354286
Retaliate 3	0.354839
Limited Retaliate 3	0.355630
Retaliate	0.355880
Retaliate 2	0.357143
Limited Retaliate 2	0.360656
Hopeless	0.369128
Arrogant QLearner	0.405263
Cautious QLearner	0.407111
Risky QLearner	0.419890



Name	Normalised rank overall
Limited Retaliate 3	0.285714
Retaliate 3	0.297872
Limited Retaliate 2	0.301370
Retaliate 2	0.304348
Limited Retaliate	0.309629
Retaliate	0.317073
BackStabber	0.322034
DoubleCrosser	0.327188
Nice Meta Winner	0.350000
PSO Gambler 2_2_2 Noise 05	0.351104
Grudger	0.352941
Forgetful Fool Me Once	0.355140
NMWE Memory One	0.357576
Evolved HMM 5	0.358333
Stein and Rapoport	0.359375




Feature importances
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Cooperation_rating
Cooperation_rating_comp_to_median
CD_to_C_rate
SSeeror
Cooperation_rating_comp_to_min
DD_to_C_rate
Cooperation_rating_comp_to_max
CC_to_C_rate
DC_to_C_rate
Cooperation_rating_comp_to_mean
turns
Cooperation_rating_median
Cooperation_rating_mean
Memory_usage
Cooperation_rating_min
Makes_use_of_game
Cooperation_rating_max
Stochastic
Makes_use_of_length

[2] Tsz-Chiu Au and Dana Nau. Accident or intention: that is the question (in the noisy iterated prisoner's dilemma). In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 561–568. ACM, 2006.

[3] Marc Harper, Vincent Knight, Martin Jones, Georgios Koutsovoulos, Nikoleta E. Glynatsi, and Owen Campbell. Reinforcement learning produces dominant strategies for the iterated prisoners dilemma. *PLOS ONE*, 12(12):1–33, 12 2017.

[4] Philippe Mathieu and Jean-Paul Delahaye. New winning strategies for the iterated prisoner's dilemma. *Journal of Artificial Societies and Social Simulation*, 20(4):12, 2017.