On Voting Approaches for Robust Reinforcement Learning Ensembles

Jose Gallego, Alexandre Lasbleis, Isa Steinecker, Yujie Xing and Ulle Endriss
University of Amsterdam

Abstract

The use of ensembles methods in Machine Learning has proven to be both theoretically sound and effective in practice for solving complex problems. However, while strategies for combining the output of several artificial agents in Supervised Learning problems might be rather straightforward, the decision on how to appropriately aggregate the preferences of such agents in a Reinforcement Learning setting is not trivial. In this paper we evaluate the performance of employing Social Choice Theory as a framework for collective decision making. In particular, we assess the behavior of well known voting rules at the pole balancing task under several uncertainty conditions. Our results suggest that voting is an effective ensemble technique with a high robustness against extreme noise conditions. Besides, we are able to outperform fully-trained agents in noisy environments with no training time trade-off generated by the creation of the multiple agents. Finally, the reliability of the agents' quantitative information, e.g. learned action-value function, can have a significant impact in the performance of a chosen voting rule.[*]

1 Introduction

Consider the task of developing a system for solving a complex problem as self-driving cars. Suppose we are given agents which have been trained to solve the problem under diverse conditions, even perhaps unknown to us. For practical reasons, it might be unfeasible to test the provided agents and evaluate their performance. This is particularly relevant in real applications, in which the environment faced by the agent can be highly noisy compared to the training setting. It is natural to consider voting as a mechanism for aggregating the preferences of such agents in order to obtain overall robustness in the system.

In fact, Social Choice Theory has been extensively applied in Multi-Agent Systems [Wooldridge, 2009]. However, considerations on how to choose an adequate voting strategy are still scarce.

[*]Source code for this project: `github.com/jgalle29/rl_voting`

Earlier work by Hans and Udluft [2010] investigated the influence of weighted plurality voting on performance in the pole balancing problem. According to their findings, this method proved to be more robust and reliable than action-value function averaging, in particular when using agents with different training sets. However, in their set up, the aggregation was performed using only weighted plurality with three candidates.

In this paper we not only show that voting provides a high-performance robust strategy for creating ensembles of Reinforcement Learning agents with no training time trade-off, but also investigate on the influence of the properties of the selected voting rule in the performance of the joint system.

The rest of this paper is structured as follows: in section 2 we describe the methodology used to conduct our experiments. Section 3 presents our results and analysis regarding the differences in performance of the several voting strategies. Finally, section 4 contains our main conclusions and possible directions for future research.

2 Methodology

In this section we provide some background information for the techniques and experiments used in our work.

2.1 Environment

The task we are aiming to solve is the pole-balancing problem in which the goal is to keep a pole balanced on a cart that moves in a one dimensional space. This is achieved by applying forces to the cart within the range $[-10 : 2 : 10]$ newtons based on information about the position, angle, velocity and angular velocity. A vector comprising these four variables represents a state of the system.

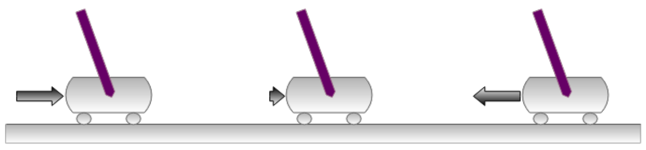

Figure 1: Pole balancing problem: apply a force to keep the pole balanced as long as possible

At the start of each run the state is initialized randomly. Then, at each time step the agent chooses a force to apply and receives a reward of 1 if the pole stays in an upright position within some threshold angle and the cart is inside the track, see Figure 1. A run ends if either of the previous conditions is not satisfied or the agent has maintained the pole balanced for 500 time steps (thus achieving a reward of 500). The lapse between the start and end of a run is called an episode.

