Every Team Deserves a Second Chance: Identifying when Things Go Wrong

Vaishnavh Nagarajan[1*], Leandro Soriano Marcolino[2*], Milind Tambe[2]
[1] Indian Institute of Technology Madras, Chennai, Tamil Nadu, 600036, India
vaish@cse.iitm.ac.in
[2] University of Southern California, Los Angeles, CA, 90089, USA
{sorianom, tambe}@usc.edu

ABSTRACT

Voting among different agents is a powerful tool in problem solving, and it has been widely applied to improve the performance in finding the correct answer to complex problems. We present a novel benefit of voting, that has not been observed before: we can use the voting patterns to assess the performance of a team and predict their final outcome. This prediction can be executed at any moment during problem-solving and it is completely domain independent. We present a theoretical explanation of why our prediction method works. Further, contrary to what would be expected based on a simpler explanation using classical voting models, we argue that we can make accurate predictions irrespective of the strength (i.e., performance) of the teams, and that in fact, the prediction can work better for diverse teams composed of different agents than uniform teams made of copies of the best agent. We perform experiments in the Computer Go domain, where we obtain a high accuracy in predicting the final outcome of the games. We analyze the prediction accuracy for three different teams with different levels of diversity and strength, and we show that the prediction works significantly better for a diverse team. Since our approach is domain independent, it can be easily applied to a variety of domains.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems

General Terms

Algorithms, Experimentation, Theory

Keywords

Teamwork, Collective intelligence, Distributed problem solving, Social choice theory, Single and multiagent learning

1. INTRODUCTION

It is well known that aggregating the opinions of different agents can lead to a great performance when solving complex problems. For example, voting has been extensively used to improve the performance in machine learning [33], crowdsourcing [28, 1], and even board games [30, 32]. Besides, it is an aggregation technique that does not depend on any domain, being very suited for wide applicability. However, a team of voting agents will not always be successful in problem-solving. It is fundamental, therefore, to be able to quickly assess the performance of teams, so that a system operator can take actions to recover the situation in time.

Current works in the multi-agent system literature focus on identifying faulty or erroneous behavior [22, 25, 38, 5], or verifying correctness of systems [8]. Such approaches are able to identify if a system is not correct, but provide no help if a correct system of agents is failing to solve a complex problem. Other works focus on team analysis. Raines et al. (2000) [34] present a method to automatically analyze the performance of a team. The method, however, only works offline and needs domain knowledge. Other methods for team analysis are heavily tailored for robot-soccer [35] and focus on identifying opponent tactics [31].

In fact, many works in robotics propose monitoring a team by detecting differences in the internal state of the agents (or disagreements), mostly caused by malfunction of the sensors/actuators [21, 20, 18, 19]. In a system of voting agents, however, disagreements are inherent in the coordination process and do not necessarily mean that an erroneous situation has occurred due to such malfunction. Meanwhile, the works in social choice are mostly focused on studying the guarantees of finding the optimal choice given a noise model for the agents and a voting rule [6, 26, 7], but provide no help in assessing the performance of a team of voting agents.

In this paper, we show a novel method to predict the final performance (success or failure) of a team of voting agents, without using any domain knowledge. Hence, our method can be easily applied in a great variety of scenarios. Moreover, our approach can be quickly applied online at any step of the problem-solving process, allowing a system operator to identify when the team is failing. This can be fundamental in many applications. For example, consider a complex problem being solved in a cluster of computers. It is undesirable to allocate more resources than necessary, but if we notice that a team is failing in problem solving, we can increase the allocation of resources. Or consider a team playing together a game against an opponent (such as board games, or poker). Different teams might play better against different opponents. Hence, if we notice that a team is having issues, we could dynamically change it. Under time constraints, however, such prediction must be done quickly.

Our approach is based on a prediction model derived from a graphical representation of the problem-solving process, where the final outcome is modeled as a random variable that is influenced by the subsets of agents that agreed together over the actions taken at each step towards solving the problem. Hence, our representation depends uniquely on the coordination method, and has no dependency on the domain. We explain theoretically why we can make

*Vaishnavh Nagarajan and Leandro Soriano Marcolino are both first authors of this paper.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

accurate predictions, and we also show the conditions under which we can use a reduced (and scalable) representation. Moreover, our theoretical development allows us to expect situations that would not be foreseen by a simple application of classical voting theories. For example, our model indicates that the accuracy can be better for diverse teams composed of different agents than for uniform teams, and that we can make equally accurate predictions for teams that have significant differences in playing strength (which is later confirmed in our experiments).

We present experimental results in the Computer Go domain, where we predict the performance of three different teams of voting agents: a diverse, a uniform, and an intermediate team (with respect to diversity). We show that we can predict win/loss of Go games with around 73% accuracy for the diverse and intermediate team, and 64% for the uniform team. We also study the predictions at every turn of the games, and compare with an analysis performed by using an in-depth search. We show that our method agrees with the analysis, from around the middle of the games, more than 60% of the time for all teams, but is significantly faster.

2. RELATED WORK

Recently, researchers in artificial intelligence have been showing a great interest for developing and testing general methodologies that can handle a great variety of problems [24, 15, 13]; culminating with the recent development of the arcade learning environment, where researchers explore the applicability of machine learning techniques for a range of different games [4]. In particular, voting is a technique that can be applied in many different domains, such as: crowdsourcing [28, 1], board games [29, 30, 32], machine learning [33], forecasting systems [16], etc.

Voting is extensively studied in social choice. Normally, it is presented under two different perspectives: as a way to aggregate different opinions, or as a way to discover an optimal choice [26, 7]. In this work we present a novel perspective. We show that we can use the voting patterns as a way to assess the performance of a team. Such a "side-effect" of voting has not been observed before, and was never explored in social choice theory and/or applications.

Concerning team assessment, the traditional methods rely heavily on tailoring for specific domains. Raines et al. (2000) [34] present a method to build automated assistants for post-hoc, offline team analysis; but domain knowledge is necessary for such assistants. Other methods for team analysis are heavily tailored for robot-soccer, such as Ramos and Ayanegui (2008) [35], that present a method to identify the tactical formation of soccer teams (number of defenders, midfielders, and forwards). Mirchevska et al. (2014) [31] present a domain independent approach, but they are still focused on identifying opponent tactics, not in assessing the current performance of a team.

In the multi-agent systems community, we can see many recent works that study how to identify agents that present faulty behavior [22, 25, 38]. Other works focus on verifying correct agent implementation [8] or monitoring the violation of norms in an agent system [5]. Some works go beyond the agent-level and verify if the system as a whole conforms to a certain specification [23], or verify properties of an agent system [14]. However, a team can still have a poor performance and fail in solving a problem, even when the individual agents are correctly implemented, no agent presents faulty behavior, and the system as a whole conforms to all specifications.

