ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Selective Neural Network Ensembles in Reinforcement Learning

Stefan Faußer[1] and Friedhelm Schwenker[1]

University of Ulm - Institute of Neural Information Processing
89069 Ulm - Germany

Abstract. Ensemble models can achieve more accurate predictions than single learners. Selective ensembles further improve the predictions by selecting an informative subset of the full ensemble. We consider reinforcement learning ensembles, where the members are neural networks. In this context we study a new algorithm for ensemble subset selection in reinforcement learning scenarios. The aim of the proposed learning strategy is to minimize the Bellman errors of the collected states. In the empirical evaluation, two benchmark applications with large state spaces have been considered, namely SZ-Tetris and generalized maze. Here, our selective ensemble algorithm significantly outperforms other approaches.

1 Introduction

Whereas ensemble or committee-based models have been successfully applied to supervised [1, 2], semi-supervised and active learning [3, 4], in Reinforcement Learning (RL) not much research has been done to combine agents in a committee. One of the first attempts were done in [5], where fitted Q iteration used ensembles of regression trees. In [6], values learned from different RL algorithms were combined in joint decisions. However, Q-Learning single learners seemed to outperform committees in the more difficult problems. In another work [7, 8], it has been analytically shown that a committee with joint decisions, with estimated values from agents with function approximator (FA), can perform more rewarding decisions than a single agent. Each agent were trained by the same RL method. An RL committee benefits from the diversities on the value estimations, both from unstable value estimators and from large state spaces. Empirical evaluations with SZ-Tetris and generalized maze confirmed the analytical results [7]. In a selective ensemble, only a subset of a large ensemble are taken with the aim of performing better predictions. In [9], this were done both for classification and regression. Later on, unlabelled data were included to improve the performance of a selective ensemble [10]. Up to now, there were no attempts to combine selective ensemble learning with RL.

Our contribution in this paper is a novel method for subset selecting in a large ensemble of agents. The aim is to minimize the absolute differences between the estimated values and the Bellman equation of the collected states. With this selection, the combined value estimations are more consistent and, thus, more accurate. In case of an RL control problem, it will result in a higher total reward. Each of the agents does the value estimations with an FA, trained by an RL method. We empirically evaluate the method on generalized maze and on SZ-Tetris.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

2 Neural Network Ensembles

Given is a set $D = \{A_1, A_2, \ldots, A_M\}$ of M agents. Each agent is represented by an FA with weights θ_m and trained through some RL method, including Value Iteration, Temporal-Difference (TD) or Monte-Carlo (MC). In this work we focus on Multi-Layer Perceptrons (MLP) as FA. After the training of T iterations or episodes, the performance of these M agents is tested keeping their weights fixed. From now on this is called simulation phase, benchmark or testing in contrast to the training phase, where the agents learn and update their weights. In the simulation phase, the agents can act as a committee and perform joint decisions. The Joint Average policy is:

$$\pi^{AV}(s) = \underset{a \in \mathcal{A}(s)}{\arg\max} \Big[\sum_{m \in D} Q_{\theta_m}(s, a) \Big] \tag{1}$$

where s is the current observed state, $\mathcal{A}(s)$ is a discrete set of actions available in state s and $Q_{\theta_m}(s, a)$ is the estimated value of agent m. Further, the Joint Majority Voting policy is:

$$\pi^{VO}(s) = \underset{a \in \mathcal{A}(s)}{\arg\max} \Big[\sum_{m \in D} N_m(s, a) \Big] \tag{2}$$

where $N_m(s, a)$ is one if the agent m votes for action a in state s, else zero.

3 Selective Neural Network Ensembles

Out of the set D with agent indices of a large ensemble, we can select a subset $\tilde{D} \subset D$, $|\tilde{D}| = \tilde{M}$, by solving the following quadratic programming problem:

$$E(w) = \sum_{s \in \mathcal{S}} \hat{p}(s) \Big[\sum_{m \in D} w_m \sum_{a \in \mathcal{A}(s)} \alpha_m(s, a) e_m(s, a) \Big]^2 \tag{3}$$

under the constraints $\sum_{m \in D} w_m = 1$, $w_m \geq 0$, $w_m \leq \frac{1}{M}$, $\forall m$, where $\mathcal{S}$ is a discrete set of states, $\hat{p}(s)$ the probability to observe the state s, $\sum_{a \in \mathcal{A}(s)} \alpha_m(s, a) = 1, \forall s, \forall m$, $e_m(s, a)$ is a bounded real-valued error of agent m for state-action pair (s, a) and w are Lagrange multipliers. Set $\tilde{D}$ gets the indices of the agents with the $\tilde{M}$ highest weights $w_m > 0$. This forms the selective ensemble. As the true values $Q(s, a)$, and thus the true errors $\hat{e}_m(s, a) = Q(s, a) - Q_{\theta_m}(s, a)$, are naturally unknown in an RL problem, we instead consider Bellman errors:

$$\tilde{e}_m(s, a) = \sum_{s' \in \mathcal{S}} p(s, a, s') \Big[r(s, a, s') + \sum_{a' \in \mathcal{A}(s')} \pi_m(s', a') \gamma Q_{\theta_m}(s', a') \Big] - Q_{\theta_m}(s, a) \tag{4}$$

where $r(s, a, s')$ is the immediate reward for taking action a in state s and observing the next state s', $\pi_m(s', a')$ is the probability to select action a' in state s' and $0 < \gamma \leq 1$ is the discounting factor used for learning the Q-values.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Algorithm 1 Selective Neural Network Ensembles

Input: Set D with M indices of trained agents, selective ensemble size $\tilde{M} < M$, state collection parameter ϵ_{col}, maximum number of ranked actions n_A, error threshold θ_{err}, problem with Markov property, discounting value γ, set ST with starting states, testing iterations $\tilde{T}$, prediction: fixed policy π, control: Joint Average (1) or Joint Majority Voting (2) policy π for exploiting states

Output: benchmark value or total reward

1: Collect states and state probabilities $\hat{p}(s)$: Perform simulation in given environment with committee in set D and policy π, collect observed state if random value $\in [0, 1)$ is smaller than ϵ_{col}

2: Calculate the errors of agent m for the collected states using (4), $\forall m$.

3: Get set $\tilde{D} \subset D$: Perform selective ensemble learning with agents in set D by minimizing (3), using errors from last step, θ_{err}, n_A and state probabilities $\hat{p}(s)$. Error weights are (5) (Voting, or prediction) or (6) (Average). Choose $\tilde{M}$ indices with highest Lagrange multipliers w_m.

4: Get benchmark value: Perform simulation with committee in set $\tilde{D}$, starting states ST, $\tilde{T}$ iterations and policy π

In case the RL model is not known, the $p(\cdot)$, $r(\cdot)$ and, if necessary, $\pi_m(\cdot)$ values can be estimated by sampling state-action pairs from the environment. Further, we consider hard errors:

$$e_m(s,a) = \begin{cases} 1, & \text{if } |\tilde{e}_m(s,a)| > \theta_{err} \\ 0, & \text{else} \end{cases}$$

where θ_{err} is the error threshold parameter. The error weights $\alpha_m(s,a)$ in (3) depend on the desired joint decisions, for a Joint Majority Voting policy it is:

$$\alpha_m^V(s,a) = \begin{cases} 1, & \text{if } a = \arg\max_b \pi_m(s,b) \\ 0, & \text{else} \end{cases} \tag{5}$$

and for a Joint Average policy it is:

$$\alpha_m^A(s,a) = \frac{c_m(s,a)}{\sum_{b \in \mathcal{A}(s)} c_m(s,b)} \tag{6}$$

with

$$c_m(s,a) = \begin{cases} 0, & \text{if } \operatorname{rank}_a[Q_{\theta_m}(s,\cdot)] \leq |\mathcal{A}(s)| - min(n_A, |\mathcal{A}(s)|) \\ \operatorname{rank}_a[Q_{\theta_m}(s,\cdot)], & \text{else} \end{cases}$$

where n_A is the maximum number of ranked actions and $\operatorname{rank}_a[Q_{\theta_m}(s,\cdot)]$ is the rank of action a in state s for agent m. With (5), only the errors of the agents with the best actions, and with (6), the errors of the agents with up to n_A best actions have a non-zero weight. The reason is that in (1), only a single action

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

per agent, and in (2), all actions of all agents contribute to the joint decisions. By minimizing cost function (4), agents are selected whose value estimations are more consistent. A state, observed by a committee, is collected with a probability of ϵ_{col}. The complete procedure is described in Algorithm (1).

4 Experiments

We evaluated the methods on two applications with large state spaces. In both applications, we used a 2-layer MLP as FA with $tanh(\cdot)$ in the hidden layer, and the linear function in the output layer, as activation functions. For each application, we trained 100 agents in parallel with randomly initialized weights. A committee ($M > 1$) either consists of randomly selected M agents out of the 100 without replacement, or performed selective ensemble learning with 50, randomly drawn from the 100, to select the best agents (M is 'Sl.' in result tables). 50 runs per committee. For the selective ensemble learning, the state-action pairs were collected with randomly chosen 50 agents and with $\epsilon_{col} = 1.0$ (generalized maze) or $\epsilon_{col} = 0.3$ (SZ-Tetris).