2.2 Reinforcement Learning

Reinforcement Learning (RL) is a sub-field of Machine Learning in which an agent learns optimal behavior in a stochastic environment in order to maximize a reward signal. This can be formally described by a Markov Decision Process (MDP), which is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:

- $\mathcal{S}$ is a finite set of states,
- $\mathcal{A}$ is a finite set of actions,
- $\mathcal{P}(s, a, s')$ is a state transition probability function for ending in state s' after having chosen action a while being in state s,
- $\mathcal{R}(s, a)$ describes the expected immediate reward of executing action a in state s,
- γ is a discount factor $\gamma \in [0, 1]$, which represents the importance of long-term goals for the agent

The aim is to learn a action-value function $\mathcal{Q}(s, a)$ which describes what is the long term quality of choosing an action at a particular state. Then, use this function to define the policy for how to react in each situation:

$$\pi(s) = \text{argmax}_{a \in \mathcal{A}} \mathcal{Q}(s, a)$$

Note that given a state s the $\mathcal{Q}$ function enriched with a tie-breaking strategy induces a linear order on the space of actions for the agent.

In our approach we used a Policy-Gradient method for the training of the agents, heavily based on the work by Juliani [2016]. For a complete description of the Reinforcement Learning problem and associated solution techniques, we refer the reader to [Sutton and Barto, 1998].

2.3 Voting Rules

In this study we investigated the influence of four different voting rules: Plurality, Borda, Copeland and Range Voting. These are well known rules and represent the largest classes of voting procedures used in Social Choice Theory. Additionally, it is important to consider rules that do not introduce a large overhead in the winner determination, specially in cases in which the action selection time might be critical. A more detailed description on the properties of these rules can be found in [Zwicker, 2016].

We denote by m the number of alternatives (actions) and n the number of voters (agents).

Plurality: Each candidate receives a point for every time it is ranked first.

Borda: Each alternative is ranked according to the agents preference. The top ranked candidate is assigned $m-1$ points the second $m-2$ this continues until the least ranked alternative is assigned 0 points.

Copeland: Each candidate receives a point for every won pairwise majority contest, and loses one point otherwise.

Range Voting: Each voter has the same fixed budget that can be distributed among the candidates in accordance with the voters preference. In this study the budget was set to 100 points (hence we also refer to Range Voting as 100 points).

Finally, we turn all these rules in to social welfare functions (SWF) by ranking the candidates according to their accumulated score. Note that this might lead to ties between two or more candidates. In our experiments ties are broken randomly. However, there might exist a domain-specific information on how to break ties appropriately.

2.4 Pipeline

We defined our agents as Neural Networks (NN) since the state representation contains continuous variables. These networks were trained as mention in Section 2.2 on Reinforcement Learning. In order to create diversity among our agents, we used different architectures for the hidden layers: $[3, 5, 10, 3-3, 5-5]$. We trained 2 agents for each architecure. The numbering in the result section coincides with the positions in the previous array mod 5. All our networks had a 4 dimensional input and 11 dimensional output, corresponding to dimension of the state and action spaces.

Since our proposed methodology requires several agents for running an election, we used agents which were only partially trained in order to avoid the introduction of a training time trade off. Specifically, we observed that fully trained agents (which consistently achieved reward 500 per episode in an environment with no noise) required a training of around 10.000 episodes. For this reason, we trained all of our 10 semi-trained agents for up to 1.000 episodes.

Figure 2 displays the pipeline for the execution of the ensemble system. At every time step the environment provides a state vector which is affected by noise for each of the agents. Based on this blurred vision of the environment, each agent ranks the possible actions according to its learned $\mathcal{Q}$ function and casts a ballot given by this ranking. A profile is then formed by the collection of the ballots. Finally, the election is held and the winner of the election is the action chosen to be executed next.

2.5 Statistical Measures

Now we describe the statistical measures we applied for comparing the similarity between the chosen SWFs:

Top 1 Match: Estimates the proportion of how often two SWFs coincide on the chosen winner given the same profile.

Kendalls τ [1938]: Measures the similarity between two rankings by comparing the number of agreements and disagreements.

Spearman ρ [1904]: Similar to Kendall's tau, but further takes into account the magnitude of the disagreements, e.g. swapping the first and last positions has a larger magnitude than changing two adjacent elements.

For both of the previous coefficients, values close to 1 indicate strong agreement, while values close to -1 indicate strong disagreement between the compared rankings.

t-Test [Fisher, 1925]: Evaluates whether there are significant differences between the performance of two SWFs in terms of their average reward per episode. All our test were conducted under a 99% confidence level.