Sometimes even correct agents might fail to solve a task, especially embodied agents (robots) that could suffer sensing or actuating problems. Kaminka and Tambe (1998) [21] present a method to detect clear failures in an agent team by social comparison (i.e., each agent compares its state with its peers). Such an approach is fundamentally different than this work, as we are detecting a tendency towards failure for a team of voting agents (caused, for example, by simple lack of ability, or processing power, to solve the problem), not a clearly problematic situation that could be caused by imprecision/failure of the sensors or actuators of an agent/robot. Later, Kaminka (2006) [20], and Kalech and Kaminka (2007, 2011) [18, 19] study the detection of failures by identifying disagreement among the agents. In our case, however, disagreements are inherent in the voting process. They are easy to detect but they do not necessarily mean that a team is immediately failing, or that an agent presents faulty behavior/perception of the current state.

This work is also related to multi-agent learning [40], but normally multi-agent learning methods are focused on learning how agents should perform, not on team assessment. An interesting approach has recently been presented in Torrey and Taylor (2013) [39], where they have studied how to teach an agent to behave in a way that will make it achieve a high utility. Besides teaching agents, it should also be possible to teach agent teams. During the process of teaching, it is fundamental to identify when the system is leading towards failure. Hence, our approach could be integrated within a team teaching framework.

Finally, it has recently been shown that diverse teams of voting agents are able to outperform uniform teams composed of copies of the best agent [29, 30, 17]. Here we present an extra benefit of having diverse teams: we show that we can make better predictions of the final performance for diverse teams than for uniform teams.

3. PREDICTION METHOD

We start by presenting our prediction method, and in Section 4 we will explain why the method works.

We consider scenarios where agents vote at every step (i.e., world state) of a complex problem, in order to take common decisions at every step towards problem-solving. Formally, let $\mathbf{T}$ be a set of agents t_i, $\mathbf{A}$ be a set of actions a_j and $\mathbf{S}$ be a set of world states s_k. The agents must vote for an action at each world state, and the team takes the action decided by the *plurality voting rule*, that picks the action that received the highest number of votes (we assume ties are broken randomly). The team obtains a final reward r upon completing all world states. In this paper, we assume two possible final rewards: "success" (1) or "failure" (0).

We define the prediction problem as follows: without using any knowledge of the domain, identify the final reward that will be received by a team. This prediction must be executable at any world state, allowing a system operator to take remedy procedures in time.

We now explain our algorithm. The main idea is to learn a prediction function, given the frequencies of agreements of all possible agent subsets over the chosen actions. Let $\mathcal{P}(\mathbf{T}) = \{\mathbf{T_1}, \mathbf{T_2}, \ldots\}$ be the power set of the set of agents, a_i be the action chosen in world state s_j and $\mathbf{H_j} \subseteq \mathbf{T}$ be the subset of agents that agreed on a_i in that world state.

Consider the feature vector $\vec{\mathbf{x}} = (x_1, x_2, \ldots)$ computed at world state s_j, where each dimension (feature) has a one-to-one mapping with $\mathcal{P}(\mathbf{T})$. We define x_i as the *proportion* of times that the chosen action was agreed upon by the subset of agents $\mathbf{T_i}$. That is,

$$x_i = \sum_{k=1}^{|\mathbf{S_j}|} \frac{\mathbb{I}(\mathbf{H_k} = \mathbf{T_i})}{|\mathbf{S_j}|}$$

where $\mathbb{I}$ is the indicator function and $\mathbf{S_j} \subseteq \mathbf{S}$ is the set of world states from s_1 to the current world state s_j.

Hence, given a set $\tilde{\mathbf{X}}$ such that for each feature vector $\vec{\mathbf{x_t}} \in \tilde{\mathbf{X}}$ we have the associated reward r_t, we can estimate a function, $\hat{f}$,

	Agent 1	Agent 2	Agent 3
Iteration 1	a_1	a_1	a_2
Iteration 2	a_2	a_2	a_1
Iteration 3	a_1	a_2	a_2

Table 1: A simple example of voting profiles after three iterations of problem-solving.

that returns an estimated reward between 0 and 1 given an input $\vec{\mathbf{x}}$. We classify estimated rewards above 0.5 as "success", and below 0.5 as "failure". In order to *learn* the classification model, the features are computed at the final world state.

We use classification by logistic regression, which models $\hat{f}$ as

$$\hat{f}(\vec{\mathbf{x}}) = \frac{1}{1 + e^{-(\alpha + \vec{\boldsymbol{\beta}}^T \vec{\mathbf{x}})}},$$

where α and $\vec{\boldsymbol{\beta}}$ are parameters that will be learned given $\tilde{\mathbf{X}}$ and the associated rewards. While training, we eliminate two of the features. The feature corresponding to the subset $\emptyset$ is dropped because an action is chosen only if at least one of the agents voted for it. Also, since the rest of the features sum up to 1, and are hence linearly dependent, we also drop the feature corresponding to all agents agreeing on the chosen action.

We also study a variant of this prediction method, where we use only information about the number of agents that agreed upon the chosen action, but not which agents exactly were involved in the agreement. For that variant, we consider a reduced feature vector $\vec{\mathbf{y}} = (y_1, y_2, \ldots)$, where we define y_i to be the proportion of times that the chosen action was agreed upon by any subset of i agents:

$$y_i = \sum_{k=1}^{|\mathbf{S_j}|} \frac{\mathbb{I}(|\mathbf{H_k}| = i)}{|\mathbf{S_j}|},$$

where $\mathbb{I}$ is the indicator function and $\mathbf{S_j} \subseteq \mathbf{S}$ is the set of world states from s_1 to the current world state s_j. We compare the two approaches in Section 5.

3.1 Example of Features

We give a simple example of our proposed feature vectors. Consider a team of three agents: t_1, t_2, t_3. Let's assume two possible actions: a_1, a_2. Consider that, in three iterations of the problem solving, the voting profiles were as shown in Table 1, where we show which action each agent voted for at each iteration. Based on plurality voting rule, the action chosen for the respective iterations would be a_1, a_2, and a_2.

We can see an example of how the full feature vector will be defined at each iteration in Table 2, where each column represents a possible subset of the set of agents, and we mark the frequency that each subset agreed on the chosen action.

In Table 3, we show an example of the reduced feature vector, where the column headings define the number of agents involved in an agreement over the chosen action. Note that the reduced representation is much more compact, but we have no way to represent the change in which specific agents were involved in the agreement, from iteration 2 to iteration 3.

4. THEORY

We consider here the view of social choice as a way to estimate a "truth", or the correct action to perform in a given world state. Hence, we can model each agent as a pdf: that is, given the correct outcome, each agent will have a certain probability of voting for the best action, and a certain probability of voting for some incorrect

	$\{t_1\}$	$\{t_2\}$	$\{t_3\}$	$\{t_1, t_2\}$	$\{t_1, t_3\}$	$\{t_2, t_3\}$
Iteration 1	0	0	0	1	0	0
Iteration 2	0	0	0	1	0	0
Iteration 3	0	0	0	2/3	0	1/3

Table 2: Example of the full feature vector after three iterations of problem solving.