4.1 Generalized Maze

In the generalized maze problem, an agent tries to find the shortest path in a maze with some barriers and one goal. We consider a 5×5 maze with $3 - 5$ randomly placed barriers. Out of 1000 generated unique mazes, 900 are kept fixed as a training set and the rest as a validation set. In the training phase, a maze of the training set is randomly selected for each episode. Barriers and the goal are terminal states. The starting set ST includes all non-terminal states of the selected maze. The agent is not allowed to leave the maze. State-action values are learned with RG-SARSA (on-policy updates). With a probability of 0.3, the agent moves one state further to the north, if not blocked by a barrier (up-wind). The agent gets a reward of 1 if state s' is the goal or 0 if s' is a barrier or non-terminal state. The discounting value is set to $\gamma = 0.9$. In the simulation phase, all valid starting positions of each maze in the validation set are evaluated and the total reward is calculated. It is repeated 10 times per starting position. We roughly guess that the total reward, averaged over all mazes, is at ~ 15. The MLP has 5 neurons in the hidden layer and 150 input neurons, where 25 are for the barriers, 25 for the goal position and 25 per action for the agent position. Learning rates are optimized for a single agent, $\alpha = 0.01$. Results for a single agent ($M = 1$), and the committees ($M > 1$) with the softmax action selection strategy with $\tau = 0.08$ are in table 1. Evaluated is the total reward, measured in a benchmark setting after $2.5 \cdot 10^6$ to $25 \cdot 10^6$ training iterations. The second column denotes the policy used in the benchmark (S = Single, A = Average and V = Majority Voting). Selective ensemble sizes (from 2.5M to 25M), Average: 20, 30, 15, 15, Voting: 25, 15, 30, 30. The differences between the results of the selective ensemble and the large ensemble with $M = 30$ are statistically significant ($p < 0.02$). The selective ensembles are better than the large ensembles, both in terms of the total reward and in the ensemble size.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

M	π	reward 2.5M	reward 5M	reward 15M	reward 25M
1	S	12.10 ± 0.81	13.36 ± 0.37	13.80 ± 0.34	13.87 ± 0.32
5	A	13.62 ± 0.33	14.05 ± 0.33	14.39 ± 0.27	14.44 ± 0.23
30	A	14.17 ± 0.12	14.37 ± 0.11	14.59 ± 0.05	14.67 ± 0.06
50	A	14.24 ± 0.09	14.34 ± 0.08	14.61 ± 0.07	14.69 ± 0.04
Sl.	A	14.29 ± 0.08	14.47 ± 0.06	14.72 ± 0.06	14.79 ± 0.06
5	V	13.73 ± 0.29	14.31 ± 0.15	14.59 ± 0.10	14.62 ± 0.08
30	V	14.28 ± 0.08	14.56 ± 0.08	14.73 ± 0.06	14.85 ± 0.05
50	V	14.33 ± 0.09	14.56 ± 0.06	14.78 ± 0.05	14.85 ± 0.04
Sl.	V	14.37 ± 0.06	14.68 ± 0.07	14.84 ± 0.04	14.90 ± 0.03

Table 1: Empirical results with Generalized Maze.

4.2 SZ-Tetris

In stochastic SZ-Tetris, an agent places tetrominos on top of already placed tetrominos and tries to complete rows. The board has a width of 10 and a height of 20. S-shaped or Z-shaped pieces randomly appear with equal probabilities. The agent chooses a rotation and horizontal position of the piece within the board. If the piece does not fit on the board, then the game ends. SZ-Tetris with its two tetrominos, out of the seven available, is considered to be the hard core of Tetris [11]. In the benchmark-type setting [12], the agent gets one point per completed row. Quality criterion is the sum of completed rows until the board is full. The best learning agent reached a score of 133 [12]. Only states with a full board are terminal states. If rows are completed, then they are immediately removed. The starting set ST only includes the empty board. (Half-) state values are learned with $TD(\lambda = 0.5)$. On each step, the agent receives a reward of $r(s) = exp(-\beta \cdot$number of holes in s$)$, with $\beta = 1/33$ and counting the number of holes below the pieces (column-wise) in a state s. The discounting value is set to $\gamma = 0.95$. The MLP has 5 neurons in the hidden layer and 330 input neurons, whereas 180 are the discretized height differences and 150 are the discretized number of holes (more than 150 holes are treated like 150 holes). Learning rates are optimized for a single agent, $\alpha = 0.001$. The results with the ϵ-greedy exploration strategy with $\epsilon = 0.04$ are in table 2. Evaluated is the number of cleared lines, measured in a benchmark setting after $0.5 \cdot 10^6$ to $5 \cdot 10^6$ training iterations. The scores are averaged over 100 repeated tests (games). Selective ensemble sizes (from 0.5M to 5M), Average: 20, 40, 15, 5, Voting: 30, 35, 25, 25. The differences between the results of the selective ensemble and the large ensemble with $M = 30$ are statistically significant ($p < 0.02$). The selective ensembles show better results than the large ensembles, both in terms of the total reward and in the ensemble size.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