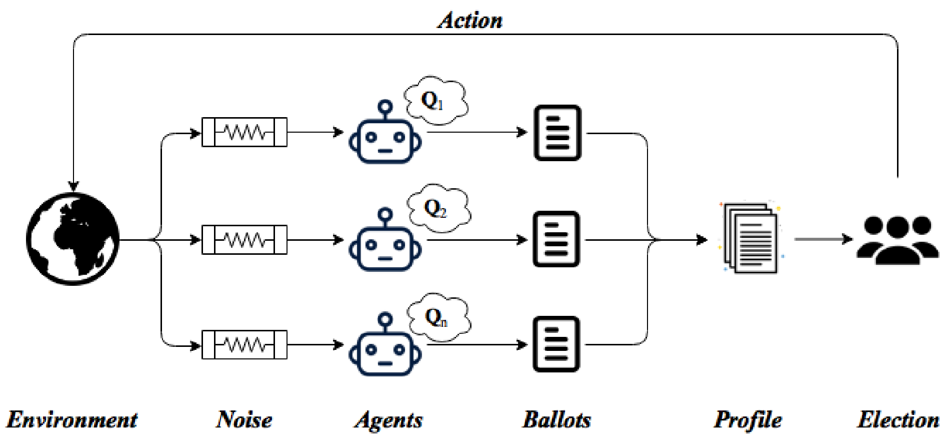

Figure 2: Flow diagram for the proposed voting ensemble mechanism

2.6 Experiments

The agents were tested in the environment described in section 2.2 and their reward per episode was monitored. We display cumulative reward over episodes as it eases the visual analysis of the different agents.

Different uncertainty conditions were created by adding zero-centered Gaussian noise. The covariance matrix of the noise is a scaled version of the estimated covariance matrix of the state. We run our experiments with the following scale factors: 0, 2, 10, 20. These values simulate the performance under no noise (0Σ), extreme noise (20Σ) and further allow us to assess what happens in between those extreme cases.

These conditions are meant to represent real situations in which, for example, the sensors used by the self-driving car can be noisy and can return inaccurate information at every step. Under each of the 4 types of noise the single performance of a semi-trained agent, full-trained agents as well as the ensemble were tested.

3 Results

For improved readability, the reward values presented in the rest of this paper are compressed by a factor of 1.000. For example, the optimal cumulative reward over 100 episodes is now presented as 50 instead of 50.000. Additionally, we draw the attention of the reader to differences in the scales of the cumulative reward axis while comparing plots.

First we will evaluate case in which no noise is added. In Figure 3 the individual performances of the agents are displayed. It can be seen that the most complex network (NN9) has a high performance in comparison to the other individual agents, close to 30. Note that the single performance of an agent is equivalent to the performance of a dictatorial voting rule in which the mentioned agent acts as the dictator.

The average of the score of all agents is around 9.5. This performance could be achieved by an ensemble in which one agent is randomly selected at each episode.

One can observe in Figure 4 the performance of the SWFs. This indicates that Borda and Copeland clearly outperform Plurality and Range Voting. This is confirmed by the p-values shown in Table 4 in the Appendix.