	1	2
Iteration 1	0	1
Iteration 2	0	1
Iteration 3	0	1

Table 3: Example of the reduced feature vector after three iterations of problem solving.

action. These pdfs are not necessarily the same across different world states [29]. Hence, given the voting profile in a certain world state, there will be a probability p of picking the correct choice (by the plurality voting rule).

We will start by showing some examples and observations, based on classical voting theories, to give an intuitive idea of why we can use the voting patterns to predict success or failure of a team of voting agents. However, these are not enough for a deeper understanding of the prediction methodology, and fail to explain some of the results in Section 5. Hence, we will later present our main theoretical model, that provides a better understanding of our results.

4.1 Classical Voting Model

We start with a simple example to show that we can use the outcome of plurality voting to predict success. Consider a scenario with two agents and two possible actions, a correct and an incorrect one. We assume, for this example, that agents have a probability of 0.6 of voting for the correct action and 0.4 of making a mistake.

If both agents vote for the same action, they are either both correct or both wrong. Hence, the probability of the team being correct is given by $0.6^2/(0.6^2 + 0.4^2) = 0.69$. Therefore, if the agents agree, the team is more likely correct than wrong. If they vote for different actions, however, one will be correct and the other one wrong. Given that profile, and assuming that we break ties randomly, the team will have a 0.5 probability of being correct. Hence, the team has a higher probability of taking a correct choice when the agents agree than when they disagree ($0.69 > 0.5$). Therefore, if across multiple iterations these agents agree often, the team has a higher probability of being correct across these iterations, and we can predict that the team is going to be successful. If they disagree often, then the probability of being correct across the iterations is lower, and we can predict that the team will not be successful.

More generally, we can consider all cases where plurality is the optimal voting rule. In social choice, optimal voting rules are often studied as maximum likelihood estimators (MLE) of the correct choice [7]. That is, given a voting profile and a noise model (the probability of voting for each action, given the correct outcome) of each agent, we can estimate the likelihood of each action being the best. Plurality is going to be optimal if it corresponds to always picking the action that has the maximum likelihood, according to the noise model of the agents.

In the following observation, we show that if plurality is the optimal voting rule (MLE), we can use the amount of agreement among the agents to predict success.

OBSERVATION 1. *The probability that a team is correct increases with the number of agreeing agents m in a voting profile.*

PROOF. Let a^* be the best action (whose identity we do not know). Let $v_1, v_2, \ldots v_n$ be the votes of n agents. Let w be the action chosen by the highest number of agents. We want to know the probability of $a^* = w$:

$$P(a^* = w|v_1, v_2 \ldots v_n) \propto P(v_1, v_2 \ldots v_n|a^* = w)P(a^* = w)$$

For any noise model where plurality is MLE, we have that $P(v_1, v_2 \ldots v_n|a^* = w)$ is proportional to the number of agents m that voted for a^*. Therefore, we have that $P(a^* = w|v_1, v_2 \ldots v_n)$ is also proportional to m.

Hence, given two voting profiles $\mathbf{V}_1, \mathbf{V}_2$, with $m_{\mathbf{V}_1} > m_{\mathbf{V}_2}$, we have that $P_{\mathbf{V}_1}(a^* = w|v_1, v_2 \ldots v_n) > P_{\mathbf{V}_2}(a^* = w|v_1, v_2 \ldots v_n)$. Therefore, the team is more likely correct in profiles where a higher number of agents agree. □

Hence, if across multiple voting iterations, a higher number of agents agree often, we can predict that the team is going to be successful. If they disagree a lot, we can expect that they are wrong in most of the voting iterations, and we can predict that the team is going to fail.

In the next observation we show that we can increase the prediction accuracy by knowing not only how many agents agreed, but also which specific agents were involved in the agreement. Basically, we show that the probability of a team being correct depends on the agents involved in the agreement. Therefore, if we know that the best agents are involved in an agreement, we can be more certain of a team's success.

OBSERVATION 2. *Given two profiles $\boldsymbol{V}_1, \boldsymbol{V}_2$ with the same number of agreeing agents m, the probability that a team is correct is not necessarily equal for the two profiles.*

PROOF. We can easily show by example. Consider a problem with two actions. Consider a team of three agents, where t_1 and t_2 have a probability of 0.8 of being correct, while t_3 has a probability of 0.6 of being correct. We should always pick the action chosen by the majority of the agents, as the probability of picking the correct action is the highest for all agents [26]. Hence, plurality is MLE.

However, when only t_1 and t_2 agree, the probability that the team is correct is given by: $0.8^2 \times 0.4/(0.8^2 \times 0.4 + 0.2^2 \times 0.6) = 0.91$. When only t_2 and t_3 agree, the probability that the team is correct is given by: $0.8 \times 0.6 \times 0.2/(0.8 \times 0.6 \times 0.2 + 0.2 \times 0.4 \times 0.8) = 0.59$. Hence, the probability that the team is correct is higher when t_1 and t_2 agree than when t_2 and t_3 agree. □

However, based on the classical voting models, one would expect that given two different teams, the predictions would be more accurate for the one that has greater performance (i.e., likelihood of being correct), as we formalize in the following observation. We will use the term *strength* to refer to a team's performance.

OBSERVATION 3. *Under the classical voting models, given two different teams, one can expect to make better predictions for the strongest one.*

PROOF. Under the classical voting models, assuming the agents have a noise model such that plurality is a MLE, we have that the best team will have a greater probability of being correct given a voting profile where m agents agree than a worse team with the same amount of m agreeing agents.

Hence, the probability of the best team being correct will be closer to 1 in comparison with the probability of the worse team being correct. The closer the probability of success is to 1, the easier it is to make predictions. Consider a Bernoulli trial with probability of success $p \approx 1$. In the learning phase, we will see many successes accordingly. In the testing phase, we will predict the majority of the two for every trial, and we will go wrong only with probability $|1 - p| \approx 0$.

Of course, we could also have an extremely weak team, that is wrong most of the time. For such team, it would also be easy to predict that the probability of success is close to 0. Notice, however, that we are assuming here the classical voting models, where plurality is a MLE. In such models, the agents must play "reasonably well": classically they are assumed to have either a probability of being correct greater than 0.5 or the probability of voting for the best action is the highest one in their pdf [26]. Otherwise, plurality is not going to be a MLE. □

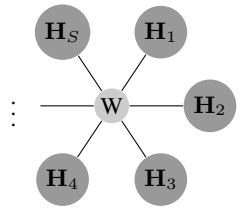


Figure 1: Graphical model of the problem solving process across a series of voting iterations.