M	π	score 0.5M	score 1M	score 3M	score 5M
1	S	88.40 ± 22.1	108.6 ± 19.7	125.8 ± 13.9	129.7 ± 11.4
5	A	127.6 ± 12.3	135.2 ± 8.50	144.1 ± 6.60	147.0 ± 6.60
30	A	135.9 ± 7.20	150.2 ± 7.10	146.2 ± 4.40	149.6 ± 5.50
50	A	139.1 ± 7.00	151.2 ± 5.50	145.1 ± 3.20	149.7 ± 3.50
Sl.	A	142.0 ± 6.60	154.1 ± 7.10	153.0 ± 6.60	152.4 ± 6.20
5	V	109.9 ± 12.7	124.6 ± 10.9	137.8 ± 8.80	140.8 ± 7.00
30	V	130.8 ± 7.00	141.2 ± 6.20	144.1 ± 4.70	148.3 ± 5.00
50	V	132.5 ± 5.90	142.5 ± 6.50	144.4 ± 4.40	150.8 ± 4.00
Sl.	V	137.7 ± 8.20	145.6 ± 7.00	152.3 ± 6.20	152.4 ± 5.20

Table 2: Empirical results with SZ-Tetris.

5 Conclusions

We described a method to select an informative subset of agents from a large ensemble with the aim to achieve more accurate value estimations. To our knowledge this is the first attempt being made to combine selective ensemble learning and RL. The algorithm generally applies to a wide range of RL problems, including environments with large state spaces, but with the restriction that either the RL model is known, or the RL model parameters can be estimated by sampling state-action pairs from the environment. In the empirical evaluation, the selective ensembles significantly outperformed the full ensembles.

References

[1] L. Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, 1996.

[2] D. L. Shrestha and D. P. Solomatine. Experiments with AdaBoost.RT, an Improved Boosting Scheme for Regression. *Neural Computation*, 18(7):1678–1710, 2006.

[3] M. Farouk Abdel Hady and F. Schwenker. Combining Committee-Based Semi-Supervised Learning and Active Learning. *J. Comput. Sci. Technol.*, 25(4):681–698, 2010.

[4] M.-L. Zhang and Z.-H. Zhou. Exploiting unlabeled data to enhance ensemble diversity. *Data Min. Knowl. Discov.*, 26(1):98–129, 2013.

[5] D. Ernst, P. Geurts, and L. Wehenkel. Tree-Based Batch Mode Reinforcement Learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[6] M. A. Wiering and H. van Hasselt. Ensemble Algorithms in Reinforcement Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38(4):930–936, 2008.

[7] S. Faußer and F. Schwenker. Neural Network Ensembles in Reinforcement Learning. *Neural Processing Letters*, 2013.

[8] S. Faußer and F. Schwenker. Ensemble Methods for Reinforcement Learning with Function Approximation. In *MCS*, volume 6713 of *LNCS*, pages 56–65. Springer, 2011.

[9] Z. H. Zhou, J. Wu, and W. Tang. Ensembling Neural Networks: Many Could Be Better Than All. *Artificial Intelligence*, 137(1-2):239–263, 2002.

[10] N. Li and Z.-H. Zhou. Selective Ensemble under Regularization Framework. In *MCS*, volume 5519 of *Lecture Notes in Computer Science*, pages 293–303. Springer, 2009.

[11] H. Burgiel. How to Lose at Tetris. *The Mathematical Gazette*, 81(491):194–200, 1997.

[12] I. Szita and C. Szepesvári. SZ-Tetris as a Benchmark for Studying Key Problems of Reinforcement Learning. In *ICML Workshop on Machine Learning and Games*, 2010.