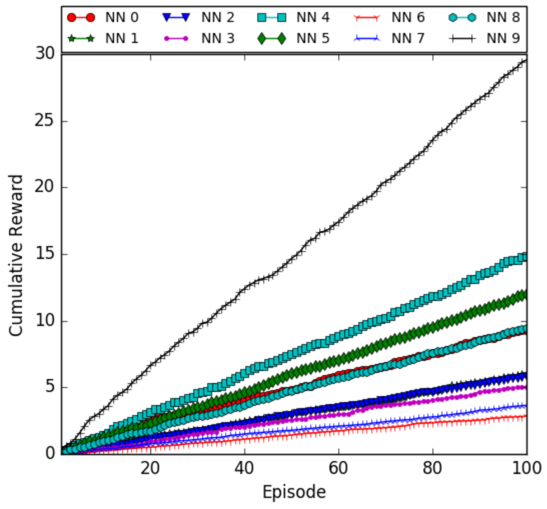

Figure 3: Individual agent performance under no noise

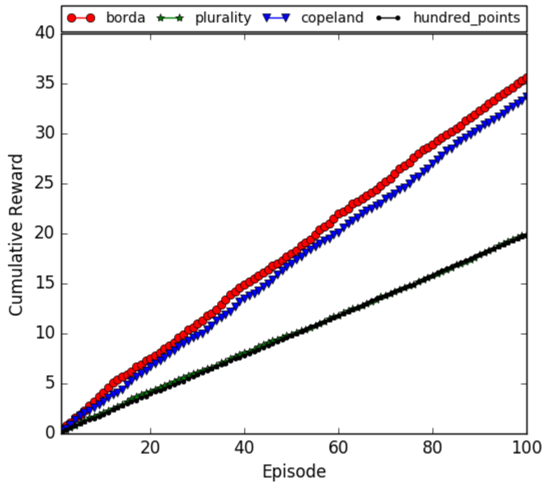

Figure 4: Voting rule performance under no noise

First observe that even though Plurality and Range Voting do not achieve a score as high as Borda and Copeland, the former still improve the results compared to the average of the NNs. Additionally, we notice that the cumulative rewards for Borda and Copeland are greater than the one achieved by the best neural network by around 10%, and 3.5 times the average performance among the NNs.

As the results for the noise settings are similar to each other, we will only discuss the extreme case 20Σ. We refer the interested reader to the graphs in the Appendix.

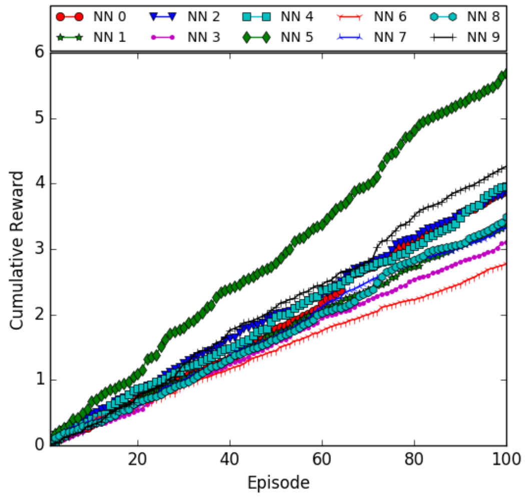

Figure 5: Individual agent performance under 20Σ noise

Figure 5 displays the cumulative reward for the individual performance under noise. Notice that NN9, which had a clear advantage in the no-noise case had a dramatic drop (ca. 90%). Now, network NN5, with the simplest architecture, has the best individual performance. This indicates that the NN9 agent might have overfitted to the training environment. Observe that the variance of the performances is smaller than before and the average cumulative reward is around 3.5 (roughly a 60% decrease compared to Figure 3).

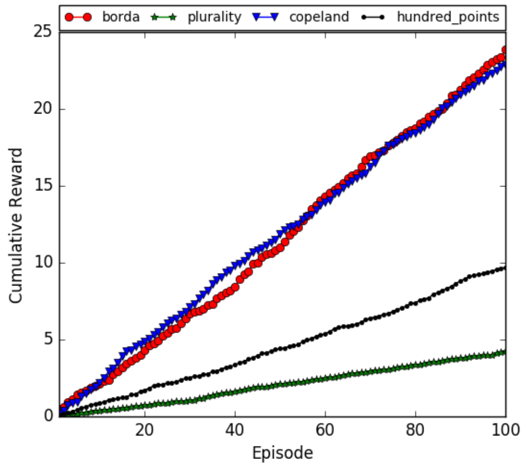

Figure 6: Voting rule performance under 20Σ noise

In Figure 6 the performance of the SWFs under noise is presented. The similarity in the performance of Borda and Copeland remains with both achieving a reward of around 23, while Range Voting improved in comparison to Plurality. Nevertheless, there is still a clear dominance of Borda and Copeland over the other two rules. Borda and Copeland experienced a decrease in performance of 32%, but still outperform the average of the NNs by a factor of 4.3.

4 Discussion

To assess the similarity between the SWFs we applied the methods described in Section 2.5. See Appendix for full results.

The Top 1 Match for Borda and Copeland under all noise conditions is above 73%, hence their top ranked actions often coincide, which accounts for the similarity in the cumulative reward. The τ and ρ statistics are higher than any other pair of rules and indicate that there is a overall similarity in the collective rankings generated by these two rules, beyond the Top 1 Match. Finally, under the selected confidence level it is never possible to conclude that there is a significant difference in the performance of these two rules based on their average reward per episode.