Consider, however, that the strongest team is composed of copies of the best agent (which would often be the case, under the classical assumptions). We actually have that, in fact, such agents will not necessarily have noise models (pdfs) where the best action has the highest probability in all world states. In some world states, a suboptimal action could have the highest probability, making the agents agree on the same mistakes [29, 17]. Therefore, when plurality is not actually a MLE in all world states, we have that Observation 1 will not hold in the world states where this happens. Hence, we will predict that the team made a correct choice, when actually the team was wrong, causing problems in our accuracy. We give more details in the next section.

4.2 Main Theoretical Model

We now present our main theory, that holds irrespective of plurality being a MLE or not. Again, we consider agents voting across multiple world states. We assume that all iterations equally influence the final outcome, and that they are all independent.

Let the final reward of the team be defined by a random variable W, and let the number of world states be S. We model the problem solving process by the graphical model in Figure 1, where $\mathbf{H}_j$ is the instance of a random variable that represents the subset of agents that agreed on the chosen action at world state s_j.

For any subset $\mathbf{H}$, let $P(\mathbf{H})$ be the probability that the chosen action was correct given the subset of agreeing agents. If the correct action is a^*, $P(\mathbf{H})$ is equivalent to:

$$\frac{P(\forall t \in \mathbf{H}, t \text{ chooses } a^*, \quad \forall t \notin \mathbf{H}, t \text{ chooses } a \neq a^*)}{P(\exists a' \quad \forall t \in \mathbf{H}, t \text{ chooses } a', \quad \forall t \notin \mathbf{H}, t \text{ chooses } a \neq a')}$$

Note that $P(\mathbf{H})$ depends on both the team and the world state. However, we marginalize the probabilities to produce a value that is an average over all world states. We consider that, for a team to be successful, there exists a unique δ such that:

$$\left\{ \prod_{j=1}^{S} P(\mathbf{H}_j) \right\}^{1/S} > \delta \tag{1}$$

We use the exponent $1/S$ in order to maintain a uniform scale across all problems. Each problem might have a different number of world states; and for one with many world states, it is likely that the incurred product of probabilities is sufficiently low to fail the above test, independent of the actual subsets of agents that agreed upon the chosen actions. However, the final reward is not dependent on the number of world states.

We can show, then, that we can use a linear classification model (such as logistic regression) that is equivalent to Equation 1, to predict the final reward of a team.

THEOREM 1. *Given the model in Equation 1, the final outcome of a team can be predicted by a linear model (like the one described in Section 3).*

PROOF. Getting the log in both sides of Equation 1, we have:

$$\sum_{j=1}^{S} \frac{1}{S} \log(P(\mathbf{H}_j)) > \log(\delta)$$

The sum over the steps (world states) of the problem-solving process can be transformed to a sum over all possible subset of agents that can be encountered, $\mathcal{P}$:

$$\sum_{\mathbf{H} \in \mathcal{P}} \frac{n_{\mathbf{H}}}{S} \log(P(\mathbf{H})) > \log(\delta),$$

where $n_{\mathbf{H}}$ is the number of times the subset of agreeing agents $\mathbf{H}$ was encountered during problem solving. Hence, $\frac{n_{\mathbf{H}}}{S}$ is the frequency of seeing the subset $\mathbf{H}$, which we will denote by $f_{\mathbf{H}}$.

Recall that $\mathbf{T}$ is the set of all agents. Hence, $f_{\mathbf{T}}$ (which is the frequency of all agents agreeing on the same action), is equal to $1 - \sum_{\mathbf{H} \in \mathcal{P} \setminus \{\mathbf{T}\}} f_{\mathbf{H}}$. Also, note that $n_{\emptyset} = 0$, since at least one agent must pick the chosen action. The above equation can, hence, be rewritten as:

$$\log(P(\mathbf{T})) + \sum_{\mathbf{H} \in \mathcal{P} \setminus \mathbf{T}} f_{\mathbf{H}} \log\left(\frac{P(\mathbf{H})}{P(\mathbf{T})}\right) > \log(\delta)$$

Hence, our final model will be:

$$\sum_{\mathbf{H} \in \mathcal{P} \setminus \mathbf{T}} \log\left(\frac{P(\mathbf{H})}{P(\mathbf{T})}\right) f_{\mathbf{H}} > \log\left(\frac{\delta}{P(\mathbf{T})}\right) \quad (2)$$

Note that $\log(\frac{\delta}{P(\mathbf{T})})$ and the "coefficients" $\log(\frac{P(\mathbf{H})}{P(\mathbf{T})})$ are all *constants* with respect to a given team, as we have discussed earlier. Considering the set of all $f_{\mathbf{H}}$ (for each possible subset of agreeing agents $\mathbf{H}$) to be the characteristic features of a single problem, the coefficients can now be *learned* from training data that contains many problems represented using these features. Further, the outcome of a team can be estimated through a linear model. □

The number of constants is exponential, however, as the size of the team grows. Therefore, in the following corollary, we show that (under some conditions) we can approximate well the prediction with a reduced feature vector that grows linearly. In order to differentiate different possible subsets, we will denote by $\mathbf{H}^i$ a certain subset $\in \mathcal{P}$, and by $|\mathbf{H}^i|$ the size of that subset (i.e., the number of agents that agree on the chosen action).

COROLLARY 1. *If $P(\mathbf{H}^i) \approx P(\mathbf{H}^j)\ \forall \mathbf{H}^i, \mathbf{H}^j$ such that $|\mathbf{H}^i| = |\mathbf{H}^j|$, we can approximate the prediction with a reduced feature vector, that grows linearly with the number of agents. Furthermore, in a uniform team the reduced representation is equal to the full representation.*

PROOF. By the assumption of the corollary, there is a $P_{\mathbf{H}'^n}$, defined as $P_{\mathbf{H}'^n} \approx P(\mathbf{H}^j)$, $\forall \mathbf{H}^j$ such that $|\mathbf{H}^j| = n$. Let $f_n = \sum f_{\mathbf{H}^j}$, over all $|\mathbf{H}^j| = n$. Also, let $\mathbf{N}'$ be the set of all integers $0 < x < N$, where N is the number of agents. We thus have that:

$$\sum_{\mathbf{H} \in \mathcal{P} \setminus \mathbf{T}} f_{\mathbf{H}} \log\left(\frac{P(\mathbf{H})}{P(\mathbf{T})}\right) \approx \sum_{x \in \mathbf{N}'} f_x \log\left(\frac{P_{\mathbf{H}'^n}}{P(\mathbf{T})}\right) \quad (3)$$

As $P_{\mathbf{H}'^n}$ depends only on the number of agents, we have that such representation grows linearly with the size of the team.

Moreover, note that for a team made of copies of the same agent, we have that $P_{\mathbf{H}'^n} = P(\mathbf{H}^j)$, $\forall \mathbf{H}^j$ such that $|\mathbf{H}^j| = n$. Hence, the left hand side of Equation 3 is going to be equal to the right hand side. □

Also, notice that our model does not need any assumptions about plurality being a MLE. In fact, there are no assumptions about the voting rule at all. Hence, Observations 1, 2 and 3 do not apply, and we can still make accurate predictions irrespective of the performance of a team.

We can also note that the accuracy of the predictions is not going to be the same across different teams. Given two teams where plurality is the optimal voting rule in general (i.e., $P(\mathbf{H}^i) > P(\mathbf{H}^j)$, $\forall \mathbf{H}^i, \mathbf{H}^j$ where $|\mathbf{H}^i| > |\mathbf{H}^j|$): one *uniform*, made of copies of the best agent, and a *diverse*, made of different agents, we have that we can actually make better predictions for *diverse* than *uniform*, irrespective of the actual playing performance of these two teams.

COROLLARY 2. *Given a* diverse *and a* uniform *team, the accuracy of the prediction for the* diverse *team can be higher, even if the* diverse *team has a lower probability of victory.*

Proof Sketch. Note that the *learned* constants $\hat{c}_{\mathbf{H}} \approx \log(\frac{P(\mathbf{H})}{P(\mathbf{T})})$ represent the marginal probabilities across all world states. However, Marcolino et al. (2013) [29] and Jiang et al. (2014) [17] show that the agent with the highest marginal probability of voting for the correct choice will not necessarily have the highest probability of being correct at all world states.

Hence, let $\mathbf{Bad}$ be the set of world states where the best agent does not have the highest probability of voting for the correct action in its pdf. In such world states, the agents tend to agree over the incorrect action that has the highest probability. This follows from modeling the voting of all agents as a multinomial distribution. Hence, the expected number of agents that vote for an action a_j, in a team with n agents, is given by: $E[|\mathbf{H}|] = n \times p_j$, where p_j is the probability of the agent voting for action a_j. Therefore, the action with the highest probability will tend to receive the highest number of votes.

However, in our prediction model, the estimation that the team is correct will get higher as more agents agree together upon the final choice. Hence, we will tend to make wrong predictions for the world states in the set $\mathbf{Bad}$, and our accuracy will be lower as $|\mathbf{Bad}|$ gets higher.

Since a diverse team is composed of agents with different pdfs, it is less likely that in a given world state they will all have the highest probability in the same incorrect action [29, 17]. Hence, it is less likely that the situation described above happens, and we can have better predictions for a diverse team. □

Although we only give here a proof sketch, in the next section we experimentally show a statistically significant higher accuracy for the predictions for a *diverse* team than for a *uniform* team, even though they have similar strength (i.e., performance in terms of

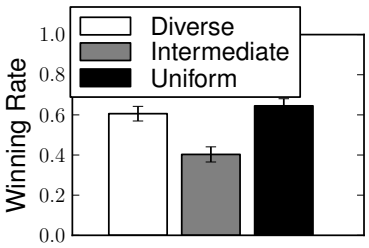


Figure 2: Winning rates of the three different teams used in our experiments.

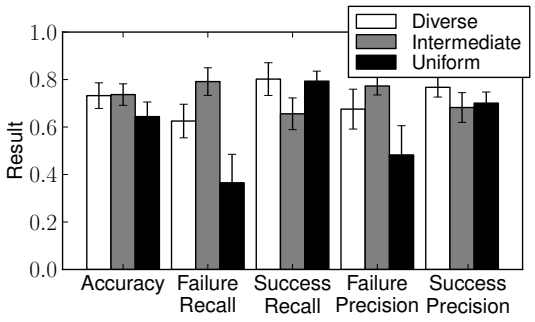


Figure 3: Performance when predicting in the end of games, using the full feature vector.

winning rates). We are able to achieve a better prediction for *diverse* teams both in the end of the problem-solving process and also while doing online predictions at any world state.

5. RESULTS

We test our prediction method in the Computer Go domain. We use four different Go software: Fuego 1.1 [9], GnuGo 3.8 [10], Pachi 9.01 [3], MoGo 4 [12], and two (weaker) variants of Fuego (FuegoΔ and FuegoΘ), in a total of six different, publicly available, agents. Fuego is considered the strongest agent among all of them. The description of FuegoΔ and FuegoΘ is available in [30].

We study three different teams: *Diverse*, composed of one copy of each agent; *Uniform*, composed of six copies of the original Fuego (initialized with different random seeds); *Intermediate*, composed of six random parametrized versions of Fuego (from [17]). As shown in [29, 30], Fuego is the strongest agent among them. In all teams, the agents vote together, playing as white, in a series of 9x9 Go games against the original Fuego playing as black.

In order to evaluate our predictions, we use a dataset of 691 games for each team. For all results, we used 5-fold cross validation (each fold had approximately the same class ratio as the original distribution). In all graphs, the error bars show the 95% confidence interval.

First, we show the winning rates of the teams in Figure 2. The difference between *uniform* and *diverse* is not statistically significant ($p = 0.1492$), and both teams are clearly significantly better than *intermediate* ($p < 6.3 \times 10^{-14}$).

We will now evaluate our prediction results according to five different metrics: Accuracy, Failure Precision, Success Precision, Failure Recall, Success Recall. Accuracy is defined as the sum of the true positives and true negatives, divided by the total number of tests (i.e., true positives, true negatives, false positives, false negatives). Hence, it gives an overall view of the quality of the classification. Precision gives the percentage of data points classified with a certain label ("success" or "failure") that are correctly labeled. Recall denotes the percentage of data points that truly pertain to a certain label, that are correctly classified.

We start by studying, in Figure 3, the quality of our prediction in the end of the games (i.e., after the last move). As we can see, we could make high-quality predictions for all teams. This is expected according to our Theorem 1, where we show that a linear classifier is able to predict well the performance of a team of voting agents.

For *diverse* and *intermediate*, we have around 73% accuracy, while for *uniform*, 64%. This difference is statistically significant, with $p \approx 0.003467$. In fact, the prediction in *diverse* is better than in *uniform* for all metrics: concerning failure precision, success precision and failure recall, the difference is statistically significant, with $p \approx 0.002361$, 3.821×10^{-6} and 3.821×10^{-6}, respectively. Note that this is expected, according to our Corollary 2.

It is also interesting to note that although *intermediate* is significantly weaker than *uniform*, we could achieve a higher accuracy for *intermediate* (with $p \approx 0.00379$). Hence, it is not the case that we can make better predictions for stronger teams (as it would be expected if we simply consider the classical voting models, such as in Observations 1, 2, and 3, instead of our main model).

The prediction accuracy for *diverse* and *intermediate* teams is actually very similar. However, by analyzing the other metrics, we can notice that the prediction for "failure" is better for *intermediate*, while the one for "success" is better for *diverse*.