We would like to remark that even though our experiments do not exhibit significant differences between these two rules, the computational complexity of Borda might be an important advantage against Copeland in settings in which the number of the actions or voters is large.

An important factor in the analysis of the performance of the SWFs is the amount and type of information each of them requires. For instance, the Plurality rules is simplistic in the sense that only the top ranked candidate is taken into account for the winner determination; while Borda and Copeland requires ballots with a complete linear order for each of the voters. In the opposite extreme of Plurality we locate Range Voting for which it is not only necessary to provide a ranking of the candidates, but also there is a quantitative characterization of how much a candidate is preferred over another. This provides Range Voting with a large expressive value at the expense of the reliability of the numbers on which the calculation of the winner is based.

This claim is justified by an additional experiment were we used fully trained agents (which achieve full reward in no-noise conditions). In Figure 7 it is possible to observe that the average performance of these agents under extreme noise conditions is around 6 (rougly twice as good as the individual semi-trained agents in Figure 5). There is not a significant difference between the usage of Borda or Copeland with fully or semi trained agents, as shown in Figure 8. This results allow us to conclude that it is not necessary to have fully trained agents for achieving high rewards under extreme noise conditions. On the other hand, the fact that these agents are fully trained suggests that their corresponding learned Q functions are more reliable, which explains why Range Voting achieves a performance which is much similar to the one of Borda and Copeland.

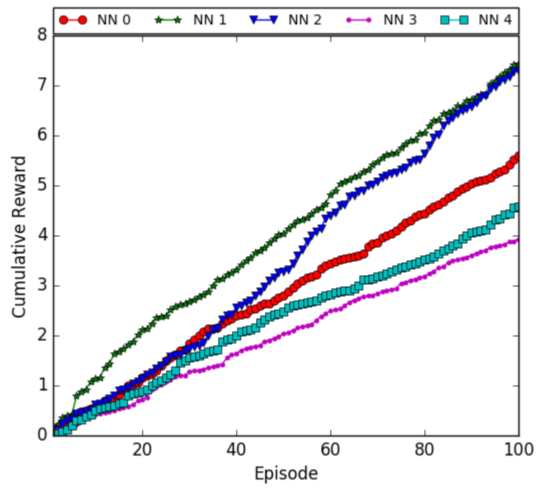

Figure 7: Individual performance of fully trained agents under 20**Σ** noise

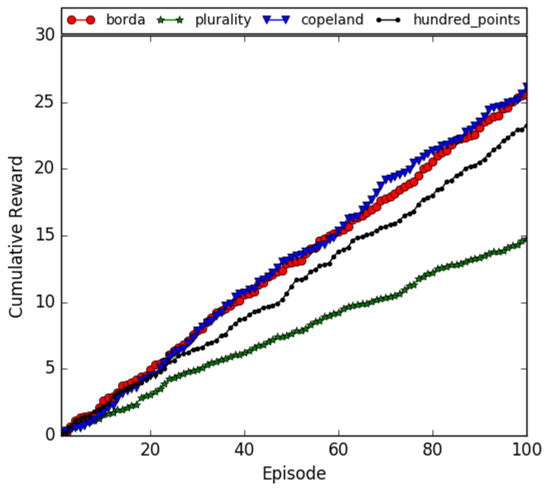

Figure 8: Voting rule performance for fully trained agents under 20**Σ** noise

5 Conclusions and Future Work

Our results suggest that voting is an effective ensemble technique with a high robustness against extreme noise conditions. We are able to outperform fully-trained agents in noisy environments. Besides, our approach does not imply a training time trade-off due to the creation of the multiple agents.

The amount and nature of the information used by a voting rule, as well as the reliability of the information itself, can have a dramatic impact in performance. In particular, too simplistic voting rules, such as Plurality and Dictatorship, perform poorly under noise. Rules like Borda and Copeland, which rely on a richer information type of ballots usually outperform Range Voting, which is directly based on quantitative information given by the agents. However, less uncertainty on the learned action-value functions reduces the gap between the top performing rules.

Further efforts could be focused on the application of this framework to more challenging Reinforcement Learning tasks or the analysis of the impact of the dimensionality of the spaces of voters and alternatives. This latter case could be important for understanding the role of the tie-breaking strategy in the performance. Finally, another open area is an accurate representation of the noise sources in the system that match real world conditions.