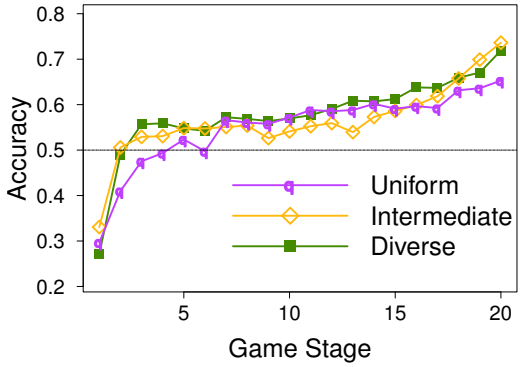
As we could see, with absolutely no data about which specific actions were made and which specific world states were encountered, we are able to predict the outcome of the games with high accuracy for all the three teams, with better results for *diverse* than *uniform*, even though these two teams have similar winning rates. Therefore, as we argue in our Theorem 1, only the frequencies of agreement among different subsets of agents is enough to have high-quality predictions over the final performance of teams.

Of course, a prediction made at the end of the problem solving process (in this case, a game) is not really useful anymore. Our real objective is to get high-quality predictions online, at any stage of the games. Therefore, to evaluate that, we also ran our classifier at every turn of the games.

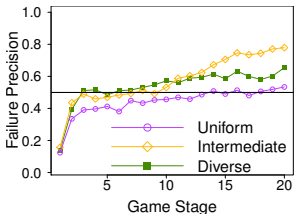
In order to verify the online predictions, we used the evaluation of the original Fuego, but we give it a time limit $50\times$ longer. Since this version is approximating a perfect evaluation of a board configuration, we will refer to it as "Perfect". We, then, use Perfect's evaluation of a given board state to estimate its probability of victory, allowing a comparison with our approach. Considering that an evaluation above 0.5 is "success" and below is "failure", we compare our predictions with the ones given by Perfect's evaluation, at each turn of the games.

We use this method because the likelihood of victory changes dynamically during a game. That is, a team could be in a winning position at a certain stage, after making several good moves, but suddenly change to a losing position after committing a mistake. Similarly, a team could be in a losing position after several good moves from the opponent, but suddenly change to a winning position after the opponent makes a mistake. Therefore, simply comparing with the final outcome of the game would not be a good evaluation. However, in the appendix (available at `http://teamcore.usc.edu/people/sorianom/a15-ap.pdf`), we also show how the evaluation would be comparing with the final outcome of the game, for the interested reader — and we note here that our online accuracy is still high in such alternative.

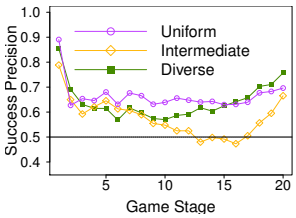
Since the games have different lengths, we divide all games in 20 stages, and show the average evaluation of each stage, in order



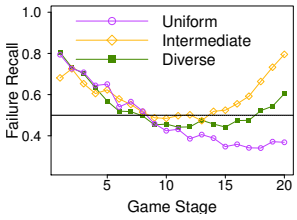
(a) Accuracy



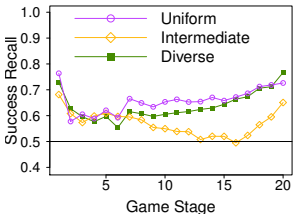
(b) Failure Precision



(c) Success Precision



(d) Failure Recall



(e) Success Recall

Figure 4: Performance metrics over all turns of 691 games, using the full feature vector.

to be able to compare the evaluation across all games uniformly. Therefore, a stage is defined as a small set of turns (on average, 2.43 ± 0.5 turns). Again, we used 5-fold cross validation, where each fold had approximately the same class ratio as the original distribution. We can see the result in Figure 4. We were able to obtain a high-accuracy very quickly, already crossing the 0.5 line in the 3^{rd} stage. In fact, the accuracy is significantly higher than the 0.5 mark (with $p < 0.005$ most of the time, and $p < 0.015$ always) for all teams from around the 5^{th} stage.

From around the middle of the games (stage 10), the accuracy for *diverse* and *uniform* already gets close to 60% (with *intermediate* only close behind). Although we can see some small drops – that could be explained by the sudden changes in the game –, overall the accuracy increases with the game stage number, as expected. Moreover, for most of the stages, the accuracy is higher for *diverse* than for *uniform*. The prediction for *diverse* is significantly better than for *uniform* (with $p < 0.1$) in 25% of the stages.

In order to show that we can also have high quality predictions with a much more scalable representation, we also run experiments using the reduced feature vector for all three teams. In Figure 5 we can see the results when predicting in the end of the games with the reduced feature vector. When comparing the results with the full representation, we notice that the accuracy does not change much for *diverse* and *intermediate*, and the difference is not significant ($p = 0.9929$ and $p = 0.8403$, respectively). For *uniform*

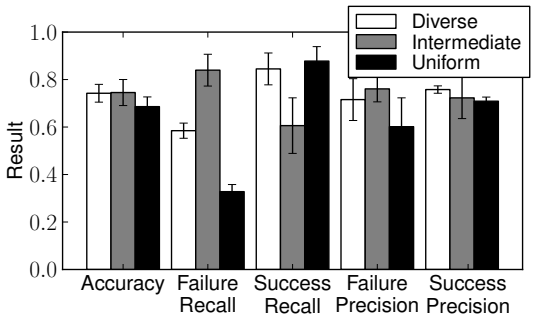


Figure 5: Performance when predicting in the end of games, using the reduced feature vector.

we observe an improvement in the accuracy of 4%, but such improvement is also not statistically significant ($p = 0.2867$). Hence, as expected from Corollary 1, the reduced representation approximates well the full one.