References

[Fisher, 1925] Ronald Aylmer Fisher. *Statistical methods for research workers*. Genesis Publishing Pvt Ltd, 1925.

[Hans and Udluft, 2010] Alexander Hans and Steffen Udluft. Ensembles of Neural Networks for Robust Reinforcement Learning. *Machine Learning and Applications (ICMLA), 2010 Ninth International Conference*, pages 401–406, 2010.

[Juliani, 2016] Arthur Juliani. Policy-based agents, 2016. Available at `medium.com/@awjuliani/super-simple-reinforcement-learning-tutorial-part-2-ded33892c724`. Accessed on 9 May 2017.

[Kendall, 1938] Maurice Kendall. A New Measure of Rank Correlation. *Biometrika*, 30(1/2):81–93, 1938.

[Spearman, 1904] Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72, 1904.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1 edition, 1998.

[Wooldridge, 2009] Michael Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2009.

[Zwicker, 2016] William S. Zwicker. Introduction to the theory of voting. In Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, editors, *Handbook of Computational Social Choice*, pages 23–56. Cambridge University Press, 1 edition, 2016.

A Appendix

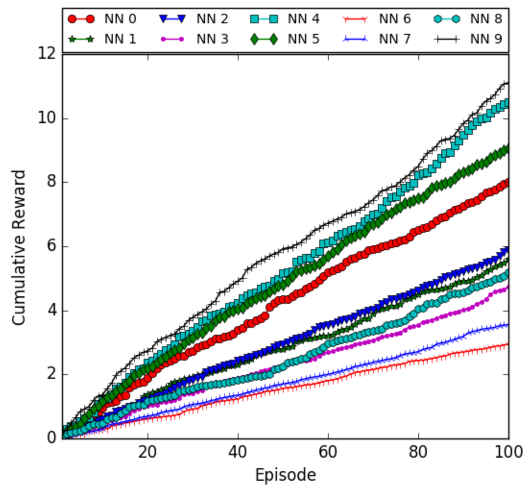

Figure 9: Individual agent performance under 2**Σ** noise

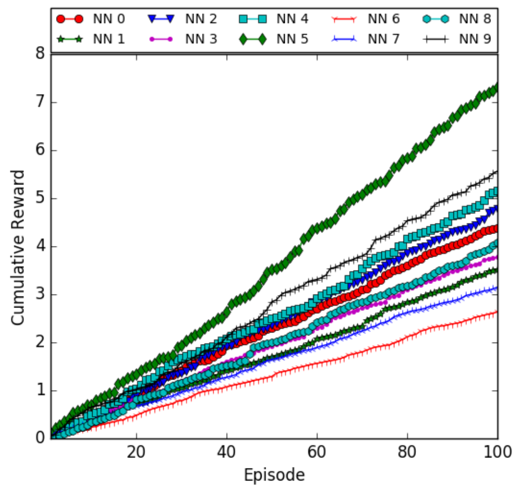

Figure 11: Individual agent performance under 10**Σ** noise

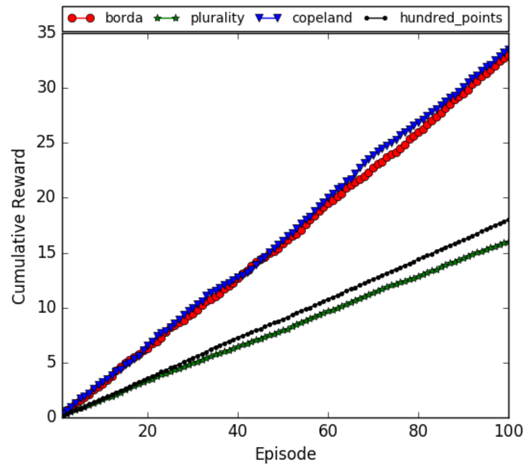

Figure 10: Voting rule performance under 2**Σ** noise

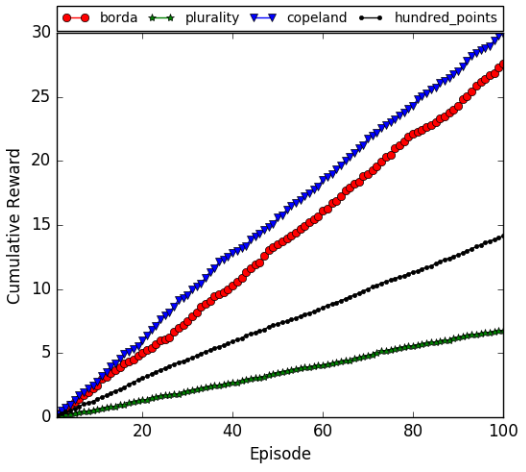

Figure 12: Voting rule performance under 10**Σ** noise

Noise	**Top 1 Match**	Borda	Plurality	Copeland	Range Voting
0Σ	Borda	1	0.239	0.779	0.188
	Plurality	0.357	1	0.398	0.600
	Copeland	0.758	0.277	1	0.260
	Range Voting	0.453	0.526	0.462	1
2Σ	Borda	1	0.296	0.743	0.315
	Plurality	0.308	1	0.336	0.478
	Copeland	0.742	0.326	1	0.343
	Range Voting	0.347	0.490	0.377	1
10Σ	Borda	1	0.293	0.742	0.313
	Plurality	0.306	1	0.335	0.482