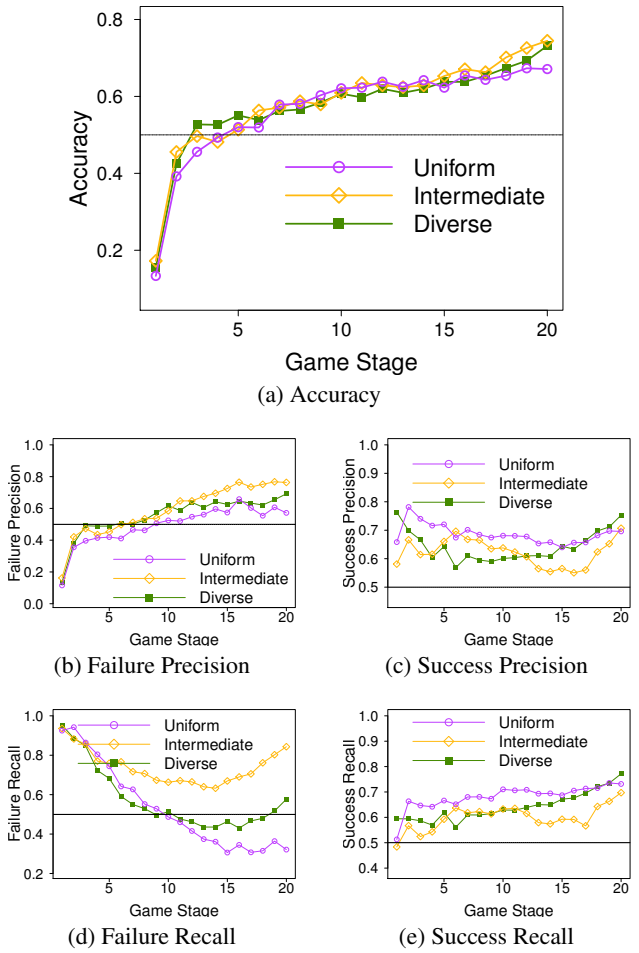
In Figure 6 we can see the prediction at each stage of the game, again comparing with Perfect's evaluation. As we can see, we can also obtain a high accuracy quickly with the reduced feature vector, reaching 60% again towards the middle of the games. This time, there is less difference in the accuracy for the *diverse* and *uniform* teams, but we can still show that the accuracy for *diverse* is significantly better than for *uniform* (with $p < 0.1$) in 15% of the stages (20% including a stage where $p \approx 0.1$). Note that, again, the accuracy for the *intermediate* team is close to the one for *uniform*, even though *intermediate* is a significantly weaker team.

As we can see, for all teams and both feature vectors, our predictions match Perfect's evaluation roughly 60% of the time. However, our method is much faster, since it only requires one linear calculation that takes a few microseconds, while Perfect's evaluation takes a few minutes. Therefore, we can easily use our method online, and dynamically take measures to improve the problem solving process when necessary. For the interested reader, we present all learned functions of these experiments in the appendix.

6. DISCUSSION

We show, both theoretically and experimentally, that we can make high-quality predictions about the performance of a team of voting agents, using only information about the frequency of agreements among them. We present two kinds of feature vectors, one that includes information about which specific agents were involved in an agreement and one that only uses information about how many agents agreed. Although the number of features in the former increases exponentially with the number of agents, causing scalability concerns, the latter representation scales much better, as it increases linearly. Theoretically, the full feature vector should have better results in general, but as we discuss in Corollary 1, the reduced representation approximates well the full one under some conditions. In particular, in our experiments both approaches achieved a similar high accuracy. Hence, for large teams we can safely use the reduced feature vector, avoiding scalability problems.

Moreover, in real applications we usually do not have extremely large teams of voting agents. Unless we have an "idealized" diverse team, the performance is expected to converge after a certain number of agents [17]. In Marcolino et al. (2013) [29] and Marcolino et al. (2014) [30], significant improvements are already obtained with only 6 agents, while Jiang et al. (2014) [17] show little improvement as teams grow larger than 15 agents. Therefore, even in



(a) Accuracy

(b) Failure Precision

(c) Success Precision

(d) Failure Recall

(e) Success Recall

Figure 6: Performance metrics over all turns of 691 games, using the reduced feature vector.

cases where Corollary 1 does not apply, and the reduced vector is not a good approximation, the scalability of the full feature vector might not be a real concern.

Based on classical voting theory, and in our Observations 1, 2 and 3, we would expect the predictions to work better for the *uniform* team, if it is composed of copies of the best agent. However, we present a more general model in Theorem 1, that does not really depend on plurality being a maximum likelihood estimator (MLE). In fact, we show in our experiments that the prediction works significantly better for the *diverse* team, and we explain this phenomenon in Corollary 2. Moreover, the prediction for *intermediate* works as well as for the other teams, even though it is significantly weaker. We would not expect this result, based on classical voting theory and Observations 1, 2 and 3. We can, however, expect such result based on our more general model in Theorem 1, as we show that the prediction does not really depend at all on plurality being a MLE. Hence, it can still work well in cases where the team is weaker, and the MLE assumption does not hold well.

Although we showed a great performance in prediction, we did not present in this paper what an operator should actually do as the prediction of failure goes high. Possible remedy procedures (and the relative qualities of each one of them) vary according to each domain, but here we discuss some possible situations.

For example, consider a complex problem being solved in a cluster of computers. We do not want to overly allocate resources, as that would be a waste of computational power that could be allocated to other tasks (or a waste of electrical energy, at the very least). However, we could increase the allocation of resources for solving the current problem when it becomes necessary, according to our prediction.

While playing a game, it is well known that the best strategy changes according to each specific opponent we are facing [11, 37, 27, 36, 2]. However, it is in general hard to know which player is our current antagonist. Therefore, we could start playing the game with the team that works the best against general opponents, and dynamically change the team as our prediction of failure goes high, trying to adapt to the current situation.

Moreover, it is known that the best voting rule depends on the noise model of the agents [7]. However, in general, such model is not known for existing agents [29]. Therefore, we could start by playing a game with a very general rule, such as plurality voting, and dynamically try different voting rules according to our current prediction. Note that although Observations 1, 2 and 3 refer to plurality voting, our general model presented in Theorem 1 does not really depend on the voting rule.

7. CONCLUSION

Voting is a widely applied domain independent technique, that has been used in a variety of domains, such as: machine learning, crowdsourcing, board games, forecasting systems, etc. In this paper, we present a novel method to predict the performance of a team of agents that vote together at every step of a complex problem. Our method does not use any domain knowledge and is based only on the frequencies of agreement among the agents.

We explain theoretically why our prediction works. First, we present an explanation based on classical voting theories, but such explanation fails to fully explain our experimental results. Hence, we also present a more general model, that is independent of the voting rule, and also independent of any optimality assumptions about the voting rule (i.e., the voting rule does not need to be a maximum likelihood estimator across all world states). Such model allows a deeper understanding of the prediction methodology, and a more complete comprehension of our experimental results.

We perform experiments in the Computer Go domain with three different teams, each having different levels of diversity and strength (i.e., performance). We could achieve a high accuracy for all teams, despite their differences. In particular, we show that, contrary to what would be expected based on classical voting models, our prediction is not better for stronger teams, and can actually work equally well for a very weak team. Moreover, we showed that we could achieve a higher accuracy for a diverse team than for a uniform team. All that could be expected based in our more general model.

Finally, we showed that the prediction works online at each step of the problem solving process, and matches often with an in-depth analysis (that takes orders of magnitude longer time). Hence, a system operator can use this to take remedy procedures if the team is not performing well, or incorporated within an automatic procedure to dynamically change the team and/or the voting rule. We discussed in detail how that could be executed in different domains.

Hence, this work is a significant step towards not only a deeper understanding of voting systems, but also towards a greater applicability of voting in a variety of domains, by combining the potential of voting in finding correct answers with the ability to access the performance of teams and the predictability of the final outcome.

Acknowledgments: This research was supported by MURI grant W911NF-11-1-0332, and by IUSSTF. The authors would like to thank Chao Zhang and Amulya Yadav for useful comments.

REFERENCES

[1] Y. Bachrach, T. Graepel, G. Kasneci, M. Kosinski, and J. Van Gael. Crowd IQ: aggregating opinions to boost performance. In *AAMAS*, pages 535–542, 2012.

[2] S. Bakkes, P. Spronck, and J. van den Herik. Opponent modelling for case-based adaptive game AI. *Entertainment Computing*, 1(1):27–37, 2009.

[3] P. Baudiš and J.-l. Gailly. Pachi: State of the Art Open Source Go Program. In *Advances in Computer Games 13*, Nov. 2011.

[4] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[5] N. Bulling, M. Dastani, and M. Knobbout. Monitoring norm violations in multi-agent systems. In *AAMAS*, 2013.

[6] I. Caragiannis, A. D. Procaccia, and N. Shah. When do noisy votes reveal the truth? In *EC*, pages 143–160, New York, NY, USA, 2013. ACM.

[7] V. Conitzer and T. Sandholm. Common voting rules as maximum likelihood estimators. In *UAI*, pages 145–152. Morgan Kaufmann Publishers, 2005.

[8] T. T. Doan, Y. Yao, N. Alechina, and B. Logan. Verifying heterogeneous multi-agent programs. In *AAMAS*, 2014.

[9] M. Enzenberger, M. Müller, B. Arneson, and R. Segal. Fuego - An open-source framework for board games and go engine based on Monte Carlo Tree Search. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):259 –270, Dec. 2010.

[10] Free Software Foundation. Gnugo. `http://www.gnu.org/software/gnugo/`, 2009.

[11] S. Ganzfried and T. Sandholm. Game theory-based opponent modeling in large imperfect-information games. In *AAMAS*, 2011.

[12] S. Gelly, Y. Wang, R. Munos, and O. Teytaud. Modification of UCT with patterns in Monte-Carlo Go. Technical report, Institut National de Recherche en Informatique et en Automatique, 2006.

[13] M. R. Genesereth, N. Love, and B. Pell. General game playing: Overview of the AAAI competition. *AI Magazine*, 26(2):62–72, 2005.

[14] J. Hunter, F. Raimondi, N. Rungta, and R. Stocker. A synergistic and extensible framework for multi-agent system verification. In *AAMAS*, 2013.

[15] M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, 2005.

[16] I. S. Isa, S. Omar, Z. Saad, N. Noor, and M. Osman. Weather forecasting using photovoltaic system and neural network. In *Proceedings of the Second International Conference on Computational Intelligence, Communication Systems and Networks*, CICSyN, pages 96–100, July 2010.

[17] A. X. Jiang, L. S. Marcolino, A. D. Procaccia, T. Sandholm, N. Shah, and M. Tambe. Diverse randomized agents vote to win. In *NIPS*, 2014.

[18] M. Kalech and G. A. Kaminka. On the design of coordination diagnosis algorithms for teams of situated agents. *Artificial Intelligence*, 171:491–513, 2007.

[19] M. Kalech and G. A. Kaminka. Coordination diagnostic algorithms for teams of situated agents: Scaling-up. *Computational Intelligence*, 27(3):393–421, 2011.

[20] G. A. Kaminka. *Coordination of Large-Scale Multiagent Systems*, chapter Handling Coordination Failures in Large-Scale Multi-Agent Systems. Springer, 2006.

[21] G. A. Kaminka and M. Tambe. What is wrong with us? Improving robustness through social diagnosis. In *AAAI*, 1998.

[22] E. Khalastchi, M. Kalech, and L. Rokach. A hybrid approach for fault detection in autonomous physical agents. In *AAMAS*, 2014.

[23] P. Kouvaros and A. Lomuscio. Automatic verification of parameterised interleaved multi-agent systems. In *AAMAS*, 2013.

[24] S. Legg. *Machine Super Intelligence*. PhD thesis, University of Lugano, 2008.

[25] M. Q. Lindner and N. Agmon. Effective, quantitative, obscured observation-based fault detection in multi-agent systems. In *AAMAS*, 2014.

[26] C. List and R. E. Goodin. Epistemic democracy: Generalizing the Condorcet Jury Theorem. *Journal of Political Philosophy*, 9:277–306, 2001.

[27] A. J. Lockett, C. L. Chen, and R. Miikkulainen. Evolving explicit opponent models in game playing. In *Proceedings of the 9th annual conference on genetic and evolutionary computation*, GECCO, 2007.

[28] A. Mao, A. D. Procaccia, and Y. Chen. Better Human Computation Through Principled Voting. In *AAAI*, 2013.

[29] L. S. Marcolino, A. X. Jiang, and M. Tambe. Multi-agent team formation: Diversity beats strength? In *IJCAI*, 2013.

[30] L. S. Marcolino, H. Xu, A. X. Jiang, M. Tambe, and E. Bowring. Give a hard problem to a diverse team: Exploring large action spaces. In *AAAI*, 2014.

[31] V. Mirchevska, M. Luštrek, A. Bežek, and M. Gams. Discovering strategic behaviour of multi-agent systems in adversary settings. *Computing and Informatics*, 2014.

[32] T. Obata, T. Sugiyama, K. Hoki, and T. Ito. Consultation algorithm for Computer Shogi: Move decisions by majority. In *Computer and Games'10*, volume 6515 of *Lecture Notes in Computer Science*, pages 156–165. Springer, 2011.

[33] R. Polikar. *Ensemble Machine Learning: Methods and Applications*, chapter Ensemble Learning. Springer, 2012.

[34] T. Raines, M. Tambe, and S. Marsella. Automated assistants to aid humans in understanding team behaviors. In *AGENTS*, 2000.

[35] F. Ramos and H. Ayanegui. Discovering tactical behavior patterns supported by topological structures in soccer-agent domains. In *AAMAS*, 2008.

[36] F. Schadd, S. Bakkes, and P. Spronck. Opponent modeling in real-time strategy games. In *Proceedings of the 2007 Simulation and AI in Games Conference*, GAMEON, pages 61–68, 2007.

[37] F. Southey, M. Bowling, B. Larson, C. Piccione, N. Burch, D. Billings, and C. Rayner. Bayes' bluff: Opponent modeling in poker. In *UAI*, 2005.

[38] D. Tarapore, A. L. Christensen, P. U. Lima, and J. Carneiro. Abnormality detection in multiagent systems inspired by the adaptive immune system. In *AAMAS*, 2013.

[39] L. Torrey and M. E. Taylor. Teaching on a budget: Agents advising agents in reinforcement learning. In *AAMAS*, 2013.

[40] C. Zhang and V. Lesser. Coordinating multi-agent reinforcement learning with limited communication. In *AAMAS*, 2013.