	Copeland	0.738	0.323	1	0.341
	Range Voting	0.341	0.481	0.364	1
20Σ	Borda	1	0.295	0.737	0.312
	Plurality	0.330	1	0.351	0.499
	Copeland	0.742	0.314	1	0.331
	Range Voting	0.332	0.492	0.353	1

Table 1: Top 1 Match for several noise

Noise	τ	Borda	Plurality	Copeland	Range Voting
0Σ	Borda	1	0.097	0.373	0.222
	Plurality	0.195	1	0.202	0.233
	Copeland	0.402	0.114	1	0.253
	Range Voting	0.419	0.220	0.394	1
2Σ	Borda	1	0.163	0.448	0.246
	Plurality	0.145	1	0.145	0.166
	Copeland	0.444	0.120	1	0.244
	Range Voting	0.282	0.153	0.278	1
10Σ	Borda	1	0.131	0.458	0.282
	Plurality	0.168	1	0.171	0.185
	Copeland	0.457	0.133	1	0.274
	Range Voting	0.327	0.177	0.317	1
20Σ	Borda	1	0.119	0.444	0.237
	Plurality	0.127	1	0.131	0.152
	Copeland	0.444	0.116	1	0.229
	Range Voting	0.260	0.147	0.261	1

Table 2: τ between rules for several noise

Noise	ρ	Borda	Plurality	Copeland	Range Voting
0$\mathbf{\Sigma}$	Borda	1	0.016	0.547	0.320
	Plurality	0.198	1	0.199	0.224
	Copeland	0.543	0.164	1	0.319
	Range Voting	0.366	0.208	0.360	1
2$\mathbf{\Sigma}$	Borda	1	0.178	0.559	0.365
	Plurality	0.228	1	0.233	0.250
	Copeland	0.557	0.181	1	0.354
	Range Voting	0.420	0.239	0.407	1
10$\mathbf{\Sigma}$	Borda	1	0.163	0.543	0.320
	Plurality	0.198	1	0.199	0.224
	Copeland	0.543	0.164	1	0.244
	Range Voting	0.366	0.208	0.360	1
20$\mathbf{\Sigma}$	Borda	1	0.163	0.543	0.310
	Plurality	0.174	1	0.180	0.206
	Copeland	0.542	0.158	1	0.300
	Range Voting	0.340	0.198	0.340	1

Table 3: ρ between rules for several noise

p-Values		0$\mathbf{\Sigma}$	2$\mathbf{\Sigma}$	10$\mathbf{\Sigma}$	20$\mathbf{\Sigma}$
Borda	Plurality	0.00	0.00	0.00	0.00
Borda	Copeland	0.09	0.70	0.10	0.48
Borda	Range Voting	0.00	0.00	0.00	0.00
Plurality	Copeland	0.00	0.00	0.00	0.00
Plurality	Range Voting	0.78	0.00	0.00	0.00
Copeland	Range Voting	0.00	0.00	0.00	0.00

Table 4: P-values for test of difference between average reward per episode among several rules